

MATLAB EXPO 2018

What's New in MATLAB and Simulink

Emelie Andersson, *Application Engineer*
Magnus Jung, *Application Engineer*

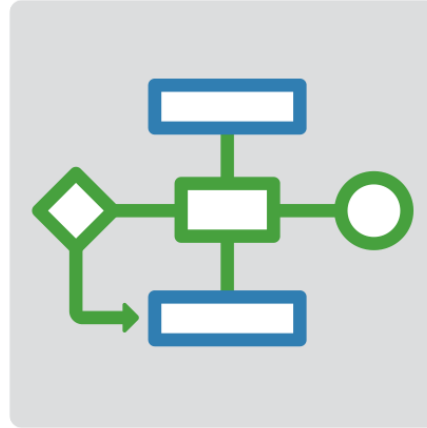


Platform Productivity



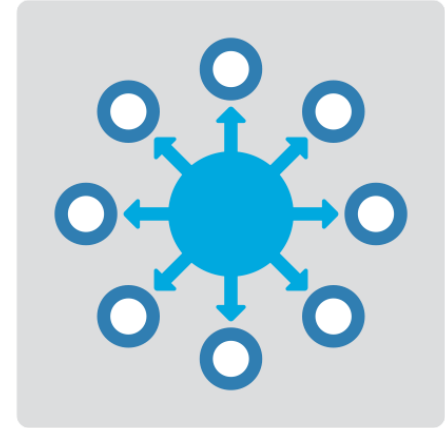
**Getting your work
done faster**

Workflow Depth



**Support for your
entire workflow**

Application Breadth

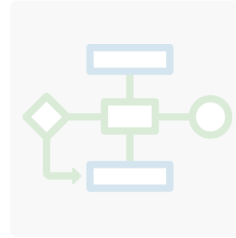


**Products for the
work you do**

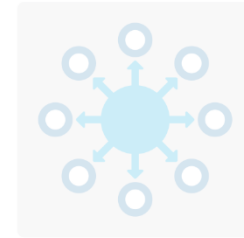
Platform Productivity



Workflow Depth



Application Breadth



- **Create Your Designs Faster**
- **Simplify Analysis**
- **Execute Faster and Scale Your Work**
- **Collaborate**

Create Your Designs Faster

The screenshot shows the MATLAB Live Editor interface. The document title is "Explore and Analyze Storm Events". The content is organized into sections:

- Frequency of Events**: Explains the goal of exploring storm event frequency and damage costs. It includes a code block:


```
clear
load prepEvents
data = timetable2table(data);
head(data)
```
- Visualize with a Heatmap**: Explains that heatmaps are useful for exploring patterns across categories. It includes a code block:


```
bigFigure;
heatmap(data,'state','weathercats');
xlabel('State')
ylabel('Storm Event')
title('Frequency of Events by Location')
```

On the right side of the editor, there is a variable viewer showing an 8x18 table of data with columns for "Time" and "Storm Event". Below the table is a heatmap visualization of the data, with "Storm Event" on the y-axis and "State" on the x-axis. The y-axis categories include: Avalanche, Blizzard, Coastal Weather, Debris Flow, Dense Fog, Drought, Dust Devil, Dust Storm, Excessive Heat, Flood, Freezing Fog, Frost/Freeze, Funnel Cloud, Hail, Heat, Heavy Rain, Hurricane, Ice Storm, Lightning, Sandstorm, Snow, Thunderstorm Wind, Tornado, and Waterspout.

- Embedded results – avoid context switching
- Contextual hints - both for variables and available options for functions
- Interactive MATLAB code generation
- Tell a story – title, images, hyperlinks

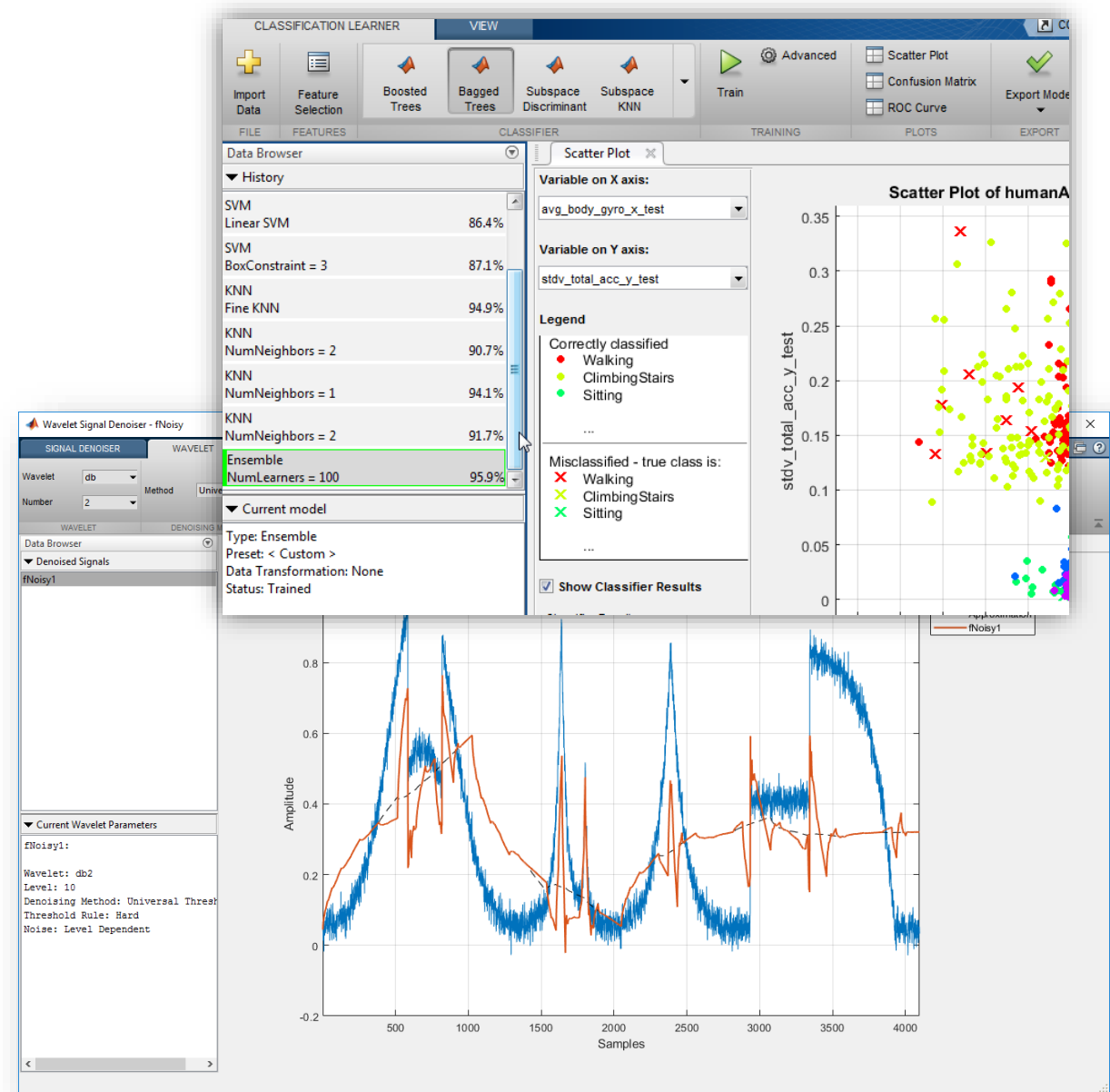
MATLAB
Live Editor

Create Your Designs Faster

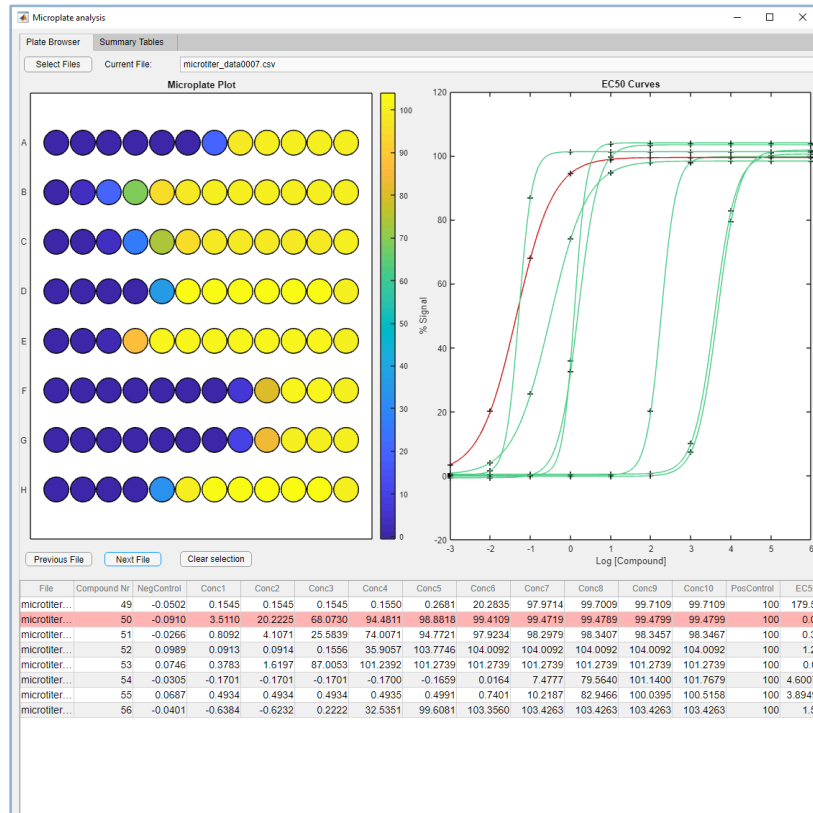
- Simplify Analysis with Apps

Interactive tools to speed up prototyping

- For signal data, machine learning, image labeling, and much more
- New Apps:
 - Econometric Modeler app
 - Analog Input Recorder app
 - Wavelet Signal Denoiser app



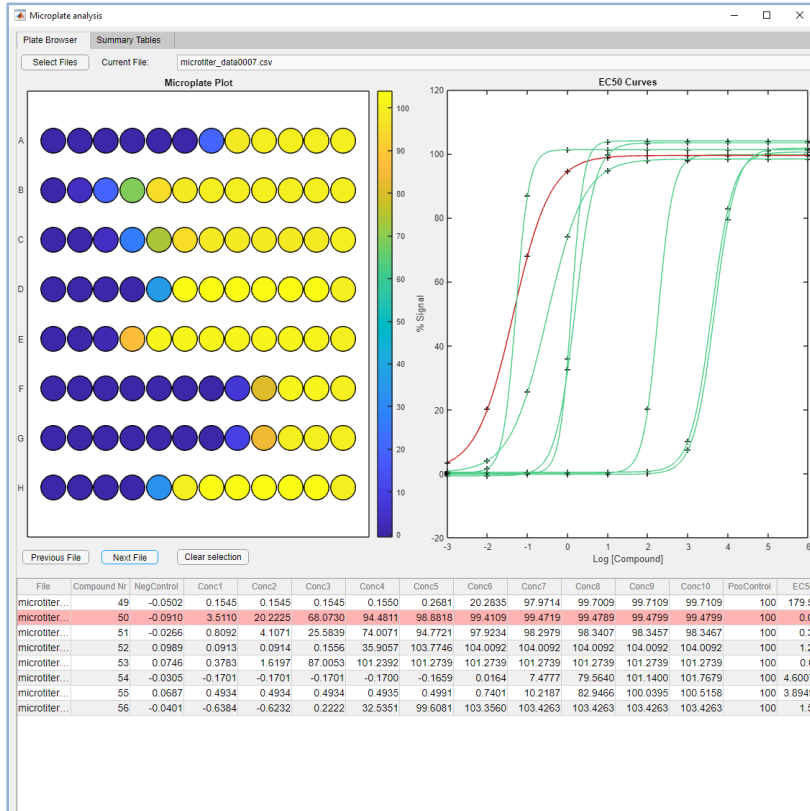
Create Your Designs Faster



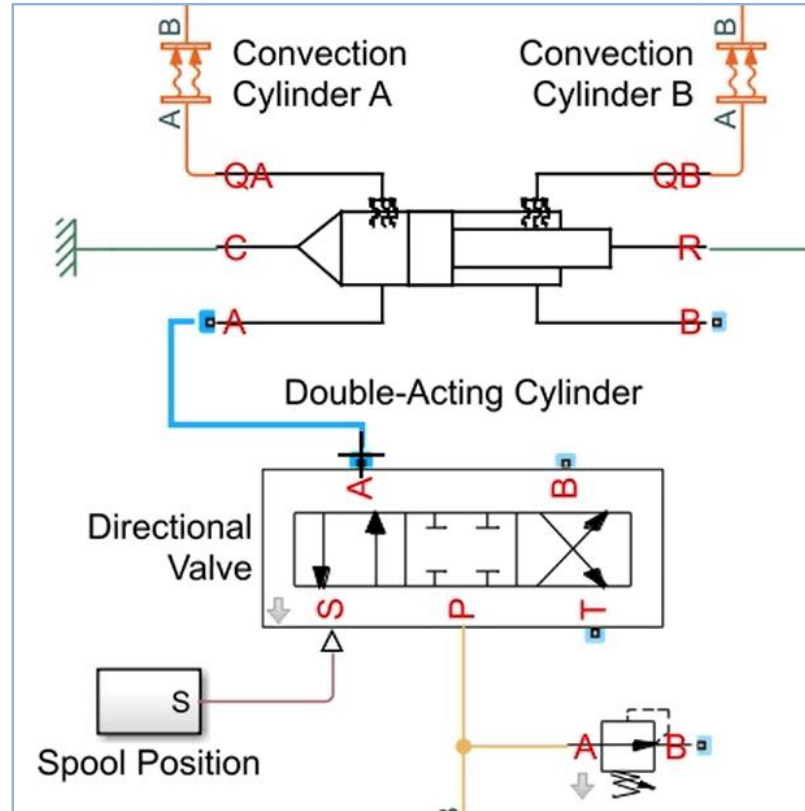
MATLAB
App Designer

The screenshot shows the MathWorks File Exchange interface. The main heading is 'File Exchange'. Below it, there is a thumbnail image of the migration tool's GUI. The title of the tool is 'GUIDE to App Designer Migration Tool for MATLAB', version 1.0 (15.1 KB) by the MathWorks App Designer Team. The description states: 'Use the GUIDE to App Designer Migration tool to help transition your GUIDE apps to App Designer.'

Create Your Designs Faster

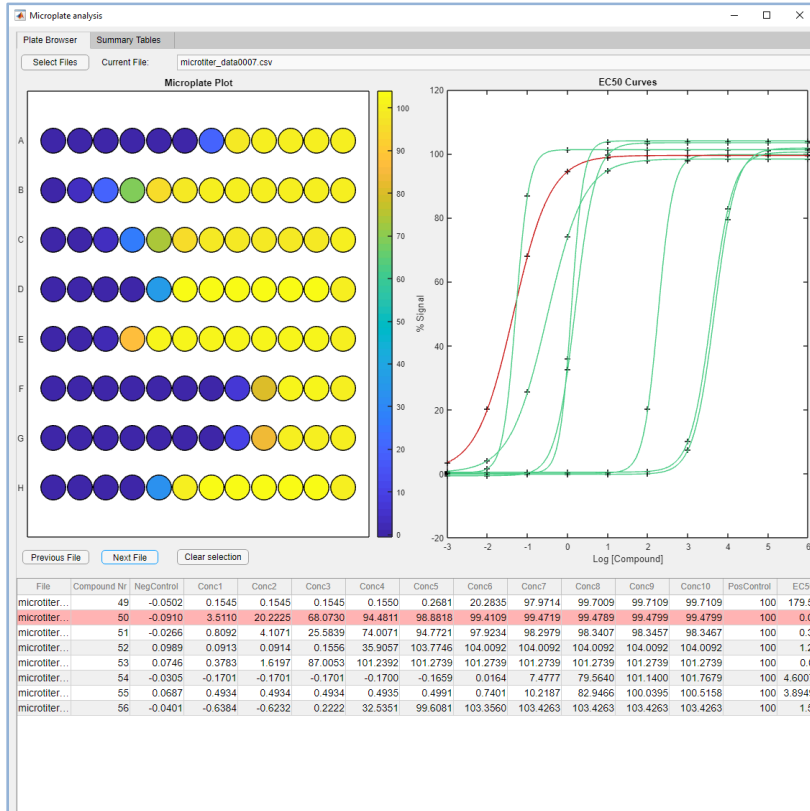


MATLAB

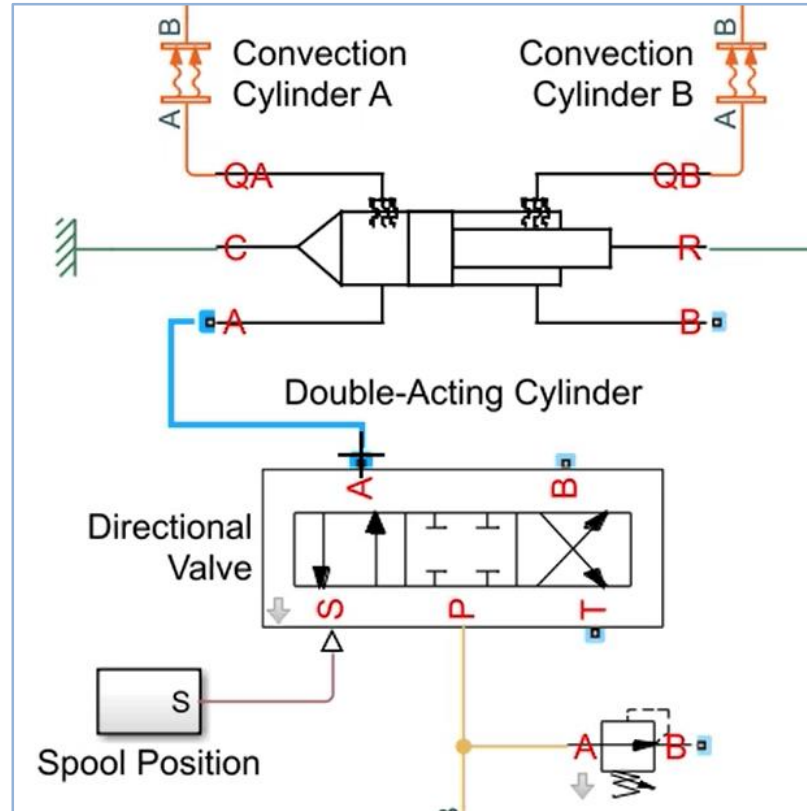


Simulink

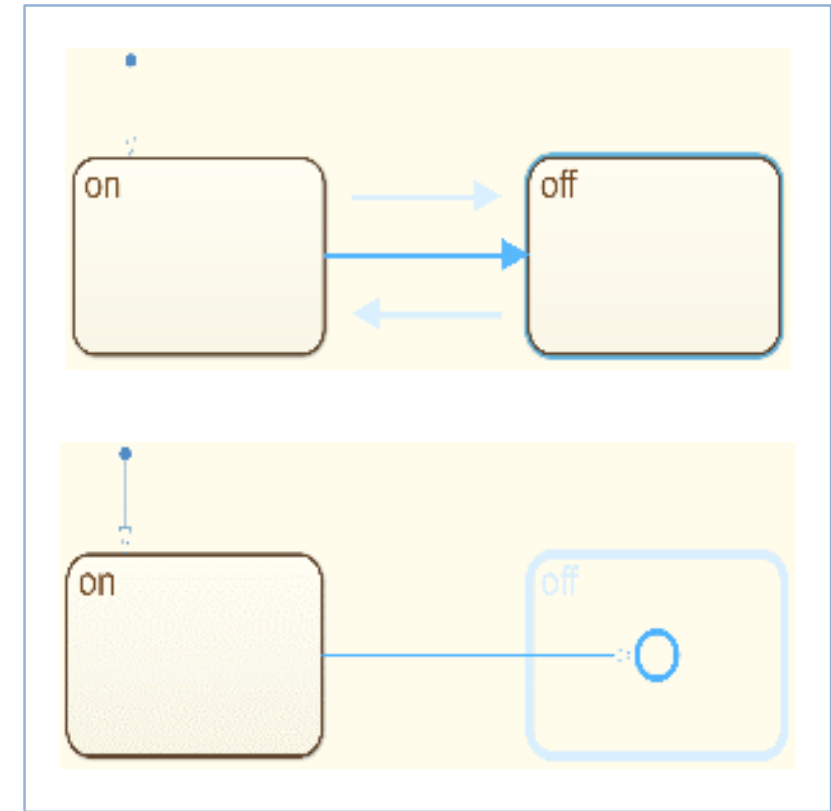
Create Your Designs Faster



MATLAB



Simulink

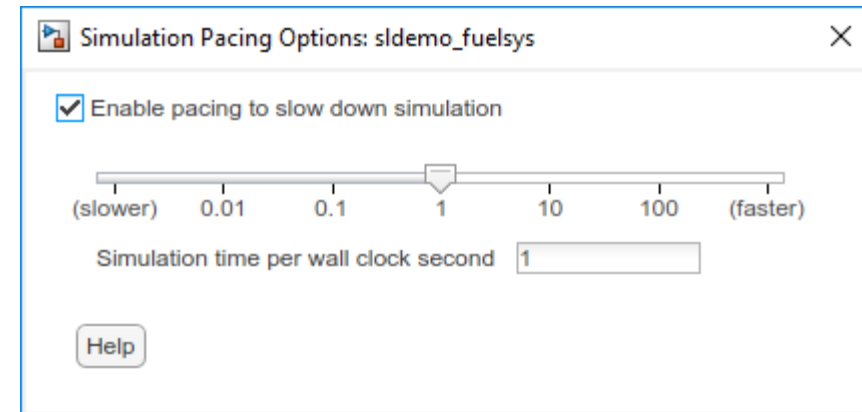


Stateflow

Simplify Analysis by Simulating at Wall Clock Speed

Slow down the simulation for easier model interactivity

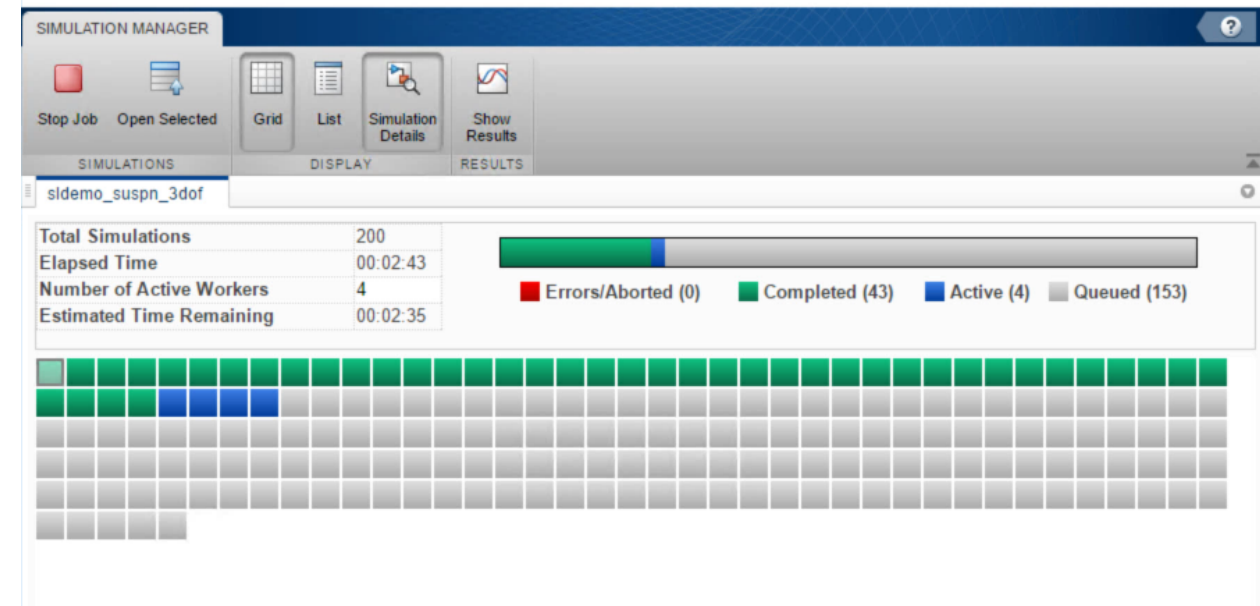
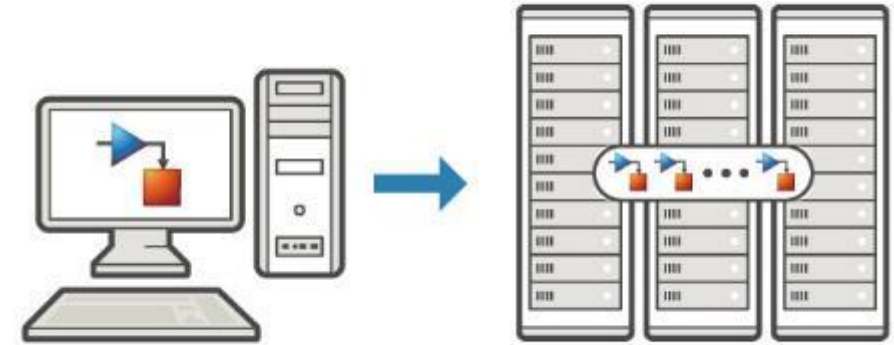
- Especially for models controlled and monitored via Dashboard blocks and other displays
- Useful when model is connected to hardware



Scale Your Work

Use parallel computing to run multiple simulations faster

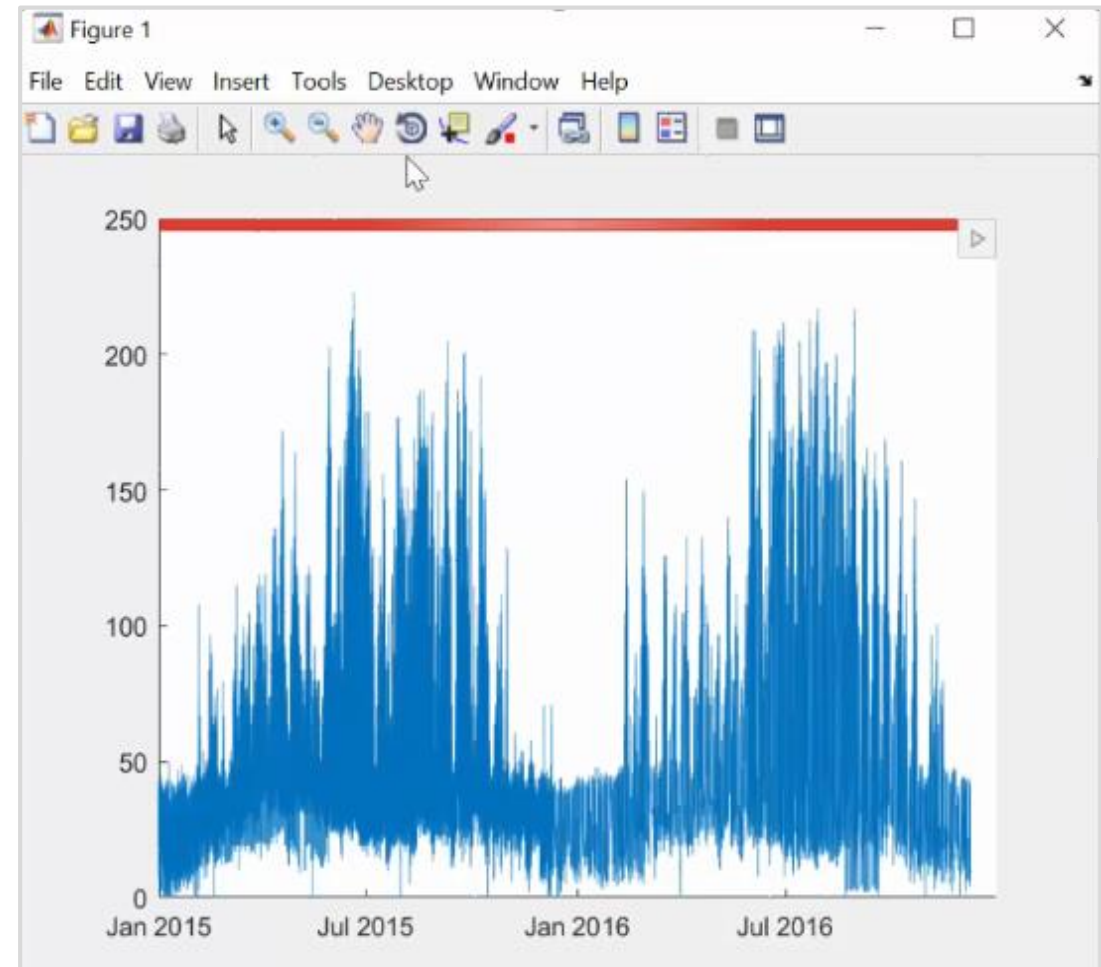
- Run multiple parallel simulations with `parsim`
- Monitor simulation status and progress in the Simulation Manager



Scale Your Work

Use tall arrays to manipulate and analyze data that is too big to fit in memory

- Use familiar MATLAB functions and syntax
- Support for hundreds of functions
- Works with Spark + Hadoop clusters

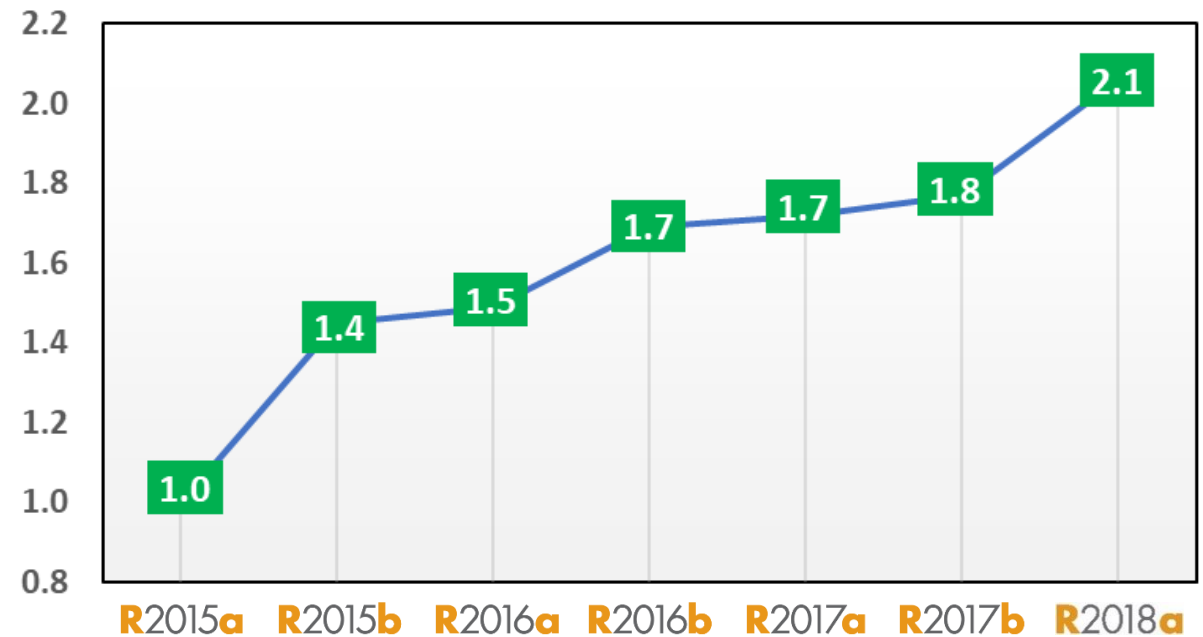


Execute Faster

Redesigned execution engine runs MATLAB code faster

- All MATLAB code can now be JIT compiled
- MATLAB runs your code over twice as fast as it did just three years ago
- No need to change a single line of your code
- Increased speed of MATLAB startup in R2018a

Average Speedup in Customer Workflows



Team Collaboration

Diff and Merge to support team collaboration

The screenshot displays the 'Three-Way Merge' interface for a Simulink project. The top toolbar includes navigation (Previous, Next), highlighting (Top Model, Bottom Model), and finishing (Filter, Accept & Close) options. The main area is divided into three columns: 'Theirs' (340c64c37beb096a316e58a11358a8387d026b5f), 'Base' (e317566e2ad5f02f38f648e8e7d08716367a0fac), and 'Mine' (mine_slproject_f14.slx). Each column shows a hierarchical tree of the Simulink model, including blocks like Pilot, PilotGain, and Solver. A conflict is visible in the 'Theirs' column where the 'PilotGain' block is highlighted in purple. Below the columns is a 'Target' section (targetFile.slx) with a red exclamation mark icon. On the right, a 'Resolve remaining 1 changes' dialog box is open, showing a summary table:

Filtered View (1)		All Changes (1)	
TYPE		UNRESOLVED	RESOLVED
Conflict	1	1	0
Conflicted manual merge	0	0	0
Manual merge	0	0	0
Automatic	0	0	4
Total	1	1	4

Upgrade your MATLAB Code and Simulink Models

Web Browser - (3 Errors) Code Compatibility Report

(3 Errors) Code Compatibility Report

Code Compatibility Report [Top](#) [3 Errors](#) [1 Warning](#) [304 Checks](#) [2 Files](#)

Analysis Date: 05-Sep-2017 14:32:08
MATLAB Version: R2017b

Incompatibility and Syntax Errors

Row	Filename	Line	Description
1	classifyBloodPressure.m	18	TREEFIT has been removed
2	classifyBloodPressure.m	21	TREEDISP has been removed. Use VIEW methods instead.
3	classifyBloodPressure.m	24	TREEVAL has been removed. Use PREDICT methods instead.

Warnings and Other Recommendations

Row	Filename	Line	Description
1	classifyBloodPressure.m	Z	RAND or RANDN with true random number generation is not recommended. Use RAND or RANDN with the 'true' option.

Upgrade Advisor - sf_climate_control

File Edit Run Settings Help

Find:

Upgrade Project Report

100% Passed

	Models	Libraries	MATLAB Code
Passed	7	1	8
Need attention	-	-	-

Show:

Filename	Check Name	Result
AnalogControl.mdl	Check model settings for migration to simplified initialization mode	Passed
analyzeModelFiles.m	Check that the model is saved in SLX format	Passed with fixes
billOfMaterials.m	Check usage of function-call connections	Need attention
checkCodeProblems.m	Check and set embedded target model to use ert.tlc system target file	Passed
DigitalControl.slx	Check and update masked blocks in library to use promoted parameters	Passed
f14_airframe.slx	Check and update mask image display commands with unnecessary imread() function calls	Passed
f14_airframe_test.m	Check and update mask to affirm icon drawing commands dependency on mask workspace	Passed
find_top_models.m	Check and update model to use toolchain approach to build generated code	Passed
LinearActuator.slx		
NonLinearActuator.mdl		
rebuild_s_functions.m		
runUnitTest.m		
slproject_f14.slx		
upgrade_project.m		
vertical_channel.slx		
wind_gust_lib.slx		

Check model settings for migration to simplified initialization mode [Learn more](#)

Check for model level messages
This check finds and reports model level messages for migrating to simplified initialization mode.

See Also

- Check model settings for migration to simplified initialization mode
- Underspecified initialization detection

Checks run on 02/01/2018 10:44

Identify Variant Model blocks and convert those to Vari

Analysis

Upgrade Variant Model blocks to Variant Subsystems contain offers enhanced capabilities while maintaining equivalent fun variant models will be removed in a future release.

Result: ✔ Passed

Identify Variant Model blocks at model level.

Passed
No Variant Model blocks found.

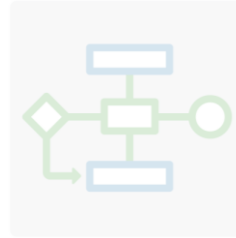
MATLAB EXPO 2018

14

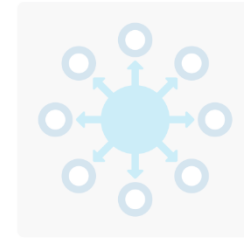
Platform Productivity



Workflow Depth



Application Breadth

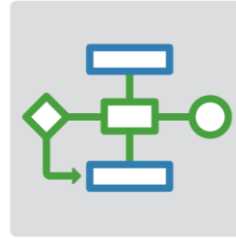


- **Create Your Designs Faster**
- **Simplify Analysis**
- **Execute Faster and Scale Your Work**
- **Collaborate**

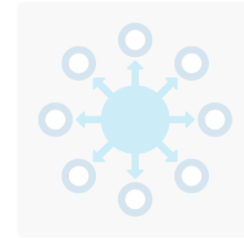
Platform Productivity



Workflow Depth



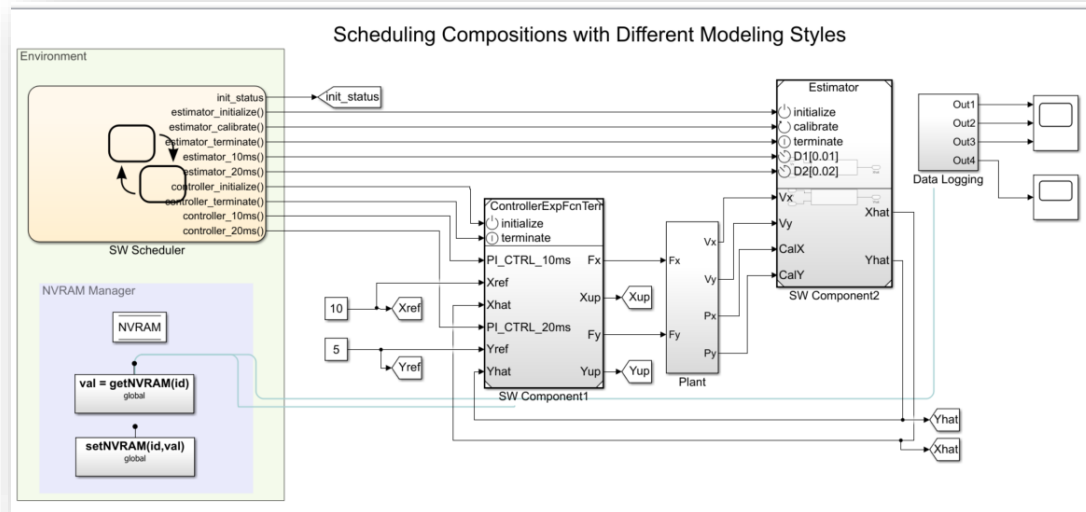
Application Breadth



- **Code Generation from Simulink Models**
- **Verification and Validation**

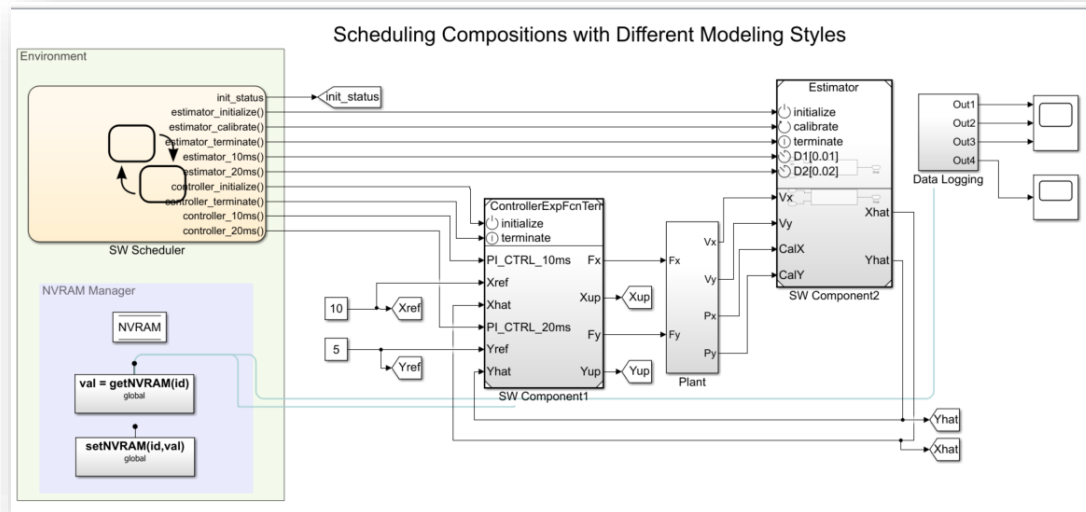
Prepare Your Model for Code Generation

Prepare model components
for code generation



Prepare Your Model for Code Generation

Prepare model components for code generation



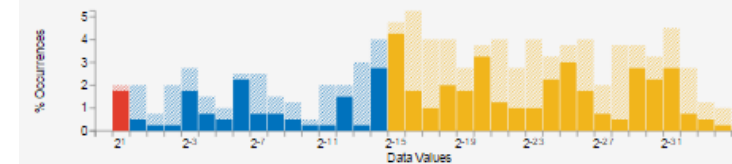
Prepare model data for code generation

The screenshot shows the "FIXED-POINT TOOL" interface. Key elements include:

- Simulation Ranges:** A dropdown menu showing "Simulation Ranges" and "Derived Ranges".
- Collect Ranges:** A button to collect simulation ranges.
- Propose Data Types:** A button to propose data types for code generation.
- Apply Data Types:** A button to apply the proposed data types.
- Simulate with Embedded Types:** A button to simulate the model with the embedded types.
- Compare Results:** A button to compare simulation results.

The interface also shows a "MODEL HIERARCHY" tree on the left and a "Results" table on the right.

Visualization of Simulation Data



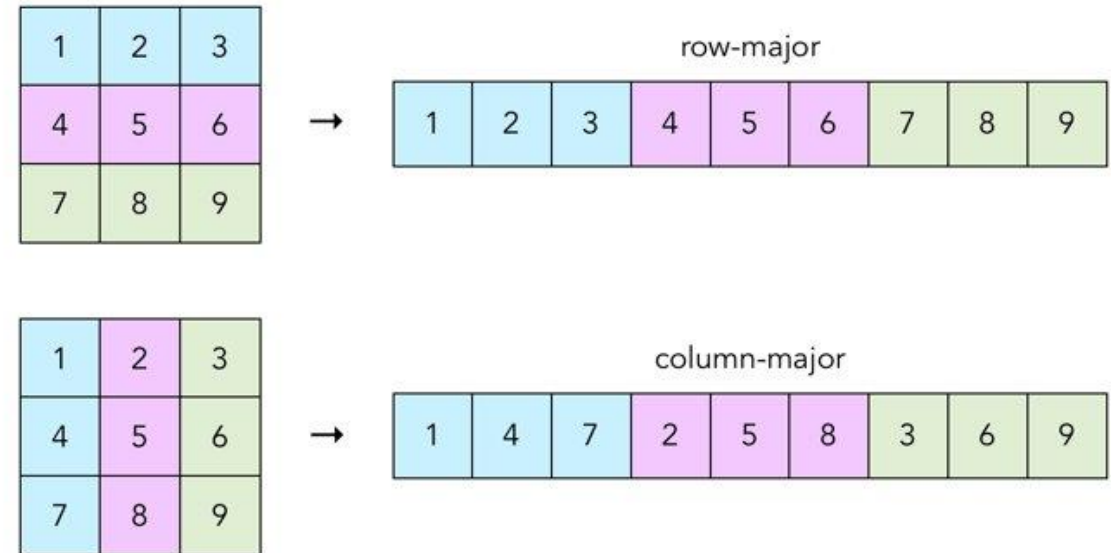
	Potential Overflows	In-Range	Potential Underflows
Positive Values	7	53	139
Negative Values	1	64	135

Number of times zero occurred: 0

Code Generation Workflow and Improvements

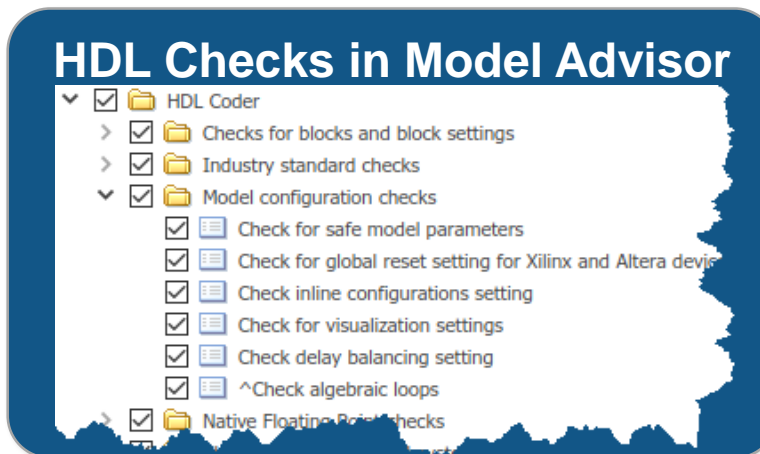
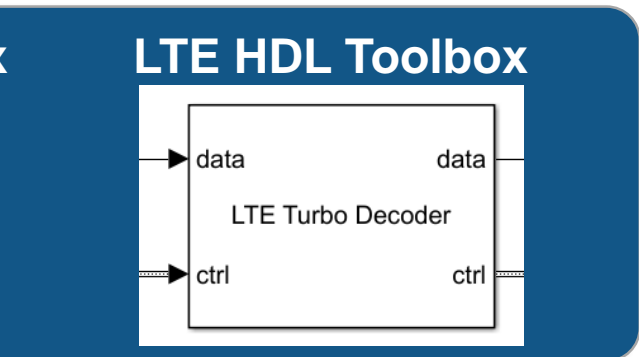
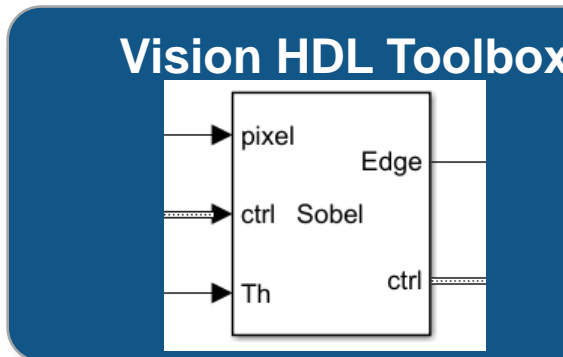
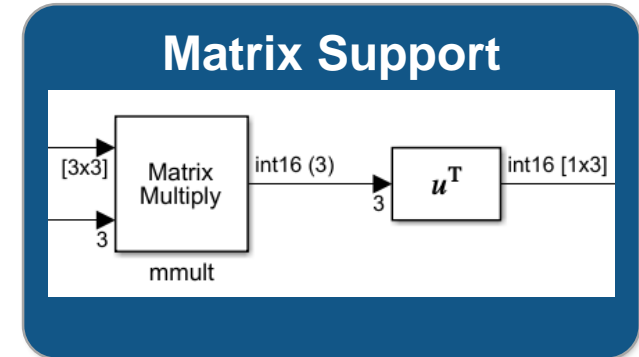
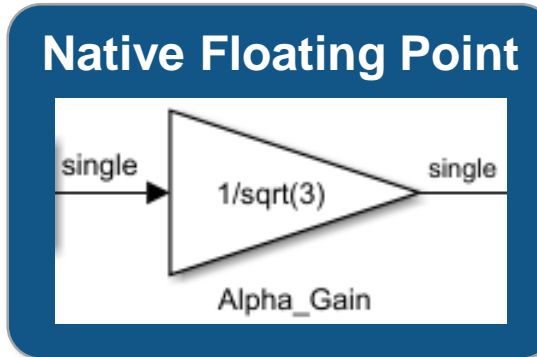
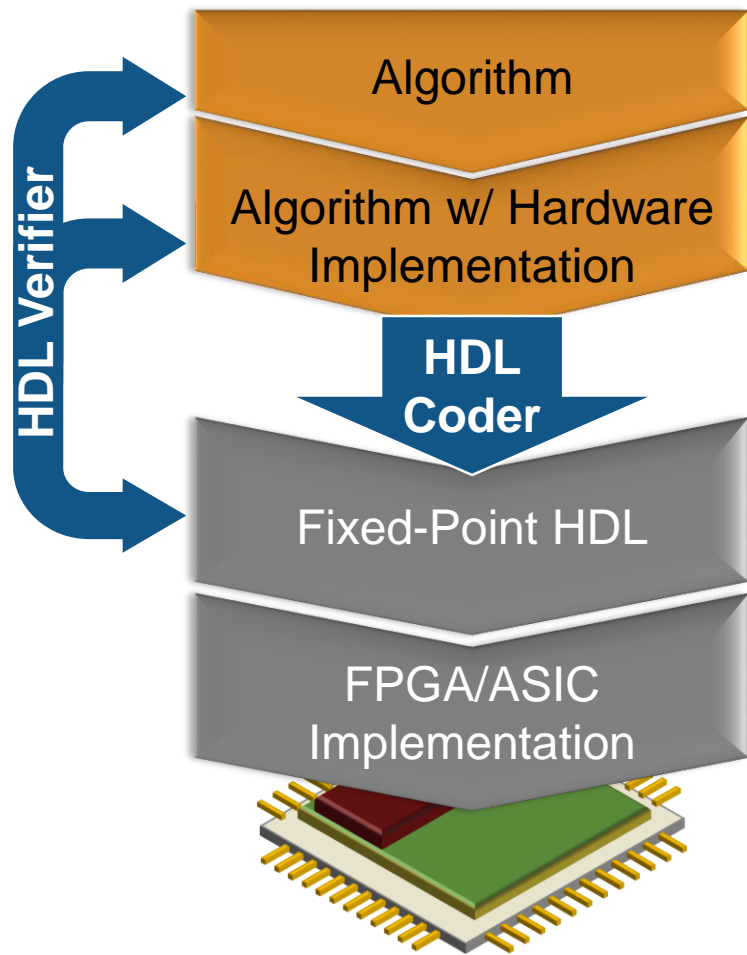
Access and define all the information in your model related to code generation

- View and define implementation data in one place
- View implementation details without model details
- Improve code performance and ease integration with other C code



Row-major memory layout option

Deploying to FPGA or ASIC Hardware



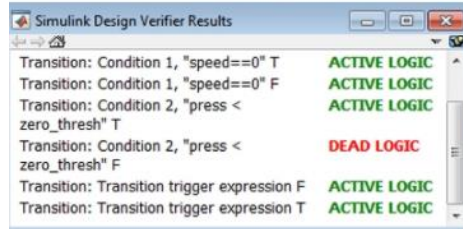
Verification and Validation

Products for the entire workflow

Simulink Requirements R2017b

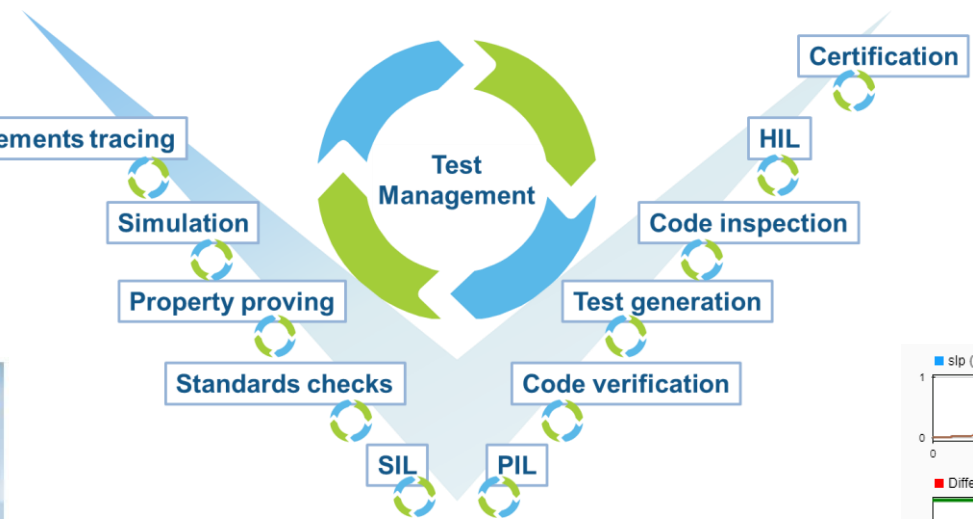


Simulink Design Verifier

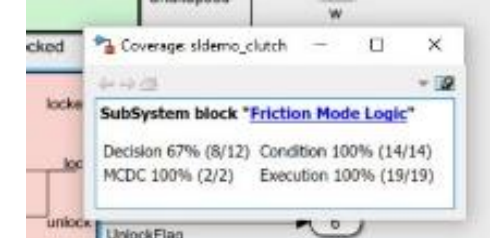


Simulink Check R2017b

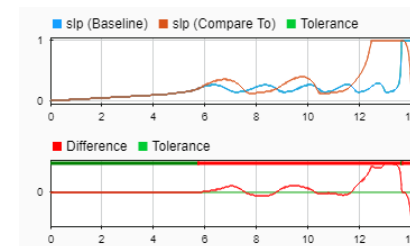
- Modeling Standards for Secure Coding (CERT C, CWE, ISO/IEC TS 17961)
 - Check configuration parameters for secure coding standards
 - Check for blocks not recommended for C/C++ production code deployment
 - Check for blocks not recommended for secure coding standards
 - Check usage of Assignment blocks
 - Check for switch case expressions without a default case
 - Check for bitwise operations on signed integers
 - Check for equality and inequality operations on floating-point values
 - Check integer word lengths
 - Detect Dead Logic



Simulink Coverage R2017b



Simulink Test



Polyspace

```

29  -----
30  |-----|
31  int intdiv(int p)
32  {
33      int i;
34      int j = 1;
35
36      i = 1024;
37      return i;
    
```

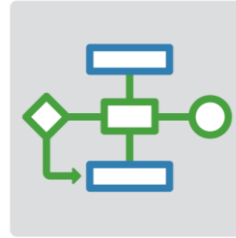
Probable cause for 'Integer division by zero':
 intdiv(1);
 operator / on type int 32
 left: 1024
 right: 0
 result: [-1024 .. 1024]

now supports
AUTOSAR
 R2018a

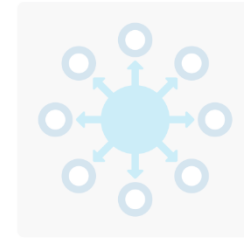
Platform Productivity



Workflow Depth



Application Breadth

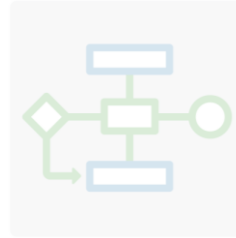


- **Code Generation from Simulink Models**
- **Verification and Validation**

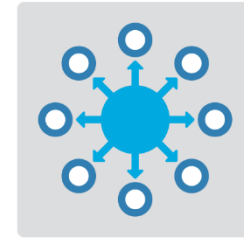
Platform Productivity



Workflow Depth

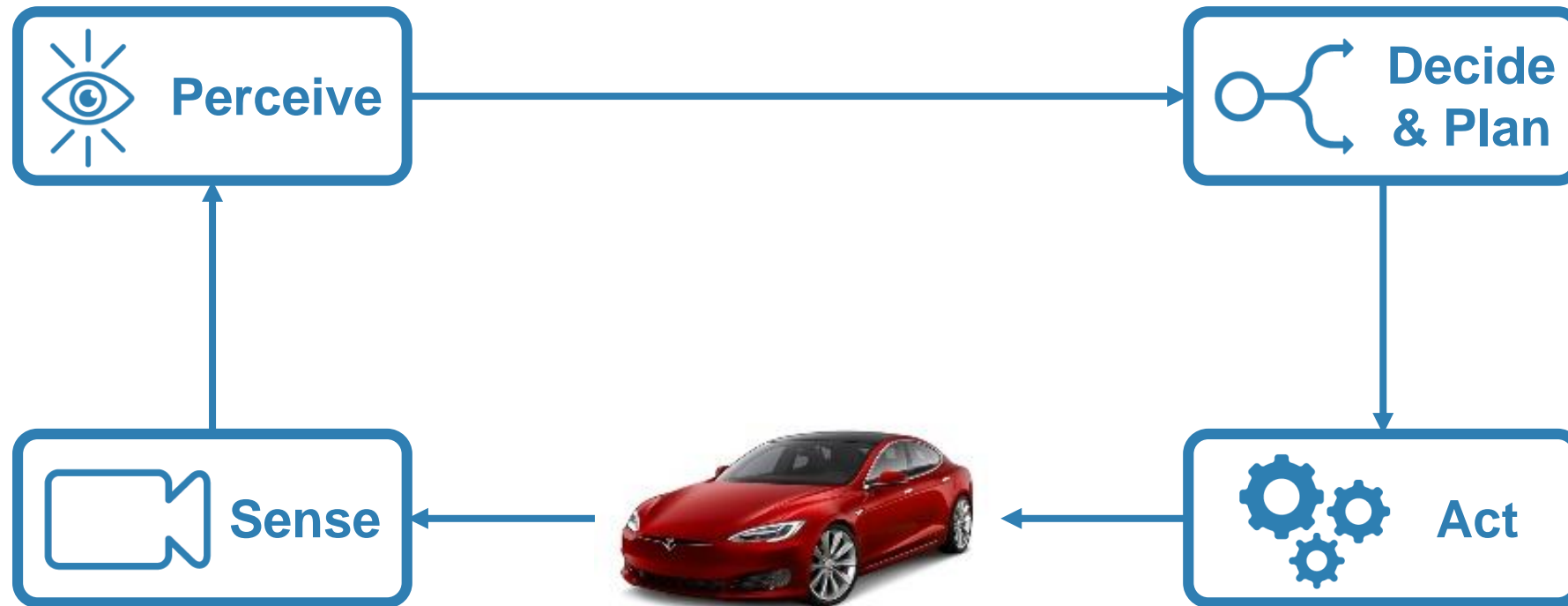


Application Breadth



- **Autonomous Systems**
- **Wireless Communications**
- **Artificial Intelligence (AI)**

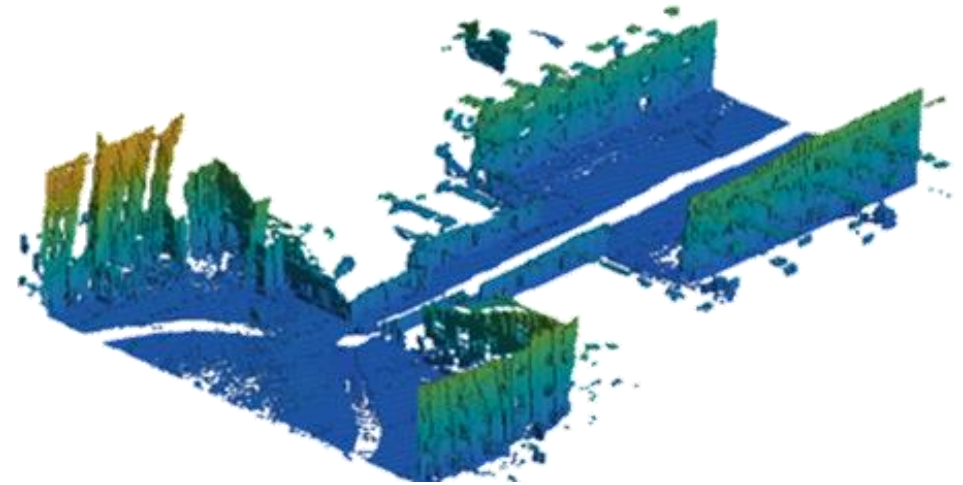
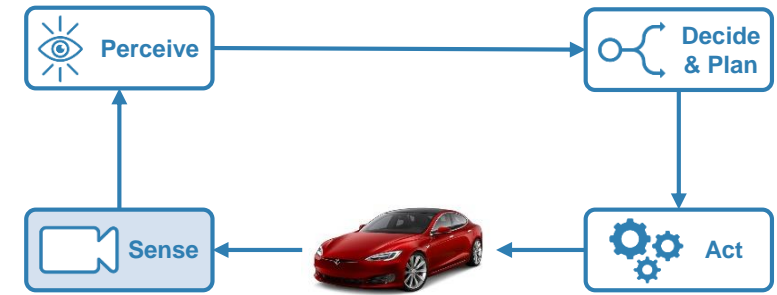
Designing Autonomous Systems



Designing Autonomous Systems

Mapping of environments using sensor data

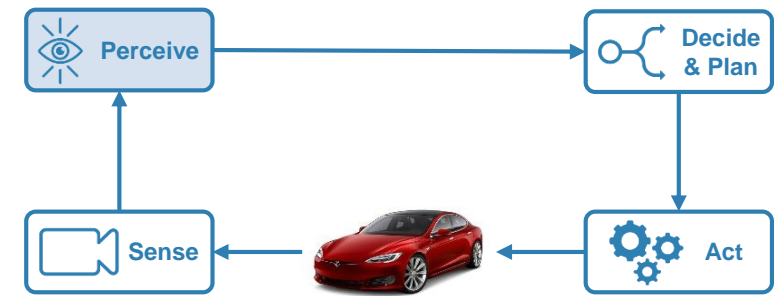
- Segment and register lidar point clouds
- Lidar-Based SLAM: Localize robots and build map environments using lidar sensors



Designing Autonomous Systems

Understanding the environment using computer vision and deep learning techniques

- Object detection and tracking
- Semantic segmentation using deep learning

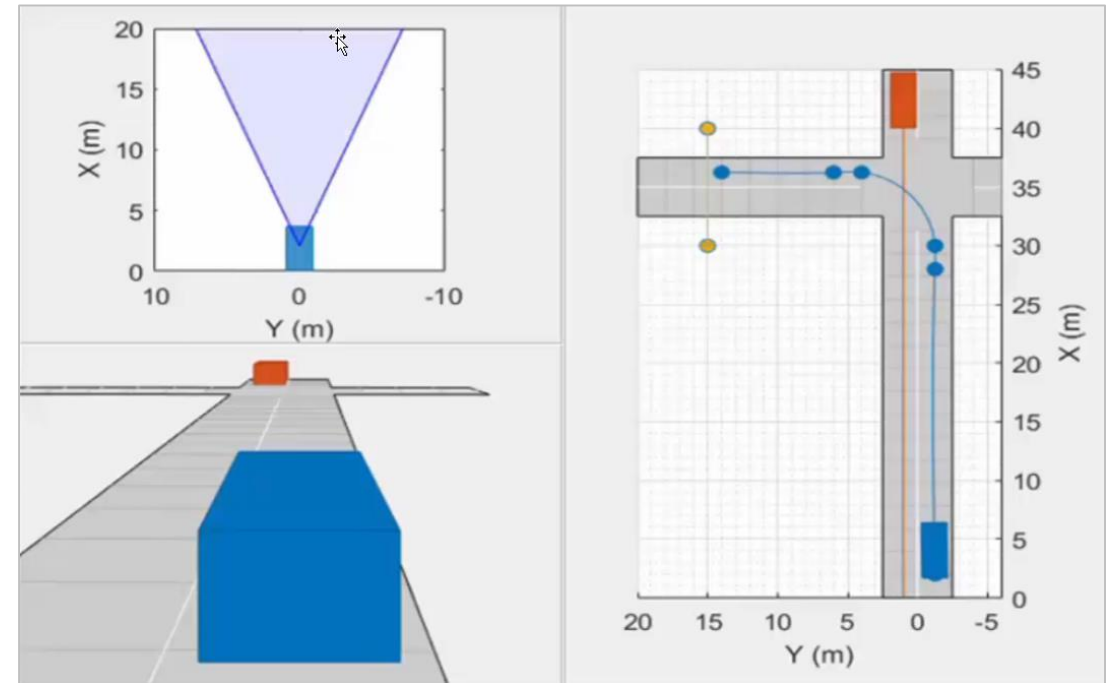
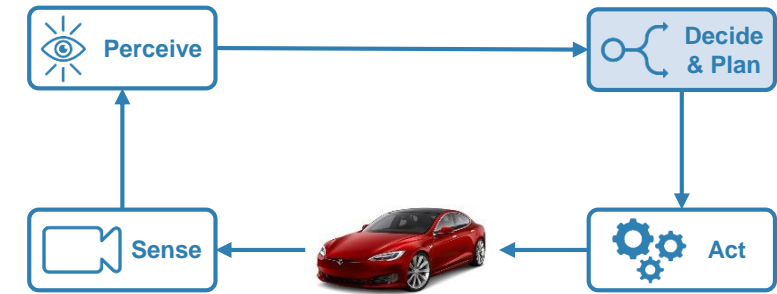


CamVid Database: Brostow, Gabriel J., Julien Fauqueur, and Roberto Cipolla. "Semantic object classes in video: A high-definition ground truth database." *Pattern Recognition Letters* Vol 30, Issue 2, 2009, pp 88-97.

Designing Autonomous Systems

Design synthetic driving scenarios to test controllers and sensor fusion algorithms

- Interactively design synthetic driving scenarios composed of roads and actors (*vehicles, pedestrians, etc.*)
- Generate visual and radar detections of actors

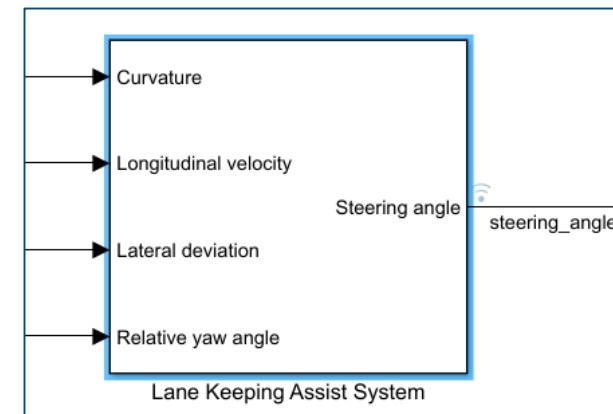
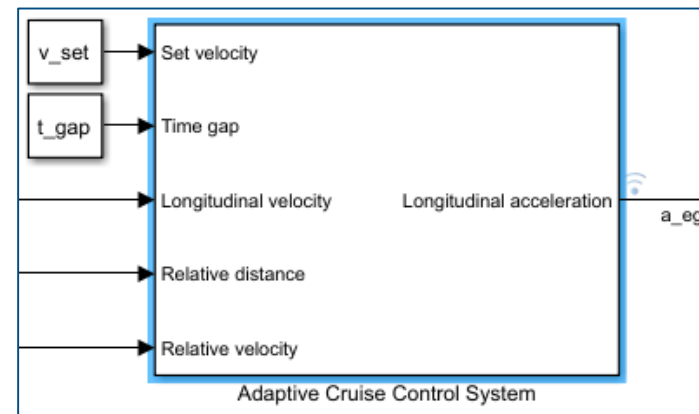
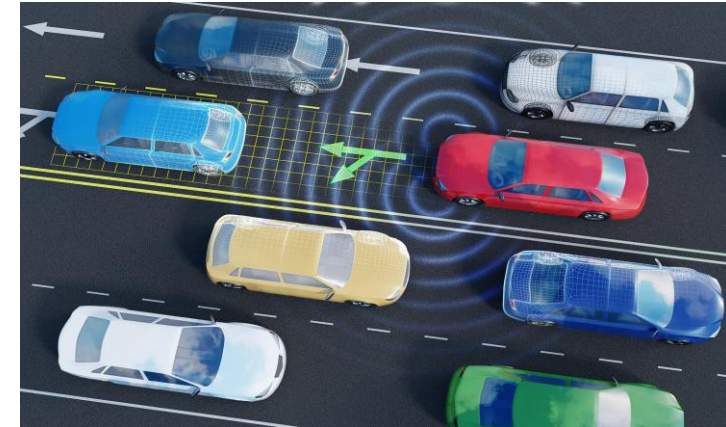
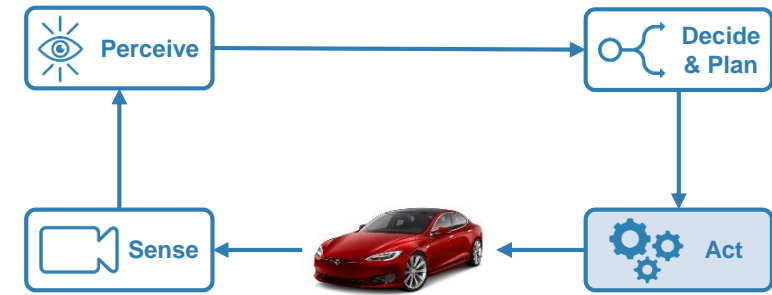


Driving Scenario Designer App

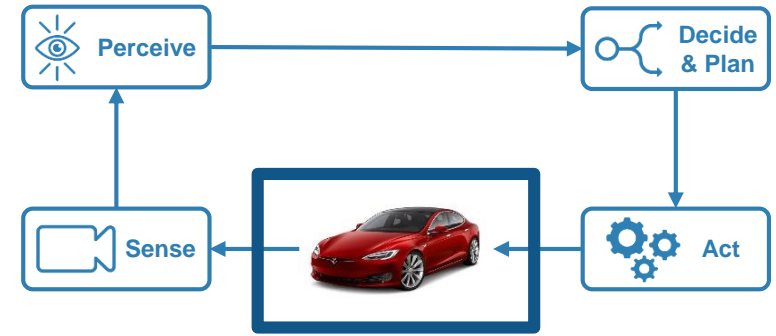
Designing Autonomous Systems

Model predictive control for adaptive cruise control and lane-keeping algorithms

- Use prebuilt blocks instead of starting from scratch
- Simplified application-specific interfaces for configuring model predictive controllers
- Flexibility to customize for your application



Full Vehicle Simulation



Ride & handling



Chassis controls



Automated Driving

Design with the Latest Wireless Standards



Lte™
Advanced
Pro



5G™



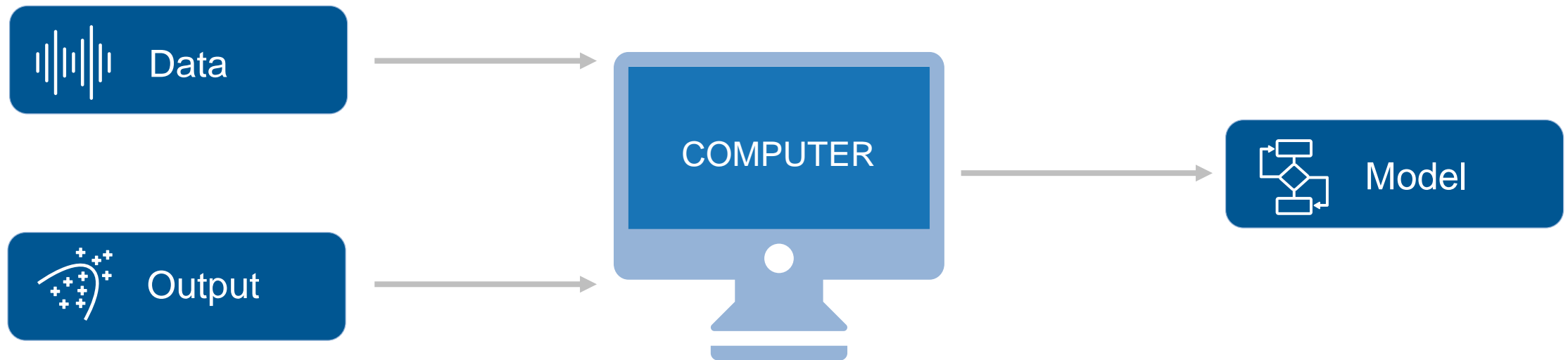
WiFi™
802.11ax



ZigBee®

NB-IoT

Artificial Intelligence

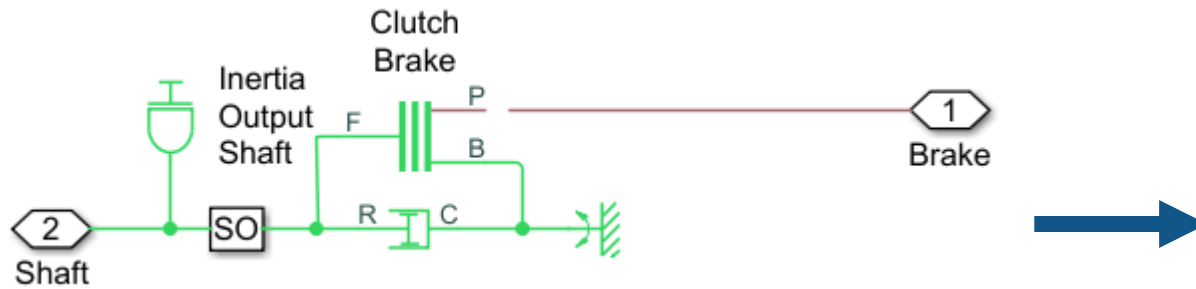


Predictive Maintenance

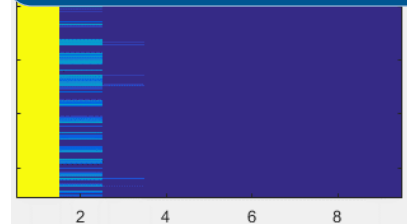
 Data

 Sensors

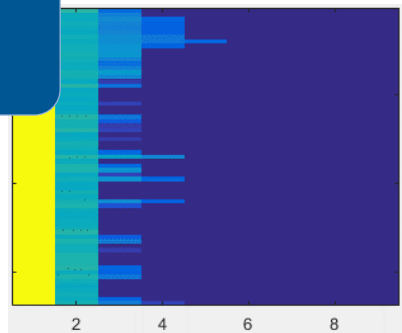
 Model



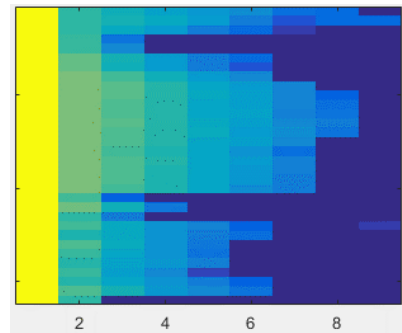
 Output



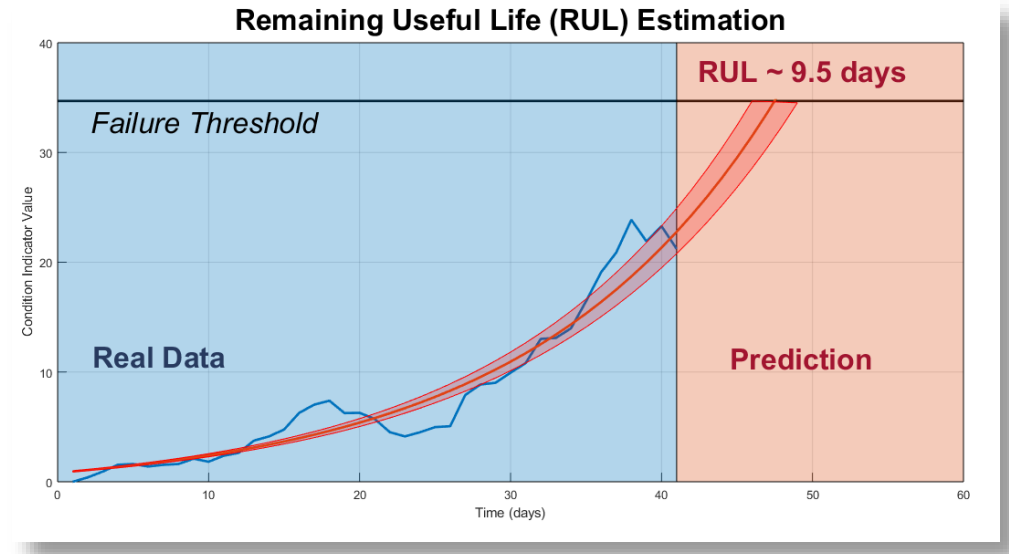
Normal Operation



Monitor Closely



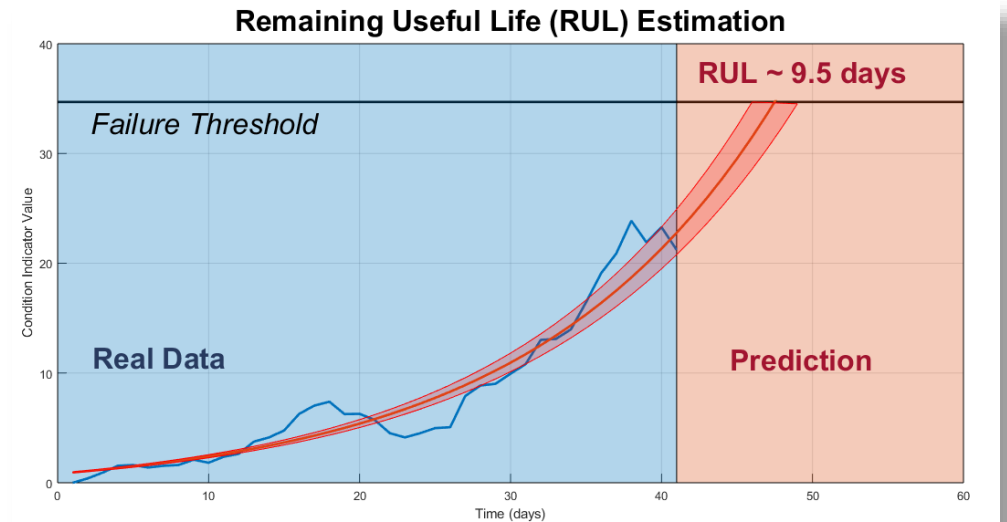
Maintenance Needed



Predictive Maintenance

Design and test condition monitoring and predictive maintenance algorithms

- Import sensor data from local files and cloud storage (*Amazon S3, Windows Azure Blob Storage, and Hadoop HDFS*)
- Use simulated failure data from Simulink models
- Get started with examples (*motors, gearboxes, batteries, and other machines*)

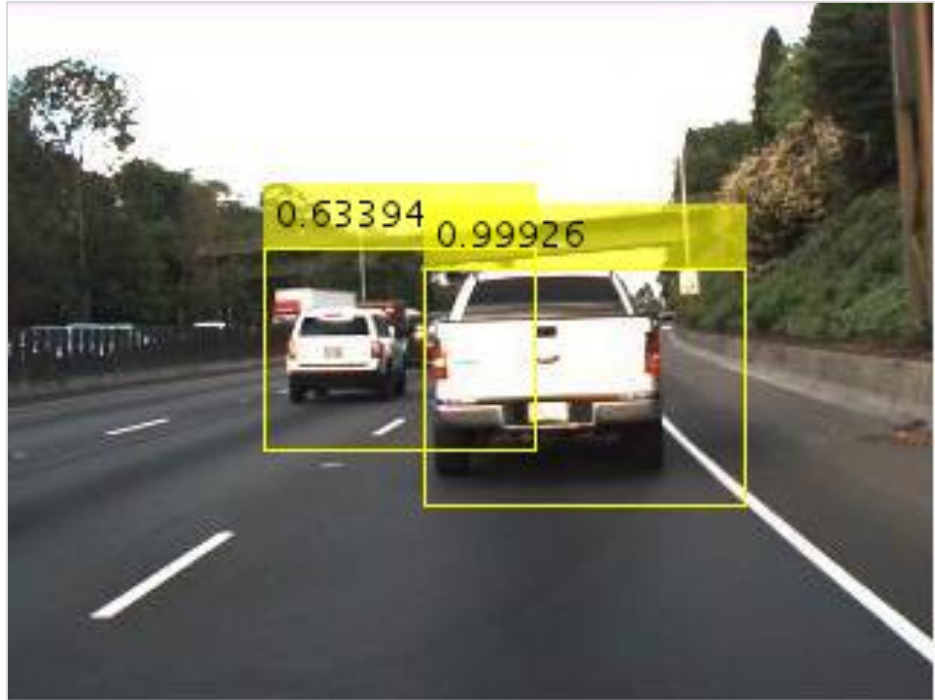


Deep Learning

 Data



 Model



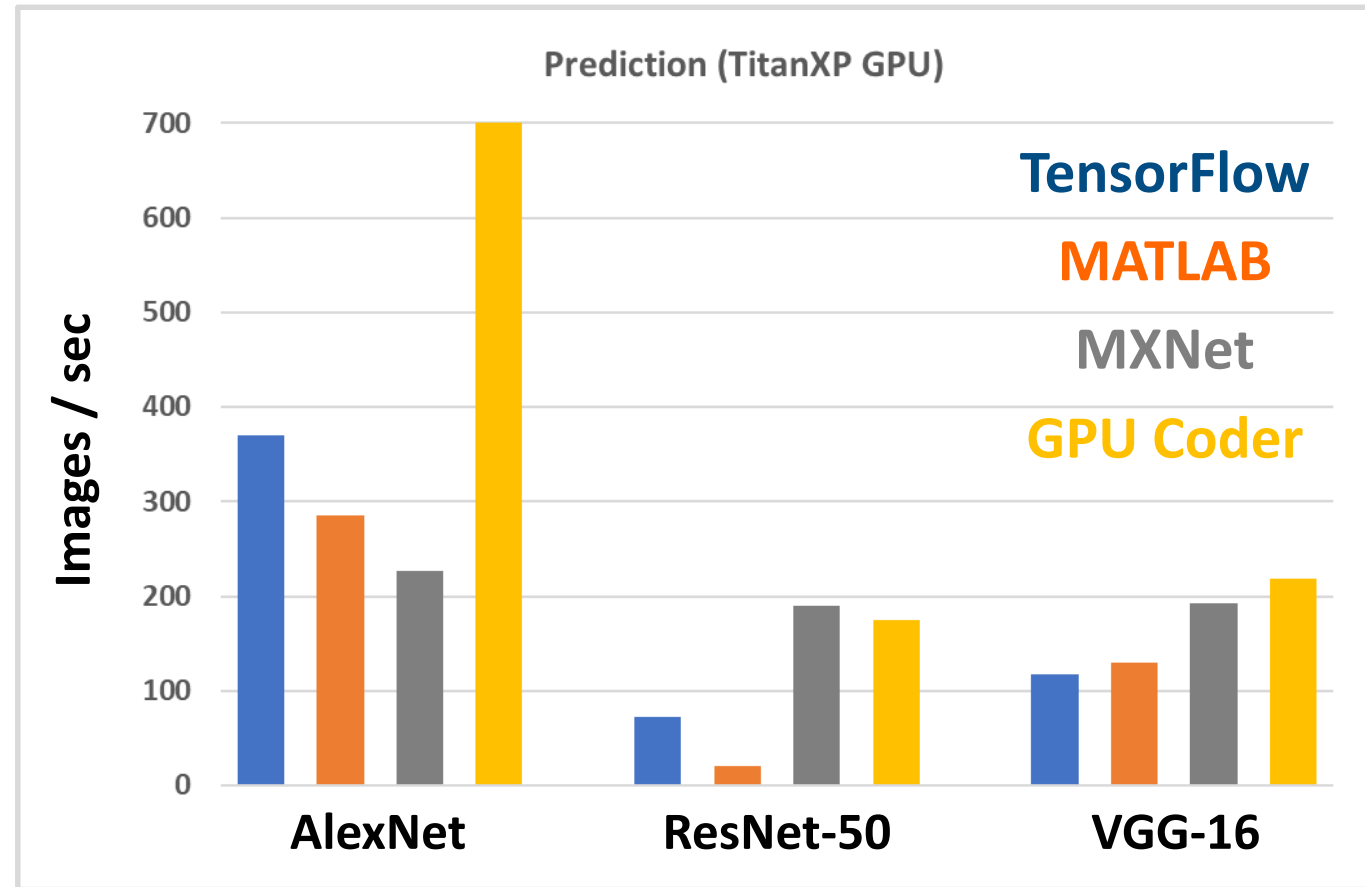
 Output



Deep Learning

Design, build, and visualize neural networks

- Access the latest models
- Import pretrained models and use transfer learning
- Automate ground-truth labeling using apps
- Design and build your own models
- Use NVIDIA GPUs to train your models
- Automatically generate high-performance CUDA code for embedded deployment



FREE

Learn to Use MATLAB for Deep Learning in 2 Hours

Launch Deep Learning Onramp

The screenshot displays the MATLAB Deep Learning Onramp interface. The top navigation bar shows "My Courses" and "Deep Learning Onramp 51% complete". The current task is "2.2 Making Predictions: (1/2) Make a prediction".

Task 2 (Reset):

Info: You can use the `classify` function to make a prediction on an image.

```
pred = classify(net,img);
```

Use the `classify` function with the pretrained AlexNet network to predict the subject of the image stored in the variable `img1`. Store the network's prediction in a variable called `pred1`.

You may want to leave off the semicolon to see the result.

Buttons: Submit, Hint, See Solution, Next task

Correct!

Test Suite

- ✓ Is `pred1` created correctly?
 - Show test suite details

Task 3

Further Practice

Classify images

Load pretrained network

Task 1: Use the `alexnet` function to load a pretrained network.

```
deepnet = alexnet;
```

Import, view, and classify an image

Import and display the image in `file01.jpg`.

```
img1 = imread('file01.jpg');
imshow(img1)
```

Task 2: Classify the image in the variable `img1`.

```
pred1 = classify(deepnet,img1)
```

Classify further images

Task 3: Classify the images in `file02.jpg` and `file03.jpg`.

```
img2 = imread('file02.jpg');
```

The workspace shows the result: `pred1 = categorical seashore`. An image of a seashore is displayed in the workspace.

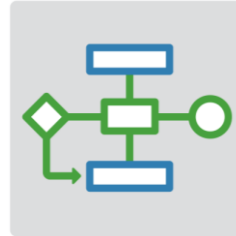
What's New in MATLAB and Simulink?

Platform Productivity



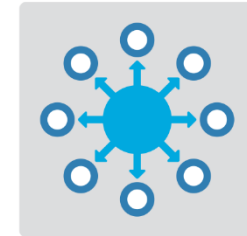
- Design Creation
- Analysis
- Simulation, Scaling
- Collaboration

Workflow Depth



- Code Generation
- Verification and Validation

Application Breadth



- Autonomous Systems
- Wireless Communications
- Artificial Intelligence (AI)

MATLAB EXPO 2018

