

# MATLAB EXPO 2018

What's New in MATLAB  
and Simulink **R2017b** **R2018a**

Michael Glaßer

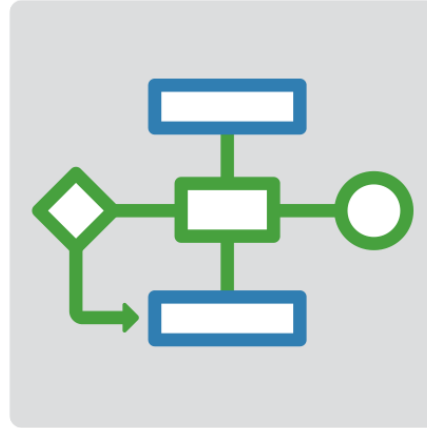


## Platform Productivity



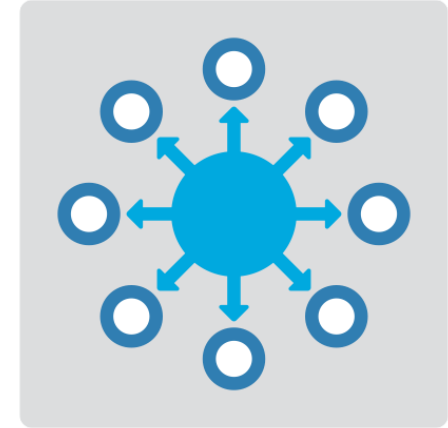
**Getting your work  
done faster**

## Workflow Depth



**Support for your  
entire workflow**

## Application Breadth

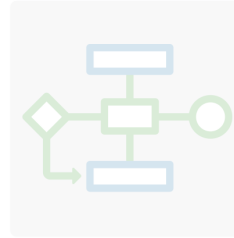


**Products for the  
work you do**

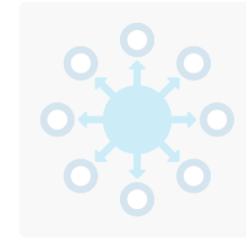
## Platform Productivity



## Workflow Depth

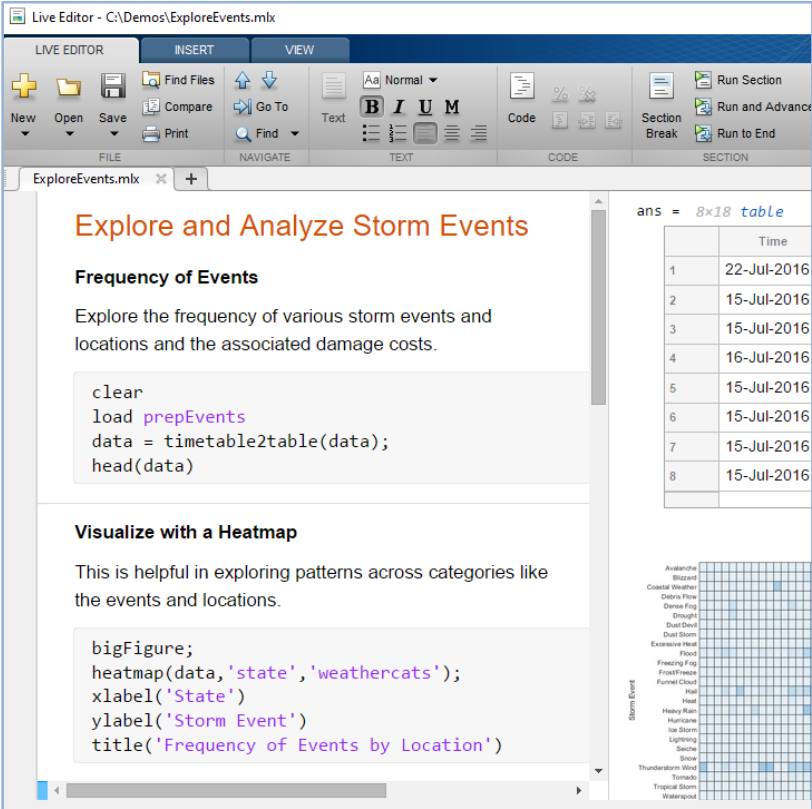


## Application Breadth



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

# Create Your Designs Faster



The screenshot displays the MATLAB Live Editor window titled "Live Editor - C:\Demos\ExploreEvents.mlx". The interface includes a ribbon with tabs for LIVE EDITOR, INSERT, and VIEW. The LIVE EDITOR tab is active, showing a script editor on the left and a live preview on the right. The script contains two sections: "Frequency of Events" and "Visualize with a Heatmap". The "Frequency of Events" section includes a code block that clears the workspace, loads a dataset named 'prepEvents', converts it to a timetable, and displays the first few rows. The "Visualize with a Heatmap" section includes a code block that creates a big figure, generates a heatmap of the data, and sets appropriate labels and titles. The live preview on the right shows the output of the script, including a table of dates and a heatmap titled "Frequency of Events by Location".

**Explore and Analyze Storm Events**

**Frequency of Events**

Explore the frequency of various storm events and locations and the associated damage costs.

```
clear
load prepEvents
data = timetable2table(data);
head(data)
```

**Visualize with a Heatmap**

This is helpful in exploring patterns across categories like the events and locations.

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

ans = 8x18 table

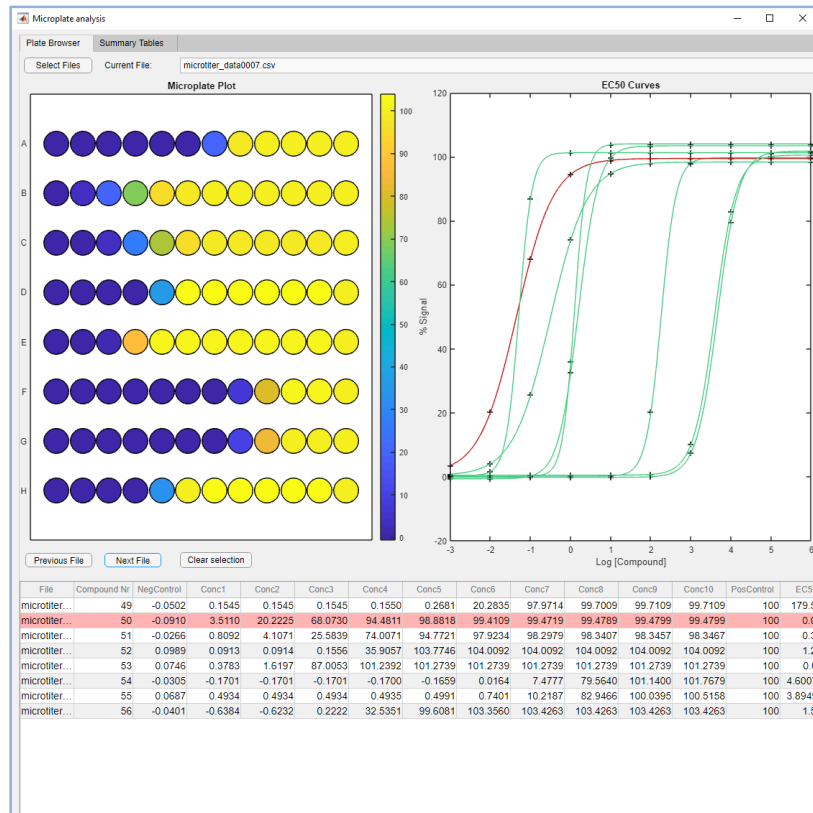
	Time
1	22-Jul-2016
2	15-Jul-2016
3	15-Jul-2016
4	16-Jul-2016
5	15-Jul-2016
6	15-Jul-2016
7	15-Jul-2016
8	15-Jul-2016

Storm Event

Heatmap visualization showing the frequency of various storm events (Y-axis) across different states (X-axis). The Y-axis categories include: Avalanche, Blizzard, Coastal Weather, Dense Fog, Dense Fog, Drought, Dust Storm, Explosive Heat, Flood, Freezing Fog, Frost/Freeze, Funnel Cloud, Hail, Heat, Heavy Rain, Hurricane, Ice Storm, Lightning, Seiche, Snow, Thunderstorm Wind, Tornado, Tropical Storm, and Waterspout. The X-axis categories are represented by columns in the heatmap grid.


**MATLAB**  
**Live Editor**

# Create Your Designs Faster




**MATLAB**

**App Designer**



## File Exchange

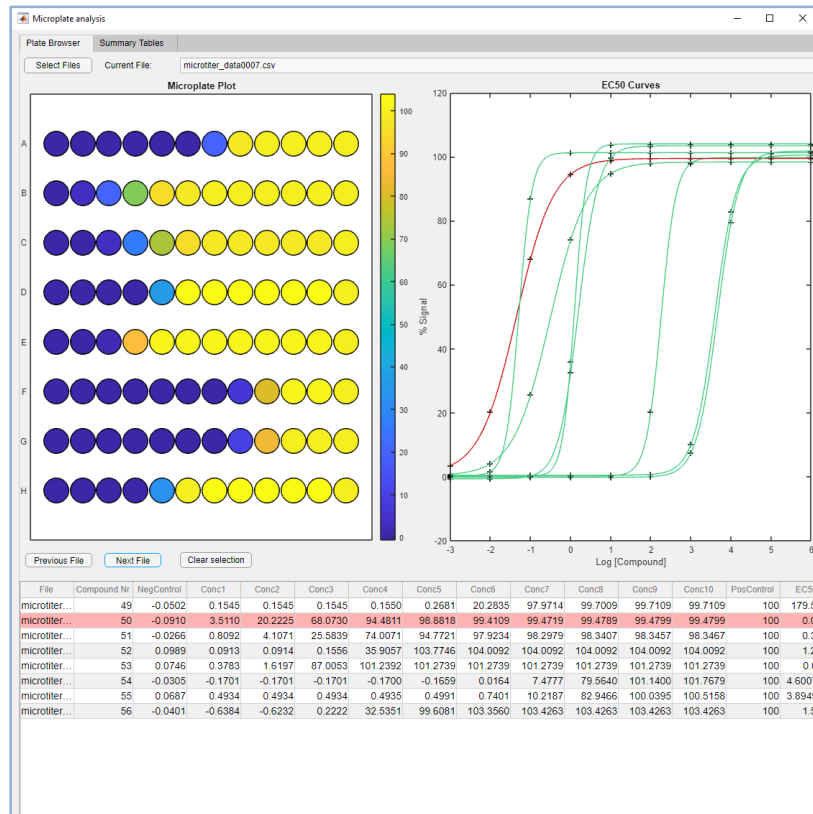


### GUIDE to App Designer Migration Tool for MATLAB

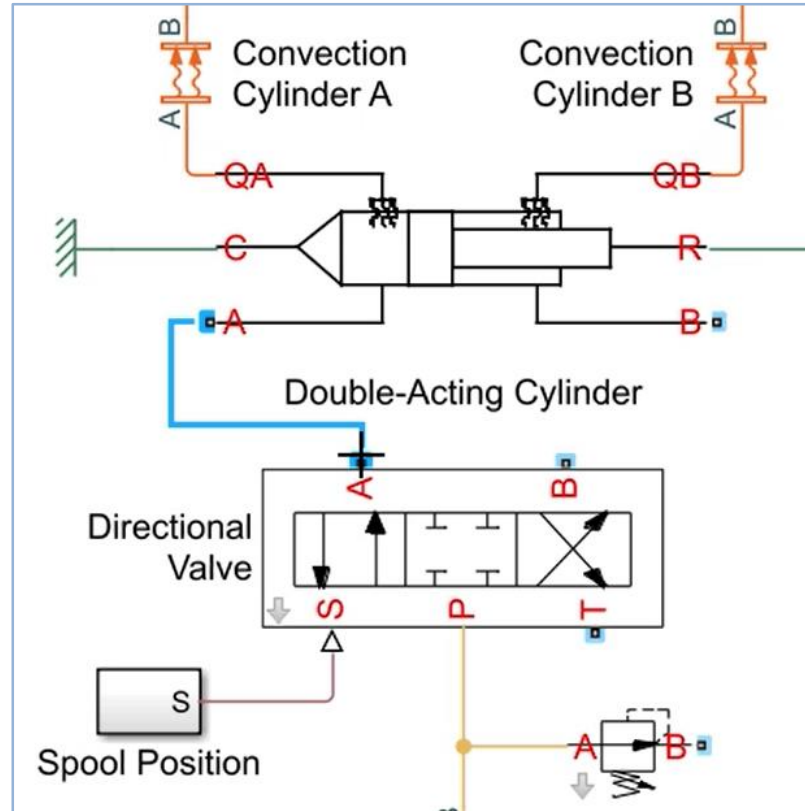
version 1.0 (15.1 KB) by MathWorks App Designer Team

Use the GUIDE to App Designer Migration tool to help transition your GUIDE apps to App Designer.

# Create Your Designs Faster

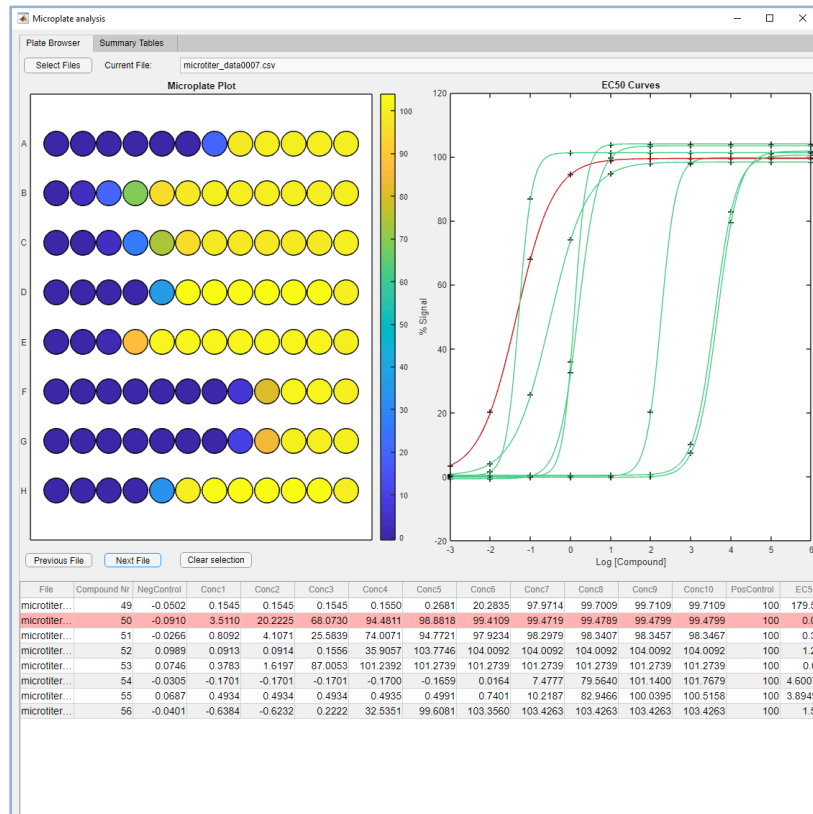


**MATLAB**

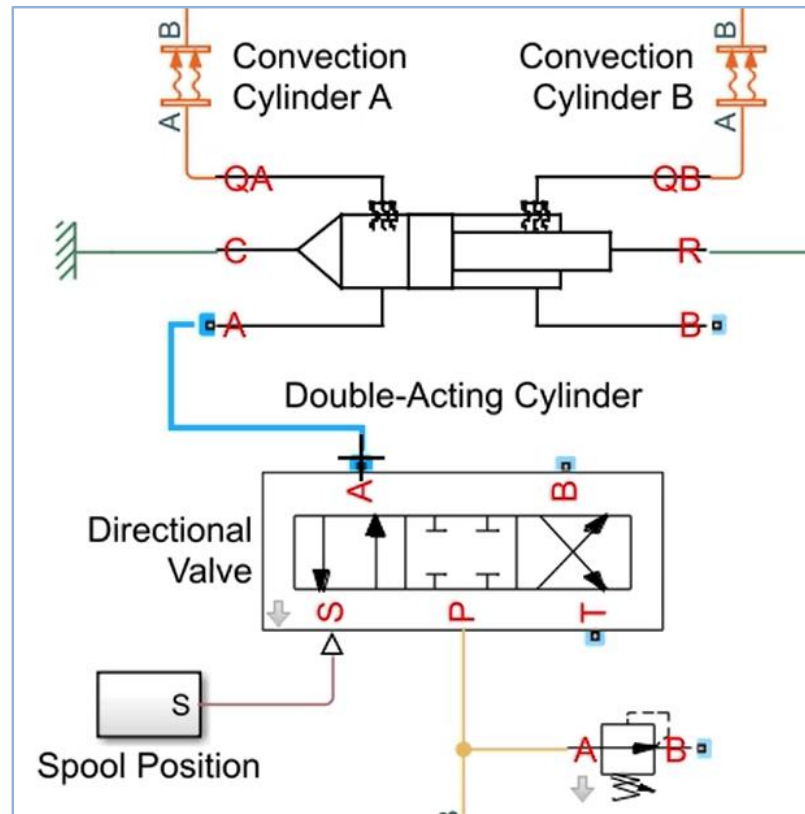


**Simulink**

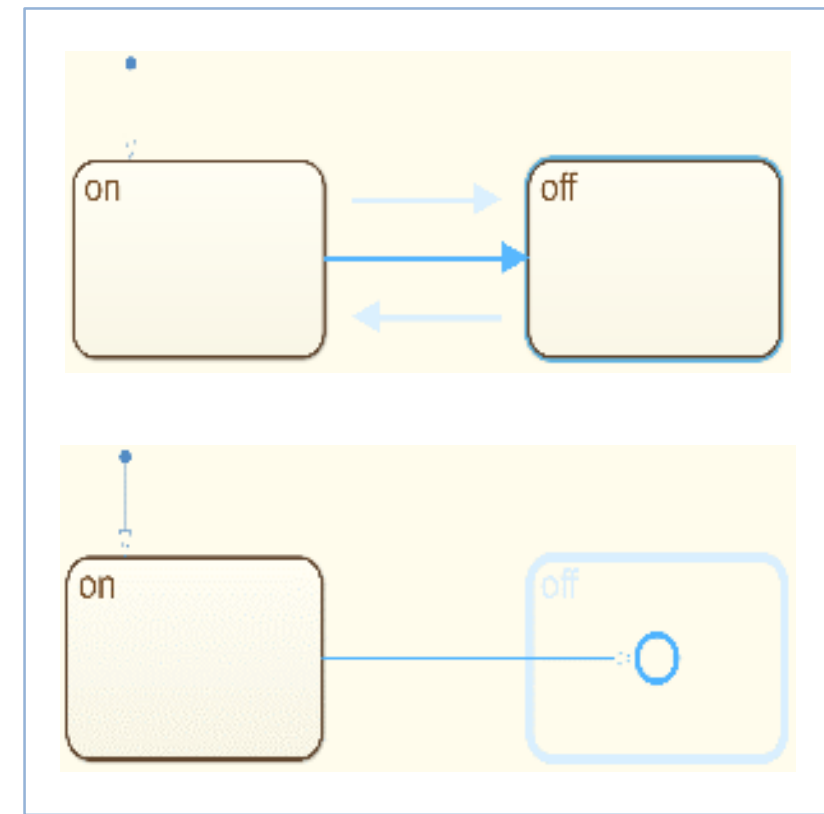
# Create Your Designs Faster



**MATLAB**



**Simulink**

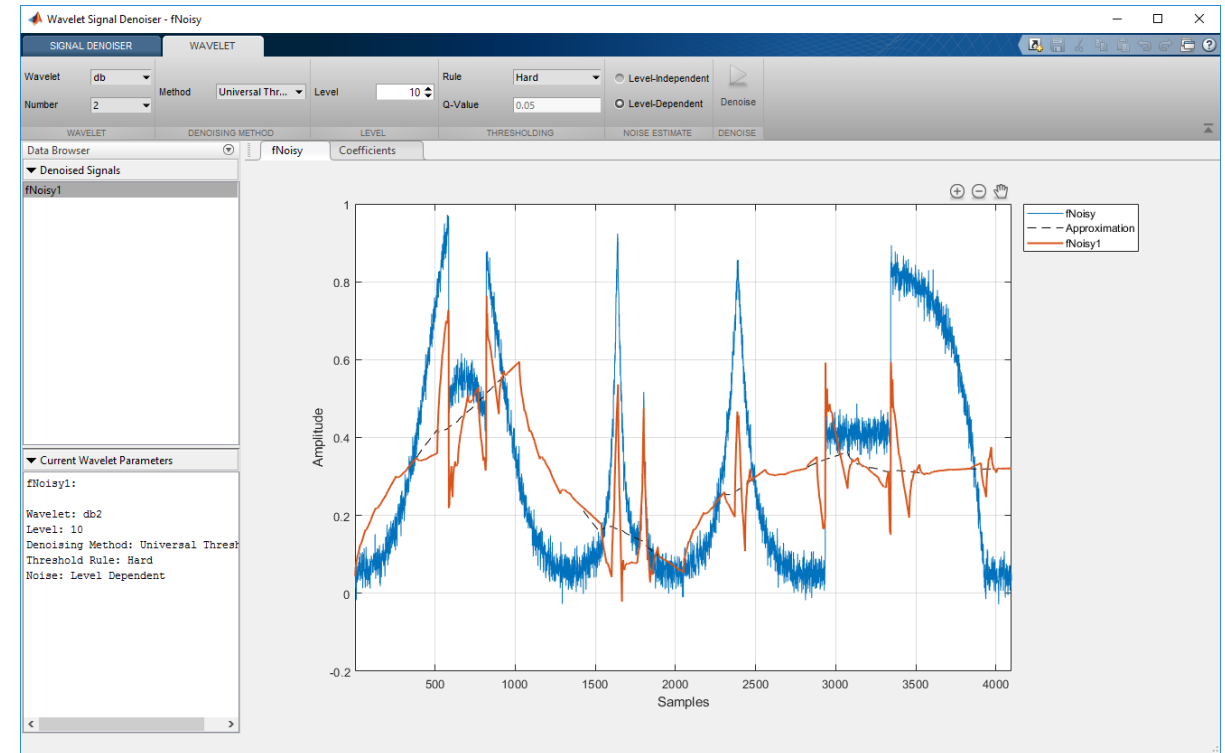


**Stateflow**

# Simplify Analysis with Apps

These interactive applications automate common technical computing tasks

- **Econometric Modeler app**
  - Perform time series analysis, specification testing, modeling, and diagnostics
- **Analog Input Recorder app**
  - Acquire and visualize analog input signals
- **Wavelet Signal Denoiser app**
  - Visualize and denoise time series data

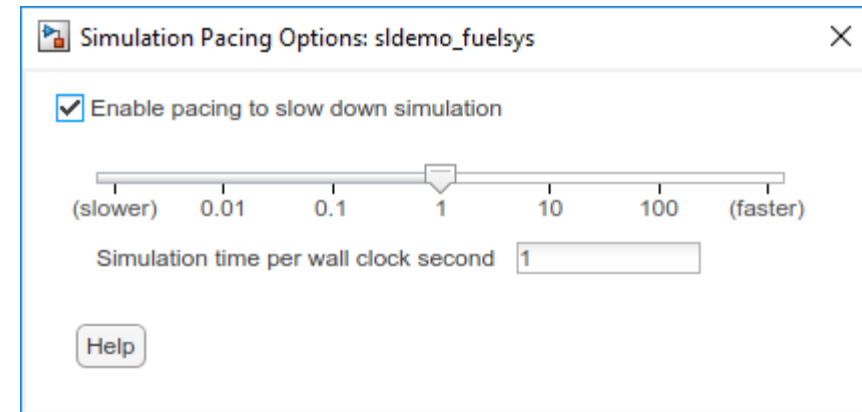




# Simplify Analysis by Simulating at Wall Clock Speed

## Slow down the simulation for easier model interactivity

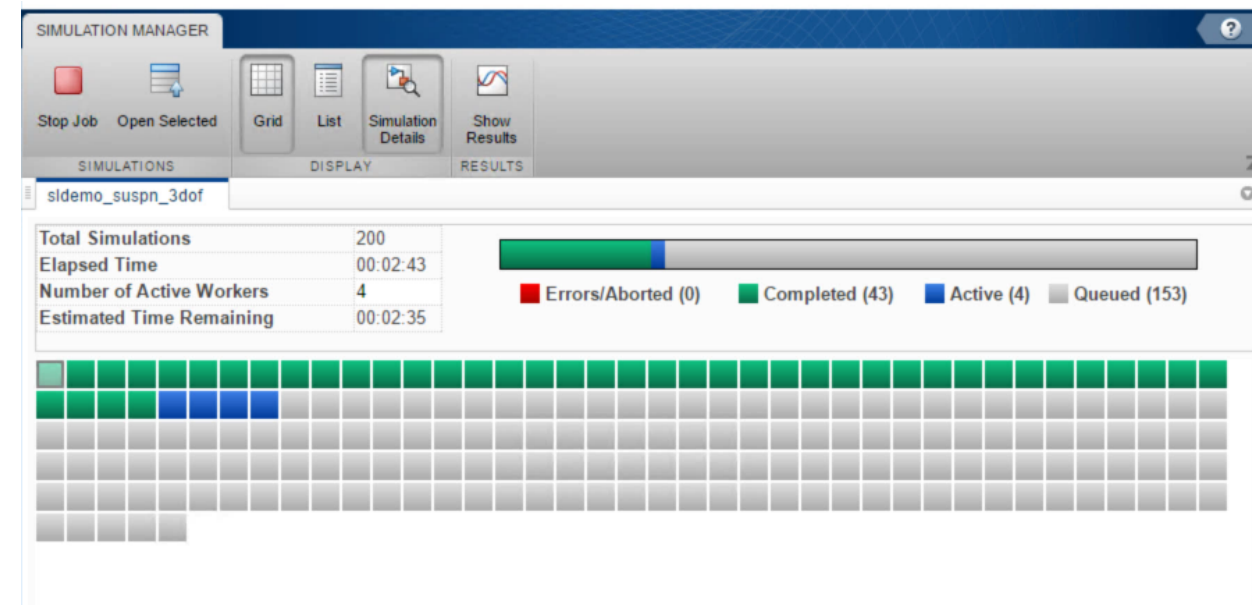
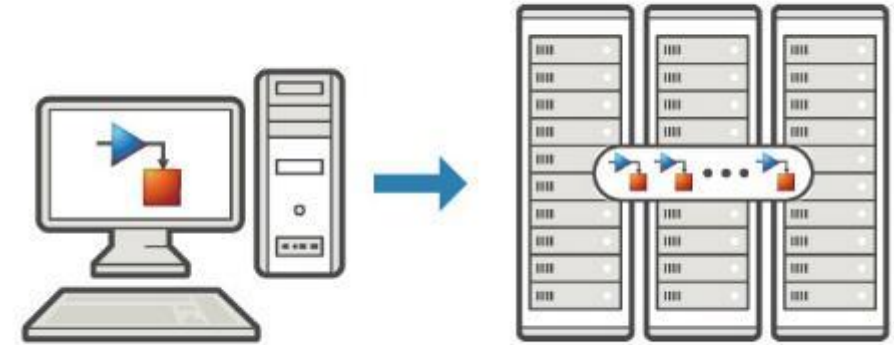
- Especially for models controlled and monitored via Dashboard blocks and other displays
- Useful when model is connected to hardware



# Scale Your Work

Use parallel computing to run multiple simulations faster

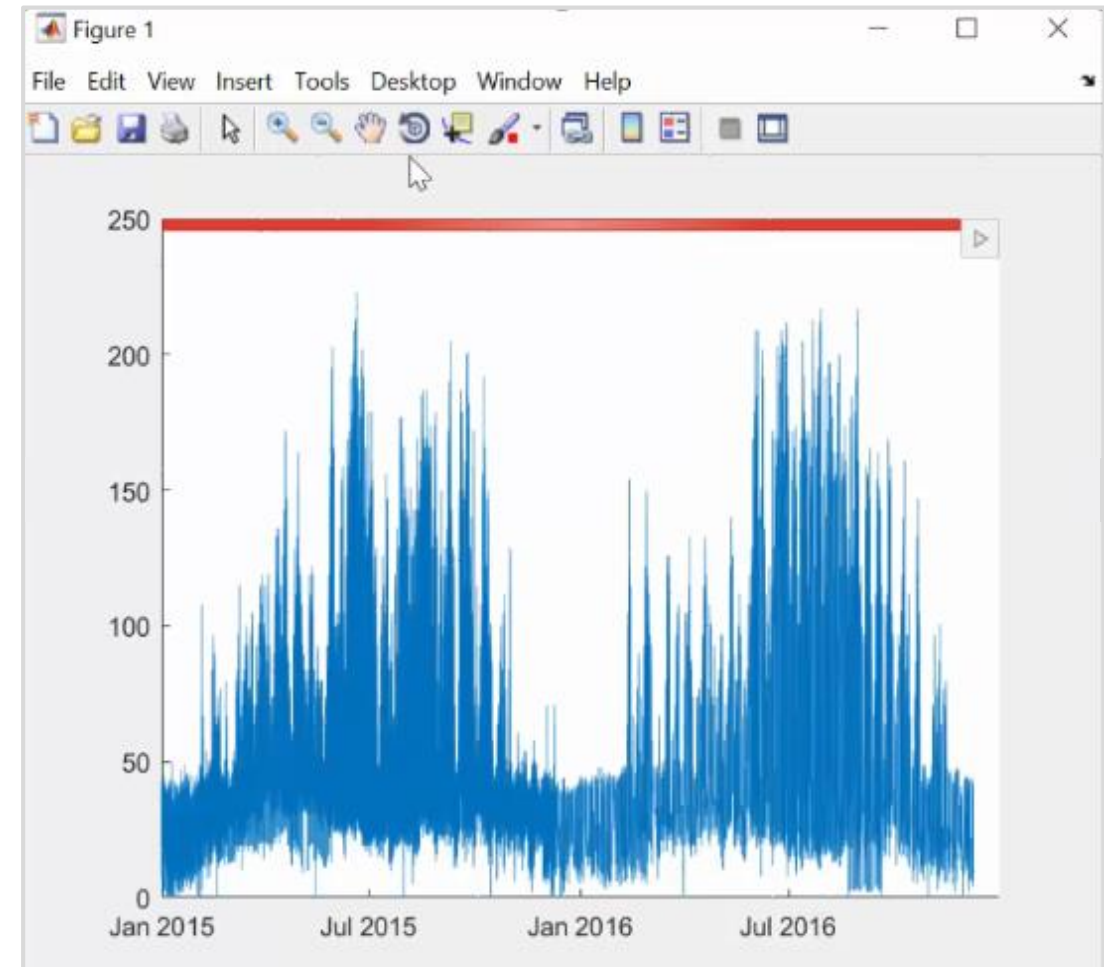
- Run multiple parallel simulations with **parsim**
- Monitor simulation status and progress in the Simulation Manager



# 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

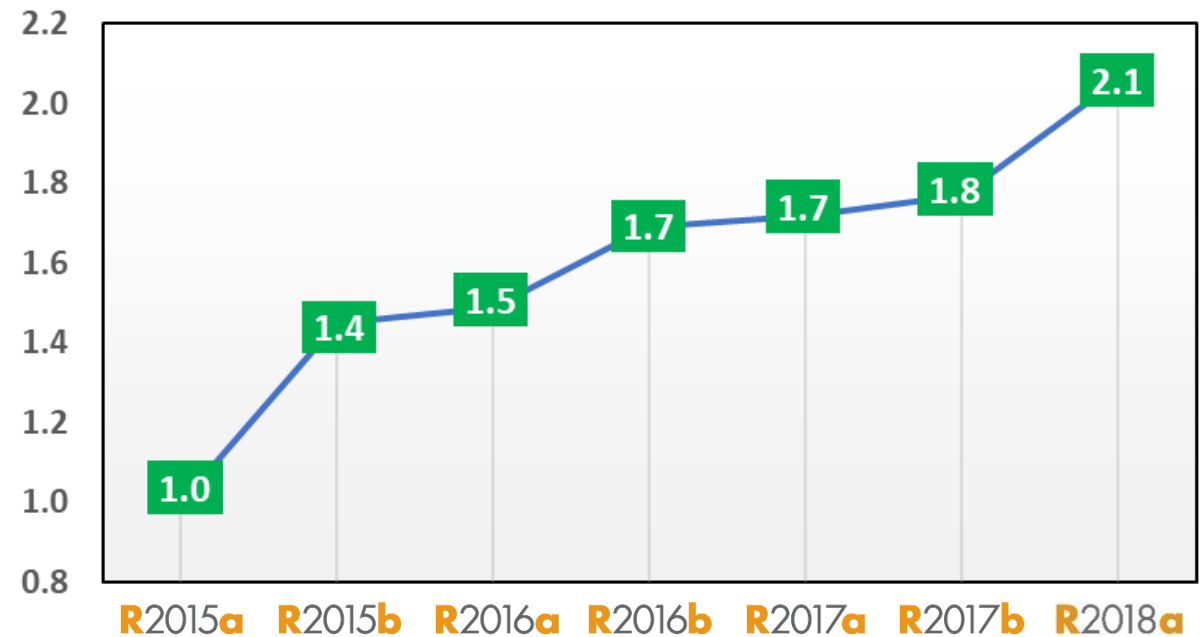


# Simulate Faster

## 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

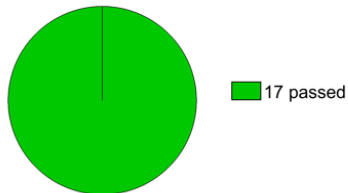
Use advanced software development features  
to manage, test, and integrate MATLAB code

## MATLAB® Test Report

Timestamp: 04-Jan-2017 13:28:06  
Host: AH-SDE  
Platform: win64  
MATLAB Version: 9.1.0.441655 (R2016b)

Number of Tests: 17  
Testing Time: 0.4516 seconds

Overall Result: PASSED



### Overview

[C:\Documents\MATLAB\OOP\Blip\Demos\Extensions\UnitTest\Class\](#)

<a href="#">BlipTests.BlipSizeLengthTests</a>	0.1403 seconds
<a href="#">BlipTests.BlipSubsasnTests</a>	0.1542 seconds
<a href="#">BlipTests.BlipSubsrefTests</a>	0.1572 seconds

### Details

[C:\Documents\MATLAB\OOP\Blip\Demos\Extensions\UnitTest\Class\](#)

[BlipTests.BlipSizeLengthTests](#)

• [scalarBlipSize](#)

The test passed.  
Duration: 0.0863 seconds

[\(Overview\)](#)

• [vectorBlipSize](#)

The test passed.  
Duration: 0.0027 seconds

[\(Overview\)](#)

• [scalarBlipLength](#)

The test passed.  
Duration: 0.0044 seconds

[\(Overview\)](#)

# Team Collaboration

Use advanced software development features to manage, test, and integrate MATLAB code

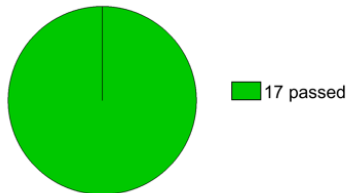
Identify differences between model elements, Stateflow charts, and MATLAB Function blocks

## MATLAB® Test Report

Timestamp: 04-Jan-2017 13:28:06  
Host: AH-SDE  
Platform: win64  
MATLAB Version: 9.1.0.441655 (R2016b)

Number of Tests: 17  
Testing Time: 0.4516 seconds

Overall Result: PASSED



### Overview

C:\Documents\MATLAB\OOP\Blip\Demos\Extensions\UnitTest\Class\

BlipTests.BlipSizeLengthTests	0.1403 seconds
BlipTests.BlipSubsasgnTests	0.1542 seconds
BlipTests.BlipSubsrefTests	0.1572 seconds

### Details

C:\Documents\MATLAB\OOP\Blip\Demos\Extensions\UnitTest\Class\

BlipTests.BlipSizeLengthTests

scalarBlipSize

The test passed.  
Duration: 0.0863 seconds

(Overview)

vectorBlipSize

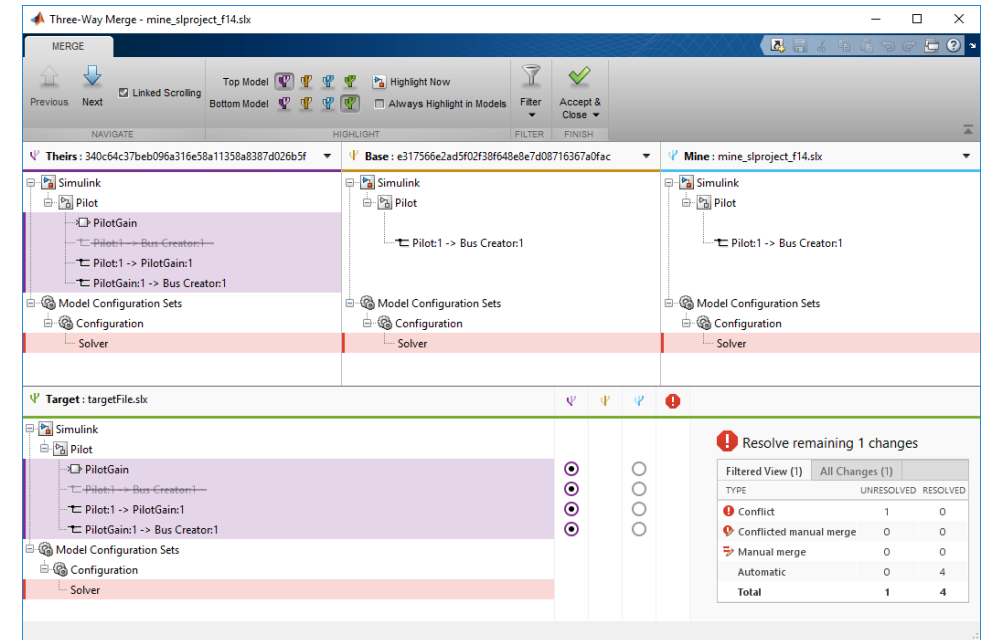
The test passed.  
Duration: 0.0027 seconds

(Overview)

scalarBlipLength

The test passed.  
Duration: 0.0044 seconds

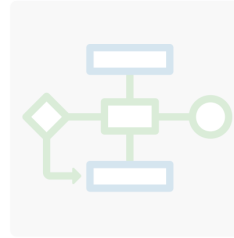
(Overview)



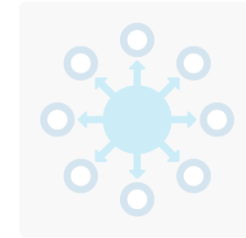
## Platform Productivity



## Workflow Depth



## Application Breadth

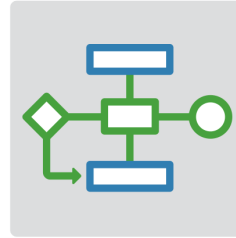


- **Create Your Designs Faster**
- **Simplify Analysis**
- **Simulate Faster and Scale Your Work**
- **Collaborate**

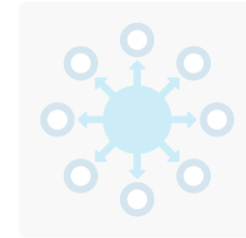
## Platform Productivity



## Workflow Depth



## Application Breadth



- **Deployment of MATLAB Algorithms and Applications**
- **Code Generation from Simulink Models**
- **Verification and Validation**



# Deploy MATLAB Algorithms and Applications

## Access Data



Sensors



Files



Databases

## Analyze Data



Data exploration



Preprocessing



Domain-specific algorithms

## Develop



AI model



Algorithm development



Modeling & simulation

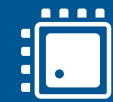
## Deploy



Desktop apps

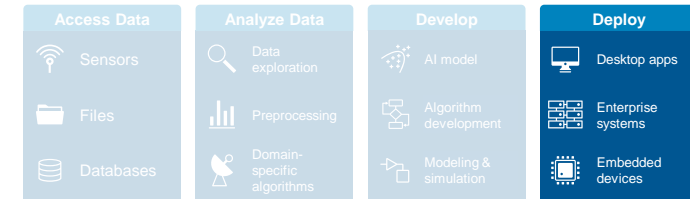


Enterprise systems



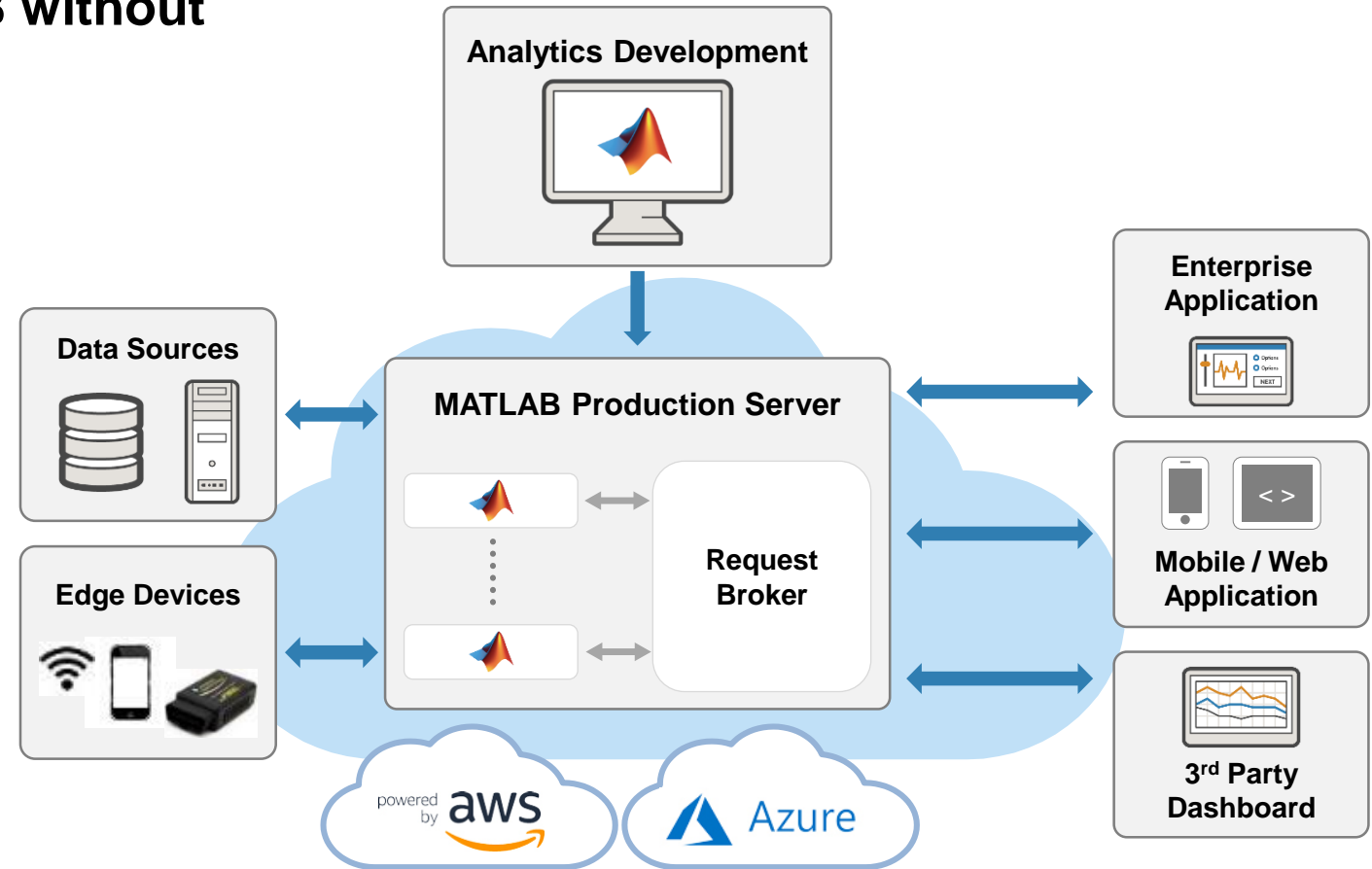
Embedded devices

# Deploy MATLAB Algorithms and Applications

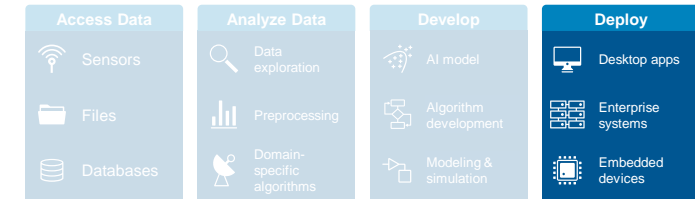


## Share your work outside of MATLAB without having to recode your algorithms

- Standalone desktop applications
- Add-ins for Microsoft Excel
- Software components to integrate with other languages (*C/C++*, *.NET*, *Python*, *Java*)
- Software components for web and enterprise applications

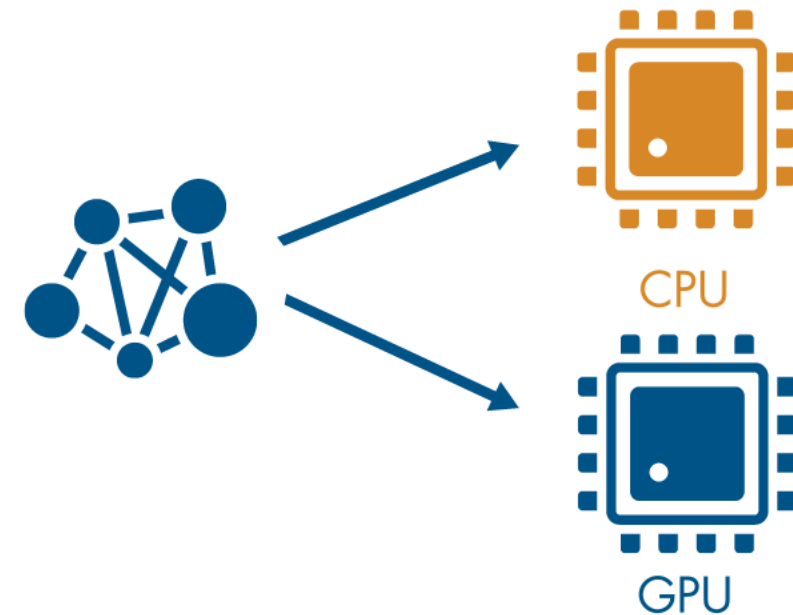


# Deploy MATLAB Algorithms



## Deploy machine learning and deep learning models using automatically generated code

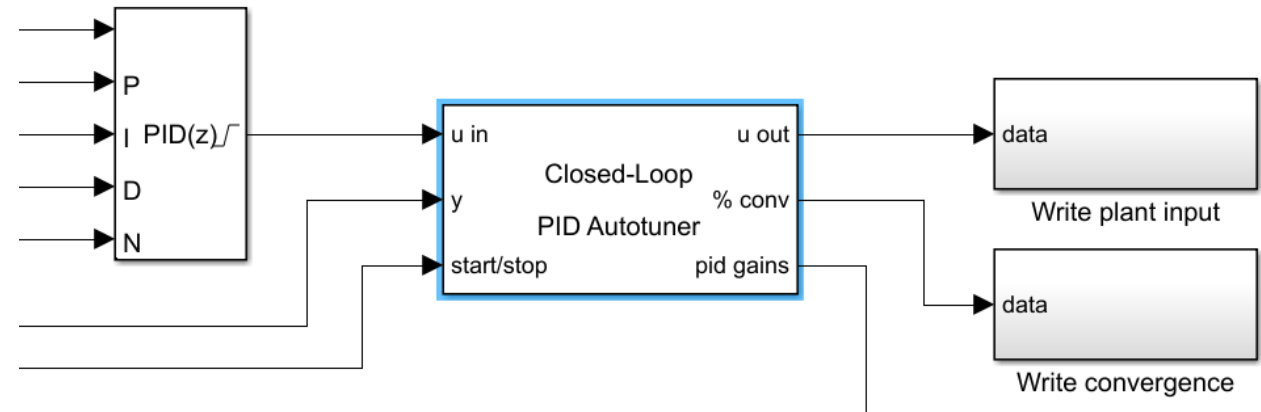
- Generate C code for predictive machine learning and deep learning models
- Generate optimized CUDA code for deep learning, embedded vision, and autonomous systems



# PID Control Tuning

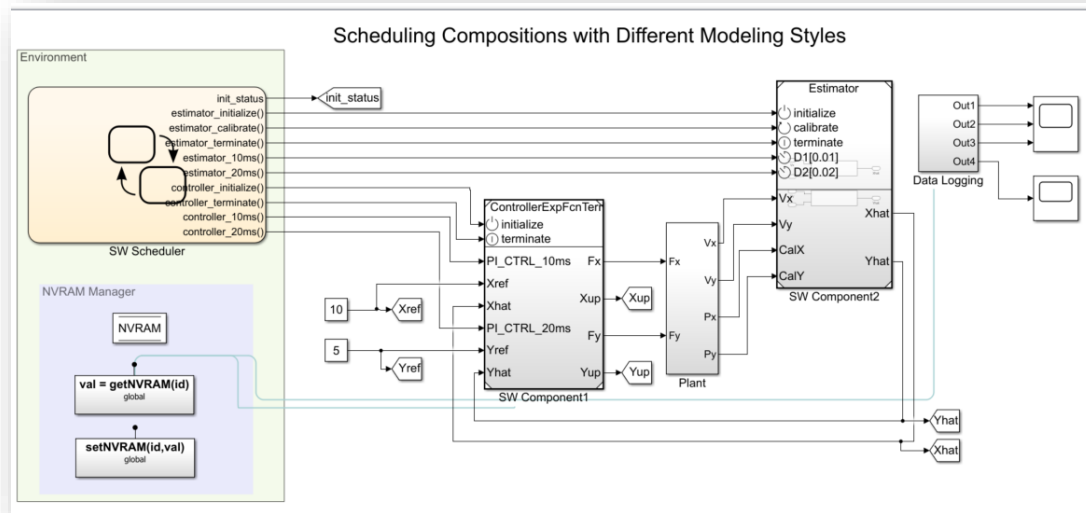
## Implement an embedded PID auto-tuning algorithm

- Automatically tune PID controller gains in real time against a physical plant
- No model of plant dynamics required
- Deploy the auto-tuning algorithm to embedded software using automatic code generation



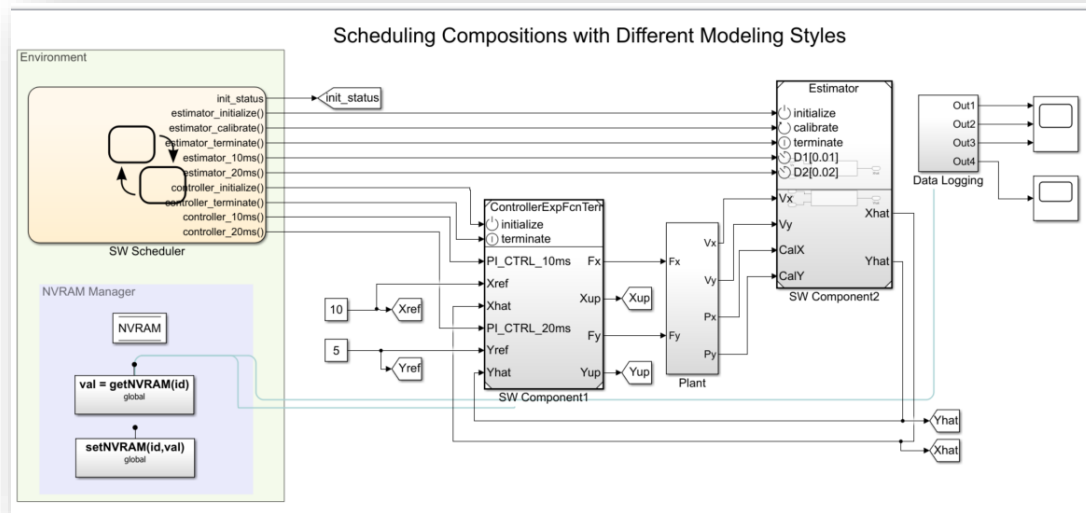
# Prepare Your Model for Code Generation

**Prepare model components  
for code generation**

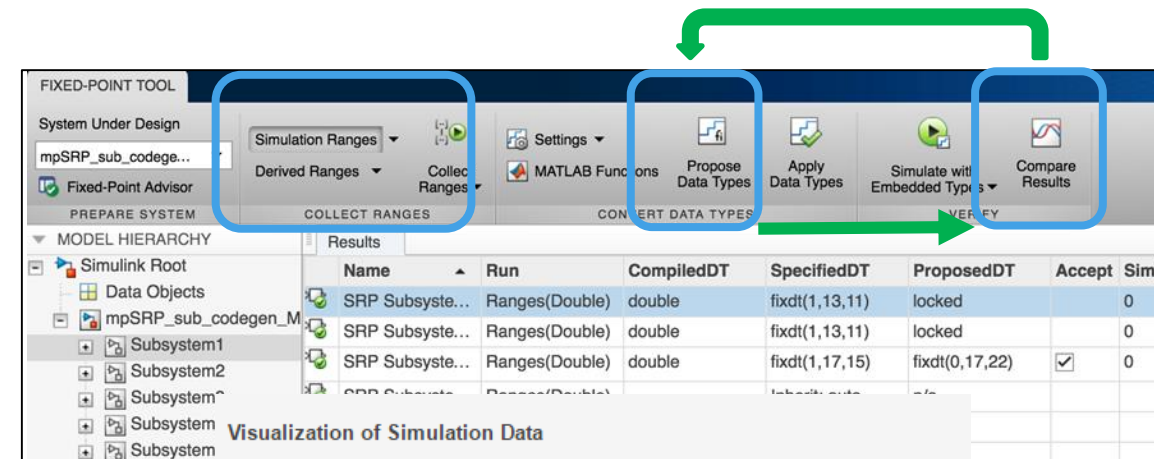


# Prepare Your Model for Code Generation

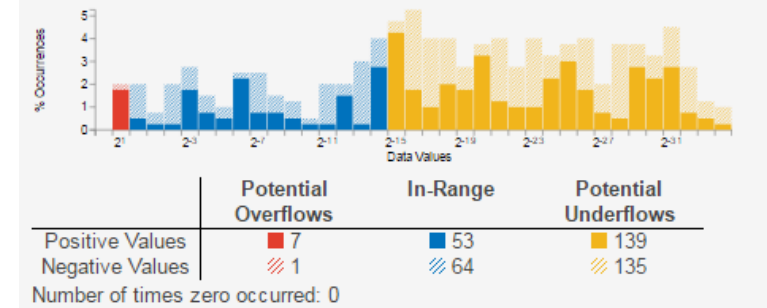
Prepare model components  
for code generation



Prepare model data  
for code generation



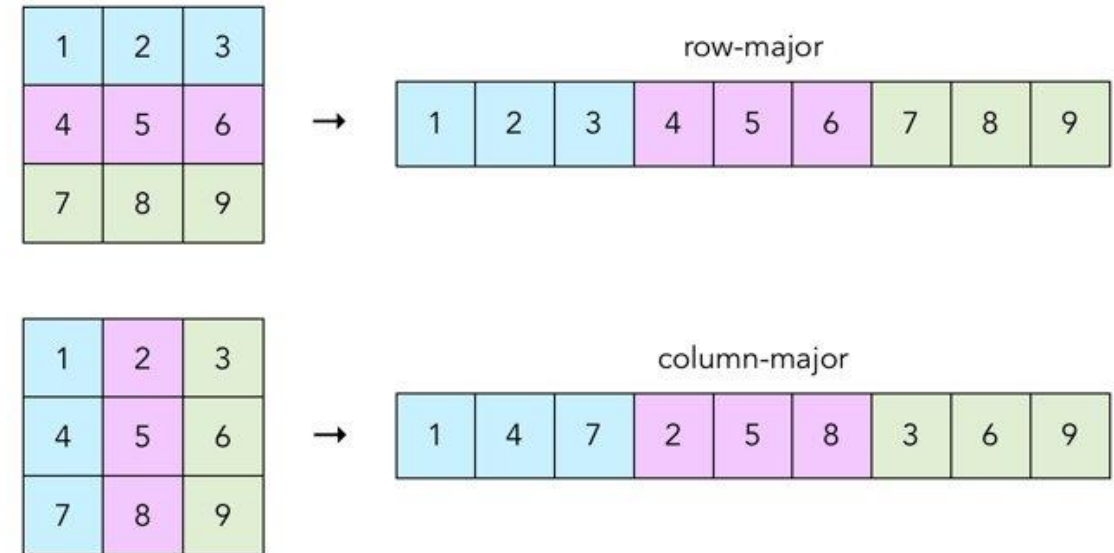
Visualization of Simulation Data



# Generate Code from Simulink Models

**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**

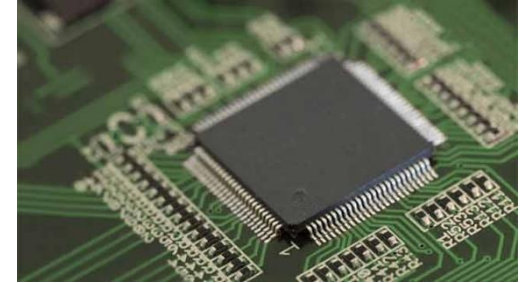
# Connecting Your Design to Hardware

**Connect directly to hardware with support packages**

- Live streaming to and from hardware
- Run Simulink models on low-cost hardware, such as Arduino, Raspberry Pi, and LEGO
- Automatically generate code and run it on microprocessors, FPGAs, and more.



**Arduino**



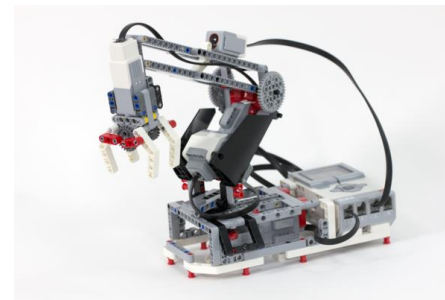
**ARM Cortex**



**Raspberry Pi**



**Microsemi FPGA**



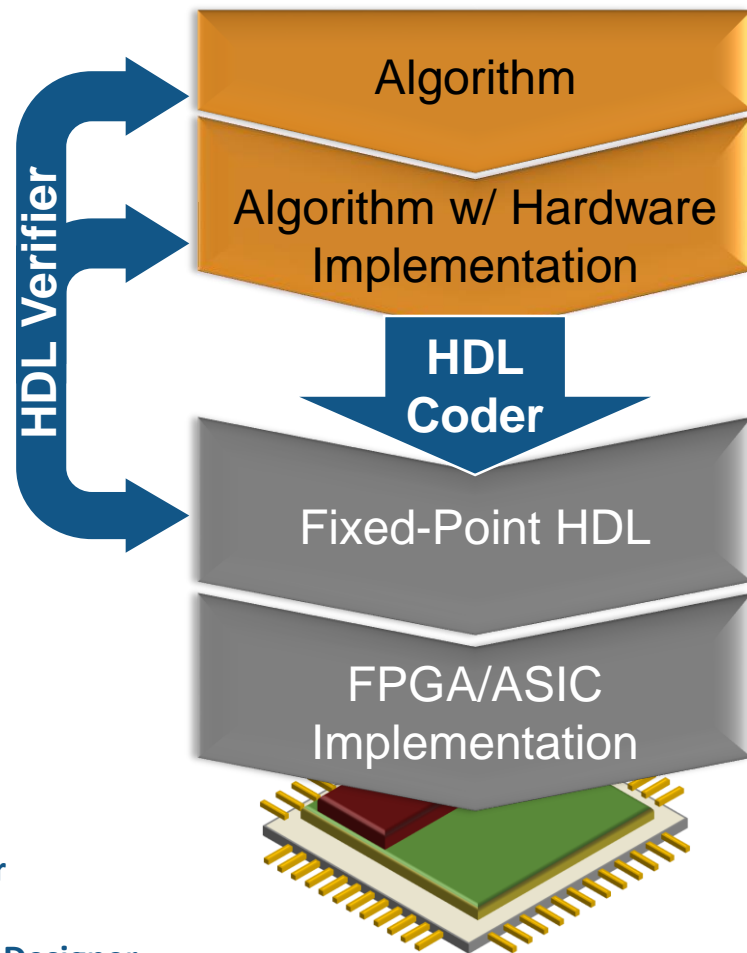
**LEGO**



**ADALM-PLUTO**

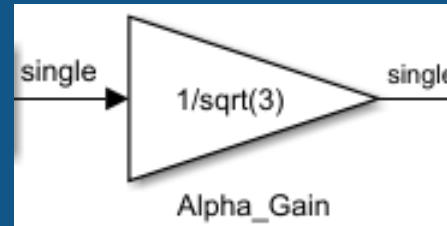


# Deploying to FPGA or ASIC Hardware

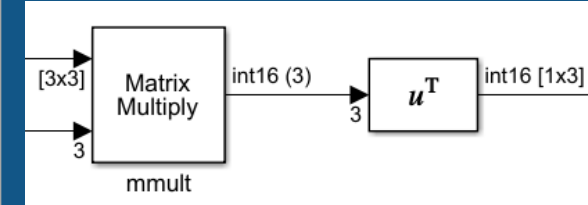


HDL Verifier  
 HDL Coder  
 Fixed-Point Designer  
 Vision HDL Toolbox  
 LTE HDL Toolbox

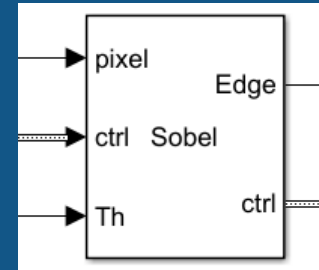
## Native Floating Point



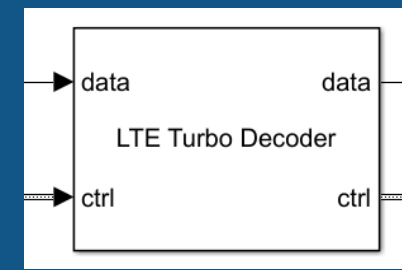
## Matrix Support



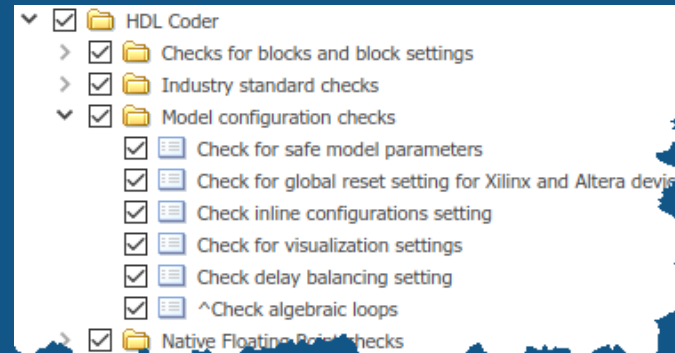
## Vision HDL Toolbox



## LTE HDL Toolbox



## HDL Checks in Model Advisor



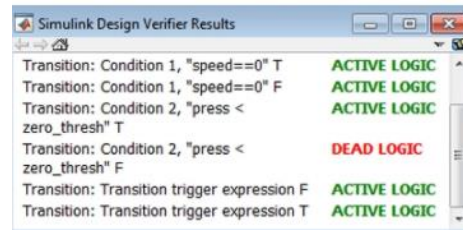
# Verification and Validation

## 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

### Certification

### HIL

### Code inspection

### Test generation

### Code verification

### SIL

### PIL

### Standards checks

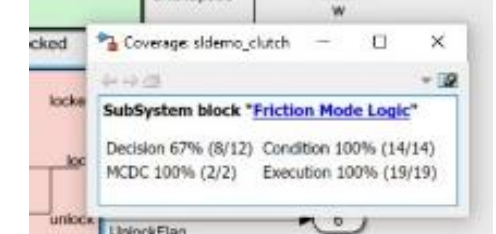
### Property proving

### Simulation

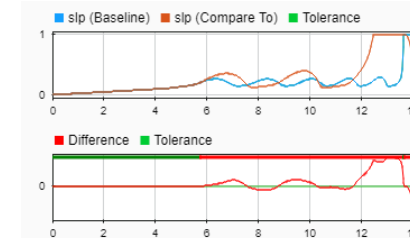
### Requirements tracing



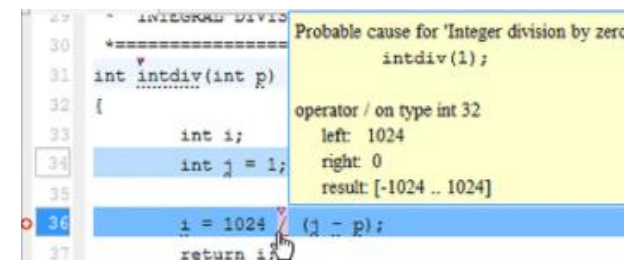
### Simulink Coverage R2017b



### Simulink Test



### Polyspace

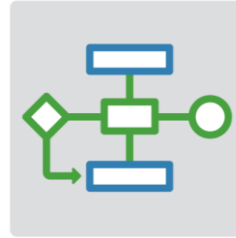


now supports  
**AUTOSAR**  
 R2018a

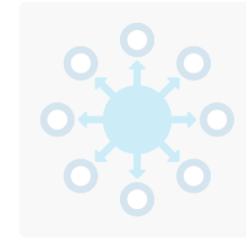
## Platform Productivity



## Workflow Depth



## Application Breadth

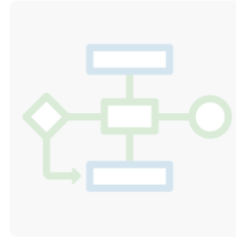


- **Deployment of MATLAB Algorithms and Applications**
- **Code Generation from Simulink Models**
- **Verification and Validation**

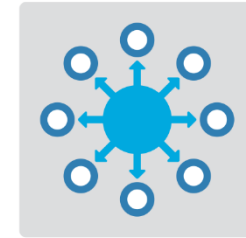
## Platform Productivity



## Workflow Depth

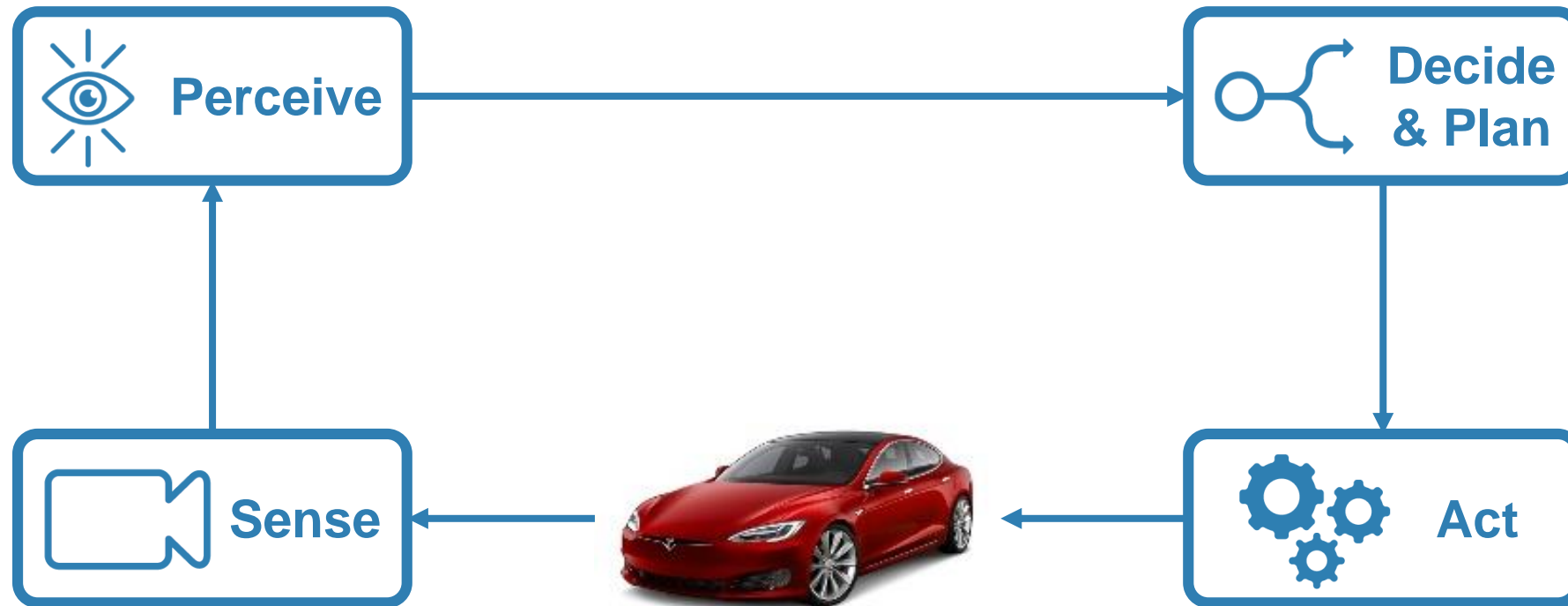


## Application Breadth



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

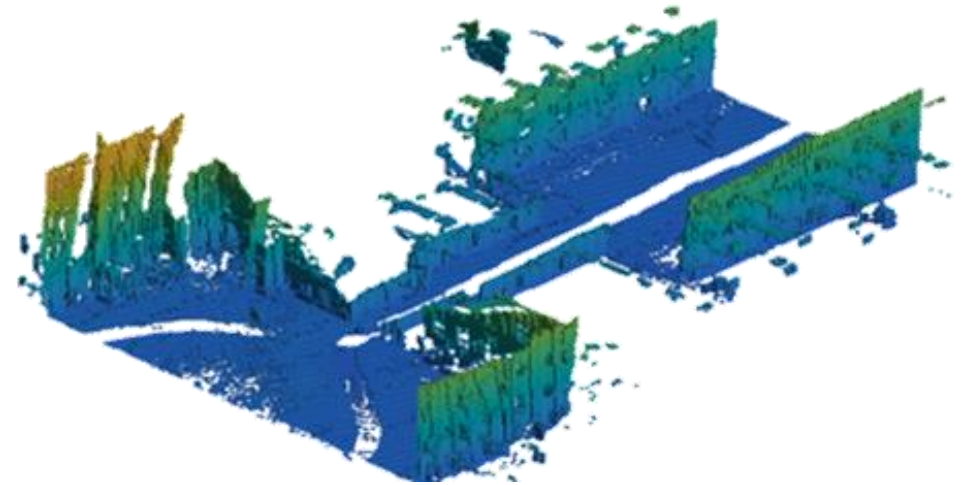
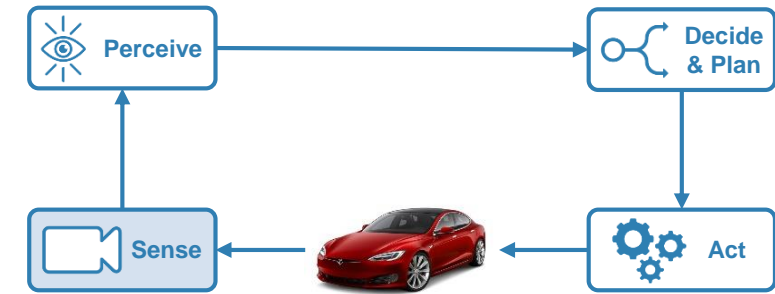
# Designing Autonomous Systems



# Designing Autonomous Systems

## Mapping of environments using sensor data

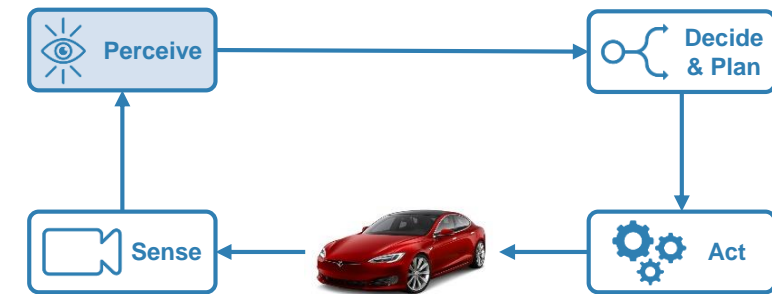
- Segment and register lidar point clouds
- Lidar-Based SLAM: Localize robots and build map environments using lidar sensors



# Designing Autonomous Systems

## Understanding the environment using computer vision and deep learning techniques

- Object detection and tracking
- Semantic segmentation using deep learning

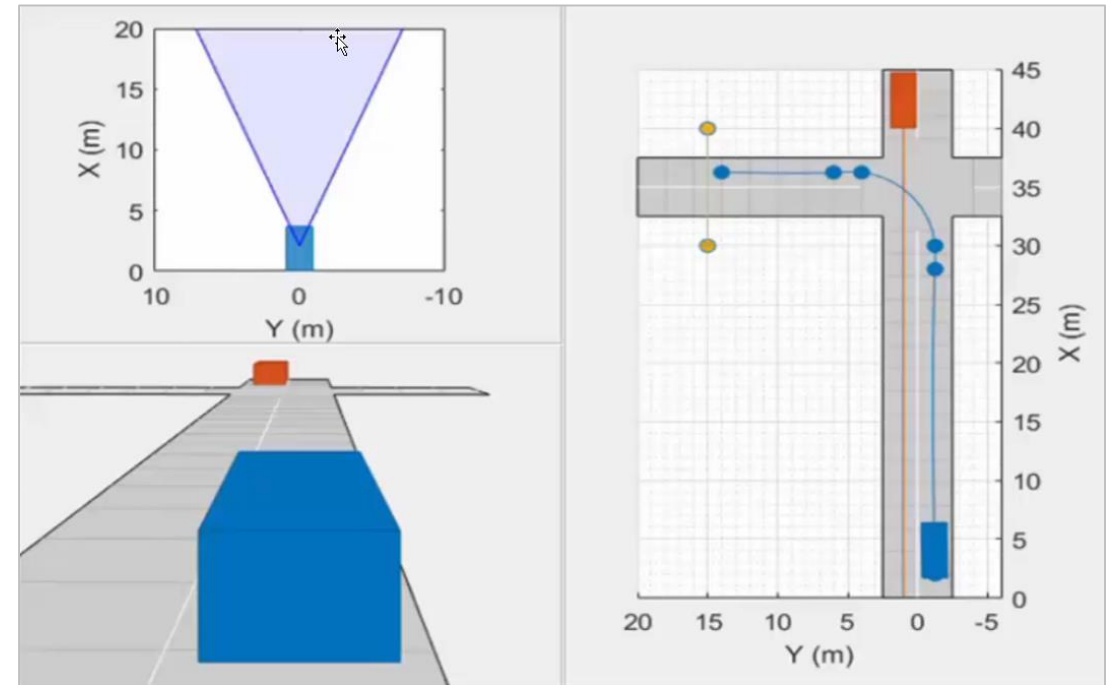
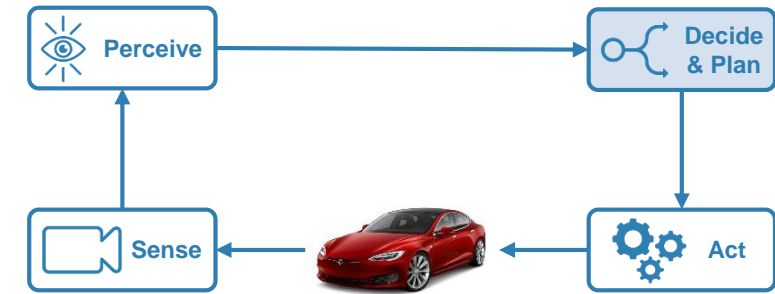




# Designing Autonomous Systems

**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



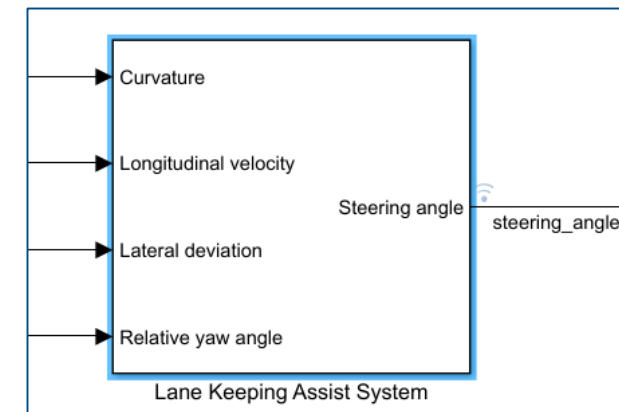
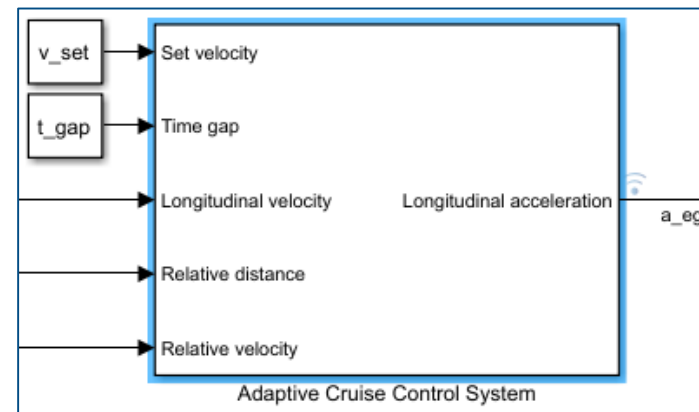
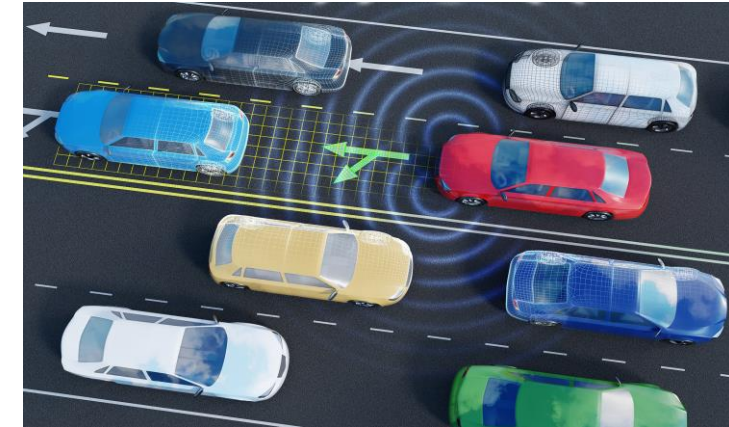
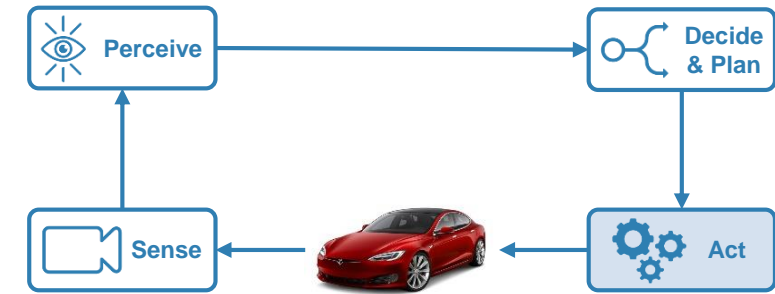
**Driving Scenario Designer App**



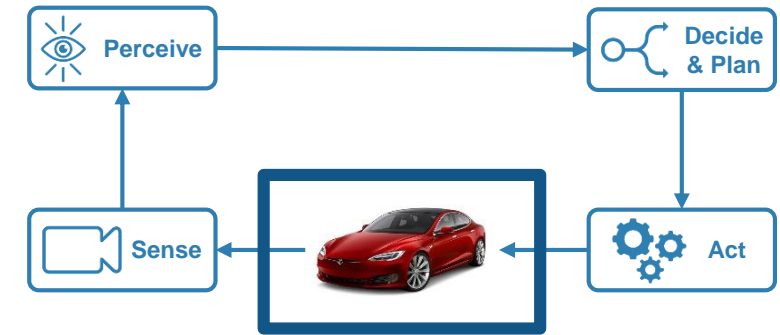
# Designing Autonomous Systems

## 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



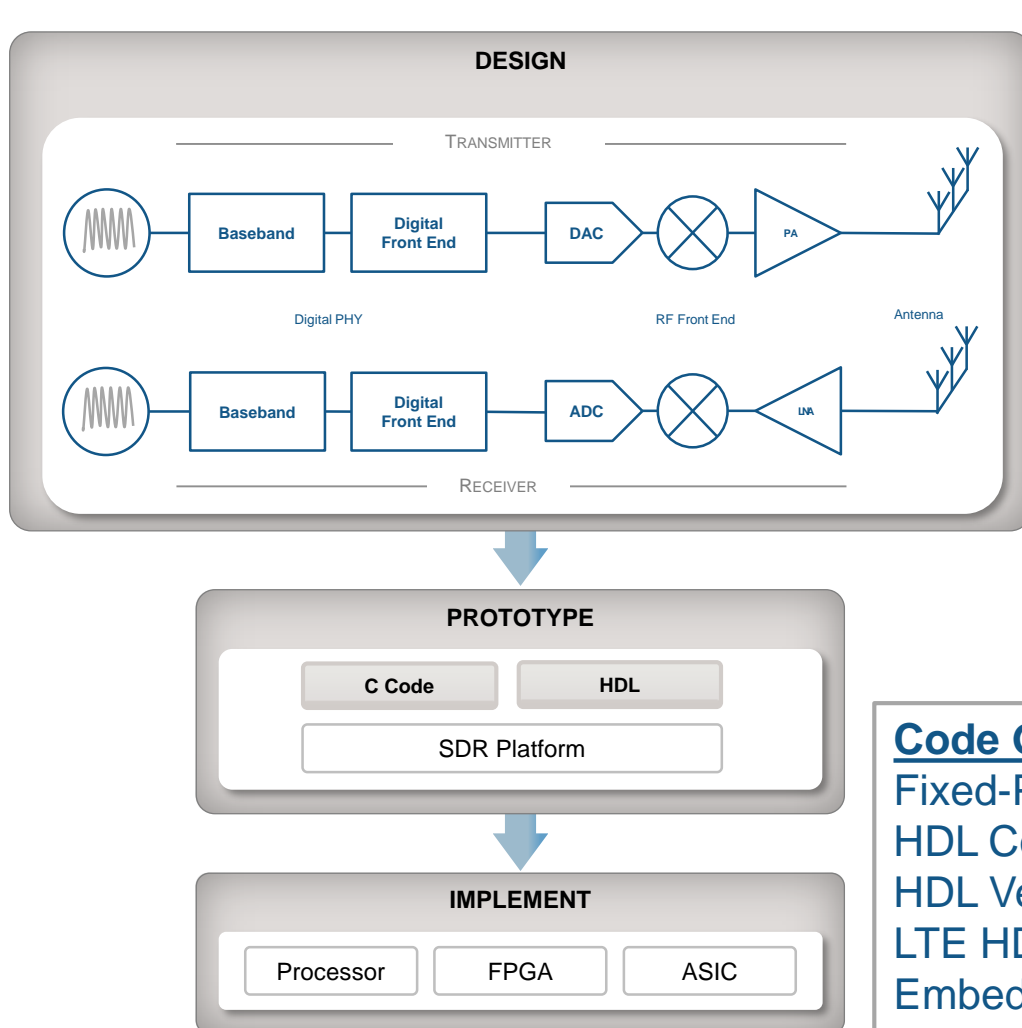
Automated Driving

# Design with the Latest Wireless Standards



**NB-IoT**

# Model-Based Design for Wireless Communications



- Algorithm Design and Verification
- RF, Digital and Antenna Co-Design
- System Verification and Testing
- Rapid Prototyping and Production

## Code Generation and Verification

Fixed-Point Designer

HDL Coder

HDL Verifier

LTE HDL Toolbox **R2017b**

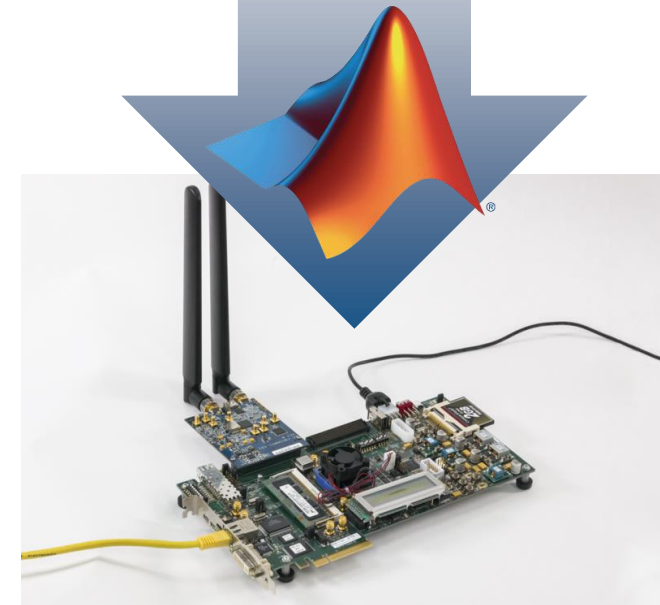
Embedded Coder

# RF and Antenna Design and Prototyping

Use RF and Antenna models through  
your entire development cycle

- RF top-down design with RF Budget Analyzer app
- Adaptive hybrid beamforming and MIMO system modeling
- RF Power Amplifier modeling and DPD linearization
- RF propagation and 3D terrain visualization
- Design and fabrication of printed (PCB) antennas

? ? ? ? ?  
? ? ? ? ?  
*From idea ...*

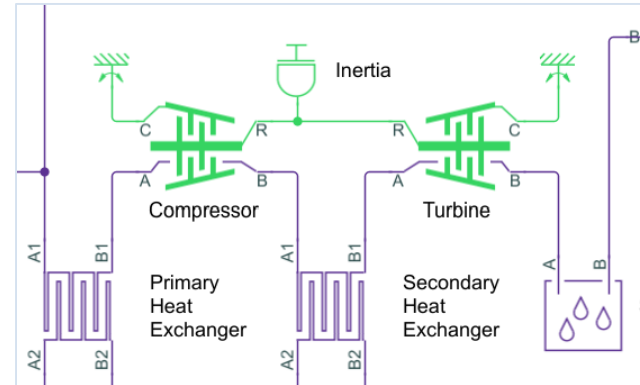


*... to implementation*

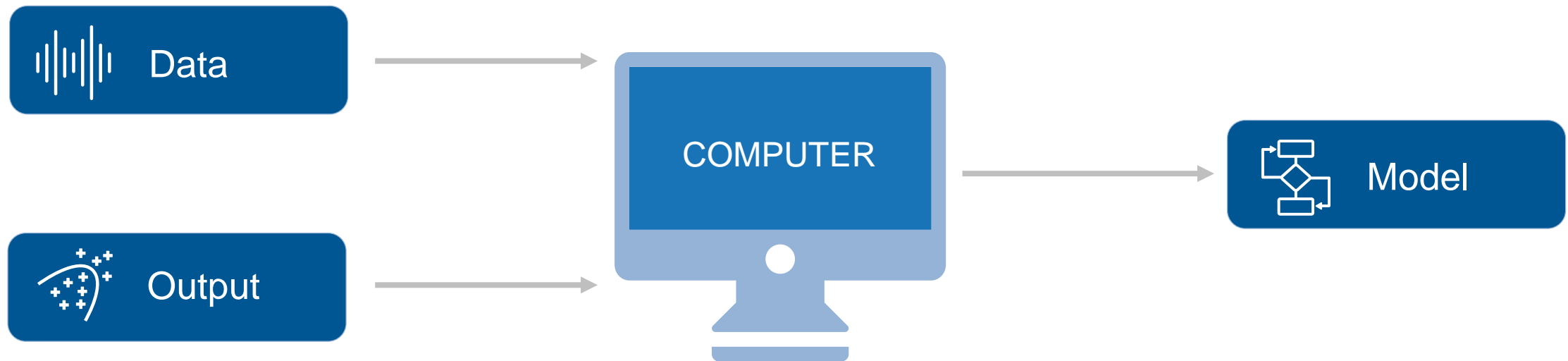
# Model Moist Air Systems

## Model HVAC and environmental control systems

- Model and simulate HVAC systems for a plant, such as a building, automobile, aircraft
- New library contains chambers, reservoirs, local restrictions, energy converters, sources and sensors
- Ensure acceptable temperature, pressure, humidity, condensation within the environment
- Note for Simscape in general: Run simulations about 5x faster with local solver option



# Artificial Intelligence





# Text Analytics

 Data

```
repairNotes = 617x1 string array  
"PM SERVICE, CHECK TURN SIGNAL, CLUNKING NOISE"  
"SERVICEROB,EXT,5604"  
"NEED 4 PLOW PINS"  
"INSTALL SPINNER ASSY"  
"DONT START"  
"DOG BONE PIN BROKEN"  
"NEED SERVICE, CHECK BRAKES"  
"HYD CAP CHECK ENGINE LIGHT ON"  
"TARP VALVE STICKINGRIGHT SIDE MIRROR BRACKET"  
"HANDLES IN CAB LOOSE"  
"NO PLOW LIGHTS"  
"WILL NOT START"
```

 Output



 Model

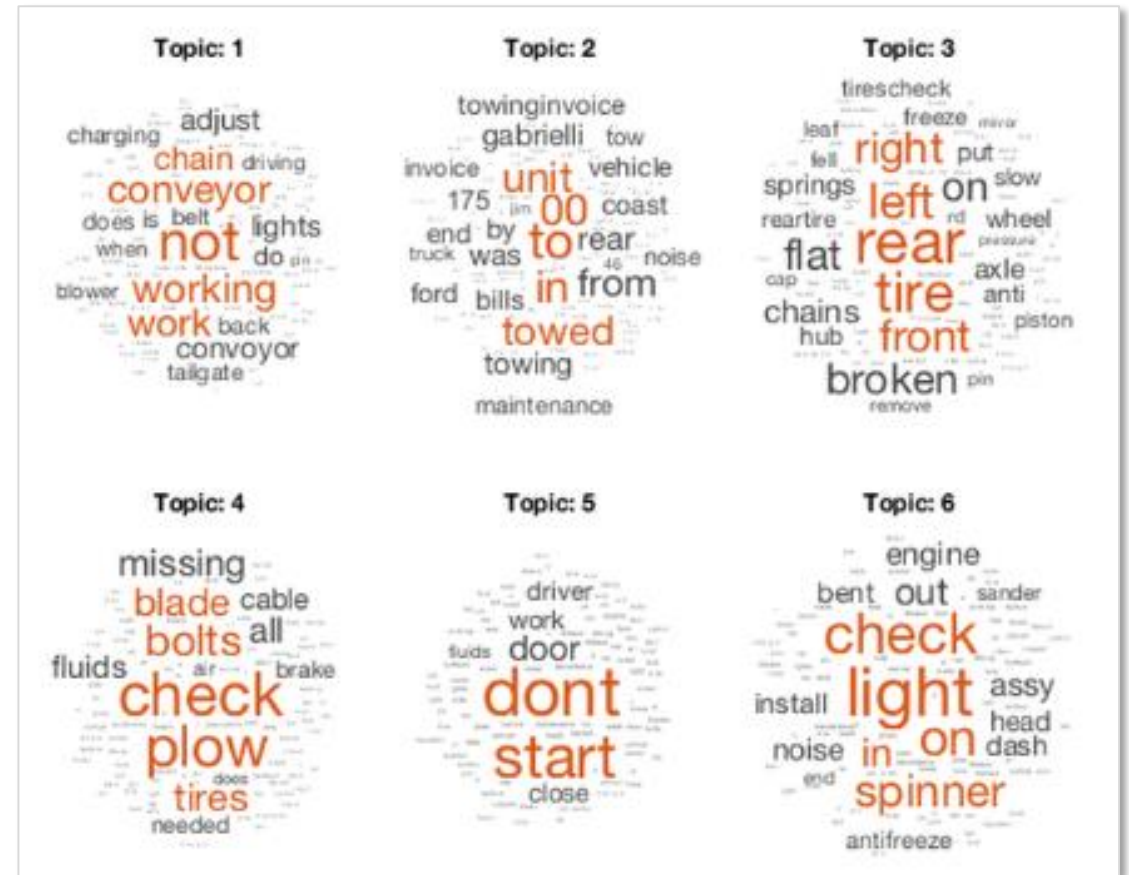




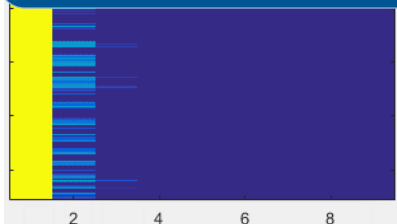
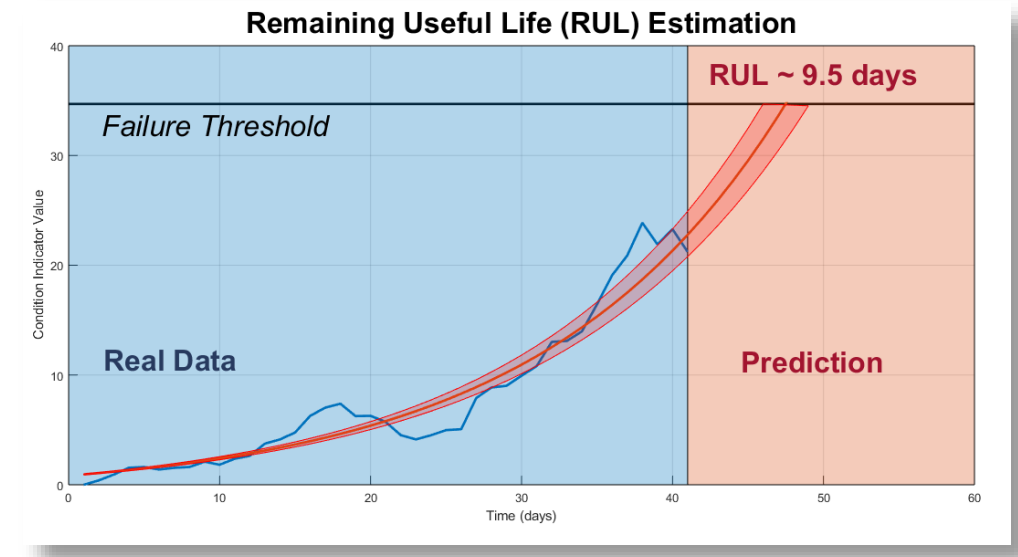
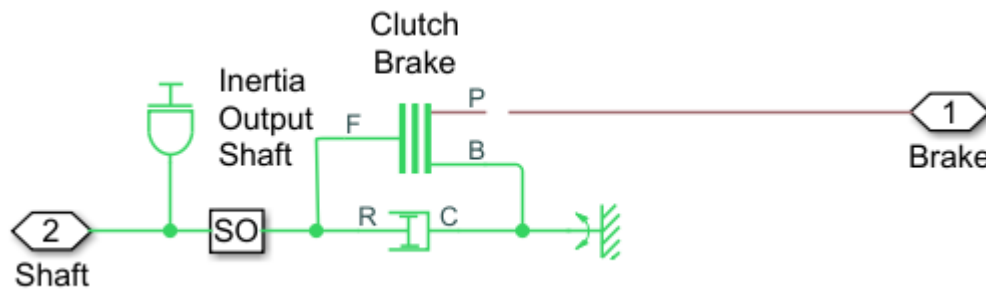
# Text Analytics

## Work with text from equipment logs and operator reports

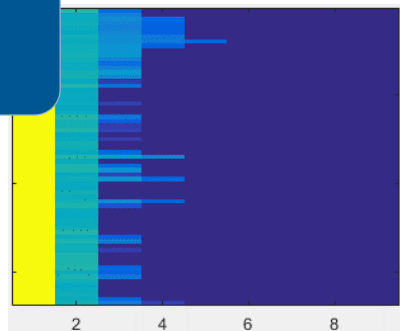
- **Preprocess** raw text data by extracting, filtering, and splitting
- **Visualize** text using word clouds and text scatter plots
- **Develop** predictive models using built-in machine learning algorithms (LDA, LSA, word2vec)



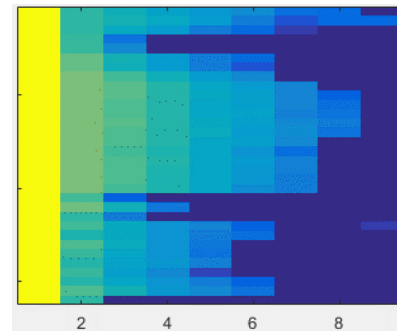
# Predictive Maintenance



Normal Operation



Monitor Closely

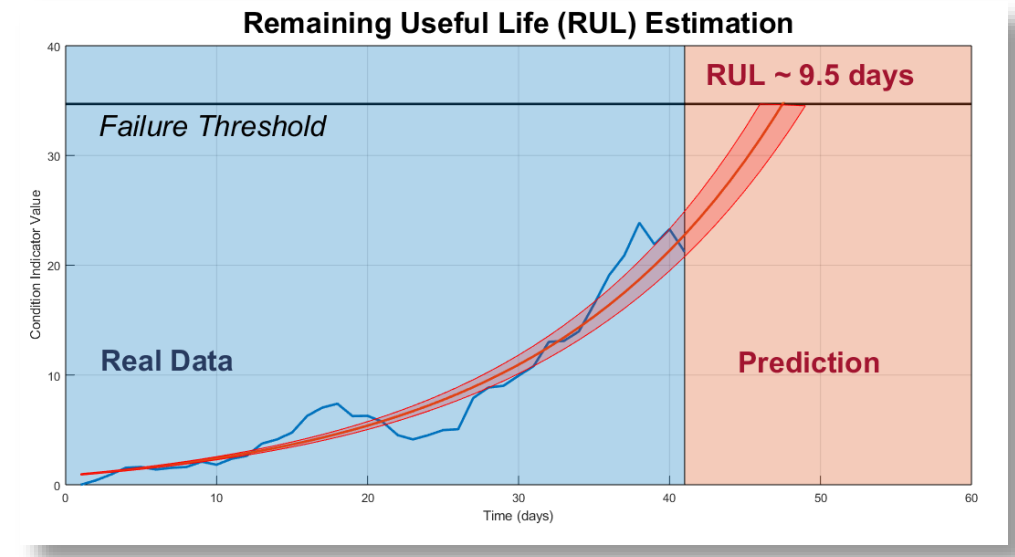


Maintenance Needed

# 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
- Estimate remaining useful life (RUL)
- Get started with examples (*motors, gearboxes, batteries, and other machines*)



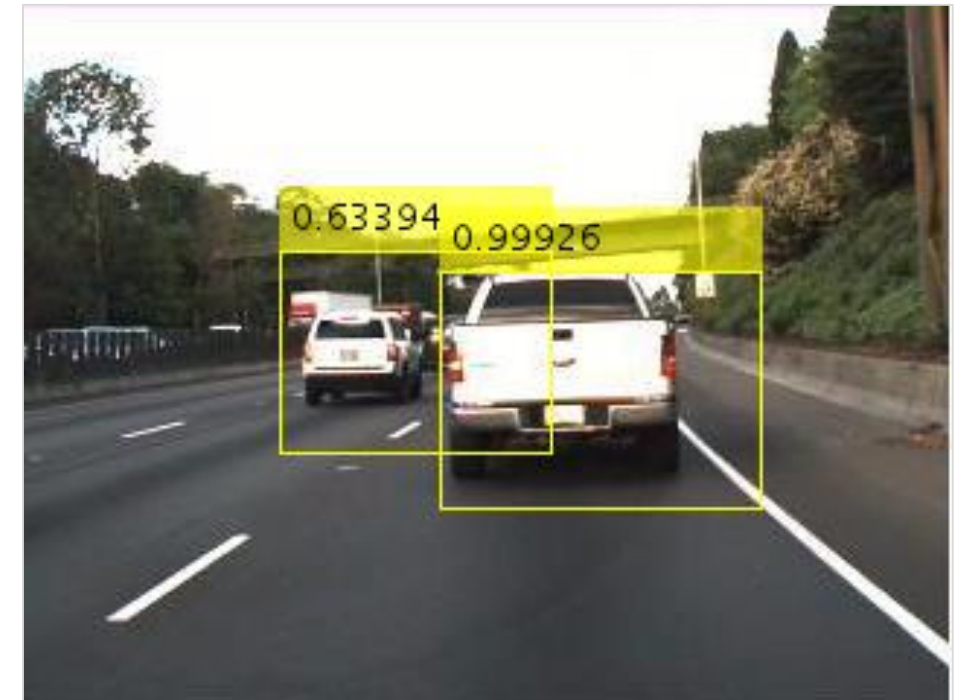
# Deep Learning

 Data



 Model

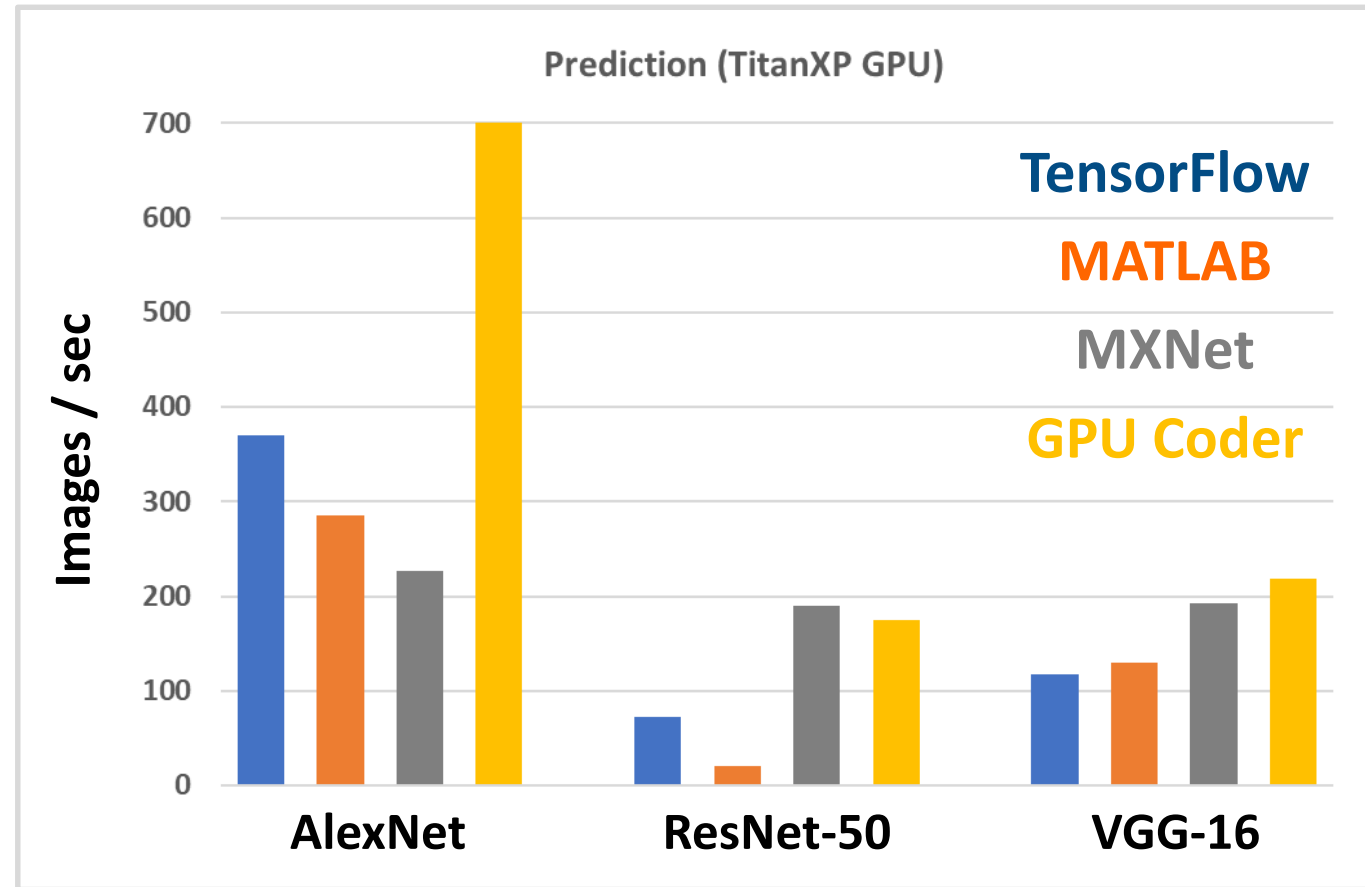
 Output



# Deep Learning

## Design, build, and visualize convolutional 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 bar shows "Deep Learning Onramp" with a progress indicator at 51% complete. The left sidebar lists "Task 1", "Task 2", and "Task 3". The main content area is titled "Classify images" and contains the following tasks:

- 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 right sidebar shows the workspace with the variable `pred1` assigned the value `categorical seashore`. The bottom of the interface shows a command window.

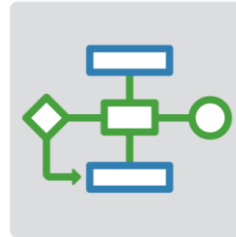
# What's New in MATLAB and Simulink?

## Platform Productivity



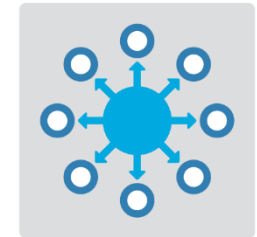
- Design Creation
- Analysis
- Simulation, Scaling
- Collaboration

## Workflow Depth



- Deployment
- Code Generation
- Verification and Validation

## Application Breadth



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



# 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
3	classifyBloodPressure.m	24	TREEVAL has been removed

**Warnings and Other Recommendations**

Row	Filename	Line	Description
1	classifyBloodPressure.m	1	RAND or RANDN with t recommended. Use RN

Upgrade Advisor - sf\_climate\_control

File Edit Run Settings Help

Find:  [Disable Upgrade Notifications](#)

**Upgrade Project Report**

100% Passed

Passed Need attention

Models	Libraries	MATLAB Code
7	1	8
-	-	-

Show: [All Files](#) [All Results](#)

AnalogControl.mdl

analyzeModelFiles.m

billOfMaterials.m

checkCodeProblems.m

DigitalControl.slx

f14\_airframe.slx

f14\_airframe\_test.m

find\_top\_models.m

LinearActuator.slx

NonLinearActuator.mdl

rebuild\_s\_functions.m

runUnitTest.m

slproject\_f14.slx

upgrade\_project.m

vertical\_channel.slx

wind\_gust\_lib.slx

Check Name

Check model settings for migration to simplified initialization mode

Check that the model is saved in SLX format

Check usage of function-call connections

Check and set embedded target model to use ert.tlc system target file

Check and update masked blocks in library to use promoted parameters

Check and update mask image display commands with unnecessary imread() function calls

Check and update mask to affirm icon drawing commands dependency on mask workspace

Check and update model to use toolchain approach to build generated code

[Show All](#)

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

[Publish Report](#) [Close](#)

**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.

[Run This Check](#)

Result: [Passed](#)

Identify Variant Model blocks at model level.

**Passed**

No Variant Model blocks found.



MATLAB EXPO 2018

