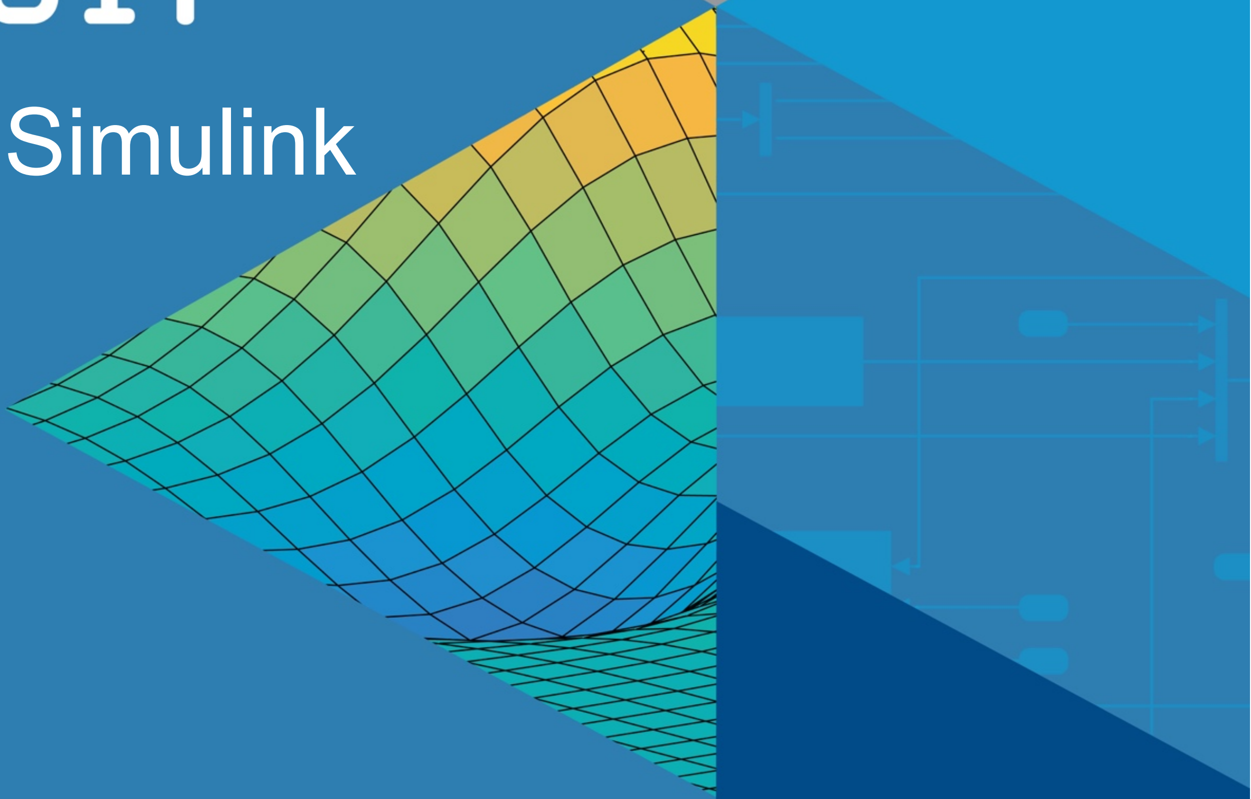
MATLAB EXPO 2017

What’s New in MATLAB and Simulink

R2017a R2016b

Dr. Marc Segelken



Platform Productivity

Getting your work done faster

Workflow Depth

Support for your entire workflow

Application Breadth

Products for the work you do

Platform Productivity

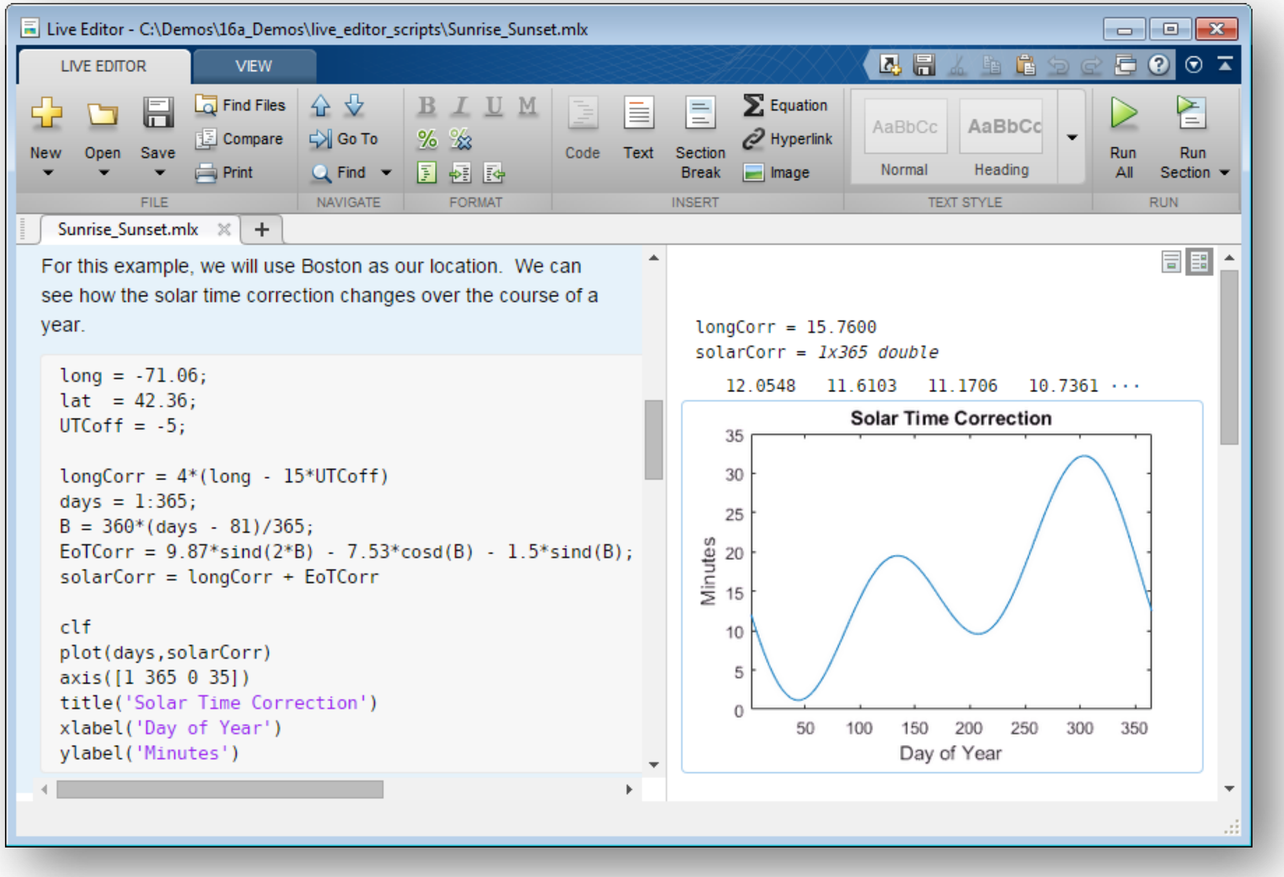
Getting your work done faster

Change the Way You Work in MATLAB

R2016b R2017a

See results together with your MATLAB code in the Live Editor ***(introduced in R2016a)***

- Add equations, images, hyperlinks, and formatted text
- Present, share, and collaborate using interactive documents
- Interactive figure updates
  - Pan , zoom, and rotate axes
  - Interactive plot customization, with MATLAB code generation to automate work
- Interactive equation editor



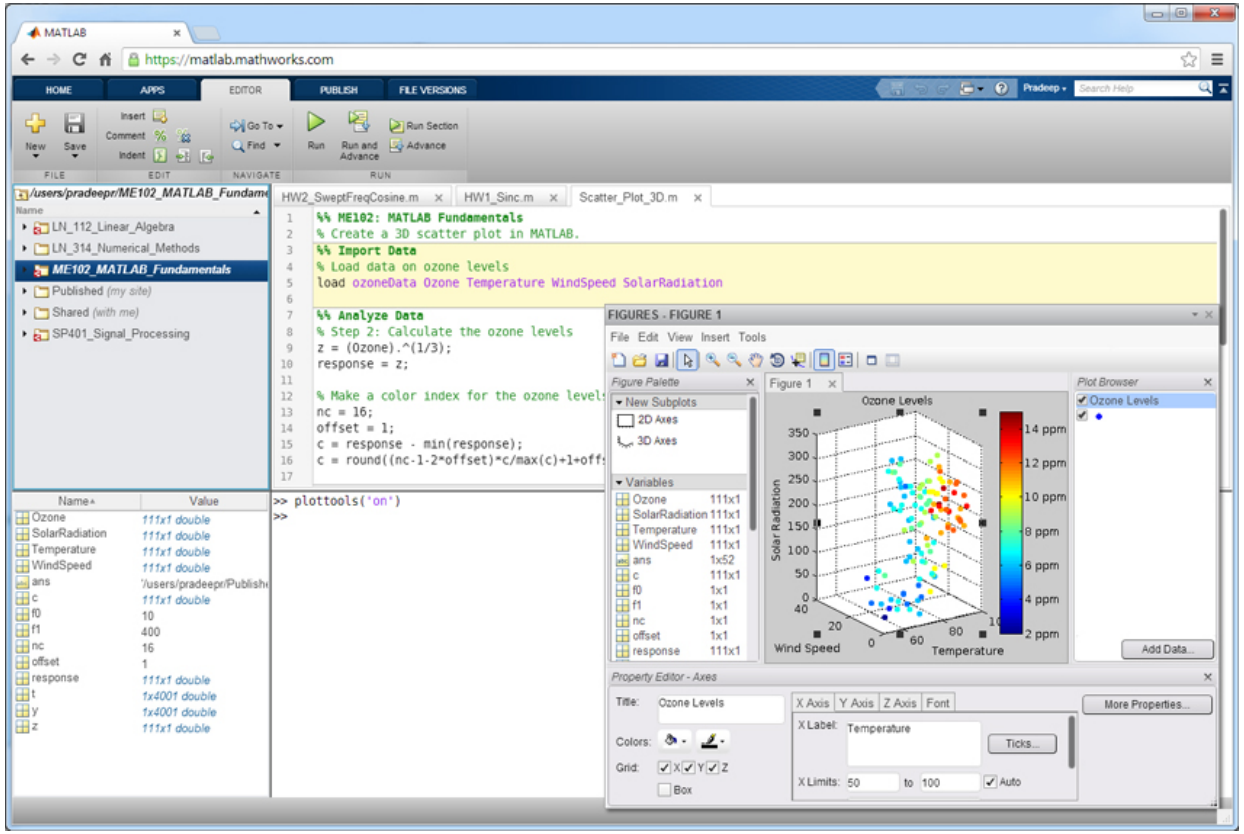
For this example, we will use Boston as our location. We can see how the solar time correction changes over the course of a year.

```
long = -71.06;
lat  = 42.36;
UTCoff = -5;

longCorr = 4*(long - 15*UTCoff)
days = 1:365;
B = 360*(days - 81)/365;
EoTCorr = 9.87*sind(2*B) - 7.53*cosd(B) - 1.5*sind(B);
solarCorr = longCorr + EoTCorr

clf
plot(days,solarCorr)
axis([1 365 0 35])
title('Solar Time Correction')
xlabel('Day of Year')
ylabel('Minutes')
```

```
longCorr = 15.7600
solarCorr = 1x365 double
   12.0548   11.6103   11.1706   10.7361 ...
```

MATLAB Online

- Provides access to MATLAB desktop and full MATLAB language support from any standard web browser
- No downloads or installs.
- Cloud Storage and synchronization via MATLAB Drive
- Log in here with your MathWorks Account: https://matlab.mathworks.com/



```
%% ME102: MATLAB Fundamentals
% Create a 3D scatter plot in MATLAB.
%% Import Data
% Load data on ozone levels
load ozoneData Ozone Temperature WindSpeed SolarRadiation

%% Analyze Data
% Step 2: Calculate the ozone levels
z = (Ozone).^(1/3);
response = z;

% Make a color index for the ozone level
nc = 16;
offset = 1;
c = response - min(response);
c = round((nc-1-2*offset)*c/max(c)+1+off
```

```
>> plottools('on')
>>
```

Name	Value
Ozone	111x1 double
SolarRadiation	111x1 double
Temperature	111x1 double
WindSpeed	111x1 double
ans	'/users/pradeepr/Publishe
c	111x1 double
f0	10
f1	400
nc	16
offset	1
response	111x1 double
t	1x4001 double
y	1x4001 double
z	111x1 double

Ozone	111x1	
SolarRadiation	111x1	
Temperature	111x1	
WindSpeed	111x1	
ans	1x52	
c	111x1	
f0	1x1	
f1	1x1	
nc	1x1	
offset	1x1	
response	111x1	

App Designer

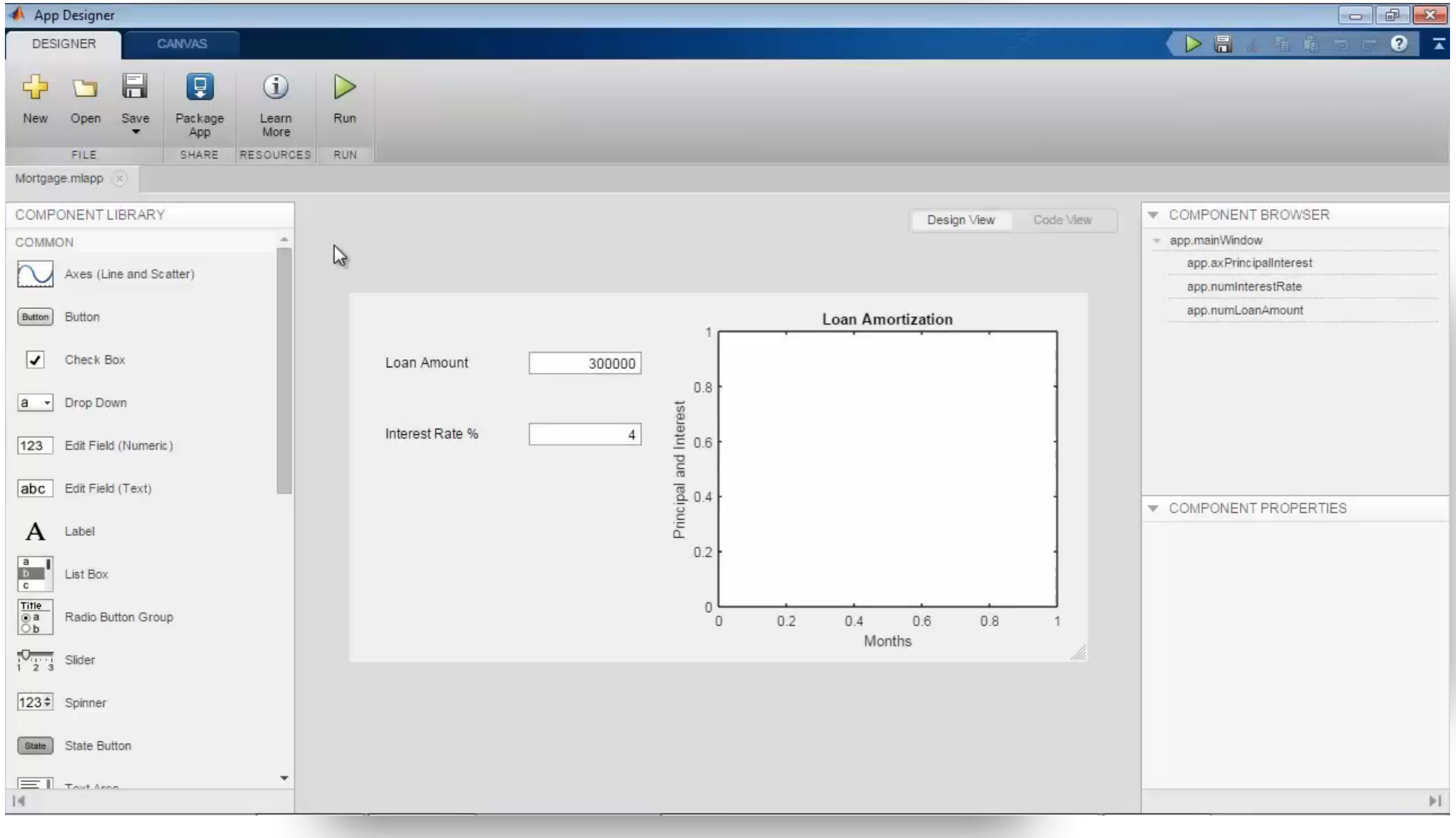
R2016b R2017a

Environment for building MATLAB apps

(introduced in R2016a)

- Full set of standard user interface components, as well as gauges, knobs, switches, and lamps
- Rich design environment for laying out apps
- Object-based code format for easily sharing data between parts of the app

- Enhancements include:
  - Majority of 2-D plots supported
  - Embed tabular displays using `uitable`
  - Zoom and pan plots in apps



Loan Amount 300000

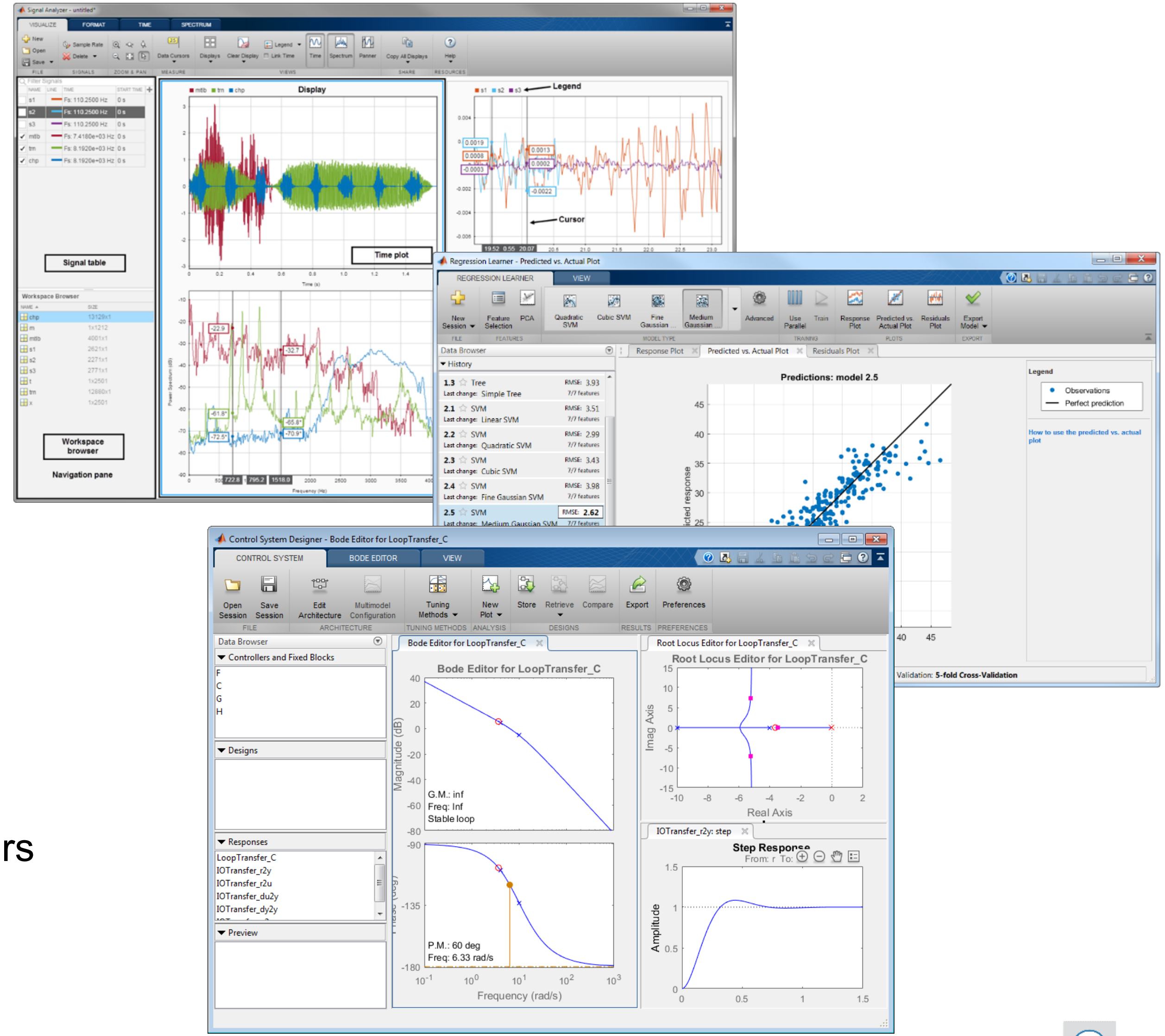
Interest Rate % 4

Apps Simplify Modeling and Simulation

R2016a R2016b R2017a

These interactive applications automate common technical computing tasks

- Signal Analyzer app
  - Perform time- and frequency-domain analysis of multiple time series
- Regression Learner app
  - Train regression models using supervised machine learning
- Control System Designer app
  - Design single-input, single-output (SISO) controllers



1.3 Tree RMSE: 3.93
Last change: Simple Tree 7/7 features

2.1 SVM RMSE: 3.51
Last change: Linear SVM 7/7 features

2.2 SVM RMSE: 2.99
Last change: Quadratic SVM 7/7 features

2.3 SVM RMSE: 3.43
Last change: Cubic SVM 7/7 features

2.4 SVM RMSE: 3.98
Last change: Fine Gaussian SVM 7/7 features

2.5 SVM RMSE: 2.62
Last change: Medium Gaussian SVM 7/7 features

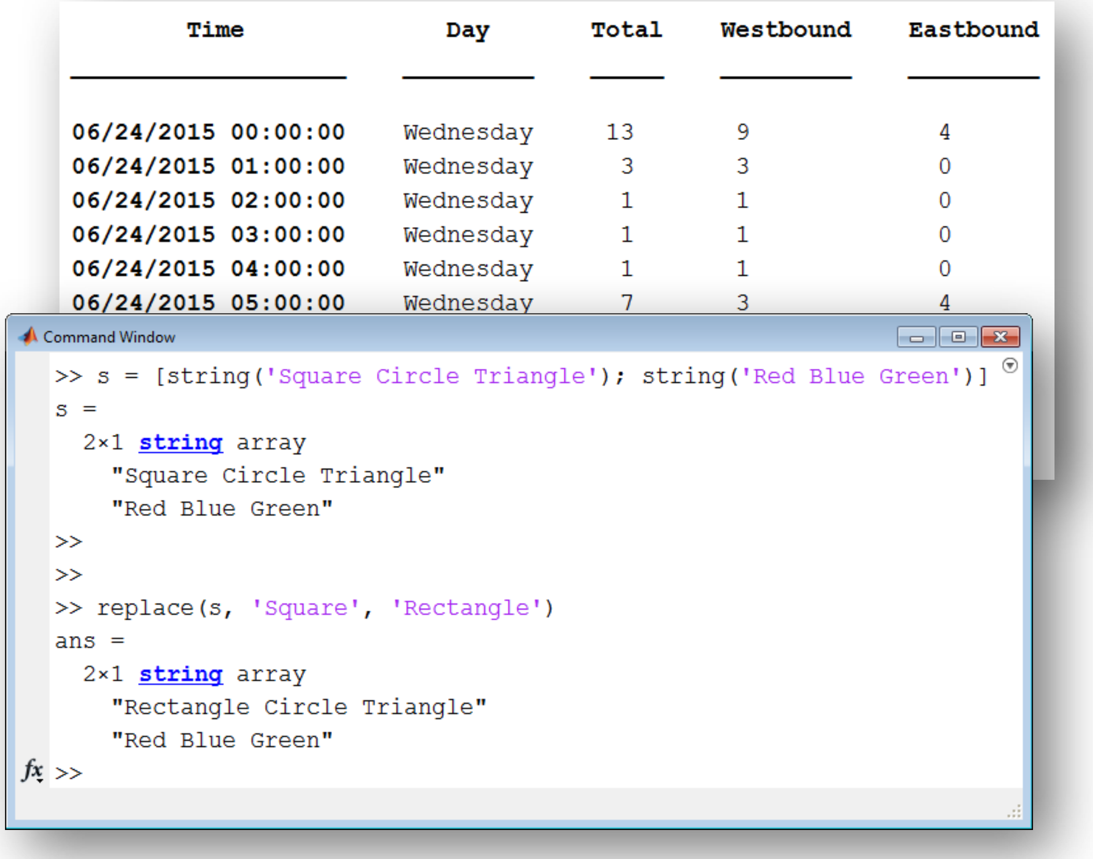
Signal Processing Toolbox
Statistics and Machine Learning Toolbox
Control System Toolbox

Working with Data Just Got Easier

R2016b R2017a

New data types and functionality for more efficient storage and managing of data

- `timetable` data container *(introduced in R2016b)*
  - Store time-stamped tabular data
  - Reorganize, evenly space, and align data
- `string` arrays *(introduced in R2016b)*
  - Memory efficient, faster string operations
  - New functions for common string manipulation
- New capabilities for **preprocessing** data
  - Find, fill, and remove missing data
  - Detect and replace outliers
  - Smooth noisy data



```
        Time                 Day        Total    Westbound    Eastbound
___________________    _________    _____    _________    _________

06/24/2015 00:00:00    Wednesday     13          9            4
06/24/2015 01:00:00    Wednesday      3          3            0
06/24/2015 02:00:00    Wednesday      1          1            0
06/24/2015 03:00:00    Wednesday      1          1            0
06/24/2015 04:00:00    Wednesday      1          1            0
06/24/2015 05:00:00    Wednesday      7          3            4
```

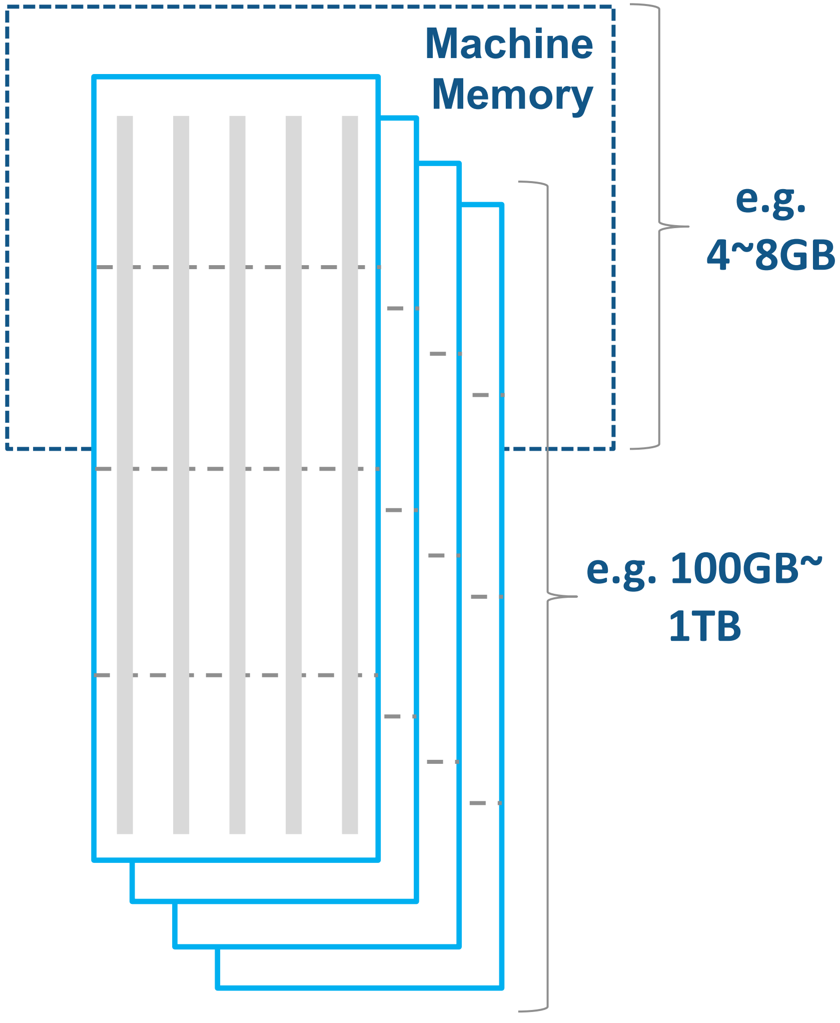
```
>> s = [string('Square Circle Triangle'); string('Red Blue Green')]
s =
  2×1 string array
    "Square Circle Triangle"
    "Red Blue Green"
>>
>>
>> replace(s, 'Square', 'Rectangle')
ans =
  2×1 string array
    "Rectangle Circle Triangle"
    "Red Blue Green"
fx >>
```

Working with Big Data Just Got Easier

R2016b R2017a

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

- Tall arrays let you use familiar MATLAB functions and syntax to work with big datasets, even if they don't fit in memory
- Support for hundreds of functions in MATLAB and Statistics and Machine Learning Toolbox
- Works with Spark + Hadoop Clusters



Tall Data

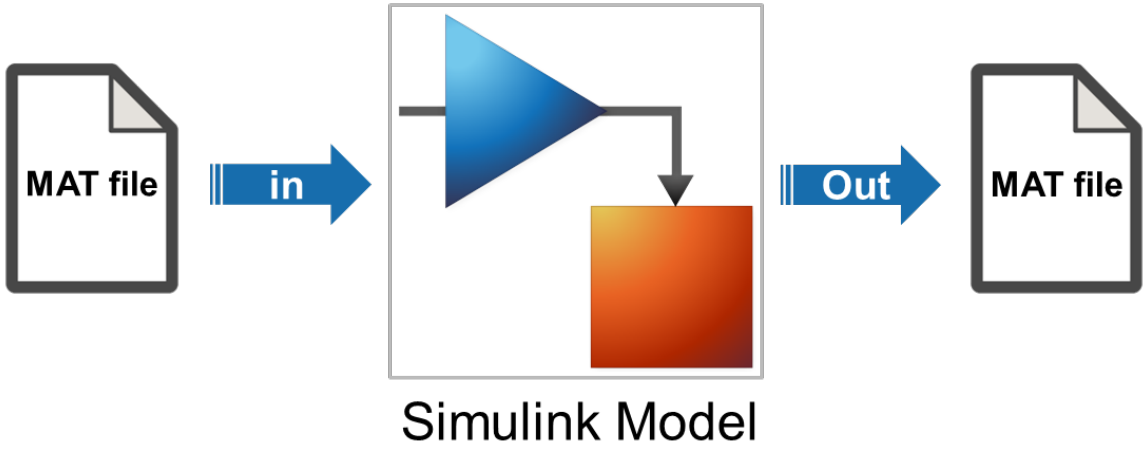
Learn more at this session:
Big Data and Machine Learning

Working with Big Data Just Got Easier in Simulink Too

R2017a

Stream large input signals from MAT-files without loading the data into memory

- Provides a big data workflow for Simulink simulations
- Use big data in Simulink logging and loading
- Especially useful when running many simulations where data retrieved is too large to fit into memory

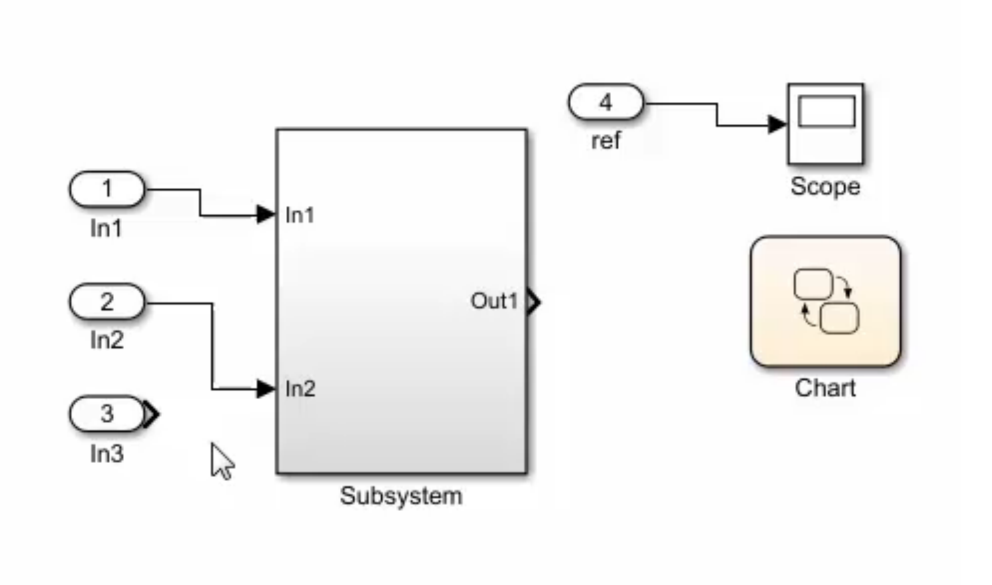


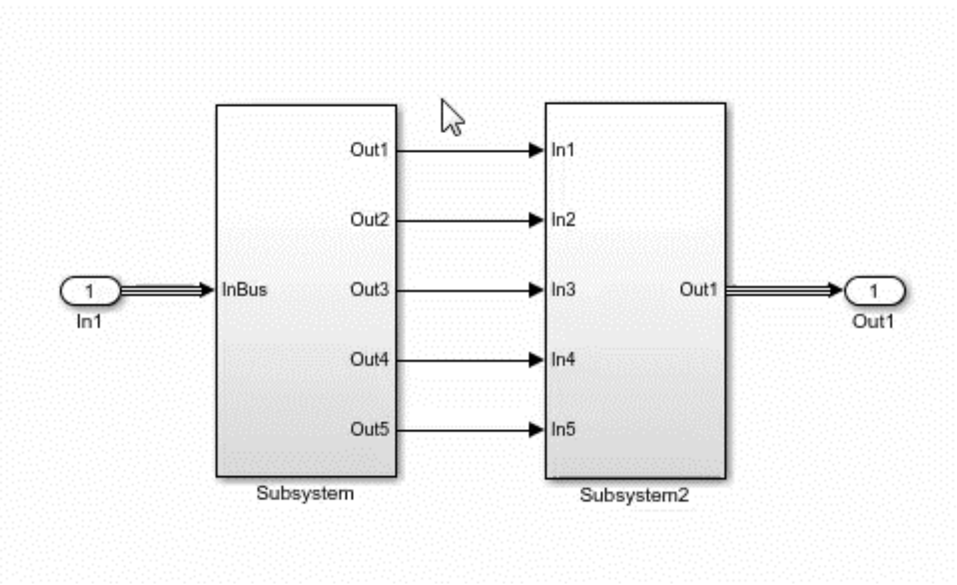
Learn more at this session:
Simulink as Your Enterprise Simulation Platform

Create Your Models Faster

Use automatic port creation and reduced bus wiring

- Add inports and outports to blocks when routing signals
- Quickly group signals as buses and automatically create bus element ports for fewer signal lines





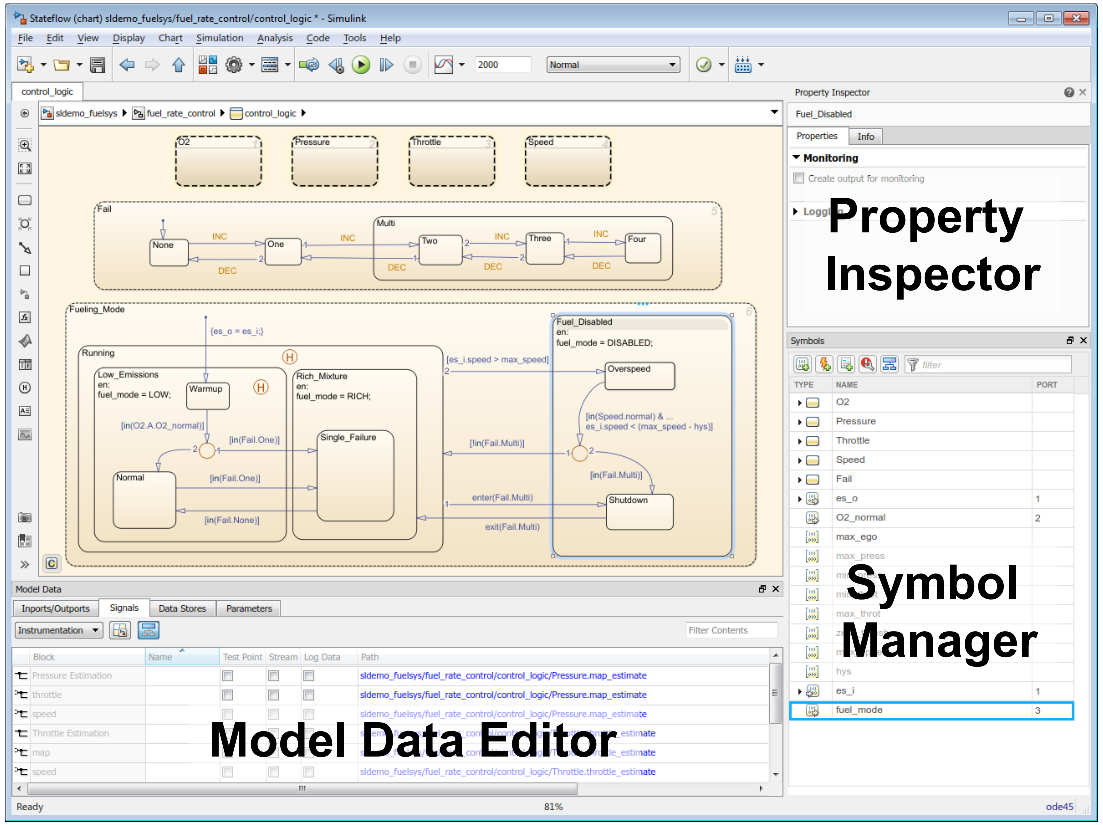
Define your Data Faster

R2016b

Reduces the need to open separate dialog boxes

- Model and block parameter data is now accessible within the main editor window

- Accessing and defining Stateflow data is also much easier



Property Inspector

Fuel_Disabled

Properties | Info

Monitoring

□ Create output for monitoring

Logging

Symbols

filter

TYPE	NAME	PORT
	O2	
	Pressure	
	Throttle	
	Speed	
	Fail	
	es_o	1
	O2_normal	2
	max_ego	
	max_press	
	min_press	
	min_throt	
	max_throt	
	zero_thresh	
	max_speed	
	hys	
	es_i	1
	fuel_mode	3

Model Data

Inports/Outports | Signals | Data Stores | Parameters

Instrumentation

Filter Contents

Block	Name	Test Point	Stream	Log Data	Path
Pressure Estimation		□	□	□	sldemo_fuelsys/fuel_rate_control/control_logic/Pressure.map_estimate
throttle		□	□	□	sldemo_fuelsys/fuel_rate_control/control_logic/Pressure.map_estimate
speed		□	□	□	sldemo_fuelsys/fuel_rate_control/control_logic/Pressure.map_estimate
Throttle Estimation		□	□	□	sldemo_fuelsys/fuel_rate_control/control_logic/Throttle.throttle_estimate
map		□	□	□	sldemo_fuelsys/fuel_rate_control/control_logic/Throttle.throttle_estimate
speed		□	□	□	sldemo_fuelsys/fuel_rate_control/control_logic/Throttle.throttle_estimate

Property Inspector

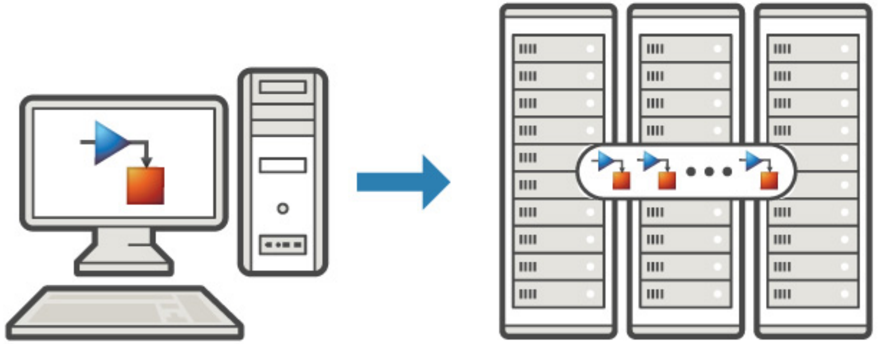
Symbol Manager

Model Data Editor

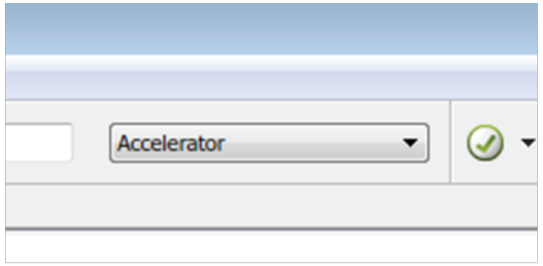
Simulate your Model Faster

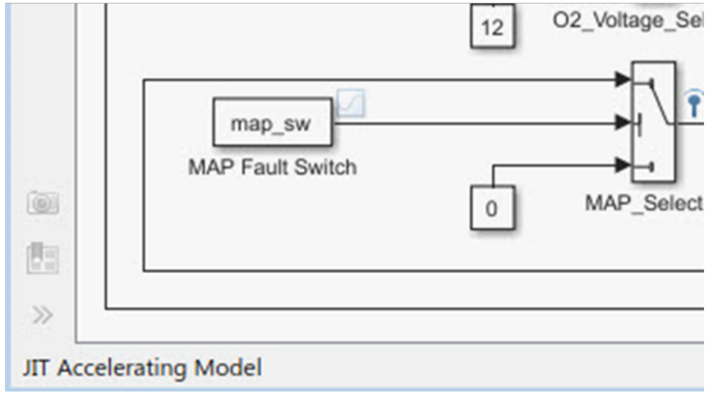
R2017a

Use the new `parsim` command and JIT acceleration to speed up your simulations



- Directly run multiple parallel simulations from the `parsim` command
- Especially use for Monte Carlo simulations and Design of Experiments
- Quickly build the top-level model for improved performance when running simulations in Accelerator mode





Learn more at this session:
Parallel Computing with MATLAB and Simulink

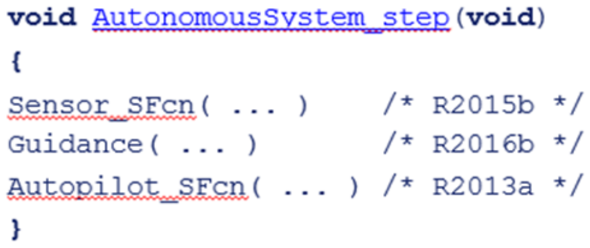
Parallel Computing Toolbox
MATLAB Distributed Computing Server

Cross-Release Code Integration

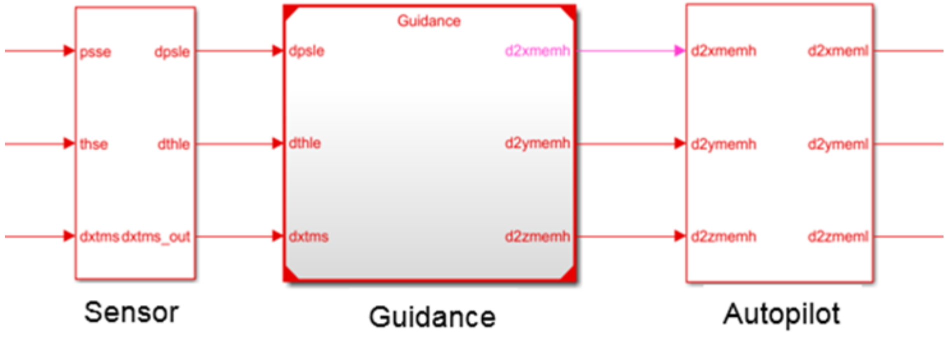
R2016b

Reuse code generated from previous releases

- Reuse code that you generated from previous releases (R2010a and later)
- Avoid reverification cost due to the reuse of unmodified code

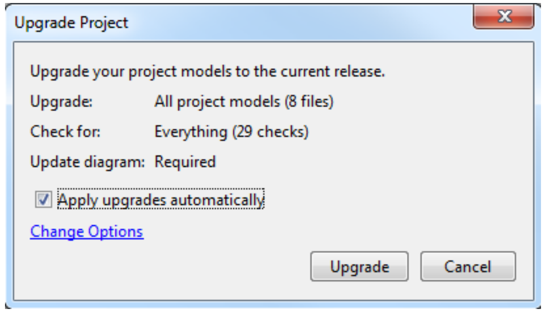


```
void AutonomousSystem_step(void)
{
Sensor_SFcn( ... )     /* R2015b */
Guidance( ... )        /* R2016b */
Autopilot_SFcn( ... ) /* R2013a */
}
```



Simulink Project Upgrade

R2017a



Upgrade Project

Upgrade your project models to the current release.

Upgrade: All project models (8 files)

Check for: Everything (29 checks)

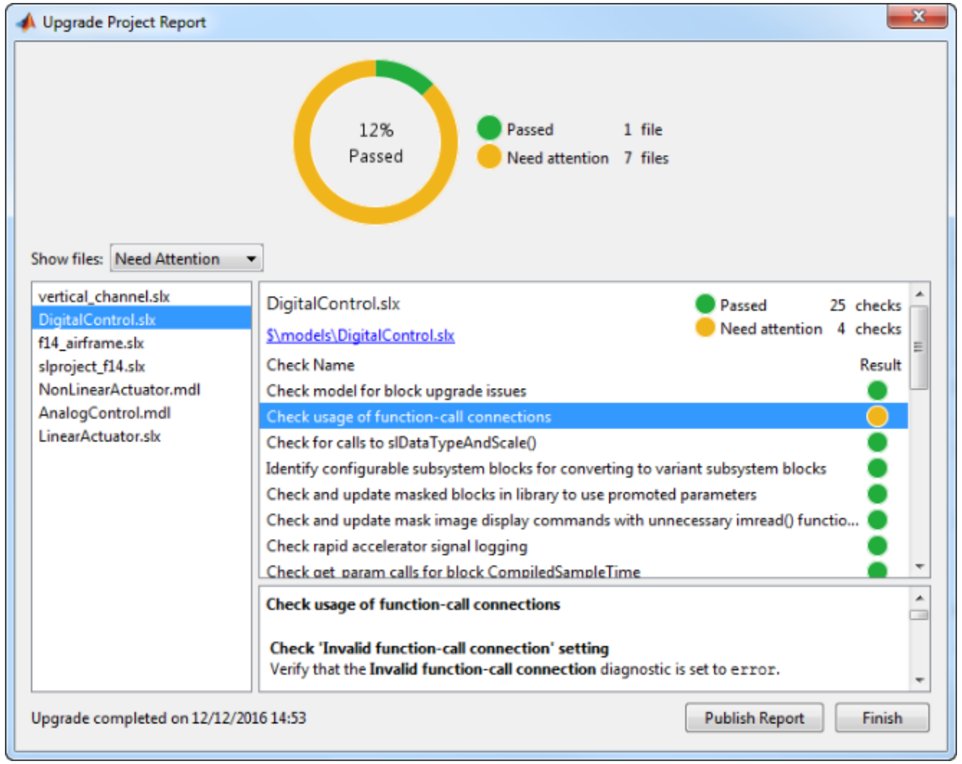
Update diagram: Required

☑ Apply upgrades automatically

Change Options

Easily update all the models in your Simulink Project to the latest release

- Avoid the manual process of upgrading one model at a time
- Simulink Project upgrade is an easy to use UI to automate the upgrade process of all the models in a Simulink project
- Fixes are automatically applied and a report gets generated



Upgrade Project Report

12% Passed

● Passed 1 file

● Need attention 7 files

Show files: Need Attention

vertical_channel.slx
DigitalControl.slx
f14_airframe.slx
slproject_f14.slx
NonLinearActuator.mdl
AnalogControl.mdl
LinearActuator.slx

DigitalControl.slx

$\models\DigitalControl.slx

● Passed 25 checks

● Need attention 4 checks

Check Name	Result
Check model for block upgrade issues	●
Check usage of function-call connections	●
Check for calls to slDataTypeAndScale()	●
Identify configurable subsystem blocks for converting to variant subsystem blocks	●
Check and update masked blocks in library to use promoted parameters	●
Check and update mask image display commands with unnecessary imread() functio...	●
Check rapid accelerator signal logging	●
Check get_param calls for block CompiledSampleTime	●

Check usage of function-call connections

Check 'Invalid function-call connection' setting
Verify that the **Invalid function-call connection** diagnostic is set to `error`.

Upgrade completed on 12/12/2016 14:53

Learn more at this session:
Simulink as Your Enterprise Simulation Platform

Platform Productivity

Getting your work done faster

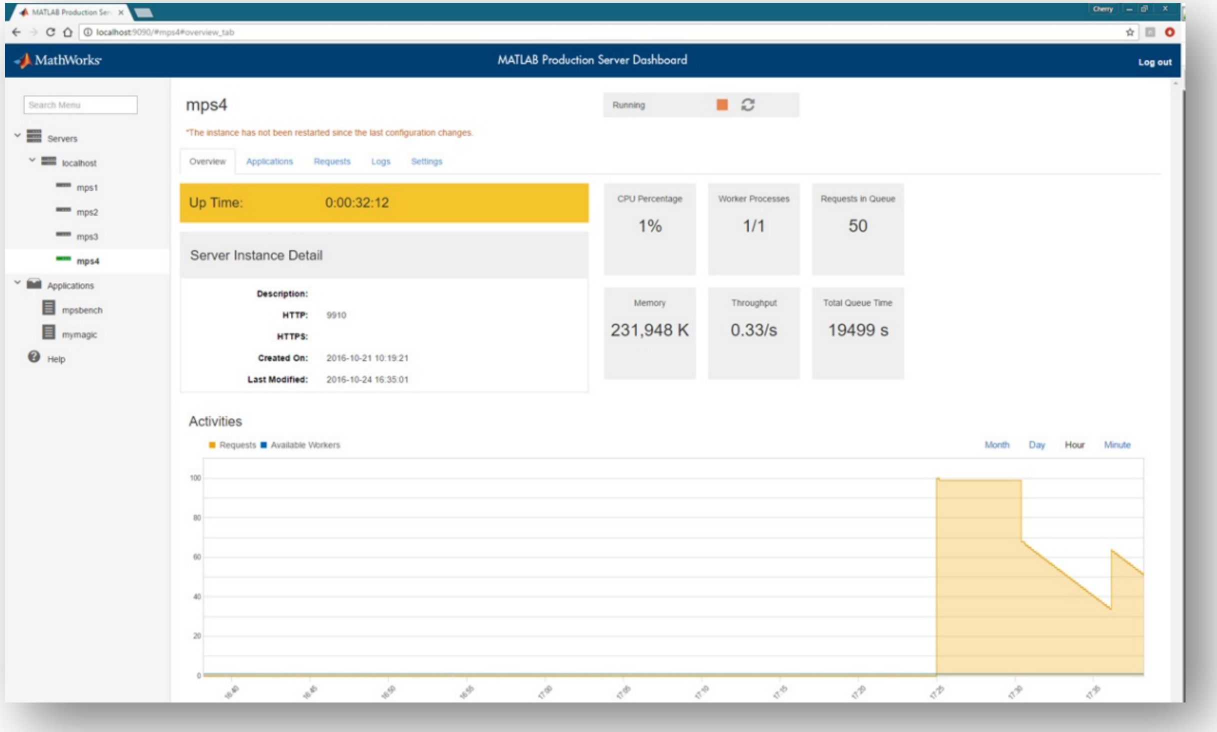
Workflow Depth

Support for your entire workflow

Integrate MATLAB Analytics into Enterprise Applications

Deploy MATLAB algorithms without recoding or creating custom infrastructure

- Develop clients for MATLAB Production Server in any programming language that supports HTTP using RESTful API and JSON — R2016a
- Configure and manage multiple server instances using a web-based interface — R2017a



mps4

Running

*The instance has not been restarted since the last configuration changes.

Up Time: 0:00:32:12

Server Instance Detail

Description:
HTTP: 9910
HTTPS:
Created On: 2016-10-21 10:19:21
Last Modified: 2016-10-24 16:35:01

CPU Percentage	Worker Processes	Requests in Queue
1%	1/1	50

Memory	Throughput	Total Queue Time
231,948 K	0.33/s	19499 s

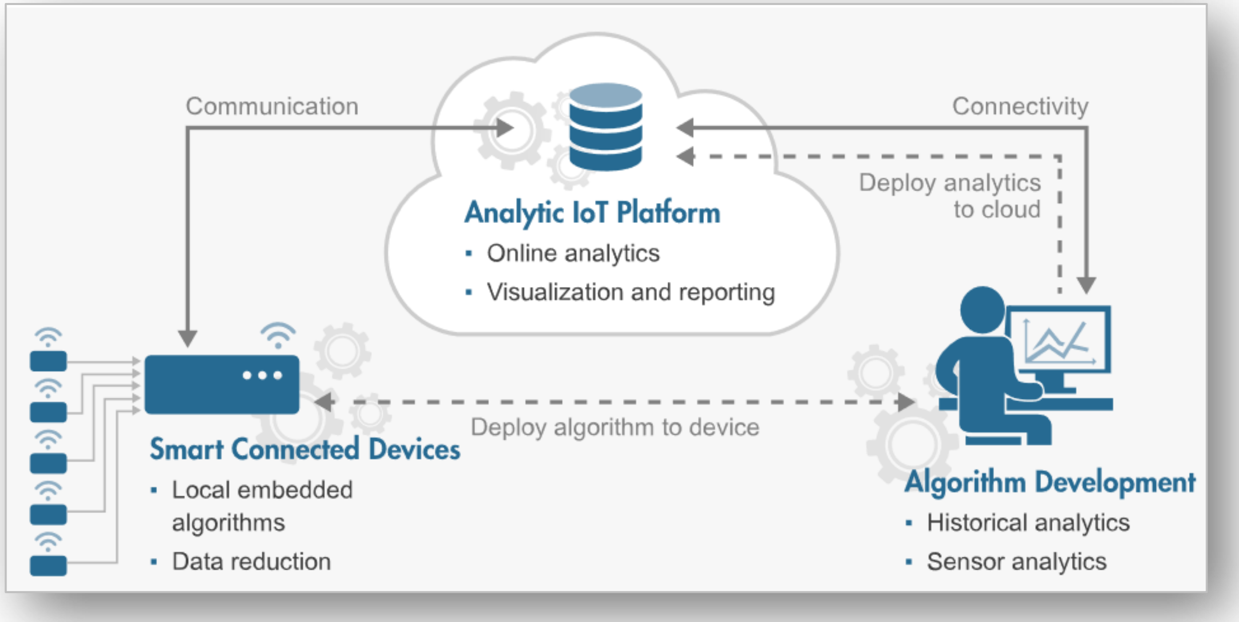
Activities

Learn more at this session:
Integrate MATLAB analytics into enterprise applications

Connecting MATLAB Analytics to IoT Systems

Develop analytics and deploy IoT systems

- Quickly collect and analyze IoT data with ThingSpeak and MATLAB
- Develop analytics algorithms using MATLAB and toolboxes
- Deploy on smart devices using code generation and embedded target support
- Deploy at scale on cloud using ThingSpeak and MATLAB Production Server

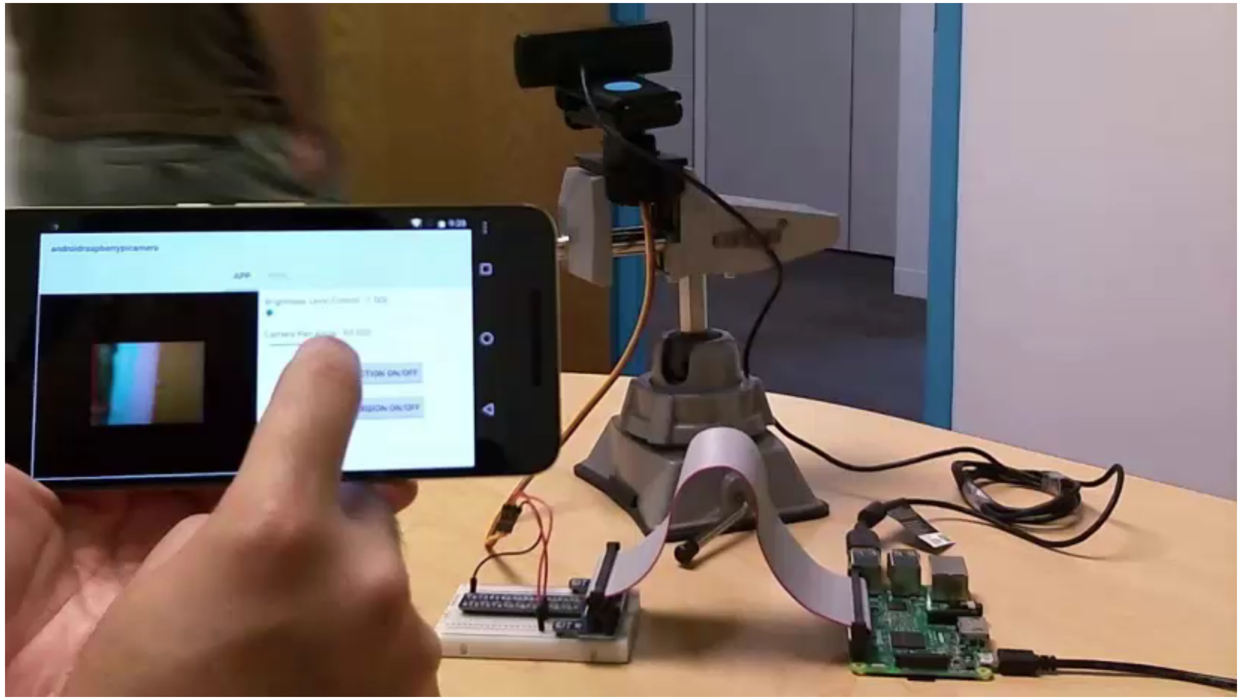


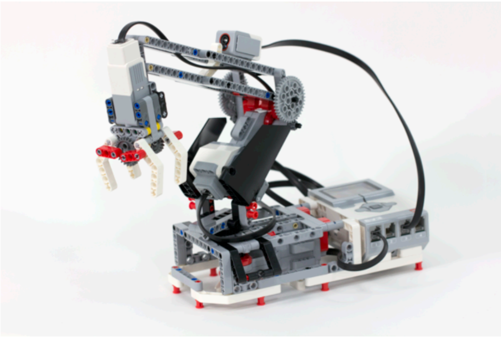
Learn more at this session:
Developing Analytics and Deploying IoT Systems

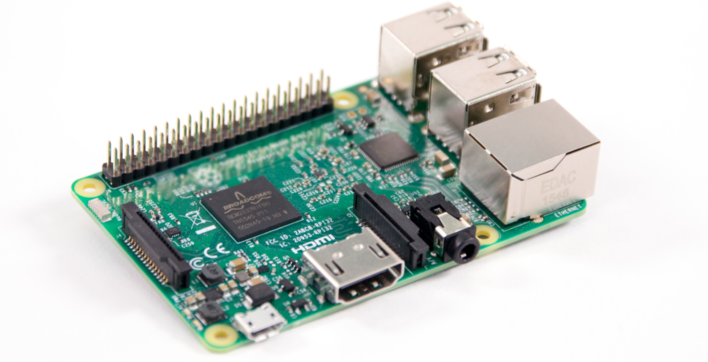
New Hardware Support

Run Simulink models on low-cost hardware devices

- Run Simulink models on Raspberry Pi 3 and Google Nexus devices
- Adds to existing hardware support, including LEGO, Arduino, iPhone, and Android devices



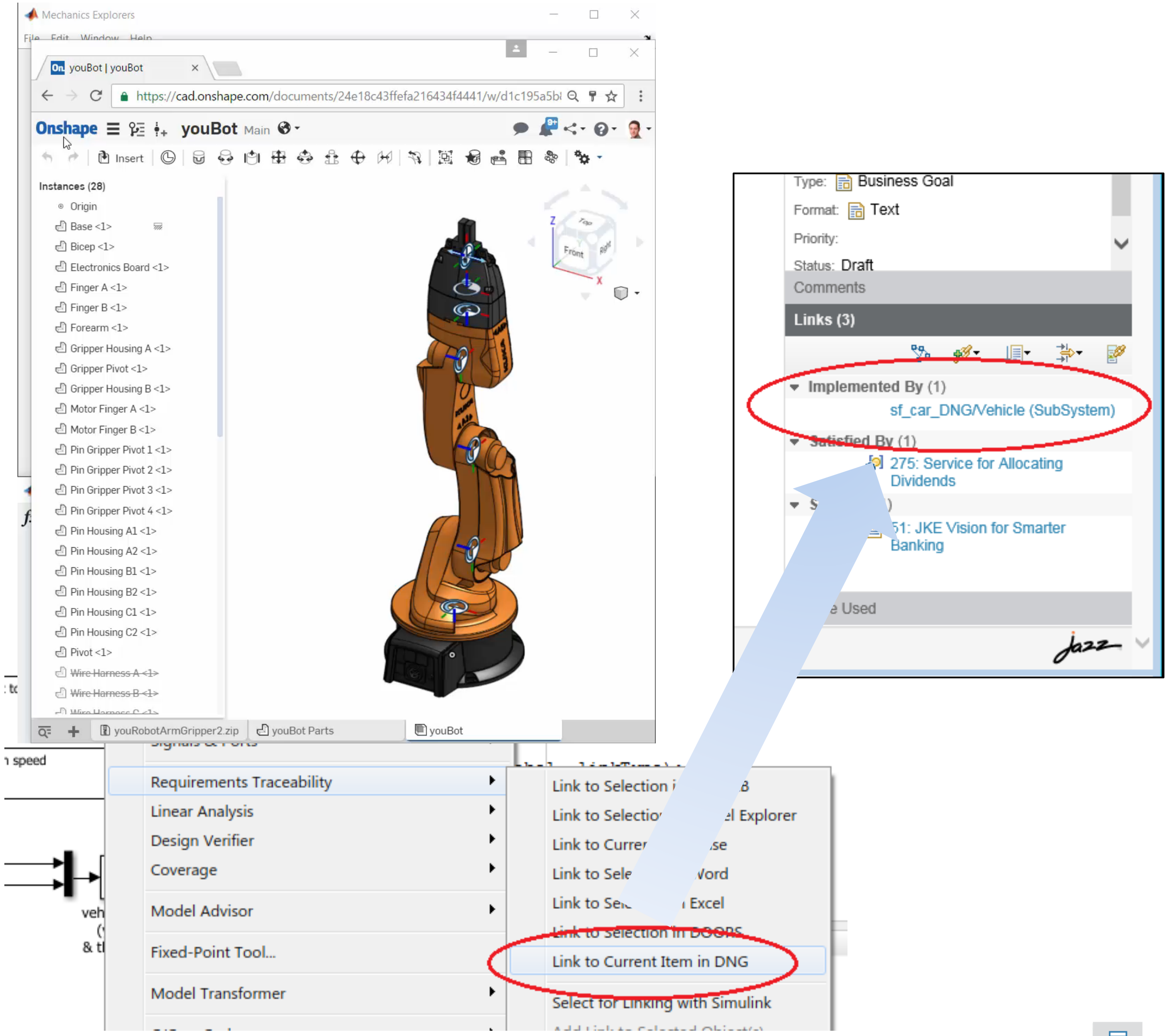




More Connections to 3rd Party Tools

Connect your models to Onshape and DOORS Next Generation

- Convert an Onshape CAD assembly into a Simscape Multibody model
- Link and trace model elements to requirements in DOORS Next Generation



Type: Business Goal
Format: Text
Priority:
Status: Draft

Comments
Links (3)
Implemented By (1)
sf_car_DNG/Vehicle (SubSystem)
Satisfied By (1)
275: Service for Allocating Dividends
51: JKE Vision for Smarter Banking
e Used
Jazz
Requirements Traceability
Linear Analysis
Design Verifier
Coverage
Model Advisor
Fixed-Point Tool...
Model Transformer
Link to Selection i
Link to Selectio el Explorer
Link to Curre se
Link to Sele Vord
Link to Se Excel
Link to Selection in DOORS
Link to Current Item in DNG
Select for Linking with Simulink

Learn more at this session:
Modeling Mechanical and Hydraulic Systems in Simscape

Simscape Multibody
Simulink Verification and Validation

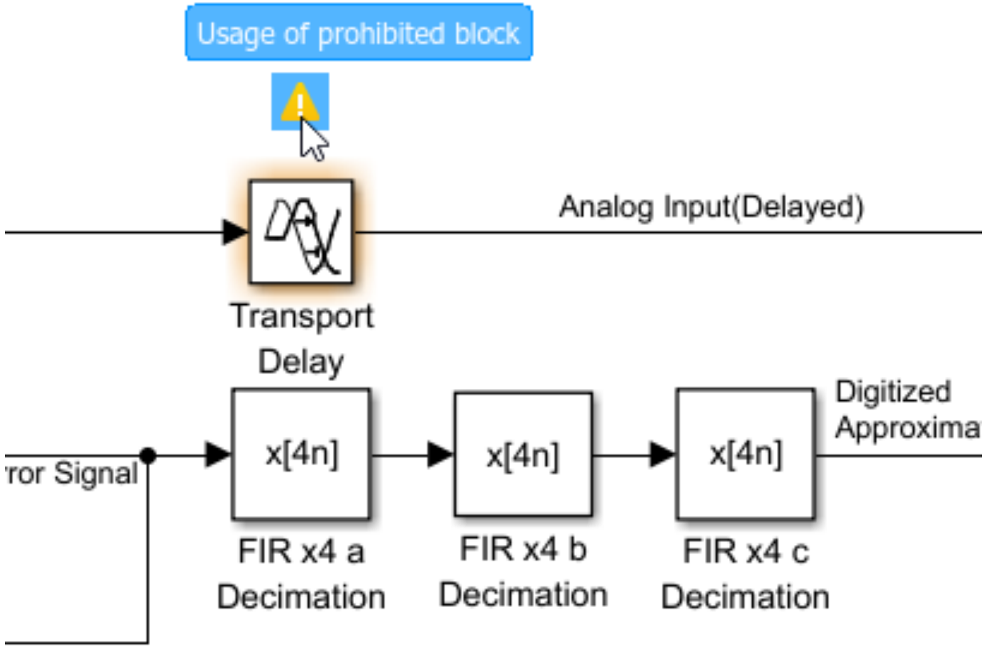
Complying with Safety-Critical Standards

Detect and fix standards compliance issues at design time with edit-time checking

- Quickly address compliance and modeling standards issues before running the model

- For example, check for prohibited blocks or block names

- Especially useful for applications that require compliance to standards such as DO-178, ISO 26262, IEC 62304

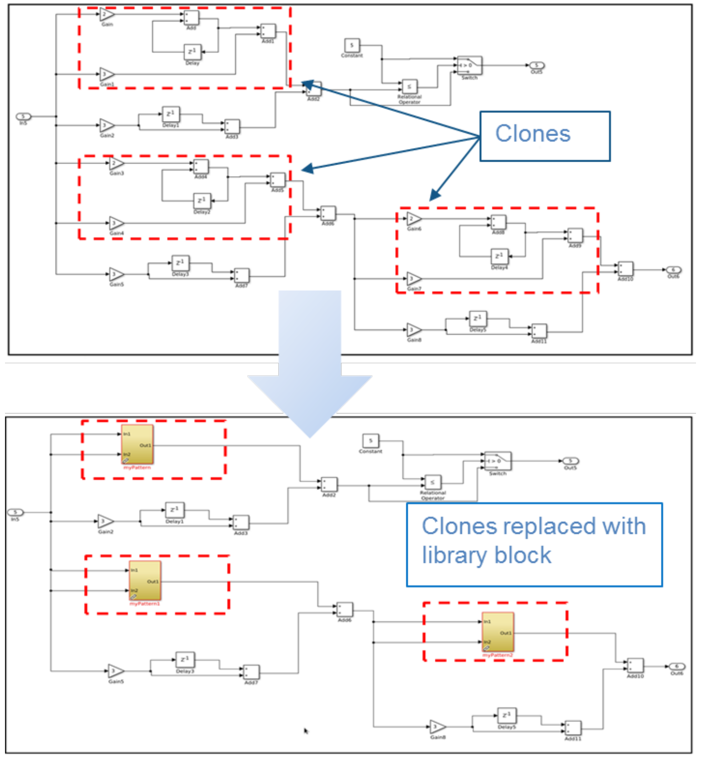


Learn more at this session:
Verification, Validation and Test in Model Based Design

Efficient Code Generation

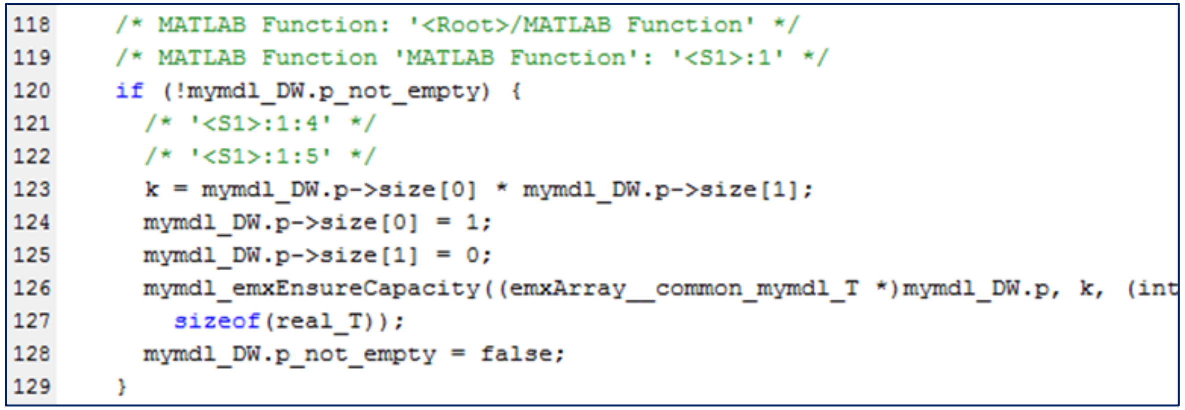
Improve code quality with clone detection and dynamic memory allocation

- Refactor repeating library patterns and subsystem clones
  - Reduces redundancy
  - Improves reusability
- Generate C code that uses dynamic memory allocation from MATLAB Function blocks
  - Allocate memory as needed at runtime



Clones

Clones replaced with library block



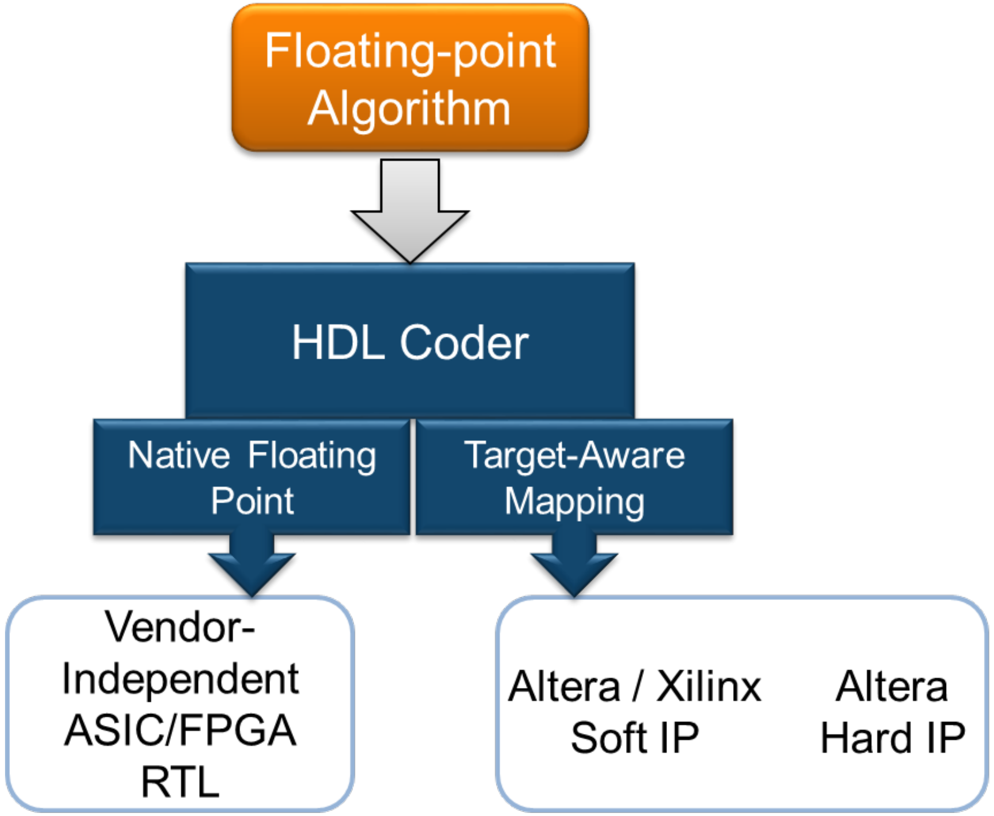
```
    /* MATLAB Function: '<Root>/MATLAB Function' */
    /* MATLAB Function 'MATLAB Function': '<S1>:1' */
    if (!mymdl_DW.p_not_empty) {
      /* '<S1>:1:4' */
      /* '<S1>:1:5' */
      k = mymdl_DW.p->size[0] * mymdl_DW.p->size[1];
      mymdl_DW.p->size[0] = 1;
      mymdl_DW.p->size[1] = 0;
      mymdl_emxEnsureCapacity((emxArray__common_mymdl_T *)mymdl_DW.p, k, (int
        sizeof(real_T));
      mymdl_DW.p_not_empty = false;
    }
```

Learn more at this session:
Generating Optimized Code for Embedded Microcontroller Algorithms

Floating Point HDL Code Generation

Generate HDL code directly from single-precision floating point Simulink models

- Generates native floating-point arithmetic HDL code complying to IEEE-754 standard
- Optimize for speed versus area using custom block-level settings
- Balance numerical accuracy versus hardware resource usage by mixing integer, fixed-point, and floating point operations.

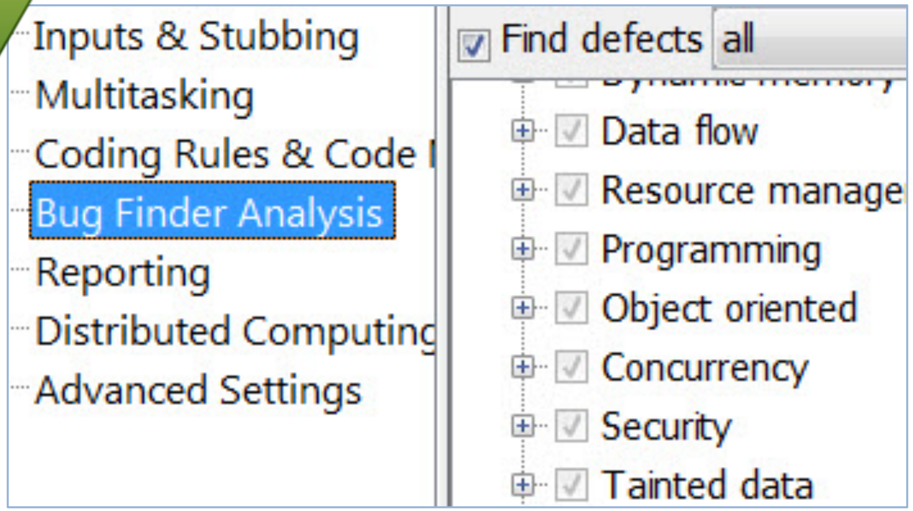


Learn more at this session:
Designing and Implementing Real-Time Signal Processing Systems

Code Verification

Detect and prove the absence of run-time errors in your source code using static analysis

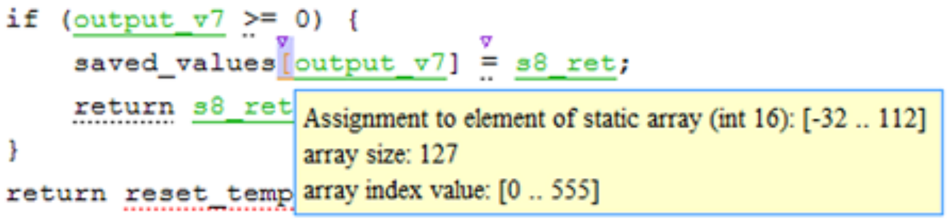
- Identify CERT C violations using defect checkers and coding rules
- Detect security vulnerabilities highlighted by the CERT C standard
- Addresses growing concern over software security with the rise in system connectivity



Inputs & Stubbing
Multitasking
Coding Rules & Code I
Bug Finder Analysis
Reporting
Distributed Computing
Advanced Settings

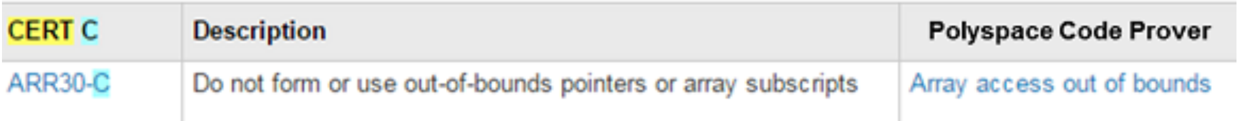
☑ Find defects all

⊞ ☑ Data flow
⊞ ☑ Resource manage
⊞ ☑ Programming
⊞ ☑ Object oriented
⊞ ☑ Concurrency
⊞ ☑ Security
⊞ ☑ Tainted data



```
if (output_v7 >= 0) {
    saved_values[output_v7] = s8_ret;
    return s8_ret
}
return reset_temp
```

Assignment to element of static array (int 16): [-32 .. 112]
array size: 127
array index value: [0 .. 555]



CERT C	Description	Polyspace Code Prover
ARR30-C	Do not form or use out-of-bounds pointers or array subscripts	Array access out of bounds

Learn more at this session:
Prove the Quality and Achieve MISRA compliance with Formal Methods Based Technique for High Integrity applications

Polyspace Bug Finder

Platform Productivity

Getting your work done faster

Workflow Depth

Support for your entire workflow

Application Breadth

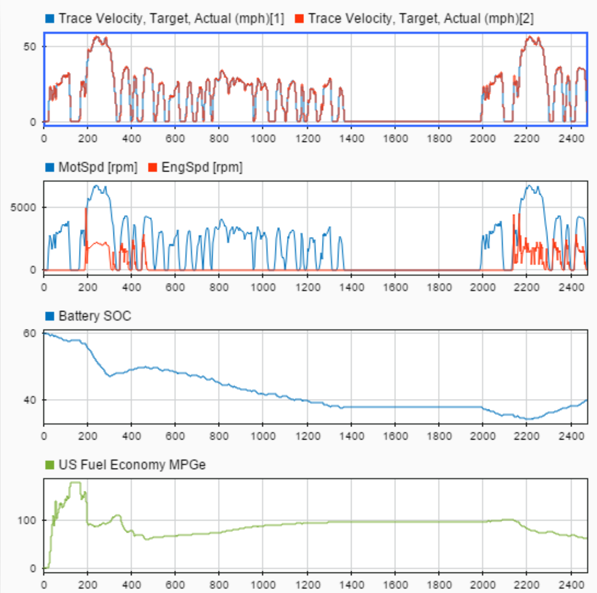
Products for the work you do

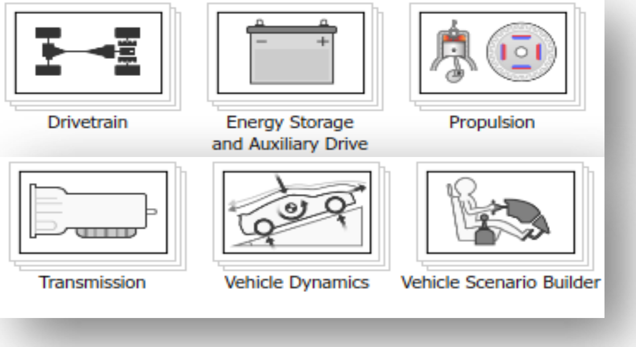
Model and simulate automotive powertrain systems

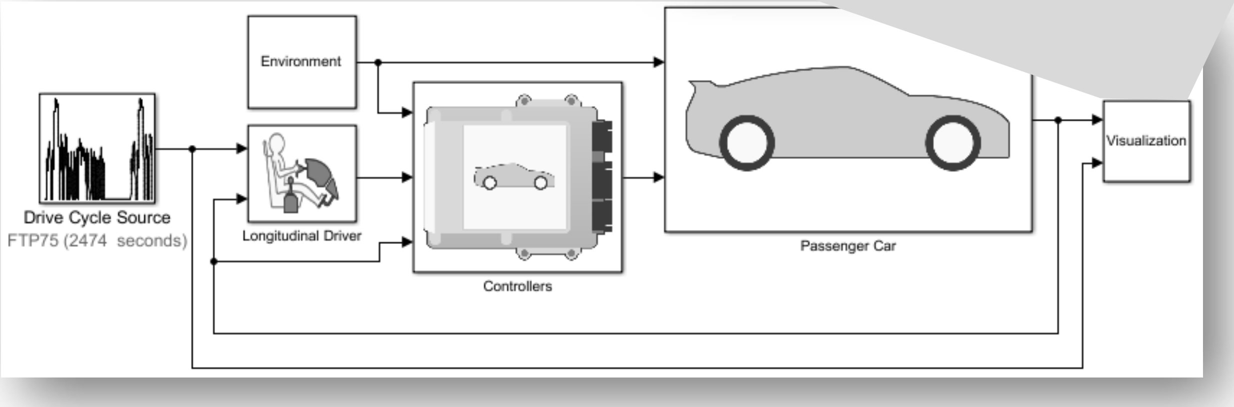
R2017a

Accelerate your powertrain controls development process

- Simulate engine and controller subsystems, transmission assemblies, battery packs
- Use pre-built conventional, EV, and HEV vehicle models that can be parameterized and customized
- Run fuel economy and performance simulations
- Deploy fast-running models onto HIL systems
- Connect to 3rd party engine models for specific components of the system







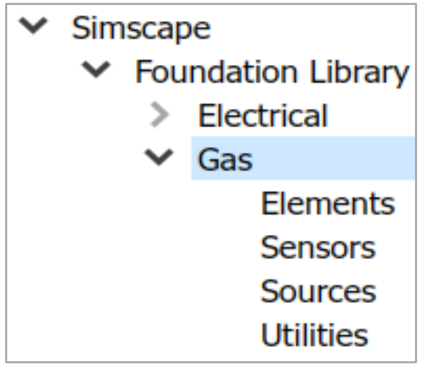
Powertrain Blockset

Gas Domain and Block Library

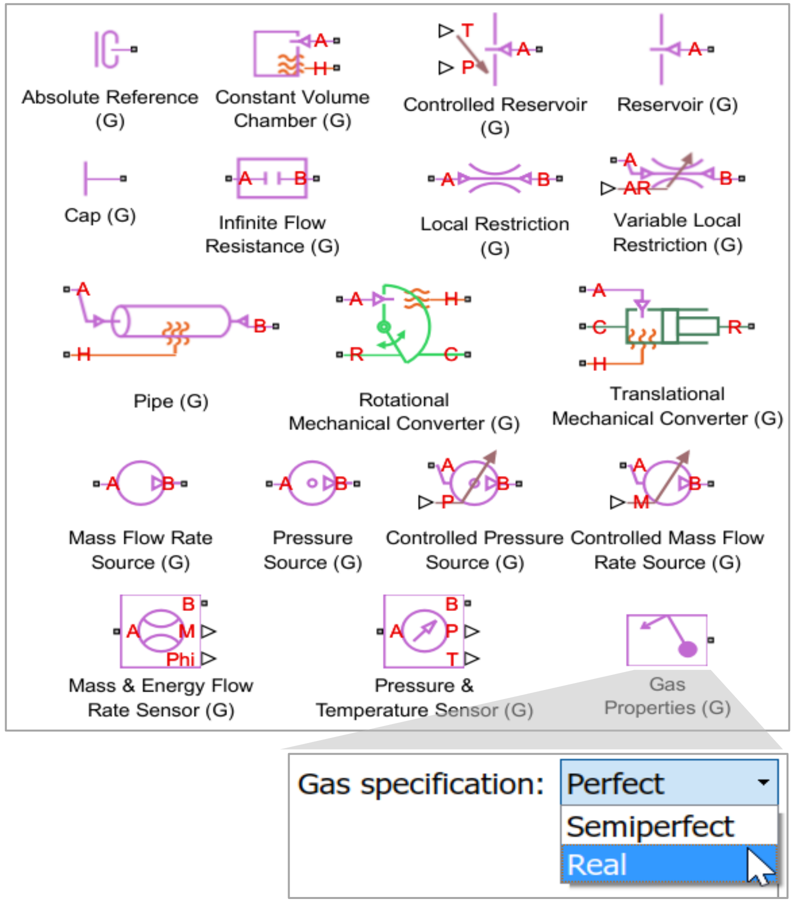
R2016b

Model gas systems with various levels of idealization

- Pneumatic actuation
- Gas transport in pipe networks
- Gas turbines for power generation
- Air cooling of thermal components
- Perfect gas, semiperfect gas, or real gas



- Simscape
  - Foundation Library
    - Electrical
    - Gas
      - Elements
      - Sensors
      - Sources
      - Utilities


Absolute Reference (G)
Constant Volume Chamber (G)
Controlled Reservoir (G)
Reservoir (G)
Cap (G)
Infinite Flow Resistance (G)
Local Restriction (G)
Variable Local Restriction (G)
Pipe (G)
Rotational Mechanical Converter (G)
Translational Mechanical Converter (G)
Mass Flow Rate Source (G)
Pressure Source (G)
Controlled Pressure Source (G)
Controlled Mass Flow Rate Source (G)
Mass & Energy Flow Rate Sensor (G)
Pressure & Temperature Sensor (G)
Gas Properties (G)
A
B
H
T
P
C
R
AR
M
Phi

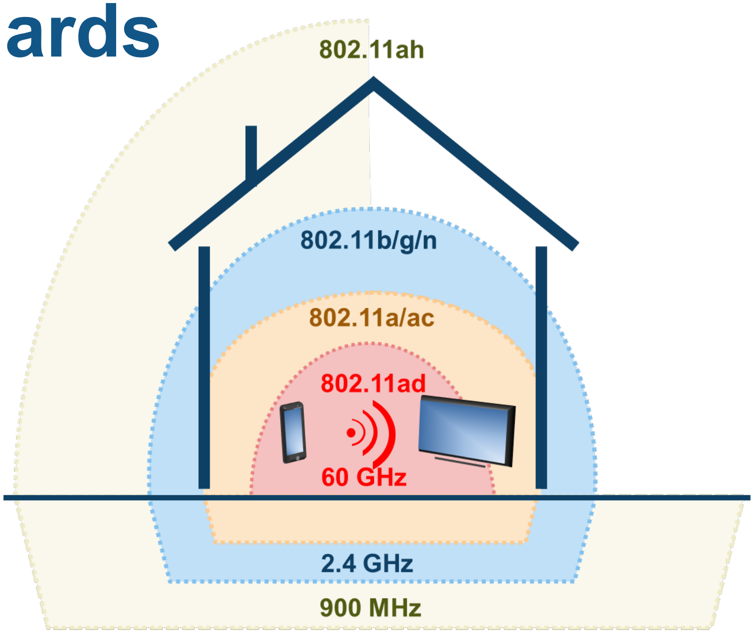
Gas specification: Perfect

Support for the Latest Wireless Standards

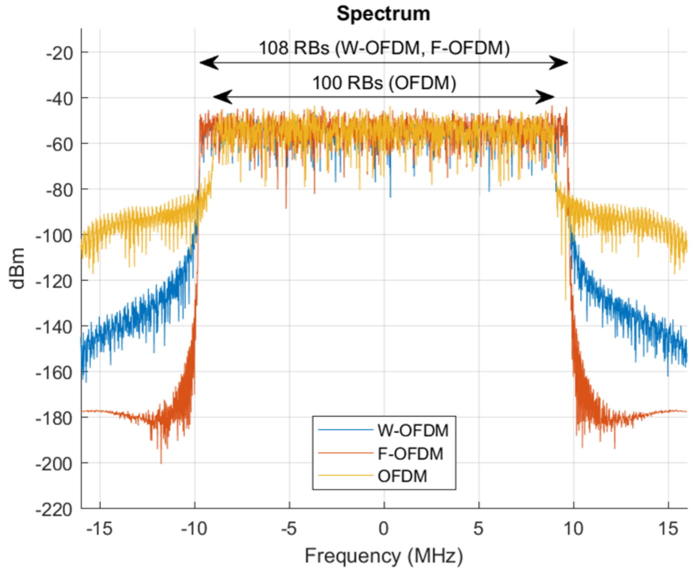
R2017a

Generate IEEE 802.11ad compliant waveforms and simulate 3GPP 5G radio technologies

- IEEE 802.11ad is a new Wi-Fi standard intended for high data rate short range communication
  - e.g., streaming video between a phone and a TV



- A new 5G library is available to explore the behavior and performance of new proposed 5G radio technologies



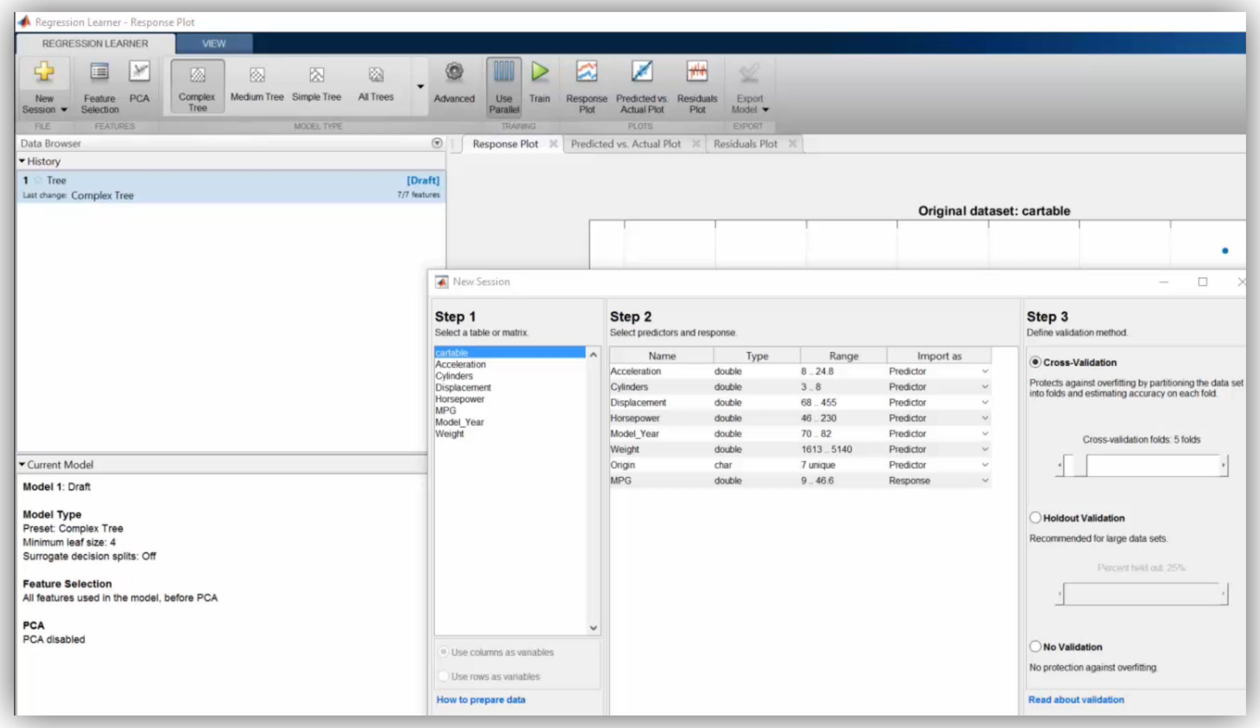
Learn more at this session:
Developing and Prototyping Next-Generation Communications Systems

WLAN System Toolbox
LTE System Toolbox

Machine Learning

“Learn" information directly from data without assuming a predetermined equation as a model

- Regression Learner app
  - Choose from multiple algorithms
  - Train and validate multiple models
  - Assess model performance, compare results, and choose the best model
- Code generation
  - Generate C code for predictive models that can be deployed directly to hardware devices



History

1 Tree [Draft]
Last change: Complex Tree 7/7 features

Current Model

Model 1: Draft

Model Type
Preset: Complex Tree
Minimum leaf size: 4
Surrogate decision splits: Off

Feature Selection
All features used in the model, before PCA

PCA
PCA disabled

Original dataset: cartable

Step 1
Select a table or matrix.

cartable
Acceleration
Cylinders
Displacement
Horsepower
MPG
Model_Year
Weight

● Use columns as variables
○ Use rows as variables

How to prepare data

Step 2
Select predictors and response.

Name	Type	Range	Import as
Acceleration	double	8 .. 24.8	Predictor
Cylinders	double	3 .. 8	Predictor
Displacement	double	68 .. 455	Predictor
Horsepower	double	46 .. 230	Predictor
Model_Year	double	70 .. 82	Predictor
Weight	double	1613 .. 5140	Predictor
Origin	char	7 unique	Predictor
MPG	double	9 .. 46.6	Response

Step 3
Define validation method.

● **Cross-Validation**
Protects against overfitting by partitioning the data set into folds and estimating accuracy on each fold.
Cross-validation folds: 5 folds

○ **Holdout Validation**
Recommended for large data sets.
Percent held out: 25%

○ **No Validation**
No protection against overfitting

Read about validation

Learn more at this session:
Big Data and Machine Learning

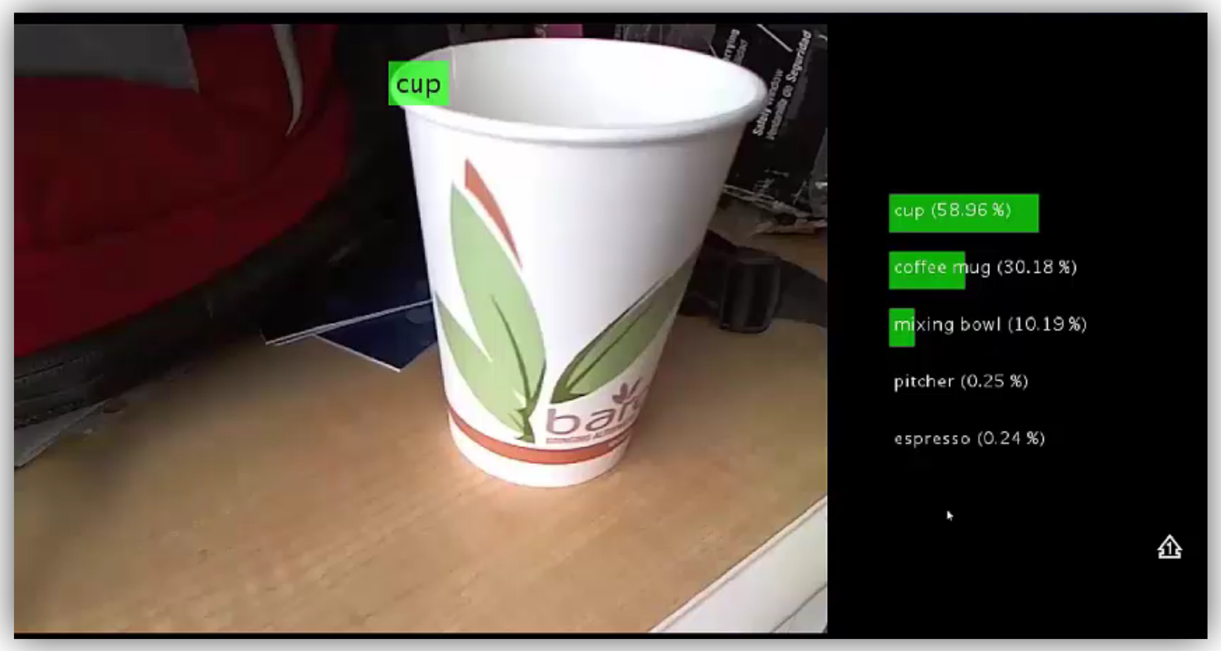
Statistics and Machine Learning Toolbox

Deep Learning

R2016b R2017a

Apply deep learning to computer vision problems

- Configure and train models using object detection algorithms
  (R-CNN, Fast R-CNN, Faster R-CNN)
- Leverage pretrained models for transfer learning
  (AlexNet, VGG-16, VGG-19)
- Import models from Caffe
- Train networks using multiple GPUs
  (including on Amazon EC2)



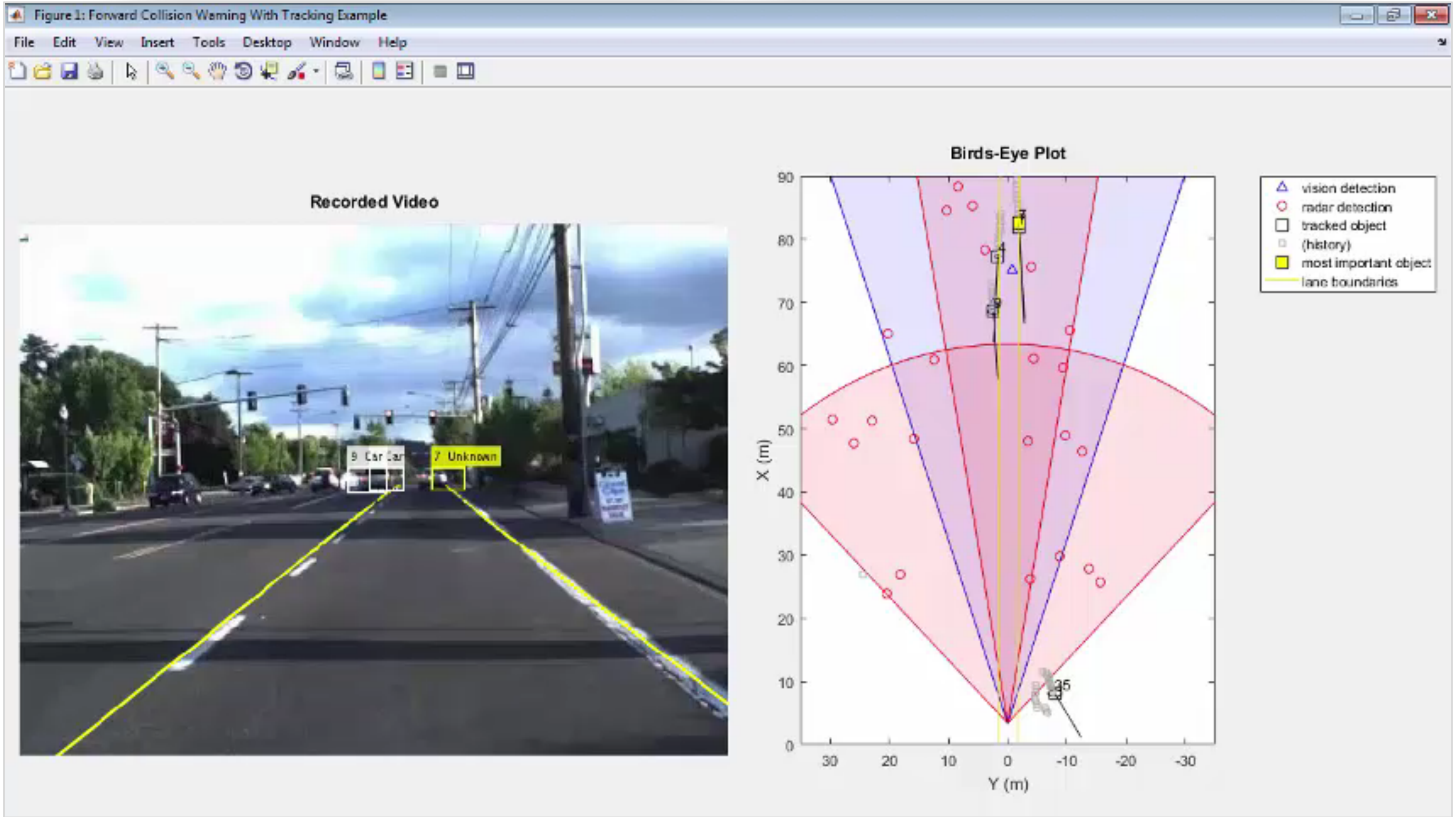
Learn more at this session:
Simplifying Image Processing and Computer Vision Application Development

Neural Network Toolbox
Computer Vision System Toolbox

Autonomous Driving Systems

Design, simulate, and test ADAS and autonomous driving systems

- Algorithm development
  - Sensor Fusion
  - Computer Vision
  - Deep learning
- Visualization tools
- Testing and verification
  - Ground Truth Labeling App
  - Traffic scenario generation



Learn more at this session:
Developing and Validating Perception Systems for ADAS & Automated Driving

Explore. Enroll. Excel.

New Training Courses

- Code Generation for AUTOSAR Software Components
- Testing Generated Code in Simulink
- Accelerating and Parallelizing MATLAB Code
- Communications System Design with MATLAB
- SimEvents for Discrete-Event System Modeling
- Software-Defined Radio with Zync using Simulink

New Training Modules

- Object Oriented Design with MATLAB
- Modeling RF Systems using MathWorks Tools
- Modeling Radar Systems using Phased Array Systems Toolbox
- Modeling Wireless Communication Systems using Phased Array Systems Toolbox
- Real-Time Testing with Simulink Real-Time and Speedgoat Hardware

Self-Paced Online Training Offerings

- MATLAB Onramp (Free)
- MATLAB Fundamentals
- MATLAB Programming Techniques
- MATLAB for Data Processing and Visualization
- Machine Learning with MATLAB
- MATLAB for Financial Applications

Platform Productivity

Getting your work done faster

Workflow Depth

Support for your entire workflow

Application Breadth

Products for the work you do

What’s New in MATLAB and Simulink?

Platform Productivity

- Live Editor
- MATLAB Apps
- New (big) data types
- Modeling enhancements
- Release adoption

Workflow Depth

- Enterprise applications
- IoT systems
- 3rd party tool integration
- Standards compliance
- Code generation and verification

Application Breadth

- Powertrain systems
- New wireless standards
- Machine learning
- Deep learning
- Autonomous driving

MATLAB EXPO 2017

Thank You