Reasons and Strategies for Upgrading to MATLAB
Continuous evolution of MATLAB & Simulink
Data Types Facilitate Data Analytics (and Data Analysis)

<table>
<thead>
<tr>
<th>Numeric</th>
<th>Heterogeneous</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>double, single, …</td>
<td>timetable</td>
<td>char, cell string</td>
</tr>
<tr>
<td>logical</td>
<td></td>
<td>str</td>
</tr>
<tr>
<td>categorical</td>
<td></td>
<td></td>
</tr>
<tr>
<td>datetime</td>
<td></td>
<td></td>
</tr>
<tr>
<td>calendarDuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>duration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>str</td>
<td>R2016b</td>
<td>R2016b</td>
</tr>
<tr>
<td>str</td>
<td></td>
<td></td>
</tr>
<tr>
<td>str</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
string Array

Manipulate, compare, and store text data efficiently

- String class for text data
  - Memory efficient
  - Fast operations

- Host of new functions
  - Find and replace
    - `contains` / `startsWith` / `endsWith` / `replace` / `replaceBetween`
  - Join and split
    - `join` / `split` / `strsplit` / `splitlines`

Saved 50% memory freeing up 1GB of RAM!

My script runs 10X faster and is easier to read!

- Internal User
Data type for preprocessing time-stamped data

\[
m = \text{mdf('CANape.MF4')}\]

\[
t = m.\text{read}(1, m.\text{ChannelNames}{1}, 1, 15)\]

mdf4-import provided by Vehicle Network Toolbox
tall Array

- New data type in MATLAB
- Targeted for tabular/columnar data
  - Finite number of columns
  - One or more rows can fit into memory
  - **Overall data size is too big to fit into memory**
- Statistical and machine learning applications
  - Hundreds of functions supported in MATLAB and Statistics and Machine Learning Toolbox

Credit Scoring  Prognostics
Process Optimization  Fleet Analytics
MATLAB Execution Engine

Redesigned execution engine runs MATLAB code faster

- All MATLAB code can now be JIT compiled
- Average performance improvement of 40% on 76 performance-sensitive user applications
- A platform for future improvements
- Performance testing framework
  - Measure MATLAB code performance
  - Interface leverages the unit testing framework
MATLAB Live Editor

Change the way you work in MATLAB

- See results together with the code that produced them
- Add equations, images, hyperlinks, and formatted text
- Create lectures that combine explanatory text, mathematical equations, code and results
Continuous evolution of MATLAB & Simulink
Just-in-Time Acceleration Builds

Quickly build the top-level model for improved performance when running simulations in Accelerator mode

- Simulink now generates an execution engine in memory instead of generating code before running simulations in Accelerator mode
- The result: simulations run in Accelerator mode now start up much faster
Property Inspector

Edit parameters and properties of model elements using a single interface

- Open using View -> Property Explorer
- Similar to what you would see in the dialog windows for a block
- Undo any parameter edits using Ctrl-Z
Model Data Editor

Configure model data properties using a table within the Simulink Editor

- Similar to information contained in Model Explorer
- Change the names of signals and mark which signals you want to test point, log, or stream
- When you select an item in the list, it gets highlighted in the model and vice versa
Initialize and Terminate Function Blocks

Respond to events to model dynamic startup and shutdown behavior

- Model functions that embedded systems use to start up and initialize themselves, as well as those functions to terminate and shut down

- Initialize and terminate functions are allowed to be customized and aggregated in generated code

- Important new modeling workflow enabled by two features:
  - Initialize, Reset and Terminate Functions
  - State Reader and Writer Blocks
Simulink Units

Specify, visualize, and check consistency of units on interfaces

- Specify physical units for Simulink signals and bus elements at the interfaces of components such as subsystems, model references, Stateflow charts and MATLAB function blocks
- Identify unit mismatches at the component interfaces
- Enforce consistency is by restricting the unit systems for certain components using the configuration parameter, ‘Allowed unit systems’
One-Click Display

Click a signal line when the simulation is running to view the current value

- Display port value for a signal by clicking it during simulation for easy debugging
- For bus signals, select the signals of interest before simulation
New Interface for Scopes

View and debug signals with cursors and measurements

- Scope, Floating Scope, and Viewers all upgraded with new UI
- Includes simulation data analysis and debugging tools
  - Cursors
  - Measurements
  - Triggers
Default Model Template

Use your own customized settings when creating new models

- Using a default template allows team members to use the same settings every time a new Simulink model is created

- It can either be one of the shipping templates or your own custom model template

- Applies to all the ways in which a new model is created – Simulink library browser, command line API or new model menu item in an existing Simulink model
Edit-Time Checking

Detect and fix potential standards compliance issues in your model at design time

- Quickly address compliance and modeling standard issues
- Identify and fix while editing
- Avoid rework later to meet standards
Upgrade Advisor API

Automate the process of upgrading large model hierarchies

- Use `analyze` function to analyze the model for recommended upgrades
- Use `upgrade` function to analyze and automatically apply recommended fixes
- If you are using Simulink Projects, you can programmatically upgrade all the models in your project using the Upgrade Advisor and Simulink Project APIs.

```matlab
% Load the Simulink model to upgrade
load_system('MyModel');
% Get the upgrade object
upgrader = upgradeadvisor('MyModel');

% Analyze the model for recommended upgrades
analyze(upgrader);

% Analyze and automatically fix all issues
upgrade(upgrader);
```
Upgrade Constraints

32-bit

Compilers

Windows

64-bit

Co-Simulation

MathWorks

MathWorks

Linux

Partner

CONFERENCES 2016
Upgrade Challenges

- Short time between projects to upgrade
- Encountered a show-stopping issue after an upgrade
- Limited knowledge of new features and their impact on the process
- Took multiple years to upgrade to a new version
- Custom tools don’t work in the new version
- Maintaining hundreds to thousands of models
- Will have to revalidate the models and code
- Users want to stay in the old version
Upgrade Workflow

Assess → Plan → Migrate → Test → Release → Support

MathWorks
AUTOMOTIVE CONERENCE 2016
ASSESS

- Choose a Target Version
  - Review Release Notes and Bug Reports
  - Contact MathWorks partners to verify which versions they support
  - Choose and change your target version if necessary

- Initial Testing
  - Run Upgrade Advisor on a model

- Regression Testing
  - Run regression tests on tasks that users commonly perform
  - Better to find the issues now than find them later
PLAN: Create a Business Case

- New features that will improve your workflows
- Upgrading to a new operating system
- Third party software will no longer support your version
PLAN: Define Your Goals Upfront

- Limit the scope of the upgrade process
  - Don’t try to do everything

- Upgrade existing models to a new version
  - Upgrade models without introducing new features
  - Validate models in the new version

- Upgrade custom tools to a new version
  - Replace custom tools with built-in Simulink functionality
MIGRATE: Initial Migration

- Select a large model to test the upgrade process
  - Complies with your modeling style guidelines
- Run Upgrade Advisor on the model
- Test your typical workflows
- Document warning and error messages
- Expand testing to more models
  - Different modeling styles
- Resolve the issues
MIGRATE: Automated Migration

- Automate the conversion process
- Create one MATLAB script that calls everything
- Recommend that the model expert convert his/her models
  - Aware of production deadlines
  - Will need to validate the model
TEST

- Custom Tools
  - Verify the custom tools perform the intended behavior in the new version
- Third Party Tools
  - Test third party tools with the new version
- Beta Testing
  - Start using the new MATLAB version for everyday work
RELEASE

- **Training**
  - New Features
  - Custom Tools

- **Timing**
  - Needs to be flexible
  - Don’t want to impact a production schedule
  - New models should use the new version
  - Set a deadline for the rest of the models
Continuous Upgrade Philosophy

- **Prerelease Testing**
  - Test your models and custom tools for each prerelease
  - Send your feedback to the MathWorks
  - Don’t wait until your next upgrade to start testing

- **Industry Model Testing**
  - Submit models to MathWorks Industry Model Testing
  - MathWorks will run tests on your models prior to the prerelease
  - Dramatically reduces the chance of release incompatibilities

- **Seminars**
  - To keep up to date with MathWorks new features
  - Will help you decide what version to upgrade to
Benefits

- Surprises are minimized.
- You don’t have to wait to use new features.
- The new version is released in a timely manner.
- MathWorks may resolve your issues before your next upgrade.
- Your future upgrades will be easier.
Simulink Test provides tools for authoring, managing, and executing systematic, simulation-based tests of models, generated code, and simulated or physical hardware. It includes a Test Sequence block that lets you construct complex test sequences and assessments, and a test manager for managing and executing tests. Simulink Test enables functional, baseline, equivalence, and back-to-back testing, including software-in-the-loop (SIL), processor-in-the-loop (PIL), and real-time hardware-in-the-loop (HIL). You can apply pass and fail criteria that include absolute and relative tolerances, limits, logical checks, and temporal conditions. Setup and cleanup scripts help you automate or customize test execution.

You can create nonintrusive test harnesses to test components in the system model or in a separate test model. You can store test cases and their results, creating a repository for reviewing and investigating failures. You can generate reports, archive and review test results, rerun failed tests, and debug the component or system under test.

With Simulink Test and Simulink Verification and Validation™, you can link test cases to requirements captured in Microsoft® Word, IBM® Rational® DOORS®, and other documents.

Support for industry standards is available through IEC Certification Kit (for IEC 61508 and ISO 26262) and DO Qualification Kit (for DO-178C).

**Key Features**
- Create Test Harnesses
- Author Test Sequences and Assessments
- Manage Test Plans and Test Execution
- Perform Functional and Back-to-Back Testing
- Analyze and Share Test Results
RESOURCES

**Upgrade Paper**

**Upgrade Center**

**Why Upgrade?**
Get more out of MATLAB® and Simulink® by upgrading to the latest versions of your products. Most users can download the latest release and get started using the new features right away.

Users who rely on critical applications and models built in MATLAB and Simulink see the benefits of upgrading regularly:

- Maximize the value of your MATLAB and Simulink software
- Take advantage of the features you requested
- Increase workflow efficiency and build confidence in your solutions
- Save time by upgrading regularly

- How do I upgrade?
- Contact support for help upgrading
- Upgrade now

**Maximize the value of your MATLAB and Simulink software**
Get the most out of MathWorks Software Maintenance Service. Every year, MathWorks delivers a steady stream of new technology, customer-requested product enhancements, and quality improvements. Join other companies in taking advantage of the features that you are helping to improve.

*When new tools become available to improve your process, you have to maintain a growth opportunity to make use of them.*

— Darin Valenzuela, Millennium Engineering & Integration Company
SUPPORT

Technical Support

Consulting Services

Software Upgrade Service

Upgrading to the latest software version gives access to new functionality for improved performance, scalability, management of complexity, and compliance with regulatory standards.

MathWorks Consulting Services applies best practices gained from previous customer upgrades, and insider knowledge of MATLAB and Simulink enhancements to help you complete a seamless and efficient migration to a new release.

Upgrade Plan - Mitigate Risk and Maximize New Functionality Benefits

We assess your current models, tools, and development processes and develop a customized upgrade strategy. We identify new MATLAB and Simulink features that can be leveraged to meet your current and future business and technical goals. We ascertain any custom tools, patches, and infrastructure that can be retired or optimized with the upgrade.

Upgrade Impact Testing and Issues Resolution

MathWorks Consultants perform an upgrade impact test on your practice workflow, models, tools, and development processes that highlights and addresses any issues. We document any incompatibilities and issues related to your models, processes, custom and third-party tools and work with you to resolve them.