



# MathWorks/ESRF Seminar: Application of Simulink for modeling and instrumentation development

Institute-wide Access and Ressources

22/04/2026



Charbel Cherfan  
Customer Success Engineer



Mathieu Cuenant  
Application Engineer



Morgan Fremovici  
Application Engineer



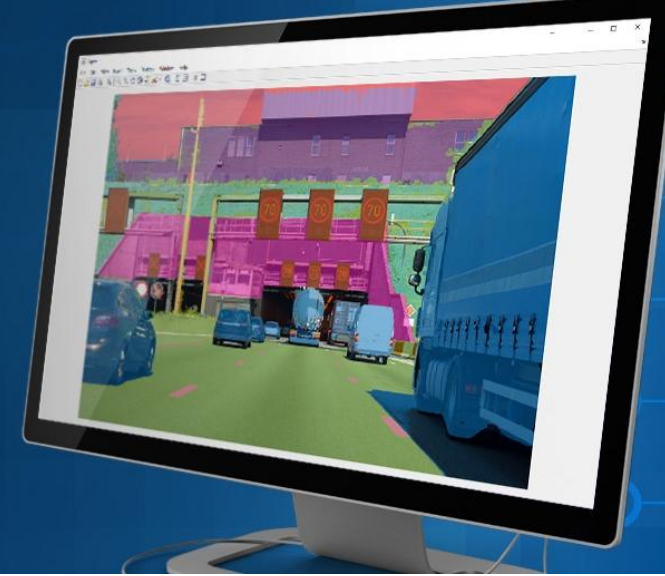
Jérôme Lespinasse  
Account Manager



# Agenda for Today

8:30 AM	→ 8:45 AM	<b>Welcome and Introduction to the seminar</b> ⌚ 15m
		<ul style="list-style-type: none"><li>▪ Purpose of the seminar</li><li>▪ MathWorks/Simulink tools accessible today at ESRF</li></ul> <p>Speaker: Charbel Cherfan (MathWorks)</p>
8:45 AM	→ 9:20 AM	<b>Model-based design</b> ⌚ 35m
		<ul style="list-style-type: none"><li>▪ General overview and industrial use cases</li><li>▪ Simulation of physical systems</li><li>▪ Development of control systems</li><li>▪ Verification and validation workflows</li></ul> <p>Speaker: Mathieu Cuenant (MathWorks)</p>
9:20 AM	→ 10:00 AM	<b>Signal processing with Simulink for particle accelerators and RF systems</b> ⌚ 40m
		<ul style="list-style-type: none"><li>▪ Motivation and context</li><li>▪ What can you do with Simulink?</li><li>▪ Signal processing workflows in Simulink</li><li>▪ Integrating custom or external MATLAB/C code into Simulink models</li><li>▪ Simulating RF controlled components at the system-level</li></ul> <p>Speaker: Charbel Cherfan (MathWorks)</p>
10:00 AM	→ 10:20 AM	<b>Coffee brak</b> ⌚ 20m
10:20 AM	→ 11:00 AM	<b>Examples of application of Simulink at ESRF today</b> ⌚ 40m
		<ul style="list-style-type: none"><li>▪ Improving the accelerator fast-orbit feedback performance through Simulink-based modelling</li><li>▪ Real-time control of high resolution mechatronic devices with Simulink</li></ul> <p>Speakers: Marie Penot (ESRF), Thomas Dehaeze (ESRF)</p>
11:00 AM	→ 11:45 AM	<b>From models to embedded code</b> ⌚ 45m
		<ul style="list-style-type: none"><li>▪ Use cases and hardware platforms<ul style="list-style-type: none"><li>▪ Micro-controllers, system on chip (SoC), FPGAs, GPUs, PLCs</li></ul></li><li>▪ Model development and code generation workflow</li><li>▪ Example of multicore C code generation</li><li>▪ Simulink Real-Time: QNX to Linux upcoming transition</li></ul> <p>Speaker: Morgan Fremovici (MathWorks)</p>
11:45 AM	→ 12:00 PM	<b>General questions and seminar wrap-up</b> ⌚ 15m

# MATLAB® & SIMULINK®



```

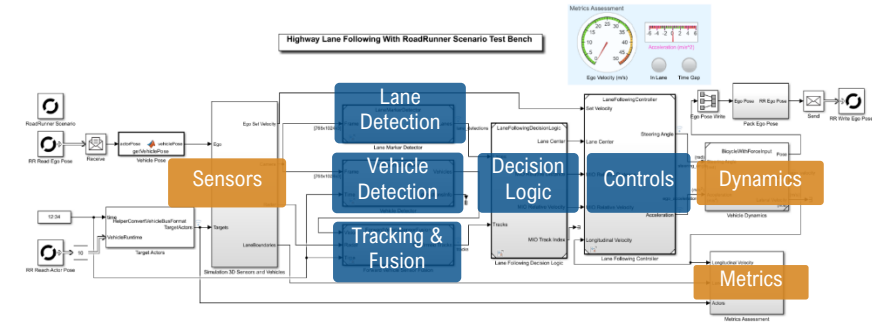
145 rightEdgeBoundaryIndex = distanceToBoundaries == xMinRightDistance;
146 side = 0;
147 for i = 1 : size(rightEdgeBoundaryIndex, 1)
148     if rightEdgeBoundaryIndex(i) == 3
149         [dA i] = min(dA i);
150     end
151     rightEdgeBoundaryIndex(i) = strongBoundaries(i);
152 end
153 else
154     rightEdgeBoundary = parabolicLaneBoundary.empty();
155 end

Show the detected lane markers in the bird's-eye-view image and in the regular view.
156 vVehiclePoints = bottomOffsetDistanceSensor;
157 birdseyeViewLane = insertLaneBoundary(birdseyeView, leftEdgeBoundary, birdseyeView_vVehiclePoints, 'color', 'red');
158 birdseyeViewLane = insertLaneBoundary(birdseyeViewLane, rightEdgeBoundary, birdseyeView_vVehiclePoints, 'color', 'green');
159 frameLeftLane = insertLaneBoundary(frame, leftEdgeBoundary, sensor_vVehiclePoints, 'color', 'red');
160 frameRightLane = insertLaneBoundary(frame, rightEdgeBoundary, sensor_vVehiclePoints, 'color', 'green');
161
162 figure
163 subplot('position', [0, 0, 0.5, 1.8]) % [left, bottom, width, height] in normalized units
164 imshow(birdseyeViewLane)
165 subplot('position', [0.5, 0, 0.5, 1.8])
166 imshow(frameLeftLane)
167
Locate Vehicles in Vehicle Coordinates
Detecting and tracking vehicles is critical in front collision warning (FCW) and autonomous emergency braking (AEB) systems.
Load an aggregate channel features (ACF) detector that is pre-tuned to detect the front and rear of vehicles. A detector like this can handle scenarios where issuing a collision
warning is important. It is not sufficient, for example, for detecting a vehicle traveling across a road in front of the ego vehicle.
    
```

**MATLAB**

#Math #Graphics #Programming

Event-Based Modeling	Physical Modeling	
Real-Time Simulation and Testing	Verification, Validation, and Test	Simulation Graphics and Reporting
<p>Simulation and Model-Based Design</p>		
Parallel Computing		Code Generation
<p>The Language of Technical Computing</p>		
Math, Statistics, and Optimization	Application Deployment	Database Access and Reporting



**Simulink**

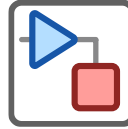
#Design #Simulate #Deploy

# Applications and product family



**MATLAB**  
PRODUCT FAMILY

AI, Data Science, and Statistics  
Parallel Computing  
Math and Optimization  
Reporting and Database Access



**SIMULINK**  
PRODUCT FAMILY

Event-Based Modeling  
Physical Modeling  
Systems Engineering  
Real-Time Simulation and  
Testing



**APPLICATION PRODUCTS**

- Aerospace
- Automotive
- Computational Finance
- Code Verification
- Control Systems
- FPGA, ASIC, and SoC Development
- Image Processing and Computer Vision
- Robotics and Autonomous Systems
- Radar
- RF and Mixed Signal
- Signal Processing
- Test and Measurement
- Wireless Communications

Code Generation  
Application Deployment  
Verification, Validation, and Test

# Applications and product family

## MATLAB<sup>®</sup> PRODUCT FAMILY

### MATLAB

MATLAB Copilot

### Parallel Computing

Parallel Computing Toolbox  
MATLAB Parallel Server

### AI, Data Science, and Statistics

Deep Learning Toolbox  
Statistics and Machine Learning Toolbox  
Curve Fitting Toolbox  
Text Analytics Toolbox

### Math and Optimization

Optimization Toolbox  
Global Optimization Toolbox  
Symbolic Math Toolbox  
Mapping Toolbox  
Partial Differential Equation Toolbox

### Reporting and Database Access

Database Toolbox  
MATLAB Report Generator

### Code Generation

MATLAB Coder  
Embedded Coder  
HDL Coder  
HDL Verifier

Fixed-Point Designer

GPU Coder

### Application Deployment

MATLAB Compiler  
MATLAB Compiler SDK  
MATLAB Production Server  
MATLAB Web App Server

### Verification, Validation, and Test

Requirements Toolbox  
MATLAB Test

## SIMULINK<sup>®</sup> PRODUCT FAMILY

### Simulink

### Event-Based Modeling

Stateflow  
SimEvents

### Physical Modeling

Simscape  
Simscape Battery  
Simscape Driveline  
Simscape Electrical  
Simscape Fluids  
Simscape Multibody

### Real-Time Simulation and Testing

Simulink Real-Time  
Simulink Desktop Real-Time

### Reporting

Simulink Report Generator

### Systems Engineering

System Composer  
Requirements Toolbox

### Code Generation

Simulink Coder  
Embedded Coder  
DDS Blockset  
AUTOSAR Blockset  
C2000 Microcontroller Blockset  
Fixed-Point Designer  
Simulink PLC Coder  
Simulink Code Inspector  
DO Qualification Kit (for DO-178)

IEC Certification Kit (for ISO 26262 and IEC 61508)

HDL Coder  
HDL Verifier

### Application Deployment

Simulink Compiler

### Verification, Validation, and Test

Requirements Toolbox  
Simulink Check  
Simulink Coverage  
Simulink Design Verifier  
Simulink Fault Analyzer  
Simulink Test  
Polyspace Bug Finder  
Polyspace Bug Finder Server  
Polyspace Code Prover  
Polyspace Test  
Polyspace Access  
Polyspace Code Prover Server  
Polyspace Client for Ada  
Polyspace Server for Ada

## APPLICATION PRODUCTS

### Wireless Communications

Communications Toolbox  
5G Toolbox  
LTE Toolbox  
WLAN Toolbox  
Bluetooth Toolbox  
Satellite Communications Toolbox  
Wireless HDL Toolbox  
Wireless Testbench

### Radar

Radar Toolbox  
Phased Array System Toolbox  
Sensor Fusion and Tracking Toolbox  
Mapping Toolbox

### Automotive

Model-Based Calibration Toolbox  
Powertrain Blockset  
Vehicle Dynamics Blockset  
Automated Driving Toolbox  
IEC Certification Kit (for ISO 26262 and IEC 61508)  
Vehicle Network Toolbox  
AUTOSAR Blockset  
RoadRunner  
RoadRunner Asset Library  
RoadRunner Scenario  
RoadRunner Scene Builder  
Simulink 3D Animation

### Aerospace

Aerospace Blockset  
Aerospace Toolbox  
UAV Toolbox  
DO Qualification Kit (for DO-178)  
Simulink 3D Animation

### RF and Mixed Signal

Antenna Toolbox  
RF Toolbox  
RF PCB Toolbox  
RF Blockset  
Mixed-Signal Blockset  
SerDes Toolbox  
Signal Integrity Toolbox

### Control Systems

Control System Toolbox  
System Identification Toolbox  
Predictive Maintenance Toolbox  
Robust Control Toolbox  
Model Predictive Control Toolbox  
Fuzzy Logic Toolbox  
Simulink Control Design  
Simulink Design Optimization  
Reinforcement Learning Toolbox  
C2000 Microcontroller Blockset  
Motor Control Blockset

### Test and Measurement

Data Acquisition Toolbox  
Instrument Control Toolbox  
Image Acquisition Toolbox  
Industrial Communication Toolbox  
Vehicle Network Toolbox  
ThingSpeak

### Signal Processing

Signal Processing Toolbox  
DSP System Toolbox  
Audio Toolbox  
Wavelet Toolbox  
DSP HDL Toolbox

### Image Processing and Computer Vision

Image Processing Toolbox  
Computer Vision Toolbox  
Lidar Toolbox  
Medical Imaging Toolbox  
Vision HDL Toolbox  
Image Acquisition Toolbox

### Computational Finance

Datafeed Toolbox  
Database Toolbox  
Econometrics Toolbox  
Financial Toolbox  
Financial Instruments Toolbox  
Risk Management Toolbox  
Spreadsheet Link (for Microsoft Excel)

### Computational Biology

Bioinformatics Toolbox  
SimBiology

### Code Verification

Polyspace Bug Finder  
Polyspace Bug Finder Server  
Polyspace Code Prover  
Polyspace Test  
Polyspace Access  
Polyspace Code Prover Server  
Polyspace Client for Ada  
Polyspace Server for Ada

### Robotics and Autonomous Systems

Automated Driving Toolbox  
Robotics System Toolbox  
UAV Toolbox  
Navigation Toolbox  
ROS Toolbox  
Sensor Fusion and Tracking Toolbox  
RoadRunner  
RoadRunner Asset Library  
RoadRunner Scenario  
RoadRunner Scene Builder  
Simulink 3D Animation

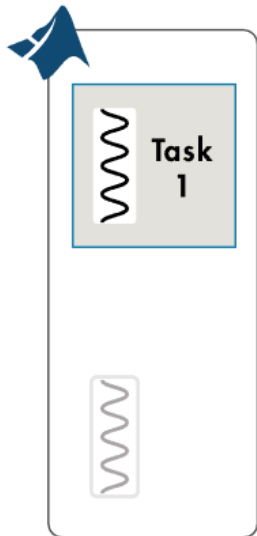
### FPGA, ASIC, and SoC Development

HDL Coder  
HDL Verifier  
Deep Learning HDL Toolbox  
Wireless HDL Toolbox  
Vision HDL Toolbox  
DSP HDL Toolbox  
Fixed-Point Designer  
SoC Blockset

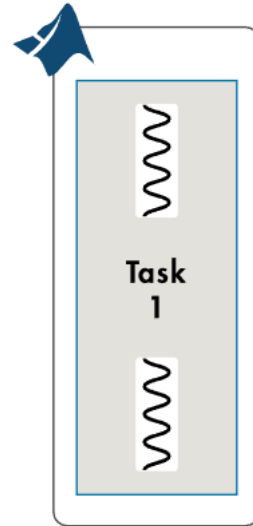
# Selected application use cases

# Take advantage of multicore CPUs with built-in multithreading and explicit parallelism using Parallel Computing Toolbox

**No Parallelization:  
Serial**

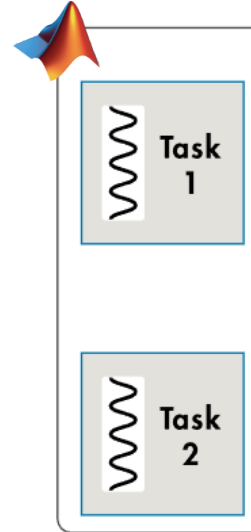


**Implicit Parallelization  
(Thread-Based)**



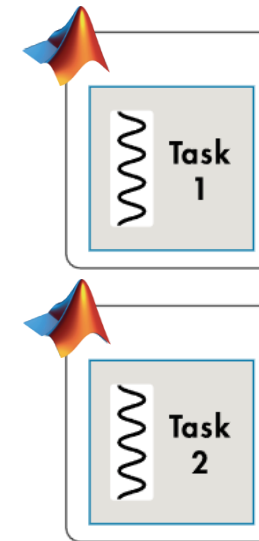
Shorter computation time for a task that can take advantage of implicit multithreading

**Explicit Parallelization  
(Thread-Based)**

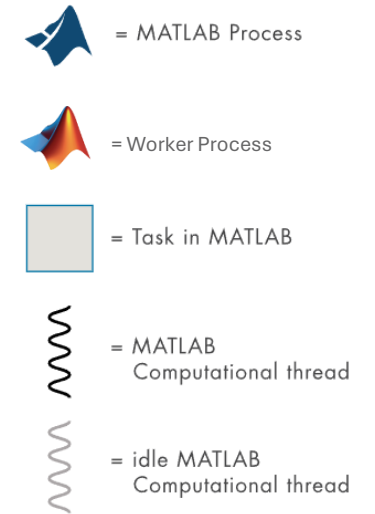


More tasks completed in the same computation time

**Explicit Parallelization  
(Process-Based)**



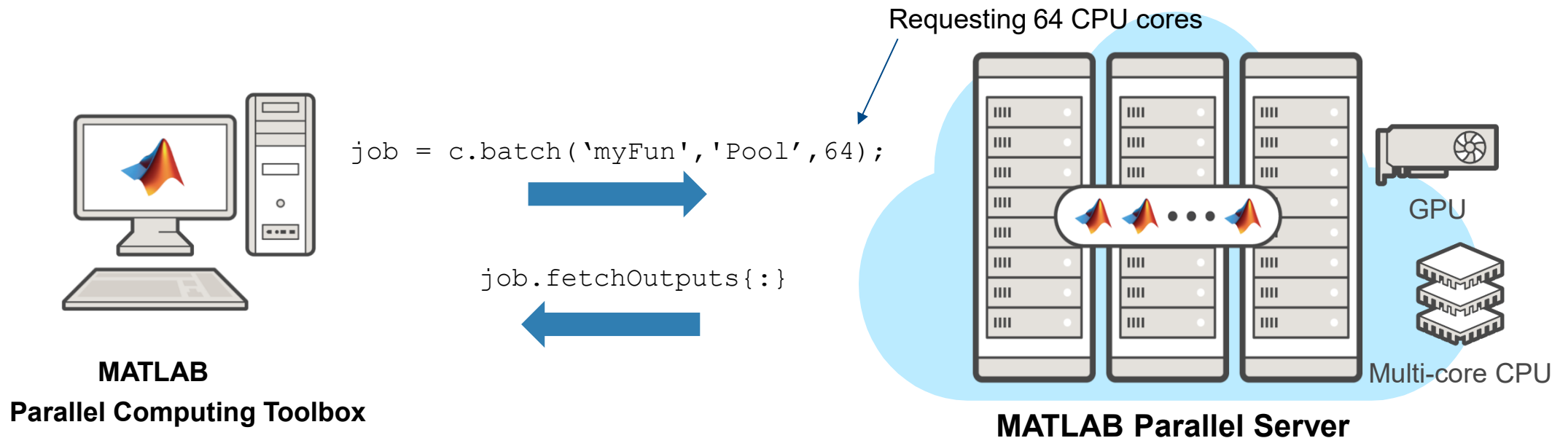
More tasks completed in the same computation time



Assumes a machine with two physical cores

# MATLAB Parallel Server

...Without changing the code, you can run the same applications on clusters or clouds (using MATLAB Parallel Server).

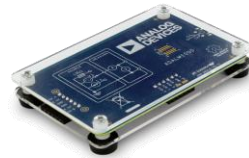


**No recoding for distributed memory! MATLAB uses all requested cores even if on different nodes!**

# MATLAB Connects to Your Hardware and Test Equipment

## Instrument Control Toolbox

Benchtop and PXI test instruments from Keysight, Tektronix, R&S, NI and others



## Data Acquisition Toolbox

Plug-in data acquisition devices from NI, Measurement Computing and advanced sound card support



## MATLAB

built-in and downloadable support for communicating with a wide range of devices like Arduino and Raspberry Pi.



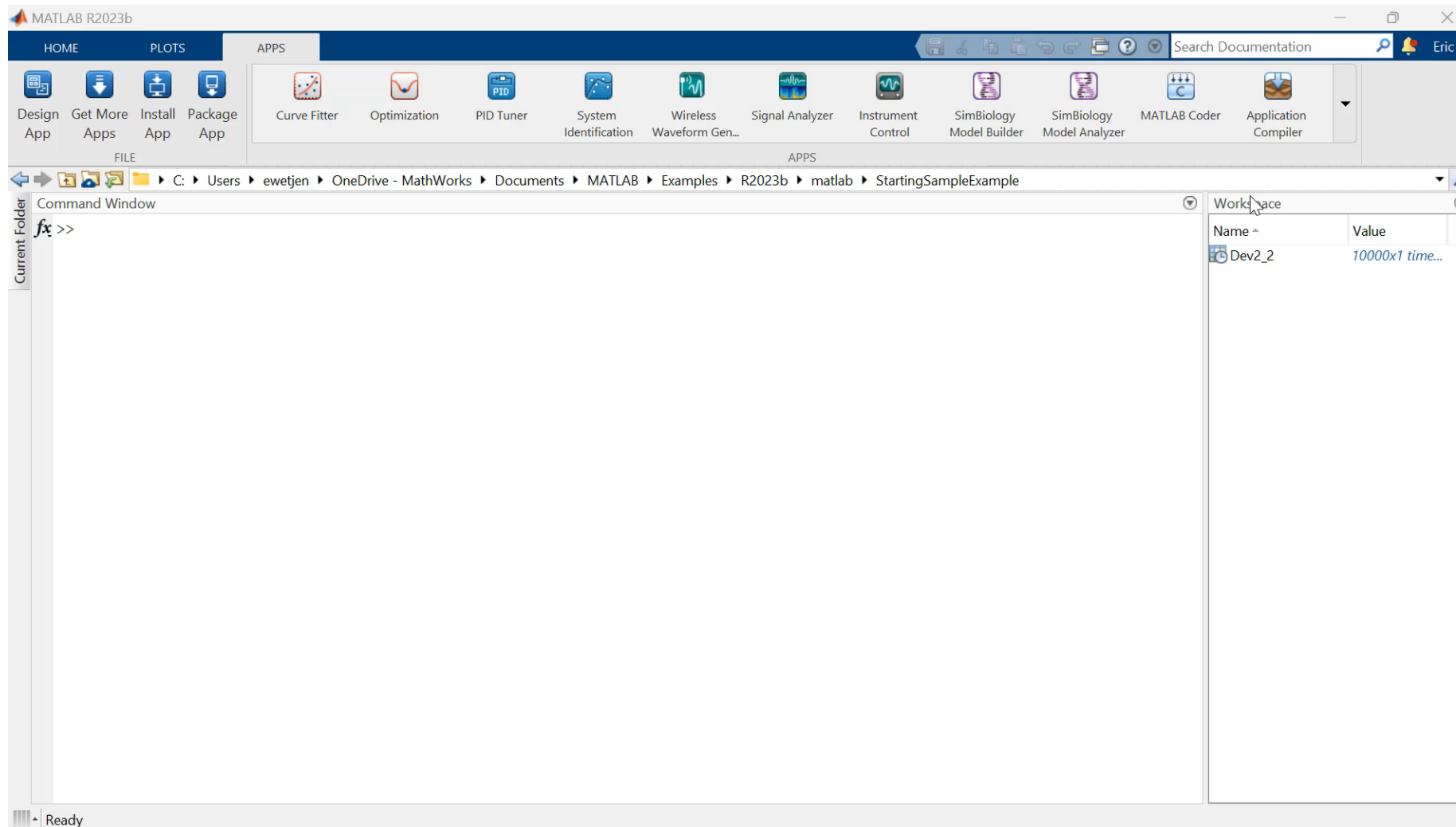
## Image Acquisition Toolbox

Industrial and scientific cameras



# Analog Input Recorder Video Demo

[Video Demo: Automate Data Acquisition in MATLAB Using the Analog Input Recorder App - MATLAB](#)



# MATLAB Copilot – AI assistant optimized for MATLAB

New product from MathWorks

You can:

- Chat with MATLAB Copilot
- Autocomplete code
- Use Copilot Actions in the Editor

The screenshot displays the MATLAB R2025a environment. The main editor window shows a MATLAB script with the following code:

```

19 ylabel('X(t)')
20
21 % Compute the magnitude of the signal
22 Y = fft(X);
23
24 % Compute the frequency axis
25 f = Fs*(0:(L/2))/L;
26
27 % Plot the magnitude of the FFT
28 P2 = abs(Y/L); % Two-sided spectrum
29 P1 = P2(1:L/2+1); % Single-sided spectrum
30 P1(2:end-1) = 2*P1(2:end-1); % Convert amplitude
  
```

The Copilot menu is open, showing the following options:

- Ask Copilot (Generate code inline) - Ctrl+Shift+P
- Generate Test (Generate test for the current file)
- Explain Code (Explain selected code in chat)
- Generate Comments (Generate code comments for the selected code in editor)
- Copilot Settings (Configure MATLAB Copilot)

The Copilot Chat window on the right displays the following explanation:

**Explanation**

- Signal Creation:** A signal is created with a DC offset and two sinusoidal components (50 Hz and 120 Hz).
- Noise Addition:** Random noise is added to simulate a noisy signal.
- FFT Calculation:** The `fft` function computes the Fourier transform of the noisy signal.
- Magnitude Plotting:** The magnitude of the FFT is plotted to visualize the frequency components.

This example illustrates how to use the FFT to analyze the frequency content of a signal in MATLAB.

**References:**

- [fft](#)
- [fft2](#)
- [fft](#)

**Follow-up Prompts:**

- How can I interpret the results of the FFT plot in terms of frequency components?
- What are some common applications of FFT in signal processing?

Ask Copilot

Validate generated output before use.

Editor: 125% UTF-8 CRLF Script Ln 20 Col 1

# R2026a Copilot is built to help with your everyday work

✓ Chat & Learn

The screenshot displays the Simulink R2026a interface for a project named "BatteryManagementSystem". The main workspace shows a block diagram titled "Battery Management System". The diagram includes a "Scenario" block providing a "Signal" to a "BMS Algorithms" block. The "BMS Algorithms" block outputs "StateRequest" and "BMS\_Info" to a "BatteryPlantModel" block. The "BatteryPlantModel" block outputs "KnownSOC" and "Sensors" data. There are feedback loops from "Sensors" back to "BMS Algorithms" and "BMS Cmd" blocks. A "1/z" block is also present in the feedback path.

On the right side, the "Copilot Chat" window is open. It features a header "Simulink Copilot" and a greeting: "Hello! Simulink Copilot is here to assist you in understanding models, tools and errors." Below this is a warning box: "Copilot sometimes produces output that seems accurate but is not. Validate generated output before use. Share feedback on the output to help improve the responses." A button labeled "Shuffle Example Prompts" is visible. Below the button are three example prompts:
 

- Help me understand this model in general.
- What is happening in this model?
- Explain the functionality and purpose of this model.

 An information icon is followed by the text: "Get familiar with Copilot's [limitations](#) before you start". At the bottom of the chat window is a text input field with the placeholder "Ask Simulink Copilot" and a "Deeper Insights" button with a right-pointing arrow. A footer note reads "Validate generated output before use."

# R2026a Copilot is built to help with your everyday work

- ✓ Chat & Learn
- ✓ Understand

The screenshot displays the Simulink R2026a environment. The main workspace shows a Simulink model titled "Battery Management System". The model includes a "Scenario" block that sends a "StateRequest" signal to a "BMS Algorithms" block. The "BMS Algorithms" block outputs "BMS\_Info" and "BMS\_Cmd" signals to a "BatteryPlantModel" block. The "BatteryPlantModel" block contains a "Plant Model" and "Sensors" blocks. The "Plant Model" outputs "KnownSOC" and "Sensors" signals. The "Sensors" block outputs "Sensors" signals back to the "BMS Algorithms" block. The "BMS Algorithms" block also outputs "StateRequest" signals to the "BatteryPlantModel" block.

The Copilot Chat window is open on the right side of the interface. It features a header "Copilot Chat" and a Simulink Copilot icon. The chat content includes a greeting: "Hello! Simulink Copilot is here to assist you in understanding models, tools and errors." Below this is a warning box: "Copilot sometimes produces output that seems accurate but is not. Validate generated output before use. Share feedback on the output to help improve the responses." A button labeled "Shuffle Example Prompts" is visible. Below the button are several example prompts:
 

- ✦ What are your capabilities?
- ✦ Give me a brief overview of control systems theory
- ✦ Tell me about this model
- ℹ Get familiar with Copilot's [limitations](#) before you start

 At the bottom of the chat window is an input field with the placeholder text "Ask Simulink Copilot" and a cursor. To the right of the input field are the options "Deeper Insights" and a send button. A footer note reads "Validate generated output before use."

# R2026a Copilot is built to help with your everyday work

✓ Deeper Insights

The screenshot displays the Simulink R2026a environment. The main workspace shows a Simulink model titled "Battery Management System". The model consists of several interconnected blocks: a "Scenario" block providing a "Signal" input; a "BMS Algorithms" block receiving "StateRequest" and "Sensors" inputs and outputting "BMS\_Info" and "BMS\_Cmd"; a "BatteryPlantModel" block receiving "StateRequest" and "BMS\_Cmd" inputs and outputting "KnownSOC" and "Sensors"; and a "Plant Model" block receiving "KnownSOC" and "Sensors" inputs and outputting "StateRequest". A feedback loop is shown with a gain of 1/r.

On the right side, the "Copilot Chat" window is open, featuring the "Simulink Copilot" header. The chat content includes a welcome message: "Hello! Simulink Copilot is here to assist you in understanding models, tools and errors." Below this is a warning box: "Copilot sometimes produces output that seems accurate but is not. Validate generated output before use. Share feedback on the output to help improve the responses." A "Shuffle Example Prompts" button is present. Three example prompts are listed: "What are your capabilities?", "Explain this model to me.", and "What does this model do?". An information icon is followed by the text: "Get familiar with Simulink Copilot's [limitations](#) before you start." At the bottom of the chat, there is a text input field containing "Describe the equation of the state of charge algorithm", a "Deeper Insights" button, and a send button. A final warning at the bottom of the chat reads: "Validate generated output before use."

# Thank you !

