L'IA au service de la simulation des systèmes

Moubarak Gado, MathWorks
System-level Simulation
Systems complexity is increasing

Increasing system complexity

Common tools used for designing systems

Components:
- Component A
- Component B
- Component C
- Component D
- Component XYZ
Building complex systems with Model-Based Design (MBD)

- Modeling and simulation
- Research
- Requirements
- Continuous test and verification
- Rapid prototyping
- Generation of outputs
  - Production of code
  - Reports
  - Certification artifacts
AI as a new tool to address modeling and simulation challenges

Improve algorithm accuracy:
train AI model using high quality data

Managing complexity:
replace algorithms that would be too difficult to design otherwise

Save time:
replace models that would be too long to simulate
User stories

**Automotive**
- **Mercedes-Benz:** Deep Neural Networks virtual sensors on ECU
- **Vitesco:** Reinforcement Learning based controller for powertrain control
- **Renault:** Estimating Nox emission with Deep Learning

**Aerospace**
- **Lockheed Martin:** Deep Learning based fleet performance optimization

**Robotics & Smart manufacturing**
- **ASTRI:** AI driven digital twin for robotic welding system

**Food and beverage**
- **Coca-Cola:** Virtual Sensor with Machine Learning to improve beverage diagnostics

**Medical**
- **Dutch Epilepsy Clinics Foundation:** Diagnosis of epileptic seizures using Machine Learning

**Automotive**
- **Autonomous Vehicle**
- **Monarch tractor:** AI for camera and sensor data analysis in smart electric tractor

**Energy**
- **Plug Power:** AI based predictive model for fuel cell
AI, simulation and MBD: MATLAB and Simulink for system design workflow

- Easy to use interfaces and apps
- Domain specific examples
- Use AI in your area of expertise without being AI specialist
- Common and collaborative workflow
- Integrate AI models developed in 3rd party frameworks (TensorFlow, PyTorch, …)
Where can you integrate AI into Model-Based Design?

2 categories in general

AI for component modeling

- Speeding up desktop and HIL simulations
- Modeling component dynamics from data when first-principles models cannot be obtained

AI for algorithm development

- Virtual sensor modeling
- Sensor fusion
- Object detection
Observed (major) trends for AI in simulation

**Data synthesis**
Use AI for realistic data generation

**Component modeling**
Use AI for physical environment modeling, Reduced order models

**Algorithm modeling**
Virtual Sensors, Predictive maintenance, ADAS, Signal Processing, Natural Language Processing, Electrification

**Control systems, planning, decision making**
Advanced control algorithms, end to end modeling
Observed (major) trends for AI in simulation

**Data synthesis**
- Measurements are difficult to obtain. I want to do a what if analysis.
- Use AI (generative AI models like GANs, diffusion models, ...) to generate realistic data. Use AI based digital twin to generate data for what if analysis.

**Algorithm modeling**
- I want to design a condition monitoring algorithm and deploy it on hardware. I have enough labeled data.
- Use AI to train a classification algorithm and use automatic code generation tools to deploy it on embedded hardware.

**Control systems, planning, decision making**
- I want to replace several subsystems with a single black box and design a control algorithm, but traditional methods don’t work.
- Use Reinforcement Learning (where simulation and AI combined) to build an end-to-end solution that is self-tuned through a training process.

**Component modeling**
- I have high quality data (inputs and responses) for a physical component. Can I represent the component without using high fidelity tools?
- Use AI to train a reduced order model. You can use leverage AI-based ROM early in design process and high-fidelity model later.
Observed (major) trends for AI in simulation

1. Reduced Order Model
2. Virtual Sensors
3. Reinforcement learning
4. Data synthesis
Application example: Virtual sensors

**What**
A software component that mimics the behavior of a physical sensor by leveraging information available from other measurements and estimate the quantity of interest.

**When**
Physical sensors are impractical, expensive, slow, noisy, unreliable, not feasible, etc.

**How**
Kalman Filters, Grey-Box Models
Lookup tables
Time series modeling
AI (Machine Learning and Deep Learning)
Application example: Virtual sensors

Industries

- Automotive
- Medical devices
- Aerospace
- Industrial Automation and Machinery
- Energy production

Applications

- Control systems
- Electrification
- Fault detection and Predictive maintenance

User Story

Poclain Hydraulics:
Soft Sensors to measure Motor Temperature in real time using Deep Learning
Case study

I need to provide other teams with good trade-off models for faster simulation reuse

AI for component modeling

Replacing a first-principles engine model with an AI-based Reduced Order Model
Application example: Reduced Order Modeling

- Data-driven and adaptive methods: feature extraction, selection
- Reduced computational time and memory, real-time model updating
- Accelerated design process: faster parametric studies and optimization
- More time for exploration and iteration: edge cases, alternative evaluation, faster high-fidelity simulations
- Integration of 2D and 3D models from other tools into system level simulation, enhanced controller design
- Perform hardware-in-the-loop testing without complete system hardware
From first principles models to reduced order models

What is a model?
A simplified abstraction of a system, concept, phenomenon

Physics based model
A useful (not perfect) representation using governing laws of nature that embed concepts of time, space and causality.

Reduced Order Model
Techniques that aim to simplify the original high-fidelity model in a lower-dimensional approximation and extracting most relevant features

Can run faster
From first principles models to reduced order models

- Physics based reduction models
- Model based techniques
- Nonlinear models that are linearized at given operating points
- Data driven approaches (curve fitting, lookup table, AI based models)
Data-driven vs. first-principles modeling

Data-driven models and first-principles models can co-exist

DATA-DRIVEN MODELS
Statistics, optimization, AI

FIRST-PRINCIPLES MODELS
Physics, math, domain knowledge

BLACK BOX
AI-BASED
• Machine learning
• Deep learning
• Reinforcement learning

GREY BOX
PARAMETER ESTIMATION / HYBRID MODELS
• Kalman estimator
• System identification
• Regression

WHITE BOX
PHYSICS-BASED
• Systems/components (electrical, mechanical, algorithms, etc.)
• Can integrate models from other tools such as FEM
Case study: ROM of engine model
Challenges with AI and Simulation for designing complex systems

Choosing best AI technique

How to choose the right AI techniques and algorithms?

Moving from Prototype to production is time-consuming

How can I deploy easily on embedded device easily and get to production faster?

Managing trade-off

How to balance trade-off between complexity and fidelity of the reduced model?

Data

Data preparation is time consuming

Errors and uncertainties

Can I quantify uncertainties? Quantifying errors and uncertainties?

AI model integration

Some teams are using TensorFlow and PyTorch, other are using MATLAB and Simulink. How can the teams work together?

Model validation and verification

How to validate and verify the AI model and its predictions
MATLAB/Simulink for AI and complex system design

Choosing best AI technique

Over 500+ examples using AI for domain-specific applications
Fast and easy experimentation: train and quickly compare different AI models
Choose the best AI technique not only for design, but also for deployment efficiency on intended system

Managing trade-off

Specific tools to save time in every stage of design process

Data

With Simulink, you can integrate easily your AI model (MATLAB, TensorFlow, PyTorch) into the overall simulation environment

AI model integration

Systematically test your model by simulating different test scenario before deploying to production
MATLAB has a growing list of Verification, validation and explainable AI functionality

Model validation and verification

Errors and uncertainties

Development to production made easy

Automatically generate source for embedded AI (CPUs, GPUs or FPGAs)
AI-driven system design workflow

Data Preparation
- Data cleansing and preparation
- Human insight
- Simulation-generated data

AI Modeling
- Model design and tuning
- Hardware accelerated training
- Interoperability

Simulation & Test
- Integration with complex systems
- System simulation
- System verification and validation

Deployment
- Embedded devices
- Enterprise systems
- Edge, cloud, desktop

Simulation-generated data
Human insight
AI workflow – What technique to Consider?

Data Suitability
- Images / Video
- Time Series / Text
- Tabular

Training Data
- 100s
- 1000s
- M+

Training Time
- Seconds
- Minutes
- Hours - Days

Interpretability
- Good
- Better

Footprint / Power Consumption
- Microprocessor
- CPU
- GPU

Accuracy Potential
- Machine Learning
- Deep Neural Networks
Al-driven system design

Data Preparation

AI Modeling

Simulation & Test

Deployment
MATLAB is a Data Manipulation Environment

Spend less time preprocessing and labeling your data

Extract useful features from raw data

Data Simulation & Validation
Use Simulink and Simscape to generate realistic data or build Digital Twin

Use MATLAB and Simulink to create environment models for training agents (Reinforcement Learning)
Feature extraction

**Time series**
- Signal processing techniques
  - Wavelet
- Time, frequency, time/frequency transformation

**Images**
- Deep Learning is now the state of the art
- Specialized feature extraction techniques (HOG, SURF, LBP, …)

**Domain specific feature extraction techniques**

- **Predictive Maintenance** Toolbox
  - DiagnosticFeatureDesigner App
- **Audio** Toolbox
  - audioFeatureExtractor
- **Text** Analytics Toolbox
Example: Reduced order modeling
Design of Experiments & synthetic Data Generation

Input features
- Engine speed (RPM)
- Ignition timing
- Throttle position
- Wastegate valve

Response
- Engine Torque

<table>
<thead>
<tr>
<th>EngTrqReq</th>
<th>EngSpdRPM</th>
<th>SpkAdvOfst</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>60</td>
<td>2000</td>
</tr>
<tr>
<td>2</td>
<td>128</td>
<td>2500</td>
</tr>
<tr>
<td>3</td>
<td>94</td>
<td>2750</td>
</tr>
<tr>
<td>4</td>
<td>111</td>
<td>2875</td>
</tr>
<tr>
<td>5</td>
<td>77</td>
<td>2625</td>
</tr>
<tr>
<td>6</td>
<td>144</td>
<td>2125</td>
</tr>
<tr>
<td>7</td>
<td>85</td>
<td>2563</td>
</tr>
<tr>
<td>8</td>
<td>119</td>
<td>3313</td>
</tr>
<tr>
<td>9</td>
<td>68</td>
<td>2938</td>
</tr>
</tbody>
</table>

Vary model parameters
Log data
Run simulation
AI-driven system design

Data Preparation
AI Modeling
Simulation & Test
Deployment
AI modeling
Multiple approaches

Build with MATLAB

Interoperate with 3rd party framework
Start with a complete set of algorithms, pre-built models and domain specific examples

- Machine Learning
- Deep Learning
- Reinforcement Learning
- Predictive Maintenance
- Bayesian Optimization
- … and more

Flexible modeling approach

- Low code or programmatic
- choose approach that best suits your needs

Pre-builts models

- Audio
- Lidar
- Natural Language Processing
- Computer Vision

Application specific reference examples

Access model from MATLAB Deep Learning Model Hub on Github
Increase productivity using Apps for design and analysis

**Design your AI model**

Deep Network Designer app to build, visualize, and edit deep learning networks

Machine Learning Apps to train machine Learning Models

Reinforcement Learning Designer app to design, train, and simulate agents for existing environments

Run multiple experiments, compare results and optimize your AI model

Experiment Manager app to manage multiple deep learning experiments, analyze and compare results and code
Example: engine model AI based ROM using LSTM
Example: engine model AI based ROM using LSTM

Data-driven Reduced Order Model

Input features
- Engine speed (RPM)
- Ignition timing
- Throttle position
- Wastegate valve

Response
- Engine Torque

LSTM

Neural State Space

nonlinear state function \( f \) and output function \( g \)

\[
\begin{align*}
    & x = f(x, u) \\
    & y = g(x, u)
\end{align*}
\]

State network \( f \)

Output network \( g \)

Nonlinear ARX (NLARX)

<table>
<thead>
<tr>
<th>accuracyResults</th>
<th>RMSE</th>
<th>R^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test 1</td>
<td>1.0252</td>
<td>0.9950</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>accuracyResults</th>
<th>RMSE</th>
<th>R^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test 1</td>
<td>6.2167</td>
<td>0.9443</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>accuracyResults</th>
<th>RMSE</th>
<th>R^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test 1</td>
<td>5.8068</td>
<td>0.9356</td>
</tr>
</tbody>
</table>
AI modelling
Multiple approaches

Data Preparation  AI Modeling  Simulation & Test  Deployment

Build with MATLAB

Interoperate with 3rd party framework
MATLAB interoperates with other frameworks

Interoperate with 3rd party framework

Convert a Deep Learning TensorFlow / PyTorch / ONNX model

Coexecute a TensorFlow or PyTorch or any Python model from MATLAB
Example: Import trained network from TensorFlow

```matlab
YPred = predict(net, X);
Ts = 0.1;
t = Ts*(0:size(X,2)-1)';
plot(t,YPred); hold on, plot(t,Y); hold off
xlabel("Time (s)")
ylabel("y")
```
AI-driven system design

Data Preparation  AI Modeling  Simulation & Test  Deployment
AI is part of a larger system
Integrate your AI model into Simulink

Use **AI libraries blocks** (recommended workflow)

Deep Learning Toolbox
Statistics and Machine Learning Toolbox
System identification Toolbox
Computer Vision Toolbox
Audio Toolbox

Use **MATLAB Function Blocks**
(when no equivalent built-in block)
What if I have Python AI models?

Model Converter (Python → MATLAB)
- TensorFlow
- PyTorch
- ONNX

Co-exécution (Python + MATLAB)

MATLAB Function
Whether you use MATLAB or not, Simulink is an enabler of your AI model

- Use result of simulation to inform model selection and use variants to compare design options
- Test scenarios that would be difficult, expensive, or dangerous to run on hardware or in a physical environment
- Experiment with multiple AI models of an algorithm and rapidly compare tradeoffs in accuracy, model size and on-device performance.
- Uncover system integration issues earlier
Example: AI-based engine reduced-order-model

Integration of trained AI model into Simulink ➔ System-level simulation

Integrate AI models into Simulink for system-level simulation and test
## Integration of trained AI models into Simulink

<table>
<thead>
<tr>
<th>Path</th>
<th>Time Plot (Dark Band = Self Time)</th>
<th>Total Time (s)</th>
<th>Self Time (s)</th>
<th>Number of Calls</th>
</tr>
</thead>
<tbody>
<tr>
<td>AI_ROM</td>
<td></td>
<td>49.440</td>
<td>45.732</td>
<td>142760</td>
</tr>
<tr>
<td>&gt; LSTM</td>
<td></td>
<td>2.643</td>
<td>0.000</td>
<td>0</td>
</tr>
<tr>
<td>&gt; NLARX Sigmoid</td>
<td></td>
<td>0.284</td>
<td>0.000</td>
<td>0</td>
</tr>
<tr>
<td>&gt; Neural State Space</td>
<td></td>
<td>0.195</td>
<td>0.000</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Scope</td>
<td>0.188</td>
<td>0.188</td>
<td>23795</td>
</tr>
<tr>
<td></td>
<td>From Workspace2</td>
<td>0.161</td>
<td>0.161</td>
<td>23794</td>
</tr>
<tr>
<td></td>
<td>Demux</td>
<td>0.128</td>
<td>0.128</td>
<td>05184</td>
</tr>
<tr>
<td></td>
<td>From Workspace1</td>
<td>0.054</td>
<td>0.054</td>
<td>23794</td>
</tr>
<tr>
<td></td>
<td>Prediction_LSTM</td>
<td>0.040</td>
<td>0.040</td>
<td>23794</td>
</tr>
<tr>
<td></td>
<td>Prediction_NeuralSS</td>
<td>0.006</td>
<td>0.006</td>
<td>23794</td>
</tr>
<tr>
<td></td>
<td>Prediction_NLARXSigmod</td>
<td>0.005</td>
<td>0.005</td>
<td>23794</td>
</tr>
<tr>
<td></td>
<td>Prediction_NLARXSVM</td>
<td>0.004</td>
<td>0.004</td>
<td>23794</td>
</tr>
<tr>
<td>&gt; NLARX SVM</td>
<td></td>
<td>0.001</td>
<td>0.000</td>
<td>0</td>
</tr>
<tr>
<td>&gt; Normalize</td>
<td></td>
<td>0.000</td>
<td>0.000</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Cast To Double</td>
<td>0.000</td>
<td>0.000</td>
<td>3</td>
</tr>
<tr>
<td>&gt; Denormalize</td>
<td></td>
<td>0.000</td>
<td>0.000</td>
<td>0</td>
</tr>
</tbody>
</table>
Understanding and Verifying your AI models

Verified AI: Interpretable, explainable
Understanding and Verifying your AI models

Interpretability methods

- **Pre-model**
  - Model-specific
  - Regression
  - Tree Models
  - Naive Bayes
  - Clustering

- **Intrinsic**
  - Predictor Importance (trees & ensemble)
  - Occlusion Sensitivity
    - CAM, Grad-CAM, Activations

- **Post-hoc**
  - Partial Dependence Plot
  - Individual Conditional Exp
  - LIME
  - Shapley values
  - KNN
  - Fuzzy Logic

- **Model-agnostic**
  - PCA
  - tSNE
  - MRMR
  - Correlation
Understanding and Verifying your AI models

Neural Network Verification

R2022b

Deep Learning Toolbox Verification Library
by MathWorks Deep Learning Toolbox Team

Verify and test robustness of deep learning networks
https://www.mathworks.com/help/deeplearning/verification.html

perturbation = 0.01;
XLower = XTest - perturbation;
XUpper = XTest + perturbation;
result = verifyNetworkRobustness(dnet,...
XLower,XUpper,TTest);

summary(result)
  verified 402
  violated 13
  unproven 289

In-distribution ✔  Out-of-distribution ✗
Why MATLAB for Explainable AI?

- Explainable AI plays an important role in Verification and Validation of AI-enabled systems

- MATLAB has a growing list of Explainable AI functionality
  - There is no one-size-fits-all method

- MathWorks is actively engaging with research groups and certification bodies

EUROCAE WG-114 / SAE G-34
Standardization Working Group
“Artificial Intelligence in Aviation”
AI-driven system design

Data Preparation
AI Modeling
Simulation & Test
Deployment
From development to production

Save time and reduce errors

Simplify process, eliminate compatibility issues, deploy on different platforms

End-to-end workflow for designing, testing, and deploying
Deploy to many targets with zero coding errors

- CPU: Any CPU, No Library Needed
- GPU: OneDNN Library
- μC: ARM Compute Library
- FPGA: TensorRT and cuDNN Libraries

Support for TensorFlow Lite
Code generation workflows for embedded target

Through MATLAB

Using apps

Using command line

codegen

Example

Generate generic C++ code through command line for LSTM trained model

```matlab
% Set up a Code Generation Configuration Object for a Static Library
cfg = coder.config('lib');
%cfg编码名为lib
%cfg.targetLang = 'c++';

% Set up a Configuration Object for Deep Learning Code Generation
cfg = coder.config('codegen');
%cfg编码名为codegen
%cfg.targetLang = 'c++';

% Attach the Deep Learning Configuration Object to the Code Generation Configuration Object
%cfg = deepLearningConfig(cfg, 'codegen');

% Generate Source C++ Code By Using codegen
codegen options: files function -args (func_inputs) ... -args (func_inputs)

codegen project
```

Through Simulink (more suitable for MBD workflow)
Getting closer to real hardware prototype

Deployment use cases

Development

Test

Production

Get closer to real hardware
System-level test: Processor-in-the-loop simulation

AI model

Plant

input

output

Code generation from AI algorithm

Deployed code communicates with simulated plant (host computer)

Deploy and validate your embedded AI algorithm on real production processor
System-level test: Hardware-in-the-loop simulation

SIMULINK

Engine AI-based ROM example
Increasing software quality with MATLAB Test

- Code quality dashboard
- Advanced coverage
- Equivalence testing

**Code Coverage Report**

The code coverage report provides a detailed analysis of the source code covered by the tests.

**Overall Coverage Summary**

Summary of the code coverage metrics for all source files.

<table>
<thead>
<tr>
<th>Coverage Metric</th>
<th>Executable</th>
<th>Missed</th>
<th>Code Coverage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>3</td>
<td>0</td>
<td>100%</td>
</tr>
<tr>
<td>Statement</td>
<td>92</td>
<td>15</td>
<td>80.0%</td>
</tr>
<tr>
<td>Decision</td>
<td>36</td>
<td>5</td>
<td>86.13%</td>
</tr>
<tr>
<td>Condition</td>
<td>30</td>
<td>6</td>
<td>80%</td>
</tr>
<tr>
<td>MCC/MDC</td>
<td>15</td>
<td>6</td>
<td>66%</td>
</tr>
</tbody>
</table>

**Test Manager**

- Test Manager: All Tests in Current Project
- 70 Tests: 64 Passed, 4 Failed, 2 Incomplete
- Test Details: Expand All

**MATLAB®**

- MATLAB® Compiler estimates the current branch shortest path from
  - MATLAB® code file to:
  - MATLAB® code file
  - MATLAB® code file
  - MATLAB® code file

**NET, Java, Python, C/C++**

- MATLAB® Test™
- MATLAB® Compiler SDK™
- MATLAB® Coder™
Link to Requirements Verification
Simulink Test
Develop, manage, and execute simulation-based tests

**Test Manager**
- Author, manage, organize tests
- Execute simulation, equivalence and baseline tests
- Review, export, report

**Test Harnesses**
- Isolate Component Under Test
- Synchronized, simulation test environment

**Test Authoring**
- Specify test inputs, expected outputs, and tolerances
- Construct complex test sequences and assessments

---

**Examples**

**Test Browser**

**Test Results**

**Test Harness**

**Test Manager**

**Test Harnesses**

**Test Authoring**

---

**Signal Editor**

**Temporal Assessments**

**Examples**
How to optimized performance in hardware constrained environment?

Compression

Pruning and Model Compression techniques to reduce model size and speed up inference
How to optimized performance in hardware constrained environment?

Data Preparation | AI Modeling | Simulation & Test | Compression | Deployment

- **Quantization**
  - Conversion from floating point to fixed point

- **Pruning**
  - Removing unimportant parts of the network

- **Projection**
  - Project learnable parameters into a lower dimensional space

**Deep Network Quantization App**
- Taylor Pruning
- Projection

**Pruning and Model Compression techniques to reduce model size and speed up inference**
AI model compression workflow

1. Determine Hardware Constraints
2. Select Model
3. Simplify Model
4. Quantize Model Parameters
5. Deploy & Integrate

Select Model
- Size aware model selection

Simplify Model
- Projection and Pruning

Quantize Model
- Deep Learning Toolbox Model Quantization Library
- Fixed-Point Designer
Conclusion

- Many promising application in the intersection between AI and Simulation

- Combining AI and simulation for designing complex system is all about tradeoffs

- MATLAB and Simulink
  - Run simulation of AI model at the system level and collect metric
  - Refine model and implement the optimal AI technique
  - Balance AI accuracy and deployment efficiency
  - One toolchain for seamless interaction between AI and simulation
  - Select and implement the optimal AI technique balance
Key takeaways

Data Preparation
- Data cleansing and preparation
- Human insight
- Simulation-generated data

AI Modeling
- Model design and tuning
- Hardware accelerated training
- Interoperability

Toolboxes for domain-specific pre/post-processing

Simulation & Test
- Integration with complex systems
- System simulation
- System verification and validation

Simulink blocks for AI models make integration easy

Compression
- Size aware model selection
- Compression techniques (pruning, projection)
- Quantization

Model compression techniques to reduce model size and speed up inference

Deployment
- Embedded devices
- Enterprise systems
- Edge, cloud, desktop

Code generation for embedded targets
Thank you!

Q&A