Production AI, Digital Twinning, and Simulation for Industrial Companies

Paola Jaramillo Garcia
Pjaramil@mathworks.com
Application Engineer, MathWorks
• Signal Processing
• Computer Vision
• Sensor Fusion

Toon Weyens
tweyens@mathworks.com
Application Engineer, MathWorks
• Machine Learning and Deep Learning
• Enterprise Integration
• High-performance Computing
Let us help you getting started with predictive maintenance systems

**MATLAB onramp** ([link](#))
- Get started quickly with the basics of MATLAB.

**Simulink onramp** ([link](#))
- Get started quickly with the basics of Simulink.

**Deep Learning onramp** ([link](#))
- Get started quickly using deep learning methods to perform image recognition.

**Practical Data Science with MATLAB** ([link](#))
- Learn with a flexible schedule through online courses

**Data Science portal** ([link](#))
- Explore data; build machine learning models; do predictive analytics

---

**Introduction to Predictive Maintenance** ([link](#))
- Remaining Useful Life

**Enterprise Integration & IT Systems** ([link](#))

**Use MATLAB in the Cloud or Container** ([link](#))
- powered by: [AWS](#), [Azure](#)
We can simulate this!
Baker Hughes Develops Predictive Maintenance Software for Gas and Oil Extraction Equipment Using Data Analytics and Machine Learning

**Challenge**
Develop a predictive maintenance system to reduce pump equipment costs and downtime

**Solution**
Use MATLAB to analyze nearly one terabyte of data and create a neural network that can predict machine failures before they occur

**Results**
- Savings of more than $10 million projected
- Development time reduced tenfold
- Multiple types of data easily accessed

“MATLAB gave us the ability to convert previously unreadable data into a usable format; automate filtering, spectral analysis, and transform steps for multiple trucks and regions; and ultimately, apply machine learning techniques in real time to predict the ideal time to perform maintenance.”
- Gulshan Singh, Baker Hughes

Link to user story
Key Takeaways

- AI has tremendous potential

- Industry struggles to implement it:
  - Data scientists are scarce
  - Common AI platforms are not well-suited for scaling with industrial data

- To bridge AI and industrial product development:
  - Simulation as low-cost source for machine failure data
  - Team collaboration with platform for interoperability
  - Scaling up to more powerful hardware and deployment with automatic code generation
Example **Engineering Problem**: Develop and operationalize a machine learning model to predict failures in industrial pumps

Current system requires operator to manually monitor operational metrics for anomalies. Their expertise is required to detect and take preventative action

- **Process Engineer**
  - Develops models in MATLAB and Simulink

- **System Architect**
  - Deploys and operationalizes model on cloud

- **Operator**
  - Makes operational decisions based on model output

![AWS and Kibana icons]
The Need for Large-Scale Streaming

Predictive Maintenance

Increase Operational Efficiency
Reduce Unplanned Downtime

More applications require near real-time analytics

Jet engine: ~800TB per day
Turbine: ~ 2 TB per day

Medical Devices

Patient Safety
Better Treatment Outcomes

Connected Cars

Safety, Maintenance
Advanced Driving Features

Car: ~25 GB per hour
 Modeling approach

1. Access and Explore Data
   - Files
   - Databases
   - Sensors

2. Preprocess Data
   - Working with Messy Data
   - Data Reduction/Transformation
   - Feature Extraction

3. Develop Predictive Models
   - Model Creation e.g. Machine Learning
   - Parameter Optimization
   - Model Validation

4. Integrate with Production Systems
   - Desktop Apps
   - Enterprise Scale Systems
   - Embedded Devices and Hardware

5. Visualize Results
   - 3rd party dashboards
   - Web apps

Process Engineer
Physics of Triplex Pump

- Crankshaft drives three plungers
  - Each 120 degrees out of phase
  - One chamber always discharging
  - Three types of failures
Use sensor data from pump to identify levels of failure
Build digital twin and generate sensor data
Simulate data with many failure conditions

Access Data

```python
ens = simulationEnsembleDatastore(location)
```

```
ens = 
simulationEnsembleDatastore with properties:

DataVariables: [25×1 string]
IndependentVariables: [0×0 string]
ConditionVariables: [0×0 string]
SelectedVariables: [25×1 string]
ReadSize: 1
NumMembers: 702
LastMemberRead: [0×0 string]
Files: [702×1 string]
```

Run parallel simulations

Store data on HDFS
Preprocess data

```matlab
data = synchronize(Flow, Pressure, Current, t, 'linear');
data = normalize(data, 'center');
```
Represent signal information
Represent signal information
Signal processing

```matlab
[Spectrum,Frequencies] = psppectrum(data.Flow);
[pLow,pHigh] = bounds(Spectrum);
fPeak = Frequencies(Spectrum == pHigh);
qPeak2Peak = peak2peak(data.Flow);
qCrest = peak2rms(data.Flow);
qRMS = rms(data.Flow);
qMAD = mad(data.Flow);
[pKur,fKur] = pkurtosis(data.Flow);
qSkewness = skewness(data.Flow);
qMean = mean(data.Flow);
qVar = var(data.Flow);
```
Develop Predictive Models in MATLAB

Process Engineer

<table>
<thead>
<tr>
<th>Time</th>
<th>LeakFault</th>
<th>BlockingFault</th>
<th>BearingFault</th>
<th>FaultType</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 sec</td>
<td>2.8472</td>
<td>-0.1477</td>
<td>1.8000</td>
<td>All</td>
</tr>
<tr>
<td>0.001 sec</td>
<td>-0.1498</td>
<td>-0.4207</td>
<td>1.3103</td>
<td>Bearing &amp; Blocking</td>
</tr>
<tr>
<td>0.002 sec</td>
<td>0.6511</td>
<td>1.6521</td>
<td>-0.5357</td>
<td>Leak</td>
</tr>
<tr>
<td>0.003 sec</td>
<td>0.1469</td>
<td>-0.2775</td>
<td>1.0074</td>
<td>All</td>
</tr>
<tr>
<td>0.004 sec</td>
<td>-0.6480</td>
<td>0.7065</td>
<td>-0.8876</td>
<td>Blocking</td>
</tr>
<tr>
<td>0.005 sec</td>
<td>-0.8165</td>
<td>-0.5434</td>
<td>-0.3079</td>
<td>Blocking</td>
</tr>
<tr>
<td>0.006 sec</td>
<td>-1.0061</td>
<td>1.2083</td>
<td>0.0661</td>
<td>Bearing</td>
</tr>
<tr>
<td>0.007 sec</td>
<td>1.0125</td>
<td>-1.9098</td>
<td>-0.7027</td>
<td>Leak &amp; Blocking</td>
</tr>
</tbody>
</table>

Label Faults

Scale

tt = tall(ds);
tt = preprocessData(tt);
model = TreeBagger(50,tt,'Event');

Evaluating tall expression using the Spark Cluster:
- Pass 1 of 2: Completed in 11 sec
- Pass 2 of 2: Completed in 2.3333 min
Evaluation completed in 2.6167 min

Represent Signals

Train Model

Validate Model
Develop Predictive Models in MATLAB

3

Develop Predictive Models

Process Engineer

Remaining Useful Life (Regression)

Type of Fault (Classification)
Estimate Remaining Useful Life

Model Coeff: $\phi = 2.1396 \quad \theta = -0.038836 \quad \beta = 0.13184$

\[ S(t) = \phi + \theta(t) e^{(\beta(t)t+\epsilon(t)+\frac{\sigma}{2})} \]
Develop Machine Learning Models

Process Engineer

Develop Predictive Models

MathWorks
**Batch Processing:** Build and test model on historical and simulated data

**Stream Processing:** Apply model to sensor data in near real-time
Develop a Stream Processing Function

Streaming Function

```matlab
function new_state = streamingFunction(data, old_state)

Preprocess signals
[data, features] = preprocessData(data);

Predict faults
[Leak, Blocking, Bearing] = predictFaultValues(features);
FaultType = predictFault(features);
[RUL, Model] = predictUpdateRUL(data.Timestamp, data.Flow, 500);

Update state
new_state = updateState(data, old_state);

Write results
writeResults(Leak, Blocking, Bearing, FaultType, RUL, Model)
end
```

Process each window of data as it arrives

Previous state

Current window of data to be processed
Integrate Analytics with Production Systems

System Architect

Edge

Generate telemetry

Production System

Analytics Development

Apache Kafka

MATLAB Production Server

Worker processes

Request Broker

Connector

State Persistence

Storage Layer

MATLAB Compiler SDK

Debug

Package & Deploy

Model

Algorithm Developers

Business Decisions

Presentation Layer

End Users

kibana

Integrate with Production Systems
Integrate with Production Systems

4

Generate telemetry

System Architect

Debug your streaming function on live data

Production System

Analytics Development

Azure

Connector

Apache Kafka

MATLAB Compiler SDK

Debug

MATLAB

Algorithm Developers

Business Decisions

kibana

Presentation Layer

End Users
Connecting MATLAB Production Server to Kafka

- Lightweight connector that feeds a single Kafka topic to a MATLAB function
- Simple publisher library for MATLAB for writing output to a results stream

Connector Features:
- Deploy as a simple micro-service with Docker
- Drive everything through config (no programming outside of MATLAB)
- Group data into time windows and pass to MATLAB as a timetable
- Use Kafka’s check-pointing (i.e. at-least-once)
Share with the team

Share code with System Architect

Source Control

Review results with Operator

.pdf, html, LaTeX
Complete your application

Production System

Apache Kafka

MATLAB Production Server
Worker processes
Request Broker

State Persistence

Storage Layer

Analytics Development

MATLAB Compiler SDK
MATLAB

Debug
Package & Deploy
Algorithm Developers

Business Decisions

kibana
Presentation Layer

End Users

Edge
Generate telemetry

Generate telemetry

System Architect

Integrate with Production Systems
Complete Your Application
Key Takeaways

- AI has tremendous potential

- Industry struggles to implement it:
  - Data scientists are scarce
  - Common AI platforms are not well-suited for scaling with industrial data

- To bridge AI and industrial product development:
  - Simulation as low-cost source for machine failure data
  - Team collaboration with platform for interoperability
  - Scaling up to more powerful hardware and deployment with automatic code generation

We can help you with these challenges