Scaling up MATLAB Analytics with Kafka and Cloud Services

Pallavi Kar
The Need for Large-Scale Streaming

Predictive Maintenance

*Increase Operational Efficiency*
*Reduce Unplanned Downtime*

Many 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
Example Problem – How’s my driving?

Analyzing vehicle data to score driving habits

- A group of MathWorks employees installed an OBD dongle in their car that monitors the on-board systems
- Data is streamed to the cloud where it is aggregated and stored
Case Study: Stream based Analytics on drive data with MATLAB
Challenges in building such a system...

Edge Devices

- Aggregate data from multiple devices
- Build accurate algorithms
- Integrate algorithms into production systems
- Run analytics on multiple streams of data

Analytics Development

MATLAB
Algorithm Developers

Business Decisions

Business Systems
End Users

MATLAB EXPO 2018
Solution: MATLAB Production Server and Streaming engine

- MATLAB Analytics to REST APIs
- Serves concurrent requests from web clients
- Scale by adding workers

Integrate

Data sources

MATLAB Production Server

Worker processes

Request Broker

Access

- Enterprise Application
- Mobile / Web Application
- 3rd party dashboard

Deploy

Analytics Development

MATLAB

Compiler SDK

Package

Code / test

Integrate

kafka

Event Hub

Kinesis

Scale and secure

MATLAB EXPO 2018
Fleet Analytics Architecture

Edge Devices

- API Gateway
- AWS Lambda
- kafka

Production System

- Kafka Connector
- Storage Layer

Analytics Development

- MATLAB
- MATLAB Compiler
- Algorithm Developers

Business Decisions

- Business Systems
- Power BI
- Qlik
- Tableau

Connectors provided by MathWorks
Development to Deployment Workflow

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
     - MATLAB
     - Excel
     - C/C++
     - Java
   - Enterprise Scale Systems
   - Embedded Devices and Hardware

5. Visualize Results
   - 3rd party dashboards
     - Spotfire
     - Tableau
     - Qlik
     - Power BI
   - Web apps
Accessing data in MATLAB

Edge Devices
- API Gateway
- AWS Lambda
- Kafka

Production System
- MATLAB Production Server
- MATLAB Analytics
- Kafka Connector

Analytics Development
- MATLAB
- Algorithm Developers

Business Decisions
- Power BI
- Qlik
- Tableau

End Users
- Business Systems
The Data: Timestamped messages with JSON encoding

```
{
  "vehicles_id": "55a3fd0069702d5b41000000",
  "time": "2015-07-13T18:01:35.000Z",
  "kc": 1975.0, "kff1225": 100.65293, "kff125a": 110.36619,
}

{
  "vehicles_id": "55a3fe3569702d5c5c000020",
  "time": "2015-07-13T18:01:53.000Z",
  "kc": 2000.0, "kff1225": 109.65293, "kff125a": 115.36619,
}

{
  "vehicles_id": "55a4193569702d115b000001",
  "time": "2015-07-12T19:04:04.000Z",
  "kc": 2200.0, "kff1225": 112.65293, "kff125a": 112.36619,
}
```
Access a Sample of Data

Raw Data

<table>
<thead>
<tr>
<th>timestamp</th>
<th>value</th>
<th>key</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-Jan-2015</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22:12:23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-Jan-2015</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22:12:24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-Jan-2015</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22:12:25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-Jan-2015</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22:12:26</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Timetable

<table>
<thead>
<tr>
<th>trip_id</th>
<th>VIN</th>
<th>kft1001</th>
<th>kft1006</th>
<th>kft1020</th>
<th>kft1211</th>
<th>kft1221</th>
<th>kft1222</th>
<th>kft1223</th>
<th>kft122a</th>
</tr>
</thead>
<tbody>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>17.1000</td>
<td>-84.9323</td>
<td>45.4704</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>59.0434</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>17.1000</td>
<td>-84.9322</td>
<td>45.4704</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>57.8609</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>18.9000</td>
<td>-84.9322</td>
<td>45.4705</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>52.7147</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>18.9000</td>
<td>-84.9321</td>
<td>45.4705</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>51.1983</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>18.0000</td>
<td>-84.9321</td>
<td>45.4706</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>49.1095</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>68.5000</td>
<td>-84.9305</td>
<td>45.4686</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>73.2005</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>56.7000</td>
<td>-84.9304</td>
<td>45.4685</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>75.3612</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>57.6000</td>
<td>-84.9304</td>
<td>45.4683</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>70.7542</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>66.7000</td>
<td>-84.9303</td>
<td>45.4682</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>62.8340</td>
</tr>
</tbody>
</table>
Access and Explore Data

Working with JSON in MATLAB

Reading JSON data

Functions like `jsonencode` and `jsondecode` make it very simple.

Read a file:

```matlab
fname = 'sample.json';
fid = fopen(fname);
raw = fread(fid,inf);
str = char(raw);
fclose(fid);
```

Decode JSON data to MATLAB datatypes:

```matlab
data_struct = jsondecode(str)
```

Convert to table for easy operations.
Ad Hoc Access to Data from MATLAB

1. Access and Explore Data

Access the data in S3

Bring up the AthenaClient

```matlab
athenaClient = aws.athena.Client();
athenaClient.Database = 'trainingdata';
athenaClient.initialize();
```

Create a query and submit

```matlab
athenaClient.submitQuery('SELECT * FROM "trainingdata"."sampledata" limit 100', 's3://fleettrainingdata')
```

Fetch data as a table for easy analysis

```matlab
ds = datastore('s3://fleettrainingdata/*.csv');
ds.NumHeaderLines = 2;
data = table(ds);
```

Your usual MATLAB workflow goes here
Pre-processing and Feature Engineering

**Preprocess Data**

**Matlab Datatype**

- Data: Designed to organize and work with time series data.
- Components of a Timetable:
  - Time (offset from a reference time)
  - Duration
  - Variable Names
- Datetime
- Temperature
- Humidity

**Timetable Manipulation**

- Access Data: These return the same array:
  - `tt.Temperature`
  - `tt(:, 'Temperature')`
- Add a New Variable:
  - `var = zeros(height(tt), 1)`

**Data Cleaning**

- Smooth Data:
  - `B = smoothdata(A, method)`
  - Methods:
    - `movmean`, `movmedian`, `gaussian`, `lowess`, `loess`, `lroess`, `sgolay`

**Merge Timetables**

- Synchronize multiple timetables to a common time vector.
- `tt = synchronize(tt1, tt2, ..., ttN)`

**Missing Data**

- Find Missing Values:
  - `TF = ismissing(tt)`
- Fill Missing Values:
  - `tt = fillmissing(tt, method)`
  - Replace missing values with calculated from nearby points:
    - `previous`, `next`, `nearest`, `linear`, `spline`, `pchip`
- Remove Rows Containing Missing Values:
  - `tt = rmmissing(tt)`

**Big Data**

- Tall arrays extend MATLAB functions to work on data too big to load into memory.
- Create a “tall” timetable:
  - `t = tall(ds)`
  - `Create a datastore that points to the data:
    - `ds = datastore('*.csv')`
  - `Create a tall table from the datastore:
    - `tt = tall(ds)``
Develop a Preprocessing Function

Timetable

2 Preprocess Data

✓ Clean up
✓ Enrich
✓ Restructure

MATLAB EXPO 2018
Building predictive models

**Selecting an Algorithm**

**MACHINE LEARNING**

- Supervised Learning
  - Classification: Support Vector Machines, Linear Regression, GLM, Discriminant Analysis, Naive Bayes, Nearest Neighbor, Neural Networks
  - Regression: K-Means, K-Medoids, Fuzzy C-Means
  - Clustering: SVR, GPR, Hierarchical, Gaussian Mixture, Ensemble Methods
- Unsupervised Learning

**ITERATIVE MACHINE LEARNING PROCESS**

1. Capture Sensor Data
2. Extract Features
3. Run Model
4. Evaluate Model

**Algorithm**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Prediction Speed</th>
<th>Training Speed</th>
<th>Memory Usage</th>
<th>Required Tuning</th>
<th>General Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic Regression (and Linear SVM)</td>
<td>Fast</td>
<td>Fast</td>
<td>Small</td>
<td>Minimal</td>
<td>Good for small problems with linear decision boundaries</td>
</tr>
<tr>
<td>Decision Trees</td>
<td>Fast</td>
<td>Fast</td>
<td>Small</td>
<td>Some</td>
<td>Good generalist, but prone to overfitting</td>
</tr>
<tr>
<td>(Nonlinear) SVM (and Logistic Regression)</td>
<td>Slow</td>
<td>Slow</td>
<td>Medium</td>
<td>Some</td>
<td>Good for many binary problems, and handles high-dimensional data well</td>
</tr>
<tr>
<td>Nearest Neighbor</td>
<td>Moderate</td>
<td>Minimal</td>
<td>Medium</td>
<td>Minimal</td>
<td>Lower accuracy, but easy to use and interpret</td>
</tr>
<tr>
<td>Naive Bayes</td>
<td>Fast</td>
<td>Fast</td>
<td>Medium</td>
<td>Some</td>
<td>Widely used for text, including spam filtering</td>
</tr>
<tr>
<td>Ensembles</td>
<td>Moderate</td>
<td>Slow</td>
<td>Varies</td>
<td>Some</td>
<td>High accuracy and good performance for small- to medium-sized datasets</td>
</tr>
<tr>
<td>Neural Network</td>
<td>Moderate</td>
<td>Slow</td>
<td>Medium to Large</td>
<td>Lots</td>
<td>Popular for classification, compression, recognition, and forecasting</td>
</tr>
</tbody>
</table>
Develop a Predictive Model in MATLAB

What happens when data is large?
Submit Big Data jobs from MATLAB on HADOOP & SPARK

MATLAB workers on worker nodes in the cluster
- MDCS workers (working from MATLAB)
Develop a Predictive Model

MATLAB EXPO 2018
Develop and Deploy a Stream Processing Function

Edge Devices

API Gateway
AWS Lambda

Kafka Connector

Production System

Kafka

MATLAB Production Server
MATLAB Analytics

Analytics Development
MATLAB Compiler SDK
Algorithm Developers

Business Decisions
Power BI
Qlik
Tableau
End Users
Business Systems

Integrate with Production Systems

MATLAB EXPO 2018
A quick Intro to Stream Processing

- **Batch Processing** applies computation to a finite sized historical data set that was acquired in the past

- **Stream Processing** applies computation to an unbounded data set that is produced continuously
Why stream processing?

Why stream processing?

- **Edge Processing with MATLAB Coder**
- **Stream Processing with MATLAB Production Server**
- **Near Real-time decisions**
  - **Stream Processing**
  - **Time critical decisions**
  - **Big Data processing on historical data**

- **Value of data to decision making**
  - **Preventive/Predictive**
  - **Actionable**
  - **Reactive**

- **Time**
  - **Seconds**
  - **Minutes**
  - **Hours**
  - **Days**
  - **Months**

- **Integrate with Production Systems**

- **Today’s example focuses here**

- **Kafka**
- **Event Hub**
- **Kinesis**
- **MDCS, Compiler with Hadoop/Spark**

MATLAB EXPO 2018
Connecting MATLAB Production Server to Kafka

- Kafka client for MATLAB Production Server feeds topics to functions deployed on the server

- Each consumer process feeds one topic to a specified function

- Configurable batch of messages passed as a MATLAB Timetable

- Drive everything from a simple config file
  - No programming outside of MATLAB!
### Input Table

<table>
<thead>
<tr>
<th>Event Time</th>
<th>Vehicle</th>
<th>RPM</th>
<th>Torque</th>
<th>Fuel Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>18:01:10</td>
<td>55a3fd</td>
<td>1975</td>
<td>100</td>
<td>110</td>
</tr>
<tr>
<td>18:10:30</td>
<td>55a3fe</td>
<td>2000</td>
<td>109</td>
<td>115</td>
</tr>
<tr>
<td>18:05:20</td>
<td>55a3fd</td>
<td>1980</td>
<td>105</td>
<td>105</td>
</tr>
<tr>
<td>18:10:45</td>
<td>55a3fd</td>
<td>2100</td>
<td>110</td>
<td>100</td>
</tr>
<tr>
<td>18:30:10</td>
<td>55a419</td>
<td>2000</td>
<td>100</td>
<td>110</td>
</tr>
<tr>
<td>18:35:20</td>
<td>55a419</td>
<td>1960</td>
<td>103</td>
<td>105</td>
</tr>
<tr>
<td>18:20:40</td>
<td>55a3fe</td>
<td>1970</td>
<td>112</td>
<td>104</td>
</tr>
<tr>
<td>18:39:30</td>
<td>55a419</td>
<td>2100</td>
<td>105</td>
<td>110</td>
</tr>
<tr>
<td>18:30:00</td>
<td>55a3fe</td>
<td>1980</td>
<td>110</td>
<td>113</td>
</tr>
<tr>
<td>18:30:50</td>
<td>55a3fe</td>
<td>2000</td>
<td>100</td>
<td>110</td>
</tr>
</tbody>
</table>

### Output Table

<table>
<thead>
<tr>
<th>Time window</th>
<th>Vehicle</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>18:00:00</td>
<td>55a3fd</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>55a3fe</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>55a419</td>
<td>...</td>
</tr>
<tr>
<td>18:10:00</td>
<td>55a3fd</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>55a3fe</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>55a419</td>
<td>...</td>
</tr>
<tr>
<td>18:20:00</td>
<td>55a3fd</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>55a3fe</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>55a419</td>
<td>...</td>
</tr>
<tr>
<td>18:30:00</td>
<td>55a3fd</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>55a3fe</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>55a419</td>
<td>8</td>
</tr>
</tbody>
</table>

Streaming data is treated as an unbounded Timetable.
Develop a Stream Processing Function in MATLAB

Process each window of data (input table) as it arrives

Current window of data to be processed

Current score (pointer)

Previous state (pointer)

MATLAB EXPO 2018
4 Integrate with Production Systems

Develop a Stream Processing Function in MATLAB

Process each window of data (input table) as it arrives

```
function current_data = preprocessData(current_data)
    % Preprocess and perform calculations
    current_data = removeMissingData(current_data);
    % Remove records with all missing data
    current_data = removeMissingData(current_data,'MinNumMissing',width(current_data)-1);
    % Smooth and calculate approximate gradients
    current_data.Speed = smoothData(current_data.Speed);
    current_data.D1 = [0;diff(current_data.D1)];
    current_data.D2 = [0;0;diff(current_data.D2)];

function new_state = calculateScores(car_id, current_data, old_state)
    % Preprocess and perform calculations
    current_data = preprocessData(current_data);
    % Predict driving events
    current_data = predictEvents(current_data);
    % Count events for each ten second window
    countsByTime = countEvents(current_data);
    % Write discrete data to MongoDB
    updateResultsStore(car_id, countsByTime, resultsStore);
    % Update new state
    new_state = updateState(countsByTime, old_state);
end
```
Develop a Stream Processing Function in MATLAB

Process each window of data (input table) as it arrives

Use the model you created with Classification Learner App
Develop a Stream Processing Function in MATLAB

Process each window of data (input table) as it arrives

Database is updated with results of driver scoring:
- Count of events by type and location
- Stored in MongoDB instance

MATLAB EXPO 2018
Operationalize Analytics into Production Systems

Production System

Edge Devices

Analytics Development

Business Decisions

Integrate with Production Systems
MATLAB Compiler SDK Workflow

Integrate with Production Systems

MATLAB Production Server

MATLAB Compiler SDK

MATLAB Analytics

Archive information

Files required for your archive to run

AvailableChan... computeDriving... dropCollection... jsonPreProces... remove.m
ByteDecoder.m count.m find.m KafkaMessage... StateStore.m
calculateScor... createCollect... insert.m MessageParser... update.m
changeTimetab... DefaultParser.m isopen.m mon...
close.m distinct.m JSONParser.m MongoStore.m

MATLAB EXPO 2018
Debug a Stream Processing Function in MATLAB
Debug a Stream Processing Function in MATLAB

Integrate with Production Systems
Tie in your Dashboard Application

Production System

MATLAB Production Server

MATLAB Analytics

Storage Layer

Kafka

Kafka Connector

API Gateway

AWS Lambda

Amazon Web Services

Business Decision Systems

Business Decisions

End Users

Power BI

Qlik

Tableau

MATLAB EXPO 2018
Scalable Analytics with Enterprise BI Tools

TIBCO Spotfire

Tableau
MATLAB based applications in Production Level Ecosystem

Data
- Databases: DynamoDB, MongoDB, SQL Server, Cassandra, Cosmos DB
- Cloud Storage: Azure Blob
- Streaming: AWS Kinesis, Azure IoT Hub

Analytics
MATLAB Production Server
Request Broker

Business System
- Dashboards: Qlik, Tableau, Microsoft Power BI, Spotfire
- Web: Microsoft IIS, Apache Tomcat, WebSphere
- Custom Apps: Google Cloud Platform, Azure, Amazon Web Services, Rackspace, OpenStack, VMware

Platform
MATLAB EXPO 2018
Volkswagen Data Lab develops driver recognition algorithms with MATLAB

Develop technology building block for tailoring car features and services to individual

- Need to identify individual drivers based on their driving behavior using collected data

Challenges

- Accuracy despite low training data
- Robustness despite environmental conditions
- Computing time

Data sources

- Logged CAN bus data and travel record

Source: „Connected Car – Fahrererkennung mit MATLAB“ Julia Fumbarev, Volkswagen Data Lab, MATLAB EXPO Germany, June 27, 2017, Munich Germany
Key Takeaways

➢ Spend your time understanding the data and designing algorithms

➢ You can run MATLAB on any development engine, desktop, server or cloud

Solution architects

➢ MATLAB can connect directly to your data repositories

➢ MATLAB can deploy within your ecosystem and on platform of your choice using MATLAB Production Server
Resources to learn and get started

- Data Analytics with MATLAB
- Statistics and Machine Learning Toolbox
- Database Toolbox
- Mapping Toolbox

- MATLAB Production Server
- MATLAB Compiler SDK
- MATLAB with TIBCO Spotfire
- MATLAB with Tableau
- MATLAB with MongoDB

MATLAB EXPO 2018
Your feedback is valued.
Please complete the feedback form provided to you.
<table>
<thead>
<tr>
<th>Data Analytics Applications</th>
<th>Controls and Embedded Systems</th>
<th>Signal Processing Systems</th>
<th>Robotics and Autonomous Systems</th>
</tr>
</thead>
</table>
| Predictive Maintenance Using MATLAB and Simulink  
Amit Doshi, MathWorks | Designing Efficient Power Electronics Systems Using Simulation  
Vivek Raju, MathWorks  
Naga Chaitrapani Penmaraju, MathWorks | Designing and Testing Voice Interfaces through Microphone Array Modeling, Audio Prototyping, and Text Analytics  
Vidya Viswanathan, MathWorks | Automated Driving Development with MATLAB and Simulink  
Manesar Reddy, MathWorks |

**Exhibition Break**

**Using Fleet Analytics and MATLAB to Build Strategies for SS-VI Development**  
Shubham Garg, Honda

**Lithium-Ion Battery Parameter Estimation for HIL, SIL, and MIL Validation**  
Thayalan Shanmugam, RNTBCI

**Verifying the Hardware Implementation of Automotive Radar Signal Processing with MATLAB**  
Sainath K and Shashank Venugopal, NXP

**Autonomous Drive**  
Gopinath Chidambaram, L&T Technology Services

**Exhibition Break**

**Scaling up MATLAB Analytics with Kafka and Cloud Services**  
Pallavi Kar, MathWorks

**Full Vehicle Simulation for Electrification and Automated Driving Applications**  
Prasanna Deshpande, MathWorks  
R V

**SG: What's Behind the Next Generation of Mobile Communications?**  
Tabrez Khan, MathWorks

**Demystifying Deep Learning**  
Dr. Amid Anandkumar, MathWorks

**Exhibition Break**

**Developing Optimization Algorithms for Real-World Applications**  
Dr. Lakshminarasayan Ravichandran, MathWorks  
Gautam Ponnapa PG, MathWorks

**Verification and Validation of High- Integrity Systems**  
Chethan CU, MathWorks  
Vaishnavi H R, MathWorks

**Designing and Integrating Antenna Arrays with Multi-Function Radar Systems**  
Shashank Kulkarni, Ph.D., MathWorks  
Swathi Balir, MathWorks

**Deploying Deep Neural Networks to Embedded GPUs and CPUs**  
Rishu Gupta, Ph.D., MathWorks

**Exhibition Break**

**Tackling Big Data Using MATLAB**  
Aika Nair, MathWorks

**Generating Industry Standards Production C Code Using Embedded Coder**  
Rajat Arora, MathWorks  
Durvesh Kulkarni, MathWorks

**Designing and Verifying Digital and Mixed-Signal Systems**  
Aniruddha Dayatu, MathWorks

**Developing Algorithms for Robotics and Autonomous Systems**  
Dhirendra Saigh, MathWorks  
Abhishek Roy, MathWorks
THANK YOU