MATLAB EXPO 2018

Scaling up MATLAB Analytics with Kafka and Cloud Services

Christoph Stockhammer
# Agenda

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
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

MATLAB EXPO 2018
Example Problem – How’s my driving?

- 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

- We would like to use this data to score the driving habits of participants
Example: Fleet Analytics with MATLAB
Fleet Analytics Architecture

**Edge Devices**
- API Gateway
- AWS Lambda
- Kafka

**Production System**
- Kafka Connector
- MATLAB Production Server
- MATLAB Analytics
- Storage Layer

**Analytics Development**
- MATLAB Compiler SDK
- MATLAB
- Algorithm Developers

**Business Decisions**
- Business Systems
- End Users

**Production System**
- MATLAB EXPO 2018
The first step is to clean up the incoming data.
The Data: Timestamped messages with JSON encoding

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

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

```
{
    "vehicles_id": {"$oid":"55a4193569702d115b000001"},
    "time": {"$date":"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>1 value</th>
<th>2 key</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-Jan-2015 22:12:23</td>
<td>&quot;id&quot;: &quot;55a41cb069702d115b0598e0&quot;, &quot;trip_id&quot;: &quot;55a41cb069702d115b059edc&quot;</td>
<td></td>
</tr>
<tr>
<td>15-Jan-2015 22:12:24</td>
<td>&quot;id&quot;: &quot;55a41cb069702d115b0598e1&quot;, &quot;trip_id&quot;: &quot;55a41cb069702d115b059edc&quot;</td>
<td></td>
</tr>
<tr>
<td>15-Jan-2015 22:12:25</td>
<td>&quot;id&quot;: &quot;55a41cb069702d115b0598e2&quot;, &quot;trip_id&quot;: &quot;55a41cb069702d115b059edc&quot;</td>
<td></td>
</tr>
<tr>
<td>15-Jan-2015 22:12:26</td>
<td>&quot;id&quot;: &quot;55a41cb069702d115b0598e3&quot;, &quot;trip_id&quot;: &quot;55a41cb069702d115b059edc&quot;</td>
<td></td>
</tr>
</tbody>
</table>

Timetable

<table>
<thead>
<tr>
<th>trip_id</th>
<th>VIN</th>
<th>kT1001</th>
<th>kT1006</th>
<th>kT120</th>
<th>kT121</th>
<th>kT122</th>
<th>kT123</th>
<th>kT125a</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>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>57.8609</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>18.5000</td>
<td>-84.9322</td>
<td>45.4705</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>-84.9322</td>
<td>45.4705</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>51.1983</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>18.9000</td>
<td>-84.9321</td>
<td>45.4706</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>49.1095</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>18.5000</td>
<td>-84.9305</td>
<td>45.4686</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>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>70.7542</td>
</tr>
<tr>
<td>55a3fe356...</td>
<td>55a3fe356...</td>
<td>56.7000</td>
<td>-84.9303</td>
<td>45.4682</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>62.8340</td>
</tr>
</tbody>
</table>
Develop a Preprocessing Function

Timetable

Preprocess data

\[ t = \text{sortrows}(t); \]
\[ t = \text{rmmissing}(t,'\text{MinNumMissing}',\text{width}(t)-2); \]

Perform windowed calculations

\[ t.\text{Speed} = \text{movmedian}(t.\text{SpeedGPS},3); \]
\[ t.D1 = [0;\text{diff}(t.\text{SpeedGPS})]; \]

\[ [\text{tmin},\text{tmax}] = \text{bounds}(t.\text{time}); \]
\[ \text{tnew} = \text{tmin}:\text{seconds}(10):\text{tmax}; \]
\[ \text{countsByTime} = \text{retine}(t(:,:,\text{Event}),\text{tnew},@\text{histcounts}); \]
Ad Hoc Access to Data from MATLAB

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
Develop a Predictive Model
Everything you need to develop a predictive model is found in MATLAB

3: Develop Predictive Models

- Label Events
- Represent Signals
- Train Model
- Validate Model
- Scale Up

MATLAB EXPO 2018
Develop a Predictive Model in MATLAB
Integrate Analytics with Production Systems

Integrate Analytics with Production Systems

Edge Devices

Production System

Analytics Development

Business Decisions

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.

MATLAB EXPO 2018
Why stream processing?

- Near Real time decisions
- Time critical decisions
- Big Data processing on historical data

Edge Processing with MATLAB Coder

Stream Processing with MATLAB Production Server

Value of data to decision making
- Actionable
- Reactive
- Historical

Today's example focuses here

Integrate with Production Systems

MATLAB Distributed Computing Server, MATLAB Compiler

Integrate with Production Systems

MATLAB EXPO 2018
Streaming data is treated as an unbounded Timetable

**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>18:10:00</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>
Introducing MATLAB Production Server

Data

- Databases
  - DynamoDB
  - 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
- Web
  - IIS
  - Apache Tomcat
  - WebSphere
- Custom Apps
  - OSIsoft PI System
  - AWS Kinesis
  - Kafka
  - MQTT
  - Azure IoT Hub

Platform

MATLAB EXPO 2018
MATLAB Production Server is an application server that publishes MATLAB code as APIs.

Integrate with Production Systems

Data sources

Analitics Development

MATLAB

Compiler SDK

Package

Code / test

MATLAB Production Server

Worker processes

Request Broker

Scale and secure

Deploy

Access

Enterprise Application

Mobile / Web Application

3rd party dashboard

MATLAB EXPO 2018
Connecting MATLAB Production Server to Kafka

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

- Configurable batch of messages passed as a MATLAB Timetable

- Each consumer process feeds one topic to a specified function

- Drive everything from a simple config file
  - No programming outside of MATLAB!
Develop and Deploy a Stream Processing Function

Edge Devices

Integrate with Production Systems

Production System

Analytics Development

Business Decisions

Edge Devices

AWS Lambda

API Gateway

kafka

MATLAB Production Server

MATLAB Compiler SDK

Algorithm Developers

Business Systems

End Users

Storage Layer

Integrate with Production Systems

MATLAB EXPO 2018
Integrate with Production Systems

Develop a Stream Processing Function in MATLAB

Process each window of data as it arrives

Current score

Previous state

Current window of data to be processed

Develop a Streaming Function

```matlab
function new_state = calculateScore(car_id, current_data, old_state, resultsStore)

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

Apply your pre-processing algorithm
Develop a Stream Processing Function in MATLAB

Use the model you created with Classification Learner App

function new_state = calculateScores(car_id, current_data, old_state, resultsStore)
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

function current_data = predictEvents(current_data)
% Predict events for current data based on machine learning model
    predictorNames = {'kff1005','kff1006','kff125a','k10','kff1249','Speed','D1','D2',...
                      'kff1001','kff1220','kff1221','kff1222','kff1223',... 
                      'k47','kff124d'};
    predictors = current_data(:,predictorNames);
    mdl = load('machineLearningModel.mat');
    current_data.Event = predict(mdl.model,predictors);
end
Develop a Stream Processing Function in MATLAB

Develop a Streaming Function

```matlab
function new_state = calculateScores(car_id, current_data, old_state, resultsStore)

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
```

Update Mongo database
- Count of events by type and location
- Results of driver scoring
Debug a Stream Processing Function in MATLAB

Production System

Analytics Development

Kafka Connector

MATLAB Compiler SDK

Algorithm Developers

kafka

Storage Layer

End Users

Integrate with Production Systems

Edge Devices

MATLAB EXPO 2018
Debug a Stream Processing Function in MATLAB
Tie in your Dashboard Application

Integrate with Production Systems

Edge Devices

- API Gateway
- AWS Lambda
- kafka

Production System

- Kafka Connector
- MATLAB Production Server
- MATLAB Analytics
- Storage Layer

Analytics Development

- MATLAB Compiler SDK
- Algorithm Developers

Business Decisions

- Power BI
- Qlik
- Spotfire
- Tableau

End Users

Integrate with Production Systems

MATLAB EXPO 2018
Complete Your Application

Visualize Results

Fleet Summary
- Aggressive update: Enabled

Fleet Statistics
- Total Events: 133351

Acceleration/Deceleration Events, 2014 - 2017
- Aggressive acceleration: 24215 Events
- Speeding: 1632 Events
- Safe deceleration: 3979 Events
- Moderate deceleration: 1969 Events
- Acceleration: 25363 Events
Scalable Analytics with Enterprise BI Tools

TIBCO Spotfire

Tableau

MATLAB EXPO 2018
Key Takeaways

- MATLAB connects directly to your data so you can quickly design and validate algorithms

- The MATLAB language and apps enable fast design iterations

- MATLAB Production Server enables easy integration of your MATLAB algorithms with enterprise production systems

- You to spend your time understanding the data and designing algorithms
Resources to learn and get started

- Data Analytics with MATLAB
- MATLAB Production Server
- MATLAB Compiler SDK
- Statistics and Machine Learning Toolbox
- Database Toolbox
- Mapping Toolbox
- MATLAB with TIBCO Spotfire
- MATLAB with Tableau
- MATLAB with MongoDB