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
     - Desktop Apps
     - Enterprise Scale Systems
       - AWS Kinesis
     - Embedded Devices and Hardware

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

Production System

Analytics Development

Business Decisions

Edge Devices

API Gateway

AWS Lambda

kafka

Kafka Connector

Production System

MATLAB Production Server

MATLAB Analytics

Storage Layer

Storage

End Users

Algorithm Developers

Business Systems

Power BI

Qlik

Spotfire

Tableau

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 and Explore Data

Access a Sample of Data

Raw Data

<table>
<thead>
<tr>
<th>timestamp</th>
<th>1</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>15-Jan-2015 22:12:23</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-Jan-2015 22:12:24</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-Jan-2015 22:12:25</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15-Jan-2015 22:12:26</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

✓ Decode JSON data
✓ Create Timetable

Timetable

<table>
<thead>
<tr>
<th>trip_id</th>
<th>VIN</th>
<th>kff1001</th>
<th>kff1006</th>
<th>kff1008</th>
<th>kff11006</th>
<th>kff1120</th>
<th>kff1121</th>
<th>kff1122</th>
<th>kff1123</th>
<th>kff1123a</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</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>NaN</td>
<td>59.0434</td>
</tr>
<tr>
<td>2</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>NaN</td>
<td>57.8609</td>
</tr>
<tr>
<td>3</td>
<td>55a3fe356...</td>
<td>18.0000</td>
<td>-84.9322</td>
<td>45.4705</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>NaN</td>
<td>52.7147</td>
</tr>
<tr>
<td>4</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>NaN</td>
<td>51.1983</td>
</tr>
<tr>
<td>5</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>NaN</td>
<td>49.1095</td>
</tr>
<tr>
<td>6</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>NaN</td>
<td>73.2005</td>
</tr>
<tr>
<td>7</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>NaN</td>
<td>75.3612</td>
</tr>
<tr>
<td>8</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>NaN</td>
<td>70.7542</td>
</tr>
<tr>
<td>9</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>NaN</td>
<td>NaN</td>
<td>62.8340</td>
</tr>
</tbody>
</table>
Develop a Preprocessing Function

Preprocess Data

Timetable

```
t = sortrows(t);
t = rmmissing(t,'MinNumMissing',width(t)-2);
```

Perform windowed calculations

```
t.Speed = movmedian(t.SpeedGPS,3);
t.D1 = [0;diff(t.SpeedGPS)];
```

```
[tmin,tmax] = bounds(t.time);
tnew = tmin:seconds(10):tmax;
countsByTime = retime(t(:,['Event'],tnew,@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

MATLAB EXPO 2018
Everything you need to develop a predictive model is found in MATLAB

Develop Predictive Models

1. Label Events
2. Represent Signals
3. Train Model
4. Validate Model
5. Scale Up

MATLAB EXPO 2018
Develop a Predictive Model in MATLAB
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?

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

Stream Processing with MATLAB Production Server

Today’s example focuses here

Integrate with Production Systems

Edge Processing with MATLAB Coder

C/C++

Value of data to decision making

Preventive / Predictive

Actionable

Reactive

Historical

Time

Real-Time

Seconds

Minutes

Hours

Days

Months

Time critical decisions

Big Data processing on historical data

MATLAB Distributed Computing Server, MATLAB Compiler

Kinesis

Event Hub

Spark

Hadoop

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>

MATLAB Function

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>18:20:00</td>
<td>55a419</td>
<td>...</td>
</tr>
<tr>
<td>18:10:00</td>
<td>55a3fd</td>
<td>7</td>
</tr>
<tr>
<td>18:20:00</td>
<td>55a3fe</td>
<td>3</td>
</tr>
<tr>
<td>18:20:00</td>
<td>55a419</td>
<td>...</td>
</tr>
<tr>
<td>18:30:00</td>
<td>55a3fd</td>
<td>...</td>
</tr>
<tr>
<td>18:30:00</td>
<td>55a3fe</td>
<td>5</td>
</tr>
<tr>
<td>18:30:00</td>
<td>55a419</td>
<td>8</td>
</tr>
</tbody>
</table>

Streaming data is treated as an unbounded Timetable.
Introducing MATLAB Production Server

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
  - OSIsoft PI System
  - Kafka

Platform

Google Cloud Platform

19

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

Integrate with Production Systems

Data sources

Worker processes

Request Broker

Scale and secure

Deploy

Analytics Development

MATLAB

MATLAB Compiler SDK

Package

Code / test

Integrate

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

Integrate with Production Systems

Edge Devices

Production System

Analytics Development

Algorithm Developers

Business Decisions

End Users

API Gateway

AWS Lambda

kafka

Kafka Connector

MATLAB Production Server

MATLAB Compiler SDK

Business Systems

MATLAB

Power BI

Qlik

Tableau

Spotfire

Storage Layer
Develop a Stream Processing Function in MATLAB

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
```

Process each window of data as it arrives
Current score
Previous state
Current window of data to be processed
Develop a Stream Processing Function in MATLAB

Develop a Streaming Function

```matlab
function current_data = preprocessData(current_data)
    % Preprocess and perform calculations
    current_data = preprocessData(current_data);
    % Remove records with all missing data
    current_data = rmmissing(current_data,'MinNumMissing',width(current_data)-1);
    % Smooth and calculate approximate gradients
    current_data.Speed = movmedian(current_data.kff1001,5);
    current_data.D1 = [0;diff(current_data.kff1001)];
    current_data.D2 = [0;0;diff(current_data.kff1001,2)];
end
```

Integrate with Production Systems

Apply your pre-processing algorithm
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
```

Use the model you created with Classification Learner App

```matlab
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

Integrate with Production Systems

MATLAB EXPO 2018
Debug a Stream Processing Function in MATLAB
4. Integrate with Production Systems

Debug a Stream Processing Function in MATLAB

MATLAB EXPO 2018
4  Integrate with Production Systems

Tie in your Dashboard Application

Edge Devices

- AWS Lambda
- API Gateway

Production System

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

Analytics Development

- MATLAB Compiler SDK
- MATLAB
- Algorithm Developers

Business Decisions

- Business Systems
- Power BI
- Qlik
- Tableau

End Users

MATLAB EXPO 2018
Complete Your Application
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