Data Analytics & Machine Learning in Healthcare

solutions for biotech & medical devices industry
Arvind Ananthan
A Brief introduction

• M.S.E.E – Signal/Image Processing, Machine Learning

• 14+ years of technical, marketing, and business expertise

• Medical Devices Industry Manager since 2012
  • Worldwide responsibility
  • FDA and regulatory affairs
Cheaper, Plenty, Everywhere

1. Low-cost and powerful sensors.
2. Lots of data. Health data.
3. Storage and computing is cheaper.

... but how about using smart algorithms to process all this data to achieve better patient outcomes?

MATLAB & Simulink form a technical foundation for this fusion and transformation.
Medtronic launches smartphone App enabling diabetics to remotely monitor their condition

APPLE’S RESEARCHKIT IS A NEW WAY TO DO MEDICAL RESEARCH

GOOGLE TAKES ON THE CHALLENGE OF MAKING ROBOT SURGERY SAFER

Quest Diagnostics launches patient analytics tool for healthcare providers
Medtronic launches smartphone App enabling diabetics to remotely monitor their condition.

Quest Diagnostics launches patient analytics tool for healthcare providers.

IBM estimates images account for 90% of all medical data.

IBM's Watson deepens engagement with Apple, adds new customers.

Biogen CEO says tech aims to make wearable and ingestible devices safer.

Quest Diagnostics launches patient analytics tool for healthcare providers.
Focus for today...

- **Examples of smart analytics for healthcare**
  - #1 Bio-signal acquisition and ECG processing
  - #2 Computer vision in biomedical imaging
  - #3 Machine learning for diagnostic classification

- **System development using Model-Based Design**
  - Quick overview

- **Certification and Regulatory Impact**
  - FDA’s views on modeling and simulation
Bio-signal Acquisition and processing

#1 Stream and process live ECG signal in MATLAB
Connect to low-cost Bitalino hardware using Bluetooth
Process ECG data to identify heart rate and other parameters
Patient Monitoring

Compute Heart Rate
Fine Location of QRS, ST, PR, etc.
Analyzing ECG signals

Using **Wavelets Toolbox**

Explore advanced algorithm ideas on different test ECG signals

Identify candidates for implementation

```matlab
%% Load ECG signal
load tEECG; x = tEECG(1300:5300);
load noisyEECG; x = noisyEECG(1:5000);

%% Create Continuous Wavelet Transform and plot the scalogram
wlets = {'rbio2.8', 'gaus2', 'db3', 'bior3.1', 'gaus8', 'sym8', 'mexh', 'dmey'};
wname = wlets{1}; scales = 10:5:350;
coefs = cwt(x,scales,wname);
s = wscalogram('contour',coefs,'scales',scales,'ydata',x);
```
Analyzing ECG signals

Using **Wavelets Toolbox**

**Implement streaming version** that can run on saved or live ECG signals

```matlab
function [actual_Rlocs, actual_Tlocs, actual_Plocs] =...
    Find QRS peaks using lower scale values (high frequency)...
    r_coeffs = cwt(x,44,'db3');
    cline = abs(r_coeffs);
    mc = max(cline);
    cline(cline<mc*75)=0;
    [~, riocs] = findpeaks(cline);
    Find P and T wave positions approximately using wavelet...
    pt_coeffs = cwt(x,190,'db8');
    pt = abs(pt_coeffs);
    [~, ptlocs] = findpeaks(pt);
    Find the P, R and T locations...
    Lpt = length(ptlocs); Lr = length(riocs);
    ptl = repmat(ptlocs',1,Lr); rtl = repmat(riocs,Lpt,1);
    [~, ptlndx] = min(abs(ptl-rtl));
    validPT = (ptlndx<(Lpt-2) & ptlndx>1);
    actual_Rlocs = rlocs(validPT);
    actual_Tlocs = ptlocs(ptlndx(validPT)+2); % 2nd peak from R peak is the T wave
    actual_Plocs = ptlocs(ptlndx(validPT)-1); % 1 peak behind R peak is the P wave with offset for a regular ECG
```

See MathWorks table for demo
Wearables that detect cardiac arrhythmias

“The **fixed-point** test platform we built with **MATLAB** enabled us to conduct rigorous tests at every stage and automatically identify discrepancies in the results.”

-VivaQuant

The arrhythmia service uses an FDA 510k cleared Holter recorder to non-invasively record a 24-hour or longer three-lead ECG
#2 Measuring Drug Effectiveness from Motion Detection

Object recognition, and tracking

Quantification of movement to position \((x,y)\) and velocity
Pixels $\rightarrow$ Features $\rightarrow$ Object

Detection / Tracking / Classification

Image Pixels

Feature Extraction

Object Detection

Object Tracking

Object Classification
Using Vision System Objects

simple video loop

%%% Set up Input/Output Vision System Objects
vReader = vision.VideoFileReader('ZebraFish.avi');           %<--- Reader
vPlayer1 = vision.VideoPlayer('Position', [20, 400, 700, 400]); %<--- Player

%%% Loop through frames
frameNumber = 0;
while frameNumber < 100
    frameNumber = frameNumber + 1;
    frame = vReader.step();
    frame = imcrop(frame,[225 220 672 235]);  %<--- Standard function call
    vPlayer1.step(frame);
end

%%% Release the objects used
release(vReader)
release(vPlayer1)
Using Vision System Objects

**simple video loop + object detection**

```matlab
%% Set up Vision System Objects
vReader = vision.VideoFileReader('video.mpg');
vPlayer1 = vision.VideoPlayer;
detector = vision.ForegroundDetector('NumTrainingFrames', 10000);
blobAnalyser = vision.BlobAnalysis('AreaOutput', true); % Use a blob analysis block

%% Loop through frames
frameNumber = 0;
while frameNumber < 500
    frameNumber = frameNumber + 1;
    frame = vReader.step();
    frame = imcrop(frame, [225 220 670 225]);
    mask = detector.step(frame);
    [area, centroids, boxes] = blobAnalyser(mask);
    frame_markers = insertMarkers(frame, centroids, 'circle', 12, 'Color', 'g');
    vPlayer1.step(frame_markers);
end

%% Release the objects when done
release(vReader); release(vPlayer1);
```

**Initialize vision objects**

- Foreground Detector
- Centroid Marker
Using Vision System Objects

object detection + object tracking + motion analysis

See MathWorks table for demo
#3 Classify parasite type in blood smear images
Handle large sets of images
Develop and evaluate classifiers
Pixels $\rightarrow$ Features $\rightarrow$ Object

Detection / Tracking / Classification

Image Pixels $\rightarrow$ Feature Extraction $\rightarrow$ Object Detection $\rightarrow$ Object Tracking $\rightarrow$ Object Classification
Machine Learning Workflow

Machine learning uses **data** and **produces** a **program** to perform a **task**

**Train:** Iterate till you find the best model

**Predict:** Integrate trained models into applications
Classification of Parasites in Blood

Step 1. Detect features

Class 1  Class 2  Class 3

Bag of Visual Words
Classification of Parasites in Blood

2. Create Classifiers with Machine Learning App

See MathWorks table for demo
Deploying Applications with MATLAB
MATLAB Beyond the Desktop

Share Applications

Scale Computation

Integrate with Web & Enterprise

Scale Data

Support for Hadoop/Cloud Computing

MATLAB Compiler SDK
MATLAB Production Server

MATLAB Compiler
MATLAB Coder

Parallel Computing Toolbox
MATLAB Distributed Computing Server
The AirSonea device connects to an asthma patient’s smartphone and communicates with wheeze analysis algorithms on iSonea’s server.

“MATLAB enables us to rapidly develop, debug, and test sound-processing algorithms, and **MATLAB Coder** simplifies the process of implementing those algorithms in C.

There’s no other environment or programming language that we could use to produce similar results in the same amount of time.”

- iSonea
Now that you have developed your smart algorithms...

How do you implement and verify in software...

...as part of the full system?
Model-Based Design

for safer and faster medical devices

**RESEARCH**

**REQUIREMENTS**

**DESIGN**
- Environment Models
- Physical Components
- Algorithms

**IMPLEMENTATION**
- C, C++
- VHDL, Verilog
- Structured Text
  - MCU
  - DSP
  - FPGA
  - ASIC
  - PLC

**INTEGRATION**

**TEST AND VERIFICATION**

- Design as Executable Specification
- Requirements Traceability
- Continuous and early Verification
- Document & Report Generation

---

**MathWorks**
Why Model-Based Design?

Weinmann Medical

**Developing and reviewing code for ventilators 50% faster**

“Model-Based Design enabled us to generate 100% of the embedded software... We also spent about 50% less time on internal review, because we worked with the models instead of low-level source code.” – Dr. Florian Dietz, Weinmann

ITK Engineering

**Accelerating dental drill motor development by 2x**

“We completed controller development in about four months. Without Model-Based Design it would have taken at least twice as long.” – Michael Schwarz, ITK Engineering

Philips Healthcare

**Reducing risk and accelerating FPGA design by 80x**

“Design changes that took us a week to hand-code in VHDL and test on hardware can be simulated and verified in 30 minutes with Model-Based Design.”

– Mark van Helvoort, Philips Healthcare
What about certification?

...and the FDA?

Early Interest in MBD

1/1/2009 - 12/31/2011

MBD Paper

2/1/2010

Initial Engagement with OSEL

6/14/2011

1/1/2012

MathWorks focus on Medical Devices

4/1/2013

Collaboration agreement between MathWorks and CDRH

Couple of ODE engineers working on Modeling/Simulation visit MathWorks

5/1/2014

MBD Workshop for 10 ODE/CDRH members (from DAGRID)

9/1/2014

MathWorks visits FDA

3/15/2015

PCLC Workshop in DC

10/12/2015

UVa Virtual Patient Model

4/1/2009

MathWorks visits FDA
MBD as a new approach
January 2010

Model-Based Development: A New Approach to Engineering Medical Software
Arnab Ray and Raoul Jetley

With an increasing number of medical device features being implemented in code, the amount of software that is present in a modern device as well as its complexity and criticality has grown sharply over the years. Existing quality-control regimes for software, dependent as they are on traditional inspection and ad-hoc testing, have proved inadequate to meet many of these challenges. In 1998, close to 8% of blood glucose devices and diabetes insulin pumps, some to be about 18%.

Model-based development (MBD) is an approach that aims at building and maintaining software systems that are modular, understandable, and maintainable. It can be used in the development and quality control phases of software development. Model-based development allows a description of the software in terms of a model, rather than in terms of a set of source code files. The model is then used to generate the source code.

MBD has attained increasing popularity in the aerospace and automotive industry because of how it supports the production of better software.

Arnab Ray is a research scientist at the Fraunhofer Center for Experimental Software Engineering and also an assistant adjunct professor at the department of Computer Science, University of Maryland College Park. Email: Aray@fc-md.umd.edu

Raoul Jetley is a research scientist at the U.S. Food and Drug Administration’s (FDA) Center for Devices and Radiological Health/Office of Science and Engineering Laboratories. Email: Raoul.Jetley@fda.hhs.gov

“Compilation” “Auto-coding”

- Machine/Assembly language
  - Instruction level
    - Load, stores, moves

- Procedural/OO languages
  - “High-level” constructs
    - Guards (guardian loops)

- Executable modeling notations
  - Captures “high-level” design information, portable and re-usable
FDAsuggestsModel-BasedDesignforinfusionpumps
~2011

Infusion Pump Software Safety Research at FDA

- Software Safety
- Model-Based Design of Infusion Pumps
- Generic Infusion Pump Project
- Static Analysis
- Recent Publications About Model-Based Software Development

Software Safety

Many infusion pumps are controlled by software that governs key aspects of the user interface, controls the pumping rate, and determines the drug delivery. Unfortunately, inevitably...

The FDA has recognized that if product developers had tools that enable them to examine and evaluate software earlier in the development cycle, then there would be a greater likelihood that the resulting software would be more robust. Just as architects now have 3D modeling tools that allow them to take their clients on virtual tours of a new building before ground has been broken, the software engineering community has been developing tools for modeling software and its interactions with the system it controls. The safety properties of the model can be systematically examined, and once the model has been verified, the software derived from it can be proven to conform to the model. The result is software designs that are far more robust than those developed using traditional methods.
Research and Collaboration Agreement between MathWorks and CDRH
April, 2013

• Signed April 2013
• 5-Year validity
• Topics of focus
  • MBD
  • Formal software verification
• Principal Investigators
  • FDA: Paul Jones
  • MW: Arvind Ananthan

http://www.mathworks.com/tagteam/76380_80633v00_rcasummary.pdf
MBD Workshop for FDA @ MathWorks
September, 2014

• Experiential Learning Program (ELP)
• 2-Day Workshop conducted in Natick at MathWorks
• 10 attendees from CDRH (ODE/DAGRID)
• Collaboration with Dräger Medical, Germany
• Blog published in Medical Design Technology
PCLC Devices Workshop @ FDA’s campus
October 2015

- **Physiological Closed Loop Controller (PCLC) Devices workshop**
  - Organized by ODE members focusing on computational modeling and simulation activities
  - Focus on critical care devices (ventilators, anesthesia, etc.)

- Oct 13th and 14th @ FDA’s campus in Silver Springs

- MathWorks invited to participate and present on Model-Based Design

- Key industry thought leaders/researchers using MBD invited to present
Tool Validation Kit

• IEC Certification Kit
  • TÜV SÜD certificates and reports
  • Templates for delivering documentation to certification authorities
  • Test suites for tool validation/qualification

• FDA Tool Validation Planning Kit for MATLAB (prototype)
MATLAB & Simulink form a technical foundation for this fusion and transformation.

www.mathworks.com/medical
medical@mathworks.com