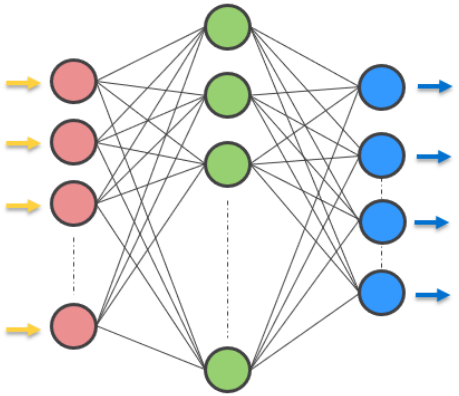
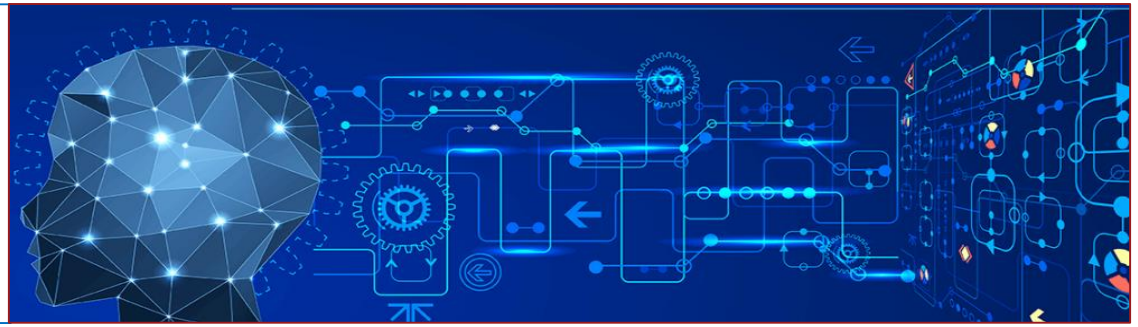
Machine Learning Classifiers to understand Emergent Behavior in complex systems & system-of-systems

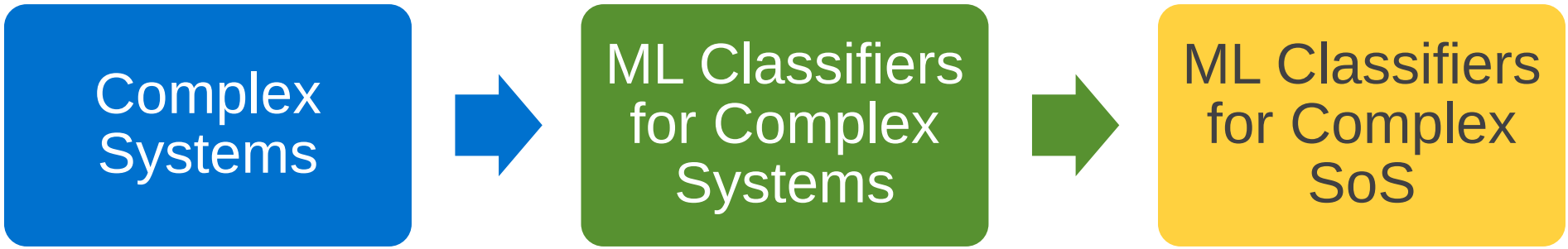


Dr. Ramakrishnan Raman, ESEP
Principal Systems Engineer, Honeywell
Asst. Director – Asia Oceania, INCOSE
ramakrishnan.raman@incose.net
https://www.linkedin.com/in/ramakrishnanraman/

INCOSE | HEARTLAND CHAPTER | MICHIGAN | THREE RIVERS | CANADA | WRIGHT BROTHERS



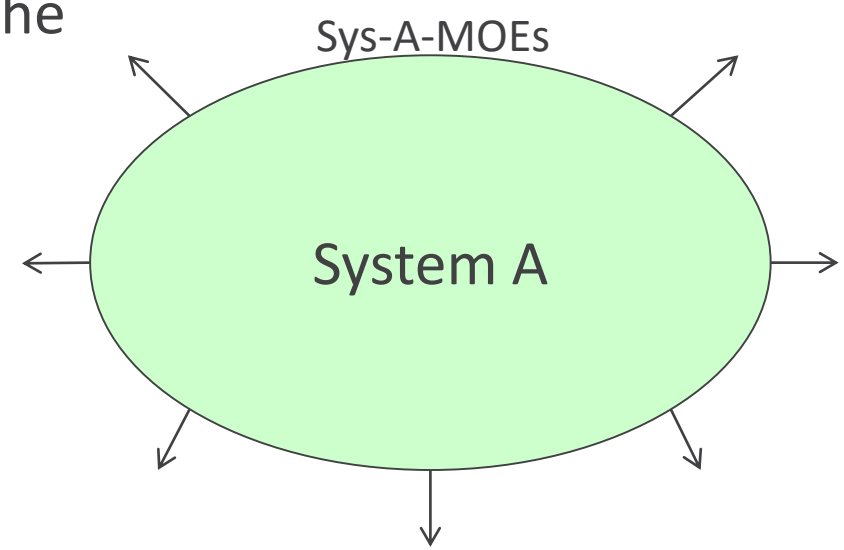
Presentation Outline



- MOEs/MOPs
- Complex Systems
- Emergent Behavior

Measures of Effectiveness - MOEs

- Operational measures of success that are closely related to the achievement of the objective of the system of interest
- Related to the achievement of the mission or operational objective being evaluated
  - In the intended operational environment
  - Under a specified set of conditions
- Manifest at the boundary of the system
- Examples
  - Response time to a user action
  - Time to Alert
  - Availability of the system

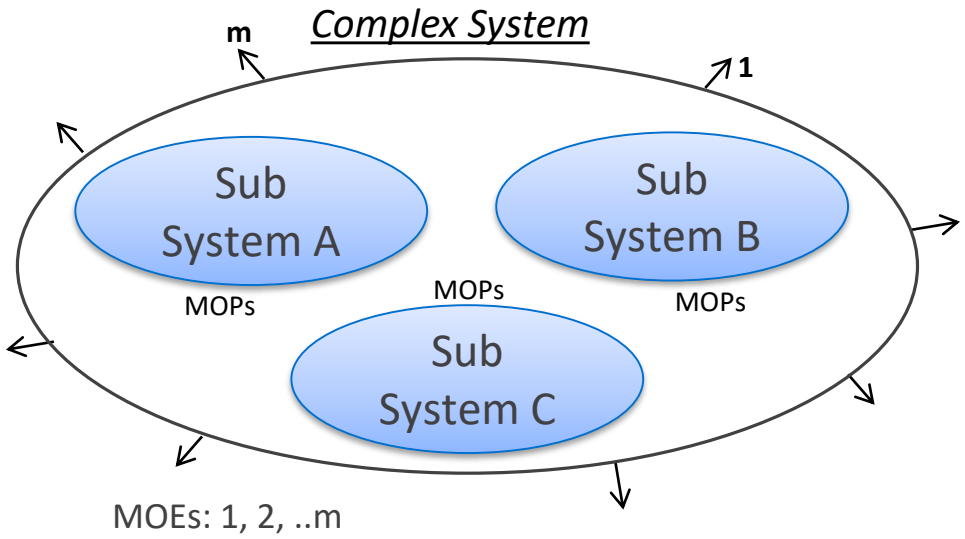


Each system is associated with a desired set of MOEs.

Measures of Performance - MOPs

- MOPs: measures that characterize physical or functional attributes relating to the system operation, measured or estimated under specified test and/or operational environmental conditions

- MOPs define the key performance characteristics the system should have when fielded and operated in its intended operating environment, to achieve the desired MOEs of the system



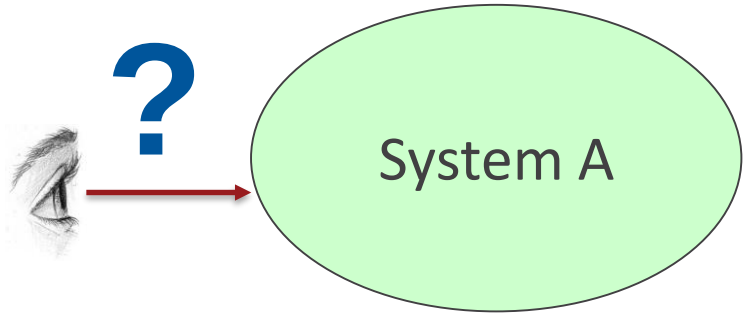
MOPs are dependent on the particular solution

Complex Systems

Complexity: Degree of difficulty in accurately predicting the future behavior

Complexity is determined by the system being observed, the capabilities of the observer, and the behavior that the observer is attempting to predict

The perspective of complexity used in this paper is with respect to the degree of difficulty in accurately predicting the future behavior



- Multiplex of relationships/ forces/ interactions between subsystems & constituent systems
- Difficulties in establishing cause-and-effect chain
- Characteristics: Emergence, hierarchical organization, numerosity….

Emergent Behavior

Emergence refers to the ability of a system to produce a highly-structured collective behavior over time, from the interaction of individual subsystems

Examples: flock of birds flying in a V-formation, and ants forming societies of different classes of individual ants

(howitworksdaily.com)

Complex Systems – Emergent Behavior

For a system, emergent behavior refers to all that arises from the set of interactions among its subsystems and components

For system-of-system, emergent behavior refers to all that arises from the set of interactions among its constituent systems

(earth.com)

Complex systems are expressed by the emergence of global properties. It is difficult, if not impossible, to anticipate emergence just from a complete knowledge of component or subsystem behaviors

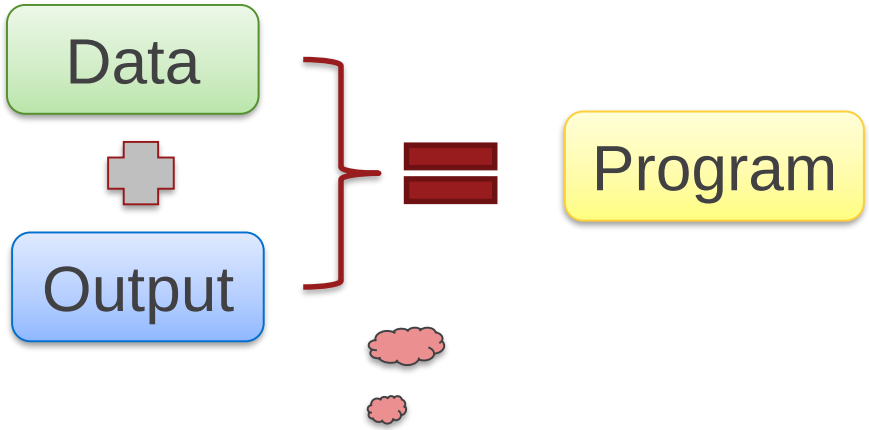
- Machine Learning
- Classifier Approach
- Aircraft Pitch Control Example
- ML Classifier Training
- Behavior Predictions

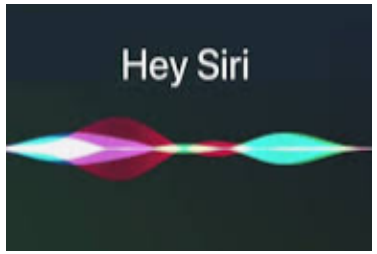
Machine Learning

Mohri M, Rostamizadeh A, Talwalkar A. Foundations of Machine Learning Cambridge, MA: MIT Press; 2012.

"Machine learning can be broadly defined as computational methods using experience to improve performance or to make accurate predictions"

"Machine Learning represents the field of study that allows computer programs to learn without being explicitly programmed"





Neural Networks

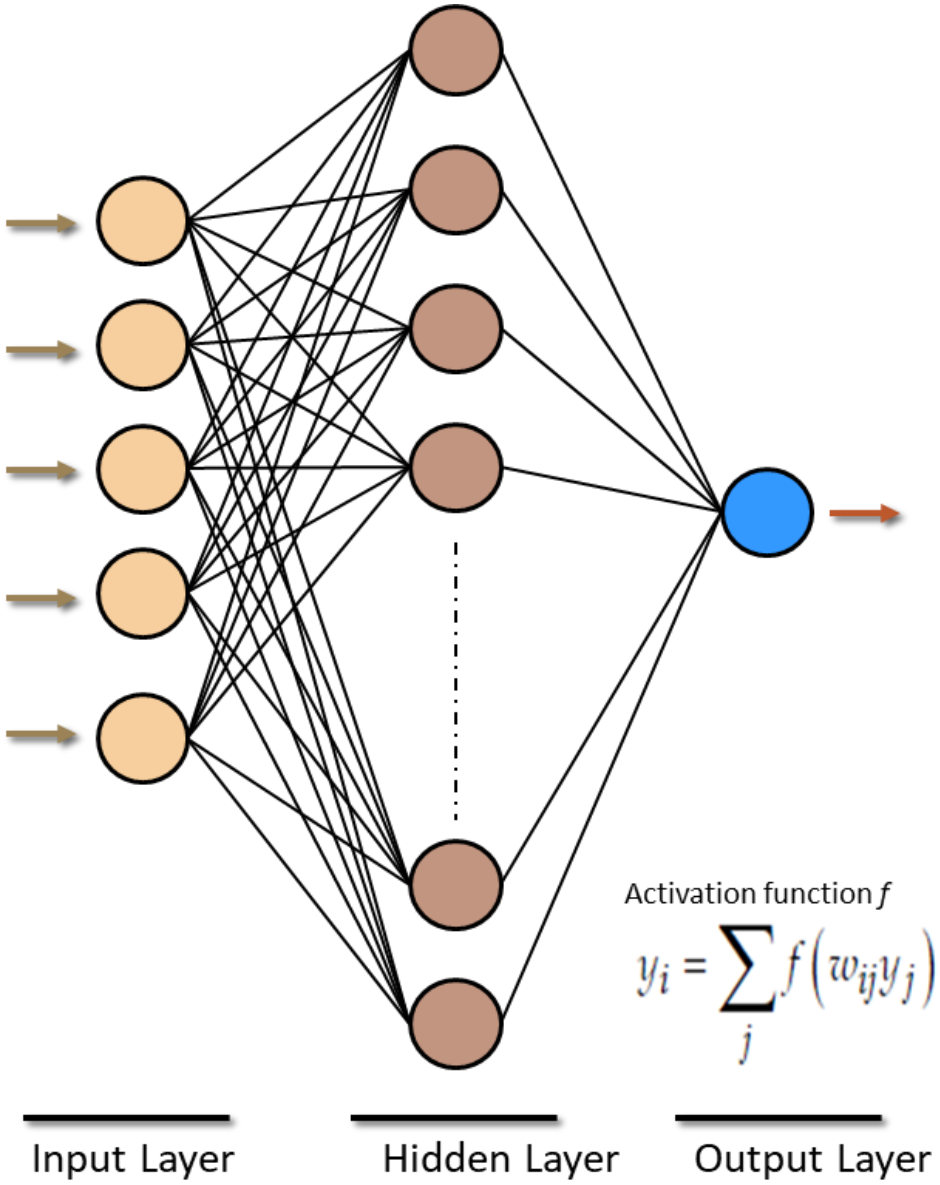
Artificial Neural Networks (NN):

- *collection (organized in layers) of interconnected units (nodes)*
- *each node having the capability to receive a signal, process the signal, and transmit the processed signal to other units linked to it.*

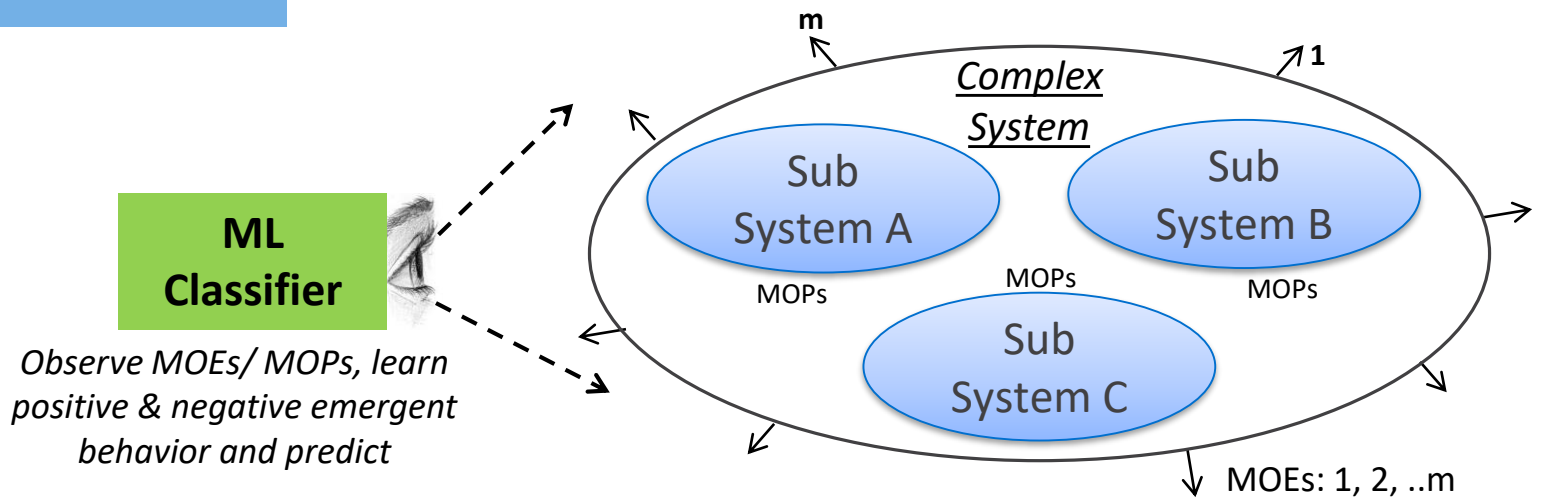
The work presented involves

Dimensionality Reduction

Supervised Learning



ML Classifier

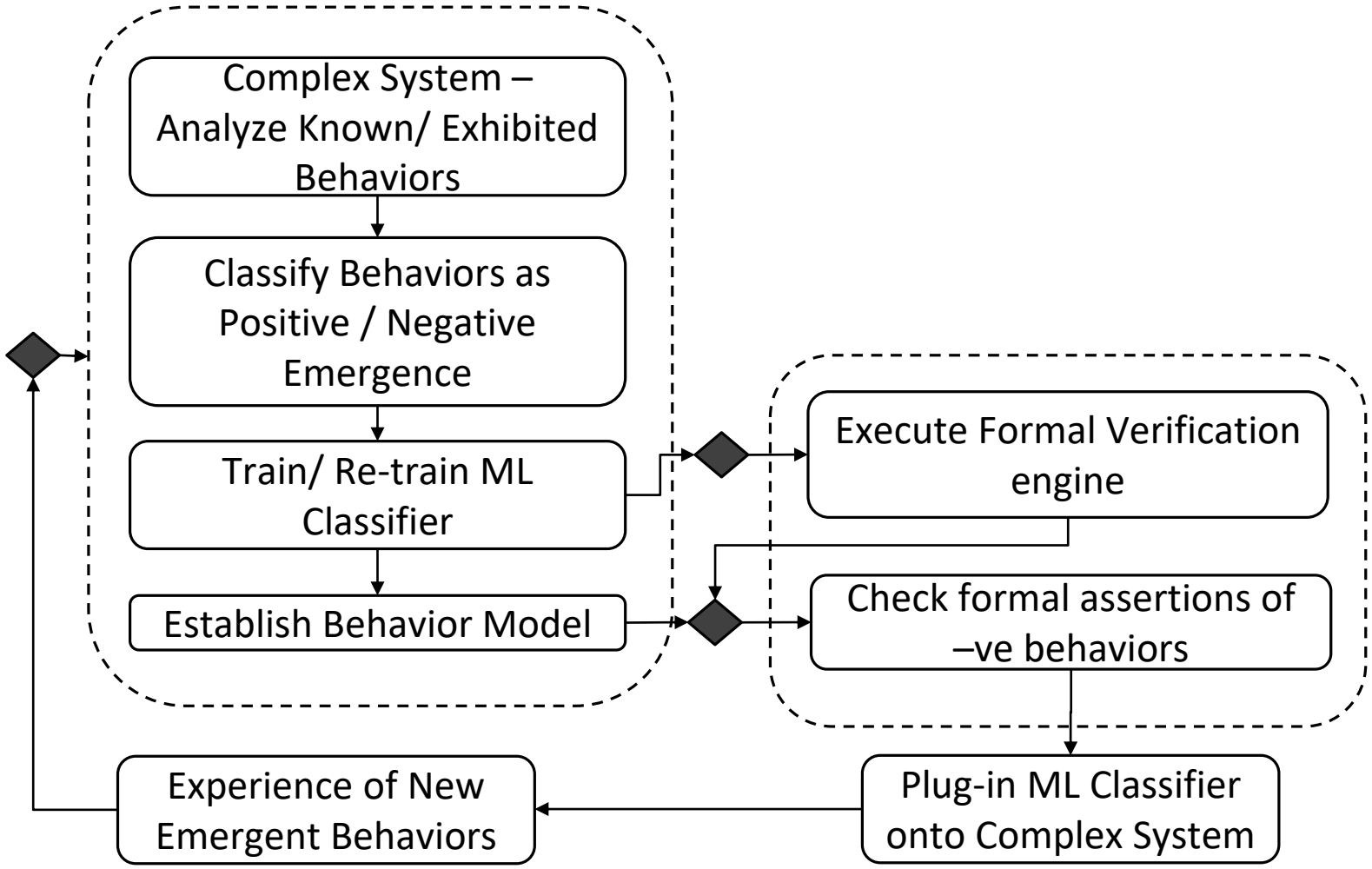


The system has its MOEs, and comprises one or more subsystems, with corresponding performance characteristics manifesting as MOPs

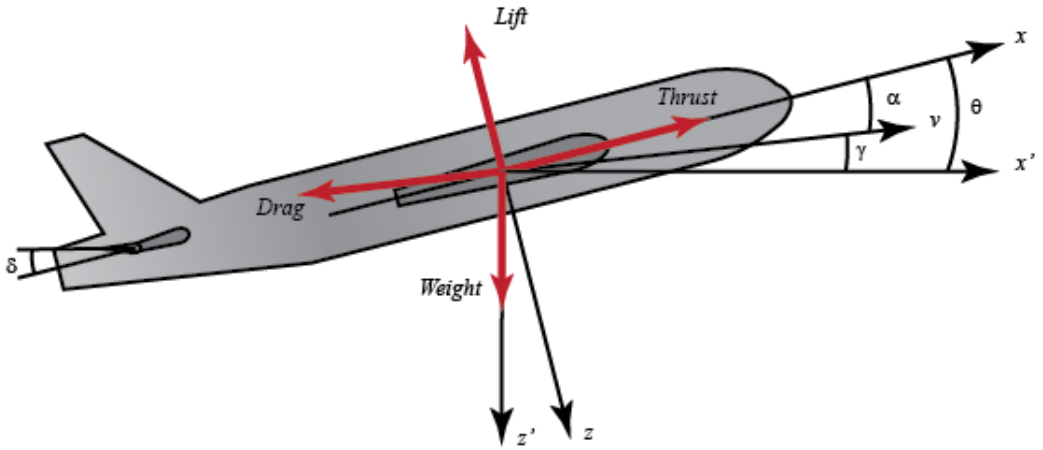
The proposed approach involves building a Machine Learning (ML) Classifier that observes the various MOPs and MOEs, and learns the emergent behavior.

The classifier can then be used by a Formal Verification Engine to assert the occurrence of negative emergent behavior

ML Classifier – in tandem with verification engine



Aircraft Pitch Control System

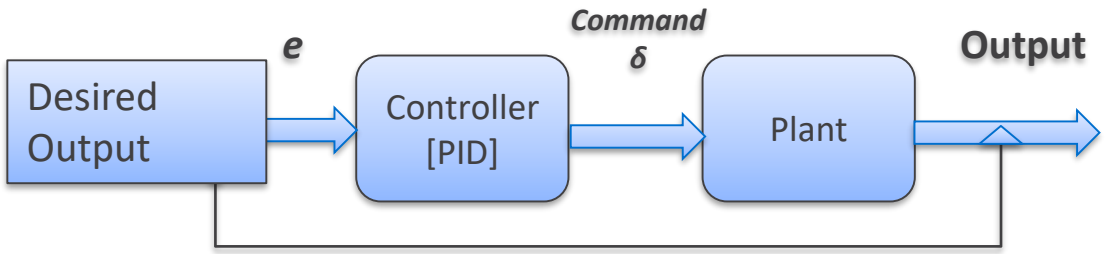


$$\dot{\alpha} = \mu\Omega\sigma\left[-(C_L + C_D)\alpha + \frac{1}{(\mu - C_L)}q - (C_W \sin\gamma)\theta + C_L\right]$$

$$\dot{q} = \frac{\mu\Omega}{2i_{yy}}\left[\left[C_M - \eta(C_L + C_D)\right]\alpha + \left[C_M + \sigma C_M(1 - \mu C_L)\right]q + (\eta C_W \sin\gamma)\delta\right]$$

$$\dot{\theta} = \Omega q$$

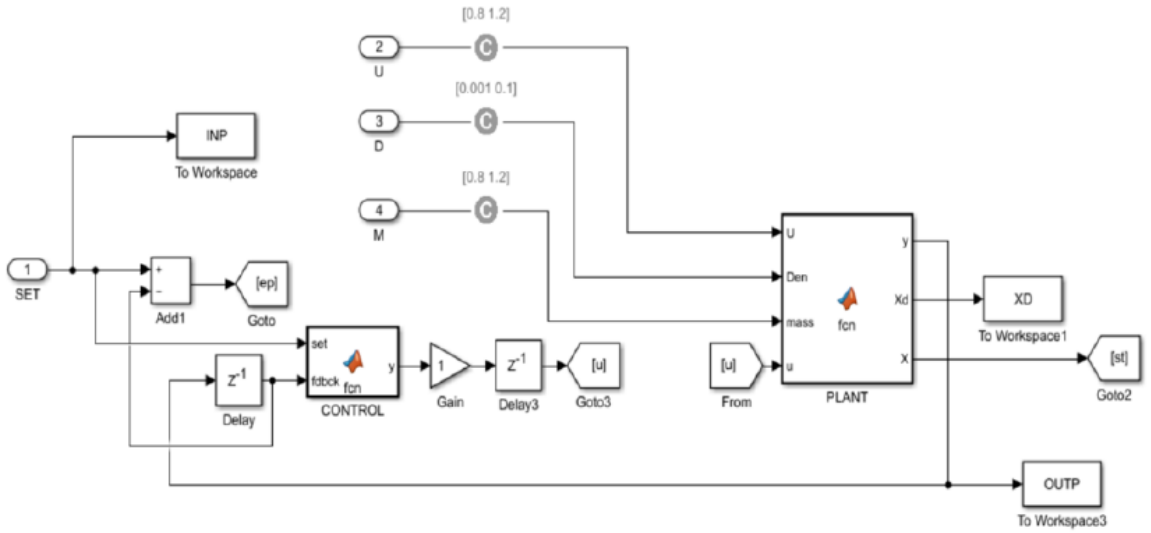
- MOEs for this system pertain to the comfort level of the flight
  - Comfortable pitch
  - Marginal Discomfort: marginal issues, the flight will be felt like a roller coaster with lower amplitude, finally settling down to a stable flight path
  - Significant Discomfort: will feel like a roller coaster ride, not divergent but an oscillatory unsettled behavior

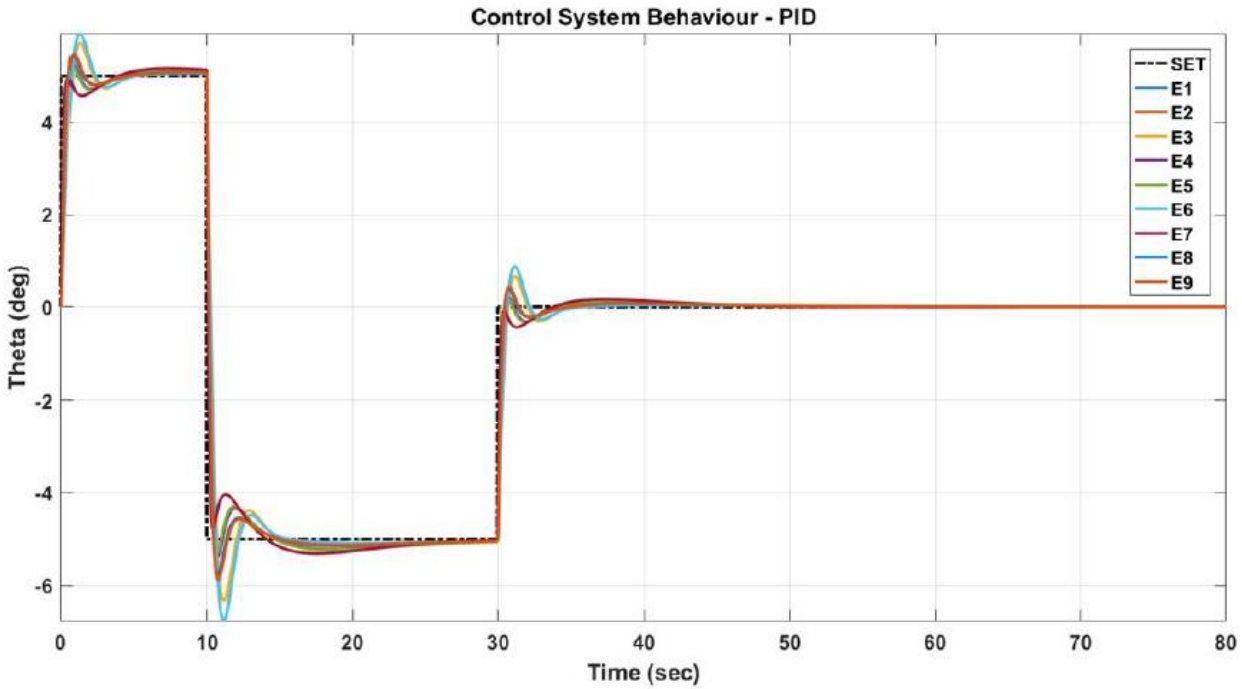


The pitch control system is adapted from publicly available control tutorials website - Control Tutorials for Aircraft Pitch: PID Controller & System Modeling – available University of Michgan website

Behavior Analysis Experiments

Model of the Control System

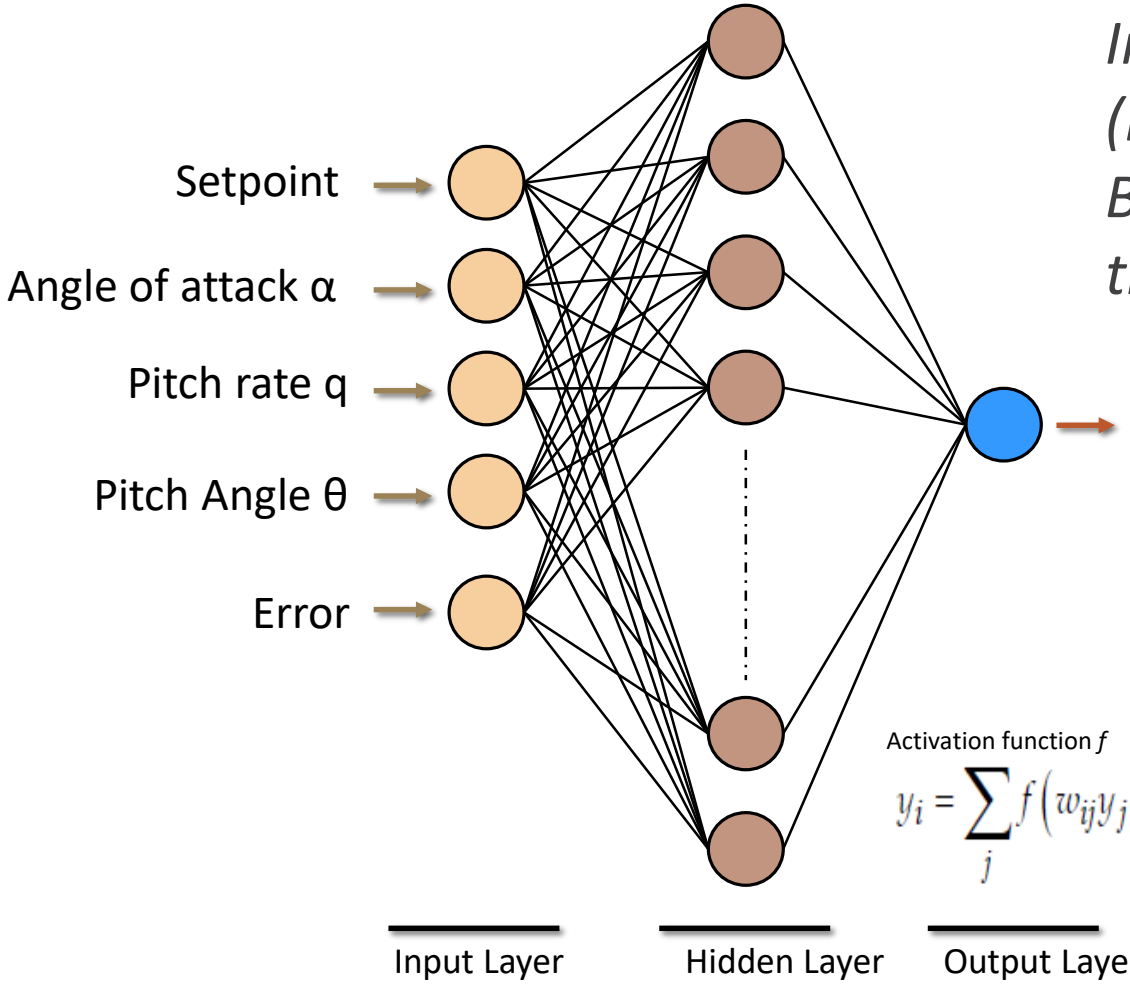




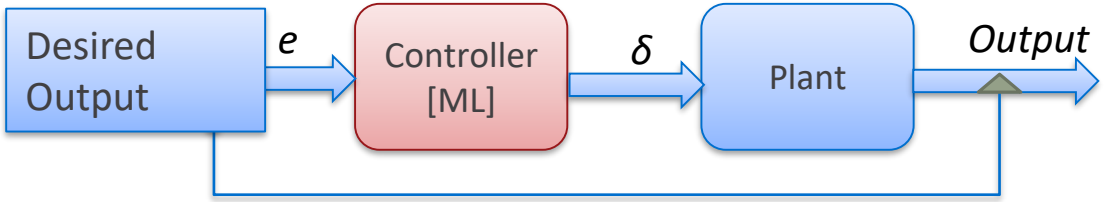
- Figure depicts MATLAB model of pitch control system
- An orthogonal array of experiments was devised to analyze the behavior of the control system for different values of mass, speed and density

- Figure depicts stable behavior for variation in the response to a set point of 5 degrees (E1,E2,… are different combinations of mass, speed, density)
- This stable behavior is classified as a positive emergent behavior

ML Controller

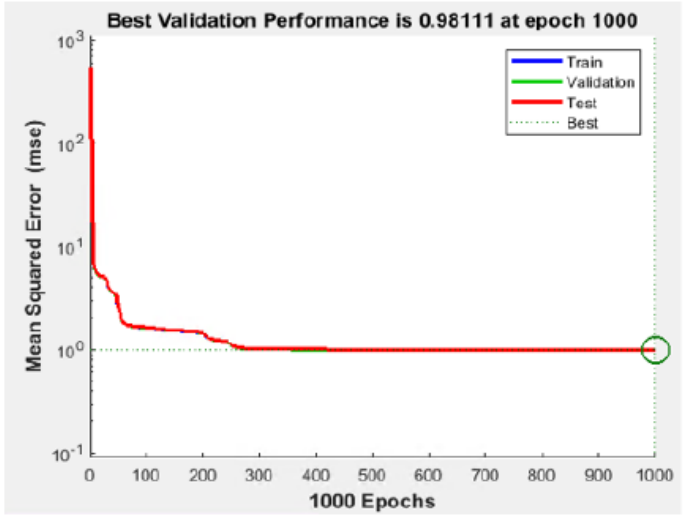


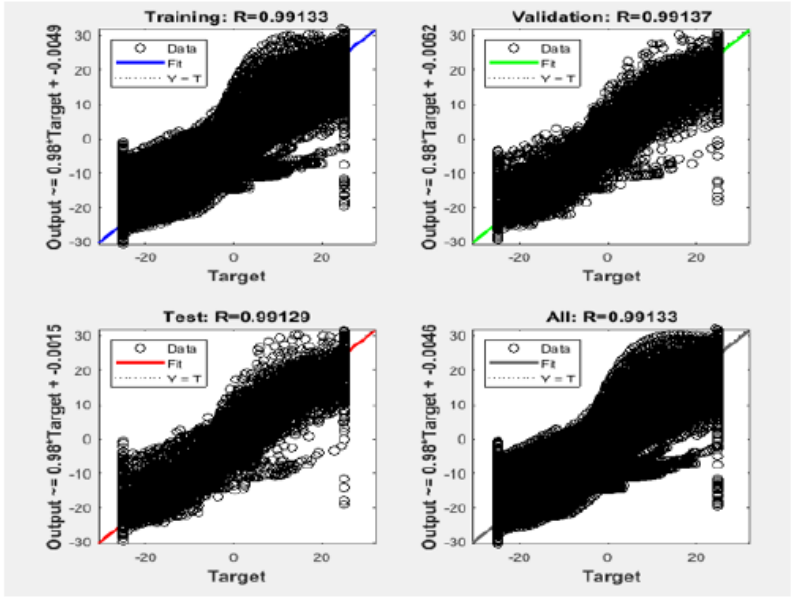
Instead of the Proportional–Integral–Derivative (PID) controller, the machine learning model (ML Based Controller) commands the plant to achieve the desired output.

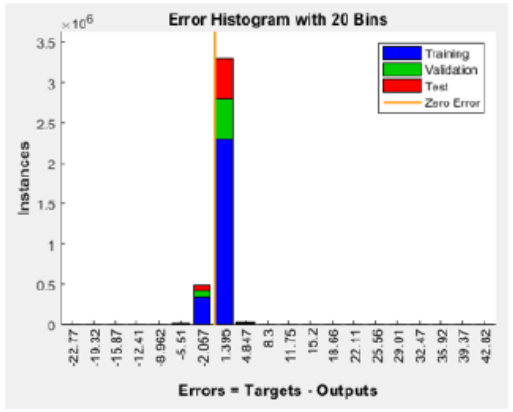


20% variation in the plant model for speed, density and mass was used for getting the data set (orthogonal array)

Training the ML Controller







Results

	Samples	MSE	R
Training:	2695269	9.81089e-1	9.91325e-1
Validation:	577558	9.81105e-1	9.91372e-1
Testing:	577558	9.87390e-1	9.91285e-1

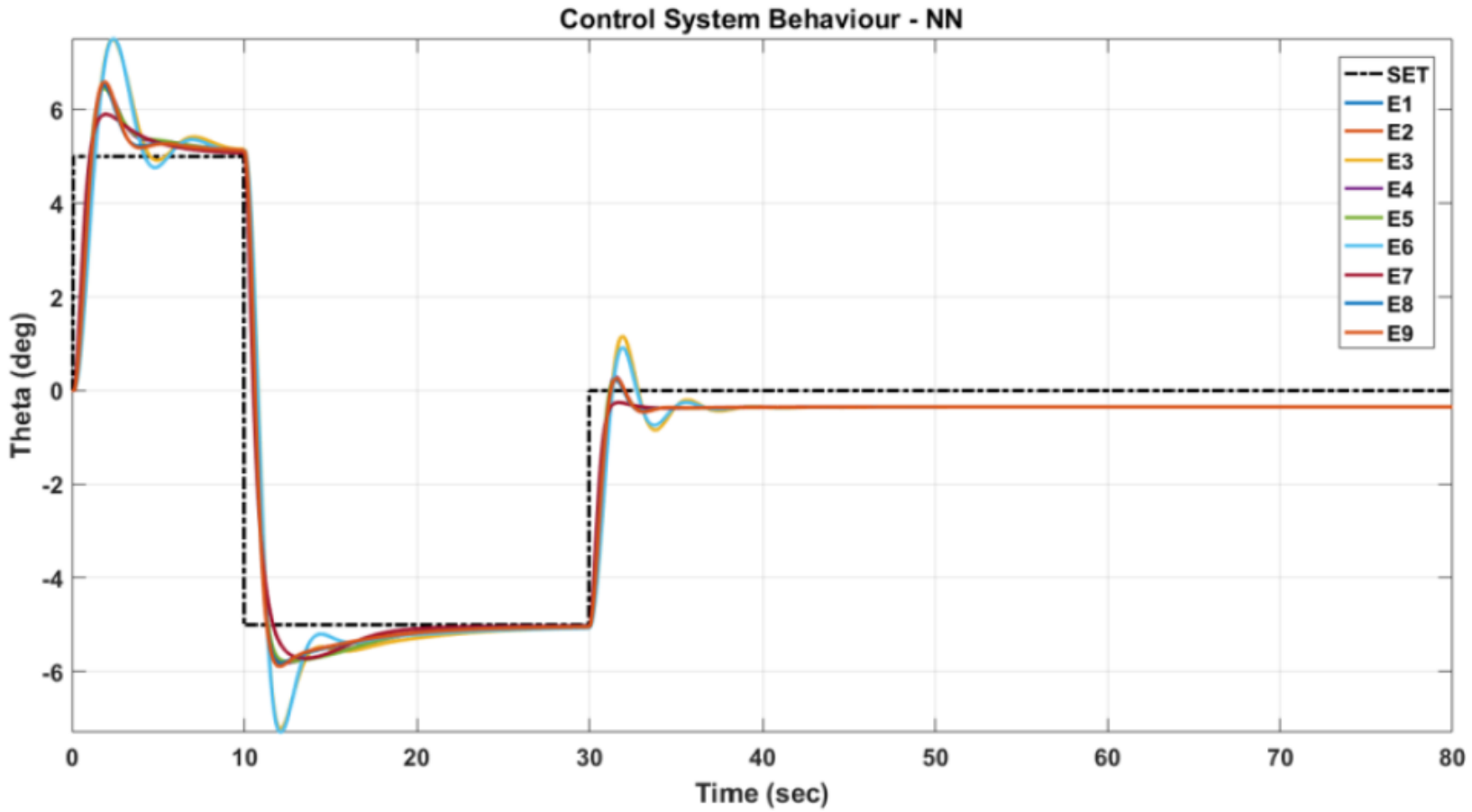
[screenshot from MATLAB - Machine Learning toolbox was used]

Mean Square Error (MSE): average squared difference between the outputs and targets (i.e. lower values of MSE are better)

R values: measures the correlation between outputs and targets (i.e. R value closer to 1 is better)

Learning is stopped at epoch 1000, to serve the purpose of dealing with a system that can exhibit negative emergent behavior at times.

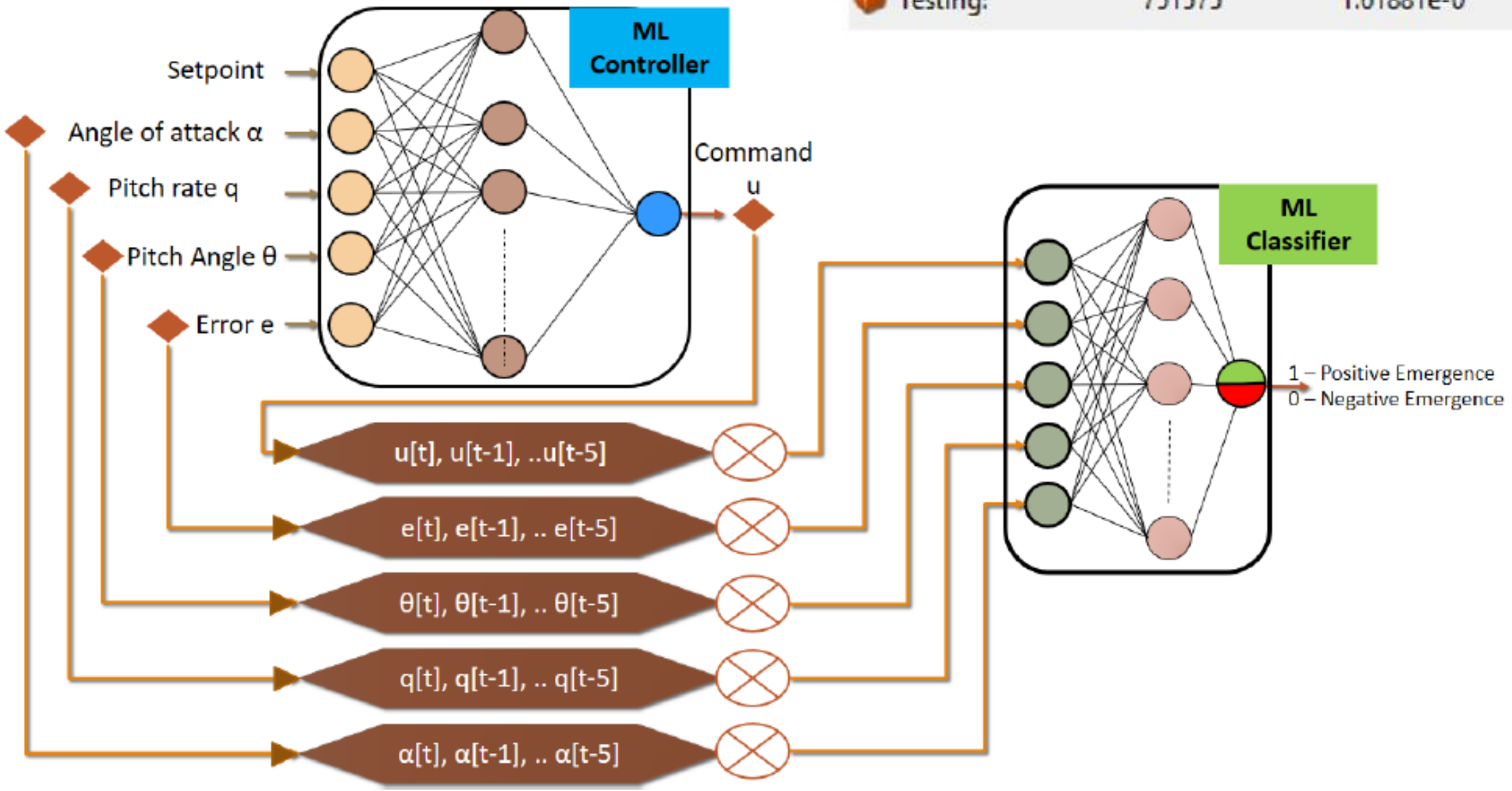
System Behavior – ML Controller



Behavior is not as stable as conventional PID controller [intentional].

ML Classifier

	Samples	CE	%E
Training:	3507351	6.00732e-1	1.21208e-0
Validation:	751575	1.61671e-0	1.24245e-0
Testing:	751575	1.61881e-0	1.21571e-0



The ML Classifier is built by tapping in the same inputs used for the ML Controller, and feeding a set of six values, corresponding to current time t, and previous five instance (data set ~5 million records)

ML Classifier Performance

[screenshot from MATLAB - Machine Learning toolbox was used]

The column on the far right of the plot shows the percentages of all the examples predicted to belong to each class that are correctly and incorrectly classified.

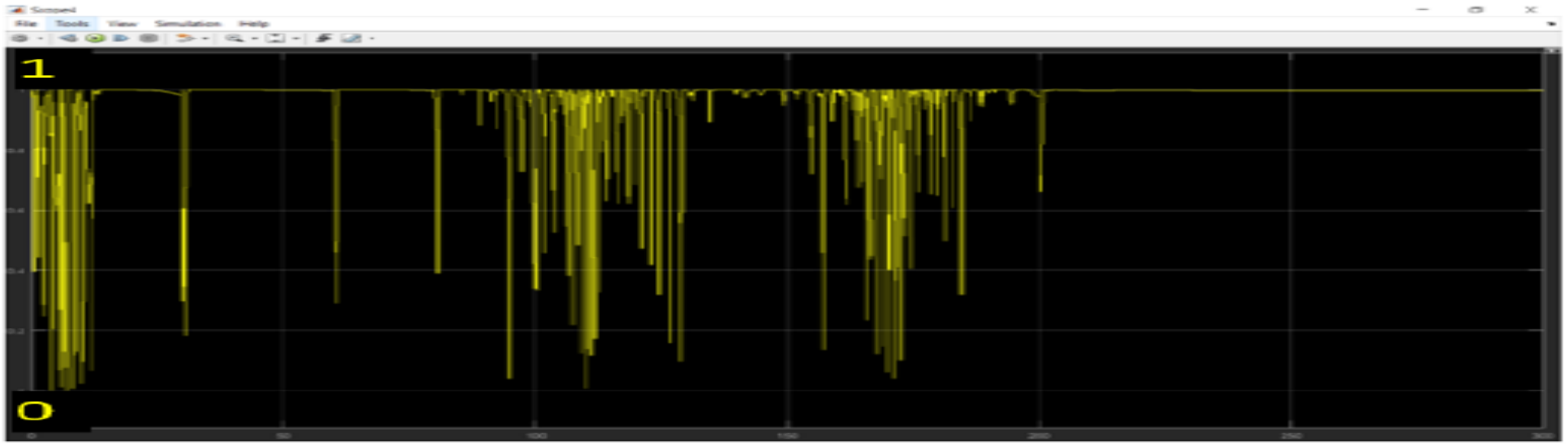
- *These metrics are the precision (or positive predictive value) and false discovery rate, respectively.*

The row at the bottom of the plot shows the percentages of all the examples belonging to each class that are correctly and incorrectly classified.

- *These metrics are often called the recall (or true positive rate) and false negative rate, respectively.*

The cell in the bottom right of the plot shows the overall accuracy

ML Classifier Predictions



0 – Negative Emergence

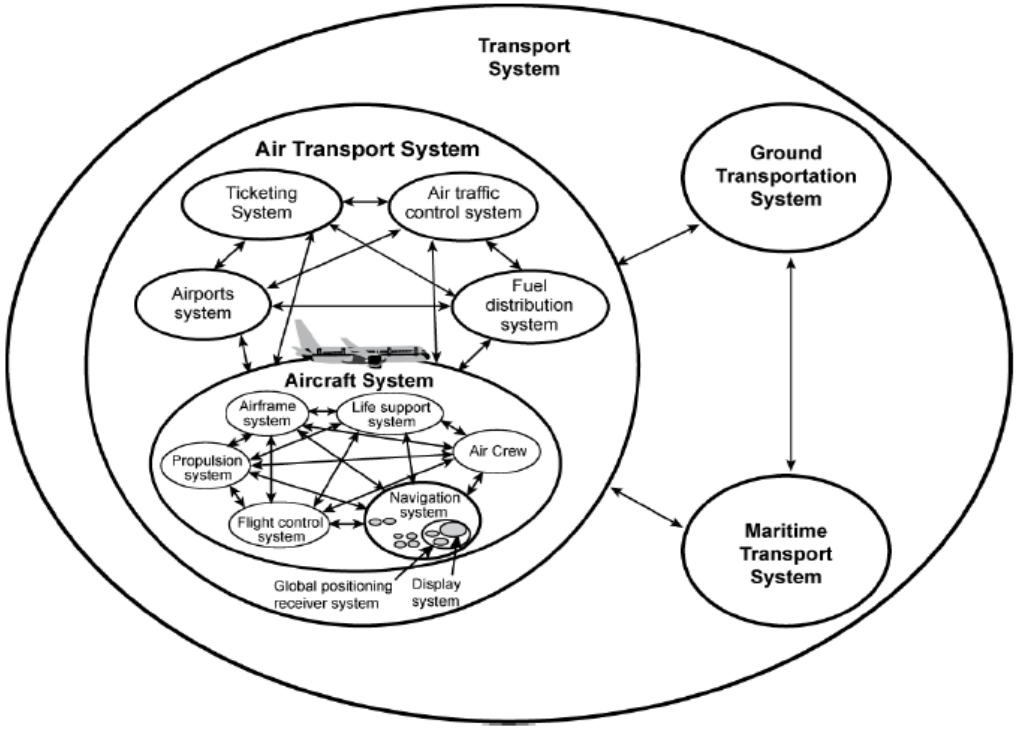
- The ML Classifier is plugged on to the control model
- Monitoring is through scope monitor
- Values closer to 0 indicates negative emergent behavior

- SoS
- MOE Relationships
- Swarm UAV Example
- Principal Comp Analysis
- ML Classifier Training
- Behavior Predictions

System-of-System (SoS)

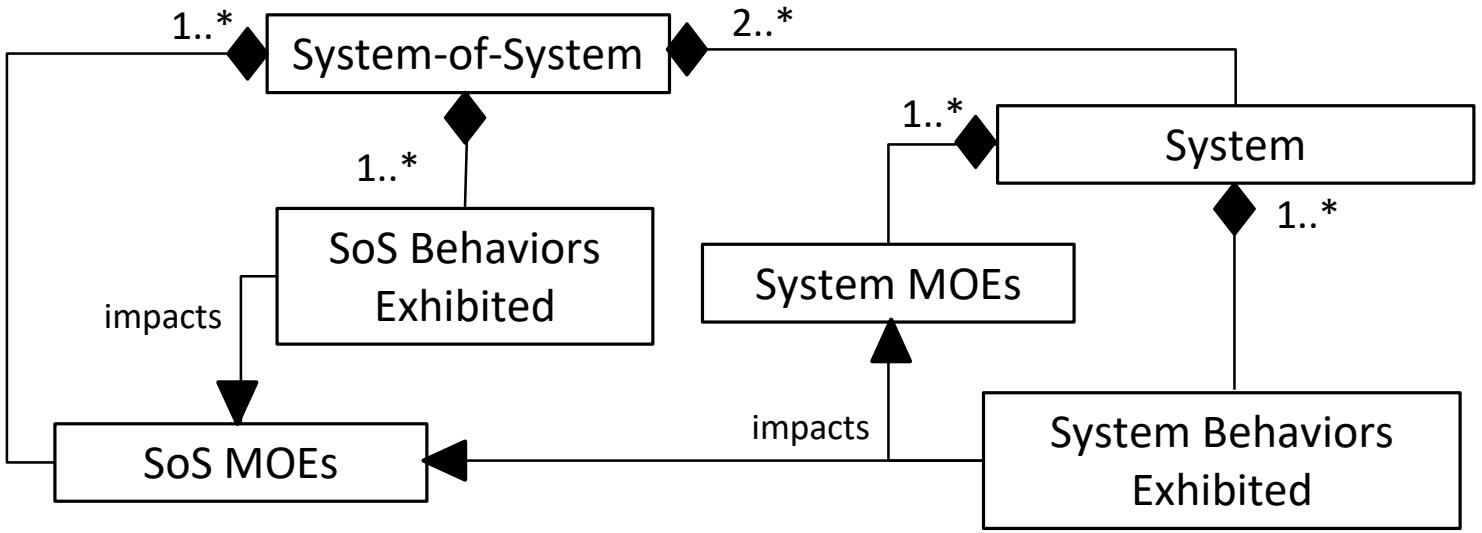
System-of-Systems are systems-of-interest whose system elements are themselves systems - they typically entail large-scale inter-disciplinary problems involving multiple, heterogeneous and distributed systems

Each system has an independent purpose and viability, in addition to the SoS by itself having an independent purpose and viability



Source: INCOSE SE Handbook

MOE Relationships

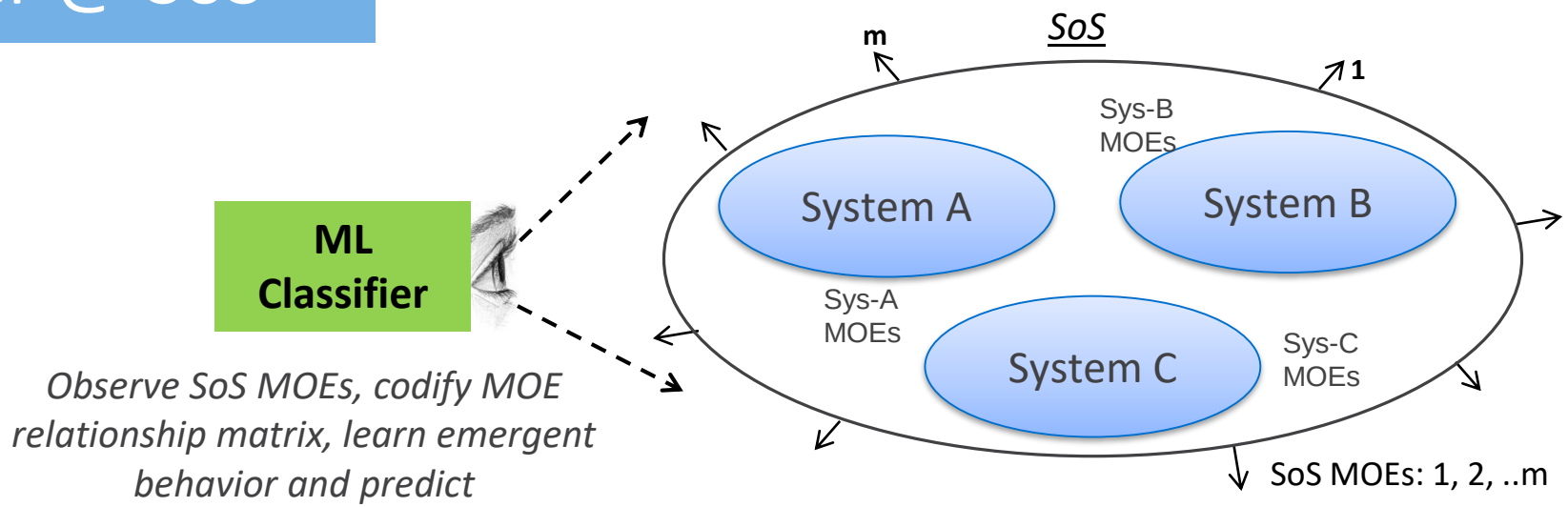


The MOEs of the system are impacted by the behaviors exhibited by the system.

Similarly, the MOEs of the SoS are impacted by the behaviors exhibited at SoS level.

Further, the behaviors exhibited at constituent system level also impacts the SoS MOEs.

ML Classifier @ SoS

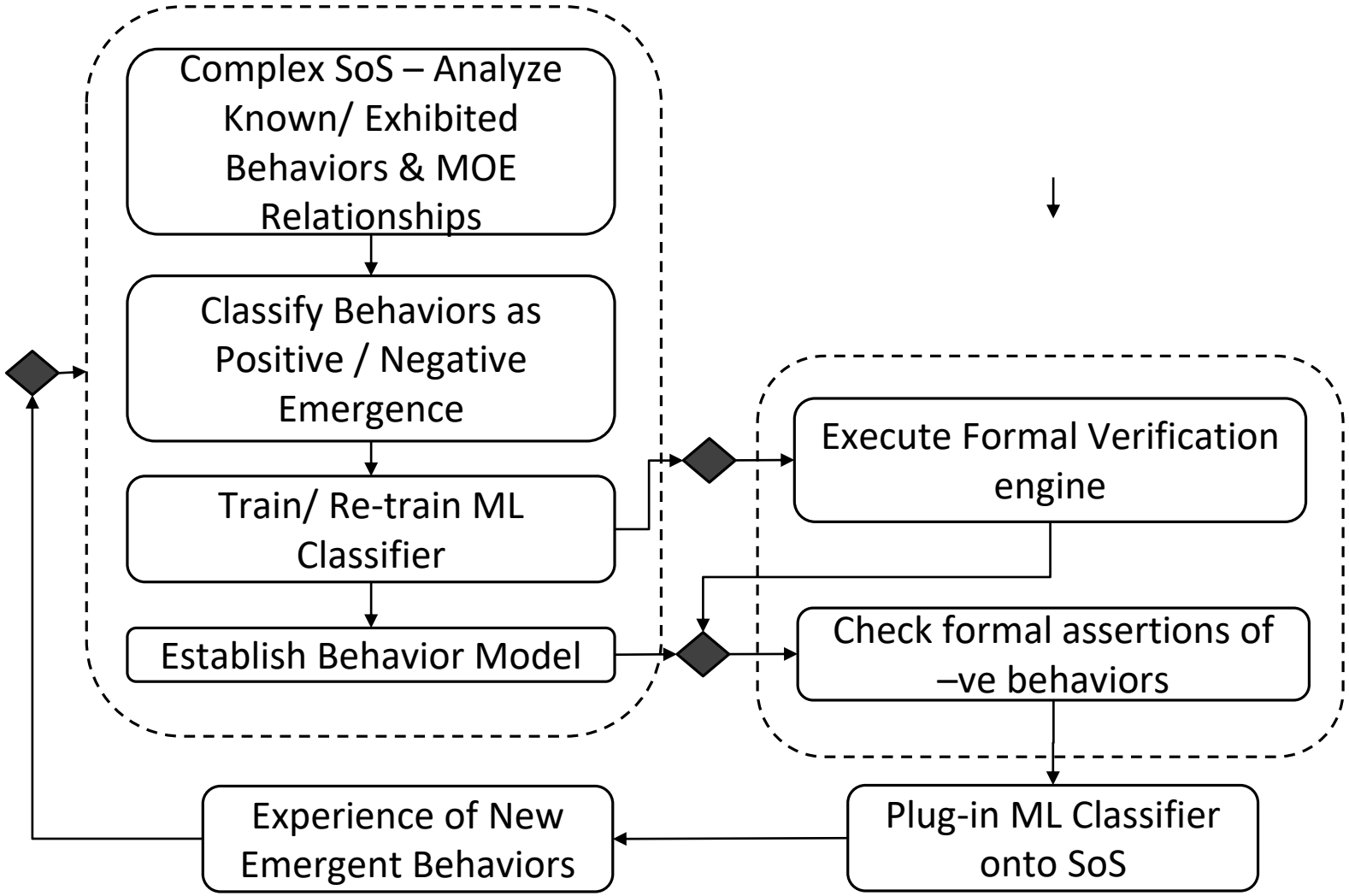


The SoS has its MOEs, and comprises one or more systems, each with its own MOEs

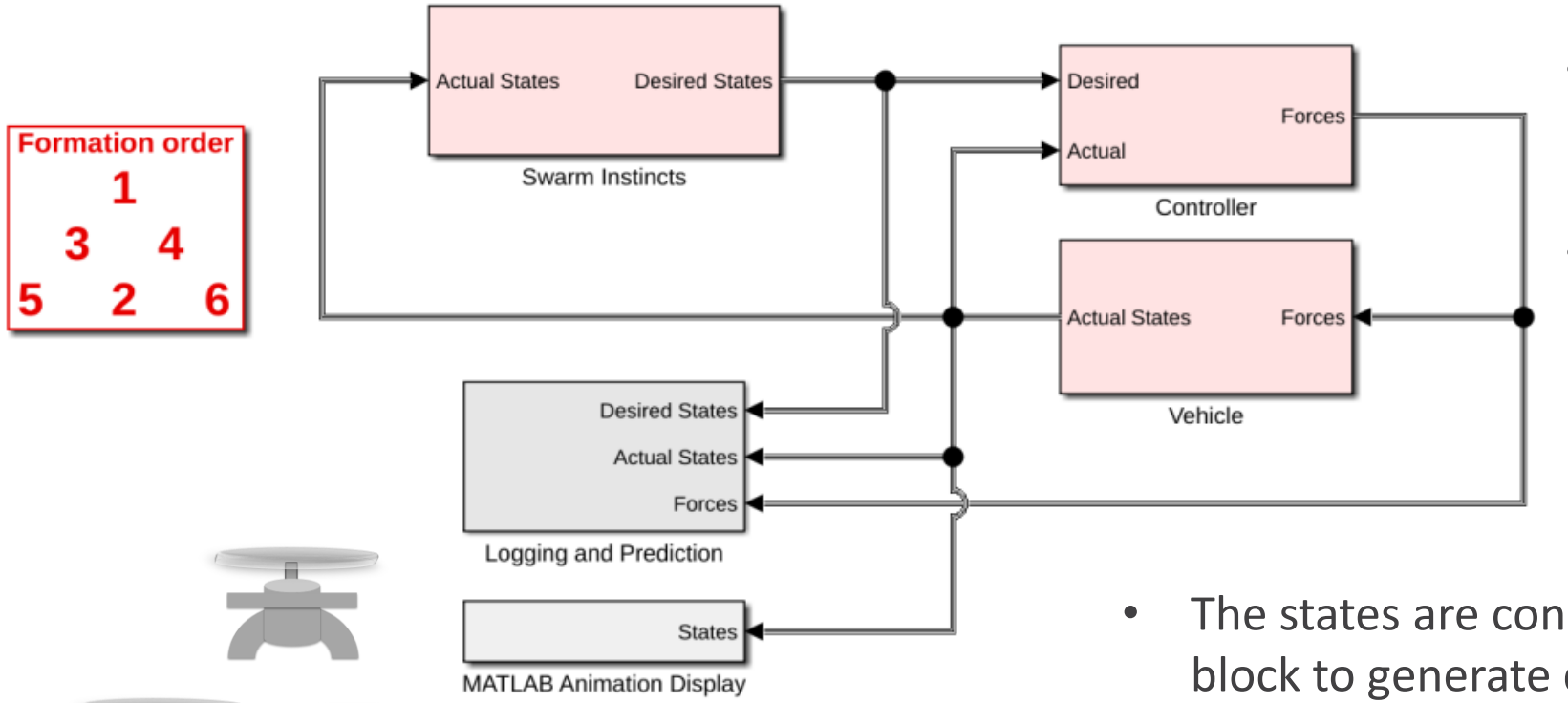
The proposed approach involves building a Machine Learning (ML) Classifier that observes the various MOPs and MOEs, and learns the emergent behavior.

The classifier can then be used by a Formal Verification Engine to assert the occurrence of negative emergent behavior

ML Classifier – in tandem with verification engine



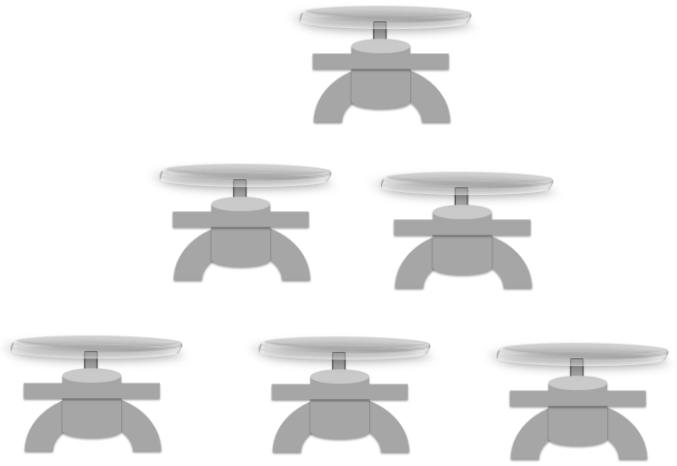
Swarm of UAVs



- The states of each individual UAV include the inertial position and Velocities
- The states are initialized in the Vehicle block with random initial positions with respect to the leader of the formation

- The states are consumed by the Swarm Instincts block to generate desired states to maintain or change the formation.
- The desired states are passed onto the Controller which generates required forces to be applied on the vehicle.
- The required forces are then sent to the vehicle, which integrates them to get the current states.

SoS State Data



SoS level:

UAVs pair-wise distances: 15 pairs between the five UAVs

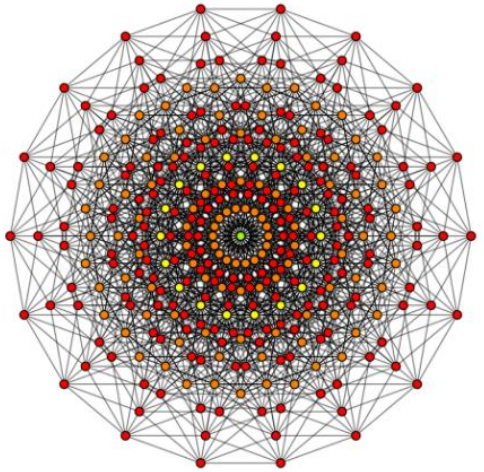
UAVs pair-wise slopes: 15 pairs between the five UAVs, with two slopes along XZ and YZ

Principal Component Analysis

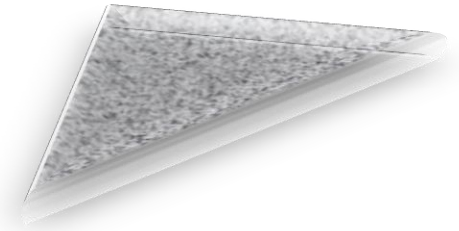
There are many factors that impact the emergent behavior of SoS, and visualization of the SoS state parameters would provide deeper insights

However, to understand the interplay of the various factors on SoS emergent behavior, a higher dimensional visualization is required which would be difficult to represent.

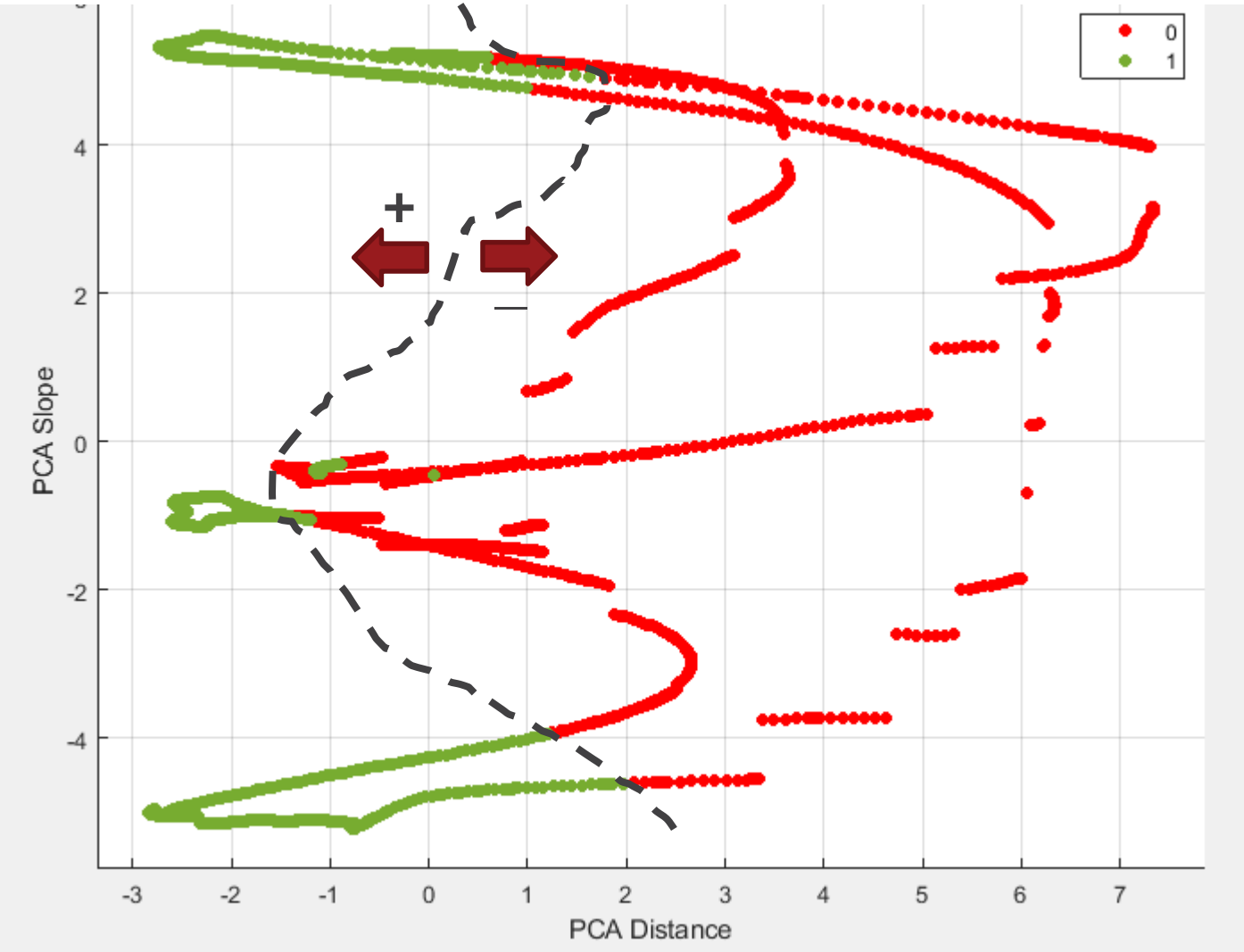
PCA (Principal Component Analysis) is used towards getting lower dimensional views



projection of an n-dimensional input data onto a reduced k-dimensional linear subspace,



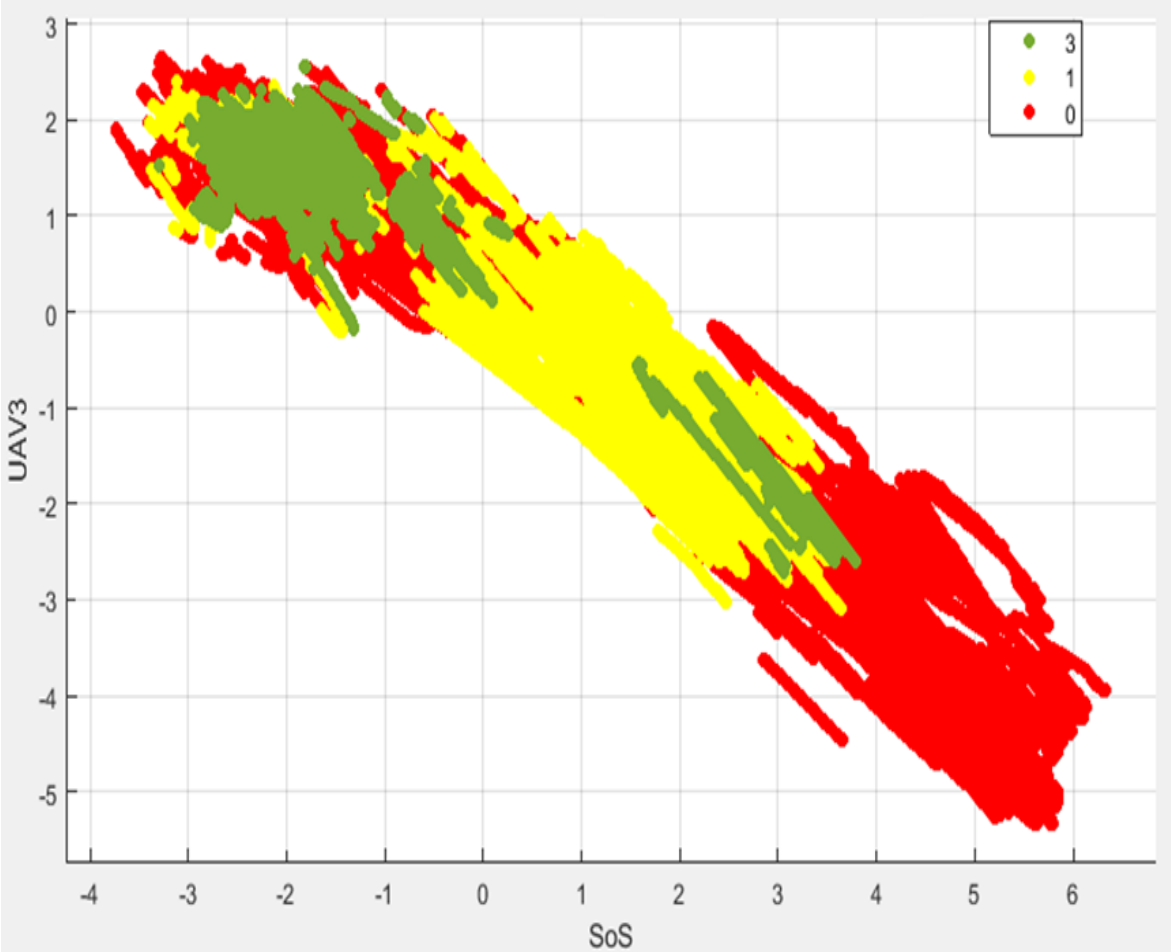
Behavior Analysis through PCA



The figure represents the state of the SoS, reducing the multi-dimensional state parameters to lower dimensions – one for pairwise distances, and one for pairwise slopes

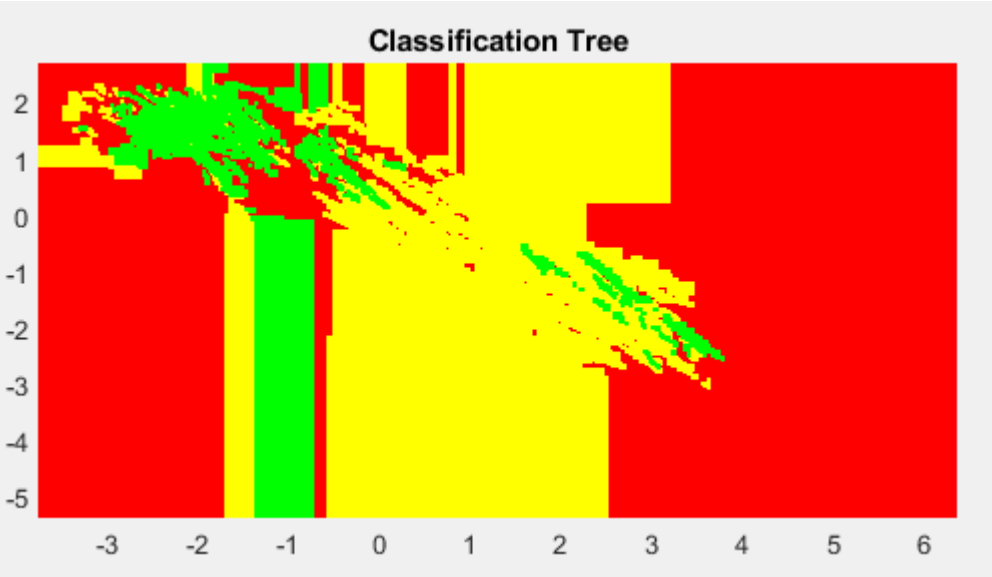
This enables identification of specific regions/zones of positive (1) and negative (0) emergence

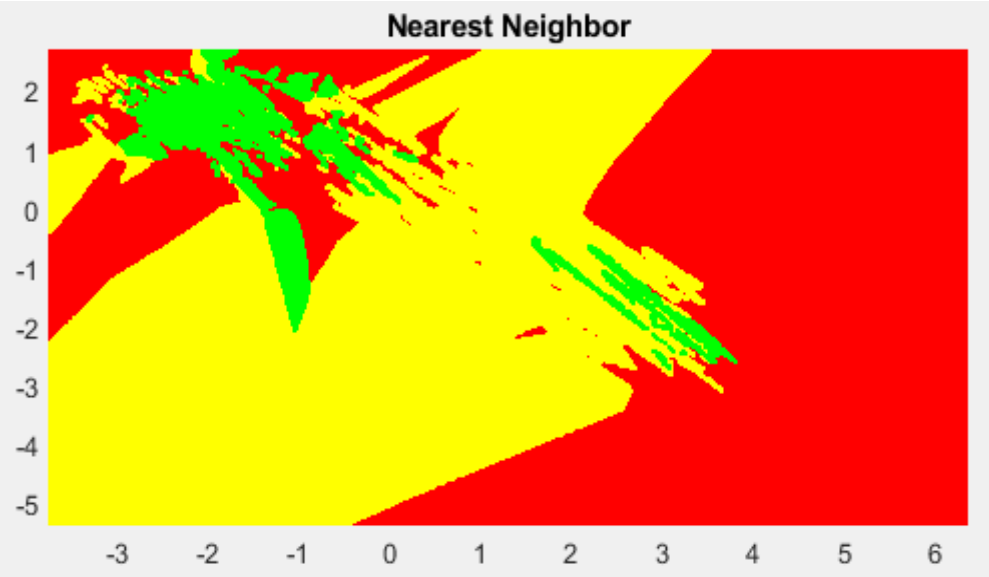
SoS vs Constituent System Behavior Analysis



- *Scenario of both SoS and UAV#3 exhibiting bad behavior (legend 0 in the plot, red)*
- *Scenario of UAV#3 meeting its own MOEs, but SoS is exhibiting negative behavior (legend 1 in the plot, yellow);*
- *Scenario of both UAV#3 and the SoS exhibiting good behavior (legend 3 in the plot, green).*

Supervised Learning Classification





The figure illustrates the decision surface of the following classification algorithms: (a) fitted binary Classification decision tree (b) KNN-Nearest Neighbor

Conclusions

Conclusions

- *Presented a novel approach for understanding and analyzing the emergent behavior for complex systems & SoS*

- *The ML Classifier can learn by observing the various MOEs/MOPs and predict negative emergent behavior of the system/ SoS*

AUGMENT THE INTELLIGENCE OF THE SYSTEM / SOS

For more details…

Wiley Online Library

INCOSE International Symposium

Volume 29, Issue 1
July 2019
Pages 544-559

Session 7 Track 1: Machine Learning/Artifical Intelligence 1

An Approach for Formal Verification of Machine Learning based Complex Systems

Ramakrishnan Raman, Yogananda Jeppu

First published: 27 September 2019 | https://doi.org/10.1002/j.2334-5837.2019.00620.x

Framework for Formal Verification of Machine Learning Based Complex System-of-System [under review]

Ramakrishnan Raman, Nikhil Gupta, Yogananda Jeppu

THANK YOU