**Abstract**

The general goal of this research effort was to design and develop a novel biologically inspired framework for flight envelope estimation and protection, which is conceptually integrated with the abnormal detection and accommodation processes and is general and comprehensive such that it can operate at all conditions, normal and abnormal, a priori known or unknown. Two major methodologies have been proposed and investigated: one based on a dendritic cell mechanism and one on structuring of the self/non-self. Data for development and demonstration has been obtained from the WVU motion-based flight simulator for a supersonic fighter aircraft. Results show promising capabilities of the artificial immune system paradigm for providing an integrated and comprehensive solution to the problem of aircraft abnormal condition detection, identification, evaluation, and accommodation.

**Subject Terms**

- Artificial Immune System
- Flight Envelope
- Estimation and Protection

**Distribution/Availability Statement**

Distribution A

Peter Erbland, ADPM. DARPA Research Services, Defense Technical Information Center, Christopher L. Glista, Frank Friedl, Michael Oppenheimer, Chris Hall
Sponsored by: Defense Advanced Research Projects Agency, Tactical Technology Officer (TTO)
Program: Artificial Immune Systems for Flight Envelope – West Virginia University
Issued by DARPA/CMO under Contract No. HR0011-13-C-0024

“The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.”

**Artificial Immune System for Flight Envelope Estimation and Protection**

Technical Point of Contact:

**Dr. Mario Perhinschi**  
Department of Mechanical and Aerospace Engineering,  
West Virginia University  
PO Box 6106, ESB 521  
Morgantown, WV26505-6106  
Phone Number: (304) 293-3301  
Fax Number: (304) 293-6689  
E-Mail: Mario.Perhinschi@mail.wvu.edu

ERAU Technical Point of Contact:

**Dr. Hever Moncayo**  
Department of Aerospace Engineering,  
Embry-Riddle Aeronautical University  
600 South Clyde Morris Blvd  
Daytona Beach, FL 32114  
Phone Number: (386) 323-8092  
E-Mail: moncayoh@erau.edu

Administrative Point of Contact:

**Mr. Alan Martin**  
West Virginia University Research Corporation  
886 Chestnut Ridge Road  
P.O. Box 6845  
Morgantown, WV26506  
Phone Number: (304) 293-3998  
Fax Number: (304) 293-7435  
E-Mail: Alan.Martin@mail.wvu.edu

12/31/2014
# Table of Contents

Standard Form 298                    1  
Title Page                     2  
Table of Contents                     3  
List of Figures                     6  
List of Tables                   10  
List of Acronyms                   12  

Project Summary                      13  
1. Task Objectives                   13  
2. Technical Problems                 14  
3. General Methodology               15  
4. Technical Results                  16  
5. Important Findings and Conclusions 16  
6. Significant Hardware Development   17  
7. Special Comments                  17  
8. Implications for Future Research  17  

Chapter 1  General Framework for Immunity-based Abnormal Condition Detection,  Identification, Evaluation, and Accommodation  18 
1.1. Biological Immune System          18 
1.2. Artificial Immune System Paradigm 23 
1.3. Definitions, Nomenclature, and Notation 26 
1.4. Aircraft Abnormal Condition Management Process 31 
1.5. Definition of the Self and Non-Self 34  
  1.5.1. General Algorithm               34 
  1.5.2. Raw Data Set Union Method       38 
  1.5.3. Cluster Set Union Method         40 
  1.5.4. Comparison of the Proposed Methods for Antibody Generation 42 
1.6. Hierarchical Multi-Self Strategy  43 
1.7. Selection of Development and Demonstration Test Cases 45 
  1.7.1. Aircraft Subsystems             45 
  1.7.2. Types of Subsystem Abnormal Conditions 45 
  1.7.3. Flight Envelope Analysis Parameters 45 
  1.7.4. Features Selection              47 
1.8. WVU Simulation Environment       49 
1.9. WVU Self/Non-self Analysis Tool   51 

Chapter 2  Immunity-based Detection of Aircraft Subsystem Abnormal Conditions  53 
2.1. Detection through Direct Self/Non-self Discrimination  53 
  2.1.1. General Aspects                  53 
  2.1.2. Generation of Detectors          54 
  2.1.3. Analysis of Detection Performance of Individual Selves 55 
  2.1.4. Detection Logic for Direct Self/Non-self Discrimination Scheme 56 
  2.1.5. Detection Performance Using Self/Non-self Discrimination 59 
2.2. Dendritic Cell Mechanism for Abnormal Condition Detection  61
2.2.1. The Artificial Dendritic Cell
2.2.2. Detection Performance Using the DC Mechanism

Chapter 3 Immunity-based Identification of Aircraft Subsystem Abnormal Conditions

3.1. Structured Non-self Approach for Abnormal Condition Identification
3.1.1. Principles of AC Identification Using the Structured Non-self Approach
3.1.2. Identification Algorithm Using the Structured Non-self Approach
3.1.3. Generation of Identifiers
3.1.4. The Two Phases of the SNSA Approach
3.1.5. AC Identification Performance Using Self/Non-self Discrimination
3.2. Dendritic Cell Mechanism for Abnormal Condition Identification
3.2.1. Dendritic Cell Mechanism for AC Identification
   A). Feature-pattern (F-P) approach for AC identification
   B). Projection-pattern (P-P) approach for AC identification
   C). Matrix-pattern (M-P) approach for AC identification
3.2.2. Performance of AC Identification Using a Dendritic Cell Mechanism

Chapter 4 Immunity-based Direct Evaluation of Aircraft Subsystem Abnormal Conditions

4.1. Structured Non-self Approach for Abnormal Condition Direct Evaluation
4.1.1. Structured Non-self Approach for Qualitative Direct Evaluation
4.1.2. Structured Non-self Approach for Quantitative Direct Evaluation
4.1.3. Performance of Direct Evaluation Using the SNSA
4.2. Dendritic Cell Mechanism for Abnormal Condition Direct Evaluation
4.2.1. Dendritic Cell Approach for Qualitative Direct Evaluation
   A). Feature-pattern (F-P) approach for AC qualitative evaluation
   B). Projection-pattern (P-P) approach for AC qualitative evaluation
   C). Matrix-pattern (M-P) approach for AC qualitative evaluation
4.2.2. Dendritic Cell Approach for Quantitative Direct Evaluation
4.2.3. Performance of Direct Evaluation Using the Dendritic Cell Approach

Chapter 5 Flight Envelope Estimation Under Abnormal Conditions Using Artificial Immune System

5.1. Problem Formulation
5.2. Estimation Algorithms for Actuator AC
   5.2.1. Stabilator Failure
   5.2.2. Aileron Failure
   5.2.3. Rudder Failure
   5.2.4. Throttle Failure
5.3. Estimation Algorithms for Sensor AC
   5.3.1. Roll Rate Sensor Bias
   5.3.2. Pitch Rate Sensor Bias
   5.3.3. Yaw Rate Sensor Bias
5.4. Estimation Algorithms for Structural Damage
   5.4.1. Wing Damage
   5.4.2. Horizontal Tail Damage
   5.4.3. Vertical Tail Damage
5.5. Estimation Algorithms for Propulsion AC
5.6. Performance of Flight Envelope Assessment Under AC
   5.6.1. Evaluation Metrics for Flight Envelope Assessment
   5.6.2. Flight Envelope Assessment Under Actuator Failure
   5.6.3. Flight Envelope Assessment Under Sensor Failure
   5.6.4. Flight Envelope Assessment Under Structural Damage
   5.6.5. Flight Envelope Assessment Under Propulsion System Failure

Chapter 6 Investigation of Immunity-based Accommodation of Aircraft Subsystem Abnormal Conditions
   6.1. Immunity Approaches for AC Accommodation
   6.2. The Humoral Immune Mechanism for Abnormal Condition Accommodation
      6.2.1. General Control Architecture
      6.2.2. Analysis of AIS Compensation Performance
   6.3. Structured Self/Non-self for Control
      6.3.1. Formulation of Mathematical Model
      6.3.2. Generating AIS Memory
      6.3.3. Matching Algorithm
      6.3.4. Example Results Using the Structured Self/Non-self for Control
         A). Simulation Scenarios
         B). System Implementation
         C). Simulation Results for Symmetric Climb
         D). Simulation Results for Coordinated Half-Turn

Chapter 7 Integrated Scheme for Aircraft Subsystem ACDIE
   7.1. Integrated ACDIE Scheme Using the Structured Self/Non-self Approach
   7.2. Integrated ACDIE Scheme Using the DC mechanism-based Approach

Chapter 8 Demonstration of the Integrated Scheme for Aircraft Subsystem ACDIE
   8.1. Implementation of the Integrated Scheme for Aircraft Subsystem ACDIE
   8.2. Demonstration of the Integrated Scheme for Aircraft Subsystem ACDIE Through Simulation
   8.3. Demonstration of the AIS AC Detection Through Flight Testing Using a UAV Research Platform
      8.3.1. Flight Test Scenario
      8.3.2. Test Platform
      8.3.3. Simulink Models
      8.3.4. Flight Test Results

References

APPENDIX A - List of Publications Resulting from the DARPA Sponsored Research Effort
APPENDIX B - Aircraft Subsystem ACDIE Scheme – Simulink Implementation (on DVD)
APPENDIX C - Installation and User Manual for the Aircraft Subsystem ACDIE Scheme
List of Figures

Figure 1.1. Phagocytes in the Peripheral Tissue 19
Figure 1.2. A Dendritic Cell Processing an Antigen 20
Figure 1.3. Activation of Th-, B-, and Tc-Cells by the APC 21
Figure 1.4. The Negative Selection Process 22
Figure 1.5. Design of AIS-Based Abnormal Condition Detection Scheme 24
Figure 1.6. Self/Non-Self 2-Dimensional Projection 30
Figure 1.7. AIS-Based ACM Process 31
Figure 1.8. AIS-Based ACM System Design 32
Figure 1.9. On-Line AC Detection, Identification, Evaluation, and Accommodation 32
Figure 1.10. Post-Processing of Flight Data and ACDIEA Outcomes 33
Figure 1.11. Potential of Sub-Selves to Capture the Abnormal Conditions 35
Figure 1.12. Self/Non-Self Generation Process 37
Figure 1.13. Two Methods for AIS Antibodies Generation 38
Figure 1.14. 2-D Self Clusters with AIS Antibodies Generated Using the Raw Data Set Union Method 39
Figure 1.15. 2-D Self with 22000 Clusters 41
Figure 1.16. 2-D Self with 6343 Clusters 41
Figure 1.17. Sample 2-D Self Clusters with AIS Antibodies Generated Using the Cluster Set Union Method 41
Figure 1.18. Detection Performance Comparison of Detectors Obtained with CSUM and RDSUM for a Stabilator Failure 42
Figure 1.19. The Hierarchical Multi-Self Strategy for ACDIE 44
Figure 1.20. The WVU 6-DOF Motion-based Flight Simulator 49
Figure 1.21. Simulink Supersonic Fighter Aircraft Model. 50
Figure 1.22. Testing Flight Envelope 50
Figure 1.23. The Interactive Tool for Self/Non-self Analysis 51
Figure 1.24. The Zooming Action of the Interactive Tool for Self/Non-self Analysis 52

Figure 2.1. Cluster Set Union Based Logic 54
Figure 2.2. Detection Rate of Left Aileron Locked at 2.5 deg Failure 57
Figure 2.3. Detection Rate of Left Stabilator Locked at 2.0 deg Failure 57
Figure 2.4. Detection Rate of 6% Left Wing Loss Failure 57
Figure 2.5. Detection Rate of Pitch Sensor Bias of 5 deg/sec Failure 58
Figure 2.6. Detection Rate of Self 87 for 4 Different Failures 58
Figure 2.7. On-Line Failure Detection Logic with SNSA 59
Figure 2.8. Block Diagram of the DC Algorithm for Failure Detection 64
Figure 2.9. Data Structure of an Artificial Dendritic Cell 64
Figure 2.10. Sum of Residual Cytotoxic T-cells vs. Time of a Nominal Flight Test 65
Figure 2.11. Sum of Residual Cytotoxic T-cells vs. Time of an AC Flight Test: Right Aileron Locked at 8°. Actual Failure Time = 40 s; Detection Time = 40.76 s 65
Figure 2.12. Sum of Residual Cytotoxic T-cells vs. Time of an AC Flight Test: 3° LFDB in Yaw Rate Sensor. Actual Failure Time = 40 s; Detection Time = 43.38 s 65
Figure 2.13. Sum of Residual Cytotoxic T-cells vs. Time of an AC Flight Test: 15% Loss of the Left Wing. Actual Failure Time = 40 s; Detection Time = 40.52 s 65
Figure 6.7. Baseline Controller Augmented with AIS-based Mechanism
Figure 6.8. Results of Average PI for Different ACs
Figure 6.9. Roll Response Under High Magnitude Stabilator Failure
Figure 6.10. Pitch Response Under High Magnitude Stabilator Failure
Figure 6.11. Total Pilot Activity Under High Magnitude Stabilator Failure
Figure 6.12. Average Performance Index for Tracking Error
Figure 6.13. Task Execution Time Histogram
Figure 6.14. Time per Simulation vs Simulation Count
Figure 6.15. Failure Accommodation Definition
Figure 6.16. Accommodation Problem Main Components
Figure 6.17. AIS Memory Structure
Figure 6.18. Symmetric Uniform Climb
Figure 6.19. Coordinated Half Turn
Figure 6.20. Aircraft Altitude at Nominal
Figure 6.21. Longitudinal Channel Commands
Figure 6.22. Aircraft Altitude
Figure 6.23. Longitudinal Channel Commands
Figure 6.24. Lateral Channel Commands
Figure 6.25. Lateral Aircraft Position Coordinate
Figure 6.26. Aircraft Bank Angle
Figure 6.27. Aircraft Altitude Under Sensor Failure
Figure 6.28. Longitudinal Channel Commands
Figure 6.29. Lateral Channel Commands
Figure 6.30. Aircraft Bank Angle
Figure 6.31. Coordinated Turn Flight Trajectory Under Nominal Conditions
Figure 6.32. Lateral Channel Commands for Coordinated Turn at Nominal Conditions
Figure 6.33. Aircraft Bank Angle
Figure 6.34. Longitudinal Channel Commands
Figure 6.35: Aircraft Altitude - Nominal Conditions Coordinated Half Turn
Figure 6.36. Coordinated Turn Trajectory Under Stab Failure
Figure 6.37. Longitudinal Channel Commands Under Stab Failure
Figure 6.38. Lateral Channel Commands Under Stab Failure
Figure 6.39. Aircraft Bank Angle Under Stab Failure
Figure 6.40. Flight Trajectory for Coordinated Turn Under Roll Rate Sensor Failure
Figure 6.41. Lateral Channel Commands Under Roll Rate Sensor Failure
Figure 6.42. Longitudinal Channel Commands for Coordinated Turn Under Roll Rate Sensor Failure

Figure 7.1. ACDIE with the Structured Non-Self Approach
Figure 7.2. ACDIE with the DC Mechanism Approach

Figure 8.1. Top-level Simulink Model of the Integrated Aircraft ACDIE Using the Structured Non-Self Approach
Figure 8.2. Simulink Block Diagram of the Structured Non-Self Approach
Figure 8.3. Top-level Simulink Model of the Integrated Aircraft ACDIE Scheme Using the DC Mechanism
Figure 8.4.  Simulink Block Diagram of the DC Mechanism  155
Figure 8.5.  ACDIE Outcome Visualization Interface  156
Figure 8.6.  Demonstration Results for a Nominal Flight  159
Figure 8.7.  Demonstration Results for a 15% Damage of the Left Wing  159
Figure 8.8.  Skywalker 1880 RC  161
Figure 8.9.  Onboard APM 2.5  161
Figure 8.10.  Simulink Model Top Level  163
Figure 8.11.  Simulink Sensor Model  163
Figure 8.12.  Kalman Filter Architecture  164
Figure 8.13.  Self #1 Detector Activity for Left Aileron Locked at Bank Maneuver  167
Figure 8.14.  Self #17 Detector Activity for Left Aileron Locked at Bank Maneuver  167
Figure 8.15.  Self #18 Detector Activity for Left Aileron Locked at Bank Maneuver  168
Figure 8.16.  Method 2 Detector Activity for Left Aileron Locked at Bank Maneuver  168
Figure 8.17.  Method 1 Detector Activity for Left Aileron Locked at Bank Maneuver  169
List of Tables

Table 1. Activity Chart for the Project 14
   Table 1.1. Main Biological Terms and Their AIS Paradigm Representations 24
   Table 1.2. Parameters for AC Indirect Evaluation Analysis 45
   Table 1.3. Features for Self/Non-self Definition 47
   
   Table 2.1. Features of Selected Projections 55
   Table 2.2. Detection Performance of Selected Individual Selves 56
   Table 2.3. Aircraft Subsystems Considered for AC Detection 59
   Table 2.4. Average Detection Performance of the SNSA Scheme 60
   Table 2.5. Average DC-based Detection Rate for Different Subsystems 65
   
   Table 3.1. Self/Non-self 2-D Projections for SNSA Approach 69
   Table 3.2. Phase I SNSA Subsystem Failures 73
   Table 3.3. Detection Sensitivity of a Sample Set of Projections (% DR) 74
   Table 3.4. Total Number of Projections Activated per Failure 75
   Table 3.5. Average Identification Rates Using the DC Mechanism 80
   
   Table 4.1. Projections Used for Simultaneous Identification and Direct Evaluation 86
   Table 4.2. Total Number of Projections Used for Simultaneous Identification and Direct Evaluation 86
   Table 4.3. Performance of AC Identification and Direct Evaluation 87
   Table 4.4. AC Detection and Qualitative Evaluation Rates of the DC Mechanism Scheme 94
   
   Table 5.1. Performance of Flight Envelope Prediction Under Stabilator Failure 123
   Table 5.2. Performance of Flight Envelope Prediction Under Aileron Failure 124
   Table 5.3. Performance of Flight Envelope Prediction Under Rudder Failure 124
   Table 5.4. Performance of Flight Envelope Prediction Under Roll Rate Sensor Failure 125
   Table 5.5. Performance of Flight Envelope Prediction Under Pitch Rate Sensor Failure 125
   Table 5.6. Performance of Flight Envelope Prediction Under Yaw Rate Sensor Failure 125
   Table 5.7. Performance of Flight Envelope Prediction Under Wing Damage 126
   Table 5.8. Performance of Flight Envelope Prediction Under Horizontal Tail Damage 126
   Table 5.9. Performance of Flight Envelope Prediction Under Engine Failure 127
   
   Table 6.1. Computer Characteristics for Execution Time Assessment 136
   Table 6.2. Task Execution Time Assessment 136
   
   Table 8.1. Detection Results for a Set of Demonstration Tests 157
   Table 8.2. Identification Results for the Demonstration Tests 157
   Table 8.3. Failure Type Evaluation Results for the Demonstration Tests 158
   Table 8.4. Failure Severity Evaluation Results for the Demonstration Tests 158
   Table 8.5. Failures Injected in Flight Tests 160
   Table 8.6. Skywalker Dimensions and Mass Properties 161
   Table 8.7. Self/Non-self Features for Flight Testing 164
Table 8.8. Skywalker Self Projections 165
Table 8.9. Detection Rate and False Alarm Rate for 4 Failures 165
Table 8.10. Detection Rates and False Alarms for 3 Configurations of Selves 166
# List of Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC</td>
<td>abnormal condition</td>
</tr>
<tr>
<td>ACDIEA</td>
<td>abnormal condition detection, identification, and evaluation</td>
</tr>
<tr>
<td>ACDIEA</td>
<td>abnormal condition detection, identification, evaluation, and accommodation</td>
</tr>
<tr>
<td>ADALINE</td>
<td>adaptive linear network</td>
</tr>
<tr>
<td>AIS</td>
<td>artificial immune system</td>
</tr>
<tr>
<td>ANN</td>
<td>artificial neural network</td>
</tr>
<tr>
<td>APC</td>
<td>antigen presenting cells</td>
</tr>
<tr>
<td>CSUM</td>
<td>cluster set union method</td>
</tr>
<tr>
<td>DC</td>
<td>dendritic cells</td>
</tr>
<tr>
<td>DIV</td>
<td>directly involved variable</td>
</tr>
<tr>
<td>DOF</td>
<td>degrees-of-freedom</td>
</tr>
<tr>
<td>DQEE</td>
<td>decentralized quadratic estimation error</td>
</tr>
<tr>
<td>DR</td>
<td>detection rate</td>
</tr>
<tr>
<td>EDIV</td>
<td>equivalent directly involved variables</td>
</tr>
<tr>
<td>ERAU</td>
<td>Embry-Riddle Aeronautical University</td>
</tr>
<tr>
<td>ERV</td>
<td>envelope relevant variables</td>
</tr>
<tr>
<td>F-P</td>
<td>feature-pattern</td>
</tr>
<tr>
<td>FA</td>
<td>false alarm</td>
</tr>
<tr>
<td>FN</td>
<td>false negative</td>
</tr>
<tr>
<td>FP</td>
<td>false positive</td>
</tr>
<tr>
<td>HMS</td>
<td>hierarchical multi-self</td>
</tr>
<tr>
<td>IQE</td>
<td>indirect quantitative evaluation</td>
</tr>
<tr>
<td>M-P</td>
<td>matrix-pattern</td>
</tr>
<tr>
<td>MHC II</td>
<td>major histocompatibility complexes II</td>
</tr>
<tr>
<td>MQEE</td>
<td>main quadratic estimation error</td>
</tr>
<tr>
<td>NCE</td>
<td>no-cost-extension</td>
</tr>
<tr>
<td>NLDI</td>
<td>nonlinear dynamic inversion</td>
</tr>
<tr>
<td>NS</td>
<td>negative selection</td>
</tr>
<tr>
<td>OQEE</td>
<td>output quadratic estimation error</td>
</tr>
<tr>
<td>P-P</td>
<td>projection-pattern</td>
</tr>
<tr>
<td>PID</td>
<td>proportional-integral-derivative</td>
</tr>
<tr>
<td>PR</td>
<td>prediction rate</td>
</tr>
<tr>
<td>PS</td>
<td>positive selection</td>
</tr>
<tr>
<td>RDSUM</td>
<td>raw data set union method</td>
</tr>
<tr>
<td>SNSA</td>
<td>structured non-self approach</td>
</tr>
<tr>
<td>TN</td>
<td>true negative</td>
</tr>
<tr>
<td>TO</td>
<td>technical objectives</td>
</tr>
<tr>
<td>TP</td>
<td>true positive</td>
</tr>
<tr>
<td>UAV</td>
<td>unmanned aerial vehicle</td>
</tr>
<tr>
<td>WVU</td>
<td>West Virginia University</td>
</tr>
</tbody>
</table>
Project Summary

1. Task Objectives

The general goal of this research effort was to design and develop a novel biologically inspired framework for flight envelope estimation and protection, which is conceptually integrated with the abnormal detection and accommodation processes and is general and comprehensive such that it can operate at all conditions, normal and abnormal, a priori known or unknown. The proposed methodology relies primarily on the artificial immune system (AIS) paradigm. To reach this objective, the following 3 technical objectives (TO) with a total of 14 main research tasks and one additional task over the no-cost-extension period have been planned.

TO #1. Develop a novel bio-mimetic framework for flight envelope estimation and protection under upset flight conditions.
Task #1.1. Problem formulation including definition of abnormal conditions addressed, parameter ranges, and envelope parameters targeted.
Task #1.2. Development of schemes for qualitative failure evaluation.
Task #1.3. Development of schemes for direct quantitative failure evaluation.
Task #1.4. Development of schemes for indirect quantitative failure evaluation. Flight envelope reduction evaluation.
Task #1.5. Design and execution of flight simulator tests for AIS abnormal condition evaluation schemes development.

TO #2. Investigate the use of the artificial immune system paradigm for aircraft adaptive guidance and control under abnormal conditions.
Task #2.1. Formulation of the AIS-based framework for flight envelope protection.
Task #2.2. Immune adaptive control laws design and implementation.
Task #2.3. System integration.
Task #2.4. Design and execution of flight simulator tests for development of AIS-based abnormal condition accommodation schemes.

TO #3. Implement and demonstrate the proposed system on a motion based flight simulator.
Task #3.1. Design of flight simulator tests for proposed methodology demonstration.
Task #3.2. Customization of existing computational tools for aircraft simulation, adaptive control laws, and artificial immune system generation.
Task #3.3. Implementation and integration of a simulation environment for design, testing, and demonstration of the novel flight envelope estimation and protection methodology.
Task #3.4. Execution of the 6-degrees-of-freedom (DOF) flight simulator tests and data processing.
Task #3.5. Analysis of results and evaluation of the novel flight envelope estimation and protection methodology.
Task #3.6 Reduced system demonstration on an unmanned aerial vehicle (UAV).

The following project milestones were defined and scheduled as follows:
M#1. Formulation of a generalized conceptual framework for flight envelope estimation based on the AIS paradigm (End of Month #8)
M#2. Flight envelope protection algorithms implemented and tested (End of Month #14)
M#3. Integrated system demonstration in the motion-based flight simulator (End of Month #18, re-scheduled at the end of no-cost-extension period)
The activity chart for the project is presented in Table 1. The periods of performance for all tasks over the initial duration of the project are marked in blue. The re-scheduled periods for Tasks 1.4, 2.3, 3.3-3.6 and Milestone #3 over the no-cost-extension (NCE) period are marked in light blue. The due dates for the three project milestones are marked with green lines.

### Table 1. Activity Chart for the Project

<table>
<thead>
<tr>
<th></th>
<th>Months during 1st Year</th>
<th>Months during 2nd Year</th>
<th>NCE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1-4</td>
<td>5-8</td>
<td>9-12</td>
</tr>
<tr>
<td></td>
<td>1-2</td>
<td>3-4</td>
<td>5-6</td>
</tr>
<tr>
<td></td>
<td>7-9</td>
<td>10-11</td>
<td>12-13.5</td>
</tr>
<tr>
<td>TO #1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TO #2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TO #3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>M#1</td>
<td>M#2</td>
<td>M#3</td>
</tr>
</tbody>
</table>

2. Technical Problems

In this project, the AIS paradigm was used to accomplish critical steps towards the development of a holistic framework for an effective, comprehensive, and integrated solution to the problem of aircraft operation under abnormal conditions (AC). The complete aircraft AC management process is envisioned as consisting of four major components: AC detection, identification, evaluation, (ACDIE) and accommodation (ACDIEA). From this perspective, performing aircraft AC management becomes an extremely challenging, complex, and multi-dimensional task.

In continuation of previous research establishing conceptual and practical approaches for AIS-based AC detection and identification, this project was primarily focused on addressing the qualitative and quantitative evaluation of abnormal flight conditions consisting of failures/damages affecting aircraft actuators, sensors, propulsion system, and aerodynamic surfaces. The AC evaluation process is highly specific and requires individualized approaches due to the synergistic interaction between aircraft subsystems, aircraft states, and the nature of the ACs. Within the project, classes of AC were considered that alter subsystem aerodynamics in a manner consistent with and predictable based on aircraft operation at normal condition, referred to as “self” within the AIS paradigm. Integration of AC evaluation with prior detection and identification was given significant attention.
The multi-dimensionality of the feature (or relevant variable) space implied by the development of a comprehensive AIS creates computational and logical issues that need to be addressed carefully. The use of lower dimensional projections of the self (sub-selves) within the hierarchical multi-self strategy was proposed in this project. It should be noted that the issue of computational time for off-line and on-line processing has not been specifically targeted within the project.

The AC accommodation problem was also addressed within the project. The possibility of using immunity mechanisms for adaptive control and compensation of abnormal flight conditions was investigated along two directions. One relied on the antibody activation/suppression mechanism converted into an adaptive control approach. The other, relied on extending the classification capabilities of the AIS and using them not only to detect and diagnose the problem, but also to select or find the solution. In other words, once a certain abnormal situation is detected and identified, the AIS-based system can determine the adequate compensation.

3. General Methodology

Developing a generalized conceptual framework (comprehensive and integrated) for the detection, identification, evaluation, and accommodation of aircraft sub-system abnormal conditions requires specific tools that can handle the high complexity and multi-dimensionality of aircraft dynamic response in the context of abnormal conditions. The tools for the development of the proposed methodology rely on a new concept inspired from the biological immune system. The immune system protects the body against intruders by recognizing and destroying harmful cells or molecules through simple in principle yet powerful mechanisms, such as negative/positive selection, mutation, and cloning, which exhibit robust, adaptive, and highly distributed cognitive capabilities. It can be thought of as a robust adaptive system that is capable of dealing with an enormous variety of known and unknown disturbances and uncertainties. Another critical aspect of the bio-immune system is that it can “remember” how previous encounters were successfully handled. As a result, it can respond faster to similar encounters in the future. The AIS is capable of mimicking these mechanisms (such as negative/positive selection, antibody activation/suppression, state/resource assessment, and memory) to solve the problem of aircraft subsystem ACDIEA in a general and comprehensive manner.

The failure evaluation process must address several distinct aspects such as determining the type of the failure, its magnitude or severity, and evaluating the reduction of the flight envelope due to the failure, in the most general sense. Evaluation requires the previous correct detection and identification of the anomalous conditions. Detection is the process of declaring that a generic malfunction of the system has occurred in any of the subsystems, while the identification implies determining what is the failed subsystem and/or component. These two aspects are important to diagnose the effect of the failure on the aircraft operational limits and provide necessary information to the pilot and the automatic control system to avoid commands that might lead to loss of control and other dangerous/catastrophic situations.

The computational issues associated with high-dimensionality hyper-spaces is addressed by using lower dimensional projections of the self/non-self within the hierarchical multi-self (HMS) strategy, which relies on two main concepts. First, the normal/abnormal discrimination capability of complete sets of projections is the same as the entire hyper-self under certain
conditions. Second, projections along specific feature axes have better capabilities in capturing the dynamic fingerprint of specific AC than others. This hierarchical ranking of sub-selves with respect to targeted AC can be used to develop less computationally intensive detection, identification, and evaluation schemes.

The AIS paradigm is based only on data acquisition and structuring and does not need sophisticated modeling of the targeted system. However, this requires extensive amount of data, mostly at normal operational conditions, and specific processing in some instances. For the purpose of this project, data for a supersonic fighter were acquired from a motion-based 6-DOF flight simulator.

This research effort has been performed as a close collaboration between researchers from West Virginia University (WVU) and Embry-Riddle Aeronautical University (ERAU).

4. Technical Results

The proposed AIS-based generalized conceptual framework for aircraft AC management has been demonstrated to be able to handle simultaneously several major aircraft subsystems, under known and unknown abnormal conditions of different types, and severities. The subsystems considered under abnormal conditions were: actuators (left/right stabilator, aileron, rudder), sensors (pitch, roll, yaw gyros), propulsion (throttle), structural components (left/right wing, horizontal tail, vertical tail).

Two different methodologies were proposed, developed, and successfully tested for AC detection, identification, and evaluation: dendritic cell (DC) mechanism and structured non-self approach. Integrated schemes were implemented and evaluated in terms of false alarms and success rates for detection, identification, and evaluation. In most cases considered, the proposed ACFDIE schemes achieve zero false alarms and success rates between 95-100%.

Specific algorithms for post-failure flight envelope evaluation have been developed, implemented and successfully tested. The post-failure flight envelope evaluation is defined in terms of relevant features and targets classes of AC that affect the aerodynamics of the system in a manner consistent with the operation at normal conditions.

The preliminary investigation of AIS-based abnormal condition accommodation included the use of antibody activation-suppression mechanism for adaptive control laws and the transfer of immunity memory capabilities to AIS for enabling for pilot compensatory commands extraction. Both approaches have been implemented and successfully tested demonstrating very promising potential.

The outcomes of this project have been widely disseminated. The record to date of peer-reviewed journal publications includes 3 published papers, 7 in review, and 1 in preparation. The record to date of peer-reviewed conference publications includes 7 papers presented and published and 2 papers accepted (see Appendix A for complete list of titles).

5. Important Findings and Conclusions

The technical outcomes of this project include the formulation of novel concepts and methodologies that are expected to improve pilot situational awareness and improve the guidance and control capabilities of aircraft systems under normal and abnormal flight conditions. The proposed framework for ACDIEA has been demonstrated to possess the capability for providing
a comprehensive solution to the problem of aircraft AC management including flight envelope estimation and protection at post-failure conditions.

A set of integrated computational tools with a high degree of flexibility, portability, and modularity have been implemented and successfully tested. They can be used for the design, development, testing, simulation, and evaluation of fault tolerant control laws and schemes for ACDIEA.

Specific algorithms within the AIS paradigm have been developed and demonstrated to achieve high performance rates for various phases of the aircraft ACDIEA process.

The proposed methodology has the capabilities for facilitating the design of advanced aircraft with high survivability, improved operation safety, and optimized performance at both normal and abnormal/upset conditions.

The proposed framework was used to design, implement, and test through simulation integrated schemes for aircraft AC detection, identification, and evaluation that achieved excellent performance for a large variety of scenarios.

6. Significant Hardware Development
Development of hardware was not part of this project.

7. Special Comments
There are no special comments relative to the technical aspects of this project.

8. Implications for Future Research
The outcomes of this project create premises for research continuation in the following directions.

1). Extension of AC spectrum for post-failure evaluation. The algorithms developed within this project address specific classes of ACs. For a complete and comprehensive AC management process, additional classes of AC should be investigated.

2). Further investigation and development of AIS-based abnormal condition accommodation. Solving the abnormal condition accommodation problem within the AIS paradigm is expected to provide a compact, consistent, and relatively easy to implement framework that will address completely all major processes related to fault tolerant control (abnormal condition detection, identification, evaluation, and accommodation). This research effort should produce a novel adaptive architecture based on hybrid approaches and artificial intelligence techniques to increase robustness, autonomy, and safety of aircraft systems.

3). Testing the proposed methodology in flight using a remotely piloted reduced size aircraft. The proposed AIS-based framework for aircraft AC management should be implemented and tested under more realistic conditions using existing aerial platforms. This project is expected to provide the opportunity to apply and investigate lessons learned, fine-tune and extend algorithms, and validate the proposed methodology through actual flight tests. It is expected that issues related to memory allocation and computational time during both off-line development and on-board operation will be addressed.

4). Extension of the AIS paradigm into an alternative or complementary modeling methodology for dynamic systems. The versatility of the AIS and its capabilities to store
extensive system information can be expanded through combination with mechanisms inspired by genetics to develop methodologies for building data driven system models that are expected to remove or mitigate the burden of complex and challenging modeling tasks. The proposed strategy is to rely on experimental and analytical data and information that is typically available for a technical system for building a representation of the system that includes all the elements necessary for system build up and operation, similar to the genome of biological organisms. This will create the premises for the development of intelligent, evolving system with a wide range of applications.
1.1. Biological Immune System

The biological immune system consists of two equally important components: the innate system and the adaptive system\(^1\). The former is an inherited system that functions as the first line of defense against invading entities, while the latter is an acquired system built through previous exposures to invading agents. The innate immune system is always active and responds immediately to any class of pathogen without distinction. On the other hand, the adaptive immune system is normally silent and is much more potent in recognizing specific antigens with slower response.

The immune system is composed of cells that are developed from stem cells in the bone marrow and differentiate into different populations, the most important of which in the immune response are macrophages (MΦs), dendritic cells (DCs), T-cells, and B-cells. The MΦ and DC populations form what is known as phagocytes (part of the innate system) whereas populations of T-cells and B-cells form the lymphocytes (part of the adaptive system).

Phagocytes reside in the peripheral tissues searching for antigens (Figure 1.1). Proteins (or any other markers) of an antigen are recognized by the surface of the phagocyte. The phagocyte engulfs the pathogen and breaks it up into its constituent molecules. Special proteins in the phagocyte, called major histocompatibility complexes II (MHC II), bind to subsets of these molecules (specifically, peptides) and transport to the outer surface of the phagocyte along with the bound antigen peptides (Figure 1.2). MHC II play a critical role in activating the cell-mediated response of the adaptive system when the phagocyte migrates from the peripheral tissues to the lymph nodes where lymphocytes reside. B-cells process antigens in a very similar way to the phagocytes, except that phagocytes process general, non-specific antigens whereas B-cells can only process very specific types of antigens. For this reason, phagocytes are called professional antigen presenting cells (APCs). This communication is so crucial that the adaptive response cannot take place without the innate response first.

---

**Figure 1.1.** Phagocytes in the Peripheral Tissue
Figure 1.2. A Dendritic Cell Processing an Antigen

After they are developed from stem cells in the bone marrow, DCs move to peripheral tissues that are in contact with the environment, such as skin and inner lining of the nose, lungs, stomach, intestines, and oral cavities where they are in their initial “immature” state. They become mature and migrate to regions rich in T-cells, primarily the lymph nodes, whether they have already captured and processed an antigen or not (steady-state)\(^2\). In the lymph nodes, they either present peptides of the processed antigen to the T-cells to activate the adaptive immune response or to induce immune tolerance to “harmless” antigens, including those from the body’s own tissues, cells, and proteins which prevents the immune response from attacking self cells, a disease known as autoimmunity\(^3,4\).

T-cells, which mature in the thymus, exist in two main types in the lymph nodes: Helper T-cells (Th) and cytotoxic T-cells (Tc). Referring to Figure 1.3, when a DC migrates to the lymph node carrying MHC II-peptide complexes, it attracts Th-cells to bind to that MHC II-peptide complex. Note that only those Th-cells with receptors “specific” to the MHC II-peptide complex of the migrated DC can bind to it. Once bound, a Th-cell is said to be activated, upon which it proliferates and differentiates into “memory” Th-cells and “effector” Th-cells. Memory Th-cells stay in the body for years to provide faster response when the same antigen infects the body once again; this is a feature of the adaptive immune system the innate system lacks, where the adaptive memory provides stronger and faster response which effectively stops the infection with less reliance on the innate system when it occurs the second time. Effector Th-cells, on the other hand, raise the alarm by releasing small cell-signaling protein molecules known as cytokines responsible of activating both Tc-cells and B-cells. It is worth mentioning here that any infected cell in the body displays peptides of the infectious antigen on their surface via MHC I (MHC I molecules are displayed by any nucleated cell to enable the body to recognize infected cells whereas MHC II molecules are displayed by APCs, such as DCs, MΦs, and B-cells to recognize epitopes of exogenous antigens and discriminate self from non-self). The role of the Tc-cells is to kill the infected cells by binding its epitopes to that “specific” MHC I-peptide complex and releasing special proteins. These proteins are the *perforin*, which inserts itself into the infected cell membrane and forms a pore and the *granzyme*, which induces *apoptosis* (the healthy programmed cell death) in the infected cell. This adaptive immune response from T-cells is referred to as *cell-mediated response*.

B-cells, which mature in the bone marrow, carry “specific protein complexes on their surfaces known as *membrane-bound antibodies* formed by shuffling the DNA during the maturation of these B-cells. These antibodies bind to foreign pathogens with the same protein combination. Upon binding, B-cells process the antigen in a similar way to that done by phagocytes, as mentioned earlier, and display part of the processed antigen on their surface via
the MHC II-peptide complex. However, B-cells are not activated until they receive the cytokines released by the effector T<sub>H</sub>-cells that are stimulated by the migrated DCs. When activated, B-cells proliferate and differentiate into “memory” B-cells and “effector” B-cells. Like memory T<sub>H</sub>-cells, memory B-cells live for years in the body to provide faster B-cell response in case the same pathogen invades the body once again, while “effector” B-cells produce clones of their membrane-bound antibodies and release them as free antibodies. One of the important functions of these free antibodies is opsonization, in which the antibodies bind to foreign antigens and mark them for attack by phagocytes. The adaptive immune response from B-cells is called humoral response.

**Figure 1.3.** Activation of Th-, B-, and Tc-Cells by the APC

The negative selection (NS) process, through which important constituents of the immune system are generated, produces biological agents that have the capability to detect microbial and non-microbial exogenous entities (referred to as non-self) while not reacting to the cells of their own organism (referred to as self). Briefly, the process allows the proliferation of cells that do not match the self, but match the non-self. Therefore, they are only compatible with the non-self and will be capable to detect its presence. The concept is illustrated in Figure 1.4, where the term “antibodies” is used in a most generic way.
Figure 1.4. The Negative Selection Process
1.2. Artificial Immune System Paradigm

The AIS emerged in recent years as a new computational intelligence paradigm\textsuperscript{5,6} with a variety of applications in areas such as anomaly detection, data mining, computer security, adaptive control, and pattern recognition. All initial efforts were directed towards immunity-based fault detection\textsuperscript{7,8}, which operates in a similar manner as does the biological immune system - according to the principle of self-non-self discrimination - when it detects exogenous antigens while not reacting to the self cells. Prior to this project, an integrated set of methodologies\textsuperscript{9-12} for AIS-based detection and identification of a wide variety of aircraft sub-system failures/damages has been designed and implemented at WVU by the authors of this report. Integrated high-performance AIS-based failure detection and identification schemes have been demonstrated to be capable of handling several categories of sub-system abnormal conditions over extended areas of the flight envelope\textsuperscript{13}. The potential of the AIS paradigm for flight envelope reduction assessment at post-failure conditions has also been investigated with promising results\textsuperscript{14}.

The basic idea supporting the AIS paradigm for sub-system fault detection is that an abnormal situation (i.e. failure of one of the aircraft sub-systems, which is considered similar to an invasion by antigens) can be declared when a current configuration of “features” does not match with any configuration from a pre-determined set known to correspond to normal situations. These “features” – similar to the biological chemical markers – represent the encoding of the self. They can include various sensor outputs, states estimates, statistical parameters, or any other information expected to be relevant to the behavior of the system and able to capture the signature of abnormal situations. Extensive experimental data are necessary to determine the self or the hyper-space of normal conditions. Adequate numerical representations of the self/non-self must be used and the data processed such that they are manageable given the computational and storage limitations of the available hardware. The artificial counterpart of the antibodies - the detectors - must then be generated and optimized. This process typically attempts to mimic the variation followed by selection of the T-cells. The mechanisms through which DC acquire, store, process, and transfer information can be used to handle the large amounts of data involved in the ACDIEA process. Handling computationally the large feature hyper-space as a whole is impractical if at all possible for a comprehensive approach. Alternatively, lower order projections of the self can be used, but then self/non-self discrimination outcomes from a large number of such projections must be handled. A DC-based mechanism was developed for this purpose that will be described in details in following sections.

A summary of main components of the AIS paradigm and their biological source is presented in Table 1.1.

The NS process through which main cellular components of the biological immune system are generated is mimicked in the generation of detectors. Evolutionary algorithms can be used for optimization in this process\textsuperscript{11}. As an alternative mechanism, positive selection (PS) has been explored for AIS design. Through the PS strategy, the detectors are generated to coincide with the self and the process is equivalent to clustering the self data. In this case, an abnormal situation is declared if the explored current configuration does not match any of the detectors. For detection purposes, NS and PS are equivalent; however, using PS within a detection scheme is typically more computationally intensive than using NS because it is necessary to test the complete set of positive antibodies before classifying a sample as abnormal. With the NS approach, the activation of a single negative antibody is enough to declare the presence of an abnormal situation.
A detection logic must be designed for real time operation with a high detection rate and low number of false alarms. The block diagram of the AIS design process for aircraft sub-system abnormal condition detection is presented in Figure 1.5.

### Table 1.1. Main Biological Terms and Their AIS Paradigm Representations

<table>
<thead>
<tr>
<th>Biological Term</th>
<th>AIS Paradigm Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organic markers (proteins and other compounds)</td>
<td>System features or characteristic variables and their values.</td>
</tr>
<tr>
<td>Organism or self</td>
<td>Data clusters of feature values acquired from development tests under normal operating conditions.</td>
</tr>
<tr>
<td>Alien entities or non-self</td>
<td>Set of complementary clusters covering the feature hyper-space outside of self clusters.</td>
</tr>
<tr>
<td>Antigen</td>
<td>Abnormal condition or set of feature values at abnormal conditions</td>
</tr>
<tr>
<td>Antibody</td>
<td>Cluster of the non-self or detector</td>
</tr>
<tr>
<td>Dendritic cell</td>
<td>A computational unit that processes the outcomes of the self/non-self discrimination process.</td>
</tr>
</tbody>
</table>

**Figure 1.5.** Design of AIS-Based Abnormal Condition Detection Scheme
For the identification and evaluation phases, additional information is needed to structure the non-self hyperspace, which can be obtained through a PS-type of mechanism using data under failure conditions. In particular, for the quantitative direct evaluation, the non-self must be separated into sub-regions corresponding to the different “identifiable” level of failures and PS must be used to assess the distribution of the failure signature within the non-self by labeling the antibodies corresponding to different magnitudes of the failure.
1.3. Definitions, Nomenclature, and Notation

For a general formulation, it will be assumed that the targeted system is the “aircraft”, including the vehicle itself with all its subsystems (such as actuators, sensors, propulsion, etc.), the human pilot, and the environment. However, the proposed methodology is generally formulated and can be applied to any system and its components.

The generic term of “abnormal conditions” (AC) refers to faults and failures of hardware, human pilot-related abnormal situations, operational upset conditions, extreme environmental conditions, and any other situations that require specific attention and/or action for safety purposes. It should be noted that for the practical design and implementation of ACDIEA schemes, the designer is required to define the nature, type, and severity of the AC targeted.

The detection represents the process through which the existence of an AC is assessed or, in other words, a failure within at least one of the aircraft sub-systems has been acknowledged. The outcome of the detection process \( \text{Det} \) is binary:

\[
\text{Det} \in \{0, 1\}, \quad \text{Det} = 0 \quad \text{for normal conditions}
\]
\[
\text{Det} = 1 \quad \text{for abnormal conditions}
\]  

(1.1)

The identification or isolation process determines which subsystem has failed. Depending on the complexity of the targeted system, the AC identification can be performed in several phases. For example, a first identification phase could distinguish between actuator and sensor failures. If an actuator failure is identified, a second phase could determine whether it is an elevator, aileron, or rudder failure. Finally, a third identification phase could establish that the left elevator has failed or the left in-board elevator has failed, if the aircraft is equipped with additional redundancy. The outcome of the identification process, \( \text{Idt} \), can be formulated as an \( N_s \)-dimensional vector with binary components, where \( N_s \) is the total number of sub-systems considered:

\[
\text{Idt} = [i_{d_1} \quad i_{d_2} \quad \ldots \quad i_{d_{N_s}}], \quad i_{d_j} = 0 \quad \text{for subsystem j at normal conditions}
\]
\[
\text{Idt} = [i_{d_1} \quad i_{d_2} \quad \ldots \quad i_{d_{N_s}}], \quad i_{d_j} = 1 \quad \text{for subsystem j at abnormal conditions}
\]  

(1.2)

Alternatively, the outcome of the identification process can be formulated as a set of integers labeling the failed sub-system:

\[
\text{Idt} = [i_{d_1} \quad i_{d_2} \quad \ldots \quad i_{d_{N_{sf}}}] 
\]  

(1.3)

where \( N_{sf} \) is the number of failed sub-systems and \( i_{d_j} \) represents the label associated to each subsystem, typically:

\[
i_{d_j} \in \{1 \quad 2 \quad \ldots \quad N_s\}
\]  

(1.4)

Once a failure has been detected and correctly identified, the AC evaluation process must address three aspects. One is of a qualitative nature and involves determining the type of the failure. For example, the qualitative evaluation is expected to determine if an actuator failure consists of a locked actuator, or a freely moving control surface, or a reduction of control
efficiency. The other two aspects are of a quantitative nature and can be defined as direct and indirect. The direct failure evaluation consists of estimating the magnitude or severity of the failure (e.g. left aileron locked at +10deg). The indirect failure evaluation includes the re-assessment of the flight envelope and prediction of the limitations and constraints on the performance and handling qualities inflicted by the presence of the failure.

The outcome of the qualitative evaluation, \( Evl \), is an \( Nsf \)-dimensional string of integers labeling the type of failures for all \( Nsf \) affected sub-systems out of a list of \( Nft_k \), \( k = 1, 2, \ldots Ns \), potential types associated to each sub-system \( id_j, j = 1, 2, \ldots Nsf \):

\[
Evl \in \{1 \ 2 \ \ldots \ Nft_k \} \quad (1.5)
\]

Let the set \( F \) of all failures considered be expressed as:

\[
F = \{f_1 \ f_2 \ \ldots \ f_{N_F} \} \quad (1.6)
\]

where \( N_F \) is the total number of AC considered and:

\[
N_F = \sum_{k=1}^{N_s} Nft_k \quad (1.7)
\]

The outcome of the direct quantitative evaluation process, \( Ev2 \), can take a numerical value that relates to the magnitude or severity of the AC. For example:

\[
Ev2 = 40\% \quad (1.8)
\]

where 40\% represents, let’s say, the relative area of the left aileron affected by structural damage. While possible in some cases, this level of accuracy is, obviously, difficult to achieve in general. Alternatively, the direct quantitative evaluation could provide a severity estimate mapped on a discrete set:

\[
Ev2 \in \{\text{low severity} \ \text{medium severity} \ \text{high severity}\} \quad (1.9)
\]

or, it could provide a severity evaluation based on fuzzy logic and expressed as:

\[
Ev2 = [v_1 \ v_2 \ \ldots \ v_m] \quad (1.10)
\]

where \( v_i \) are fuzzy membership values with respect to the \( m \) linguistic values associated to the severity metric \( Ev2 \) (e.g. low severity, medium severity, and high severity). \( Ev2 \) can also be expressed as a scalar by using a defuzzification algorithm:

\[
Ev2 = \Delta([v_1 \ v_2 \ \ldots \ v_m]) \quad (1.11)
\]

where \( \Delta \) is the defuzzification operator.
The outcome of the indirect quantitative evaluation process, $Ev3$, typically represents a set of new ranges at post failure conditions of the variables $y_i$ that define the flight envelope, in its most general meaning. Let the number of these variables be $Ne$. Then:

$$Ev3 = [(y_{1 \min}, y_{1 \max}) (y_{2 \min}, y_{2 \max}) \ldots (y_{N_e \min}, y_{N_e \max})]$$ (1.12)

More specifically, let a directly involved variable (DIV) $v_{\delta}$ in the AC be a variable whose alteration or abnormal variation is directly and significantly the result of the AC. Typically, DIV are used to define/characterize the AC. They may be part of the feature set or not. If they are not, then a relationship between the DIV and some other variable(s) in the feature set must be established. This process will define equivalent directly involved variables (EDIV), $v_{\epsilon}$, which are part of the feature set. For example, consider the case of the left stabilator locked failure. The DIV can be defined as the left stab deflection $\delta_{eL}$. It obviously defines the failure; however, let us assume that it is not part of the feature set that has been established. A relationship can be formulated between the left stab deflection $\delta_{eL}$ and the longitudinal stick displacement $d_e$ - which presumably is a feature – of the form $d_e = f(\delta_{eL})$. Therefore, the EDIV is in this case, $d_e$. For each failure considered, a set of variables $v_E$ must be determined that are affected by the failure, that are part of the feature set, and that are relevant from the point of view of aircraft operation (they are flight envelope relevant variables). These variables may be determined through analytical means but also through the analysis of the 2-D selves.

The accommodation process can take two forms: passive accommodation through warnings and information displayed in the cockpit and active accommodation through direct compensation as integral part of the control laws. The passive accommodation represents actually the re-processing of the outcomes of the detection, identification, and evaluation and their delivery to the pilot in a form that is easy to perceive and understand. The direct accommodation involves triggering of pre-existing compensating modules within the control laws and/or actual computation of commands at post-failure conditions.

The feature variables or shortly features are the variables $\varphi_i$ that completely define the targeted system and are expected to capture the fingerprints of all AC considered, in terms of occurrence, presence, and severity. They can be (sub-)system states, inputs, control system variables, estimated values, etc. The set of all features $\mathcal{F}$:

$$\mathcal{F} = \{\varphi_i \mid i = 1,2,\ldots,N\}$$ (1.13)

defines a feature point $P$ as a set of simultaneous values of all features $\varphi_i$ that can be obtained through measurements or simulation, at normal or abnormal conditions. The set $\mathcal{F}$ defines an $N$-dimensional hyper-space $\mathcal{U}$, which will be referred to as the “Universe”. An orthogonal coordinate system (CS) equally denoted $U$ is associated to this hyper-space. Its origin is at point $O$ with coordinates $[\varphi_1 = 0, \varphi_2 = 0 \ldots \varphi_N = 0]$. Therefore, the feature point $P$ can be represented by the position vector with respect to O, $\vec{r}^{OP}$, whose coordinates with respect to CS $\mathcal{U}$ are denoted as:
The self $\mathcal{S}$ is defined as the hyper-sub-space of all possible feature points at normal conditions. All other points in $U$ form the non-self $\mathcal{S}^\complement$. Therefore:

$$\mathcal{S} \cup \mathcal{S}^\complement = U \quad \text{and} \quad \mathcal{S} \cap \mathcal{S}^\complement = \emptyset$$  \hfill (1.15)

For computational tractability and practical reasons, the self points are clustered. These self clusters can be shaped as hyper-spheres, hyper-rectangles, or hyper-ellipsoids. Clustering of the non-self generates what is referred to as detectors. They may be directly used in the detection process. Structuring the non-self by adding information to the detectors makes them capable to be used in the failure identification process. Such upgraded detectors will be referred to as identifiers. Clusters of the self or non-self that are processed to be used for evaluation or accommodation will be referred to as evaluators and compensators, respectively.

The threshold between “normal” and “abnormal” (i.e. between self and non-self) is represented by an $N$-dimensional surface:

$$\Sigma(\varphi_1, \varphi_2, \ldots, \varphi_N) = 0$$  \hfill (1.16)

Lower dimensional projections of the self will be referred to as sub-selves. Let us assume that we consider a 2-dimensional projection (or sub-self) defined by features $\varphi_1$ and $\varphi_2$ and we collect all possible feature points at normal conditions. Area 0 will result as shown in Figure 1.6. Note that typically normalized values are used for the features; therefore, a working sub-region of the universe $\mathcal{U}$ is used next, which is represented as a hyper-cube of side 1. The projection of $\Sigma$ is represented as straight line segments for convenience, without loss of generality. All points outside Area 0 belong to the non-self $\mathcal{S}^\complement$. That means that whenever a pair of values $(\varphi_1, \varphi_2)$ is measured that lies outside Area 0, a failure can be declared. However, whenever a pair of values $(\varphi_1, \varphi_2)$ is measured that lies inside Area 0, it cannot be necessarily concluded that the conditions are normal. AC whose dynamic fingerprints depend on one or more other features can produce projections inside Area 0.

The order of an AC, denoted as $\text{Ord}(AC)$, is defined as the minimum number of features necessary to completely detect that AC. A complete detection is said to be possible if all feature points produced at AC fall within the non-self. For example, a first order failure would require only one feature $\varphi^*$ for detection. In other words, the presence of the failure necessarily produces always values of $\varphi^*$ that are outside a normal range. In Figure 1.6, if Area 1 is the projection of all points under a certain failure, then that failure is first order with $\varphi^* = \varphi_1$. Similarly, Area 2 represents an AC of degree 2, and Area 3 a failure of degree 3 or higher. A complete detection of the failure represented by Area 1 can be performed using the 2-dimensional self defined by $\varphi_1$ and $\varphi_2$, but also using only the projection, or the 1-dimensional sub-self defined by $\varphi_1$. The 2-dimensional self is necessary and sufficient to completely detect the failure represented by Area 2. Using the sub-self defined by $\varphi_1$ will never succeed detecting this failure, while using the sub-self defined by $\varphi_2$ will allow detection in some instances, but
not a complete detection. Using the 2-dimensional self to detect the third failure will not ensure a complete detection. However, it should be noted that, in practice, the approach could achieve very high rates of detection if the feature points that are projected inside the self are reached with very low probability.

Figure 1.6. Self/Non-Self 2-Dimensional Projection
1.4. Aircraft Abnormal Condition Management Process

The aircraft AC management (ACM) based on the AIS paradigm can be considered to include three main components functionally connected in a closed loop as shown in Figure 1.7:

- off-line ACM system design and implementation
- on-line AC detection, identification, evaluation, and accommodation
- post-processing of flight data and ACDIEA outcomes

The off-line ACM system design and implementation has as outcome the development of an integrated and comprehensive ACDIEA scheme. The design depends on a clear definition of the aircraft sub-systems that are targeted, the types of AC (including known and unknown failures), the failure severity scale, the flight envelope variables, and the nature and level of passive and active accommodation. The development of the ACDIEA scheme requires the availability of large amounts of measured data that must be pre-processed for self/non-self generation and structuring. For a comprehensive solution, acquiring and processing these data is considerably less difficult and expensive than developing extensive accurate models, as required by alternative approaches, still with questionable level of success. The block diagram of the ACM design component is presented in Figure 1.8.

The on-line ACDIEA process implies the real time operation of the ACDIEA scheme. Sets of current values of the features measured in flight at a certain sampling rate are compared against the detectors, identifiers, evaluators, and compensators and the outcomes of the ACDIEA are generated. These outcomes are transferred to the pilot, the on-board monitoring and recording system, and the automatic fault tolerant control laws. The block diagram of the on-line ACDIEA component is presented in Figure 1.9.

The post-processing of ACDIEA outcomes and the analysis of false alarms and failed detections in conjunction with current measured values of the features can potentially be used to modify/extend the sets of detectors, identifiers, evaluators, and compensators and improve the overall performance for future operation. The block diagram of the post-processing component is presented in Figure 1.10.
Definitions of:
- targeted aircraft sub-systems
- types of failure
- failure severity scale
- the flight envelope variables
- characteristics of passive accommodation
- characteristics of active accommodation

**Figure 1.8.** AIS-Based ACM System Design

**Figure 1.9.** On-Line AC Detection, Identification, Evaluation, and Accommodation
Figure 1.10. Post-Processing of Flight Data and ACDIEA Outcomes
1.5. Definition of the Self and Non-Self

1.5.1. General Algorithm

One of the most critical elements of the design of the ACM system is the selection of features. They must be relevant to all four processes (ACDIEA). Their number and nature depend on the aircraft sub-systems that are targeted, the types of AC, the failure severity scale, the flight envelope variables, and the nature and level of passive and active accommodation. For a comprehensive and integrated approach, all states, inputs, and variable parameters of all subsystems considered must be taken into account. It should also be noted that a complete detection of an $N$-degree failure requires $N$ relevant features, therefore, an $N$-dimensional self. If the number of subsystems considered is $Ns$, then the states $Xi$, inputs $Ui$, and other relevant parameters $Pi$ corresponding to sub-system $i \ i = 1,2,\ldots Ns$ can be expressed as, respectively:

$$Xi = [x_{i1} \ x_{i2} \ \cdots \ x_{INs}]^T$$  \hspace{1cm} (1.17)

$$Ui = [u_{i1} \ u_{i2} \ \cdots \ u_{INu}]^T$$  \hspace{1cm} (1.18)

$$Pi = [p_{i1} \ p_{i2} \ \cdots \ p_{INp}]^T$$  \hspace{1cm} (1.19)

For any sub-system $i$, the maximum order an AC can have is:

$$Ni = N_{si} + N_{ui} + N_{pi}$$  \hspace{1cm} (1.20)

Therefore, for a complete detection of all AC from an exhaustive set $F$, the number of necessary features to build the self/non-self is:

$$N = \sum_{i=1}^{Ns} (N_{si}^* + N_{ui}^* + N_{pi}^*)$$  \hspace{1cm} (1.21)

where the $^*$-variables exclude duplication among the sub-systems. If all states, inputs, and parameters are distinct, then:

$$N_{si}^* = N_{si}, \ \ N_{ui}^* = N_{ui}, \ \text{and} \ \ N_{pi}^* = N_{pi}$$  \hspace{1cm} (1.22)

otherwise, for example:

$$N_{si}^* \begin{cases} N_{si} & \text{for } i = 1 \\ N_{si} - n & \text{for } i > 1 \end{cases}$$  \hspace{1cm} (1.23)

where $n$ is the number of duplicate states for each sub-system, that is the states with the property that $x_{ij} \in X_i$ and $x_{ij} \not\in X_{i-1}$. It can be easily noticed that the number of necessary features $N$
and hence the dimensionality of the self/non-self can become excessively large and create significant computational problems\textsuperscript{9}. These problems can be avoided under certain conditions by using lower dimensional sub-selves, or projections, within a hierarchical multi-self (HMS) strategy\textsuperscript{10}.

Let the maximum order of failures in the set $F$ be denoted as:

$$N_{\text{max}} = \max_j \{\text{Ord}(f_j)\}, \quad j = 1, 2, \ldots, N_F$$

(1.24)

and

$$\text{Ord}(f_m) = N_{\text{max}}$$

(1.25)

If $N_{\text{max}} = N$, then a complete detection for $f_m$ can only be obtained, in general, with an $N$-dimensional self. If the self/non-self has particular shapes, then at least on one of the $(N-1)$-dimensional projections, the projection of the feature points at AC will fall inside the non-self. For example, in Figure 1.11, a 3-dimensional case is presented in which an AC feature point, AC1, is projected outside all three 2-dimensional projections. Feature point AC2 is projected outside one of the $N-1$ dimensional sub-selves. For specific shapes of the self/non-self, it is possible that some AC feature points are projected inside all sub-selves, as illustrated by feature point AC3.

![Figure 1.11. Potential of Sub-Selves to Capture the Abnormal Conditions](image)

If $N_{\text{max}} < N$, then a complete detection for $f_m$ can be obtained by using an $N_{\text{max}}$-dimensional sub-self with proper features. In this situation, the $N$-dimensional self does not need to be considered. A complete detection for all failures can be obtained by using all possible
$N_{\text{max}}$-dimensional projections or sub-selves. The approach is conservative and some of the projections may be ignored or replaced by lower dimensional projections without loss of performance.

Determining the order of an AC may be difficult in many cases or even impossible; therefore, $N_{\text{max}}$ may be unknown. In this situation, a low value for $N_{\text{max}}$ may be assumed and the self/non-self built as the set of all possible $N_{\text{max}}$-dimensional sub-selves, with the observation that the higher the assumed value of $N_{\text{max}}$, the higher the likelihood of better performance. The total number of possible $N_{\text{max}}$-dimensional sub-selves for $N$ features is given by:

$$N_{ss} = C_N^{N_{\text{max}}} = \frac{N!}{N_{\text{max}}!(N-N_{\text{max}})!}$$

The generation of the self/non-self requires large amounts of measured data at normal conditions, ideally covering the entire flight envelope. These $N$-dimensional feature points are clustered and the self $\mathcal{S}$ is represented as a set $S$ of hyper-bodies including these clusters, which will be referred to as self clusters. Similar hyper-bodies are used to represent the non-self $\mathcal{S}^\hat{}$ as well and will be referred to as detectors. The geometry of these hyper-bodies can potentially have an impact on the efficiency of the detector generation process and on the detection itself. They determine how well the non-self is covered, how many detectors are necessary, and how intensive the computational process is. The following shapes for the self/non-self representation can typically be considered:

- Hyper-cubes – determined by an $N$-dimensional center and one value for the side;
- Hyper-rectangles – determined by an $N$-dimensional center and $N$ values for the sides;
- Hyper-spheres – determined by an $N$-dimensional center and one value for the radius;
- Hyper-ellipsoid of rotation – determined by an $N$-dimensional center and two values for the axes;
- Generalized hyper-ellipsoid – determined by an $N$-dimensional center and $N$ values for the axes.

For all shapes (except hyper-spheres) variable orientation can be considered as determined by an additional $N$-dimensional vector. For example, for the hyper-spherical representation with $N_c$ clusters $c_i$, the self and the self clusters can be expressed as:

$$S = \{c_1, c_2, \ldots, c_{N_c}\}, \quad c_i = [C_i, R_{ci}] = [\varphi_{i1}, \varphi_{i2}, \ldots, \varphi_{Ni}, R_{ci}]$$

where $C_i$ is the center and $R_{ci}$ is the radius of the cluster. For the same hyper-spherical representation with $N_d$ detectors $d_j$, the non-self and the detectors can be expressed as:

$$\hat{S} = \{d_1, d_2, \ldots, d_{N_d}\}, \quad d_j = [D_j, R_{dj}] = [\varphi_{1j}, \varphi_{2j}, \ldots, \varphi_{Nj}, R_{dj}]$$

where $D_j$ is the center and $R_{dj}$ is the radius of the detector.

When generating the self and the non-self, the following optimization criteria should be considered:
• no overlapping among detectors and self clusters;
• minimum empty space inside the self clusters;
• minimum un-covered areas inside the non-self;
• minimum overlapping among self clusters;
• minimum overlapping among detectors;
• minimum number of detectors for a desirable resolution.

The block diagram of the self/non-self generation process is presented in Figure 1.12. A set of computational tools have been developed at WVU for the generation, optimization, and verification of detector sets within the AIS paradigm. The WVU Immunity-based Failure Detector Optimization and Testing tool relies primarily on evolutionary computation providing a wide selection of algorithms and options as well as capabilities for testing and tuning.

Figure 1.12. Self/Non-Self Generation Process

To support the AC identification and evaluation processes, the non-self must be structured to provide information regarding the sub-system affected, the type of the failure, and the failure severity. This can be achieved by attaching to each detector $d_j$ a set of structuring tags $dS_j$:

$$dS_j = \{dSys_j, dFt_j, dFs_j\}$$  \hspace{1cm} (1.29)

where, in general, $dSys_j \subset \{1, 2, \ldots, N_S\}$, $dFt_j \subset \{1, 2, \ldots, N_F\}$, and $dFs_j$ consists of a set of failure severity metrics associated to each failure type in $dFt_j$ and $size(dSys_j) = size(dFt_j) = size(dFs_j)$. The implication here is that each sub-system may be affected by only one failure at a time. The case in which a combination of failures occurs can be handled within this framework by defining in the set $F$ a distinct AC with such multiplicity characteristics. Alternatively, for each sub-system represented by $dSys_j$, the elements of $dFt_j$ and $dFs_j$ may be defined as sub-sets of size larger than 1. In this case, $size(dSys_j) < size(dFt_j) = size(dFs_j)$. The three sub-sets in Eq. (1.29) will each have only one element in the ideal situation when the $N$ features have been defined perfectly and
completely, one $N$-dimensional self is used, and the size of the detectors provides perfect resolution. In other words, any given detector can be associated to only one sub-system, one failure type, and one level of failure severity.

The process of generating the antibodies requires adequate numerical representations of the self/non-self and adequate data processing such that they are manageable given the computational resources and storage limitations of the hardware. In this project, 2-dimensional projections were used for self/non-self representation for which antibodies were generated using two alternative approaches that are described next. In the first approach, all raw test data available are collected in one file before a set of antibodies is generated. In the second approach, the processing of smaller individual sets of data is performed by clustering and then combining the clusters in a single set for detector generation. Using the two approaches, data for each combination of features corresponding to a particular projection or sub-self are processed separately to produce a set of antibodies by covering the respective 2-dimensional non-self. The block diagram of the AIS antibody generation methods is presented in Figure 1.13.

\[\text{Figure 1.13. Two Methods for AIS Antibodies Generation}\]

\textit{1.5.2. Raw Data Set Union Method}

The raw data set union method (RDSUM) processes experimental data at normal conditions in four main phases or modules as presented in Figure 1.13.

\textbf{Generation of Single Data File}. Raw data from different flights or simulator tests are combined in one single data file. The data are left intact and no further processing is performed here.

\textbf{Data Preprocessing}. Preprocessing of the data includes two steps: normalization and duplicate elimination. The raw data received from the data fusion component are normalized between 0 and 1. Therefore, the feature space becomes a unit hypercube. The normalization factor for each dimension is determined as the span of the flight data plus a percentage margin. Duplicate points of the normalized data are then eliminated to reduce the size of the data file.
This process decreases the amount of storage and computing resources needed, while preserving the information content of the data. Note that implicit duplicate point elimination may also take place during the clustering process, which follows.

**Self Data Clustering.** The data produced from the previous processing component define self points that need to be represented by a definite number of geometric hyper-bodies, referred to as clusters. This can be done by using the “$k$-means” clustering algorithm. A modified version of this algorithm\textsuperscript{17} was used to represent the clusters as hyper-spheres, but they could also be represented as hyper-rectangles or hyper-ellipsoids.

**Generation of Antibodies.** Self clusters are used to generate antibodies by covering the non-self hyper-space with hyper-bodies similar to the clusters. An enhanced negative selection algorithm for real-valued representation with variable non-self radius\textsuperscript{17} was used within this project. The algorithm ensures that there is no overlapping with the self and that the non-self is covered to a desired predetermined level. It should be noted that the algorithm requires a number of specific parameters that must be carefully selected and correlated. The antibodies generation process can be stopped after a prescribed number of iterations when a preset maximum number of acceptable detectors is reached or when a desired coverage of the non-self is achieved.

Figure 1.14 shows sample 2-dimensional self clusters along with the AIS antibodies generated for a projection or sub-self defined by the non-dimensional roll rate neural network output feature $NN_p$ and the reference roll rate $p_{ref}$ using the raw data set union method described above.

![Figure 1.14. 2-D Self Clusters with AIS Antibodies Generated Using the Raw Data Set Union Method](image)
1.5.3. Cluster Set Union Method

The mechanism to generate antibodies using the cluster set union method (CSUM) is based on a 5-phase/modules process that uses an optimized algorithm to fusion different sets of clusters generated from single sets of experimental flight or simulation data. The main components of this methodology, according to Figure 1.13, are described next.

**Preliminary Data Processing.** If the amount of experimental data is large enough to exceed the available computer memory, within this approach, the data can be split in sub-sets and the following steps in the process can be applied to the individual smaller data sets. Pre-processing of the data includes two steps: normalization and data preparation for clustering. As a result of the normalization, the values of each measured feature are scaled to values between 0 and 1 and as is the case with the previous approach, the feature space becomes a unit hypercube. The normalization factor for each dimension is determined as the span of the flight data plus a percentage margin. Alternatively, desired maximum and minimum values can be specified in the computation of the normalization factor. Note that when multiple sets of experimental data are used for antibodies generation, the same normalization factors must be used for all data.

**Clustering of Individual Data Sets.** This module is similar to the previous approach; however, the clustering algorithms are applied to the individual smaller sets of experimental data. Note that parallel computation may be used to perform this phase.

**Clusters Set Union.** Once several sets of clusters have been generated, a fusion process is performed that consists of set union accomplished by overlapping elimination.

**Clusters Duplicate Elimination.** The overlapping between clusters is estimated in a similar way as between the detectors, where a minimum overlapping threshold of a detector with respect to the others is allowed during the process. Since the radius of each cluster is known, the overlapping between a current cluster and the nearest one(s) can be determined. The distance between centers must be greater than or equal to the sum of the radii of the clusters minus the permitted overlapping threshold. This approach favors clusters with bigger radii and will preserve for the final self-representation those clusters with more efficient coverage. The approach allows the update of the database when new flight tests are available by clustering only the newly acquired data and then putting old and new clusters together and eliminating any duplication.

Figures 1.15 and 1.16 present an example of the clustering reduction using the fusion algorithm. The 2-dimensional self space in this case is defined by the non-dimensional roll rate neural network output feature $NN_{p}$ and the reference roll rate $p_{ref}$. Figure 1.15 shows the union of different sets of clusters generated from 11 single simulator test data before the fusion process is applied. The union set has a total of 22000 clusters. Figure 1.16 shows the reduced set of 6343 clusters after the complete fusion process. The fusion algorithm is able to reduce down the number of clusters to approximately one third with an acceptable overlapping among them. The allowed overlap value has to be selected in such a way that the balance between the holes and the covered space is adequate.

**Generation of Antibodies.** The same enhanced negative selection algorithm for real-valued representation with variable detector radius is applied to the cluster union set generated in the previous module. The algorithm ensures no overlapping between detectors and self clusters and minimizes the un-covered areas in the non-self.

In Figure 1.17, an example of the clusters and generated antibodies is presented for a 2-dimensional self projection defined by a non-dimensional roll rate neural network output feature $NN_{p}$ and the reference roll rate $p_{ref}$ using CSUM.
Figure 1.15. 2-D Self with 22000 Clusters

Figure 1.16. 2-D Self with 6343 Clusters

Figure 1.17. Sample 2-D Self Clusters with AIS Antibodies Generated Using the Cluster Set Union Method
An comparative analysis of the two proposed methods has been performed in terms of computational time and detection performance of the resulting sets of detectors. Numerical results and additional details are presented in journal paper #1 in Appendix A.

1.5.4. Comparison of the Proposed Methods for Antibody Generation

The computational time needed by RDSUM is consistently lower than the one needed by CSUM. The difference varies quite largely depending on the self. The fewer the number of distinct data is, the lower the computational time RDSUM needs for clustering. Typically, the raw data file reduces size significantly after the elimination of duplicate points. The clustering module is invoked only one time per self for RDSUM. The CSUM clusters the same number of flight data points 11 times (the number of flight tests used for this analysis) without any duplicate point removal. Only then, duplicate clusters are eliminated. For some of the selves generated using RDSUM, the number of unique data was around 60k (which is comparable to the number of records in each of the 11 flight files). It should be noted that the single file including all flight data used for detector generation with RDSUM has about 600,000 records. In most cases, this results in a still large data set even after duplicates are eliminated, which cannot be handled by the \(k\)-means clustering method on computers with less than 8GB RAM.

The detection performance in terms of percentage detection rate and false alarms of the two sets of detectors obtained with the different methods has been compared for a sub-set of relevant 2-dimensional projections or sub-selves under several types of failure. The performance is similar and the two methods can be considered equivalent from this point of view. In Figure 1.18, typical results in terms of detection rate are presented for a stabilator failure.

The RDSUM requires large computer memory, but the total computation time is lower. The CSUM can be implemented on lower memory computers; however, the overall computation time increases, unless parallel computation is used.

![Detection Performance Comparison of Detectors Obtained with CSUM and RDSUM for a Stabilator Failure](image)

**Figure 1.18.** Detection Performance Comparison of Detectors Obtained with CSUM and RDSUM for a Stabilator Failure
1.6. Hierarchical Multi-Self Strategy

For a comprehensive and integrated solution to the ACDIEA problem using the AIS paradigm, the number of system features for any system of aircraft complexity ends up to be very large. Handling the resulting hyper-space as a whole produces critical issues related to computational resources and effort and to the characteristics of hyper-spaces relative to distances and thresholds.

The issues related to computational resources and effort are obvious considering that in order to achieve similar high resolution of the feature space – a critical element especially for the identification and evaluation phases – exponentially larger computational effort is required with unitary increase of the space dimensionality.

The implications of the characteristics of hyper-spaces relative to distances and thresholds are more subtle. It should be noted that when the dimensionality of the hyper-space goes to infinity the volume of the unit hyper-cube (the “Universe”) remains equal to one, while the volume of the inscribed hyper-sphere goes to zero. This means that our intuition in establishing thresholds and assessing distances, which is built in lower dimensional spaces (actually in the 3-D physical space), becomes in-operational. This counter-intuitive effect becomes significant once the number of features goes beyond 10.

To eliminate or at least mitigate these effects, the HMS strategy was proposed based on the observation that, while the entire set of features is necessary to capture the dynamic fingerprint of “all” ACs, only limited subsets may be necessary to capture the dynamic fingerprint of any individual AC. Under certain conditions (as discussed in Section 1.5) and with proper ACDIE logic, subsets of features may be used to build projections of the self that eventually yield similar ACDIE performance as when considering the entire multi-dimensional self as a whole. This concept is illustrated in the block diagram of Figure 1.19.

Simulation results have shown that different low-dimensional self configurations can be selected and integrated to achieve low number of false alarms and high detection rates for a variety of subsystem abnormal conditions. In other words, the ACDIE performance achieved by a complete set of features of dimension \( N_s \) may be matched by a collection of sets \( i \), each of dimension \( n_i \), where \( n_i << N_s \) if they are properly integrated. As a result, the issues with the multi-dimensionality of the hyperspace can be mitigated or even eliminated. This approach involves determining the capabilities of different sets of features relative to the detection of specific types of ACs. Careful analysis is needed and the selves with best detection results must be identified, organized, and integrated within a hierarchical scheme such that a high detection rate, low number of false alarms, and correct identification are achieved for all ACs. The process is performed off-line and can be customized continually until a configuration is obtained, which ensures the desirable performance of the AIS scheme. The outcome of this process consists of a reduced-number set of sub-selves (projection) of specific size and feature content.

Previous research by the authors of this report has demonstrated the merits of the approach. Sets of lower dimensional selves were obtained through analysis and testing that yielded excellent ACDIE performance. However, the approach requires extensive information and/or investigation of ACFDIE capabilities of different features (equivalently, sub-selves or projections).

In this project, special focus was on developing a methodology that could avoid the need for significant prior knowledge or investigation of selected feature subsets (projections or sub-selves) capability for AC fingerprint capture. In other words, the number of projections, their dimension, and feature content is expected to be the result of an automated algorithm that does
not use prior knowledge. A first target of this investigation was to consider a complete set of lower-dimensional projections of uniform size. Considering Eq. (1.26), it can be seen that the number of projections may become very large. For example, for 32 features we obtain a complete set of 2-D projections of 496 elements. This number of projections becomes 4960 if 3-D projection are considered for the same number of features. Each projection would yield a detection outcome and all outcomes must be assessed and integrated properly to obtain a global detection outcome. This need led to the development of the dendritic cell mechanism, which will be discussed later.

**Figure 1.19.** The Hierarchical Multi-Self Strategy for ACDIE
1.7. Selection of Development and Demonstration Test Cases

1.7.1. Aircraft Subsystems

Four major aircraft subsystems with components (adding up to 19 individual subsystems) were considered for the purpose of this research effort:

- Actuators: left and right (L-R) stabilator (S), L-R aileron (A), L-R rudder (R), L-R throttle
- Sensors: pitch, roll, and yaw gyro
- Structure: L-R wing (W), L-R horizontal tail (H), L-R vertical tail (V)
- Propulsion: L-R engine

Total number of individual subsystems is:

\[
Ns = 8 + 3 + 6 + 2 = 19
\]  \hspace{1cm} (1.30)

1.7.2. Types of Subsystem Abnormal Conditions

For each subsystem, a list of targeted ACs must be established. Each list has \(N_{ft_k}, k = 1, 2, \ldots, N_s\), failures. The following ACs have been considered within this project:

- \(k = 1 - 8\), \(N_{ft_k} = 2\) (actuators)
  - locked at current position (trim)
  - moving and locking at a non-trim position

- \(k = 9 - 11\), \(N_{ft_k} = 2\) (sensors)
  - sensor bias
  - constant sensor output

- \(k = 12 - 17\), \(N_{ft_k} = 1\) (structure)
  - missing structure

- \(k = 18 - 19\), \(N_{ft_k} = 1\) (propulsion)
  - reduced effectiveness

The total number of failures is:

\[
N_F = \sum_{k=1}^{N_s} N_{ft_k} = 2 \times 8 + 2 \times 3 + 1 \times 6 + 1 \times 2 = 30
\]  \hspace{1cm} (1.31)

1.7.3. Flight Envelope Analysis Parameters

In Table 1.2 below, example lists of DIV, EDIV, and envelope relevant variables are presented. The resultant set of all features considered in Table 1.2 for evaluation purposes is:

\[
\mathcal{Y}_E = \{H, M, V, p, q, r, a_x, a_y, a_z, \dot{p}, \dot{q}, \dot{r}, \alpha, \beta, \theta, \phi\} = \bigcup_{i=1}^{N_F} v_{Ei} \quad i = 1, 2, \ldots, N_F
\]  \hspace{1cm} (1.32)

The total number of features considered for evaluation purposes in Table 1.2 (i.e. flight envelope reduction assessment) is \(N_E = 16\). It should be noted that subsets of these variables have been considered within the project for methodology development and demonstration.

Table 1.2. Parameters for AC Indirect Evaluation Analysis

<table>
<thead>
<tr>
<th>(k_F)</th>
<th>(k)</th>
<th>Type of Failure</th>
<th>DIV, (v_{\delta})</th>
<th>EDIV, (v_{\epsilon})</th>
<th>Envelope var., (v_E)</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>L-S locked</td>
<td>L-S defl</td>
<td>longitudinal stick</td>
<td>(a_x, a_z, H, M, q, p, \dot{q}, \alpha, \theta)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>L-S mv/lkd</td>
<td>L-S defl</td>
<td>longitudinal stick</td>
<td>(a_x, a_z, H, M, q, p, \dot{q}, \alpha, \theta)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>R-S locked</td>
<td>R-S defl</td>
<td>longitudinal stick</td>
<td>(a_x, a_z, H, M, q, p, \dot{q}, \alpha, \theta)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
<td>R S mv/lkd</td>
<td>R S defl</td>
<td>longitudinal stick</td>
<td>$a_x$, $a_z$, $H, M, q, p, \dot{q}, \alpha, \theta$</td>
</tr>
<tr>
<td>---</td>
<td>-----</td>
<td>-----</td>
<td>-----------</td>
<td>---------</td>
<td>------------------</td>
<td>------------------------------------------------</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
<td>L A locked</td>
<td>L A defl</td>
<td>lateral stick</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>L A mv/lkd</td>
<td>L A defl</td>
<td>lateral stick</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>4</td>
<td>R A locked</td>
<td>R A defl</td>
<td>lateral stick</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>R A mv/lkd</td>
<td>R A defl</td>
<td>lateral stick</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>5</td>
<td>L R locked</td>
<td>L R defl</td>
<td>pedals</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>5</td>
<td>L R mv/lkd</td>
<td>L R defl</td>
<td>pedals</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>6</td>
<td>R R locked</td>
<td>R R defl</td>
<td>pedals</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>6</td>
<td>R R mv/lkd</td>
<td>R R defl</td>
<td>pedals</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>7</td>
<td>L T locked</td>
<td>L T defl</td>
<td>pilot throttle</td>
<td>$a_x, a_z, r, H, M, V, \alpha, \theta$</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>7</td>
<td>L T mv/lkd</td>
<td>L T defl</td>
<td>pilot throttle</td>
<td>$a_x, a_z, r, H, M, V, \alpha, \theta$</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>8</td>
<td>R T locked</td>
<td>R T defl</td>
<td>pilot throttle</td>
<td>$a_x, a_z, r, H, M, V, \alpha, \theta$</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>8</td>
<td>R T mv/lkd</td>
<td>R T defl</td>
<td>pilot throttle</td>
<td>$a_x, a_z, r, H, M, V, \alpha, \theta$</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>9</td>
<td>p bias</td>
<td>p meas</td>
<td>lateral stick</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>9</td>
<td>p const</td>
<td>p meas</td>
<td>lateral stick</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>10</td>
<td>q bias</td>
<td>q meas</td>
<td>longitudinal stick</td>
<td>$a_x, a_z, H, M, q, \dot{q}, \alpha, \theta$</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>10</td>
<td>q const</td>
<td>q meas</td>
<td>longitudinal stick</td>
<td>$a_x, a_z, H, M, q, \dot{q}, \alpha, \theta$</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>11</td>
<td>r bias</td>
<td>r meas</td>
<td>pedals</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>11</td>
<td>r const</td>
<td>r meas</td>
<td>pedals</td>
<td>$a_y, p, r, \dot{p}, \dot{r}, \beta, \phi$</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>12</td>
<td>L W damage</td>
<td>L W lift</td>
<td>longitudinal stick, lateral stick, pedals, pilot throttle</td>
<td>$a_x, a_y, a_z, H, M, p, q, r, \dot{p}, \dot{q}, \dot{r}, \alpha, \beta, \theta, \phi$</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>13</td>
<td>R W damage</td>
<td>R W lift</td>
<td>longitudinal stick, lateral stick, pedals, pilot throttle</td>
<td>$a_x, a_y, a_z, H, M, p, q, r, \dot{p}, \dot{q}, \dot{r}, \alpha, \beta, \theta, \phi$</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>14</td>
<td>L H damage</td>
<td>L H lift</td>
<td>longitudinal stick, lateral stick</td>
<td>$a_x, a_y, a_z, H, M, p, q, r, \dot{p}, \dot{q}, \dot{r}, \alpha, \beta, \theta, \phi$</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>15</td>
<td>R H damage</td>
<td>R H lift</td>
<td>longitudinal stick, lateral stick</td>
<td>$a_x, a_y, a_z, H, M, p, q, r, \dot{p}, \dot{q}, \dot{r}, \alpha, \beta, \theta, \phi$</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>16</td>
<td>L V damage</td>
<td>L V lift</td>
<td>lateral stick, pedals, pilot throttle</td>
<td>$a_x, a_y, a_z, H, M, p, r, \dot{p}, \dot{r}, \beta$</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>17</td>
<td>R V damage</td>
<td>R V lift</td>
<td>lateral stick, pedals, pilot throttle</td>
<td>$a_x, a_y, a_z, H, M, p, r, \dot{p}, \dot{r}, \beta$</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>18</td>
<td>L E red eff</td>
<td>L E thrst</td>
<td>pilot throttle</td>
<td>$a_x, a_z, r, H, M, \alpha, \theta$</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>19</td>
<td>R E red eff</td>
<td>R E thrst</td>
<td>pilot throttle</td>
<td>$a_x, a_z, r, H, M, \alpha, \theta$</td>
<td></td>
</tr>
</tbody>
</table>
1.7.4. Features Selection

Based on the subsystems considered, the nature of ACs, and the extent of the flight envelope prediction attempted, a set of variables were selected to represent features for self/non-self definition, including states, states derivatives, inputs, control system variables, and artificial neural network (ANN) estimations. These variables are listed in Table 1.3.

Table 1.3. Features for Self/Non-self Definition

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H$</td>
<td>altitude</td>
</tr>
<tr>
<td>$V$</td>
<td>aircraft ground speed</td>
</tr>
<tr>
<td>$M$</td>
<td>Mach number</td>
</tr>
<tr>
<td>$a_x$</td>
<td>longitudinal acceleration</td>
</tr>
<tr>
<td>$a_y$</td>
<td>lateral acceleration</td>
</tr>
<tr>
<td>$a_z$</td>
<td>vertical acceleration</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>angle of attack</td>
</tr>
<tr>
<td>$\beta$</td>
<td>sideslip angle</td>
</tr>
<tr>
<td>$\phi$</td>
<td>roll attitude angle</td>
</tr>
<tr>
<td>$\theta$</td>
<td>pitch attitude angle</td>
</tr>
<tr>
<td>$\psi$</td>
<td>yaw attitude angle</td>
</tr>
<tr>
<td>$p$</td>
<td>roll rate</td>
</tr>
<tr>
<td>$q$</td>
<td>pitch rate</td>
</tr>
<tr>
<td>$r$</td>
<td>yaw rate</td>
</tr>
<tr>
<td>$\dot{p}$</td>
<td>roll acceleration</td>
</tr>
<tr>
<td>$\dot{q}$</td>
<td>pitch acceleration</td>
</tr>
<tr>
<td>$\dot{r}$</td>
<td>yaw acceleration</td>
</tr>
<tr>
<td>$d_e$</td>
<td>longitudinal stick displacement</td>
</tr>
<tr>
<td>$d_a$</td>
<td>lateral stick displacement</td>
</tr>
<tr>
<td>$d_r$</td>
<td>pedal displacement</td>
</tr>
<tr>
<td>$d_T$</td>
<td>pilot throttle</td>
</tr>
<tr>
<td>$p_{ref}$</td>
<td>roll rate command</td>
</tr>
<tr>
<td>$q_{ref}$</td>
<td>pitch rate command</td>
</tr>
<tr>
<td>$r_{ref}$</td>
<td>yaw rate command</td>
</tr>
<tr>
<td>$NN_{outp}$</td>
<td>roll acceleration error</td>
</tr>
<tr>
<td>$NN_{outq}$</td>
<td>pitch acceleration error</td>
</tr>
<tr>
<td>$NN_{outr}$</td>
<td>yaw acceleration error</td>
</tr>
<tr>
<td>$MQEE$</td>
<td>main quadratic estimation error</td>
</tr>
<tr>
<td>$OQEE$</td>
<td>output quadratic estimation error</td>
</tr>
<tr>
<td>$DQEE_p$</td>
<td>decentralized quadratic roll rate estimation error</td>
</tr>
<tr>
<td>$DQEE_q$</td>
<td>decentralized quadratic pitch rate estimation error</td>
</tr>
<tr>
<td>$DQEE_r$</td>
<td>decentralized quadratic yaw rate estimation error</td>
</tr>
</tbody>
</table>

Note that the parameters based on ANN estimates of angular rates are computed as follows: The main quadratic estimation error (MQEE) is defined as:

$$MQEE(k) = \frac{1}{2} \left[ (p(k) - \hat{p}_{MNN}(k))^2 + (q(k) - \hat{q}_{MNN}(k))^2 + (r(k) - \hat{r}_{MNN}(k))^2 \right]$$ (1.33)

where $p(k)$, $q(k)$, and $r(k)$ are measurements of angular rates at sample $k$ and $\hat{p}_{MNN}(k)$, $\hat{q}_{MNN}(k)$, and $\hat{r}_{MNN}(k)$ are neural estimates of the angular rates based on sensor measurements including the respective gyro, over a specified time window. The output quadratic estimation error (OQEE) is defined as:

$$OQEE(k) = \frac{1}{2} \left[ (\hat{p}_{DN}(k) - \hat{p}_{MNN}(k))^2 + (\hat{q}_{DN}(k) - \hat{q}_{MNN}(k))^2 + (\hat{r}_{DN}(k) - \hat{r}_{MNN}(k))^2 \right]$$ (1.34)
where $\hat{p}_{DNN}(k)$, $\hat{q}_{DNN}(k)$, and $\hat{r}_{DNN}(k)$ are neural estimates of the angular rates based on sensor measurements that do NOT include the respective gyro, over a specified time window. Finally, the decentralized quadratic angular rate estimation error (DQEE) is defined as:

$$DQEE_x(k) = \frac{1}{2} (\hat{x}_{DNN}(k) - x(k))^2, \quad x = p, q, r$$  (1.35)
1.8. WVU Simulation Environment

For the purpose of this project, experimental data were collected from the WVU 6-DOF flight simulator system shown in Figure 1.20. The simulator relies on a motion platform driven by electrical induction motors to provide adequate 6-DOF translational and rotational motion cues. Laminar research X-Plane flight simulation software is used to provide external visual cues through a six-monitor system. The motion platform is interfaced with an external computer on which an aircraft model can run within the Matlab/Simulink environment driving the entire simulator system.

![Figure 1.20. The WVU 6-DOF Motion-based Flight Simulator](image)

Figure 1.20. The WVU 6-DOF Motion-based Flight Simulator

Figure 1.21 shows the top level Simulink diagram of the model interfaced with the WVU Flight Simulator. The model includes the nonlinear dynamics of a supersonic fighter including models of failures/damages of actuator, sensors, structural components, and engines. The three large blocks at the bottom of the figure include the computation of specific variables to be provided to the flight simulator to drive the generation of visual and aural cues as well as the motion of the simulator platform.

To define the AIS self/non-self as completely and accurately as possible, adequate coverage of the feature space must be achieved. Different flight scenarios are considered over a wide range of the flight envelope, which is first defined based on the nine reference points shown in Figure 1.22 for Mach numbers between 0.6 and 0.9 and altitudes between 9,000 ft and 31,000 ft. All flight tests start at steady state flight condition at point 1 and continue to cover the nine points as described by the arrows. For example, one flight test starts at point 1, the aircraft is accelerated at constant altitude to point 4, descended at constant speed to point 5, and then returned to points 4 and 1. A total of eight such tests are necessary to cover the testing flight envelope. The data collected from flight tests under normal conditions with these points were used to build the self/non-self of the aircraft. Additional intermediate points (A, B, C, and D in Figure 1.22) were used to provide validation data. The set of flight scenarios, lasting between 10 and 20 minutes each, are designed to include steady-state flight conditions, transitions between steady-state conditions, and mild to moderate maneuvers. These flight scenarios are simulated under normal flight conditions. They are repeated under various failure scenarios for both design/development and validation purposes. Only one failure at a time is considered to
capture/isolate the dynamic fingerprint of each type of failure and generate antibodies appropriately. The data acquisition rate from the simulator is 50 Hz.

Figure 1.21. Simulink Supersonic Fighter Aircraft Model

Figure 1.22. Testing Flight Envelope
1.9. WVU Self/Non-self Analysis Tool

Visualizing the 2-dimensional sub-selves is very helpful in analyzing how well the self clusters and detectors are generated using one of the methods discussed in the previous sections. A simple interactive tool (see Figure 1.23) was developed to analyze all the generated 2-D subselves and project a given flight test data on these sub-selves in order to validate them before adopting them as a basis to the ACDIE schemes.

The visualization tool is user friendly and has the following main components referenced in Figure 1.23:
1. “Data folder” textbox to specify the fully-qualified path of the main folder containing all the flight tests and the self/nonself data generated.
2. Two list boxes of features containing the all features considered for self/non-self definition. Any feature selected from the left list box represents the abscissa of the 2-D plot and any feature selected from the right list box represents the ordinate.
3. “Flight test” dropdown menu containing all available flight tests.
4. Three check boxes providing options for plotting flight test data points on the figure, plotting self/nonself generated with the CSUM method (use ERAU data checkbox), and plotting post-failure flight envelope ranges.
5. Two optional textboxes to specify the starting and ending time steps used in plotting flight test data points. If these textboxes are empty, all data points in the selected flight test are to be plotted.
6. An optional drop down menu to quickly plot the selected 2-D projection.
7. An X-Y plot area onto which the self clusters, non-self shapes, and (optional) flight test data are plotted.

The zooming feature of the analysis tool facilitates the validation of the self/non-self generation process such as how well the detectors cover the non-self space, whether the method has generated self clusters within the specified margins or not, whether detectors overlapping is reasonable or not, whether there are issues with the self generation method (or its underlying processes) or with the flight data, etc. Figure 1.23 illustrates the zoom feature.

Figure 1.24. The Zooming Action of the Interactive Tool for Self/Non-self Analysis
Chapter 2

Immunity-based Detection of Aircraft Subsystem Abnormal Conditions

2.1. Detection through Direct Self/Non-self Discrimination

2.1.1. General Aspects

Let the self and non-self be defined as in Eqs. (1.27) and (1.28). Let the current sample \( t \) of the measured feature point be:

\[
\Phi_t = [\varphi_{1t}, \varphi_{2t}, \ldots, \varphi_{Nt}]
\]  

(2.1)

The AC detection can be performed through mechanisms that can be of two types. The NS-type of detection mechanism involves comparing the measured feature point against the detectors, while the PS-type of mechanism involves comparing the measured feature point against the self clusters. If one \( N \)-dimensional self/non-self is used, then the negative selection-type detection can be modeled as:

\[
Det_t = \begin{cases} 
0 & \text{if } |\vec{r}^O\Phi_t - \vec{r}^{OD}_j| > Rd_j \text{ for all } j = 1, 2, \ldots, Nd \\
1 & \text{otherwise}
\end{cases}
\]

(2.2)

while the PS-type of mechanism can be expressed as:

\[
Det_t = \begin{cases} 
1 & \text{if } |\vec{r}^O\Phi_t - \vec{r}^{OC}_i| > Rc_i \text{ for all } i = 1, 2, \ldots, Nc \\
0 & \text{otherwise}
\end{cases}
\]

(2.3)

If \( N_{ss} \) sub-selves are used, then Eq. (2.2) or (2.3) is applied for each sub-self to obtain sets of values \( Det_i, i = 1, 2, \ldots, N_{ss} \). Then:

\[
Det_t = \max_{i=1,N_{ss}} (Det_{ti})
\]

(2.4)

Note that a more complicated function than \( \max(*) \) can be considered such as one describing a dendritic cell mechanism.

To obtain a “normal condition” outcome with the negative selection approach, all detectors must be checked at each sample \( t \). With the positive selection approach, the testing of the current feature point is stopped as soon as a matching self cluster is found. Therefore, at normal conditions, the positive selection-based detection is less computationally intensive. On the other hand, an AC will be declared as soon as a matching detector has been found with the negative selection approach instead of testing the entire set of self clusters as is required by the positive selection approach. Therefore, an AC may be detected faster with the negative selection approach. Depending on the number of self clusters and detectors in conjunction with the system update rate, these differences may be relevant or not. However, note that building and
structuring the detectors is critical for the failure identification process if the structured non-self approach is used.

At each moment, a detection outcome can be simply obtained as:

$$Det = Det_i$$  \hspace{1cm} (2.5)

However, in order to prevent excessive false alarms, instead of using Eq. (2.5) directly, an additional detection logic must be used to build the detection phase outcome as a function of current and past outcomes and/or a composition of sub-selves-based detection outcomes.

Within the HMS strategy, sets of 2-dimensional and 3-dimensional projections were investigated for the detection through direct self/non-self discrimination.

### 2.1.2. Generation of Detectors

The process of generating detectors, or antibodies, is an exhaustive and lengthy procedure that requires adequate computational data processing capabilities and adequate numerical representations of the self/non-self. However, it should be noted that the process is performed off-line and does not affect the real time operation of the on-board detection scheme.

CSUM was used to process extensive experimental data necessary to properly define the nominal “hyper-space” representation. For each combination of features corresponding to a particular projection, segments of data are processed separately to produce sets of detectors that cover the non-self hyperspace. The CSUM is a 5 phase process (see Figure 2.1) that uses an optimized algorithm to fuse different sets of clusters generated from single sets of flight or simulation data. This methodology is particularly convenient when the computer characteristics used for pre-processing are not powerful enough for an efficient application of the algorithms.

![Figure 2.1. Cluster Set Union Based Logic](image)

54
### 2.1.3. Analysis of Detection Performance of Individual Selves

A total amount of 496 2-D projections of sub-selves were generated. Out of the 4960 possibilities for 3-D sub-selves, only 45 were generated. The performance of each projection or sub-self was analyzed and compared based on the detection rate (DR) and false alarm (FA) rate. The detection function output is a binary signal that determines if a sub-system failure has been detected (output=1) or if the aircraft is flying at nominal conditions (output=0). The binary output can be categorized in four types as follows:

- **True Positive (TP):** a failure is detected and declared as failure;
- **True Negative (TN):** nominal conditions are declared as nominal;
- **False Positive (FP):** nominal conditions are declared as failures;
- **False Negative (FN):** failure condition is not detected.

These categories can be used for a quantitative evaluation of the DR and FA. The following equations can be used to calculate the DR of a specified projection.

\[
DR = \frac{TP}{TP + FN} \times 100
\]

(2.2)

Similarly, the FA rate can be calculates as follows:

\[
FA = \frac{FP}{TN + FP} \times 100
\]

(2.3)

The detection rates and false alarms were calculated for 26 different ACs varying in affected subsystem, type, and severity. Table 2.1 presents a sample set of projections among the 496 generated sub-selves that were selected as an example of the relevant combination of features utilized in this effort.

### Table 2.1. Features of Selected Projections

<table>
<thead>
<tr>
<th>Self</th>
<th>Features</th>
<th>Self</th>
<th>Features</th>
<th>Self</th>
<th>Features</th>
<th>Self</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self#3</td>
<td>(p_{ref}, NN_p)</td>
<td>Self#94</td>
<td>(NN_p,r)</td>
<td>Self#85</td>
<td>(NN_p, QEE)</td>
<td>Self#105</td>
<td>(NN_p, d_T)</td>
</tr>
<tr>
<td>Self#30</td>
<td>(q_{ref}, NN_p)</td>
<td>Self#95</td>
<td>(NN_p, \psi)</td>
<td>Self#86</td>
<td>(NN_p, DQEE_p)</td>
<td>Self#106</td>
<td>(NN_p, M)</td>
</tr>
<tr>
<td>Self#42</td>
<td>(q_{ref}, q)</td>
<td>Self#96</td>
<td>(NN_p, \theta)</td>
<td>Self#87</td>
<td>(NN_p, DQEE_q)</td>
<td>Self#224</td>
<td>(DQEE_q, \psi)</td>
</tr>
<tr>
<td>Self#53</td>
<td>(q_{ref}, d_r)</td>
<td>Self#97</td>
<td>(NN_p, \varphi)</td>
<td>Self#89</td>
<td>(NN_p, V)</td>
<td>Self#233</td>
<td>(DQEE_q, d_r)</td>
</tr>
<tr>
<td>Self#82</td>
<td>(NN_p, NN_q)</td>
<td>Self#98</td>
<td>(NN_p, H)</td>
<td>Self#90</td>
<td>(NN_p, \alpha)</td>
<td>Self#410</td>
<td>(NN_p, \hat{p})</td>
</tr>
<tr>
<td>Self#83</td>
<td>(NN_p, NN_r)</td>
<td>Self#99</td>
<td>(NN_p, a_x)</td>
<td>Self#92</td>
<td>(N_p, p)</td>
<td>Self#441</td>
<td>(NN_p, \dot{q})</td>
</tr>
<tr>
<td>Self#84</td>
<td>(NN_p, MQEE)</td>
<td>Self#100</td>
<td>(NN_p, a_y)</td>
<td>Self#93</td>
<td>(NN_p, q)</td>
<td>Self#471</td>
<td>(NN_p, \dot{r})</td>
</tr>
</tbody>
</table>

Table 2.2 presents the performance analysis of those selves under four different failures: left aileron locked at 2.5 deg., left stabilator locked at 2 deg., 6% left wing loss, and 5 deg bias in the pitch rate sensor output. A sample set of 20 projections is presented. The detection rate as well as the false alarm rate is shown. These results demonstrate that certain selves favor the detection of certain types of failures while showing poor detection rates for others. This fact has been used in previous research efforts to support the formulation of the HMS strategy. The presented mechanism uses low order projections to build sub-selves using a specific hierarchy of features relevance with respect to each type of failure. Figures 2.2 through 2.5 present the individual detection rates obtained for different projections and four different failures.
Figure 2.6 presents the detection rate that a single projection can attain when tested against four different failures. It should be noted how a single projection is able to detect three different failures successfully while showing poor performance for a fourth one. This is due to the fact that the dynamics of such failure do not have great impact on the features corresponding to Self #87.

Table 2.2. Detection Performance of Selected Individual Selves

<table>
<thead>
<tr>
<th>Self</th>
<th>L Aileron Locked at 2 deg</th>
<th>L Stabilator Locked at 2 deg</th>
<th>6% Loss of Left Wing</th>
<th>Pitch Rate Sensor Bias (5 deg/sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>DR</td>
<td>FA</td>
<td>DR</td>
<td>FA</td>
</tr>
<tr>
<td>S3</td>
<td>82</td>
<td>1</td>
<td>99.3</td>
<td>1.85</td>
</tr>
<tr>
<td>S30</td>
<td>83.5</td>
<td>0.0</td>
<td>99.2</td>
<td>0.0</td>
</tr>
<tr>
<td>S42</td>
<td>0.0</td>
<td>0.0</td>
<td>8.22</td>
<td>0.0</td>
</tr>
<tr>
<td>S53</td>
<td>22.9</td>
<td>0.0</td>
<td>24.8</td>
<td>0.0</td>
</tr>
<tr>
<td>S82</td>
<td>92.3</td>
<td>0.0</td>
<td>99.7</td>
<td>0.0</td>
</tr>
<tr>
<td>S83</td>
<td>88.1</td>
<td>0.0</td>
<td>99.6</td>
<td>0.0</td>
</tr>
<tr>
<td>S84</td>
<td>86.2</td>
<td>0.0</td>
<td>99.5</td>
<td>0.0</td>
</tr>
<tr>
<td>S85</td>
<td>88.7</td>
<td>0.0</td>
<td>99.4</td>
<td>0.0</td>
</tr>
<tr>
<td>S86</td>
<td>84.2</td>
<td>0.0</td>
<td>99.4</td>
<td>0.0</td>
</tr>
<tr>
<td>S87</td>
<td>80.4</td>
<td>0.0</td>
<td>99.4</td>
<td>0.0</td>
</tr>
<tr>
<td>S89</td>
<td>90</td>
<td>0.0</td>
<td>99.6</td>
<td>0.0</td>
</tr>
<tr>
<td>S90</td>
<td>91.1</td>
<td>0.0</td>
<td>99.5</td>
<td>0.0</td>
</tr>
<tr>
<td>S92</td>
<td>80.8</td>
<td>0.0</td>
<td>99.3</td>
<td>0.0</td>
</tr>
<tr>
<td>S93</td>
<td>82.9</td>
<td>1.25</td>
<td>99.4</td>
<td>1.55</td>
</tr>
<tr>
<td>S94</td>
<td>85.8</td>
<td>2.25</td>
<td>99.5</td>
<td>2.25</td>
</tr>
<tr>
<td>S95</td>
<td>90.9</td>
<td>0.0</td>
<td>99.6</td>
<td>0.0</td>
</tr>
<tr>
<td>S96</td>
<td>89.5</td>
<td>0.0</td>
<td>99.5</td>
<td>0.0</td>
</tr>
<tr>
<td>S97</td>
<td>86.3</td>
<td>0.0</td>
<td>99.5</td>
<td>0.0</td>
</tr>
<tr>
<td>S98</td>
<td>94.6</td>
<td>0.0</td>
<td>99.6</td>
<td>0.0</td>
</tr>
<tr>
<td>S99</td>
<td>86</td>
<td>0.0</td>
<td>99.4</td>
<td>0.0</td>
</tr>
<tr>
<td>S100</td>
<td>86.9</td>
<td>0.0</td>
<td>99</td>
<td>0.0</td>
</tr>
<tr>
<td>S105</td>
<td>91.1</td>
<td>0.0</td>
<td>99.6</td>
<td>0.0</td>
</tr>
<tr>
<td>S106</td>
<td>93.1</td>
<td>0.0</td>
<td>99.5</td>
<td>0.0</td>
</tr>
<tr>
<td>S224</td>
<td>0.36</td>
<td>0.0</td>
<td>1.85</td>
<td>0.0</td>
</tr>
<tr>
<td>S233</td>
<td>5.51</td>
<td>0.0</td>
<td>8.04</td>
<td>0.0</td>
</tr>
<tr>
<td>S410</td>
<td>81.4</td>
<td>0.0</td>
<td>99.4</td>
<td>0.0</td>
</tr>
<tr>
<td>S441</td>
<td>80.9</td>
<td>0.0</td>
<td>99.4</td>
<td>0.0</td>
</tr>
<tr>
<td>S471</td>
<td>80.5</td>
<td>0.0</td>
<td>99.3</td>
<td>0.0</td>
</tr>
</tbody>
</table>
Figure 2.2. Detection Rate of Left Aileron Locked at 2.5 deg Failure

Figure 2.3. Detection Rate of Left Stabilator Locked at 2.0 deg Failure

Figure 2.4. Detection Rate of 6% Left Wing Loss Failure
2.1.4. Detection Logic for Direct Self/Non-self Discrimination Scheme

Based on the sensitivity analysis that establishes the capability of each projection to capture the dynamic fingerprint of the ACs considered, a sub-set of projections of size $n$ is selected for detection. A weight is associated to each projection depending on their detection potential. During the detection phase, sets of current values of the features measured in flight at a certain sampling rate are compared against the detectors that have been generated for each of the $n$ 2D-projections selected for detection. A detection parameter $\zeta$ is calculated, which represents the number of consecutive points over a time window of size $\omega$ that trigger detectors, summed over all sub-selves and affected by the weights. If $\zeta$ is within a certain range (less than a threshold $Thr_1$), a failure warning is issued, but if $\zeta$ exceeds the upper bound of the range (larger than a
threshold Thr2), a failure is declared and the identification phase starts. The flow chart of the SNSA detection logic is presented in Figure 2.7.

![Flow chart of SNSA detection logic]

**Figure 2.7.** On-Line Failure Detection Logic with SNSA

### 2.1.5. Detection Performance Using Self/Non-self Discrimination

The set of aircraft features from Table 1.3 were used to build the AIS self and non-self. Several ACs were tested on the WVU 6-DOF motion-based flight simulator, as listed in Table 2.3. Different flight scenarios were considered over a wide range of the flight envelope for Mach numbers between 0.6 and 0.9 and altitudes between 9,000 ft and 31,000 ft. Each flight test lasted between 15 and 20 minutes. The data acquisition rate was 50 Hz. The two performance metrics used are the percentage detection rate and the false alarm rate, as defined in section 2.1.3.

**Table 2.3.** Aircraft Subsystems Considered for AC Detection

<table>
<thead>
<tr>
<th>Category</th>
<th>k</th>
<th>Subsystem</th>
<th>Tests Performed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actuators</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Left Stabilator</td>
<td>Control surface locked at 2° and 8°</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>Right Stabilator</td>
<td>Control surface locked at 2° and 8°</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>Left Aileron</td>
<td>Control surface locked at 2.5° and 8°</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>Right Aileron</td>
<td>Control surface locked at 2.5° and 8°</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>Left Rudder</td>
<td>Control surface locked at 8°</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>Right Rudder</td>
<td>Control surface locked at 8°</td>
</tr>
<tr>
<td>Structure</td>
<td></td>
<td>Left Wing</td>
<td>6% and 15% loss of the wing</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>Right Wing</td>
<td>6% and 15% loss of the wing</td>
</tr>
<tr>
<td>Sensors</td>
<td></td>
<td>Roll Gyro Sensor</td>
<td>5 deg/sec and 10 deg/sec Fast Drifting Bias</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>Pitch Gyro Sensor</td>
<td>5deg/sec and 10deg/sec Fast Drifting Bias</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>Yaw Gyro Sensor</td>
<td>1deg/sec and 3deg/sec Fast Drifting Bias</td>
</tr>
<tr>
<td>Engine</td>
<td></td>
<td>Left Engine</td>
<td>100% efficiency reduction</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>Right Engine</td>
<td>100% efficiency reduction</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 2.4 presents the average detection performance for the SNSA-based failure detection schemes. As it can be seen from the results, the proposed methods exhibit very good performance with high detection rates and practically zero false alarms for the variety of ACs considered.

<table>
<thead>
<tr>
<th>$k$</th>
<th>Subsystem Affected by AC</th>
<th>Detection Rate [%]</th>
<th>False Alarms [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>LeftStabilator</td>
<td>99.78</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td>RightStabilator</td>
<td>100</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>LeftAileron</td>
<td>97.5</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>RightAileron</td>
<td>97.48</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>LeftRudder</td>
<td>65.86</td>
<td>0.00</td>
</tr>
<tr>
<td>6</td>
<td>RightRudder</td>
<td>75.04</td>
<td>0.00</td>
</tr>
<tr>
<td>7</td>
<td>Left Wing</td>
<td>100</td>
<td>0.00</td>
</tr>
<tr>
<td>10</td>
<td>Right Wing</td>
<td>100</td>
<td>0.00</td>
</tr>
<tr>
<td>15</td>
<td>RollRateSensor</td>
<td>99.98</td>
<td>0.00</td>
</tr>
<tr>
<td>16</td>
<td>PitchRateSensor</td>
<td>99.98</td>
<td>0.00</td>
</tr>
<tr>
<td>17</td>
<td>YawRateSensor</td>
<td>97.19</td>
<td>0.00</td>
</tr>
<tr>
<td>18</td>
<td>LeftEngine</td>
<td>94.77</td>
<td>0.00</td>
</tr>
<tr>
<td>19</td>
<td>Right Engine</td>
<td>72.54</td>
<td>0.00</td>
</tr>
</tbody>
</table>
2.2. Dendritic Cell Mechanism for Abnormal Condition Detection

2.2.1. The Artificial Dendritic Cell

The DC mechanism-based approach for failure detection is inspired by the functionality of the biological DCs in the tissue and their interaction with the adaptive immune system. A detection outcome is produced based on the combined discrimination outcomes from all 2-D projections (possibly augmented with the outcomes from higher dimensional projections).

Let the time sample be denoted as $t$. Define the detection matrix $D_t$ as:

$$
D_t = \begin{bmatrix}
S_1 & S_2 & \ldots & S_N \\
Det_{11} & Det_{12} & \ldots & Det_{1N_{SS}} \\
Det_{21} & Det_{22} & \ldots & Det_{2N_{SS}} \\
\vdots & \vdots & \ddots & \vdots \\
Det_{w1} & Det_{w2} & \ldots & Det_{wN_{SS}} 
\end{bmatrix}
$$

where $Det_{ij} = \begin{cases} 
0 & \text{if feature point is inside } S_j, \\
1 & \text{if feature point is outside } S_j 
\end{cases}$, $i = 1,2,\ldots,w$  $j = 1,2,\ldots,N_{SS}$ (2.4)

A moving time window of size $w$ is assumed for detection. Each column of matrix $D_t$ is a set of detection outcomes of a particular sub-self over the entire time window. Ideally, all elements of this matrix are equal to zero under normal conditions and to 1 under abnormal conditions. In practice, $D_t$ usually contains both zeros and ones whether the time window is under normal or abnormal conditions due to imperfections in designing and building the sub-selves and/or due to the shape of the self that may project abnormal condition feature points into the self area. This situation may lead to false alarms and missed detections if not properly handled.

Let the complement of matrix $D_t$ be:

$$
\overline{D}_t = \{Det_{ij} \mid Det_{ij} = \text{non}(Det_{ij})\}
$$

(2.5)

Cells from a pool of DCs are randomly selected to process the input matrices $D_t$ and $\overline{D}_t$. For each DC the “triggered features matrix” $F_1$ and the “non-triggered features matrix” $F_0$ are built.

$$
F_1 = \begin{bmatrix}
S_1 & S_2 & \ldots & S_{N_{SS}} \\
F_{11} & F_{12} & \ldots & F_{1N_{SS}} \\
F_{21} & F_{22} & \ldots & F_{2N_{SS}} \\
\vdots & \vdots & \ddots & \vdots \\
F_{N1} & F_{N2} & \ldots & F_{NN_{SS}} 
\end{bmatrix}
\phi_1
$$

(2.6)

where $F_{qj} = F_{qj} + 1$ if $\phi_q$ belongs to the triggered sub-self $S_j$. 

\[
F_0 = \begin{bmatrix}
S_1 & S_2 & \cdots & S_{N_{SS}} \\
F_{11} & F_{12} & \cdots & F_{1N_{SS}} \\
F_{21} & F_{22} & \cdots & F_{2N_{SS}} \\
\vdots & \vdots & \ddots & \vdots \\
F_{N1} & F_{N2} & \cdots & F_{NN_{SS}}
\end{bmatrix} \begin{bmatrix}
\varphi_1 \\
\varphi_2 \\
\vdots \\
\varphi_N
\end{bmatrix}
\]  

(2.7)

where \( F_{qj} = F_{qj} + 1 \) if \( \varphi_q \) belongs to the not-triggered sub-self \( S_j \).

For each sub-self \( S_j \), a “confidence” factor \((w_{0i} \in [0, 1], \ w_{1i} \in [0, 1])\) for accurately capturing the self and non-self is defined, which is used to define the following weighting vectors:

\[
W_0 = \begin{bmatrix} w_{01}, & w_{02}, & \ldots, & w_{0N_{SS}} \end{bmatrix}^T
\]

(2.8)

represents the level of confidence that an output of “0” is trustworthy.

\[
W_1 = \begin{bmatrix} w_{11}, & w_{12}, & \ldots, & w_{1N_{SS}} \end{bmatrix}^T
\]

(2.9)

represents the level of confidence that an output of “1” is trustworthy.

When a DC processes an input, the concentration of the secreted proteins, known as “co-stimulatory molecules,” denoted by \( CSM \), on the surface of the cell increases, regardless of whether the input is self or non-self (i.e., antigen). It is important to note that \( T \)-cells cannot be activated unless both the antigen and the \( CSM \) coexist. If the input is an antigen, the cell produces a special cytokine called “interleukin-12,” denoted by \( IL_{12} \). Otherwise, it produces another type of cytokine called “interleukin-10,” denoted by \( IL_{10} \). Thus, the presence of \( IL_{12} \) is an indication of abnormal condition and that of \( IL_{10} \) is a normal one. Any DC cannot migrate to the lymph node until its \( CSM \) reaches a predefined threshold, \( M \), which is randomly assigned to each cell \( C_i \in \mathbb{C} \) during initialization. Like any other cells, some DCs experience “apoptosis,” the healthy programmed cell death. This is expressed by assigning a random life, \( L \), to each artificial cell \( C_i \in \mathbb{C} \) during initialization.

Some cells migrate to the lymph node (the adaptive immune system) with an indication of abnormal conditions and stimulate the adaptive immune system to generate cytotoxic T-cells while other cells migrate with an indication of normal conditions and stimulate the adaptive immune system to generate suppressor T-cells to regulate the generated cytotoxic T-cells. This stimulation/suppression of the cytotoxic T-cells determines the resultant response of the adaptive immune system which indicates whether the system is under normal or abnormal conditions.

At each time sample, the selected DCs update their internal parameters based on detection outcome from all projections according to the following relationships:

\[
IL_{10_{t}} = IL_{10_{t-1}} + I_{N_{SS}} \cdot \bar{D}_{t} \cdot W_0
\]

(2.10)

where \( I_{N_{SS}} = [l, \ l, \ \ldots, \ l] \), \( \text{size}(I_{N_{SS}}) = N_{SS} \)
\[ IL12_t = IL12_{t-1} + I_{NS} \cdot D_t \cdot W_t \] (2.11)

\[ CSM_t = CSM_{t-1} + 1 \] (2.12)

\[ L_t = L_{t-1} - 1 \] (2.13)

If \( L = 0 \) a cell is considered dead and it is replaced by a new cell. This can be done by reinitializing the dead cell with the default properties. Any cell with \( CSM \geq M \) should migrate from the tissue to the lymph node to present the processed input and must be replaced by a new cell with default properties. If these two conditions are not met, the cell continues processing the next element of \( D_t \) and \( D_t \), if the cell belongs to the randomly selected subset. Any migrated DC with \( IL12 \geq IL10 \) is called “stimulatory” since it activates cytotoxic T-cells; otherwise it is called “regulatory” since it activates suppressor (regulatory) T-cells.

Let the mature DC be denoted by \( DC_{k} \), \( k = 1, 2, \ldots, N_{DC} \), where \( N_{DC} \) is the total number of mature DCs. Let the set of activated cytotoxic T-cells be \( K = \{ K_q \mid q = 1, 2, \ldots, N \} \), where \( K_q \) is the number of activated cytotoxic T-cells corresponding to feature \( \varphi_q \), then:

\[ K_q = \sum_{m=1}^{N_{DC}} \sum_{j=1}^{N_{SS}} F_{1mq} \text{ for all } m \text{ for which } IL12 \geq IL10 \] (2.14)

Let the set of activated suppressor T-cells be:

\[ R = \{ R_q \mid q = 1, 2, \ldots, N \} \] (2.15)

where \( R_q \) is the number of activated suppressor T-cells corresponding to feature \( \varphi_q \), then:

\[ R_q = \sum_{m=1}^{N_{DC}} \sum_{j=1}^{N_{SS}} F_{0mq} \text{ for all } m \text{ for which } IL12 < IL10 \] (2.16)

The role of the suppressor T-cells is to regulate the adaptive immune response by suppressing a corresponding number of activated cytotoxic T-cells. This results in a set of residual cytotoxic T-cells given by:

\[ \tilde{K} = \{ \tilde{K}_q = K_q - R_q \mid q = 1, 2, \ldots, N \} \] (2.17)

The detection outcome at sample time \( t \) is finally defined as:

\[ Det_t = \begin{cases} 0 & \text{if } \sum_{q=1}^{N} \tilde{K}_q \leq 0 \\ 1 & \text{otherwise} \end{cases} \] (2.18)
The block diagram of the proposed algorithm is illustrated in Figure 2.8. A set $\mathcal{C}$ of $N_{DC}$ dendritic cells is initialized with each cell $C_\ell \in \mathcal{C}$ ($\ell = 1, 2, \ldots, N_{DC}$) having the data structure shown in Figure 2.9.

![Figure 2.8. Block Diagram of the DC Algorithm for Failure Detection](image)

**Figure 2.8.** Block Diagram of the DC Algorithm for Failure Detection

![Figure 2.9. Data Structure of an Artificial Dendritic Cell](image)

**Figure 2.9.** Data Structure of an Artificial Dendritic Cell
2.2.2. Detection Performance Using the DC Mechanism

The artificial DC mechanism for detection was implemented in C# for computational speed considerations. The code was compiled with Microsoft® .NET™ Framework version 4.5 and integrated with the WVU simulation environment for the fighter aircraft. Validation tests, along with the information of the generated sub-selves, were used as input data to the DC algorithm. The algorithm was initialized with a pool of 100 DCs and a moving time window of 50 time steps (i.e., 1 s). The sum of the residual cytotoxic T-cells $\sum_{j=1}^{N} \tilde{k}_j$ in Eq. 2.18 versus time is shown in Figures 2.10 - 2.13 for different tests. These figures show that the sum of the residual cytotoxic T-cells produced by the DC mechanism is negative and remains in a narrow band for normal conditions. When an AC occurs, this parameter experiences an abrupt increase and remains positive, capturing the occurrence of the AC and providing a reliable detection criterion.

Figure 2.10. Sum of Residual Cytotoxic T-cells vs. Time of a Nominal Flight Test

Figure 2.11. Sum of Residual Cytotoxic T-cells vs. Time of an AC Flight Test: Right Aileron Locked at 8°. Actual Failure Time = 40 s; Detection Time = 40.76 s

Figure 2.12. Sum of Residual Cytotoxic T-cells vs. Time of an AC Flight Test: 3° LFDB in Yaw Rate Sensor. Actual Failure Time = 40 s; Detection Time = 43.38 s

Figure 2.13. Sum of Residual Cytotoxic T-cells vs. Time of an AC Flight Test: 15% Loss of the Left Wing. Actual Failure Time = 40 s; Detection Time = 40.52 s
Table 2.5 summarizes the detection rate for different subsystems. The percentage detection rate is computed based on the ratio between the number of samples detected as AC and total number of samples under AC. Note that these rates are the averages of detection rates of the corresponding number of different tests with different failure magnitudes and/or flight paths in the third column of the table. Also note that each flight test lasts for about 20 min. on average and that all failures were introduced at 40 s. Failures can be introduced at any time during the flight test without affecting the performance of the algorithm. Samples during the first 40 s were used to determine the false alarm rate as the ratio between the number of samples detected as failure and the total number of samples under normal conditions. In all tests, the false alarm rate was zero. Without specific tuning of the parameters, the average detection time of the algorithm for the failures considered was 2 s, with a standard deviation of 2.8 s.

<table>
<thead>
<tr>
<th>k</th>
<th>Subsystem</th>
<th>Detection Rate,%</th>
<th>No. Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Left Stabulator</td>
<td>99.93</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>Right Stabulator</td>
<td>99.93</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>Left Aileron</td>
<td>99.46</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>Right Aileron</td>
<td>98.94</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>Left Rudder</td>
<td>93.09</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>Right Rudder</td>
<td>80.47</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>Left Throttle</td>
<td>54.76</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>Right Throttle</td>
<td>54.72</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>Left Wing</td>
<td>99.93</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>Right Wing</td>
<td>99.94</td>
<td>4</td>
</tr>
<tr>
<td>11</td>
<td>Left Horizontal Tail</td>
<td>99.82</td>
<td>2</td>
</tr>
<tr>
<td>12</td>
<td>Right Horizontal Tail</td>
<td>99.91</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>Left Vertical Tail</td>
<td>24.95</td>
<td>3</td>
</tr>
<tr>
<td>14</td>
<td>Right Vertical Tail</td>
<td>16.66</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>Roll Rate Sensor</td>
<td>96.46</td>
<td>6</td>
</tr>
<tr>
<td>16</td>
<td>Pitch Rate Sensor</td>
<td>99.86</td>
<td>5</td>
</tr>
<tr>
<td>17</td>
<td>Yaw Rate Sensor</td>
<td>99.78</td>
<td>5</td>
</tr>
<tr>
<td>18</td>
<td>Left Engine</td>
<td>59.72</td>
<td>4</td>
</tr>
<tr>
<td>19</td>
<td>Right Engine</td>
<td>59.22</td>
<td>4</td>
</tr>
</tbody>
</table>
Chapter 3

Immunity-based Identification of Aircraft Subsystem Abnormal Conditions

3.1. Structured Non-self Approach for Abnormal Condition Identification

3.1.1. Principles of AC Identification Using the Structured Non-self Approach

Structuring the non-self for AC identification and converting the detectors into identifiers by attaching the $dSys_j$ tags involves the identification of detector sub-sets that correspond to AC that affect specific sub-systems. This implies the prior generation of the non-self with adequate resolution and knowledge of AC characteristics. Such information can be obtained from tests, simulation, or analysis. The concept is illustrated in Figure 3.1 for the 2-dimensional case with the assumption that the sub-sets $dSys_j$ have only one element each.

![Figure 3.1. Failed Subsystem Identification Using the Structured Non-self Approach](image)

If $Det_t = 1$, then the triggering detector is checked for structural parameters and a current identification outcome is defined using Eq. (1.2), for example, as:

$$Idt_t = [id_1 \quad id_2 \quad \ldots \quad id_{Ns}], \quad \begin{cases} id_i = 0 & \text{for } i \notin dSys_j \\ id_i = 1 & \text{for } i \in dSys_j \end{cases}$$  (3.1)

67
If Eq. (1.3) is used then the current identification outcome is defined as:

$$Idt_t = \left[ id_1 \quad id_2 \quad \cdots \quad id_{Nsf} \right], \quad id_j = i \quad \text{for} \quad i \in dSys_j$$

(3.2)

If one $N$-dimensional non-self is used with perfect feature definition, then the identification process outcome can be defined as:

$$Idt = Idt_t$$

(3.3)

If the HMS strategy is used, then Eq. (3.1) or (3.2) is applied to each sub-self and a composition logic $\lor$ must be designed to obtain the current identification outcome as:

$$Idt_t = \lor (Idt_{ti}), \quad i = 1,2,\ldots,N_{ss}$$

(3.4)

then Eq. (3.3) can be applied. However, as was the case with the detection process, using Eq. (3.3) directly may produce a large number of incorrect identifications. This can be mitigated by reprocessing the current outcomes $Idt_t$ over a moving time window. The composition logic $\lor$ can also rely on the DC mechanism.

### 3.1.2. Identification Algorithm Using the Structured Non-self Approach

The identification algorithm is only activated when the detection algorithm declares that an abnormal condition is present. The identification phase assumes that the output of the detection phase is correct.

In this research effort a novel structured non-self approach (SNSA) has been developed within a HMS strategy. This approach is based on a structuring process, or arrangement, of non-self projections and intends to reduce the computational effort required and facilitate the real-time application of the AIS approach without compromising the ACFDIE performance. The SNSA consists of a dual-phase algorithm where 2-dimensional self/non-self projections, previously generated using a negative selection mechanism and tested in simulation under several abnormal conditions, are selected according to the ability to detect failures at a predefined detection rate percentage. Then, by using a positive selection-type mechanism, the resulting projections are processed in order to generate identifiers capable of differentiating similar dynamic prints among several abnormal conditions and declaring correct failure types, and magnitudes. For example, within a first phase of the SNSA, a total of 496 self/non-self projections were generated based on the availability of 32 features to capture the dynamic print of abnormal conditions. After extensive experimentation it was possible to reduce the number of self/non-self projections to 183 possible candidates with a DR equal or larger than 70%. These projections are considered to possess the ability to capture the dynamic print of several subsystem failures and, more importantly, facilitate the process of characterizing the projections that perform better during the identification of specific failures. Table 3.1 presents a sample set of 2-dimensional projections investigated within the identification phase.

It is important to note that the dynamic fingerprint of several failures may produce a very similar effect on the features of self/non-self projections. This characteristic presents a more complex problem in which incorrect identification may be produced if the identification problem is not defined appropriately. For example, let us assume that an identification algorithm, only
consisting of $Self#3 (p_{ref}, NN_p)$, is tested for two sub-system failures (i.e. right wing structural failure and left aileron stuck failure). This particular pair of failures will produce an undesired roll rate that can be successfully perceived and detected by $Self#3$. The dynamic fingerprint produced by both abnormal conditions in the selected projection may look very similar, increasing the complexity of the identification problem. Now, let us assume that the same identification algorithm is augmented with $Self#30 (q_{ref}, NN_q)$, which can also capture the abnormal condition dynamic print of the mentioned failures. Due to the fact that $Self#30$ also captures dynamic changes in pitch rate, it is possible to identify and distinguish between the two mentioned failures. Within a second phase of the SNSA, positive selection applied to the 183 self/non-self projections is performed in order to address the mentioned identification problem. For illustration, Figures 3.2 and 3.3 present the similarity of the dynamic print of two different failures in a 2-dimensional projection.

<table>
<thead>
<tr>
<th>Self</th>
<th>Features</th>
<th>Self</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Self#3$</td>
<td>$p_{ref}, NN_p$</td>
<td>$Self#56$</td>
<td>$r_{ref}, NN_p$</td>
</tr>
<tr>
<td>$Self#4$</td>
<td>$p_{ref}, NN_q$</td>
<td>$Self#57$</td>
<td>$r_{ref}, NN_q$</td>
</tr>
<tr>
<td>$Self#7$</td>
<td>$p_{ref}, OQEE$</td>
<td>$Self#60$</td>
<td>$r_{ref}, OQEE$</td>
</tr>
<tr>
<td>$Self#8$</td>
<td>$p_{ref}, DQEE_p$</td>
<td>$Self#61$</td>
<td>$r_{ref}, DQEE_p$</td>
</tr>
<tr>
<td>$Self#9$</td>
<td>$p_{ref}, DQEE_q$</td>
<td>$Self#62$</td>
<td>$r_{ref}, DQEE_q$</td>
</tr>
<tr>
<td>$Self#30$</td>
<td>$q_{ref}, NN_p$</td>
<td>$Self#69$</td>
<td>$r_{ref}, r$</td>
</tr>
<tr>
<td>$Self#31$</td>
<td>$q_{ref}, NN_q$</td>
<td>$Self#82$</td>
<td>$NN_p, NN_q$</td>
</tr>
<tr>
<td>$Self#34$</td>
<td>$q_{ref}, NN_p$</td>
<td>$Self#83$</td>
<td>$NN_p, NN_r$</td>
</tr>
<tr>
<td>$Self#35$</td>
<td>$q_{ref}, DQEE_p$</td>
<td>$Self#84$</td>
<td>$NN_p, MQEE$</td>
</tr>
<tr>
<td>$Self#36$</td>
<td>$q_{ref}, DQEE_q$</td>
<td>$Self#85$</td>
<td>$NN_p, OQEE$</td>
</tr>
<tr>
<td>$Self#42$</td>
<td>$p_{ref}, NN_p$</td>
<td>$Self#86$</td>
<td>$NN_p, DQEE_p$</td>
</tr>
<tr>
<td>$Self#52$</td>
<td>$q_{ref}, d_e$</td>
<td>$Self#87$</td>
<td>$NN_p, DQEE_q$</td>
</tr>
<tr>
<td>$Self#53$</td>
<td>$q_{ref}, d_r$</td>
<td>$Self#88$</td>
<td>$NN_p, DQEE_r$</td>
</tr>
</tbody>
</table>

The combined identification capabilities of the projections utilized within the two phases of the SNSA (see Figure 3.4) provides a more robust system capable of not only correctly identifying the detected failure but also providing information regarding the magnitude of the investigated failures. With the correct combination of projections and their corresponding identifiers, it is possible to discard incorrect identifications and ultimately determine which abnormal condition is affecting the system.
Figure 3.2. (a) Self#3 with Left Aileron Failure; (b) Self#3 with Right Wing Structural Damage

Figure 3.3. (a) Self#30 with L Aileron Failure; (b) Self#30 with R Wing Structural Damage
3.1.3. Generation of Identifiers

The process of generating identifiers is very similar to the detector generation. However, the generation of identifiers is based on a positive selection-type of algorithm, in which failure flight test data are used to expose the dynamic fingerprint of a failure throughout the entire flight envelope. The generation of identifiers consists of a multi-step process that optimizes the set. The radii of the identifiers depend mostly on their distance to the self. The flow chart of the identifier generation is presented in Figure 3.5. The main components of the process are described next.

**Abnormal Flight Tests.** Flight tests at different abnormal conditions throughout the entire flight envelope must be performed. Previously selected features corresponding to the self/non-self definition as shown in Table 3.1 must be recorded for future processing and identifier definition.

**Normalization.** The sets of raw data received from the flight tests are recorded and their values are normalized between 0 and 1. The normalization factor of each projection is determined by the range of the flight data plus a percent margin. The normalization factor is the same used for the self/non-self projections during the antibodies generation. Therefore, the normalized data points of failure data correspond to the correct hypercube projection of each specific feature combination.
Offset Hypercubes. The unit hypercube determined during the normalization process delimits the hyperspace of the nominal conditions. High magnitude failures may contain data points that lay far away from the unit hypercube of the self/non-self projection. Therefore, outward concentric hypercubes are defined in order to determine the distance of the abnormal condition point from the self (see Figure 3.6), which subsequently allows the algorithm to determine the magnitude of the corresponding failure as part of the direct quantitative evaluation. Note that these phases defined, in general, subsequently may be performed simultaneously.

Radii Assignment. The radius of any identifier is predetermined and it is assigned depending on the location of its center with respect to the offset hypercubes. The radius of an identifier increases as the position of its center lies within an outward hybercube. In other words, the radii of all identifiers increase as their distance to the self increases as illustrated in Figure 3.6.
Identifiers Elimination/Fusion. The amount of initial identifiers depends on the number of data points obtained from the flight tests. This may yield an enormous number of identifiers which will produce a degradation of the computer processing capability. A simple elimination algorithm is implemented in order to reduce the number of identifiers. Identifiers that lay inside the radius of another identifier plus a tolerance are eliminated. Finally, a fusion process is performed. The fusion process consists of a set union accompanied with overlapping elimination. After this step is concluded, the final number of identifiers is reduced considerably.

3.1.4. The Two Phases of the SNSA Approach

The first phase of the SNSA is the result of the failure detection testing within the HMS strategy and consists in the selection of lower dimensional projections. Only 2-D projection have been considered in the phase of the project. As mentioned previously, 496 2-dimensional self/non-self projections were generated for failure detection algorithm experimentation. These projections were then tested against over 20 different failures. Different magnitudes/severities of the ACs were also considered. Extensive experimentation was required in order to determine which projections could substantially detect a failure with good detection rates and minimum false alarms within a NS approach. It was determined that a total of 183 projections were able to fulfill the objectives of a DR equal or higher than 70%. This process is referred to as the Phase I Non-Self Structuring. The selected projections were chosen as potential candidates for identification included sensor outputs, state estimates, and statistical parameters among other features. The set of abnormal conditions involved sensor failures, structural damage on the wings, engine failures, and control surface failures. Table 3.2 presents a list of the failures investigated in this research effort.

<table>
<thead>
<tr>
<th>Failure #</th>
<th>Failure Type</th>
<th>Failure #</th>
<th>Failure Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Left Aileron 2deg</td>
<td>9</td>
<td>Left Wing Loss 6%</td>
</tr>
<tr>
<td>2</td>
<td>Right Aileron 2deg</td>
<td>10</td>
<td>Right Wing Loss 6%</td>
</tr>
<tr>
<td>3</td>
<td>Left Aileron 8deg</td>
<td>11</td>
<td>Left Wing Loss 15%</td>
</tr>
<tr>
<td>4</td>
<td>Right Aileron 8deg</td>
<td>12</td>
<td>Right Wing Loss 15%</td>
</tr>
<tr>
<td>5</td>
<td>Left Stabilator 2deg</td>
<td>13</td>
<td>Left Engine Out</td>
</tr>
<tr>
<td>6</td>
<td>Right Stabilator 2deg</td>
<td>14</td>
<td>Right Engine Out</td>
</tr>
<tr>
<td>7</td>
<td>Left Stabilator 8deg</td>
<td>15</td>
<td>Roll Sensor Bias 5deg/sec</td>
</tr>
<tr>
<td>8</td>
<td>Right Stabilator 8deg</td>
<td>16</td>
<td>Roll Sensor Bias 10deg/sec</td>
</tr>
</tbody>
</table>

Several failures presented similar dynamic prints on several 2D-projections, which subsequently led to the repetition of several projections with the ability to detect multiple failures. On the other hand, certain failures that are difficult to detect, such as rudder failure, only resulted in the activation of a small number of projections. The negative selection logic behind the Phase I Structuring resulted in the reduction of the number of the original projections into a smaller set, reducing the complexity and the hardware requirements for its implementation. Table 3.3 presents a sample of the projections that are considered to be adequate for abnormal condition identification based on a detection performance equal or higher than 70%. The detection capability of this sample set of projections is presented for five types of failures. Various projections present the ability to capture the dynamic fingerprint of several
abnormal conditions while others can only capture the dynamics of a small set or just a single abnormal condition. For example, Self#3 demonstrated its ability to capture the dynamic fingerprint of a left aileron locked, a right wing structural damage and a left stabilator locked failure. On the other hand, Self#4 only demonstrated the ability to capture the dynamic fingerprint of a left stabilator locked type of failure.

Table 3.3. Detection Sensitivity of a Sample Set of Projections (%DR)

<table>
<thead>
<tr>
<th>Failure</th>
<th>Left Aileron stuck at 2.5 deg</th>
<th>Right Wing 6% Structural Damage</th>
<th>Left Stabilator Stuck at 8 deg.</th>
<th>Right Engine Out</th>
<th>Pitch Sensor 10deg/sec Bias</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self# 3</td>
<td>82.02</td>
<td>99.84</td>
<td>99.96</td>
<td>10.51</td>
<td>3.42</td>
</tr>
<tr>
<td>Self# 4</td>
<td>1.45</td>
<td>3.87</td>
<td>99.85</td>
<td>1.42</td>
<td>4.08</td>
</tr>
<tr>
<td>Self# 30</td>
<td>83.49</td>
<td>99.83</td>
<td>99.96</td>
<td>10.57</td>
<td>30.02</td>
</tr>
<tr>
<td>Self# 31</td>
<td>0.85</td>
<td>1.94</td>
<td>99.82</td>
<td>0.52</td>
<td>60.68</td>
</tr>
<tr>
<td>Self# 52</td>
<td>0.99</td>
<td>0.76</td>
<td>1.56</td>
<td>1.10</td>
<td>71.52</td>
</tr>
<tr>
<td>Self# 56</td>
<td>86.85</td>
<td>99.94</td>
<td>99.88</td>
<td>12.94</td>
<td>0.59</td>
</tr>
<tr>
<td>Self# 82</td>
<td>92.33</td>
<td>99.96</td>
<td>99.97</td>
<td>21.32</td>
<td>15.01</td>
</tr>
<tr>
<td>Self# 83</td>
<td>88.06</td>
<td>99.93</td>
<td>99.97</td>
<td>14.13</td>
<td>0.74</td>
</tr>
<tr>
<td>Self# 84</td>
<td>86.23</td>
<td>99.94</td>
<td>99.98</td>
<td>12.05</td>
<td>0.30</td>
</tr>
<tr>
<td>Self# 85</td>
<td>88.76</td>
<td>99.91</td>
<td>99.96</td>
<td>12.80</td>
<td>37.42</td>
</tr>
<tr>
<td>Self# 100</td>
<td>86.92</td>
<td>99.45</td>
<td>99.25</td>
<td>15.20</td>
<td>0.46</td>
</tr>
<tr>
<td>Self# 142</td>
<td>0.06</td>
<td>29.35</td>
<td>56.08</td>
<td>72.42</td>
<td>0.99</td>
</tr>
<tr>
<td>Self# 233</td>
<td>5.51</td>
<td>7.47</td>
<td>7.02</td>
<td>5.49</td>
<td>92.07</td>
</tr>
<tr>
<td>Self# 259</td>
<td>13.44</td>
<td>54.05</td>
<td>77.33</td>
<td>72.54</td>
<td>9.41</td>
</tr>
<tr>
<td>Self# 350</td>
<td>15.39</td>
<td>33.10</td>
<td>60.70</td>
<td>72.47</td>
<td>22.27</td>
</tr>
<tr>
<td>Self# 351</td>
<td>26.07</td>
<td>50.30</td>
<td>67.13</td>
<td>71.49</td>
<td>14.23</td>
</tr>
<tr>
<td>Self# 433</td>
<td>1.39</td>
<td>1.44</td>
<td>6.76</td>
<td>2.33</td>
<td>77.7</td>
</tr>
</tbody>
</table>

Within this analysis it was possible to isolate the projections that can be used for identification purposes. From the Phase I analysis, it was possible to determine which specific projections correspond to every specific failure investigated. Furthermore, it is also possible to determine how many projections capture the dynamic fingerprint of an abnormal condition. Table 3.4 below presents the number of projections from the 183 total initially selected that have the potential to be used for successful identification purposes.

The outcome of the Phase I self/non-self structuring reduces the total amount of projections needed to perform ACFDIE. Its outcomes also allow for adequate design of the mapped-based positive selection algorithm utilized in the second phase of the SNSA by reducing the number of possible projections for the generation of identifiers as well as the reduction of identifiers required in the identification algorithm for each individual projection.

Phase II of SNSA includes a positive selection process where flight failure test data are used to generate higher resolution non-self detectors called identifiers. Resulting projections from Phase I are processed in order to generate identifiers capable of differentiating similar dynamic fingerprints among several abnormal conditions and declaring correct failure types, and
magnitudes. In order to obtain correct identification results, the identification logic must be carefully formulated and the generation and selection of identifiers must be adequate. Subsets of antibodies or identifiers must be generated with sufficient resolution to avoid incorrect outputs.

Table 3.4. Total Number of Projections Activated per Failure

<table>
<thead>
<tr>
<th>Actuator</th>
<th>Engine</th>
<th>Structural</th>
</tr>
</thead>
<tbody>
<tr>
<td>Aileron Stuck (8 deg)</td>
<td>Stablator Stuck (8 deg)</td>
<td>Rudder Stuck (8 deg)</td>
</tr>
<tr>
<td>L</td>
<td>R</td>
<td>L</td>
</tr>
<tr>
<td>31</td>
<td>31</td>
<td>72</td>
</tr>
</tbody>
</table>

The identifiers generated during Phase I and II are loaded into an identification function and organized in a single array such that the index of each identifier corresponds to a failure type and magnitude. The arrangement of the identifiers is inspired by a mapping-based algorithm, which simplifies the selection scheme. The positive selection process is performed in parallel by all the projections included in the identification algorithm. Each projection outputs a single index that corresponds to a type and magnitude of failure. The outputs of all projections are compared among each other and the most frequent value is determined. If a specific failure index is constant throughout the majority of the projections’ outputs, its value is selected and a proper identification is declared.

3.1.5. AC Identification Performance Using Self/Non-self Discrimination

Since the SNSA covers not only general identification logic, but also a qualitative and quantitative evaluation logic integrated into a single less complex algorithm, the identification results are presented in Section 4.1 integrated with classification results of magnitude and category of failure. This novel approach intends to reduce the computational processing for real time application of the solution to the ACDIE problem. The proposed mapping-based positive selection logic targets a multi-dimensional problem by means of a simpler but effective logic that can result in a more efficient real time algorithm.
3.2. Dendritic Cell Mechanism for Abnormal Condition Identification

3.2.1. Dendritic Cell Mechanism for AC Identification

The DC mechanism-based approach for AC identification infers the subsystem affected based on the topography of the detection outcomes from self/non-self projections, within the HMS strategy. Therefore, with the DC approach, the AC identification is formulated as a pattern recognition problem. The AC identification consists of determining which one of the \( N_S \) subsystems has been affected by the AC and it starts once an AC has been detected. Patterns for AC identification purposes may be defined based on the \( F_l \) matrix. Note that the \( F_0 \) matrix may also be introduced into the process. \( N_S \) different patterns must be established, one associated to each subsystem. The process of establishing the library of pattern is presented in Figure 3.7. There are three possible alternatives for defining the patterns:

- Feature-pattern (F-P) approach
- Projection-pattern (P-P) approach
- Matrix-pattern (M-P) approach

A). Feature-pattern (F-P) approach for AC identification

For each AC, define the vector of membership values of each feature to the set of equivalent directly involved variables:

\[
FP_{ACj} = [m_1 \ m_2 \ \ldots \ m_N]^T, \ m_i \in [0, \ 1] \tag{3.5}
\]

For each subsystem \( k \), define the \( FP \) vector as:

\[
FP_k = \frac{1}{N_{fk}} \sum_{j=1}^{N_{fk}} FP_{ACj} = [\overline{m}_1 \ \overline{m}_2 \ \ldots \ \overline{m}_N]^T \tag{3.6}
\]

where \( \overline{m}_i \in [0, \ 1], \ k = 1, 2, \ldots, N_S \tag{3.7} \)

All these membership values can be determined based on tests and heuristics. Note that binary logic can be used instead of fuzzy logic. In this case: \( m_i \in \{0, \ 1\} \) and \( \overline{m}_i \in \{0, \ 1\} \).

At each sample time, after an AC is detected, use the \( F_l \) matrices of the mature DCs to compute the current AC FP as:

\[
F_{l\phi} = \left[ \sum_{q=1}^{N_{DC}} F_{l\phi} \cdot I_{N_{SS}} \right] \cdot I_{N_{SS}} \in \{I\}^{N_{SS} \times I} \tag{3.8}
\]

Use a matching algorithm to determine which one of the \( N_S \) FP vectors best matches \( F_{l\phi} \).

B). Projection-pattern (P-P) approach for AC identification

For each AC, define the vector of membership values of each projection to the set of AC-relevant projections (projections that can capture the dynamic fingerprint of the AC):
For each sub-system $k$, define the $P$-$P$ vector as:

$$PP_k = \frac{1}{\text{max}(\sum_{j=1}^{N_{jk}} PP_{ACj})} \sum_{j=1}^{N_{jk}} PP_{ACj} = [\overline{n}_1 \quad \overline{n}_2 \quad \ldots \quad \overline{n}_{N_{SS}}]^T$$

where $\overline{n}_i \in [0, 1], \quad k = 1, 2, \ldots, N_S$ \hspace{1cm} (3.11)

All these membership values can be determined based on tests and heuristics. Note that binary logic can be used here as well.

At each sample time, after an AC is detected, use the $F_1$ matrices of the mature DCs to compute the current AC PP as:

$$F_{1P} = I_N \cdot \left[ \sum_{q=1}^{N_{DC}} F_{1q} \right], \quad I_N \in \{1\}^{1 \times N_{SS}}$$

Use a matching algorithm to determine which one of the $N_S$ PP vectors best matches $F_{1P}$.

### C. Matrix-pattern (M-P) Approach for AC Identification

For each sub-system $k$, define the FP vector $FP_k$ and the PP vector $PP_k$ as described above. Use them to build the MP matrix $MP_k$.

$$MP_k = \begin{bmatrix}
    m_{11}^* & m_{12}^* & \ldots & m_{1N_{SS}}^* \\
    m_{21}^* & m_{22}^* & \ldots & m_{2N_{SS}}^* \\
    \vdots & \vdots & \ddots & \vdots \\
    m_{N1}^* & m_{N2}^* & \ldots & m_{NN_{SS}}^*
\end{bmatrix}, \quad m_{ij}^* \in [0, 1], \quad i = 1, 2, \ldots, N, \quad j = 1, 2, \ldots, N_{SS}$$

where $m_{ij}^* = \begin{cases} 
\text{max}(\overline{m}_i, \overline{n}_j) & \text{if } \varphi_i \text{ belongs to } S_j \\
0 & \text{otherwise}
\end{cases}$ \hspace{1cm} (3.14)

At each sample time, after the AC is detected, compute the sum of all mature DC $F_1$ matrices as:

$$F_{1M} = \sum_{q=1}^{N_{DC}} F_{1q}, \quad q = 1, 2, \ldots, N_{DC}$$

Finally, use a matching algorithm to determine which one of the $N_S$ M-P matrices best matches $F_{1M}$.
3.2.2. Performance of AC Identification Using a Dendritic Cell Mechanism

Among the three pattern approaches, the F-P approach was selected since it requires less computational resources as compared to the P-P and M-P approaches. The artificial DC mechanism for identification with the F-P approach was implemented in C#. The naïve Bayes classifier was trained with the current F-P vectors $F_{\varphi}$ from a set of training failure tests for each subsystem $k$ to implicitly define the reference features-pattern $FP_k$ of that subsystem in terms of the corresponding mean vector and covariance matrix. Figure 3.8 illustrates the variation of the current F-P vectors $F_{\varphi}$ over the entire test time of a failed right wing subsystem ($k = 13$) in one of these tests. Figures 3.9 through 3.11 depict the reference $FP_k$ vectors of the right wing ($k = 13$), left aileron ($k = 3$), and the yaw rate sensor ($k = 11$), respectively. Once a failure is declared by the DC mechanism, the current $F_{\varphi}$ vector is passed to the naïve Bayes classifier to select the closest $FP_k$ to the current $F_{\varphi}$ vector, and thus identify the failed subsystem. A sample current $F_{\varphi}$ is shown in Figure 3.12 for a validation failure test with right wing damage.

Listed in Table 3.5 are the average identification rates for different subsystems. These results show the capability of the proposed AC identification scheme in isolating the failed subsystem with very high identification rates. Note that the first column in Table 3.4 represents the subsystem affected by the failure and the first row represents the subsystem identified as the failed one. The off-diagonal numbers in the table represent the incorrect identification (misclassification) rates. For example, 99.13% of the samples under failure of the right stabilator were correctly identified, whereas for 0.81% of these samples, the failed subsystem was incorrectly identified as the left stabilator; for 0.05% of these samples, the failed subsystem was incorrectly identified as the left rudder; and for 0.01% of these samples, the failed subsystem was
incorrectly identified as the pitch rate sensor. The small variations in performance between subsystems may be typically attributed to imperfect coverage of the self, imperfect detector generation, errors introduced by the projections, and variations of the classification algorithm used in conjunction with the DC mechanism.

**Figure 3.8.** Variation of the Feature-Pattern Vector with Time of a Failed Right Wing Subsystem

**Figure 3.9.** Reference Features-Pattern of a Failed Right Wing

**Figure 3.10.** Reference Features-Pattern of a Failed Left Aileron
**Figure 3.11.** Reference Features-Pattern of a Failed Yaw Rate Sensor

**Figure 3.12.** Sample Current Feature-Pattern of a Failed Right Wing

**Table 3.5.** Average Identification Rates Using the DCMechanism

<table>
<thead>
<tr>
<th>Sub-system</th>
<th>Left Stab.</th>
<th>Right Stab.</th>
<th>Left Aileron</th>
<th>Right Aileron</th>
<th>Left Rudder</th>
<th>Right Rudder</th>
<th>Left Wing</th>
<th>Right Wing</th>
<th>Roll Rate</th>
<th>Pitch Rate</th>
<th>Yaw Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left Stab.</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Right Stab.</td>
<td>0.81</td>
<td>99.13</td>
<td>0.00</td>
<td>0.05</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>0.00</td>
</tr>
<tr>
<td>Left Aileron</td>
<td>0.00</td>
<td>0.00</td>
<td>99.78</td>
<td>0.14</td>
<td>0.01</td>
<td>0.04</td>
<td>0.00</td>
<td>0.03</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Right Aileron</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>99.89</td>
<td>0.08</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>0.00</td>
<td>0.02</td>
<td></td>
</tr>
<tr>
<td>Left Rudder</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>99.95</td>
<td>0.05</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Right Rudder</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.04</td>
<td>99.96</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Left Wing</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>99.95</td>
<td>0.05</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Right Wing</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.02</td>
<td>99.98</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Roll Rate</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>0.00</td>
<td>0.00</td>
<td>99.99</td>
<td>0.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Pitch Rate</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>100.00</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>Yaw Rate</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.01</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>99.99</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 4

Immunity-based Direct Evaluation of Aircraft Subsystem Abnormal Conditions

AC direct evaluation consists of two parts: qualitative direct evaluation and quantitative direct evaluation. The qualitative evaluation refers to determining the nature or the type of the AC, while the quantitative direct evaluation refers to determining the severity of the AC or the actual values of parameters that are a direct consequence of the AC. The AC direct evaluation within the AIS paradigm can be approached in a similar way as the identification phase.

4.1. Structured Non-self Approach for Abnormal Condition Direct Evaluation

The Structured Non-self approach can be used not only to perform general AC identification, but also a qualitative and quantitative AC evaluation within an integrated logical framework, such that the overall algorithm is less complex. This novel approach intends to reduce the computational processing for real time application of the solution to the ACFDIE problem. The proposed mapping-based positive selection-type of approach targets a multi-dimensional problem by means of a simpler but effective logic that can result in a more efficient real time algorithm. Within this method, the determination of the type and severity of an AC, also known as direct evaluation, is performed simultaneously with the identification stage as a single process, leaving the indirect evaluation as an individual stage in the SNSA. This architecture is presented in Figure 4.1.

4.1.1. Structured Non-self Approach for Qualitative Direct Evaluation

The qualitative evaluation can be performed in a manner similar to the identification by structuring the non-self and attaching to each detector $d_j$ the tags $dFt_j$. This process involves the identification of detector sub-subsets within each subset corresponding to one single subsystem. As was the case with the identification process, prior generation of the non-self with adequate resolution and knowledge of AC characteristics is required. This concept is illustrated in Figure 4.2 for the 2-dimensional case with the assumption that the sub-sets $dFt_j$ have only one element each.

If $Det_t = 1$, then the triggering detector is checked for structural parameters and a current qualitative evaluation outcome for the case when only one sub-system is affected can be defined as:

$$ EvI_t = \begin{bmatrix} evI_1 & evI_2 & \cdots & evI_{Nft} \end{bmatrix}, \text{with } Nft = Nft_{dSys_j} \text{ and } evI_i = 0 \text{ for } i \not\in Ft_j \text{ and } evI_i = 1 \text{ for } i \in Ft_j $$ (4.1)

Note that $EvI_t$ can be formulated to consist of non-zero elements only, case in which:

$$ EvI_t = dFt_j $$ (4.2)
If several subsystems may be affected (that is \(\text{size}(dSys_j) > 1\)), then \(\text{Evl}_i\) from Eq. (4.1) becomes a matrix with \(\text{size}(dSys_j)\) rows and \(\max_k(Nft_k)\) columns. Note that some rows of this matrix must be completed with zeros to account for the different values of \(Nft_k\).

Figure 4.1. Integration of Direct Evaluation and Identification Within SNSA

If one \(N\)-dimensional non-self is used with perfect feature definition, then the qualitative evaluation outcome can be defined as:

\[
\text{Evl} = \text{Evl}_i
\]

(4.3)

If the HMS strategy is used, then Eq. (4.1) or (4.2) is applied to each sub-self and a composition logic \(\overline{\cdot}\) must be designed to obtain the current qualitative evaluation outcome as:

\[
\text{Evl}_i = \overline{\text{Evl}_{ii}}, \quad i = 1,2,...,N_{ss}
\]

(4.4)
then Eq. (4.3) can be applied. However, the current qualitative evaluation outcomes $Ev_l$ can be further processed over a moving time window to improve performance. As was the case with the identification process, structuring of the non-self can be avoided within the HMS strategy if an adequate composition logic $\mathcal{C}$ can be formulated.

![Figure 4.2. Failure Qualitative Evaluation Using the Structured Non-Self Approach](image)

**4.1.2. Structured Non-self Approach for Quantitative Direct Evaluation**

The direct quantitative evaluation relies on correct qualitative evaluation and must be approached in different ways depending on the specific definition of the process outcome. If a definition of $Ev_2$ as illustrated by Eq. (1.8) is used, then the $dFs_j$ tags in Eq. (1.29) must represent the values of some ad-hoc defined metric that can capture the severity of the failure. For example, for many types of failure, the minimum distance from the center of the detector to the nearest self cluster can be used as illustrated in Figure 4.3. If a definition illustrated by Eq. (1.9) or (1.10) is used instead, then the $dFs_j$ tags represent quantitative attributes (for Eq. (1.9)) or membership function values associated to a set of quantitative attributes (for Eq. (1.10)), as illustrated in Figure 4.4. In any case, the quantitative evaluation outcome $Ev_2$ must correspond to $Ev_l$, that is for each scalar element of $Ev_l$ there will be a scalar element of $Ev_2$ if Eq. (1.8) or (1.9) are used, or a vector element if Eq. (1.10) is used. Therefore, at each sampling time $t$: 

$$
0 1
$$

$\text{Self}$

$\text{AC Type #1}$

$\text{AC Type #2}$

$\text{AC Type #3}$

$\text{AC of Sub-system #2}$
\[ Ev_{2t} = dFs_j \quad \text{(4.5)} \]

**Figure 4.3.** Failure Direct Quantitative Evaluation Based on Distance to Self

If one \( N \)-dimensional non-self is used with perfect feature definition, then the overall direct quantitative evaluation outcome can be defined as:

\[ Ev_2 = Ev_{2t} \quad \text{(4.6)} \]

If the HMS strategy is used, then Eq. (4.6) is applied to each sub-self and a composition logic \( \sim \) must be designed to obtain the current qualitative evaluation outcome as:

\[ Ev_{2t} = \sim (Ev_{2i}), \quad i = 1, 2, \ldots, N_{ss} \quad \text{(4.7)} \]

then Eq. (4.6) can be applied. As in the previous phases, the current outcomes \( Ev_{2t} \) can be further processed over a moving time window to improve performance.

It should be noted that associating type and/or severities of ACs to regions of the non-self, that is to specific detectors (they become thus “evaluators”) requires tests at abnormal conditions of different types and severities and identification of the corresponding areas in the non-self (detectors/identifiers). The approach may prove resource prohibitive for a comprehensive solution. The distance-to-self approach for severity assessment, on the other hand, does not require extensive specific tests for development; however, distance thresholds must be
established for the discrete set of linguistic values typically used for direct quantitative evaluation.

![Figure 4.4. Failure Direct Quantitative Evaluation Using the Structured Non-Self Approach](image)

### 4.1.3. Performance of Direct Evaluation Using the Structured Non-self Approach

The identification of the affected system, the determination of the type of AC, and its severity have been performed simultaneously using the SNSA. The performance evaluation was performed using similar metrics as for the detection phase. A correct identification percentage and a miss-identification percentage are calculated depending on an accurate declaration of subsystem failure for every time step in which an upset condition is present. The identifier generation algorithm proposed in this research effort was implemented for 9 different failures considered to be high magnitude using the 183 selected projections. Based on the assumption that lower magnitude failures of the same type of failure generate similar dynamic fingerprints with a close proximity to the self, the set of identifiers was subdivided into two groups. The first set corresponds to high magnitude and the second set to low magnitude failures (i.e. closer to the self). This approach increases the total amount of equivalent AC that can be identified to 18 instead of the original 9. A total of 1647 different cases for identifier generation were implemented in order to cover all the possible failure outputs investigated.

Each set of identifiers generated per failure contains on average 36 identifiers. Considering that every set of identifiers for all failures is integrated into each projection, an approximate total of 324 identifiers per projection are used for the identification through this positive selection-type of process. After an initial analysis, the algorithm was optimized and it was determined
that out of the 183 projections, a total of 93 projections were enough to correctly identify the investigated failures. The reduction of the total number of projections required for identification has reduced the computational complexity of the algorithm considerably. Table 4.1 presents a sample set of projections used for AC identification and direct evaluation.

Table 4.1. Projections Used for Simultaneous Identification and Direct Evaluation

<table>
<thead>
<tr>
<th>Self#</th>
<th>Features</th>
<th>Self#</th>
<th>Features</th>
<th>Self#</th>
<th>Features</th>
<th>Self#</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>(p_{ref})</td>
<td>(NN_p)</td>
<td>57</td>
<td>(r_{ref})</td>
<td>(NN_q)</td>
<td>110</td>
<td>(NN_q)</td>
</tr>
<tr>
<td>4</td>
<td>(p_{ref})</td>
<td>(NN_q)</td>
<td>60</td>
<td>(r_{ref})</td>
<td>(OQEE)</td>
<td>111</td>
<td>(NN_q)</td>
</tr>
<tr>
<td>7</td>
<td>(p_{ref})</td>
<td>(OQEE)</td>
<td>82</td>
<td>(NN_p)</td>
<td>(NN_q)</td>
<td>113</td>
<td>(NN_q)</td>
</tr>
<tr>
<td>9</td>
<td>(p_{ref})</td>
<td>(DQEEq)</td>
<td>83</td>
<td>(NN_p)</td>
<td>(NN_q)</td>
<td>114</td>
<td>(NN_q)</td>
</tr>
<tr>
<td>31</td>
<td>(q_{ref})</td>
<td>(NN_q)</td>
<td>84</td>
<td>(NN_p)</td>
<td>(MQEE)</td>
<td>115</td>
<td>(NN_q)</td>
</tr>
<tr>
<td>34</td>
<td>(q_{ref})</td>
<td>(OQEE)</td>
<td>86</td>
<td>(NN_p)</td>
<td>(DQEEp)</td>
<td>116</td>
<td>(NN_q)</td>
</tr>
<tr>
<td>35</td>
<td>(q_{ref})</td>
<td>(DQEEp)</td>
<td>87</td>
<td>(NN_p)</td>
<td>(DQEEq)</td>
<td>117</td>
<td>(NN_q)</td>
</tr>
<tr>
<td>42</td>
<td>(q_{ref})</td>
<td>(q)</td>
<td>88</td>
<td>(NN_p)</td>
<td>(DQEEr)</td>
<td>118</td>
<td>(NN_q)</td>
</tr>
<tr>
<td>56</td>
<td>(r_{ref})</td>
<td>(NN_p)</td>
<td>107</td>
<td>(NN_q)</td>
<td>(NN_r)</td>
<td>120</td>
<td>(NN_q)</td>
</tr>
</tbody>
</table>

Further analysis was carried out to reduce the number of projections required to produce desirable identification outputs. In some cases, the use of a single projection was enough to obtain favorable identification rates. On the other hand, other failures require more projections in order to obtain desirable identification results and also to reduce misidentification rates. Table 4.2 presents the number of projections required for a correct failure identification output.

Table 4.2. Total Number of Projections Used for Simultaneous Identification and Direct Evaluation

<table>
<thead>
<tr>
<th>Failure #</th>
<th>Failure Type</th>
<th>Projections Used</th>
<th>Failure #</th>
<th>Failure Type</th>
<th>Proj. Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Left Aileron Stuck at 2deg</td>
<td>14</td>
<td>9</td>
<td>Left Wing Loss of 6%</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>Right Aileron Stuck at 2deg</td>
<td>7</td>
<td>10</td>
<td>Right Wing Loss of 6%</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Left Aileron Stuck at 8deg</td>
<td>8</td>
<td>11</td>
<td>Left Wing Loss of 15%</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>Right Aileron Stuck at 8deg</td>
<td>8</td>
<td>12</td>
<td>Right Wing Loss of 15%</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Left Stabilator Stuck at 2deg</td>
<td>18</td>
<td>13</td>
<td>Left Engine Out</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Right Stabilator Stuck at 2deg</td>
<td>2</td>
<td>14</td>
<td>Right Engine Out</td>
<td>18</td>
</tr>
<tr>
<td>7</td>
<td>Left Stabilator Stuck at 8deg</td>
<td>9</td>
<td>15</td>
<td>Roll Sensor Bias of 5deg/sec</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>Right Stabilator Stuck at 8deg</td>
<td>31</td>
<td>16</td>
<td>Roll Sensor Bias of 10deg/sec</td>
<td>7</td>
</tr>
</tbody>
</table>

The identification and direct evaluation algorithm was tested under 16 different failures (refer to Table 4.2). Table 4.3 shown below presents the results for the performance rate.
Note that in Table 4.3 the first column represents the actual failure and the corresponding row represents how it was identified/evaluated. For example, failure #1 is identified correctly 99.7% of the time but presents confusion with failures 3, 4 and 14 for 0.1% of the time, respectively.

**Table 4.3. Performance of AC Identification and Direct Evaluation**

<table>
<thead>
<tr>
<th>Identified Failure #</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>99.7</td>
<td>0</td>
<td>0.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>9.9</td>
<td>87.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0.4</td>
<td>0</td>
<td>95.6</td>
<td>0.6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td>97.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>2.4</td>
</tr>
<tr>
<td>5</td>
<td>0.5</td>
<td>1.7</td>
<td>1.5</td>
<td>92.5</td>
<td>1.5</td>
<td>1.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.2</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>9.9</td>
<td>0</td>
<td>1.2</td>
<td>0</td>
<td>0</td>
<td>86.8</td>
<td>0</td>
<td>2.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0.2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>96.1</td>
<td>1.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4.8</td>
<td>93.8</td>
<td>0.9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>1.2</td>
<td>2</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td>0</td>
<td>0.9</td>
<td>95.6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>2.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.1</td>
<td>94.5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.1</td>
<td>0</td>
<td>1.2</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td>7.6</td>
<td>0.1</td>
<td>92.2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1.1</td>
<td>0</td>
<td>0</td>
<td>1.1</td>
<td>0</td>
<td>97.5</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>92.6</td>
<td>7.4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td>99.9</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
<td>0.2</td>
<td>0</td>
<td>0.2</td>
<td>0</td>
<td>2.1</td>
<td>1.3</td>
<td>95.6</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2.7</td>
<td>0</td>
<td>0</td>
<td>2.1</td>
<td>0</td>
<td>94.9</td>
</tr>
</tbody>
</table>
4.2. Dendritic Cell Mechanism for Abnormal Condition Direct Evaluation

4.2.1. Dendritic Cell Approach for Qualitative Direct Evaluation

The AC qualitative evaluation assumes that correct detection and identification have been performed. At this point it is known that a failure is affecting at least one sub-system \( k \), \( k = 1, 2, \ldots, N_S \). With the DC approach, the AC qualitative evaluation is approached as a specific identification problem, using the same algorithms. While the AC identification presented previously is a pattern recognition problem, where one must distinguish between \( N_S \) different patterns each corresponding to one of the \( N_S \) sub-systems considered, the AC qualitative evaluation is equally a pattern recognition problem, where one must distinguish between \( N_{ftk} \) different types of failures corresponding to sub-system \( k \). In a similar manner to the SNSA, the AC qualitative evaluation based on the DC mechanism may be viewed as an identification process where the target is the “AC”, which has the type of the failure as defining element in association with the affected sub-system. Patterns for qualitative evaluation are similar to the ones defined for identification and rely on the \( F_f \) matrix. Note that the \( F_0 \) matrix may also be introduced into the process. The flowchart of the process of establishing qualitative evaluation patterns is presented in Figure 4.5.

For each sub-system, \( N_{ftk} \) different patterns must be established, one associated to each of the AC considered.

As is the case with AC identification, three possible alternatives exist for the definition of the patterns:

- Feature-pattern (FP) approach
- Projection-pattern (PP) approach
- Matrix-pattern (MP) approach

**Figure 4.5.** Generation of Library of Reference Patterns for Qualitative Evaluation Using the DC Mechanism and the Naïve Bayes Classifier
A). Feature-pattern (FP) approach for AC qualitative evaluation

For each sub-system $k$, define for all $N_{f,k}$ ACs the vector of membership values of each feature to the set of equivalent directly involved variables (this is the same AC FP vector as described earlier):

$$ FP_{ACj} = [m_1, m_2, \ldots, m_{N}]^T, \quad m_i \in [0, 1], \quad j = 1, 2, \ldots, N_{f,k} $$

(4.8)

All these membership values can be determined based on tests and heuristics. Note that binary logic can be used instead of fuzzy logic. In this case: $m_i \in \{0, 1\}$. The $N_{f,k}$ AC FP vectors form a library of patterns against which the current AC FP vector is compared to determine the AC type. The current AC FP vector is computed at each sample time, after an AC is detected, using the $F_I$ matrices of the mature DCs, as part of the identification process:

$$ F_{1\phi} = \left[ \sum_{q=1}^{N_{DC}} F_{1q} \right] \cdot I_{N_{SS}}, \quad I_{N_{SS}} \in \{1\}^{N_{SS} \times 1} $$

(4.9)

A matching algorithm must be used to determine which one of the $N_{f,k}$ AC FP vectors best matches $F_{1\phi}$. Note that AC qualitative can be performed before identification, with possibly some benefits (to be investigated).

B). Projection-pattern (PP) approach for AC qualitative evaluation

For each sub-system $k$, define for all $N_{f,k}$ ACs the vector of membership values of each projection to the set of AC-relevant projections (projections that can capture the dynamic fingerprint of the AC, that are the projections that get triggered under AC):

$$ PP_{ACj} = [n_1, n_2, \ldots, n_{N_{SS}}]^T, \quad n_i \in [0, 1] $$

(4.10)

This is the same AC PP vector as for the previous method. All these membership values can be determined based on tests and heuristics. Binary logic may also be used here. At each sample time, after an AC is detected, use the $F_I$ matrices of the mature DCs to compute the current ACPP vector as:

$$ F_{1p} = I_N \cdot \left[ \sum_{q=1}^{N_{DC}} F_{1q} \right], \quad I_N \in \{1\}^{1 \times N_{SS}} $$

(4.11)

A matching algorithm must be used to determine which one of the $N_{f,k}$ AC PP vectors from the library best matches $F_{1p}$. 

89
**C. Matrix-pattern (MP) approach for AC qualitative evaluation**

For each subsystem k, define for all \(N_{f_k}\) ACs the AC FP and the AC PP vectors as described previously. Build then the MP matrix as:

\[
MP_k= \begin{bmatrix}
m_{11}^* & m_{12}^* & \cdots & m_{1N_{SS}}^* \\
m_{21}^* & m_{22}^* & \cdots & m_{2N_{SS}}^* \\
\vdots & \vdots & \ddots & \vdots \\
m_{N1}^* & m_{N2}^* & \cdots & m_{NN_{SS}}^*
\end{bmatrix},
\]

(4.12)

where \(m_{ij}^*\in[0,1]\), \(i=1,2,\ldots,N\), \(j=1,2,\ldots,N_{SS}\), \(k=1,2,\ldots,N_{f_k}\) (4.13)

and \(m_{ij}^* = \begin{cases} 
\max(m_i,n_j) & \text{if } \varphi_i \text{ belongs to } S_j \\
0 & \text{otherwise}
\end{cases}\) (4.14)

At each sample time, after the AC is detected, compute the sum of all mature DC \(F_{1}\) matrices as:

\[
F_{1M} = \sum_{q=1}^{N_{DC}} F_{1q}, \quad q=1,2,\ldots,N_{DC}
\]

(4.15)

Use a matching algorithm to determine which one of the \(N_{f_k}\) MP matrices in the library best matches \(F_{1M}\).

Similarly to the SNSA, the AC identification and qualitative evaluation can be performed simultaneously by redefining ACs to include both affected subsystem and AC type as category designations. The pattern generation process for the joint approach is summarized in Figure 4.6.

**Figure 4.6.** DC Mechanism for Simultaneous AC Identification and Qualitative Evaluation
4.2.2. Dendritic Cell Approach for Quantitative Direct Evaluation

Similar to the AC qualitative evaluation, patterns for each failure severity can be defined based on the detection outcome of all subselves based on the F1 matrices of the migrated DCs. If the number of severity scales of failure $AC_j$ is $Nfs$, then $Nfs$ different reference patterns must be established: one associated to each severity scale. The reference F-P vector corresponding to the failure severity $AC_{js}$ can be expressed as:

$$FP_{AC_{js}} = [\tilde{m}_1, \tilde{m}_2, \ldots, \tilde{m}_N]^T,$$

(4.16)

where $\tilde{m}_i \in [0,1]$ are fuzzy membership values of each feature with respect to the set of directly involved variables. The reference F-P of Eq. (4.16) can be determined through a training process with experimental or simulation data under failure exposed to the artificial DC mechanism. The naïve Bayes classifier described previously can be used to construct these reference patterns by training the classifier offline against samples from a set of training tests of known failure severity scales. Training the classifier for simultaneous detection and qualitative evaluation is very similar to training it for AC qualitative evaluation as shown in the schematic diagram of Figure 4.7.

![Figure 4.7. Training the Naïve Bayes Classifier for AC Detection and Qualitative Evaluation](image)

The current features-pattern $F_{ip}$ is computed and compared to the library of reference patterns to determine the discriminant $AC_{js}$ for each severity scale from:

$$A_{js}(x) = \ln N_{fs} - \frac{1}{2} \ln |\Sigma_{fs}| - \frac{1}{2} (x - \mu_{fs})^T \Sigma_{fs}^{-1} (x - \mu_{fs}),$$

(4.17)

where $N_{fs}$ is the number of samples in tests with failure severity $fs$, $x = F_{ip}$, and $\mu_{fs}$ and $\Sigma_{fs}$ are the mean vector and sample covariance matrix given by:
\[ \boldsymbol{\mu}_{fs} = \frac{1}{N_{fs}} \sum_{m=1}^{N_{fs}} x_{fs}^{(m)}, \]  

and

\[ \Sigma_{fs} = \frac{1}{N_{fs} - 1} \sum_{m=1}^{N_{fs}} (x_{fs}^{(m)} - \mu_{fs})(x_{fs}^{(m)} - \mu_{fs})^T, \]

respectively. The block diagram of AC detection and qualitative evaluation scheme is presented in Figure 4.8.

![Block Diagram](image)

**Figure 4.8.** AC detection and Qualitative Evaluation Scheme Using the DC Mechanism

### 4.2.3. Performance of Direct Evaluation Using the Dendritic Cell Approach

The F-P approach was also used to implement the AC detection and qualitative evaluation scheme with the DC mechanism. The naïve Bayes classifier was trained using two sets of training tests under failure. The first training set provides the reference features-pattern \( FP_{fs} \) for each failure severity \( fs \). Figures 4.9 through 4.14 illustrate the reference features patterns for subsystems under different levels of failure severity.

Table 4.4 presents the AC detection and qualitative evaluation rates for the four main subsystems considered in this study. The rate is defined as the number of time steps for which the failure severity is correctly evaluated divided by the total number of time steps after failure. The second column in this table represents the “known” failure severity of the corresponding subsystem, whereas the first row represents the evaluated failure severity. Note that only “low” and “high” severity levels were simulated for the sensors and engine failures. These results show the capability of the proposed scheme in isolating the failure severity with very high rates.
Figure 4.9. Reference Feature Patterns for a Stabilator Under Low Severity Failure

Figure 4.10. Reference Feature Patterns for a Stabilator Under High Severity Failure

Figure 4.11. Reference Feature Patterns for a Wing Under Low Severity Failure

Figure 4.12. Reference Feature Patterns for a Wing Under High Severity Failure

Figure 4.13. Reference Feature Patterns for a Roll Rate Sensor Under Low Severity Failure

Figure 4.14. Reference Feature Patterns for a Roll Rate Sensor Under High Severity Failure
Table 4.4. AC Detection and Qualitative Evaluation Rates of the DC Mechanism Scheme

<table>
<thead>
<tr>
<th>Subsystem</th>
<th>Failure Severity</th>
<th>Low</th>
<th>Medium</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stabilator</td>
<td>Low</td>
<td>98.72</td>
<td>1.28</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>0.04</td>
<td>99.96</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>0.05</td>
<td>0.00</td>
<td>99.95</td>
</tr>
<tr>
<td>Aileron</td>
<td>Low</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>3.10</td>
<td>95.47</td>
<td>1.43</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>0.11</td>
<td>0.00</td>
<td>99.89</td>
</tr>
<tr>
<td>Rudder</td>
<td>Low</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>0.01</td>
<td>99.99</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>0.00</td>
<td>0.00</td>
<td>100.00</td>
</tr>
<tr>
<td>Wing</td>
<td>Low</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>0.01</td>
<td>99.99</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>0.06</td>
<td>0.00</td>
<td>99.94</td>
</tr>
<tr>
<td>Horizontal Tail</td>
<td>Low</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>0.00</td>
<td>99.99</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>0.01</td>
<td>0.00</td>
<td>99.99</td>
</tr>
<tr>
<td>Vertical Tail</td>
<td>Low</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>Medium</td>
<td>0.01</td>
<td>99.99</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>0.00</td>
<td>0.00</td>
<td>100.00</td>
</tr>
<tr>
<td>Roll Rate Sensor</td>
<td>Low</td>
<td>99.99</td>
<td>—</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>0.00</td>
<td>—</td>
<td>100.00</td>
</tr>
<tr>
<td>Pitch Rate Sensor</td>
<td>Low</td>
<td>100.00</td>
<td>—</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>0.00</td>
<td>—</td>
<td>100.00</td>
</tr>
<tr>
<td>Yaw Rate Sensor</td>
<td>Low</td>
<td>100.00</td>
<td>—</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>0.01</td>
<td>—</td>
<td>99.99</td>
</tr>
<tr>
<td>Engine</td>
<td>Low</td>
<td>100.00</td>
<td>—</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>High</td>
<td>0.00</td>
<td>—</td>
<td>100.00</td>
</tr>
</tbody>
</table>
Chapter 5

Flight Envelope Estimation Under Abnormal Conditions Using Artificial Immune System

5.1. Problem Formulation

The AC indirect quantitative evaluation requires that the direct evaluation is successful. While for detection, identification, qualitative evaluation, and direct quantitative evaluation consistent general design algorithms can be applied as presented so far that are valid for a variety of systems and AC types without significant changes, the indirect quantitative failure evaluation process or altered flight envelope evaluation requires specific customization depending on the subsystem, nature of the AC, and the affected parameter of the flight envelope addressed by the evaluation. However, a general framework can be still formulated up to a point. A comprehensive indirect evaluation or altered flight envelope prediction must rely on a combined strategy based on analytical flight envelope assessment and AIS-based approaches for parameter space alteration assessment. The analytical methods require accurate modeling of the failures and significant on-line computational capabilities. The AIS method implies that all pertinent parameters to the flight envelope – considering its generalized meaning – are part of the feature sets that define the “self”.

Let us assume that the self $S$ and non-self $\hat{S}$ are defined as sets of $N$-dimensional hyperspheres according to Eq. (1.27) and (1.28). Some main concepts previously defined for the AC evaluation process will be repeated here for completeness. A directly involved variable (DIV) in the AC is a variable whose alteration or abnormal variation is directly and significantly the result of the AC. Typically, DIV are used to define(characterize the AC. They may be part of the feature set or not. If they are not, then a relationship between the DIV and some other variable(s) in the feature set must be established. This process will define equivalent directly involved variables (EDIV), which must be part of the feature set. For example, consider the case of the left stabilator locked failure. The DIV can be defined as the left stab deflection $e_{L\delta}$. It obviously defines the failure, but let us assume that it is not part of the feature set. A relationship between the left stab deflection $e_{L\delta}$ and the longitudinal stick displacement $d_e$, which is a feature, can be established:

$$d_e = f(e_{L\delta})$$

Therefore, the EDIV is in this case $d_e$. The self can be viewed as a generalized flight envelope based on features $\varphi_i$. Assume that each failure $f_i \in F$, $i = 1,2,\ldots,N_F$, produces a set of $N_{\Gamma_i}$ constraints $\Gamma_i$ on a set of known variables $X_i$, where:

$$\Gamma_i = \{\gamma_{i1}, \gamma_{i2}, \ldots, \gamma_{iN_{\Gamma_i}}\}$$

$$\gamma_{ij} = \gamma_{ij}(X_i) = 0, \quad i = 1,2,\ldots,N_F \quad \text{and} \quad j = 1,2,\ldots,N_{\Gamma_i}$$

The variables $X_i$ must be part of the feature set, that is:
Let us assume that the alteration of the flight envelope is assessed in terms of a set \( Y \subseteq \mathcal{X} \) of \( N_Y \) variables \( y_i \), that is:

\[
Y = \{y_1, y_2, \ldots, y_{N_Y}\} \subseteq \mathcal{X}
\]

(5.5)

Typically the constraints \( \Gamma_i \) are not completely known, otherwise they would specify any alteration of the variables \( y_i \) and the problem would be solved. The solution is equivalent to determining a new self at post-failure conditions based on a set of features \( Y \). If \( Y = \mathcal{X} \) then a solution is found if perfect structuring of the non-self is possible, which in the most general case is not. If \( N_Y < N \), then the flight envelope alteration assessment is equivalent to obtaining the \( N_Y \)-dimensional projection of the \( N \)-dimensional non-self region corresponding to the targeted failure \( f_i \). Again, if structuring of the non-self for \( f_i \) is available, then the problem is solved. If not, we can still obtain the \( N_Y \)-dimensional projection of the self. In many cases, depending on the failure, the constraints \( \Gamma_i \), and the set \( Y \) we can infer the relative position of the \( N_Y \)-dimensional projections of the non-self and the self. For example, let us consider a stuck elevator failure (the left or right elevator is locked at a constant deflection \( \delta_{e0} \)). The effects on aircraft pitch rate \( q \) must be assessed. Therefore, \( x_1 = \delta_e, y_1 = q \), and \( y_1 = \delta_e - \delta_{e0} = 0 \). Due to the nature of this failure, it can be inferred that the values of the pitch rate at post failure conditions will be in the range of normal conditions corresponding to \( \delta_{e0} \). In other words, the projection of the non-self on the 2-dimensional plane \((x_1, y_1)\) falls inside the projection of the self. For such situations, a “new” self \( \mathcal{S} \) or reduced envelope at post failure conditions can then be defined as:

\[
\mathcal{S} = \{c_i \mid c_i \text{ satisfies constraints } \Gamma_i\}
\]

(5.6)

The concept is illustrated in Figure 5.1 for the 2-dimensional case.

The development of schemes for indirect quantitative failure evaluation (IQE) is based on Table 1.2. Specific algorithms must be developed for all targeted variables \((N_E = 16)\) in conjunction with the nature and type of the failure.

It is assumed that failure detection, identification, and direct evaluation are successfully completed at this point. Therefore, the sets of DIV \((v_\delta)\), EDIV \((v_e)\), and envelope relevant variables \((v_E)\) are determined. Note that, in general:

\[
v_\delta = \{v_{\delta 1}, v_{\delta 2}, \ldots, v_{\delta N_\delta}\} \subseteq \mathcal{X}
\]

(5.7)

\[
v_e = \{v_{e 1}, v_{e 2}, \ldots, v_{e N_e}\} \subseteq \mathcal{X}
\]

\[
v_E = \{v_{E 1}, v_{E 2}, \ldots, v_{E N_E}\} \subseteq \mathcal{X}
\]

\[v_E \subseteq \mathcal{X}_E \subseteq \mathcal{X}\]
For IQE, a relationship between $v_e$ and $v_\delta$ must be known:

$$v_e = f_{e\delta}(v_\delta)$$  \hspace{1cm} (5.8)

where

$$f_{e\delta} = \begin{bmatrix} f_{e\delta 1} & f_{e\delta 2} & \cdots & f_{e\delta N_e} \end{bmatrix}$$  \hspace{1cm} (5.9)

such that

$$v_{ei} = f_{e\delta}(v_\delta), \quad i = 1, 2, \ldots, N_e$$  \hspace{1cm} (5.10)

Figure 5.1. Post-Failure Flight Envelope Estimation Using the AIS Paradigm
5.2. Estimation Algorithms for Actuator AC

5.2.1. Stabilator Failure

This discussion covers failures $k_F = 1$ through $k_F = 4$ of Table 1.2, since, from the point of view of the IQE, they are equivalent. The discussion will continue with reference to the left stab; however, everything applies similarly to the right surface failure as well. For $k_F = 1 \div 4$, the following parameters can be specified:

\[ v_\delta = [v_{\delta L}] = [\delta_{eL}], \text{ left (or right) stab deflection, positive downwards} \]
\[ v_e = [v_{eL}] = [d_e], \text{ longitudinal stick} \]
\[ v_E = [a_x \ a_z \ H \ M \ V \ q \ p \ \dot{q} \ \alpha \ \theta] \]

The relationship between DIV and EDIV is:

\[ d_e = f_{e\delta L} (\delta_{eL}) \]

where

\[ d_e = k_e \delta_e = k_e \frac{\delta_{eL} + \delta_{eR}}{2} \]

If the left stab is locked, then:

\[ \delta_{eL}(t) = \delta_{eLF} \]

where $\delta_{eLF}$ is a value determined through direct quantitative evaluation. The nominal range for $d_e$ is (typically $d_{e_{max}} = -d_{e_{min}}$):

\[ d_{e_{max}} = k_e \frac{\delta_{eL_{max}} + \delta_{eR_{max}}}{2} \]
\[ d_{e_{min}} = k_e \frac{\delta_{eL_{min}} + \delta_{eR_{min}}}{2} \]

After failure, the range for $d_e$ becomes:

\[ d_{e_{max}F} = k_e \frac{\delta_{eLF} + \delta_{eR_{max}}}{2} \]
\[ d_{e_{min}F} = k_e \frac{\delta_{eLF} + \delta_{eR_{min}}}{2} \]

Note that this is a rather virtual or equivalent range, since the stick in the cockpit is assumed to still be capable of moving within the nominal range, only the effects are different. Note also that
the range at post failure conditions \([d_{eminF}, d_{emaxF}]\) is invariable with respect to the surface failed (L or R) and that \([d_{eminF}, d_{emaxF}] \subseteq [d_{emin}, d_{emax}]\). With the new range after failure, new ranges for all \(v_E\) may be obtained from the 2-D projections, if there is a mutual constraint (cause/effect relationship) between the two features involved in the projection. If there is not, it means that one variable will reach all its values in the nominal range irrespective of the other variable values and the self projection will be a square. In this cases, if it is known that the variable involved is part of \(v_E\), special consideration must be given. For example, a locked stab failure will have an effect on the capability to produce rolling rate. The asymmetry induced by a locked stab failure produces undesired roll moments and rates that must be typically compensated by aileron deflections. These deflections affect the control authority on the roll channel and thus the production of roll rate. This effect is rather indirect as the self projection defined by \(d_e\) and \(p\) is a square. To determine the effects on \(p\) produced by a locked stab failure, it will be stipulated that the locked stab failure induces an aileron pseudo-failure, from the point of view of producing roll rate. This pseudo-failure involves a non-zero aileron trim deflection at a position \(\delta_{aPF}\) necessary to compensate for the rolling effect of the locked stab failure. The compensating aileron deflection can be approximated using the balance of the rolling moment:

\[
C_l \delta_{el} \cdot \delta_{eLF} + C_l \delta_{eR} \cdot \delta_{eR} = -C_l \delta_a \cdot \delta_{aPF} \tag{5.19}
\]

\[
C_l \delta_{el} \cdot (\delta_{eLF} - \delta_{eR}) = -C_l \delta_a \cdot \delta_{aPF} \tag{5.20}
\]

\[
\delta_{aPF} = -\frac{C_l \delta_{el}}{C_l \delta_a} (\delta_{eLF} - \delta_{eR}) \tag{5.21}
\]

The following convention for the definition of the generic aileron deflection will be assumed, which is the one used for the aerodynamic model of the aircraft within the simulation environment:

\[
\delta_a = \delta_{aL} - \delta_{aR} \tag{5.22}
\]

With linearity assumption, the relationship with the EDIV is:

\[
d_a = k_a \delta_a \tag{5.23}
\]

The nominal lateral control range is:

\[
d_{a max} = k_a \delta_{a max} \tag{5.24}
\]

\[
d_{a min} = k_a \delta_{a min} \tag{5.25}
\]

Under stab failure, the new range for the lateral control is:
\[ d_{a \text{max} F} = k_a (\delta_{a \text{max}} - \delta_{aPF} ) \] ..(5.26)

\[ d_{a \text{min} F} = k_a (\delta_{a \text{min}} - \delta_{aPF} ) \] (5.27)

Note that this is equivalent to shifting the range of \( d_a \) in one direction or the other (depending on the sign of the stab failure), while maintaining the same span:

\[ [d_{a \text{min} F}, d_{a \text{max} F}] \subset [d_{a \text{min}}, d_{a \text{max}}] \] (5.28)

\[ d_{a \text{max} F} - d_{a \text{min} F} = d_{a \text{max}} - d_{a \text{min}} \] (5.29)

Now the new range of \( d_a \) can be used with the \((d_a, p)\) projection to determine the effects on \( p \) of the stab failure. Note that extrapolation on the projection is needed, since the new range ends up outside the old one. This range can be expressed in terms of the failed surface as follows:

\[ d_{a \text{max} F} = k_a [\delta_{a \text{max}} + \frac{C_{l\delta_{eL}}}{C_{l\delta_a}} (\delta_{eLF} - \delta_{eR})] \] (5.30)

\[ d_{a \text{min} F} = k_a [\delta_{a \text{min}} + \frac{C_{l\delta_{eL}}}{C_{l\delta_a}} (\delta_{eLF} - \delta_{eR})] \] (5.31)

If the right stab fails, then the range can be expressed as:

\[ d_{a \text{max} F} = k_a [\delta_{a \text{max}} + \frac{C_{l\delta_{eR}}}{C_{l\delta_a}} (\delta_{eRF} - \delta_{eL})] \] (5.32)

\[ d_{a \text{min} F} = k_a [\delta_{a \text{min}} + \frac{C_{l\delta_{eR}}}{C_{l\delta_a}} (\delta_{eRF} - \delta_{eL})] \] (5.33)

Note that \( C_{l\delta_{eL}} = -C_{l\delta_{eR}} \). Therefore, the aileron pseudo-failure range is the same if the opposite stab is locked in the opposite direction relative to the trim. Span is preserved.

### 5.2.2. Aileron Failure

This discussion covers failures \( k_F = 5 \) through \( k_F = 8 \) of Table 1.2, since, from the point of view of the indirect quantitative evaluation, they are equivalent. The discussion will continue with reference to the left aileron; however, everything applies similarly to the right surface failure as well, unless otherwise noted. For \( k_F = 5 \div 8 \), one can specify:
\( v_\delta = [v_{\delta L}] = [\delta_{aL}], \) left (or right) aileron deflection, positive downwards
\( v_\epsilon = [v_{\epsilon L}] = [d_a], \) lateral stick
\( v_E = [a_y \ p \ r \ \dot{p} \ \beta \ \varphi] \)

The relationship between DIV and EDIV is:

\[ d_a = f_{\epsilon\delta L}(\delta_{aL}) \] (5.35)

\[ d_a = k_a(\delta_{aL} - \delta_{aR}) \] (5.36)

If the left aileron is locked, then:

\[ \delta_{aL}(t) = \delta_{aLF} \] (5.37)

where \( \delta_{aLF} \) is a value determined through direct quantitative evaluation. The nominal range for \( d_a \) is (typically \( d_{a_{\text{max}}} = -d_{a_{\text{min}}} \)):

\[ d_{a_{\text{max}}} = k_a(\delta_{aL_{\text{max}}} - \delta_{aR_{\text{min}}}) \] (5.38)

\[ d_{a_{\text{min}}} = k_a(\delta_{aL_{\text{min}}} - \delta_{aR_{\text{max}}}) \] (5.39)

After failure, the range for \( d_a \) becomes:

\[ d_{a_{\text{max}}F} = k_a(\delta_{aLF} - \delta_{aR_{\text{min}}}) \] (5.40)

\[ d_{a_{\text{min}}F} = k_a(\delta_{aLF} - \delta_{aR_{\text{max}}}) \] (5.41)

The same observation must be made as in the case of the stab, that is that this is a rather virtual or equivalent range, since the stick in the cockpit is assumed to still be capable of moving within the nominal range, only the effects are different. Note also that the span of the range at post failure conditions is reduced \([d_{a_{\text{min}}F}, d_{a_{\text{max}}F}] \subset [d_{a_{\text{min}}}, d_{a_{\text{max}}} ]; \) however, the post failure range is not invariable with respect to the surface failed (L or R). Assume that the range of the generic aileron deflection at normal conditions is ±30 deg. The ranges for left and right aileron deflections are ±15. If the left aileron is locked at 5 deg, then the post failure range is:

\[ d_{a_{\text{max}}F} = k_a(5 - (-15)) = 20k_a \] (5.42)

\[ d_{a_{\text{min}}F} = k_a(5 - 15) = -10k_a \] (5.43)

If the right aileron is locked at 5 deg, then the post failure range is:
\[ d_{\text{a max}} F = k_a(15 - 5) = 10k_a \]  \hspace{1cm} (5.44)

\[ d_{\text{a min}} F = k_a(-15 - 5) = -20k_a \]  \hspace{1cm} (5.45)

Note that this range will be the same if the opposite surface is locked in the opposite direction. With the new range after failure, new ranges for all \( v_E \) may be obtained from the 2-D projections.

### 5.2.3. Rudder Failure

This discussion covers failures \( k_F = 9 \) through \( k_F = 12 \) of Table 1.2, since, from the point of view of the indirect quantitative evaluation, they are equivalent. The discussion will continue with reference to the left rudder; however, everything applies similarly to the right surface failure as well, unless otherwise noted. For \( k_F = 9 \div 12 \), the following parameters can be specified:

\[ v_\delta = [v_{\delta L}] = [\delta_{rL}], \text{ left (or right) rudder deflection, positive to left of pilot such that it produces positive lift on the vertical tail in body axes} \]

\[ v_e = [v_{eL}] = [d_r], \text{ pedals} \]

\[ v_E = [a_y \ p \ r \ \dot{p} \ \dot{r} \ \beta \ \varphi] \]  \hspace{1cm} (5.46)

The relationship between DIV and EDIV is:

\[ d_r = f_{\delta \delta L}(\delta_{rL}) \]  \hspace{1cm} (5.47)

\[ d_r = k_r \frac{\delta_{rL} + \delta_{rR}}{2} \]  \hspace{1cm} (5.48)

If the left rudder is locked, then:

\[ \delta_{rL}(t) = \delta_{rLF} \]  \hspace{1cm} (5.49)

where \( \delta_{rLF} \) is a value determined through direct quantitative evaluation. The nominal range for \( d_r \) is (typically \( d_{r\text{max}} = -d_{r\text{min}} \)):

\[ d_{r\text{max}} = k_r \frac{\delta_{rL\text{max}} + \delta_{rR\text{max}}}{2} \]  \hspace{1cm} (5.50)

\[ d_{r\text{min}} = k_a \frac{\delta_{rL\text{min}} + \delta_{rR\text{min}}}{2} \]  \hspace{1cm} (5.51)

After failure, the range for \( d_r \) becomes:
\[
d_{r_{\text{max}F}} = k_{r} \frac{\delta_{rLF} + \delta_{rR_{\text{max}}}}{2} \tag{5.52}
\]
\[
d_{r_{\text{min}F}} = k_{r} \frac{\delta_{rLF} + \delta_{rR_{\text{min}}}}{2} \tag{5.53}
\]

As was the case with the other control channels, this is a rather virtual or equivalent range, since the pedals in the cockpit are assumed to still be capable of moving within the nominal range, only the effects are different. Note also that the range at post failure conditions (\([d_{r_{\text{min}F}}, \ d_{r_{\text{max}F}}]\)) is invariable with respect to the surface failed (L or R) and that \([d_{r_{\text{min}F}}, \ d_{r_{\text{max}F}}]<[d_{r_{\text{min}}}, \ d_{r_{\text{max}}}]\). The span of the range at post failure conditions is reduced (\([d_{r_{\text{min}F}}, \ d_{r_{\text{max}F}}]<[d_{r_{\text{min}}}, \ d_{r_{\text{max}}}]\)). Because of its collective deflection, the case of the rudder is similar, in many respects, to the stabilator, if the aircraft is equipped with dual rudder, as is the case with the aircraft used for demonstration in this project. With the new range after failure, new ranges for all \(v_{E}\) may be obtained from the 2-D projections.

### 5.2.4. Throttle Failure

This discussion covers failures \(k_{F} = 13\) through \(k_{F} = 16\) of Table 1.2, since, from the point of view of the indirect quantitative evaluation, they are equivalent. The discussion will continue with reference to the left (engine) throttle; however, everything applies similarly to the right actuator failure as well. For \(k_{F} = 13 \div 16\), the following key parameters can be specified:

\[
\begin{align*}
\nu_{\delta} &= [v_{\delta L}] = [\delta_{TL}], \ \text{left (or right) throttle/fuel valve, positive = open} \\
\nu_{e} &= [v_{e L}] = [d_T], \ \text{collective throttle (is assumed)} \\
\nu_{E} &= [a_{x} \ a_{z} \ r \ H \ M \ V \ \alpha \ \theta] \\
\end{align*}
\]

The relationship between DIV and EDIV is:

\[
d_{T} = f_{w\delta q}(\delta_{TL}) \tag{5.55}
\]

\[
d_{T} = k_{T}\delta_{T} = k_{T}\frac{\delta_{TL} + \delta_{TR}}{2} \tag{5.56}
\]

If the left throttle is locked, then:

\[
\delta_{TL}(t) = \delta_{TLF} \tag{5.57}
\]

where \(\delta_{TLF}\) is a value determined through direct quantitative evaluation. The nominal range for \(d_{T}\) is typically \(d_{T} \in [0, \ 100\%]\) with:
\[ d_{T_{\text{max}}} = k_T \frac{\delta_{\text{TL}_{\text{max}}} + \delta_{\text{TR}_{\text{max}}}}{2} = 100\% \quad (5.58) \]

and
\[ d_{T_{\text{min}}} = k_T \frac{\delta_{\text{TL}_{\text{min}}} + \delta_{\text{TR}_{\text{min}}}}{2} = 0\% \quad (5.59) \]

After failure, the range for \( d_T \) becomes:
\[ d_{T_{\text{max}}F} = k_T \frac{\delta_{\text{TLF}} + \delta_{\text{TR}_{\text{max}}}}{2} = k_T \frac{\delta_{\text{TLF}} + 100}{2} \quad (5.60) \]
\[ d_{T_{\text{min}}F} = k_T \frac{\delta_{\text{TLF}} + \delta_{\text{TR}_{\text{min}}}}{2} = k_T \frac{\delta_{\text{TLF}}}{2} \quad (5.61) \]

The throttle lever in the cockpit is assumed to still be capable of moving within the nominal range, only the effects are different; therefore, the range in Eq (5.60) and (5.61) is a rather virtual or equivalent range. Note also that the range at post failure conditions \( [d_{T_{\text{min}}F}, d_{T_{\text{max}}F}] \) is invariable with respect to the surface failed (L or R) and that \( [d_{T_{\text{min}}F}, d_{T_{\text{max}}F}] \subset [0, 100] \).

With the new range after failure, new ranges for all \( v_E \) may be obtained from the 2-D projections, if there is a direct mutual constraint (direct cause/effect relationship) between the two features involved in the projection. It seems that this is the case for some of the envelope relevant variables not for others. For instance, \( r \) appears to need some special attention since it is clear that the effects on \( r \) are different whether the L or R engine fails and the difference in L or R failure is not captured by the virtual control range after the failure. Due to asymmetry, a throttle failure will create an undesired yawing moment (depending on the distance of the engines with respect to the aircraft centerline), which needs to be compensated for by using rudder deflection. This will reduce the overall control authority on the yaw channel. A similar approach as in the case of the rolling moment compensated using ailerons, in the case of a stab failure, will be applied next. To determine the effects on \( r \) produced by a locked throttle failure, it will be stipulated that the locked throttle failure induces a rudder pseudo-failure, from the point of view of producing yawing rate. This pseudo-failure consists of a shift of the trim position of the rudder at \( \delta_{rPF} \) necessary to compensate for the yawing effect of the locked throttle. The compensating rudder deflection can be approximated using the balance of the yawing moment:
\[ C_n \delta_{\text{TTL}} \cdot (\delta_{\text{TLF}} - \delta_{\text{TR}}) = -C_n \delta_r \cdot \delta_{rPF} \quad (5.62) \]
\[ \delta_{rPF} = -\frac{C_n \delta_{\text{TTL}}}{C_n \delta_r} (\delta_{\text{TLF}} - \delta_{\text{TR}}) \quad (5.63) \]

Note that \( C_n \delta_{\text{TTL}} > 0 \) and \( C_n \delta_r < 0 \). Considering that \( \delta_{\text{TLF}} \in [0, 100] \) then:
\[ \delta_{rPF} \in \left[ \frac{C_n \delta_{TL}}{C_n \delta_r} \delta_{TR}, \frac{C_n \delta_{TL}}{C_n \delta_r} (100 - \delta_{TR}) \right] \] (5.64)

For full range of \( \delta_{TR} \) we get:

\[ \delta_{rPF} \in \left[ \frac{C_n \delta_{TL}}{C_n \delta_r} 100, \frac{C_n \delta_{TL}}{C_n \delta_r} 100 \right] \] (5.65)

Note that \( \frac{C_n \delta_{TL}}{C_n \delta_r} 100 > 0 \). If the right engine fails, then:

\[ \delta_{rPF} = \frac{C_n \delta_{TL}}{C_n \delta_r} (\delta_{TL} - \delta_{TRF}) \] (5.66)

Note that \( C_n \delta_{TR} < 0 \) and \( C_n \delta_r < 0 \). Considering that \( \delta_{TLR} \in [0, 100] \) then:

\[ \delta_{rPF} \in \left[ \frac{C_n \delta_{TR}}{C_n \delta_r} \delta_{TL}, \frac{C_n \delta_{TR}}{C_n \delta_r} (\delta_{TL} - 100) \right] \] (5.67)

For full range of \( \delta_{TL} \) we get:

\[ \delta_{rPF} \in \left[ -\frac{C_n \delta_{TR}}{C_n \delta_r} 100, \frac{C_n \delta_{TR}}{C_n \delta_r} 100 \right] \] (5.68)

Note that \( -\frac{C_n \delta_{TR}}{C_n \delta_r} 100 < 0 \). The generic rudder deflection is (for dual rudder aircraft), of course:

\[ \delta_r = \frac{\delta_{rL} + \delta_{rR}}{2} \] (5.69)

and the relationship with the EDIV is:

\[ d_r = k_r \delta_r \] (5.69)

The nominal directional control range is:

\[ d_{r max} = k_r \delta_{r max} \] (5.70)

\[ d_{r min} = k_r \delta_{r min} \] (5.71)

Under throttle failure, the new range for the directional control is:
\[ d_{r\text{ max} F} = k_r (\delta_{r\text{ max}} - \delta_{rPF} ) \]  \hfill (5.72)

\[ d_{r\text{ min} F} = k_r (\delta_{r\text{ min}} - \delta_{rPF} ) \]  \hfill (5.73)

Note that this is equivalent to shifting the range of \( d_r \) in one direction or the other (depending on the side of the failed engine and the throttle of the healthy engine), while maintaining the same span:

\[ [d_{r\text{ min} F}, d_{r\text{ max} F}] \sim [d_{r\text{ min}}, d_{r\text{ max}}] \]  \hfill (5.74)

\[ d_{r\text{ max} F} - d_{r\text{ min} F} = d_{r\text{ max}} - d_{r\text{ min}} \]  \hfill (5.75)

Now the new range of \( d_r \) can be used with the \((d_r, r)\) projection to determine the effects on \( r \) of the throttle failure. This range can be expressed in terms of the failed surface and the throttle of the healthy engine as follows:

\[ d_{r\text{ max} F} = k_r [\delta_{r \text{ max} } + \frac{C_n \delta_{TL}}{C_n \delta_r} (\delta_{TLF} - \delta_{TR} ) ] \]  \hfill (5.76)

\[ d_{r\text{ min} F} = k_r [\delta_{r \text{ min} } + \frac{C_n \delta_{TL}}{C_n \delta_r} (\delta_{TLF} - \delta_{TR} ) ] \]  \hfill (5.77)

Therefore, for \( \delta_{TR} = 100 \)

\[ \max(d_{r \text{ max} F}) = k_r [\delta_{r \text{ max} } + \frac{C_n \delta_{TL}}{C_n \delta_r} (\delta_{TLF} - 100 ) ] \]  \hfill (5.78)

and for \( \delta_{TR} = 0 \)

\[ \min(d_{r \text{ min} F}) = k_r [\delta_{r \text{ min} } + \frac{C_n \delta_{TL}}{C_n \delta_r} \delta_{TLF} ] \]  \hfill (5.79)

If the right engine fails, then the range can be expressed as:

\[ d_{r\text{ max} F} = k_r [\delta_{r \text{ max} } - \frac{C_n \delta_{TR}}{C_n \delta_r} (\delta_{TL} - \delta_{TRF} ) ] \]  \hfill (5.80)

\[ d_{r\text{ min} F} = k_r [\delta_{r \text{ min} } - \frac{C_n \delta_{TR}}{C_n \delta_r} (\delta_{TL} - \delta_{TRF} ) ] \]  \hfill (5.81)
Therefore, for $\delta_{TL} = 0$

$$\max(d_{r_{\max F}}) = k_r(\delta_{r_{\max}} + \frac{C_n\delta_{TL}}{C_n\delta_r}\delta_{TRF})$$

(5.82)

and for $\delta_{TL} = 100$

$$\min(d_{r_{\min F}}) = k_r[\delta_{r_{\min}} + \frac{C_n\delta_{TL}}{C_n\delta_r}(\delta_{TRF} - 100)]$$

(5.83)

Note that $C_n\delta_{TL} = -C_n\delta_{TR}$. Therefore, the rudder pseudo-failure range shifts towards positive deflections for a left engine failure, and towards negative deflections for a right engine failure. The span remains the same.
5.3. Estimation Algorithms for Sensor AC

5.3.1. Roll Rate Sensor Bias

The analysis of the sensor failures relies on the specifics of the control system that uses the sensor information. Based on stick displacement, the control laws determine first a “desired” angular rate response as determined by reference models (established using handling qualities criteria). Then, they track the desired angular rate response. For $k_F = 17$, one can specify:

\[ v_\delta = [v_{\delta 1}] = [p_{\text{meas}}], \text{ measured roll rate} \]
\[ v_E = [v_{E1}] = [d_a], \text{ lateral stick} \]
\[ v_E = [a_y \ p \ r \ \dot{p} \ \dot{r} \ \beta \ \varphi] \]

At nominal conditions, the desired or reference roll rate is:

\[ p_{\text{ref}} = F_1(d_a) \]  \hspace{1cm} (5.85)

where $F_1$ represents the roll channel reference model, a first order transfer function. If the roll rate measurement is assumed to be perfect:

\[ p_{\text{meas}} = p_{\text{act}} \]  \hspace{1cm} (5.86)

and the commanded roll rate is given by a control law of the form:

\[ p_{\text{cmd}} = F_2(p_{\text{ref}} - p_{\text{act}}) \]  \hspace{1cm} (5.87)

At failure condition, the reference is unchanged; however:

\[ p_{\text{measF}} = p_{\text{act}} + p_{\text{bias}} \]  \hspace{1cm} (5.88)
\[ p_{\text{cmdF}} = F_2(p_{\text{ref}} - p_{\text{act}} - p_{\text{bias}}) \]  \hspace{1cm} (5.89)

As a result of the failure, an undesired additional roll rate command is produced, which can be expressed with some simplifying assumptions at steady state as:

\[ \Delta p_{\text{cmd}} = p_{\text{cmdF}} - p_{\text{cmd}} = -p_{\text{bias}} \]  \hspace{1cm} (5.90)

In other words, the control laws will rotate the aircraft in the opposite direction of the sensor bias. This effect must be compensated using aileron command; therefore, the control authority on the roll channel will be affected by restricting the excursion of the cockpit controls by the amount $d_{aF}$ equal to what is necessary to command a roll rate equal to the bias:

\[ d_{aF} = F_1^{-1}(p_{\text{bias}}) \]  \hspace{1cm} (5.91)
The new equivalent range of $d_a$ is:

\[
d_{a\max F} = d_{a\max} - d_{aF}
\]

\[
d_{a\min F} = d_{a\min} - d_{aF}
\]

With the new range of lateral stick after failure, corresponding new ranges for all $v_E$ may be obtained from the 2-D projections.

### 5.3.2. Pitch Rate Sensor Bias

For $k_F = 19$, one can specify:

\[
\begin{align*}
\delta & = [\delta_1] = [q_{meas}], \text{ measured pitch rate} \\
E & = [E_1] = [d_e], \text{ longitudinal stick}
\end{align*}
\]

\[
\begin{bmatrix}
q_1 \\
v_e
\end{bmatrix} = [a_x \ a_z \ H \ M \ V \ q \ q \ \alpha \ \beta]
\]

The analysis of the pitch sensor failures follows the same approach as the roll sensor failures. At nominal conditions, the desired or reference pitch rate is:

\[
q_{\text{ref}} = F_1(d_e)
\]

where $F_1$ represents the pitch channel reference model, a second order transfer function. The pitch rate measurement is assumed to be perfect:

\[
q_{\text{meas}} = q_{\text{act}}
\]

and the commanded pitch rate is given by a control law of the form:

\[
q_{\text{cmd}} = F_2(q_{\text{ref}} - q_{\text{act}})
\]

At failure condition, the reference is unchanged; however:

\[
q_{\text{meas}F} = q_{\text{act}} + q_{\text{bias}}
\]

\[
q_{\text{cmd}F} = F_2(q_{\text{ref}} - q_{\text{act}} - q_{\text{bias}})
\]

As a result of the failure, an undesired additional pitch rate command is produced, which can be expressed with some assumptions, at steady state, as:

\[
\Delta q_{\text{cmd}} = q_{\text{cmd}F} - q_{\text{cmd}} = -q_{\text{bias}}
\]
In other words, the control laws will rotate the aircraft in the opposite direction of the sensor bias. This effect must be compensated using stabilator/elevator command; therefore, the control authority on the pitch channel will be affected by restricting the excursion of the cockpit controls by the amount $d_{eF}$ equal to what is necessary to command a pitch rate equal to the bias:

$$d_{eF} = F_1^{-1}(q_{bias}) \quad (5.101)$$

The new equivalent range of $d_e$ is:

$$d_{emaxF} = d_{emax} - d_{eF} \quad (5.102)$$

$$d_{eminF} = d_{emin} - d_{eF} \quad (5.103)$$

With the new range of longitudinal stick after failure, corresponding new ranges for all $v_E$ may be obtained from the 2-D projections.

### 5.3.3. Yaw Rate Sensor Bias

For $k_F = 2I$, the following parameters can be specified:

$$v_\delta = [v_\delta I] = [v_{meas}], \text{ measured yaw rate}$$

$$v_e = [v_e I] = [d_r], \text{ pedals} \quad (5.104)$$

$$v_E = [a_y \ p \ r \ \dot{p} \ \dot{r} \ \beta \ \varphi]$$

At nominal conditions, the desired or reference yaw rate is:

$$r_{ref} = F_1(d_r) \quad (5.105)$$

where $F_1$ represents the yaw channel reference model, a second order transfer function. The yaw rate measurement is assumed to be perfect:

$$r_{meas} = r_{act} \quad (5.106)$$

and the commanded yaw rate is given by a control law of the form:

$$r_{cmd} = F_2(r_{ref} - r_{act}) \quad (5.107)$$

At failure condition, the reference is unchanged; however:

$$r_{measF} = r_{act} + r_{bias} \quad (5.108)$$
As a result of the failure, an undesired additional yaw rate command is produced, which can be expressed with some assumptions, at steady state, as:

\[ \Delta r_{cmd} = r_{cmdF} - r_{cmd} = -r_{bias} \]  \hspace{1cm} (5.110)

In other words, the control laws will rotate the aircraft in the opposite direction of the sensor bias. This effect must be compensated using rudder command; therefore, the control authority on the yaw channel will be affected by restricting the excursion of the cockpit controls by the amount \( d_{rF} \) equal to what is necessary to command a yaw rate equal to the bias:

\[ d_{rF} = F^{-1}_1(r_{bias}) \]  \hspace{1cm} (5.111)

The new equivalent range of \( d_r \) is:

\[ d_{r \text{ max}} = d_{r \text{ max}} - d_{rF} \]  \hspace{1cm} (5.112)

\[ d_{r \text{ min}} = d_{r \text{ min}} - d_{rF} \]  \hspace{1cm} (5.113)

With the new range of pedals after failure, corresponding new ranges for all \( v_E \) may be obtained from the 2-D projections.
5.4. Estimation Algorithms for Structural Damage

5.4.1. Wing Damage

For $k_F = 23 \div 24$, the key parameters for AC indirect evaluation can be specified as:

$v_\delta = [v_{\delta 1} \ v_{\delta 2}] = [\Delta C_{LWL} \ \Delta C_{mWL}]$, lift and pitching moment alteration due to left wing failure

$v_e = [v_{e 1} \ v_{e 2} \ v_{e 3} \ v_{e 4}] = [d_\alpha \ d_q \ d_r \ d_T]$, lateral stick, longitudinal stick, pedals, throttle lever

$v_E = [H \ M \ V \ p \ q \ r \ a_x \ a_y \ a_z \ \dot{p} \ \dot{q} \ \dot{r} \ \alpha \ \beta \ \theta \ \phi] \tag{5.114}$

At normal trim conditions, the external forces acting on the aircraft are presented in Figure 5.2.

![Figure 5.2. Aircraft Forces in the Plane of Symmetry at Trim](image)

The balance equation can be expressed as:

\[
\begin{align*}
T + L \sin(\alpha) - D \cos(\alpha) - G \sin(\theta) &= 0 \\
- L \cos(\alpha) - D \sin(\alpha) + G \cos(\theta) &= 0 \\
M_{WF} &= -M_{HT} \\
C_L &= C_L(C_D) \\
C_{IWF} &= 0
\end{align*} \tag{5.115}
\]
Note that WF stands for “wing and fuselage” and HT for “horizontal tail”. Assuming that at post failure conditions the same flight condition is maintained, then:

\[ M_F = M, \quad H_F = H, \quad \gamma_F = \gamma \]  

(5.117)

The effect of the failure will consist of producing variations \( \Delta \)'s of all other variables. Therefore:

\[
\begin{align*}
T + \Delta T + (L + \Delta L) \sin(\alpha + \Delta \alpha) - (D + \Delta D) \cos(\alpha + \Delta \alpha) - G \sin(\theta + \Delta \theta) &= 0 \\
-(L + \Delta L) \cos(\alpha + \Delta \alpha) - (D + \Delta D) \sin(\alpha + \Delta \alpha) + G \cos(\theta + \Delta \theta) &= 0 \\
M_{WF} + \Delta M_{WF} &= -M_{HT} - \Delta M_{HT} \\
C_{LF} &= C_{LF}(C_D) \\
\Delta C_{lWF} &= -\Delta C_{LWL} \cdot l_{AL} \neq 0
\end{align*}
\]  

(5.118)

where \( l_{AL} \) is the distance to the plane of symmetry of the left wing aerodynamic center at post-failure conditions. Note that \( l_{AL} < 0 \), while \( l_{AR} > 0 \). The non-zero rolling moment must be compensated using aileron deflection, thus reducing the authority of the aileron command. The same concept of aileron “pseudo-failure” may be considered here as in the case of the stabilator failure. This pseudo-failure involves a non-zero aileron trim deflection at a position \( \delta_{aPF} \) necessary to compensate for the rolling effect of the damaged wing. This compensating aileron deflection can be approximated using the balance of the rolling moment:

\[
- \Delta C_{LWL} \cdot l_{AL} = -\varepsilon_a b C_l \delta_a \cdot \delta_{aPF}
\]  

(5.119)

\[
\delta_{aPF} = \frac{\Delta C_{LWL} \cdot l_{AL}}{\varepsilon_a b C_l \delta_a}
\]  

(5.120)

where \( b \) is the wingspan and \( \varepsilon_a \) is an alteration factor depending on how the left aileron is affected by the wing failure (\( \varepsilon_a = 1 \) if the aileron is not affected by the wing failure and \( \varepsilon_a = 0.5 \) if the aileron is completely out of work). For a linear relationship with the EDIV is:

\[ d_a = k_a \delta_a \]  

(5.121)

then the nominal lateral control range is:
\[ d_{a \text{ max}} = k_a \delta_{a \text{ max}} \]  
\[ d_{a \text{ min}} = k_a \delta_{a \text{ min}} \]  

With the damaged wing, the new range for the lateral control can be assumed to be:

\[ d_{a \text{ max}F} = k_a (\delta_{a \text{ max}} - \delta_{aPF}) \]  
\[ d_{a \text{ min}F} = k_a (\delta_{a \text{ min}} - \delta_{aPF}) \]

Note that this is equivalent to shifting the range of \( d_a \) in one direction or the other, while maintaining the same span:

\[ [d_{a \text{ min}F}, d_{a \text{ max}F}] \equiv [d_{a \text{ min}}, d_{a \text{ max}}] \]  
\[ d_{a \text{ max}F} - d_{a \text{ min}F} = d_{a \text{ max}} - d_{a \text{ min}} \]

The direction of the shift depends on the failed wing side and the sign of the lift alteration. However, it is reasonable to assume that \( \Delta C_{WL} < 0 \) since one would expect that wing damage would decrease the lift produced. Now the new range of \( d_a \) can be used with the respective projections to determine the effects of the wing failure on lateral-directional variables. This range can be expressed in terms of the failed surface as follows:

\[ d_{a \text{ max}F} = k_a (\delta_{a \text{ max}} - \frac{\Delta C_{WL} \cdot l_{AL}}{\varepsilon_a b C_i \delta_a}) \]  
\[ d_{a \text{ min}F} = k_a (\delta_{a \text{ min}} - \frac{\Delta C_{WL} \cdot l_{AL}}{\varepsilon_a b C_i \delta_a}) \]

Note that \( \Delta C_{WL} \), \( l_{AL} \), and \( \varepsilon_a \) must be known. How to adequately determine them must be further investigated. \( \Delta C_{WL} \) could be a measure of wing failure severity and determined as an outcome of the quantitative direct failure evaluation. The same observation applies to \( \Delta C_{mWL} \) as well. If this alteration is known, then the reduction of longitudinal control authority can be determined using the pitching moment balance, as follows:

\[ \Delta M_{WF} = -\Delta M_{HT} \]  
\[ \Delta C_{mWL} = -C_m \delta_{e} \cdot \Delta ePF \]
\[
\Delta \delta_{ePF} = -\frac{\Delta C_{mWL}}{C_m \delta_e}
\] (5.132)

With the same previous assumptions, the new equivalent range for \( d_e \) is:

\[
d_{e\text{max}} = d_{e\text{max}} - k_e \cdot (\Delta \delta_{ePF} + \delta_{\text{trim}})
\] (5.133)

\[
d_{e\text{min}} = d_{e\text{min}} - k_e \cdot (\Delta \delta_{ePF} + \delta_{\text{trim}})
\] (5.134)

The new equivalent range for \( d_e \) can be used to obtain, from the respective projections, the new ranges for longitudinal variables. To determine effects on \( H \), \( M \), and \( V \), we can consider the equations for the longitudinal force, the vertical force, and the polar:

\[
\begin{align*}
T + \Delta T + (L + \Delta L) \sin(\alpha) \cos(\Delta \alpha) + (L + \Delta L) \cos(\alpha) \sin(\Delta \alpha) - \\
- (D + \Delta D) \cos(\alpha) \cos(\Delta \alpha) + (D + \Delta D) \sin(\alpha) \sin(\Delta \alpha) - \\
- G \sin(\theta) \cos(\Delta \theta) - G \cos(\theta) \sin(\Delta \theta) &= 0 \\
- (L + \Delta L) \cos(\alpha) \cos(\Delta \alpha) + (L + \Delta L) \sin(\alpha) \sin(\Delta \alpha) - \\
- (D + \Delta D) \sin(\alpha) \cos(\Delta \alpha) - (D + \Delta D) \cos(\alpha) \sin(\Delta \alpha) + \\
+ G \cos(\theta) \cos(\Delta \theta) - G \sin(\theta) \sin(\Delta \theta) &= 0 \\
\Delta D &= C_{LF}^T (\Delta L)
\end{align*}
\] (5.135)

For keeping the same trajectory slope \( \gamma \), then \( \Delta \alpha = \Delta \theta \). \( \Delta L \) is known. If the polar \( C_{LF}(C_D) \) is known, then a system of 3 equations and 3 unknowns \( \Delta T \), \( \Delta D \), and \( \Delta \alpha \) is obtained. Solving for \( \Delta T \), allows determining new virtual ranges for \( d_T \). From respective projections, new ranges for \( M \), \( H \), and \( V \) can then be obtained. Note that the method derived here relies on several strong assumption. Therefore, the method applies for classes of ACs that satisfy these assumptions.

### 5.4.2. Horizontal Tail Damage

For \( k_F = 25 \div 26 \), one can specify:

\[
v_{\delta} = [v_{\delta}] = [\Delta C_{LHL}], \quad \text{lift alteration due to left horizontal tail damage}
\]

\[
v_{e} = [v_{eL} v_{e2}] = [d_a \ d_e], \quad \text{lateral stick and longitudinal stick}
\]

\[
v_{E} = [H \ M \ V \ p \ q \ r \ a_x \ a_y \ a_z \ \dot{p} \ \dot{q} \ \dot{r} \ \alpha \ \beta \ \theta \ \phi]
\] (5.136)

The assumption is made that the left elevator (stabilator) can be still deflected; however, the respective control derivative may be affected, such that:

\[
\Delta C_{LHL} = \Delta C_{LHL0} + \Delta C_{L\delta_e} \delta_e
\] (5.137)
It is assumed that both \( \Delta C_{L_{HL,0}} \) and \( \Delta C_{L\delta_e} \) are negative representing a reduction of the horizontal tail capability to produce lift. For longitudinal balance, this lift variation must be compensated by an additional elevator deflection \( \Delta \delta_e > 0 \) such that:

\[
- \Delta C_{L_{HL}} = \frac{C_L \delta_e}{2} - \Delta C_{L\delta_e} \gamma \delta_e + \frac{C_L \delta_e}{2} \Delta \delta_e \tag{5.138}
\]

\[
- \Delta C_{L_{HL}} = \frac{C_L \delta_e}{2} - \Delta C_{L\delta_e} \gamma \delta_e \tag{5.139}
\]

\[
\Delta \delta_e = - \frac{\Delta C_{L_{HL}}}{C_L \delta_e - \Delta C_{L\delta_e}} \tag{5.140}
\]

For lateral balance, a compensatory rolling moment must be produced using aileron, such that:

\[
\Delta C_{L_{HL}} \cdot \frac{l_{LHT}}{b} = C_{l\delta_a} \Delta \delta_a \tag{5.141}
\]

where \( l_{LHT} < 0 \) is the distance from the left horizontal tail aerodynamic center to the plane of symmetry. Note that \( l_{RHT} > 0 \). Then:

\[
\Delta \delta_a = \frac{\Delta C_{L_{HL}} \cdot l_{LHT}}{b C_{l\delta_a}} \tag{5.142}
\]

To estimate \( \Delta C_{L_{HL}} \) consider that:

\[
\Delta C_{L_{HL}} = \Delta C_{L_{HL, \text{BeforeFailure}}} - \Delta C_{L_{HL, \text{AfterFailure}}} =
= \Delta C_{L_{HL, \text{BeforeFailure}}} - sddeL \cdot \Delta C_{L_{HL, \text{BeforeFailure}}} =
= (1 - sddeL) \Delta C_{L_{HL, \text{BeforeFailure}}} =
= (1 - sddeL) \frac{C_L \delta_e}{2} \delta_e \tag{5.143}
\]

Therefore:
\[
\Delta \delta_e = -\frac{\Delta C_{L_{HL}}}{C_{L\delta_e} - \Delta C_{L\delta_e}} = -\frac{(1-sddeL)C_{L\delta_e}}{2} \delta_e
\]

\[
= -\frac{(1-sddeL)C_{L\delta_e}}{C_{L\delta_e} (\frac{1}{2} + sddeL/2)} = -\frac{(1-sddeL)\delta_e}{1+sddeL}
\]

(5.144)

The value of \( l_{LHT} \) can be approximated as:

\[
l_{LHT} = 2 \frac{C_{l\delta_e}L}{C_{L\delta_e}}
\]

(5.145)

The relationships between DIV and EDIV are:

\[
d_e = f_{e\delta 1}(\delta_e) \quad \text{and} \quad d_a = f_{e\delta 2}(\delta_a)
\]

(5.146)

or

\[
d_e = k_e \delta_e \quad \text{and} \quad d_a = k_a \delta_a
\]

(5.147)

After failure, the ranges for \( d_e \) and \( d_a \) become:

\[
d_{e\max F} = d_{e\max} - k_e \Delta \delta_e \quad \text{and} \quad d_{e\min F} = d_{e\min} - k_e \Delta \delta_e
\]

(5.148)

\[
d_{a\max F} = d_{a\max} - k_a \Delta \delta_a \quad \text{and} \quad d_{a\min F} = d_{a\min} - k_a \Delta \delta_a
\]

(5.149)

It is postulated that new ranges for longitudinal variables may be obtained from corresponding 2-D projections of \( d_e \). Similarly, new ranges for lateral-directional variables may be obtained from corresponding 2-D projections of \( d_a \).

### 5.4.3. Vertical Tail Damage

For \( k_F = 27 \div 28 \), the key parameters can be specified as:

\[
\nu_\delta = [v_\delta] = [\Delta C_{Y_{VL}}], \quad \text{lift (lateral force) alteration due to left vertical tail damage}
\]

\[
\nu_e = [v_e \nu_e 2] = [d_a \ d_T], \quad \text{lateral stick and pedals}
\]

(5.150)

\[
\nu_E = [p \ r \ a_y \ p \ i \ \beta \ \phi]
\]

It is assumed that the left rudder can be still deflected; however, the respective control derivative may be affected, such that:
\[ \Delta C_{Y_{VL}} = \Delta C_{Y_{VL0}} + \Delta C_{Y\delta_r} \delta_r \]  

(5.151)

For directional balance, this lift variation must be compensated by an additional rudder deflection \( \Delta \delta_r \) such that:

\[ -\Delta C_{Y_{VL}} = \frac{C_{Y\delta_r}}{2} \mu \delta_r + \frac{C_{Y\delta_r}}{2} \Delta \delta_r \]  

(5.152)

\[ -\Delta C_{Y_{VL}} = C_{Y\delta_r} \mu \delta_r \]  

(5.153)

\[ \Delta \delta_r = -\frac{\Delta C_{Y_{VL}}}{C_{Y\delta_r} - \Delta C_{Y\delta_r}} \]  

(5.154)

For roll balance, a compensatory rolling moment must be produced using aileron, such that:

\[ \Delta C_{Y_{VL}} \cdot \frac{h_{VT}}{b} = C_{l\delta_a} \Delta \delta_a \]  

(5.155)

where \( h_{VT} < 0 \) is the distance from the vertical tail aerodynamic center to the body horizontal plane. Note that \( h_{VT} \) is normally the same for L and R. Then:

\[ \Delta \delta_a = \frac{\Delta C_{Y_{VL}} \cdot h_{VT}}{bC_{l\delta_a}} \]  

(5.156)

The relationships between DIV and EDIV are:

\[ d_a = f_{a\delta_1}(\delta_a) \quad \text{and} \quad d_r = f_{a\delta_2}(\delta_r) \]  

(5.157)

or

\[ d_a = k_a \delta_a \quad \text{and} \quad d_r = k_r \delta_r \]  

(5.158)

After failure, the ranges for \( d_r \) and \( d_a \) become:

\[ d_{r_{maxF}} = d_{r_{max}} - k_r \Delta \delta_r \quad \text{and} \quad d_{r_{minF}} = d_{r_{min}} - k_r \Delta \delta_r \]  

(5.159)

\[ d_{a_{maxF}} = d_{a_{max}} - k_a \Delta \delta_a \quad \text{and} \quad d_{a_{minF}} = d_{a_{min}} - k_a \Delta \delta_a \]  

(5.160)

It is postulated that new ranges for directional channel variables may be obtained from corresponding 2-D projections of \( d_r \). Similarly, new ranges for lateral channel variables may be obtained from corresponding 2-D projections of \( d_a \). This hypothesis requires further investigation given the typical coupling between the two dynamic channels.
5.5. Estimation Algorithms for Propulsion AC

This discussion covers failure $k_F = 29$ and $k_F = 30$ of Table 1.2. It makes reference to the left engine; however, everything applies similarly to the right engine failure. For $k_F = 29 \div 30$, it can be specify that:

$$
\begin{align*}
\nu_\delta &= [\nu_\delta I] = [T_L], \text{ left (or right) engine thrust} \\
\nu_e &= [\nu_e I] = [d_T], \text{ collective throttle (is assumed)} \\
\nu_E &= [\nu E I] = [a_x \ a_z \ r \ H \ M \ V \ \alpha \ \theta]
\end{align*}
$$

(5.161)

The relationship between DIV and EDIV is:

$$
d_T = f_{\varepsilon \delta I}(T_L + T_R)
$$

(5.162)

The nominal range for $d_T$ is typically $d_T \in [0, \ 100\%]$, with:

$$
d_{T\text{max}} = f_{\varepsilon \delta I}(T_{L\text{max}} + T_{R\text{max}}) = f_{\varepsilon \delta I}(2T_{\text{max}}) = 100\% 
$$

(5.163)

and

$$
d_{T\text{min}} = f_{\varepsilon \delta I}(T_{L\text{min}} + T_{R\text{min}}) = f_{\varepsilon \delta I}(2T_{\text{min}}) = 0\%
$$

(5.164)

The failure consists of changing the range of $T_L$ to $T_{L\text{min}}$ and $T_{L\text{max}}$ and changing the function $f_{\varepsilon \delta I}$ accordingly. After failure, the equivalent range of $d_T$ can be defined as:

$$
d_{T\text{max}} = f_{\varepsilon \delta I}(T_{L\text{max}} + T_{R\text{max}})
$$

(5.165)

and

$$
d_{T\text{min}} = f_{\varepsilon \delta I}(T_{L\text{min}} + T_{R\text{min}})
$$

(5.166)

Most often, $T_{L\text{min}} = 0$ and $T_{L\text{max}} < T_{\text{max}}$. The values of $T_{L\text{min}}$ and $T_{L\text{max}}$ must be determine through direct quantitative evaluation. Also, $f_{\varepsilon \delta I}$ is assumed known; it can be approximated as a linear relationship. Note also that the equivalent range at post failure conditions ([$d_{T\text{min}}$, $d_{T\text{max}}$]) is invariable with respect to the surface failed (L or R) and that $[d_{T\text{min}}$, $d_{T\text{max}}$] = [0, 100]. With the new range after failure, new ranges for all $\nu_E$ may be obtained from the 2-D projections, except for $r$ (see discussion for throttle failure). An undesired yawing moment (depending on the distance of the engines with respect to the aircraft centerline) may be produced, which needs to be compensated for by using rudder deflection. This will reduce the overall control authority on the yaw channel. A similar approach as in the case of the locked throttle can be applied to handle this case. However, this time, the pseudo-failure of the rudder consists of a variable trim deflection depending on $d_T$.

Let us assume linear relationships at normal conditions between pilot throttle input and total thrust as well as between pilot throttle input and fuel valve deflection. Therefore, the total yawing moment is:

$$
C_{n\delta TL} \cdot \delta TL + C_{n\delta TR} \cdot \delta TR = 0
$$

(5.167)
or \[(C_n\delta_{TL} + C_n\delta_{TR}) \cdot d_T = 0\] (5.168)

where, of course, \(C_n\delta_{TL} = -C_n\delta_{TR} > 0\). At post failure conditions, the healthy engine control derivative remains unchanged, while the failed engine control derivative is affected:\[(C_n\delta_{TLF} + C_n\delta_{TR}) \cdot d_T \neq 0\] (5.169)

For \(l_{eL} = -l_{eR}\):
\[(C_n\delta_{TLF} + C_n\delta_{TR}) \cdot d_T = \frac{-l_{eL}(T_L - T_R)}{P/SbV^2} \] (5.170)
\[C_n\delta_{TLF} \cdot d_T = \frac{-l_{eL}T_L}{P/2SbV^2} \] (5.171)

Assuming a linear relationship and \(T_{L\min_{PF}} = 0\):
\[C_n\delta_{TLF} = \frac{-l_{eL} \cdot T_{L\max_{PF}}}{P/2SbV^2 d_{T\max}} \] (5.172)

where \(l_{eL}\) is the distance from the aircraft center of gravity to the axis of the left engine (negative for L, positive for R). The yawing moment is compensated by the rudder such that:
\[(C_n\delta_{TLF} + C_n\delta_{TR}) \cdot d_T = -C_n\delta_r \cdot \delta_{rPF} \] (5.173)

and
\[\delta_{rPF} (d_T) = -\frac{C_n\delta_{TLF} + C_n\delta_{TR}}{C_n\delta_r} d_T \] (5.174)

The relationship with the EDIV for this pseudo-failure is:
\[d_r = k_r\delta_r \] (5.175)

The nominal directional control range is:
\[d_{r\max} = k_r\delta_{r\max} \] (5.176)
\[d_{r\min} = k_r\delta_{r\min} \] (5.177)
Under the loss of efficiency failure, the new range for the directional control is:

\[ d_{r_{\text{max}}} F(d_T) = k_r (\delta_{r_{\text{max}}} - \delta_{r_{PF}}(d_T)) \]  \hspace{1cm} (5.178)

\[ d_{r_{\text{min}}} F(d_T) = k_r (\delta_{r_{\text{min}}} - \delta_{r_{PF}}(d_T)) \]  \hspace{1cm} (5.179)

Note that this effect is similar to the throttle failure. The new range of \( d_r \) can be used with the \((d_r, r)\) projection to determine the effects on \( r \) of the engine loss of efficiency failure.
5.6. Performance of Flight Envelope Assessment Under AC

5.6.1. Evaluation Metrics for Flight Envelope Assessment

Assumed nominal ranges for flight envelope relevant variables (ERV) have been determined based on simulator tests. Verification tests under AC have been performed for each specific subsystem failure attempting to achieve the same tasks and level of maneuverability as in the nominal tests.

To achieve desirable performance of the post-failure envelope prediction methodology, data points acquired through properly designed verification tests under abnormal conditions must stay within the predicted range and be close to the predicted limits. Therefore, for performance evaluation, three metrics have been defined attempting to capture the level of prediction confidence, the level of possible range exceedance, and the level of predicted range conservativeness. The first evaluation criterion is expressed by a prediction rate (PR) calculated as the percentage of all verification data points $ERV_{val}$ that fall inside the predicted range. If the predicted range of any $ERV$ is denoted as $R_{ERV} = [ERV_{min}, ERV_{max}]$, then the total number of validation points $N_V$ is the sum of points inside ($N_R$) and outside ($N_{\bar{R}}$) the predicted range:

$$N_V = N_R + N_{\bar{R}}$$  \hspace{1cm} (5.180)

The prediction rate is then calculated as:

$$PR = \frac{N_R}{N_V} \cdot 100$$  \hspace{1cm} (5.181)

The level of possible range exceedance is assessed as the relative maximum amount by which points in the verification data set exceed the predicted range. The predicted range exceedance index (REI) is defined as:

$$REI = \frac{\max\{\min(ERV_{val} - ERV_{max}, ERV_{val} - ERV_{min})\}}{ERV_{max} - ERV_{min}} \cdot 100$$  \hspace{1cm} (5.182)

Obviously, it is desirable to have large values of $PR$ and low values of $REI$. However, it should be noted that, if the predicted range is exceedingly/unrealistically large, then $PR = 100\%$ and $REI = 0\%$ without representing good performance. Therefore, the values of the previous two evaluation metrics must be considered in conjunction with a margin index (MI) that is expected to capture the level of predicted range conservativeness. MI is defined as:

$$MI = \frac{\max\{\min(ERV_{val} - ERV_{max}, ERV_{val} - ERV_{min})\}}{ERV_{max} - ERV_{min}} \cdot 100$$  \hspace{1cm} (5.183)
It should be also noted that, if the verification test does not cover the nominal range of the tests used for self generation, then high values of MI do not necessarily indicate a flaw in the post-failure range prediction.

5.6.2. Flight Envelope Assessment Under Actuator Failure

Table 5.1 presents the predicted post-failure flight envelope ranges under an elevator failure along with the corresponding nominal ranges and the performance metrics defined above. Figure 5.3 shows an example of a 2-D projection in the AIS WVU Self Viewer (the 2-D analysis tool developed for the purpose of this project) with the verification test points and the corresponding predicted post-failure ranges represented by the vertical and horizontal lines. The failure considered consists of left stab locked at 8 degrees. Two different verification tests have been performed covering the altitude range of the nominal tests for Mach between 0.7 and 0.9. The results show alterations of the ranges of pitch rate, pitch acceleration, and roll rate as expected. Note that the reduced range of Mach and altitude considered when building the self is not affected by this elevator failure. The verification points are well within predicted ranges for all variables considered. Only very few points exceed the predicted ranges by small relative amounts, as captured by REI, except for pitch acceleration. The variation of this variable experiences a short-duration sudden peak at the failure occurrence moment. The duration of the peak is very short as shown by the values of the pitch rate, which remain practically within predicted range. The low values of MI show that the predicted limits of the respective variables are not unrealistically large. It should be noted that the large MI value recorded for Mach number is due to the fact that the verification tests do not cover the range between 0.5 and 0.7, which was included in the self. This is one example of a situation when large values of MI reflect the incompleteness of the verification tests rather than flaws in the range prediction process.

<table>
<thead>
<tr>
<th>ERV</th>
<th>Nominal Range</th>
<th>Post-Failure Range</th>
<th>PR. %</th>
<th>REI. %</th>
<th>MI. %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H$</td>
<td>[2012,9929]</td>
<td>[2012,9929]</td>
<td>100.0</td>
<td>0.0</td>
<td>8.0</td>
</tr>
<tr>
<td>$M$</td>
<td>[0.526,0.95]</td>
<td>[0.526,0.95]</td>
<td>99.9</td>
<td>1.6</td>
<td>32.9</td>
</tr>
<tr>
<td>$q$</td>
<td>[–0.225,0.183]</td>
<td>[–0.141,0.105]</td>
<td>100.0</td>
<td>2.6</td>
<td>0.0</td>
</tr>
<tr>
<td>$\dot{q}$</td>
<td>[–0.383,0.386]</td>
<td>[–0.23,0.32]</td>
<td>99.7</td>
<td>24.6</td>
<td>0.0</td>
</tr>
<tr>
<td>$p$</td>
<td>[–0.94,1.08]</td>
<td>[–0.90,0.79]</td>
<td>100.0</td>
<td>0.0</td>
<td>1.0</td>
</tr>
</tbody>
</table>

The predicted post-failure flight envelope ranges under the aileron failure are presented in Table 5.2. Corresponding nominal ranges and values of the three performance metrics are included. The failure consists of a right aileron locked at 8°. Two different verification tests have been performed covering an altitude range between 2,000 m and 10,000 m and a Mach number range between 0.7 and 0.9. The results show alterations of the ranges of roll rate and roll acceleration, as expected. The verification points are all within predicted ranges for both variables considered with only insignificant exceedence as indicated by REI. The low values of REI and MI indicate that the predicted ranges are not unrealistically large.

The predicted post-failure flight envelope ranges under the rudder failure are presented in Table 5.3. The failure consists of a right rudder locked at 8°. Two different verification tests have been performed covering the altitude and Mach ranges of the nominal tests. The results
show alterations of the ranges of yaw rate and yaw acceleration, as expected. The verification points are all within predicted ranges for both variables considered. The relatively low values of MI lead to the conclusion that the predicted limits of the respective variables are not unrealistically large.

![Figure 5.3. Post-Failure Range of the Pitch Rate Under Stabilator Failure](image)

**Table 5.2. Performance of Flight Envelope Prediction Under Aileron Failure**

<table>
<thead>
<tr>
<th>ERV</th>
<th>Nominal Range</th>
<th>Post-Failure Range</th>
<th>PR, %</th>
<th>REI, %</th>
<th>MI, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta$</td>
<td>$[-0.94, 1.08]$</td>
<td>$[-0.77, 1.01]$</td>
<td>99.94</td>
<td>5.6</td>
<td>1.2</td>
</tr>
<tr>
<td>$\dot{\beta}$</td>
<td>$[-2.95, 3.1]$</td>
<td>$[-2.98, 2.94]$</td>
<td>99.98</td>
<td>1.5</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Table 5.3. Performance of Flight Envelope Prediction Under Rudder Failure**

<table>
<thead>
<tr>
<th>ERV</th>
<th>Nominal Range</th>
<th>Post-Failure Range</th>
<th>PR, %</th>
<th>REI, %</th>
<th>MI, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r$</td>
<td>$[-0.064, 0.062]$</td>
<td>$[-0.047, 0.047]$</td>
<td>100.0</td>
<td>0.0</td>
<td>7.8</td>
</tr>
<tr>
<td>$\dot{r}$</td>
<td>$[-0.117, 0.126]$</td>
<td>$[-0.114, 0.104]$</td>
<td>100.0</td>
<td>0.0</td>
<td>10.4</td>
</tr>
</tbody>
</table>

5.6.3. **Flight Envelope Assessment Under Sensor Failure**

Table 5.4 presents the predicted post-failure flight envelope ranges under the roll rate sensor failure. The failure consists of a 5°/sec bias in the roll rate sensor output. Two different verification tests have been performed covering the entire altitude and Mach ranges of the nominal tests. The results show alterations of the ranges of roll rate and roll acceleration, as expected. The verification points are all within predicted ranges for all variables considered with only insignificant exceedance in the roll attitude angle, as reflected by the corresponding REI value. The relatively low values of MI show that the predicted limits of the respective variables are not unrealistically large.
Table 5.4. Performance of Flight Envelope Prediction Under Roll Rate Sensor Failure

<table>
<thead>
<tr>
<th>ERV</th>
<th>Nominal Range</th>
<th>Post-Failure Range</th>
<th>PR, %</th>
<th>REI, %</th>
<th>MI, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$</td>
<td>$[-0.94, 1.08]$</td>
<td>$[-0.82, 1.1]$</td>
<td>100.0</td>
<td>0.0</td>
<td>15.3</td>
</tr>
<tr>
<td>$\dot{p}$</td>
<td>$[-2.95, 3.1]$</td>
<td>$[-1.8, 3.13]$</td>
<td>100.0</td>
<td>0.0</td>
<td>15.5</td>
</tr>
</tbody>
</table>

Listed in Table 5.5 are the predicted post-failure flight envelope ranges under the pitch rate sensor failure. The failure consists of a 5°/sec bias in the pitch rate sensor output. Four different verification tests have been performed covering the entire altitude and Mach ranges of the nominal tests. The results show alterations of the ranges of pitch rate and pitch acceleration, as expected. The high values of $PR$ show that the majority of the verification points are within the predicted ranges for all variables considered. Although the values of $REI$ for the pitch acceleration and pitch attitude angle are relatively high, very few verification points exceed the corresponding predicted limits, as indicated by the corresponding $PR$ values. The relatively low values of $MI$ lead to the conclusion that the predicted limits of the respective variables are not unrealistically large.

Table 5.5. Performance of Flight Envelope Prediction Under Pitch Rate Sensor Failure

<table>
<thead>
<tr>
<th>ERV</th>
<th>Nominal Range</th>
<th>Post-Failure Range</th>
<th>PR, %</th>
<th>REI, %</th>
<th>MI, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q$</td>
<td>$[-0.225, 0.183]$</td>
<td>$[-0.364, 0.153]$</td>
<td>99.8</td>
<td>8.7</td>
<td>13.0</td>
</tr>
<tr>
<td>$\dot{q}$</td>
<td>$[-0.383, 0.386]$</td>
<td>$[-0.417, 0.32]$</td>
<td>99.9</td>
<td>30.9</td>
<td>6.0</td>
</tr>
</tbody>
</table>

The predicted post-failure flight envelope ranges under the yaw rate sensor failure are listed in Table 5.6. The failure consists of a 5°/sec bias in the yaw rate sensor output. Four different verification tests have been performed covering the entire altitude and Mach ranges of the nominal tests. The results show alterations of the ranges of yaw rate and yaw acceleration, as expected. Although a rudder failure would be expected to affect sideslip capability, this effect is not captured because the sideslip range considered for normal operation was already limited within the capability under failure of such low severity. The verification points are all within predicted ranges for all variables considered with only insignificant exceedance as reflected by $REI$. The simultaneous occurrence of non-zero values of $REI$ and $MI$ reflect the situation when one limit of the predicted range is exceeded and the other is not reached during the verification tests. The relatively low values of $MI$ lead to the conclusion that the predicted limits of the respective variables are not unrealistically large.

Table 5.6. Performance of Flight Envelope Prediction Under Yaw Rate Sensor Failure

<table>
<thead>
<tr>
<th>ERV</th>
<th>Nominal Range</th>
<th>Post-Failure Range</th>
<th>PR, %</th>
<th>REI, %</th>
<th>MI, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r$</td>
<td>$[-0.064, 0.062]$</td>
<td>$[-0.08, 0.0477]$</td>
<td>100.0</td>
<td>1.0</td>
<td>10.7</td>
</tr>
<tr>
<td>$\dot{r}$</td>
<td>$[-0.117, 0.126]$</td>
<td>$[-0.093, 0.128]$</td>
<td>100.0</td>
<td>4.8</td>
<td>5.8</td>
</tr>
<tr>
<td>$\beta$</td>
<td>$[-9.6, 9.8]$</td>
<td>$[-9.6, 9.8]$</td>
<td>100.0</td>
<td>0.7</td>
<td>16.9</td>
</tr>
</tbody>
</table>

5.6.4. Flight Envelope Assessment Under Structural Damage

The predicted post-failure flight envelope ranges under a wing damage affecting 15% of the left surface are listed in Table 5.7. Two verification tests have been performed covering the altitude and Mach number ranges of the nominal tests. The results show alterations of the ranges
of pitch rate, pitch acceleration, roll rate, and roll acceleration, as expected. Note that the reduced range of Mach and altitude considered when building the self is not affected by this failure. The verification points are well within the predicted ranges for all variables considered. Only very few points exceed the predicted ranges by small relative amounts, as captured by REI. The relatively low values of MI lead to the conclusion that the predicted limits of the respective variables are not unrealistically large.

Table 5.7. Performance of Flight Envelope Prediction Under Wing Damage

<table>
<thead>
<tr>
<th>ERV</th>
<th>Nominal Range</th>
<th>Post-Failure Range</th>
<th>PR, %</th>
<th>REI, %</th>
<th>MI, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H$</td>
<td>[2012, 9929]</td>
<td>[2012, 9929]</td>
<td>99.7</td>
<td>1.8</td>
<td>9.0</td>
</tr>
<tr>
<td>$M$</td>
<td>[0.526, 0.95]</td>
<td>[0.526, 0.95]</td>
<td>99.9</td>
<td>2.2</td>
<td>10.3</td>
</tr>
<tr>
<td>$\rho$</td>
<td>[-0.94, 1.08]</td>
<td>[-1.12, 0.51]</td>
<td>99.9</td>
<td>4.3</td>
<td>7.6</td>
</tr>
<tr>
<td>$\dot{\rho}$</td>
<td>[-2.95, 3.1]</td>
<td>[-2.97, 2.5]</td>
<td>99.9</td>
<td>2.5</td>
<td>0.0</td>
</tr>
<tr>
<td>$q$</td>
<td>[-0.225, 0.183]</td>
<td>[-0.05, 0.064]</td>
<td>100.0</td>
<td>0.0</td>
<td>1.2</td>
</tr>
<tr>
<td>$\dot{q}$</td>
<td>[-0.383, 0.386]</td>
<td>[-0.091, 0.072]</td>
<td>99.8</td>
<td>9.7</td>
<td>3.7</td>
</tr>
</tbody>
</table>

Table 5.8. presents the predicted post-failure flight envelope ranges under a horizontal tail damage affecting 60% of the left surface. One verification test has been performed covering an altitude range between 6,000 m and 10,000 m and a Mach number range between 0.7 and 0.9. The results show alterations of the ranges of pitch rate, pitch acceleration, roll rate, and roll acceleration, as expected. Note that the reduced range of Mach and altitude considered when building the self is not affected by this horizontal tail damage. The verification points are well within the predicted ranges for all variables considered. Only very few points exceed the predicted ranges by small relative amounts, as captured by REI. Relatively large values of MI are recorded, which seem to indicate that the maneuvers performed were not as aggressive as for the nominal conditions. Note that the ranges of Mach and altitude for the verification test were limited, explaining the large values of MI for these two variables.

Table 5.8. Performance of Flight Envelope Prediction Under Horizontal Tail Damage

<table>
<thead>
<tr>
<th>ERV</th>
<th>Nominal Range</th>
<th>Post-Failure Range</th>
<th>PR, %</th>
<th>REI, %</th>
<th>MI, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H$</td>
<td>[2012, 9929]</td>
<td>[2012, 9929]</td>
<td>99.0</td>
<td>2.0</td>
<td>40.3</td>
</tr>
<tr>
<td>$M$</td>
<td>[0.526, 0.95]</td>
<td>[0.526, 0.95]</td>
<td>98.5</td>
<td>0.8</td>
<td>35.5</td>
</tr>
<tr>
<td>$\dot{q}$</td>
<td>[-0.225, 0.183]</td>
<td>[-0.327, 0.165]</td>
<td>100.0</td>
<td>0.0</td>
<td>38.2</td>
</tr>
<tr>
<td>$\dot{q}$</td>
<td>[-0.383, 0.386]</td>
<td>[-0.37, 0.355]</td>
<td>100.0</td>
<td>0.0</td>
<td>9.1</td>
</tr>
<tr>
<td>$\rho$</td>
<td>[-0.94, 1.08]</td>
<td>[-0.925, 1.09]</td>
<td>100.0</td>
<td>0.0</td>
<td>18.1</td>
</tr>
<tr>
<td>$\dot{\rho}$</td>
<td>[-2.95, 3.1]</td>
<td>[-2, 3.13]</td>
<td>100.0</td>
<td>0.0</td>
<td>24.5</td>
</tr>
</tbody>
</table>

5.6.5. Flight Envelope Assessment Under Propulsion System Failure

Table 5.9 presents the predicted post-failure ranges of relevant variables with left engine throttle locked at idle. Corresponding nominal ranges and the three performance metrics are also included. One verification test has been performed covering an altitude range between 6,000 m and 10,000 m for Mach numbers between 0.7 and 0.9. The results show alterations of the ranges of the longitudinal and vertical acceleration, and roll rate, as expected. Note that the reduced
range of Mach and altitude considered when building the self is not affected by this failure. The verification points are well within predicted ranges for all variables considered. Only few points exceed the predicted ranges by small relative amounts, as captured by REI. The relatively large values of MI are the result of the verification test not completely covering the maneuver range of the self-generating data. In particular, the verification test does not include maneuvers between 2,000 m and 6,000 m or for Mach numbers between 0.5 and 0.7.

Table 5.9. Performance of Flight Envelope Prediction Under Engine Failure

<table>
<thead>
<tr>
<th>ERV</th>
<th>Nominal Range</th>
<th>Post-Failure Range</th>
<th>PR, %</th>
<th>REI, %</th>
<th>MI, %</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>$[-0.344, 0.597]$</td>
<td>$[-0.344, 0.353]$</td>
<td>99.9</td>
<td>4.8</td>
<td>20.4</td>
</tr>
<tr>
<td>$\alpha_z$</td>
<td>$[-2.46, 3.05]$</td>
<td>$[-2.46, 2.16]$</td>
<td>99.6</td>
<td>10.3</td>
<td>10.4</td>
</tr>
<tr>
<td>$r$</td>
<td>$[-0.0637, 0.0621]$</td>
<td>$[-0.0637, 0.0393]$</td>
<td>99.9</td>
<td>5.3</td>
<td>21.5</td>
</tr>
<tr>
<td>$H$</td>
<td>$[2012, 9929]$</td>
<td>$[2012, 9929]$</td>
<td>95.0</td>
<td>3.4</td>
<td>50.0</td>
</tr>
<tr>
<td>$M$</td>
<td>$[0.526, 0.95]$</td>
<td>$[0.526, 0.95]$</td>
<td>100.0</td>
<td>0.0</td>
<td>54.1</td>
</tr>
</tbody>
</table>
Chapter 6

Investigation of Immunity-based Accommodation of Aircraft Subsystem Abnormal Conditions

6.1. Immunity Approaches for AC Accommodation

The immunity-based AC accommodation can be approached based on two concepts. First, the biological feedback that establishes a balance between the activation and suppression of the antibodies generation can be converted into an adaptive mechanism augmenting a baseline controller \(^{20,21}\). For the biological system, it can be considered that the activation \(a_{Th}\) of the helper T-cells is governed by the balance between the amount of antigens \(a(t)\) and the action of antibodies \(u\) generated previously at \(t - \tau\) and whose antiseptic action is modeled by a function \(F_1\); therefore:

\[
a_{Th}(t) = a(t) - F_1(u(t - \tau)) \tag{6.1}
\]

The delay \(\tau\) models the time needed for the antibodies to mature and function. The generation of the helper T-cells \((T_h)\) depends on the activation \(a_{Th}\):

\[
T_h(t) = F_2(a_{Th}(t)), \quad \frac{dF_2}{da_{Th}} > 0 \tag{6.2}
\]

The production of the B-cells is the result of the combined effects of helper T-cells and suppressor T-cells:

\[
B(t) = F_3(T_h(t) - T_s(t)) \tag{6.3}
\]

The suppressor T-cell generation depends on the balance between the amount of antigens and antibodies and the rate at which the currently matured antibodies were generated \(u'(t - \tau)\):

\[
T_s(t) = K_sF_4(u'(t - \tau))a_{Th}(t) \tag{6.4}
\]

Finally, the antibodies are generated by activated B-cell; therefore:

\[
u(t) = F_5(B(t)) \tag{6.5}
\]

The immune system mechanisms are not completely understood; therefore, the parameters of this model, the functions \(F_1\) through \(F_5\), \(\tau\) and \(K_s\) must be determined heuristically, which is not trivial. The block diagram of the immune system feedback model is presented in Figure 6.1.
The second conceptual approach for using the AIS paradigm for control purposes relies on the assumption that the classification capabilities of the AIS can be extended and used not only to detect the problem (AC) but also to select or find the solution, which is mitigating the AC effects. The control action is considered as a mapping between control variables and controlled variables under specific performance requirements or constraints. Therefore, if the AIS is to be used for control purposes, both control and controlled variables must be part of the feature set. If that is the case, the AIS framework provides the relationship between the control and controlled variables for the normal conditions and for those AC for which detailed structuring of the non-self is available. Alternatively, the control variables associated with the adaptive immune feedback system and with the best compensation performance under specific upset conditions may be differentiated into memory cells, which can mount a faster and more aggressive secondary response in future encounters with the same AC.

Let us assume that we try to control variable $y$ using the control variable $x$, which are both part of the feature set. Let us also assume that structuring of the non-self is available such that a mapping between $x$ and $y$ is established under AC. For any desired value of the controlled variable $y_d$, corresponding clusters $c_k$ can be determined/extracted from the structured non-self as illustrated in Figure 6.2. Further processing of the clusters $c_k$ is necessary to produce compensatory commands $x_{cmd}$. These commands can be typically obtained by considering additional dimensions of the feature hyper-space, which may eventually result in one command detector, let us say $c_K$. An operator may still be needed to convert the command detector/cluster into a numerical value $x_{cmd}$, which can be formulated as:

$$x_{cmd} = O(c_K)$$  \hspace{1cm} (6.6)

Depending on the resolution of the non-self (in other words, the size of the detectors/compensators) the operator $O(*)$ can simply represent the center of the compensator or a more involved computation providing values inside the compensator.
Figure 6.2. Failure Accommodation Based on Structured Non-Self
6.2. The Humoral Immune Mechanism for Abnormal Condition Accommodation

6.2.1. General Control Architecture

A novel approach for direct adaptive control laws was investigated based on the humoral immune mechanism. Specifically, the proposed adaptive element is used to augment a baseline controller providing improved capabilities for handling subsystem ACs. The general block diagram of the humoral immune mechanism is presented in Figure 6.3. Three control architectures have been considered and compared in this study: a baseline controller (without adaptation), baseline controller augmented with ANN, and the baseline controller augmented with the artificial immune system-based mechanism.

![Figure 6.3. Humoral Immune System Representation](image)

The baseline controller relies primarily on a model reference architecture that uses the pilot stick inputs to generate desired angular rate and angular acceleration commands that are then used to produce compensating commands based on a proportional-integral-derivative (PID) approach in conjunction with approximate non-linear dynamic inversion (NLDI). The structure of the baseline control architecture\(^{24}\) is presented in Figure 6.4.

A second architecture considered here consists of the baseline control laws augmented with ANN. The neural networks are used in conjunction with the output from the PID compensation, the states of the aircraft and the reference angular rates and accelerations\(^{24}\), as shown in the control architecture of Figure 6.5. The input to the ANN is \(x = [V H \alpha \beta \omega \omega_{\text{ref}} \bar{U}_{\omega}]^T\), where \(V\) is the aircraft velocity, \(H\) is the altitude, \(\alpha\) is the angle of attack, and \(\beta\) is the sideslip angle. These four inputs are the same on all three channels. For the longitudinal channel, \(\omega = q\), while for the lateral and directional channels, \(\omega = [p r]\). Only on-channel variables are considered for \(\omega_{\text{ref}}\) and \(\bar{U}_{\omega}\). Finally, \(\bar{U}_{\omega}\) is defined as \(\bar{U}_{\omega} = \frac{1-e^{\omega(y-U_{\omega})}}{1+e^{\omega(y-U_{\omega})}}\), with \(y\), the previous output of the ANN.
Finally, the AIS is used to augment the PID part of the control laws. The augmented form of the controller using the AIS paradigm leads to the following expression in its discrete version:

\[
    u(k) = k_p \left( 1 + \frac{k_i}{z-1} + k_d \frac{z-1}{z} \right) \left[ 1 - \eta f(\Delta u(k)) \right] e(k)
\]

The function that expresses the change in concentration of the B cells will be given by:

\[
    f(\Delta u(k)) = 1 - \frac{2}{e^{\Delta u(k)^2} + e^{-\Delta u(k)^2}}
\]

where \(a\) is another parameter that will modulate the adequate response of the system under abnormal conditions.

**Figure 6.4.** General Block Diagram of the Baseline Controller

**Figure 6.5** General Block Diagram of the Baseline controller (PID+NLDI) Augmented with ANN
The following relationships are satisfied by definition for the parameters $k_p$, $k_i$, and $k_d$ and $\eta$:

$$(k_p, k_i, k_d) > 0, \text{ and } \eta \geq 0$$

(6.9)

Since the control laws perform negative feedback control action if $0 < \eta \Delta u(k) \leq 1$, and positive feedback control action if $1 < \eta \Delta u(k)$, an upper limit of the factor $\eta$ keeps the control system stable. Assuming the existence of parameters $k_{p0}$, $k_{i0}$, and $k_{d0}$, which ensure the stability of the control system with the conventional PID controller, the stability condition of the immune control algorithm is given by:

$$0 < k_p \leq k_{p0}, \quad 0 < k_p \leq k_{p0}, \quad 0 < k_p \leq k_{p0}$$

(6.10)

and

$$0 \leq \eta \leq \frac{1}{\sup f[\Delta u(k)]} \quad \text{for all } k$$

(6.11)

Consequently, if the parameter $\eta$ satisfies Eq. (6.11), the control system is guaranteed within the stable state of the conventional PID controller. The inner loop controller scheme based on the immune augmentation is shown in the Figure 6.6.

![Block Diagram of the PID-AIS-based Adaptive Mechanism](image)

**Figure 6.6.** Block Diagram of the PID-AIS-based Adaptive Mechanism

Parameters in red in Figure 6.6 must be tuned to modulate a proper response of the AIS on each channel. There will be a total of nine tunable parameters for the AIS for the roll, pitch, and yaw angular rates channels:

$$K_{\text{AIS}p} = [K_p, \eta_p, a_p]$$

$$K_{\text{AIS}q} = [K_q, \eta_q, a_q]$$

$$K_{\text{AIS}r} = [K_r, \eta_r, a_r]$$

(6.12)

Figure 6.7 illustrates the control scheme architecture of the baseline controller extended with the AIS.
6.2.2. Analysis of AIS Compensation Performance

In order to obtain quantitative assessments of the performance of the different control architectures, specific performance metrics were defined in terms of the total pilot input activity, angular rate tracking errors, and the total amount of work required by each of the control surfaces. Extensive simulation tests were performed using both a desktop setup as well as the motion-based flight simulator. The nine parameters of the AIS augmentation have been optimized using an evolutionary algorithm.

The comparison between the different control configurations shows that, in general, the baseline controller augmented with the AIS has better failure rejection capabilities than the baseline + ANN. Both augmented sets of control laws perform better than the baseline (see Figure 6.8).

![Figure 6.7. Baseline Controller Augmented with AIS-based Mechanism](image1)

![Figure 6.8. Results of Average PI for Different ACs](image2)
The level of performance improvement is illustrated in Figure 6.9 and 6.10 showing attitude responses in roll and pitch, respectively, when attempting to keep constant wing level and constant heading after a left stabilator high magnitude failure is injected at 2 seconds. These results are also supported by Figure 6.11 where the pilot activity history is plotted against time for the different control architectures considered. It can be seen that the pilot is required to exercise less activity controlling the aircraft when the AIS is part of the control architecture.

![Figure 6.9. Roll Response Under High Magnitude Stabilator Failure](image)

![Figure 6.10. Pitch Response Under High Magnitude Stabilator Failure](image)

![Figure 6.11. Total Pilot Activity Under High Magnitude Stabilator Failure](image)

On the average, the AIS augmentation achieves better tracking errors with similar level of control activity. Figure 6.12 shows the average PI for angular rate tracking errors over all failures for the three control architectures.

It should be noted that the design complexity and processing requirements of the ANN makes the AIS more desirable from the point of view of implementation and efficiency in terms of hardware task execution time, as well. A simple test was designed and executed in order to...
compare the required hardware time for each of the approaches. The test was repeated for a thousand simulations on a computer with characteristics listed in Table 6.1. The average results are presented in Table 6.2. Figures 6.13 presents the histogram of the task execution time and Figure 6.14 presents the time per simulation versus for all simulation performed. The computational efficiency of the AIS version of the control laws can be easily inferred from these results.

![Average Performance Index for Tracking Error](image)

**Figure 6.12.** Average Performance Index for Tracking Error

<table>
<thead>
<tr>
<th><strong>Table 6.1.</strong> Computer Characteristics for Execution Time Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Computer Specifications</strong></td>
</tr>
<tr>
<td>Model</td>
</tr>
<tr>
<td>Processor</td>
</tr>
<tr>
<td>Ram memory</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Table 6.2.</strong> Task Execution Time Assessment</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Time to run a hundred seconds simulation</strong></td>
</tr>
<tr>
<td>T mean AIS</td>
</tr>
<tr>
<td>T mean NN</td>
</tr>
<tr>
<td>Ratio AIS:NN</td>
</tr>
</tbody>
</table>
Figure 6.13. Task Execution Time Histogram

Figure 6.14. Time per Simulation vs Simulation Count
6.3. Structured Self/Non-self for Control

The SNSA for control attempts to use the AIS already built for ACDIE purposes. It should be noted that two scenarios can be envisioned within the AIS paradigm for control. The first scenario, assumes that substantial information about the failure and its compensation is available and already stored in the AIS. This can be achieved by memorizing the dynamic fingerprint of the AC during first encounter situations, by generating and recording pilot compensation during actual or simulated failures. The second scenario involves an unknown failure that requires a specific new compensation. It is expected that compensation can be produced by properly composing bits of information within the self and non-self.

Within this project, a preliminary investigation has been performed with the objective of assessing the possibility of extracting useful control/compensation information out of properly structured AIS. This general feasibility study was performed using data collected from the motion-based flight simulator at normal and failure conditions for two simple maneuvers: uniform symmetric climb/descent and coordinated turn. The failures considered were actuator locked (stab, aileron, and rudder) and pitch rate sensor bias. Several simplifying assumptions were made for this first step. The information about which subsystem failed, what part of subsystem is affected, and what is the nature and magnitude of the failure must be available. Figure 6.15 illustrates the concept of extracting pilot compensatory commands followed by implementing those commands to the same aircraft, which will be referred to as accommodation. In order to solve the accommodation problem, a set of specific steps must be accomplished as illustrated in Figure 6.16.

\[ FF = \{ feat_1, feat_2, ..., feat_{N_f} \} \]  

(6.13)
where \( \text{feat}_j \ (j=1,2, \ldots, N_f) \) represents the feature (e.g., roll rate, velocity). An input to the pilot (or mission objectives) can be defined as a matrix \( \text{REF} \) whose rows are represented by vectors \( \text{REF}_v \) of length \( N_f \):

\[
\text{REF}_v = [d_{f_1}, d_{f_2}, \ldots, d_{f_{N_f}}] \quad \text{(6.14)}
\]

Each element \( d_{f_j} \) of the vector \( \text{REF}_v \) represents a desirable value of the corresponding feature in set \( FF \). Duration of the flight is defined by the number of time samples \( ts \). Consequently, matrix \( \text{REF} \) consists of \( ts \) rows.

![Diagram](image.png)

**Figure 6.16.** Accommodation Problem Main Components

An outcome of pilot's performance is a matrix \( \text{PF} \). Every row in this matrix is represented by a vector \( \text{PF}_v \) of length \( N_f \):

\[
\text{PF}_v = [p_{f_1}, p_{f_2}, \ldots, p_{f_{N_f}}] \quad \text{(6.15)}
\]

where each element \( p_{f_j} \) of the vector \( \text{PF}_v \) represents a value of the corresponding feature of set \( FF \) obtained from pilot performance. Matrix \( \text{PF} \) consists of \( ts \) rows as well. Pilot generated control commands are defined by matrix \( \text{PC} \). Every row in this matrix is represented by a vector \( \text{PC}_v \) of length \( N_{cc} \), where \( N_{cc} \) is the number of control commands:
where element $ctrl_i$ of the vector $PC_v$ represents a control command provided by pilot at each
time sample during flight. Matrix $PC$ consists of $ts$ rows. The vector of measured in-coming
values of features at every time sample is defined as $INC_v$. The length of the vector is $N_f$.

$$PC_v = [ctrl_1, ctrl_2, ..., ctrl_{N_{cc}}] \quad (6.16)$$

Each element $if_j$ of the vector $INC_v$ represents an in-coming value of the corresponding feature of
set $FF$.

$$INC_v = [if_1, if_2, ..., if_{N_f}] \quad (6.17)$$

### 6.3.2. Generating AIS Memory

There is a certain flight path that the aircraft should follow. There are also specific flight
conditions that the aircraft should maintain during the flight (e.g. constant altitude, constant
velocity). These parameters define the task and are considered as the input to the pilot+aircraft
system. After being informed or making the decision about this flight path and about the specific
flight conditions, an experienced pilot intends to execute the task as accurate as possible. An
aircraft following the desirable flight path under the commanded flight conditions will be
assimilated to a healthy organism. Every value of in-coming flight variable that alters from the
desirable value during the pilot performance will be defined as invading entity/antigen. The
control commands provided by the pilot that are trying to bring the aircraft to the desirable state
will be defined as immune system antibodies. This process is an analogy to the immune system
being affected by a disease for the first time. The adaptive immune system fights back and
generates antibodies, which eventually eliminate the disease. The recorded and saved flight data
will define a set of created artificial memory cells (B-cells, T-cells, and antibodies) and antigens.

The set of antigens is defined as the difference between in-coming data and reference input
(pilot mission at each time sample $tsc$).

$$AG_v = REF_v - INC_v \quad (6.18)$$

$$AG_v = [df_1 - if_1, df_2 - if_2, ..., df_{N_f} - if_{N_f}] \quad (6.19)$$

Depending on flight scenarios, the content of the antigen cell might be extended to include not
only values of differences but actual values of incoming features as well:

$$AG_v = [df_1 - if_1, ..., df_{N_f} - if_{N_f}, if_1, ..., if_{N_f}] \quad (6.20)$$

These additional features are selected depending on the specific task and prescribed maneuver.
The collection of $AG_v$ vectors over the entire flight time defines the matrix $AG$. The overall
number of time samples during the flight is $ts$, consequently matrix $AG$ has $ts$ rows.

The immune system memory has information about the T-cells and B-cells produced during
the prior infection encounters, and antibodies that were used to suppress these antigens. Within
the AIS paradigm, T-cells together with B-cells will be assimilated to a difference between pilot
features data and reference input data. The set of T-cells and B-cells for a specific time sample
$tsc$ is a vector $TB_v$ defined by subtraction of elements of vector $PF_v$ from corresponding elements
of vector $REF_v$. 

140
Similarly to $AG_v$ vector, a $TB_v$ vector might need an extension and include some actual values of features obtained from pilot performance:

$$TB_v = [df_1 - pf_1, df_2 - pf_2, ..., df_{N_f} - pf_{N_f}]$$

(6.22)

The additional features for $TB_v$ should be the same as for $AG_v$. The collection of $TB_v$ vectors over the entire flight time defines the matrix $TB$. The overall number of time samples during the flight is $ts$, consequently, matrix $TB$ has $ts$ rows.

The set of *antibodies* for a specific time sample $tsc$ is a vector $AB_v$ defined by the corresponding vector from matrix PC.

$$AB_v = PC_v$$

(6.24)

The collection of $AB_v$ vectors over the entire flight time defines matrix $AB$. Considering all definitions and notations in this section, the generic structure of the AIS memory is presented in Figure 6.17.
6.3.3. Matching Algorithm

At each time sample, an incoming antigen $AG_v$ is compared to the sets of T-cells and B-cells from matrix $TB$. Once the “best match” is found, a corresponding set of antibodies $AB_v$ is retrieved from matrix $AB$. As a result, control commands are obtained dictating the behavior of the aircraft. The “best match” is a row vector $TB_v$ (from matrix $TB$) which is most similar to an incoming antigen vector $AG_v$. Let vector $ERR$ be the difference between $AG_v$ and $TB_v$:

$$ERR = AG_v - TB_v$$  \hspace{1cm} (6.25)

Then sum $S$ can be defined as follows:

$$S = \sum_{i=1}^{N_f} \text{abs}(ERR(i))$$  \hspace{1cm} (6.26)

where $ERR(i)$ is an element of vector $ERR$. Therefore, a row vector $TB_v$ that leads to the lowest value of $S$ is defined as the “best match”. It should be noted that more sophisticated matching algorithm may be used to improve performance. For higher performance, one should take into consideration the current state and the state in which aircraft had been before (i.e. several time samples before). Once the number of the states prior to current is defined, the sum will be calculated accordingly:

$$S = \sum_{j=tsc-Nt}^{tsc} \sum_{i=1}^{N_f} \text{abs}(ERR(i)) j$$  \hspace{1cm} (6.27)

where $tsc$ is the current time sample, and $N_t$ is the size of the floating time window over which matching is investigated.

6.3.4. Example Results Using the Structured Self/Non-self for Control

A). Simulation Scenarios

The flight scenarios, lasting between 2 and 4 minutes, were designed to include steady state flight conditions and moderate maneuvers. These flight scenarios were first simulated under nominal flight conditions. They were repeated under various failure conditions for both design/development and validation purposes. Only one failure at a time is injected. Two simplified maneuvers were considered, one in the vertical plane of symmetry (climb) and one on the lateral-directional channel with longitudinal coupling (coordinated turn).

The symmetric climb maneuver (Figure 6.18) consists of steady state symmetric flight at 6050 m and Mach 0.75 for 30 seconds, uniform climb to 6900 m at Mach 0.75, and steady state symmetric horizontal flight for 1 minute at the destination altitude (6900 m) and Mach 0.75. The coordinated half turn maneuver (Figure 6.19) consists of steady state symmetric flight at 6050 m and Mach 0.75 for 30 seconds and coordinated half turn at constant bank angle while maintaining altitude and velocity.

Two types of failures were considered: left actuator locked at +4 degrees and roll rate sensor step bias. Failure injection time for all the tests is $T_f = 15 s$. Sensor noise and mild atmospheric turbulence are included in the simulation.
B). System Implementation

For the climbing flight scenario, the following set of flight features \( FF \) was selected: aircraft velocity \( V \), roll rate \( p \), pitch rate \( q \), yaw rate \( r \), altitude \( H \), and pitch angle \( \theta \).

\[
FF = \{V, p, q, r, H, \theta\} \tag{6.28}
\]

The pilot generated commands are the following: longitudinal channel stick displacement \( de \), lateral channel stick displacement \( da \), directional channel pedals displacement \( dr \), and throttle displacement \( dt \). Therefore:

\[
PC = \{de, da, dr, dt\} \tag{6.29}
\]

The antigen vector \( AG_v \) will consist of elements that represent the differences between the desired and actual values of corresponding feature. However, antigen vector has to be extended by including additional elements that represent the actual values of certain features. The additional features for the extension of antigen vector are: bank angle \( \phi \), pitch angle \( \theta \), yaw angle \( \psi \), pitch rate \( q \), and altitude \( H \). The flight duration time is \( T = 130s \), consequently, the overall number of time samples is \( ts = 6500 \). The number of time samples used prior to the current moment in the matching algorithm is selected to be \( N_t = 4 \).

For the coordinated half turn maneuver, the following set of flight features \( FF \) was selected: aircraft velocity \( V \), roll rate \( p \), pitch rate \( q \), yaw rate \( r \), altitude \( H \), and bank angle \( \phi \). Therefore:

\[
FF = \{V, p, q, r, H, \phi\} \tag{6.30}
\]

The pilot generated commands are the same as in the climb maneuver. The additional features for the extension of antigen vector are: bank angle \( \phi \), pitch angle \( \theta \), yaw angle \( \psi \), and roll rate \( p \). The flight duration time is \( T = 215s \), consequently, the overall number of time samples is \( ts = 10750 \). The number of time samples used prior to the current moment in the matching algorithm is selected to be \( N_t = 4 \).

C). Simulation Results for Symmetric Climb

Figure 6.20 presents the variation of aircraft altitude during the symmetric flight under nominal conditions. The generated altitude was very accurately reproduced and is very close to the altitude of pilot performance. There is a little deviation at the end of the flight; however, the
drifting down trend is preserved. Figure 6.21 presents the longitudinal channel control commands. The outcomes of the failure accommodation system almost coincide with the pilot performance.

Figure 6.20. Aircraft Altitude at Nominal

Figure 6.21. Longitudinal Channel Commands

Figure 6.22 presents the aircraft altitude variation under locked stabilator. A small decrease of altitude can be noticed at around $T=20s$ due to the stabilator failure. Figure 6.23 illustrates the longitudinal channel commands. Notice that the pilot workload on the longitudinal channel has significantly increased, in comparison with nominal conditions flight. Small insignificant inaccuracies can be observed.

Figure 6.22. Aircraft Altitude

Figure 6.23. Longitudinal Channel Commands

Figure 6.24 illustrates the lateral channel commands. Although the failure affects mostly the longitudinal channel, pilot workload on the lateral channel is larger. The generated commands almost coincide with the pilot input. Figure 6.25 presents the variation of the lateral position in Earth axes. Although the stabilator failure mostly affects the longitudinal channel, a coupling with lateral channel takes place. Therefore, because of the bank angle $\phi$ alteration (see Figure 6.26), the aircraft slightly deviates from the straight forward trajectory.

Figure 6.24 illustrates the trajectory of symmetric climb under a roll rate sensor failure. Trajectory has been accurately simulated and followed the pilot's outcome. A small delay in climbing process can be noticed. Also, the altitude deviates from the desired value in the final stage of the flight. Longitudinal channel commands have been generated accurately during the sensor failure flight as presented in Figure 6.28. The generated commands followed the pilot
trend, including the beginning of the flight when the failure was injected. Several inaccuracies can be observed during the climb maneuver; however, they did not have a significant effect on aircraft altitude variation.
Lateral channel commands extraction was not as successful as the one for the longitudinal channel (see Figure 6.29). Generally, the plot exhibits visual inaccuracies in the command extraction. However, it can be noted that at the initial stage of the flight, the commands that were dedicated to overcome the injected failure have been generated very accurately. A bank angle offset results, as presented in Figure 6.30, with an integral effect eventually producing a lateral deviation from the straight flight.

![Aircraft Bank Angle](image)

**Figure 6.30.** Aircraft Bank Angle

**D). Simulation Results for Coordinated Half-Turn**

Figure 6.31 presents the trajectory of the coordinated half turn in the horizontal plane under nominal conditions. The entire maneuver was generated successfully without any significant differences from the pilot performance outcome.

![Coordinated Turn Flight Trajectory](image)

**Figure 6.31.** Coordinated Turn Flight Trajectory Under Nominal Conditions

Figure 6.32 presents the lateral channel commands. A short command at the beginning of the flight was produced to reach the desirable bank angle $\phi$ (see Figure 6.33). Since it is a nominal conditions flight, for the rest of the flight, no commands on this channel were needed.
Figure 6.32. Lateral Channel Commands for Coordinated Turn at Nominal Conditions

Figure 6.34 presents the longitudinal channel commands. When the aircraft is banked at a constant angle it tends to lose altitude; that is why the pilot must provide certain longitudinal commands to maintain the vertical position. This task is slightly more demanding because constant lateral input has to be preserved as well. Longitudinal command reproduction is not as neat and accurate; however, it follows the pilot trend.

Figure 6.33. Aircraft Bank Angle

Figure 6.34. Longitudinal Channel Commands

Figure 6.35 illustrates the aircraft altitude variation. Although the generated longitudinal commands were slightly inaccurate, the aircraft altitude is well maintained and does not differ a lot from the pilot performance outcome.

Figure 6.36 presents the flight trajectory in the horizontal plane under stabilator failure when performing a coordinated turn. An excellent duplication of pilot performance can be observed. Longitudinal channel commands are presented in Figure 6.37. There is a considerable input signal at the beginning of the flight that has been provided to mitigate the failure. The generated simulation provided that signal as well. During the rest of the time, the commands are extracted very accurately, with several inaccuracies at the end.
Lateral channel control commands are presented in Figure 6.38. Because of the coupling, the pilot had to provide some commands on the lateral channel as well right after the failure was injected. Over the entire flight, the commands have been extracted precisely: the lines almost coincide. Generated bank angle $\phi$ dictated by the lateral commands also follows the pilot performance outcome accurately (Figure 6.39). Little peak is observed at the beginning of the flight due to the failure; however, the value has been brought back close to zero by pilot compensation.

Figure 6.40 illustrates the flight trajectory in the horizontal plane for a coordinated turn executed under roll rate sensor bias. Generated trajectory follows the pilot trend; however, the generated curve has a smaller radius of turn as compared to the one from the pilot performance outcome due to the inaccuracies in reproducing lateral channel commands. Figure 6.41 presents the lateral channel commands provided during the flight simulation. The commands have not been extracted as accurately as for actuator failures; however, they behave similarly to pilot generated commands with a small delay and several inaccuracies of small magnitude. Figure
6.42 illustrates the commands on the longitudinal channel. Similarly to the lateral channel, the command extraction is not as accurate as for actuator failure; however, the similar trend can be observed in the behavior. These inaccuracies on longitudinal and lateral channels can be explained by excessive amount of the pilot workload and rapid change of stick displacement during the flight. For the flight with a large pilot workload and very intensive stick commands, a more advanced matching algorithm might be needed for obtaining higher performance results. Also, there is a delay between pilot input and aircraft response which becomes more critical when abrupt maneuvers are involved. Under such conditions, this delay should be better taken into account. One possible solution could be to increase the time window over which the matching algorithm is applied.

**Figure 6.38.** Lateral Channel Commands Under Stab Failure

**Figure 6.39.** Aircraft Bank Angle Under Stab Failure

**Figure 6.40.** Flight Trajectory for Coordinated Turn Under Roll Rate Sensor Failure

**Figure 6.41.** Lateral Channel Commands Under Roll Rate Sensor Failure
Figure 6.42. Longitudinal Channel Commands for Coordinated Turn Under Roll Rate Sensor Failure
Chapter 7

Integrated Scheme for Aircraft Subsystem Abnormal Conditions Detection, Identification, and Evaluation

7.1. Integrated ACDIE Scheme Using the Structured Self/Non-self Approach

Based on the AIS paradigm and a HMS strategy, the SNSA was developed and implemented for aircraft subsystems ACDIE. All phases of the ACDIE process have been integrated in one single architecture, that starts with the generation of antibodies (detectors) and continues with structuring of the non-self for identification and evaluation purposes, as well as providing the necessary information for AC accommodation. Flight data from the WVU 6-DOF flight simulator were initially used to define a large set of 2-dimensional self/non-self projections as well as for the generation of antibodies and identifiers designated for health assessment of an aircraft under upset conditions. As shown in Figure 7.1, this represents the first step in the complex ACDIE process. Selecting the predefined set of features or characteristic variables for the definition of the self/non-self is a critical component of the AIS-based methodology.

During “detection” phase, the process of declaring a generic failure in one or more of the aircraft sub-systems is performed. A detection logic is also designed for real time operation to ensure that high detection rates and low false alarm rates are achieved. Once a failure has been detected, the “identification” phase starts. The identification phase determines which sub-system has failed by analyzing which of the detectors has been activated through a positive selection-type scheme. In this phase, all the detectors are labeled in a previous offline process (SNSA) in order to assign specific detectors to particular categories of failures. This off-line process or “structuring” consists of outlining which non-selves are activated under a specific failure. Depending on the complexity of the targeted systems, the identification process can determine which subsystems have failed (i.e. actuator, sensor, structural, etc.). The evaluation phase can be divided into two steps, namely direct evaluation and indirect evaluation. Furthermore, the direct evaluation phase can be classified into qualitative and quantitative evaluation. The direct qualitative evaluation phase isolates and determines the specific subsystem that has failed. The quantitative evaluation phase determines the severity or magnitude of a failed subsystem. The indirect evaluation phase determines the effect on the flight envelope maneuverability and performance of the system after a failure has affected the system. Finally, once the failure has been detected and properly identified, the best solution to compensate such failure is executed using active or passive accommodation tasks that have been previously stored in the immune memory.
Figure 7.1. ACDIE with the Structured Non-Self Approach
7.2. Integrated ACDIE Scheme Using the DC Mechanism-based Approach

Figure 7.2 shows a top-level block diagram of the online ACDIEA scheme using the artificial DC mechanism. The measured data (i.e., feature values) are first normalized over a moving time window using the same normalization used in generating the self and non-self. The normalized data are then projected on the corresponding projections (sub-selfs) to generate the discrimination matrices that are used as input to the artificial DC mechanism for detection. Once a failure is detected, the features-pattern vector is computed based from the $F_i$ matrices of all migrated DCs (stimulatory and regulatory DCs) and compared with the libraries of reference patterns to identify the failed subsystem and evaluate the type of the failure and its severity simultaneously.

**Figure 7.2.** ACDIE with the DC Mechanism Approach
Chapter 8

Demonstration of the Integrated Scheme for Aircraft Subsystem Abnormal Conditions Detection, Identification, and Evaluation

8.1. Implementation of the Integrated Scheme for Aircraft Subsystem Abnormal Conditions Detection, Identification, and Evaluation

The integrated aircraft subsystem ACDIE scheme was implemented and tested with Matlab/Simulink 2010a (32-bit). Figure 8.1 shows the top level Simulink model using the SNSA, where the “immunity-HMS” block detail is shown in Figure 8.2. The top level Simulink model of the integrated scheme using the DC mechanism is illustrated in Figure 8.3 and the DC mechanism block detail is shown in Figure 8.4.

![Figure 8.1. Top-level Simulink Model of the Integrated Aircraft ACDIE Using the Structured Non-Self Approach](image1)

![Figure 8.2. Simulink Block Diagram of the Structured Non-Self Approach](image2)
Figure 8.3. Top-level Simulink Model of the Integrated Aircraft ACDIE Scheme Using the DC Mechanism

Figure 8.4. Simulink Block Diagram of the DC Mechanism
Both of the models deliver their outcome to a GUI at each time step. Figure 8.5 shows the interface displaying the current ACDIE outcome for a flight test under failure of the left wing. The implementation of the integrated scheme has two components. One allows the scheme to run in conjunction with the aircraft simulation. The other allows feeding previously obtained simulation data into the ACDIE scheme for faster testing and analysis.

Figure 8.5. ACDIE Outcome Visualization Interface

The ACDIE outcome visualization interface has four components:

1. A header displaying the current flight condition and the current sample time. If the current flight condition is normal, the header will be “Normal Flight Conditions” in green. If the current flight condition is abnormal, the header will be “Abnormal Flight Conditions” in red.

2. Four-line text on the left of the interface showing brief information about the failure (failed subsystem, its severity, detection time, and detection rate).

3. An aircraft image displaying the current status of aircraft’s flight condition. When the ACDIE outcome carries information about the failed subsystem, the image dynamically highlights the failed subsystem.

4. A table displaying the percentage rates of the identified subsystems, types of the failure, and severity of the failure.

The interface computes the detection, identification, and evaluation rates for all received ACDIE outcomes. However, the interface cannot update its contents when it receives two successive outcomes in a short time interval. Therefore, a predefined “update rate” representing the number of graphical update milliseconds was introduced in the configuration of the interface. For example, when the interface is configured with an “update rate” of 100, it will only update its graphical contents every 100 millisecond, while it keeps computing the ACDIE rates for any ACDIE outcome it receives.
8.2. Demonstration of the Integrated Scheme for Aircraft Subsystem Abnormal Conditions Detection, Identification, and Evaluation Through Simulation

The integrated ACDIE scheme was tested for 10 additional demonstration flight tests with different failed subsystems, failure types, and failure magnitudes. In these tests, the both the DC mechanism approach and the SNSA were used. Table 8.1 lists the detection rates as well as the path of the flight test (points in the test envelope of Figure 1.22) and the detection time. The failure was introduced at 40 s. The false alarm rate was zero for all tests. These example results were obtained through direct recorded data input and through on-line simulation input.

Table 8.1. Detection Results for a Set of Demonstration Tests

<table>
<thead>
<tr>
<th>Subsystem Affected by AC</th>
<th>Failure Type</th>
<th>Failure Magnitude</th>
<th>Path</th>
<th>Detection Time</th>
<th>Detection Rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left Stabilator</td>
<td>locked</td>
<td>8°</td>
<td>145</td>
<td>40.68</td>
<td>99.9</td>
</tr>
<tr>
<td>Left Wing</td>
<td>damaged</td>
<td>15%</td>
<td>167</td>
<td>40.62</td>
<td>99.9</td>
</tr>
<tr>
<td>Pitch Sensor</td>
<td>bias</td>
<td>10°</td>
<td>189</td>
<td>40.72</td>
<td>99.9</td>
</tr>
<tr>
<td>Right Aileron</td>
<td>locked</td>
<td>8°</td>
<td>165</td>
<td>40.72</td>
<td>99.9</td>
</tr>
<tr>
<td>Right Wing</td>
<td>damaged</td>
<td>6%</td>
<td>189</td>
<td>40.88</td>
<td>99.9</td>
</tr>
<tr>
<td>Roll Sensor</td>
<td>bias</td>
<td>10°</td>
<td>145</td>
<td>41.68</td>
<td>98.9</td>
</tr>
<tr>
<td>Yaw Sensor</td>
<td>bias</td>
<td>3°</td>
<td>167</td>
<td>41.12</td>
<td>99.9</td>
</tr>
<tr>
<td>Left Stabilator</td>
<td>locked</td>
<td>6°</td>
<td>1B</td>
<td>40.28</td>
<td>100.0</td>
</tr>
<tr>
<td>Left Wing</td>
<td>damaged</td>
<td>10%</td>
<td>1B</td>
<td>40.28</td>
<td>100.0</td>
</tr>
<tr>
<td>Roll Sensor</td>
<td>bias</td>
<td>10°</td>
<td>1B</td>
<td>45.58</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Table 8.2 presents the identification rates for the same tests listed in Table 8.1, whereas the corresponding failure type and severity evaluation rates are presented in Tables 8.3 and 8.4, respectively. Note that the evaluation rates are computed with respect to the number of points for which correct identification was obtained.

Table 8.2. Identification Results for the Demonstration Tests

<table>
<thead>
<tr>
<th>Identified Subsystem</th>
<th>Left Stab</th>
<th>Right Stab</th>
<th>Left Aileron</th>
<th>Right Aileron</th>
<th>Left Rudder</th>
<th>Right Rudder</th>
<th>Left Throttle</th>
<th>Right Throttle</th>
<th>Left Wing</th>
<th>Right Wing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left Stabilator</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Left Wing</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>98.7</td>
<td>1.3</td>
</tr>
<tr>
<td>Pitch Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Right Aileron</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Right Wing</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>99.9</td>
</tr>
<tr>
<td>Roll Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Yaw Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Left Stabilator</td>
<td>99.9</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Left Wing</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>99.8</td>
</tr>
<tr>
<td>Roll Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>12.7</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

157
<table>
<thead>
<tr>
<th>Identified Subsystem</th>
<th>Left V Tail</th>
<th>Right V Tail</th>
<th>Left H Tail</th>
<th>Right H Tail</th>
<th>Roll Sensor</th>
<th>Pitch Sensor</th>
<th>Yaw Sensor</th>
<th>Left Engine</th>
<th>Right Engine</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left Stabilator</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Left Wing</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Pitch Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Right Aileron</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.2</td>
<td>0.0</td>
</tr>
<tr>
<td>Right Wing</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Roll Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Yaw Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Left Stabilator</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Left Wing</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Roll Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>87.0</td>
<td>0.3</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Table 8.3.** Failure Type Evaluation Results for the Demonstration Tests

<table>
<thead>
<tr>
<th>Evaluated Type</th>
<th>Type 1</th>
<th>Type 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Left Stabilator</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Left Wing</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Pitch Sensor</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Right Aileron</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Right Wing</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Roll Sensor</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Yaw Sensor</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Left Stabilator</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Left Wing</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Roll Sensor</td>
<td>100.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

**Table 8.4.** Failure Severity Evaluation Results for the Demonstration Tests

<table>
<thead>
<tr>
<th>Evaluated Severity</th>
<th>Low</th>
<th>Medium</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Left Stabilator</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
</tr>
<tr>
<td>Left Wing</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Pitch Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
</tr>
<tr>
<td>Right Aileron</td>
<td>1.2</td>
<td>0.0</td>
<td>98.8</td>
</tr>
<tr>
<td>Right Wing</td>
<td>99.9</td>
<td>0.0</td>
<td>0.1</td>
</tr>
<tr>
<td>Roll Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
</tr>
<tr>
<td>Yaw Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
</tr>
<tr>
<td>Left Stabilator</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Left Wing</td>
<td>0.0</td>
<td>100.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Roll Sensor</td>
<td>0.0</td>
<td>0.0</td>
<td>100.0</td>
</tr>
</tbody>
</table>
Figures 8.6 and 8.7 show the ACDIE outcome visualization window for demo tests under normal flight conditions and a 15% damage of the left wing, respectively.

**Figure 8.6.** Demonstration Results for a Nominal Flight

**Figure 8.7.** Demonstration Results for a 15% Damage of the Left Wing
8.3. Demonstration of the AIS AC Detection Through Flight Testing Using a UAV Research Platform

8.3.1. Flight Test Scenario

An initial performance analysis of the proposed ACDIE solution has been assessed through flight tests. A set of two nominal flights that included figure-8-maneuvers as well as control surface doublets in all three axes were performed in order to capture the dynamics of the test aircraft. The platform used for this experimentation was the Skywalker RC aircraft instrumented with an APM 2.5 micro controller. The RC aircraft was flown manually by a pilot on the ground and the maneuvers were implemented in a sequential order. The nominal flight tests consisted of the following stages:

1. Manual flight until an altitude of 80 meters
2. Trim flight at constant speed
3. Figure 8 maneuver
4. Elevator Doublet
5. Figure 8 maneuver
6. Aileron Doublet
7. Figure 8 maneuver
8. Rudder Doublet
9. Figure 8 maneuver

The flight test sequence was performed twice and the data were saved into a flash memory for later processing. The data was recorded at a rate of 50Hz and included angular rates, linear accelerations, neural network angular rate estimations, reference model commands, Euler angles, altitude, stick inputs and velocity. A total of 21 features were recorded, which generate a total of 210 self/non-self 2-D projections. The reduced flight envelope limits of the Skywalker as well as the length of the flights yield a reduced amount of data points. This should be taken into consideration during the generation of projections in order to obtain proper coverage of the self.

For validation purposes, four different types of failures were injected into the system at later flights in order to capture the dynamic fingerprint of abnormal conditions on the test platform. The failures investigated included low and high magnitude aileron failures. The failures were injected manually by the pilot through a PWM signal sent from an RC transmitter. Once a failure was injected, the sequence of maneuvers presented above was attempted by the pilot. Table 8.5 presents the injected failures in the system.

<table>
<thead>
<tr>
<th>Failure #1</th>
<th>Left Aileron Locked at Wings Level Trim</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failure #2</td>
<td>Right Aileron Locked at Wings Level Trim</td>
</tr>
<tr>
<td>Failure #3</td>
<td>Left Aileron Locked at Trim during Bank Turn</td>
</tr>
<tr>
<td>Failure #4</td>
<td>Right Aileron Locked at Trim during Bank Turn</td>
</tr>
</tbody>
</table>

8.3.2. Test Platform

The RC airplane chosen for experimentation was the “New Skywalker 1880”. This platform offers a stable and low-cost system that is able to satisfy the needs of the flight tests. Figure 8.8 shows the actual system used for the flight tests. This platform was used in previous projects for which system identification techniques were performed. The physical characteristics of the system are presented in Table 8.6.
The Skywalker 1880 was equipped with a set of analog and digital sensors that provide essential variables for the generation of Selves and Non-selves. Primarily, the onboard microcontroller is an APM 2.5 with an “Atmel ATMEGA 2560” processor. The APM 2.5 board includes embedded sensors such as an IMU, magnetometer and a 4MB data flash chip as well as digital and analog ports for GPS, telemetry and a pitot tube sensor. Figure 8.9 shows the APM and the setup inside the fuselage of the Skywalker.
The Ardupilot Mega (APM) 2.5 is an out-of-the-shelf low-cost autopilot solution produced by 3D Robotics. Its dimensions are approximately 66x40 mm. and with the enclosure it weighs approximately 20 grams. It uses an 8-bit, 16Mhz “Atmel AT Mega 2560” processor which has 54 digital I/O pins for which 14 can be used for PWM signals.

The InvenSense MPU-6000 inertial sensor is a 6-axis motion tracking device. It combines a 3-axis gyroscope and a 3-axis accelerometer in a 4x4x0.9mm QFN footprint and it communicates through a serial interface in an I2C protocol.

The MEAS Switzerland MS5611 Barometric Pressure Sensor offers a high resolution altimeter sensor with SPI and I2C bus interfaces up to 20MHz. It offers a factory calibrated sensor with a resolution of 10 cm. and its dimensions corresponds to a 5x3x1mm QFN footprint.

The data flash card used is a 4Mb chip embedded in the APM 2.5 board. Previous efforts have shown that recording 20 floating point parameters at 50Hz allows recording approximately 17 minutes of flight.

The MediaTek MT3329 is a 66 channel single chip solution with a binary output protocol with an update rate of 10Hz. Its sensitivity can be up to -165dB tracking, a position accuracy of less than 3meters and USB/UART interfaces. Its dimensions are 38x16x7.8mm and it weighs 9.45g.

The Freescale MPXV7002DP Differential Pressure Sensor was connected directly to a miniature pitot tube located on the right wing. Its maximum rating for pressure is 2kPa at 60°C. This sensor provides true air speed measurements that are used for the definitions of selves.

The Spektrum DX7s RC transmitter and receiver is a 7-channel, 2.4 GHz remote control device used for manual control of the aircraft and control of aileron failures in the system. Five channels were used for the control of ailerons, rudder, throttle, and elevator and a sixth channel commanded the aileron to lock at a given position.

The power source chosen for the system was a G6 Pro Lite Thunder Power 5 cell LiPo 18.5V battery.

The electric motor chosen for the Skywalker was a 1000Kv RPM Turnigy brushless motor. Its maximum current is 38A and it has a maximum power of 665W. Its weight is approximately 130g and its size is 35x42mm with a shaft diameter of 5mm.

### 8.3.3. Simulink Models

The onboard microcontroller has the ability to be targeted through Simulink and the APM 2.0 Block set for Simulink. This feature provides a great advantage for any effort involving low cost autopilots and sensor fusions boards. Several Simulink models including model reference control, artificial neural networks and Kalman filter models, were designed in Simulink and later loaded into the APM 2.5 board for flight test implementation. Figure 8.10 shows the top level of the Simulink model loaded into the APM 2.5 board.

In general, the Simulink model allows the APM board to read several sensors embedded in the board while recording flight test data in real time. Once the data is recorded in the flash memory, it can be downloaded and processed off-line for the generation of projections. Most of the features are obtained from the sensors embedded in the onboard computer. On the other hand, the bank and pitch angles as well as the ANN and model reference control outputs are generated by separate Simulink models. The following sections will briefly describe the Simulink models for the mentioned systems.

It is possible to auto-generate code through the Run-On-Target-Hardware tool in the Simulink environment into the APM 2.5 board. The APM 2.0 Simulink blockset allows users to
read data from the sensors embedded in the board, command PWM signals to servos and to run guidance, navigation and control algorithms. This library allows users to read the sensors embedded in the microcontroller. Therefore, a Simulink model that reads the data from such sensors was generated and loaded into the onboard computer. Figure 8.11 shows a sample model of the sensors.

![Simulink Sensor Model](image)

**Figure 8.10.** Simulink Model Top Level

**Figure 8.11.** Simulink Sensor Model
In previous efforts, the Skywalker 1880 platform was used as an autonomous platform. For this effort, a sensor fusion solution was developed. A discrete Kalman filter was designed in Simulink in order to determine bank and pitch angles of the platform. The Kalman filter received inputs from a complementary filter that integrated data from the gyroscopes and the accelerometer. The Kalman filter demonstrated excellent performance and accuracy in the estimation of the desired Euler angles. Based on these characteristics, the estimator was included in the Simulink model with the objective to generate important variables for the definition of Self/Non-self projections. Figure 8.12 shows a Simulink model with the basic architecture of the Kalman filter implementation.

![Figure 8.12. Kalman Filter Architecture](image)

A model reference controller and ANNs were incorporated into the system in order to generate features for the definition of self/non-self. The ANN used was an ADALINE network, which requires eight variables as input. The Simulink model presents a relatively complex architecture that will not be shown in detail for simplicity.

### 8.3.4. Flight Test Results

Two nominal flights and four failure flights were performed. A total of 38 different channels were recorded into the flash memory. Out of those, only 18 were selected for the definition of projections. Table 8.7 presents the features that were recorded and selected for self/non-self definition.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( p )</td>
<td>( NN_p ) ( \varphi )</td>
</tr>
<tr>
<td>( q )</td>
<td>( NN_q ) ( \theta )</td>
</tr>
<tr>
<td>( r )</td>
<td>( NN_r ) ( \phi_e )</td>
</tr>
<tr>
<td>( p_{ref} )</td>
<td>( a_x ) ( d_a )</td>
</tr>
<tr>
<td>( q_{ref} )</td>
<td>( a_y ) ( d_r )</td>
</tr>
<tr>
<td>( r_{ref} )</td>
<td>( a_z ) ( d_T )</td>
</tr>
</tbody>
</table>

153 projections were tested against flight test data in order to determine the detection performance of the generated selves. The selves analyzed the data from four different failure flights. The analysis determined that 30 different self projections were able to capture the dynamic fingerprint of abnormal conditions. Table 8.8 presents these projections.
Table 8.8. Skywalker Self Projections

<table>
<thead>
<tr>
<th>Self#</th>
<th>Features</th>
<th>Self#</th>
<th>Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$p_{\text{ref}}, q_{\text{ref}}$</td>
<td>16</td>
<td>$p_{\text{ref}}, \theta$</td>
</tr>
<tr>
<td>2</td>
<td>$p_{\text{ref}}, r_{\text{ref}}$</td>
<td>17</td>
<td>$p_{\text{ref}}, \phi$</td>
</tr>
<tr>
<td>3</td>
<td>$p_{\text{ref}}, q$</td>
<td>18</td>
<td>$r_{\text{ref}}, a_x$</td>
</tr>
<tr>
<td>4</td>
<td>$p_{\text{ref}}, q$</td>
<td>19</td>
<td>$p, d_a$</td>
</tr>
<tr>
<td>5</td>
<td>$p_{\text{ref}}, r$</td>
<td>20</td>
<td>$q, d_a$</td>
</tr>
<tr>
<td>6</td>
<td>$p_{\text{ref}}, N\dot{N}_p$</td>
<td>21</td>
<td>$r, a_x$</td>
</tr>
<tr>
<td>7</td>
<td>$p_{\text{ref}}, N\dot{N}_q$</td>
<td>22</td>
<td>$N\dot{N}_p, d_a$</td>
</tr>
<tr>
<td>8</td>
<td>$p_{\text{ref}}, N\dot{N}_r$</td>
<td>23</td>
<td>$N\dot{N}_a, d_a$</td>
</tr>
<tr>
<td>9</td>
<td>$p_{\text{ref}}, a_x$</td>
<td>24</td>
<td>$a_x, \phi$</td>
</tr>
<tr>
<td>10</td>
<td>$p_{\text{ref}}, a_y$</td>
<td>25</td>
<td>$a_x, \theta$</td>
</tr>
<tr>
<td>11</td>
<td>$p_{\text{ref}}, a_z$</td>
<td>26</td>
<td>$a_x, d_a$</td>
</tr>
<tr>
<td>12</td>
<td>$p_{\text{ref}}, d_a$</td>
<td>27</td>
<td>$a_x, d_r$</td>
</tr>
<tr>
<td>13</td>
<td>$p_{\text{ref}}, d_e$</td>
<td>28</td>
<td>$\theta, d_a$</td>
</tr>
<tr>
<td>14</td>
<td>$p_{\text{ref}}, d_T$</td>
<td>29</td>
<td>$\phi, d_a$</td>
</tr>
<tr>
<td>15</td>
<td>$p_{\text{ref}}, d_r$</td>
<td>30</td>
<td>$d_a, d_r$</td>
</tr>
</tbody>
</table>

The selected projections obtained an average 21.11\% and 30.68\% DR for ailerons stuck at wings level trim and ailerons stuck during a bank maneuver respectively. These detection rates may seem to be low. However, they should be considered in conjunction with the expected 0 FA rate. Table 8.9 presents the DR and FA rate of a sample set of projections.

Table 8.9. Detection Rate and False Alarm Rate for 4 Failures

<table>
<thead>
<tr>
<th>Self#</th>
<th>Right Aileron Locked at Wings Level</th>
<th>Right Aileron Locked at Bank Maneuver</th>
<th>Left Aileron Locked at Wings Level</th>
<th>Left Aileron Locked at Bank Maneuver</th>
</tr>
</thead>
<tbody>
<tr>
<td>FA</td>
<td>DR</td>
<td>FA</td>
<td>DR</td>
<td>DR</td>
</tr>
<tr>
<td>1</td>
<td>0.00</td>
<td>25.6</td>
<td>0.00</td>
<td>28.7</td>
</tr>
<tr>
<td>12</td>
<td>0.00</td>
<td>25.1</td>
<td>0.00</td>
<td>34.8</td>
</tr>
<tr>
<td>13</td>
<td>0.00</td>
<td>32.5</td>
<td>0.00</td>
<td>36.4</td>
</tr>
<tr>
<td>15</td>
<td>0.00</td>
<td>24.6</td>
<td>0.00</td>
<td>44.1</td>
</tr>
<tr>
<td>22</td>
<td>0.00</td>
<td>10.2</td>
<td>0.00</td>
<td>15.6</td>
</tr>
<tr>
<td>30</td>
<td>0.00</td>
<td>20.6</td>
<td>0.00</td>
<td>23.3</td>
</tr>
<tr>
<td>Average</td>
<td>0.00</td>
<td>23.1</td>
<td>0.00</td>
<td>30.5</td>
</tr>
</tbody>
</table>

The individual selves present a low DR; however, if they are integrated into a single detection mechanism the DR improves greatly. For example, if the 30 selves selected are used for failure detection, then the DR percentage improves to 71.9\% and 90.7\% for ailerons stuck at wings level trim and ailerons stuck during a bank maneuver, respectively. This configuration is able to obtain a significant improvement in DR but it also increases the FA rates approximately to 5.2\%. For this reason, other configurations were tested in order to obtain an acceptable
tradeoff between DR and FA. Two more configurations of selves were tested. The second configuration utilizes only the 6 selves shown in Table 8.9. This configuration obtained an average of 43.3% and 58.3% DR for ailerons stuck at wings level trim and ailerons stuck during a bank maneuver respectively, with FA rates of approximately 1% for both cases. The third configuration of selves utilizes only projections that have 0% FA rate for each failure disregarding FA for other types of failures. This configuration guarantees that the FA rate will be low and it also offers the possibility to use this method for identification purposes. The third configuration presents an average of 59.6% and 77.3% DR for ailerons stuck at wings level trim and ailerons stuck during a bank maneuver, respectively and less than 0.5% FA. Table 8.10 presents a summary of these results.

Table 8.10. Detection Rates and False Alarms for 3 Configurations of Selves

<table>
<thead>
<tr>
<th>Self</th>
<th>R. Aileron Locked at Wings Level</th>
<th>R. Aileron Locked at Bank Maneuver</th>
<th>Left Aileron Locked at Wings Level</th>
<th>Left Aileron Locked at Bank Maneuver</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>FA</td>
<td>DR</td>
<td>FA</td>
<td>DR</td>
</tr>
<tr>
<td>1</td>
<td>5.2</td>
<td>74.3</td>
<td>1.54</td>
<td>90.0</td>
</tr>
<tr>
<td>2</td>
<td>0.52</td>
<td>47.2</td>
<td>0.12</td>
<td>60.2</td>
</tr>
<tr>
<td>3</td>
<td>0.3</td>
<td>61.2</td>
<td>0.0</td>
<td>80.9</td>
</tr>
</tbody>
</table>

It should be noted that the four failures investigated in flight test are considered to be low magnitude, which makes them more difficult to detect. Because the locked position of the control surface is close to trim, no significant dynamic failure effects are noticeable until pilot input is provided. The DR is computed over the entire period of time that the failure is active, disregarding the presence of pilot input. This approach produces conservative low DR values as presented in Table 8.10. This effect is increased by the short duration of the flight at failure conditions, which is approximately 20 seconds.

Table 8.9 shows that single projections can only obtain a maximum of 30.9% detection rate. Nevertheless, when the projections are integrated as a single mechanism the DR increases greatly (see Table 8.10). This occurs because each projection at certain time periods of the flight test only captures abnormal dynamics when excited by certain maneuvers and commands. On the other hand, integrating several projections allows the mechanism to capture abnormal dynamic fingerprints at different periods of time during the flight tests. Figures 8.13-8.15 show the detection activity of single projections and Figures 8.16-8.17 shows the detection activity when the projections are integrated into a single mechanism. The value of 1 represents that a detector was been activated and alternatively a value of 0 determines that none of the detectors have been activated. The first 5 seconds are nominal flight test conditions while the remainder of the time corresponds to flight test data under an upset condition.
Figure 8.13. Self #1 Detector Activity for Left Aileron Locked at Bank Maneuver

Figure 8.14. Self #17 Detector Activity for Left Aileron Locked at Bank Maneuver
Figure 8.15. Self #18 Detector Activity for Left Aileron Locked at Bank Maneuver

Figure 8.16. Method 2 Detector Activity for Left Aileron Locked at Bank Maneuver
Figure 8.17. Method 1 Detector Activity for Left Aileron Locked at Bank Maneuver
References


APPENDIX A

List of Publications Resulting from the DARPA Sponsored Research Effort
Peer-reviewed Journal Papers


Peer-reviewed Conference Papers


**Graduate Student Theses**
APPENDIX B

Aircraft Subsystem ACDIE Scheme – Simulink Implementation
(on DVD)
APPENDIX C

Installation and User Manual for the Aircraft Subsystem ACDIE Scheme
# Contents

1. System Requirements ................................................................. 3  
2. Installing “AIS Aircraft FDIE” .................................................... 4  
3. Running the Demos with SimDemo .............................................. 11  
4. Running the Demos Inside MATLAB .......................................... 20
# System Requirements

<table>
<thead>
<tr>
<th>Operating System:</th>
<th>Windows XP SP3, Windows Vista SP2, Windows 7, Windows 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>OS Architecture:</td>
<td>x64 (64-bit)</td>
</tr>
<tr>
<td>Processor:</td>
<td>Intel® Core™ @ 1.7GHz or higher</td>
</tr>
<tr>
<td>Memory:</td>
<td>8 GB RAM or more</td>
</tr>
<tr>
<td>Hard Drive:</td>
<td>4 GB free hard disk space or more</td>
</tr>
<tr>
<td>Prerequisites:</td>
<td>• Microsoft® .NET™ Framework 4 or later (included with the installer)</td>
</tr>
<tr>
<td></td>
<td>• MATLAB®/SIMULINK® R2010a or later (32-bit only)</td>
</tr>
</tbody>
</table>
Installing the “AIS Aircraft FDIE” is straightforward. Run the AISAircraftFDIEInstaller.exe file on the DVD and follow the instructions that appear on the installer interface.

**NOTE:** Enter the provided serial number when the installer asks you for one.
Ready to Install

The Setup Wizard is ready to begin the AIS Aircraft FDIE installation.

Click "Install" to begin the installation. If you want to review or change any of your installation settings, click "Back". Click "Cancel" to exit the wizard.

Installing AIS Aircraft FDIE

Configuring AIS Aircraft FDIE... Done
Computing space requirements... Done
Validating install... Done
Configuring AIS Aircraft FDIE... Done
Updating component registration... Done
Creating folders... Done
Copying new files...
To uninstall or repair the installation, run the `AISAircraftFDIEInstaller.exe` file again and choose the desired operation.

After installing the “AIS Aircraft FDIE,” the installation folder should appear as shown below.
The installation folder contains the following components:

1. **Demos** folder containing a set of demo flight tests, each in its own folder. The name of each folder has the pattern `Subsystem_FailureType_FailureMagnitude`. For example, the folder `LeftStabilator_1_6` contains `.mat` data files for a flight test under the left stabilator failure, moving and locked (type 1 failure) at 6°.

2. **Flight Tests** folder containing a set of flight tests used in the development of the different AIS components. The name of each folder has the pattern `Subsystem_FailureType_FailureMagnitude_FlightPath`. For example, the folder `LeftAileron_1_2.5_123` contains `.mat` data files for a flight test under the left aileron failure, moving and locked (Type 1 failure) at 2.5°. Flight tests are categorized in folders representing the different categories of the subsystems (actuators, sensors, aileron, wing, etc.).

3. **Self Builder** folder containing a 64-bit `SelfBuilder.exe` tool for building the selves and nonselves, a 64-bit `AISConfig64.exe` tool for configuring the SelfBuilder tool, and a 64-bit `AxComReg64.exe` tool for registering the `AISDCx64.dll` COM Interop library.

4. **Self Viewer** folder containing MATLAB script files for the SelfViewer analysis tool.

5. **Selves-NonSelves** folder containing `.mat` data files of the selves and nonselves generated at West Virginia University using the Raw Data Set Union Method.

6. **Selves-NonSelves-ERAU** folder containing `.mat` data files of the selves and nonselves generated at Embry-Riddle Aeronautical University using the Cluster Set Union Method.
7. **Simulations** folder containing implementations of the DC mechanism approach and the Structured Nonself approach with MATLAB/Simulink 2010a (32-bit).

8. **Source Code** folder containing C# code projects for the DC mechanism library and other helper tools.

9. **AISConfig.exe**: A tool for configuring the different parameters needed for the building the selves and nonselves, the DC mechanism, and the FDIE Visualization tool.

10. **AxComReg64.exe**: A tool for registering/unregistering the AISDCx86.dll library. By default, the installer usually registers this library automatically. Check the main installation folder to see if it contains the file AISDCx86.tlb or not. If this file exists, then the library registration was successful. If not, then run AxComReg64.exe, click the Browse... button to select the AISDCx86.dll file, then click the Register button.

11. **FDIEVisualizer.exe**: A tool for visualizing the FDIE outcome from the simulation models. This tool will run automatically by the simulation models.
12. **SelfViewer.exe**: A tool for viewing and analyzing the 2D selves/nonselves. Be patient when running this tool for the first time as it takes few minutes to respond after doing extensive check and validation of the installed MATLAB versions that are compatible with the tool. After it responds, select the MATLAB version you want to run the SelfViewer with and click OK. This will launch the selected MATLAB command window which, in turn, runs the Self Viewer v2.1 tool.

13. **SimDemo.exe**: A tool for the running the demo flight tests automatically. See “Running the Demo Flights Using Automatically” below.

14. A set of Dynamic Link Libraries (DLLs) and a Config.xml configuration file.
Running the Demos with SimDemo

1. To run the demo flights automatically, run SimDemo.exe tool, which takes few minutes to load the FDIE Simulation Demo window for the first time run.

2. Select the demo flight you want to run, the simulation model to run the demo with, and the MATLAB version (32-bit only) to run the selected simulation. Click the Run button.

3. In the Main Window, click the Initialize button and wait until the FDIE Outcome Visualizer v2.0 is loaded. Then select Flight Scenario that corresponds to the selected demo in the FDIE Simulation Demo window (see the above figure). Finally, select Scenario #4 in the Simulation Scenario options then click the OK button.
4. In the Pilot Input Window, select All Pre-Recorded option in the Pilot Input group, and enter LOG_CMD, LAT_CMD, DIR_CMD, and THR_CMD, respectively, in the textboxes of the Data Files Names group then click the OK button.

5. The next window is specific to the category of the selected flight demo. In this example, the Structural Damage Menu window is loaded since the selected demo was a structural damage one. Select the Mass & Aerodynamic Alteration option from the Failure Scenario group then select the Right Wing option from the Damaged Surface group and enter 40 in the Failure Time textbox, 0.06 (i.e., 6%) in the Mass Alteration textbox, 0.075 in the CL textbox, 0.075 in the CD textbox, and 0.15 in the Cm textbox, then click the OK button. **NOTE:** Other windows specific to the category of the selected demo will be described later in this chapter.
6. In the Scenario #2 window, select the \((NLDI + A + EMRAN\, NN)\) “Frozen” PTNN values – EMRAN-based AFDI scheme then click the OK button.

7. In the Scenario #2 window, select the \((NLDI + A + EMRAN\, NN)\) “Frozen” PTNN values – EMRAN-based AFDI scheme then click the OK button.

8. In the next window, select one the FDIE schemes and click the OK button.
9. Click the Start Simulation button on the toolbar of the simulation model to start the demo and monitor the FDIE Outcome Visualizer tool as the simulation is running.
Windows and Parameters Specific to Selected Flight Scenarios:

1. **Nominal Flights:** There is no specific window for this group.

2. **Control Surface (Actuator) Failure:**
The following table lists the options and values for each of the installed demos that the user must specify:

<table>
<thead>
<tr>
<th>Demo</th>
<th>Selections/Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>LeftStabilator_1_6</td>
<td>Choose Locked Surface, Locked at Imposed Deflection, and Left Stabilator. Enter 40 in the Failure Time textbox, 6 in the Deflection at Failure textbox.</td>
</tr>
<tr>
<td>LeftStabilator_1_8</td>
<td>Choose Locked Surface, Locked at Imposed Deflection, and Left Stabilator. Enter 40 in the Failure Time textbox, 8 in the Deflection at Failure textbox.</td>
</tr>
<tr>
<td>RightAileron_1_8</td>
<td>Choose Locked Surface, Locked at Imposed Deflection, and Right Aileron. Enter 40 in the Failure Time textbox, 8 in the Deflection at Failure textbox.</td>
</tr>
</tbody>
</table>
3. **Sensor Failure:**

The following table lists the options and values for each of the installed sensor failure demos that the user must specify:

<table>
<thead>
<tr>
<th>Demo</th>
<th>Selections/Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roll_1_LSB10</td>
<td>Choose Large Step Bias and Roll Rate. Enter 40 in the Failure Time textbox.</td>
</tr>
<tr>
<td>Roll_LSB_1B</td>
<td>Choose Large Step Bias and Roll Rate.</td>
</tr>
</tbody>
</table>
4. Structural Damage:

<table>
<thead>
<tr>
<th>Component</th>
<th>Instructions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pitch_1_LSB10</td>
<td>Choose Large Step Bias and Pitch Rate. Enter 40 in the Failure Time textbox.</td>
</tr>
<tr>
<td>Yaw_1_LSB3</td>
<td>Choose Large Step Bias and Yaw Rate. Enter 40 in the Failure Time textbox.</td>
</tr>
</tbody>
</table>
The following table lists the options and values for each of the installed structural damage demos that the user must specify:

<table>
<thead>
<tr>
<th>Demo</th>
<th>Selections/Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>RightWing_1_6</td>
<td>Choose Mass &amp; Aerodynamic Alteration and Right Wing. Enter 40 in the Failure Time textbox, 0.06 in the Mass Alteration textbox, 0.075 for CL, 0.075 for CD, and 0.15 for Cm.</td>
</tr>
<tr>
<td>LeftWing_1_10</td>
<td>Choose Mass &amp; Aerodynamic Alteration and Left Wing. Enter 40 in the Failure Time textbox, 0.1 in the Mass Alteration textbox, 0.1 for CL, 0.1 for CD, and 0.1 for Cm.</td>
</tr>
<tr>
<td>LeftWing_1_15</td>
<td>Choose Mass &amp; Aerodynamic Alteration and Left Wing. Enter 40 in the Failure Time textbox, 0.15 in the Mass Alteration textbox, 0.18 for CL, 0.18 for CD, and 0.2 for Cm.</td>
</tr>
</tbody>
</table>
Running the demos without the SimDemo tool and directly from MATLAB requires configuring MATLAB only for the first time before running any demo. Follow these steps to configure MATLAB to run the installed simulations smoothly:

1. Run MATLAB and check if it has already been configured to use a compiler for compiling C/C++ files. If not, type `mex -setup` in the MATLAB’s Command Window and follow the setup a C/C++ compiler.
2. Change the working directory to the “AIS Aircraft FDIE” installation directory
   `[installation root]\WVU FDIE\AIS Aircraft FDIE\Simulations\WVU GEN 2 - DC and Real-Time SNSA\fdm\ann8\source`
3. In the command window, run the command `mex dcsgl2.c` to compile this file. The compiler will generate a binary file named `dcsgl2.mexw32` in the same directory.
4. Repeat Step 3 for to compile the files `emran8.c` and `vrmult.c`.
5. Move all the compiled `.mexw32` files to the parent (ann8) directory so that they appear next to their `.dll` files.
6. Redo Steps 2 to 5 for the directory
   `[installation root]\WVU FDIE\AIS Aircraft FDIE\Simulations\WVU GEN 2 - DC and Real-Time SNSA\fdm\smxl\source`
   for the files `VRMULT.C, VRPINV.C, VRSVD.C, and VRTRSP.C`
   Click the Set Path... menu item from the File menu.
7. Click the Add with Subfolders... button and browse to the folder
   `[installation root]\WVU FDIE\AIS Aircraft FDIE\Simulations\WVU GEN 2 - DC and Real-Time SNSA\fdm`
   and click OK.
8. Click Save then Close to exit the Set Path window.

Once MATLAB is configured as described above, you can run the simulations from now on by following these two simple steps:

1. Change the working directory to the folder corresponding to the flight demo you want to run. For example, if you want to run the LeftStabilator_1_6 demo, you should change MATLAB’s working directory to
   `[installation root]\WVU FDIE\AIS Aircraft FDIE\Demos\LeftStabilator_1_6`
2. In the command window, type `WVU_GEN2` and click enter.
3. Follow the same instructions described previously to interact with simulation interfaces.