AFRL-PR-WP-TR-2004-2119

MULTISCALE SOFTWARE TOOL FOR CONTROLS PROTOTYPING IN SUPERSONIC COMBUSTORS

M.Z. Pindera
M.M. Athavale

CFD Research Corporation
215 Wynn Dr., 5th Floor
Huntsville, AL 35805

APRIL 2004

Final Report for 01 July 2003 – 30 April 2004

THIS IS A SMALL BUSINESS INNOVATION RESEARCH (SBIR) PHASE I REPORT.

STINFO FINAL REPORT

PROPULSION DIRECTORATE
AIR FORCE MATERIEL COMMAND
AIR FORCE RESEARCH LABORATORY
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7251

NOTICE

USING GOVERNMENT DRAWINGS, SPECIFICATIONS, OR OTHER DATA INCLUDED IN THIS DOCUMENT FOR ANY PURPOSE OTHER THAN GOVERNMENT PROCUREMENT DOES NOT IN ANY WAY OBLIGATE THE U.S. GOVERNMENT. THE FACT THAT THE GOVERNMENT FORMULATED OR SUPPLIED THE DRAWINGS, SPECIFICATIONS, OR OTHER DATA DOES NOT LICENSE THE HOLDER OR ANY OTHER PERSON OR CORPORATION; OR CONVEY ANY RIGHTS OR PERMISSION TO MANUFACTURE, USE, OR SELL ANY PATENTED INVENTION THAT MAY RELATE TO THEM.

THIS REPORT HAS BEEN REVIEWED BY THE AIR FORCE RESEARCH LABORATORY WRIGHT SITE OFFICE OF PUBLIC AFFAIRS (AFRL/WS/PA) AND IS RELEASABLE TO THE NATIONAL TECHNICAL INFORMATION SERVICE (NTIS). AT NTIS, IT WILL BE AVAILABLE TO THE GENERAL PUBLIC, INCLUDING FOREIGN NATIONALS.

THIS TECHNICAL REPORT HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION.

/s/

MARK A HAGENMAIER
Senior Aerospace Engineer
Propulsion Sciences Branch
Aerospace Propulsion Division

/s/

LT. ADAM J. FINK
Acting Chief
Propulsion Sciences Branch
Aerospace Propulsion Division

/s/

PARKER L. BUCKLEY
Chief, Aerospace Propulsion Division
Propulsion Directorate

This report is published in the interest of scientific and technical information exchange and does not constitute approval or disapproval of its ideas or findings.

Do not return copies of this report unless contractual obligations or notice on a specific document require its return.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
April 2004	Final	07/01/2003 – 04/30/2004

4. TITLE AND SUBTITLE
MULTISCALE SOFTWARE TOOL FOR CONTROLS PROTOTYPING IN SUPERSONIC COMBUSTORS

5a. CONTRACT NUMBER
F33615-03-M-2372

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
65502F

6. AUTHOR(S)
M.Z. Pindera
M.M. Athavale

5d. PROJECT NUMBER
3005

5e. TASK NUMBER
PA

5f. WORK UNIT NUMBER
NN

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
CFD Research Corporation
215 Wynn Dr., 5th Floor
Huntsville, AL 35805

8. PERFORMING ORGANIZATION REPORT NUMBER
8562/5

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Propulsion Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright-Patterson AFB, OH 45433-7251

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL/PRAS

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-PR-WP-TR-2004-2119

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
This is a Small Business Innovation Research (SBIR) Phase I report. Report contains color.

14. ABSTRACT

The overall project objective is to develop an integrated, hierarchical design tool for simulation of unsteady flows and prototyping of flow control strategies in high-speed combustors. The software will make use of a simulation environment that will allow full accounting of the interaction of sensor-controller-actuator-combustor dynamics, in an active flow control loop. In Phase I we have developed a proof-of-concept version of such a tool. We have developed a model-free direct control strategy with on-line training and demonstrated its capabilities in controlling isolator unstart in a hypersonic combustor. Combustor flow modeling was performed using CFD-ACE+, a high fidelity multiphysics CFD code originally developed at CFDRC. The CFD code was integrated into the environment with the aid of Application Programming Interfaces (APIs). The research consisted of five basic tasks: 1) Adaptation of Simulation Environment; 2) Refinement of APIs; 3) Refinement of GUI; 4) Adaptation of Simulation Code; and 5) Demonstration of controller capabilities on isolator unstart control. All the tasks were completed satisfactorily. In particular we have shown that the proposed direct control strategy could satisfactorily prevent isolator unstart, even when finite response time of the sensors and actuators was taken into account. Hardware development and testing are recommended as part of Phase II research.

15. SUBJECT TERMS
Unsteady flow control, reduced models, dynamical systems, virtual prototyping, CFD, multi-level resolution simulations, multi-disciplinary simulations.

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT:	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON (Monitor)
a. REPORT	b. ABSTRACT	c. THIS PAGE			Mark A. Hagenmaier
Unclassified	Unclassified	Unclassified	SAR	66	19b. TELEPHONE NUMBER *(Include Area Code)* (937) 255-2089

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18

TABLE OF CONTENTS

	Page
PREFACE and ACKNOWLEDGEMENTS	v
SUMMARY	vi
1. INTRODUCTION	1
1.1 Report Outline	2
1.2 Background and Innovation	2
1.2.1 Artificial Neural Networks: An Overview	2
1.2.2 Innovative Aspects of the Research	4
1.2.3 Related Work and Potential Appliation Areas	5
1.3 Phase I Technical Objectives	8
1.3.1 Technology Challenges	9
1.4 Summary Of Phase I Research and Accomplishments	10
2. DEVELOPMENT OF SIMULATION ENVIRONMENT	11
2.1 Simulation Environment: General Features	11
2.2 Simulation Environment: Core Architecture	12
2.3 Simulation Environment: Specific Accomplishments	13
2.3.1 Objective 1: Adaptation of Existing Simulation Environment	13
2.3.2 Objective 2: Development of a Library of Component Models	15
2.3.3 Objective 3: Enhancement of GUI-based System Connectivity	16
2.3.4 Objective 4: Adaptation of CFD-ACE+	20
2.3.6 Objective 6: Sensor/Actuator Model Design	21
2.4 Summary of Environment Development	22
3. DEMONSTRATION AND VALIDATION SIMULATIONS	24
3.1 General Problem Description	24
3.2 Preliminary Simulations	24
3.3 Controls System Arrangement	27
3.3.1 Controller Configuration	27
3.3.2 Sensor Configuration and Controller Setpoint	28
3.4 Controls Simulations	29
3.4.1 Instantaneous Sensor and Actuator Response	30
3.4.2 Finite Rate Sensor and Actuator Response	33
3.4.3 Finite Rate Sensor and Actuator Response with Improved Performance Function	42
3.4.4 Effect of Controller Pre-Training	43
3.5 Summary of Simulations	44
.	45
4.0 CONCLUSIONS AND RECOMMENDATIONS	46
4.1 Conclusions	46
4.2 Recommendations For Phase II Research	47
4.3 Technology Challenges	49
5. REFERENCES	50
Appendix A Additional Results for Large Initial Mass Flow Perturbations	52

LIST OF FIGURES

Page

Figure 1.1 Schematic of a Neural Network and Its Components (a) full connected feed forward network with one hidden layer; and (b) nonlinear model of a neuron (Haykin, 1996) 3
Figure 1.2. Computational domain with sensor locations, and velocity traces (no-control/control) at the measurement points 6
Figure 1.3. PCR geometry and controller output 7
Figure 1.4. Mass fraction time history and total mass fraction errors for ANN simulation of H2-O2 explosion 8
Figure 2.1. Schematic drawing of the ANN environment module and connections to external system 12
Figure 2.2 Schematic drawing of the environment module 13
Figure 2.3. Schematic of environment extension 15
Figure 2.4. A schematic diagram of the NARX network architecture 16
Figure 2.5. Capabilities outline of the GUI. 19
Figure 2.6. Comparison of code predictions for oblique shock wave/boundary layer interaction at Mach 3 freestream conditions. 20
Figure 2.7. Schematic diagram of direct control strategy 21
Figure 2.8 System response for selected values of α 22
Figure 3.1. Overall domain geometry and injection strategy 24
Figure 3.2. Unreacting flow Mach Number fields for the two-injector configurations: a) mixing chamber; and b) injector holder. Note that the lower bound Mach Number has been set to 1 to accentuate subsonic regions (dark blue). 25
Figure 3.3. Reacting flow Mach Number fields for the two-injector configurations: a) mixing chamber; and b) injector holder. (Note the expanded Mach Number scale) 26
Figure 3.4. Reacting flow Mach Number fields for decreasing fuel flow rates 26
Figure 3.5 A schematic diagram of controls arrangement and steady state flow conditions. 27
Figure 3.6. Schematic diagram of direct control loop. 28
Figure 3.7. Comparison between 1D and 2D simulations: Mach number field. 29
Figure 3.8. Controller response as a function an input/output delay configuration. 33
Figure 3.9. Controller response as a function an input/output delay configuration for α=1000. 36
Figure 3.10. Controller response as a function an input/output delay configuration for α=100. 39
Figure 3.11. Controller response as a function an input/output delay configuration for α=10. 42
Figure 3.12. Controller response as a function an input/output delay configuration for α=50000 and setpoint request based on shock position. 43
Figure 3.13. Effect of controller pretraining on controller response. 44
Figure A1) Controller training error, combustor pressure and controlled fuel flow for initial mdot=.2kg/s, delays=3-2. This configuration leads to isolator unstart. 52
Figure A2) Controller training error, combustor pressure and controlled fuel flow for initial mdot=.2kg/s, delays=12-8. This configuration does not lead to isolator unstart 53

PREFACE AND ACKNOWLEDGEMENTS

This is a final report documenting the work performed under an SBIR Phase I research study entitled “Multiscale Software Tool For Controls Prototyping In Supersonic Combustors”. The study was performed by the Principal and Co-Investigators under Air Force Contract: F33615-03-M-2372. The work has resulted in the development of a proof-of-concept functional version of a general prototyping environment for control systems. The developed software is a stand-alone construct that was designed for coupling of controller software with user-supplied legacy codes to emulate the system to be controlled.

We thank Mr. Mark Ostrander for his valuable suggestions regarding software communications, and Ms. Jennifer Swann for preparation of this document and related project reports. We also thank Dr. Mark Hagenmaier the coordination of this project and for invaluable suggestions regarding the system dynamics, control variables, and validation and experimental issues.

SUMMARY

The overall project objective is to develop an integrated, hierarchical design tool for simulation of unsteady flows and prototyping of flow control strategies in high-speed combustors. The software will make use of a simulation environment that will allow full accounting of the interaction of sensor-controller-actuator-combustor dynamics, in an active flow control loop. In Phase I we have developed a proof-of-concept version of such a tool. The combustor and its components such as sensors, actuators and the control system are treated as a collection of interacting software components coupled by a computational environment. The components can be represented by full or reduced scale models and allows for mixed dimensionality simulations. This part of the research was synergized with another project whose focus was on the modeling of novel combustion systems.

The environment contains a number of control strategy implementations ranging from model-based indirect control to model-free direct control. During Phase I we have applied the latter approach and demonstrated its capabilities in controlling isolator unstart in a hypersonic combustor. Full-scale modeling was performed using CFD-ACE+, a high fidelity multiphysics CFD code originally developed at CFDRC. We have verified CFD-ACE+ against a density-based code FASTRAN, that in the flow regime of interest CFD-ACE+ is capable of correctly simulation the flows of interest. The CFD code was integrated into the environment with the aid of Application Programming Interfaces (APIs). A graphical "drag-and-drop" approach was developed for system component specification and connectivity. This approach was successfully demonstrated in Phase I that oversaw the preliminary environment, API and GUI development.

The research consisted of five basic tasks: 1) Adaptation of Simulation Environment; 2) Refinement of APIs; 3) Refinement of GUI; 4) Adaptation of Simulation Code; and 5) Demonstration of controller capabilities on isolator unstart control. Tasks 1-3 were done in conjunction with a related project to develop a software environment for prototyping of novel propulsion systems. Parts of that project were leveraged and adapted to this research.

1) **Adaptation of Simulation Environment To Combustor Control**
   We generalized the existing simulation environment to allow arbitrary external (CFD) codes to couple to it and interact with it and each other. The external codes can represent full or reduced models. The environment has its own internal reduced models that can interact with the external codes. The resulting software construct is a prototype of a full simulation environment coupling native and legacy codes, for concurrent simulation of component dynamics comprising a complex system. During this stage we have added multiple-input, multiple-output (MIMO) sensor and actuator model prototypes that can account for the physical delays associated with sensing and actuation processes. The models are stored in a software library, the prototype of which we had also developed.

2) **Adaptation of Application Programming Interfaces (APIs)**
   We generalized the existing communication APIs to allow more general and faster data transfer between simulation codes and the simulation environment. The APIs are small codelets that front-end the simulation codes and are responsible for receiving, conditioning, and integrating data into the appropriate data structures and arrays within

the codes. CORBA communications protocol was used for data transfer, and implemented using the open software OmniOrb libraries. OmniOrb is considerably faster than the MICO implementation that it had replaced. As in MICO, OmniOrb allows software-software as well as software-hardware communications. This is an important feature as it allows the testing of hardware controllers using software-based combustor simulators.

3) **Development of Graphical User Interfaces (GUI)**
We rewrote an existing text-based GUI to allow on-screen, drag-and-drop and connect composition of system made up of individual components. The GUI is now user-expandable, contains a stock library of components, and allows users to define and add their own component models to the GUI library. The library uses XML files for component definitions, and allows components with internal structure, or 'box-within-box' component specification with arbitrary internal recursion. This feature is essential for specification of systems such as complex actuators made up of components connected in series and/or parallel;

4) **Adaptation of Simulation Code**
The initial statement of work called for adaptation of the existing, advanced multi-physics high-speed flow solver CFD-FASTRAN for connection to the controls environment. Subsequently CFD-ACE+ was used instead since much of the communications software was already in place. For the flow regime of interest, we have shown that CFD-ACE+ results are very close to those given by CFD-FASTRAN.

5) **Software Demonstration**
We configured and tested a controller based on artificial neural network (ANN) methodology for direct control of isolator unstarts. We had conducted a limited number of tests to determine favorable ANN architecture and topology. We have subsequently demonstrated the functionality in unstart control. Flow physics were modeled using a simplified model of a hypersonic combustor. The test case was chosen in consultation with the Air Force engineers. Unstart was caused by a pressure fluctuation due to a sudden dumping of fuel in the combustor. The simulations show the feasibility of using the proposed control strategy to prevent the unstart, even when sensor and actuator inertia are taken into account.

Commercialization Plans

The developed numerical design tools will provide fast, accurate predictions of the unsteady flow combustion characteristics of existing and future Air Force combustor systems. It will also allow virtual prototyping of flow control strategies for optimization of combustor system performance and efficiency. The numerical environment has substantial commercial potential in aviation industry where these tools will be directly applicable. Hardware implementation of the controller system will allow direct interfacing with actual combustor systems, and hence their use on control of engine hardware. Generic flow and control tools will also be useful in optimization diverse multi-component systems such as those associated with thermal management, biochip design, and manufacturing processes and equipment.

1. INTRODUCTION

Operation of modern combustors involves a complex interaction of fluid dynamic and chemical processes. The processes are non-linear and time dependent; under certain conditions they can couple and lead to an unstable combustor operation. Control of such systems to ensure efficient operation is further complicated by the dynamics of the sensor/measurements and actuator systems used in the control process. Controller design is therefore a challenging task; ideally a controller should be capable of adjusting itself automatically in real time to changing system dynamics that are expected to occur during the course of combustor operation. The need for efficiency, reliability, and performance of these interrelated systems requires detailed knowledge of the system behavior during the design stages of the combustor where the designs can be modified and optimized to meet all these goals, and during the combustor operation during which the various system parameters must be adjusted dynamically. Calculations of combustor dynamics alone are very demanding because: (a) high-fidelity codes must be used to capture the combustor dynamics; and (b) reacting flow calculations are very demanding computationally. Coupled with potentially complex control systems, full scale, high-fidelity system simulations can very quickly become prohibitively expensive. A need therefore exists for development of alternative computational approaches. The present research directly addresses this need, using an innovative hybrid, mixed dimensionality approach to this problem.

Combustor subsystems consist of a number of individual components, which interact with each other; the geometric shapes of the system and components can also be very complex. Thus flow-combustion analysis of such systems for prototyping and design optimization becomes a formidable task. Present-day computational fluid dynamics (CFD) tools have found increasing use in research and designs of high-speed combustors; these have been used for analyses of complex subsystems such as internal flow in gas and liquid fuel rocket chambers, hydraulic fuel and oxidizer delivery systems, piping, valving, as well as external vehicle aerodynamics. A number of commercially available CFD codes (e.g. CFD++, Vulcan, CFD-ACE+) have been used for analysis. Use of detailed, fully resolved simulations to calculate of the fluid, thermal and combustion fields, however, is a very resource-intensive process in terms of manpower, computational resources and turnaround time. Although a critical part of the design process, use of these tools for whole system combustor design prototyping is limited due to (i) long turnaround times; (ii) difficulty in changing the geometrical configurations for design optimization; and (iii) lack of ability to change component configurations in complex systems without having to re-do the entire flow domain.

For flow optimization and combustor prototyping during the design stages, fast response and ease of use are the two major considerations, which the full CFD simulation tools cannot satisfy. Simulation tools that are based on a simpler reduced model/resolution approach are needed in this case. We propose to address these limitations by innovative coupling of full-scale model of the combustion chamber with reduced models of selected subsystems to enable efficient simulations of system scale dynamics. Reduced models will be defined using the Artificial Neural Network (ANN) methodology. The full model will be coupled to a closed loop controller that will automatically adjust the operational parameters of selected actuator models to ensure efficient combustor operation under a wide range of operating conditions. The controller design

will also be based on ANNs using stochastic training methodology, in a direct control configuration.

1.1 Report Outline

The material in this report is organized into four major sections. The remainder of Section 1 summarizes the background of system-based modeling and recent research, and discusses in more detail the study objectives and Phase I accomplishments. Section 2 documents the details of the work accomplished in terms of work scope and technical approach. Section 3 documents selected demonstration cases that illustrate various aspects of the simulation environment's capabilities. Section 4 presents conclusions and recommendations for Phase II research. Section 5 contains references.

1.2 Background and Innovation

We propose to accelerate code performance on two levels through judicious use of reduced dimensionality models. The approach is expected to significantly accelerate the simulations since the reduced model execution will be considerably faster than that associated with full models. The proposed study will utilize artificial neural nets (ANNs) for the reduced-order models and tie these with a high-resolution multi-physics solver, CFD-FASTRAN, developed by CFDRC. The coupled system will be integrated into a computing environment that will control code-to-code communications and data flow.

Solver Level On the solver level, chemical ANN will replace kinetics calculations typically performed by ODEs based calculations. One such calculation shows Pindera (1998) that for a 2-step/4 species H_2-O_2 system, a trained ANN was over an order faster than the ODE.

System Level On the system level, actuator systems will be modeled by reduced model ANN component building blocks connected in series or parallel or both to define the system. Such components could include piping sections, valves, pumps, etc. This approach will be designed to reproduce all the salient features of the dynamical systems without the concomitant complexities such as complex gridding, unclear boundary conditions, and in the presence of more complex components nearly intractable problem definition.

1.2.1 Artificial Neural Networks: An Overview

Neural networks consist of usually nonlinear automata joined to each other through adaptive connections in a massively parallel architecture. Each of the automata is a model of a neuron consisting of activation function acting on inputs to produce a desired output. Schematics of a basic network and a neuron are given in Figures 1.1a,b.

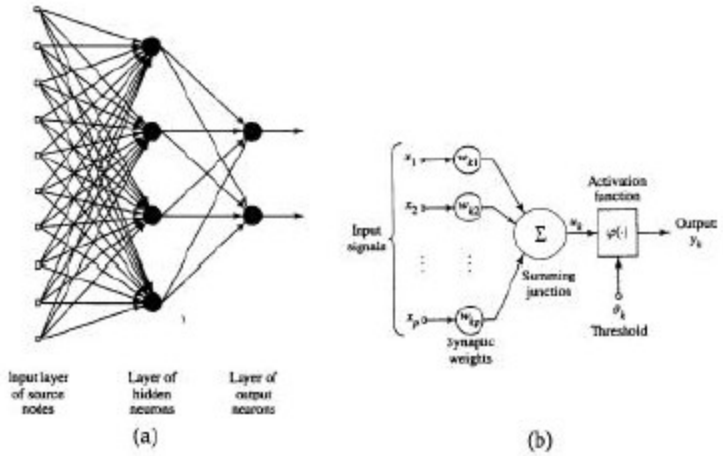


Figure 1.1 Schematic of a Neural Network and Its Components (a) full connected feed forward network with one hidden layer; and (b) nonlinear model of a neuron (Haykin, 1996)

More complicated net topologies such as recurrent networks with feedback and delay elements, as well as multi-dimensional lattice structures have also been developed. Typically, net topology depends on the problem of interest, although feed-forward topologies of the type shown in Fig. 1.1 have been widely used for a large variety of problems. The number of input and output nodes in a network depends on the problem. The number of neurons within a hidden layer and the number of hidden layers are also problem dependent and are treated as system variables. These parameters are usually determined by trial and error during the network learning process so as to minimize the complexity and maximize the generality of the network (Haykin, 1996, Rietman, 1996). However, this need for user input can be avoided through usage of networks with dynamic topologies. Such topologies will be addressed in Phase II of this project. The primary components of a neuron are the activation function and the synaptic weights. The activation function is typically a sigmoid, or for a greater dynamic range, a hyperbolic function of the input signal. Explicitly, neural networks are described by vector equations of the form:

$$\boldsymbol{y} = F(\boldsymbol{x})$$

relating an input vector **x** to an output vector **y**. In reference to Figure 1.1, the output of y_k of any neuron k is given by:

$$y_k = \varphi\left(\sum_{i=0}^{M} w_i x_i\right)$$

where x_i = neuron input; w_i = connection weight; and φ = transfer of "squashing" function

The network training process determines the synaptic weights. Training process consists of adjusting the network weight vector, **w**, is adjusted so that a desired output is obtained for a given input (e.g. Werbos, 1974; Rumelhart (1986))

Neural Net Features: Neural nets possess a number of attractive and important features Haykin, 1996, Okuda, 1997).

1. **Nonlinearity**: individual neurons are inherently nonlinear. Nets composed of such neurons are also nonlinear and are thus able to represent a very wide class of nonlinear systems.
2. **Input-Output Mapping**: nonlinear maps can be automatically constructed linking multiple inputs to multiple outputs. The linking is done through a training process allowing the net to optimize itself to any task at hand.
3. **Adaptivity**: neural nets have a built-in capability to adapt their synaptic weights to changes in the surrounding environment. A neural network trained in a specific environment can be easily retrained to deal with minor changes in operating environment conditions.
4. **Generalization**: a network can act as a sophisticated interpolant to produce output outside of the learned input data.
5. **Fast Computation**: a trained network has high computation speeds, making it viable for use in scientific simulations.

Because of their trainability, ANNs have found natural applicability in optimization problems, signal recognition, solution of inverse problems, etc. (Chen, 1996). Recent research has also extended the ANN approach to direct simulations of fluid flows, chemistry calculations, prediction of unsteady flow fields and turbulence control (Okuda, 1997, Christo, 1996; Blasco, 1998; Lee, 1997);

1.2.2 Innovative Aspects of the Research

The proposed overall research will result in an advanced software-based tool allowing a system-based sensor-controller-actuator design and optimization. The tool will have the following unique features: There are a number of innovations important to systems prototyping that this research incorporates. The more important ones include:

1. Accounting for the dynamics of the measurement process by suitably transforming the measurement variables (time delay, finite rise time, etc.) in the sensor modules;
2. Multiple sensor and sensor-type simulation capability allowing for fusion of data for improved characterization of system response (i.e., velocities, shear stress, pressure, etc.);
3. Accounting for the dynamics of the actuation process by suitably transforming the controller output variables in a dynamic ANN-based actuator module which properly emulates the dynamics of the actuation process;
4. Multiple actuator and actuator type capability allowing from Multiple Input, Multiple Output control of actuators as well as investigations of optimal actuator designs;
5. Advanced ANN-based control strategies for applications to complex flows, treated as nonlinear dynamical systems
6. Significant acceleration of combustor simulations using ANNS to calculate chemical kinetics instead of time intensive, stiff ODE solvers.
7. Cross platform portability and multi-use applications stemming from a stand-alone Java-based environment design will be possible and practical, allowing the environment to be interfaced with any FORTRAN (77, 90) based or C (C++) based simulation codes;
8. Telecomputing and rapid data exchange between the plant and the environment-based on network wide UNIX socket-based data transfer protocol.
9. Capability for simulation of components with complex internal structure using a general box-within-box subcomponent recursion

10. Capability for coupling with user-supplied full and reduced component models and CFD codes. In the current implementation, solvers coupled to the environment can be used to automatically generate data and train ANN-based reduced models during the course of a simulation.
11. User-expandable library of reduced models (ODE, algebraic, lookup tables), in addition to ANNs. The library will also contain ANN training and controls algorithms for ANN-based model design and optimization. (This capability already exists.)

A unified GUI based drag-and-drop assembly based environment will:

1. Significantly shorten design turn around times;

2. Enable changing features (such as geometry) of individual components for design optimization without affecting the overall system structure;

Facilitate the ability to change component configurations in complex systems in a quick and easy manner.

The final product will be a very versatile software tool for analysis and controls prototyping of high-speed combustors. Due to the generality of the design, the software will be of interest for a wide variety of institutional, as well as commercial, usage including aerospace, maritime, and automotive applications.

1.2.3 Related Work and Potential Appliation Areas

The investigators and their collaborators in the context of other but related projects have used ideas similar to those introduced during the course of this research. We list three examples to illustrate the potential applications of this research to other fields.

CFDRC has directly relevant experience in the development and application of multi-disciplinary, multi-physics engineering analysis software integration for aerospace vehicle design. CFDRC has extensive experience in all aspects of propulsion system hardware and soft-ware development, and propulsion system integration. We are an active developer of advanced propulsion systems hardware (ATR, fuel injectors, combustors) and design software (GEMA, NPSS, LES combustion). We are partner with major propulsion system developers (GE, Rolls Royce, Aerojet), and a participant in NASA/GRC Numerical Propulsion System Simulation (NPSS) program.

The principal investigator is the primary developer (Pindera, 2001) of a MIMO virtual controls environment, designed for investigation of control strategies of complex physical phenomena through numerical simulations. The developed ANN tools have been applied to a variety of different applications involving process control, model development of complex processes and virtual prototyping.

1. ***(MIMO) ANN Control of Flow Separation***

Example of ANN-based flow control using **direct control strategy (proposed here**) is shown in Figure 1.2. The task was to eliminate the fluctuation in the u-component of

velocity over an airfoil at a high angle of attack. In this approach, the **controller learns to control directly on line, with no prior training**. It is seen that the controller trained itself very fast and efficiently eliminated the fluctuations along the airfoil through cyclic transpiration of the v-component of the boundary layer, (Pindera, 2002).

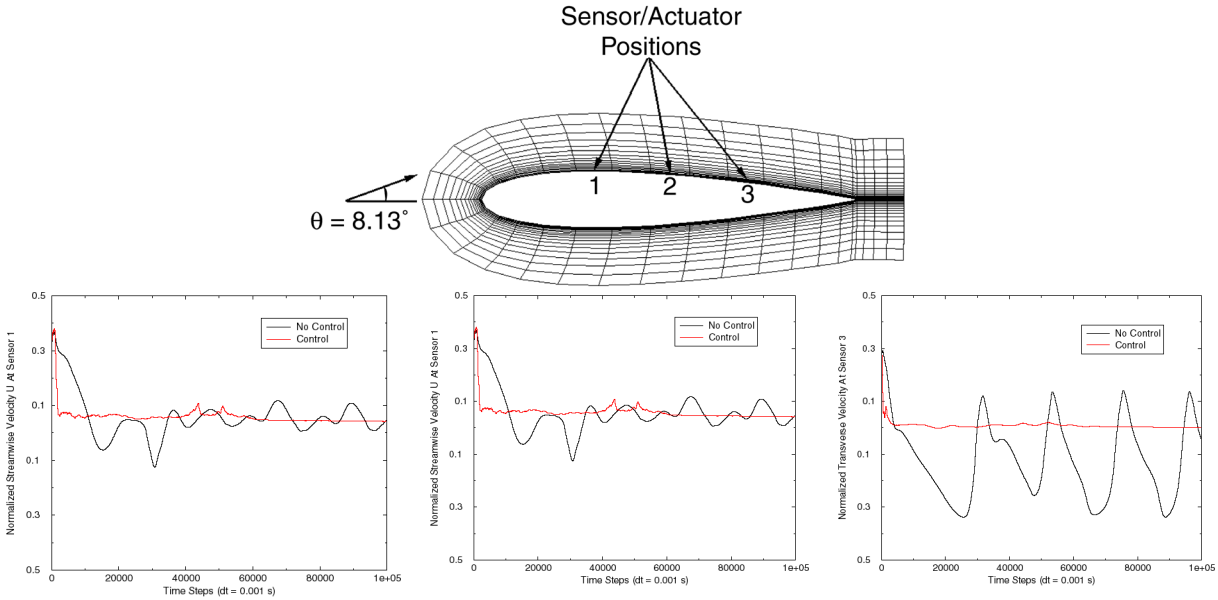


Figure 1.2. Computational domain with sensor locations, and velocity traces (no-control/control) at the measurement points

2. (SISO) Temperature Control in Polymerase Chain Reaction (PCR) Reactor

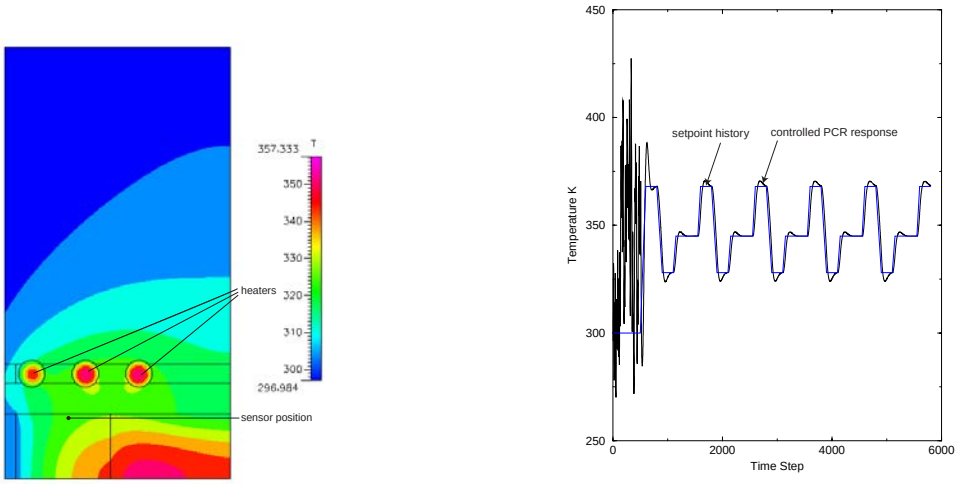
Example of ANN-based temperature control in a PCR reactor is shown in Figure 1.3. PCR reactors are used for replication of DNA strands. The operation takes place by repeated heating and cooling of the reactor, and requires precise temperature control. The controller operation depended on correct formulation of an inverse plant model of the form

$$u^n = f\left(r^{n+1}, y^n, y^{n-1}, y^{n-2}, y^{n-3}\right)$$

where

u = input heater flux; r = desired temperature; y = plant temperature at a specified sensor position; n = time lag

The physical system of the reactor is shown in Figure 1.3a. The required temperature-time curve (setpoint history) and the response of the system with an ANN-based indirect control strategy are shown in Fig. 14b. The initial oscillations in Fig. 9b represent the training period for the ANN.

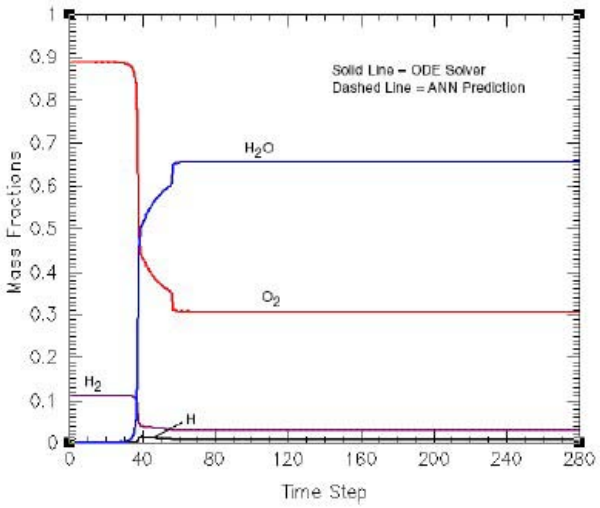


a. Reactor geometry b. Set point history, and system response

Figure 1.3. PCR geometry and controller output

3. ***ANN Characterization of Complex Systems: Chemical Reaction Modeling***

The investigators have used ANNs to model a wide variety of complex systems. One example is related to using ANN to replace the expensive, stiff ODE solvers used in chemical kinetics calculations. In this context, the ANN can be viewed as a reduced model of a complex system represented by a set of ODEs. ANN is trained with data for a 2-step H_2-O_2 reaction, and then used as the 'chemistry solver' coupled with CFD codes. The data is generated by running the ODE solver for a number of different initial conditions. ANN is then trained to predict product mass fractions and temperatures given reactant mass fractions and temperatures. Use of the subsequently trained ANN allowed very rapid calculations of the chemistry of combustion of H_2-O_2. The mass fraction histories of H_2 and O_2 with ANNs as chemistry model, and with an ODE solver chemistry model are shown in Figure 1.4, together with the error introduced by the ANN model. The ANN calculation was an order of magnitude faster for this case than that performed by a stiff ODE (Pindera, 1998).



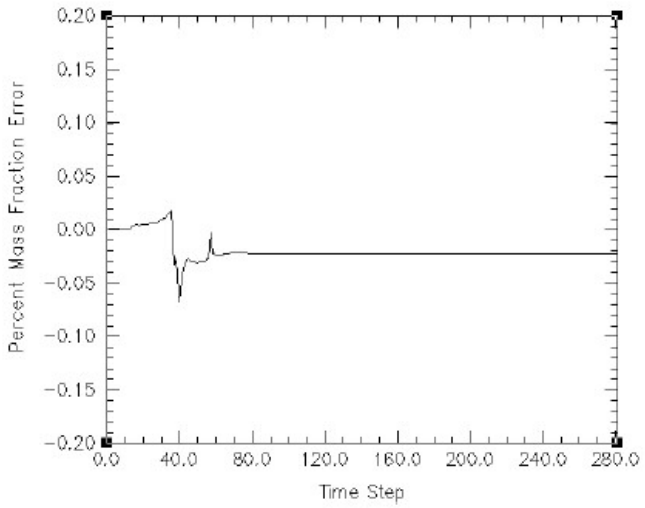


Figure 1.4. Mass fraction time history and total mass fraction errors for ANN simulation of H2-O2 explosion

1.3 Phase I Technical Objectives

The overall objective of this effort is to develop a unified multi-resolution, multi-physics hierarchical design and simulation tool for system-based prototyping of advanced sensor-controller-actuator designs for transient analysis and control of supersonic combustors. The work will focus on adaptive ANN-based control methodologies. The tool will combine reduced and full models for sensor and actuator systems from a library of components and seamlessly couple them to a high fidelity software model of the combustion chamber. Much of Phase I work involved enhancements of existing tools and adaptations and model generation needed for the propulsion system analysis and control. Phase I proof-of-concept consisted of demonstration of the software functionality and performance. Specific Phase I objectives were:

Objective 1: Adaptation of the Existing Simulation Environment

Adapt the existing controls environment for applications to combustor control. The work will entail development of reduced models to emulate control-actuator systems of varying degree of complexity. Each of the reduced models will have dynamic, multiple input multiple output (MIMO) capabilities.

Objective 2: Development of a Library of Component Models

Develop a demonstration library of 0D, 1D reduced models of sensors modules simulating the data collection process and providing input to the controller, and actuator modules simulating the functional effects of physical actuation devices.

Objective 3: Enhancement of Existing GUI-based Component Connectivity

Enhance the existing text-based **GUI** model connectivity specification tool to allow specification of complex, multiply connected actuation systems. This feature is essential for specification of systems such as complex actuators made up of components connected in series and/or parallel.

Objective 4: Adaptation of CFD-FASTRAN+

Adapt the existing, advanced multi-physics high-speed flow solver CFD-FASTRAN for connection to the controls environment.

Objective 5: Controller Design

Define and configure ANN-based controller architecture and topology for application to combustor control.

Objective 6: Sensor/Actuator Model Design

Develop a model actuator system and train an ANN-based module to emulate its functionality.

Objective 7: Demonstration Simulation on a Hypersonic Combustor System

Demonstrate the coupled ANN+CFD simulation capability on a suitable case study, defined in consultation with Air Force engineers. A candidate case is control of internal shock in a simplified model of a hypersonic engine.

Control will focus on MIMO-type strategies. Successful demonstration of the control system performance will show the feasibility and advantages of the overall virtual control systems design, as well as the potential of ANN-based adaptive flow control.

1.3.1 Technology Challenges

The major technology challenges encountered in Phase I were:

a) Design of minimally invasive APIs.
b) Insuring that the connectivity between the system components leads to proper data transfer with minimal time lags. This work has also lead to replacing MICO-based

CORBA communications implementation with a much faster OmniOrb-base implementation.

c) Realistic incorporation the interaction of reduced and full-scale models. (Alternatives include simulation of component motion, direct manipulation of field variables, or suitable representation of source and sinks).

d) Insuring that the ANN architecture and configurations learn sufficiently well the flowfield dynamics at the points of sensing and actuation.

e) Selection of time constants associated with input data sampling and controller activation so that the system is not over-controlled.

f) Sensitivity of controller to sensor placement and definition of performance function that the controller minimizes.

1.4 Summary Of Phase I Research and Accomplishments

Phase I research has resulted in a preliminary version of a prototyping software environment of sensor-controller actuator systems for general flow control. We have met all the major objectives of Phase I research successfully. We have developed a functional framework of a virtual prototyping of a wide variety of flow, combustion and control systems required for a new generation of propulsion systems.

In synthesis with a related project we have accelerated the code/controller communications protocol and developed a prototype of a general, configurable GUI with a components library that allows the user to add new or modify the existing component structure. The GUI also allows recursive composition of sub-structure within sub-structures, that can be used to define systems of increasing complexity, for example propulsion system embedded in an airframe.

For controllers we have focused on ANN-based direct control with stochastic training, although other control strategies (such as model-based) are available in the environment or can be easily implemented. We have demonstrated the feasibility of the concept of performing mixed-level, dynamic system flow and control analysis including the dynamics of sensor and controller response. We have also included the sensor and actuator response into the overall control loop and have simulated its dynamics and effect on the overall controller operation.

We have not encountered any problems other than the technology challenges listed in Section 1.3.1 above.

2. DEVELOPMENT OF SIMULATION ENVIRONMENT

2.1 Simulation Environment: General Features

The developed software environment, adapted to the current application, links and controls the reduced and full-scale computational modules comprising the proposed tool. Computational modules, at the highest level, are treated as generic black boxes with input and output channels labeled virtual sensors and actuators, respectively. Since I/O data can be specific to an individual box, an API software layer is added to precondition the data into a form appropriate to a particular box. For example, in a connection between box A and B, if the output of box A if dimensional enthalpy, and the input to box B is dimensionless internal energy, the appropriate APIs perform the require data conversion. The controller and the flow solver (plant) simulators are coupled through interchange of virtual sensor and virtual actuator data.

Data exchange is performed using the efficient CORBA communications protocol, allowing cross-platform, remote network-based tele-computing where the plant and the controller execute from different hosts. We note that *data processing* in the above relations refers to very general data operations such as normalization, inclusion of additional models, code control emulation, and others.

The external (legacy CFD or other models) and the environment codes run in parallel, exchanging data as it becomes available. The environment functionality does not depend on the origin of the input data or the destination of the output data. Note that with CORBA, the architecture of each computer taking part in the communication is hidden. The data are automatically translated to machine specific format.

Virtual sensors provide direct access to the physical data of interest in the volume conditions, as computed by the plant. Virtual actuation is modeled by variation of boundary computational cell contents, or volume conditions, as specified by the controller. A general software interface connects the plant and controller software for sensor and actuator data transfer. The controller software is written in C++, following Object Oriented Programming (OOP) code structure. The controller strategies already available exist as module software "objects". This OOP code structure makes integration of additional controller objects straightforward and independent of their structure. *Controllers not based on the ANN approach may also be integrated*. The environment was specifically designed for prototyping of plant control strategies (Pindera, 2001, 2002). Its structure and functionality is however general enough to be able to schedule and handle the interactions of subcomponents composing a general system. General structure of the coupled CFD/environment simulations is shown in Figure 2.1.

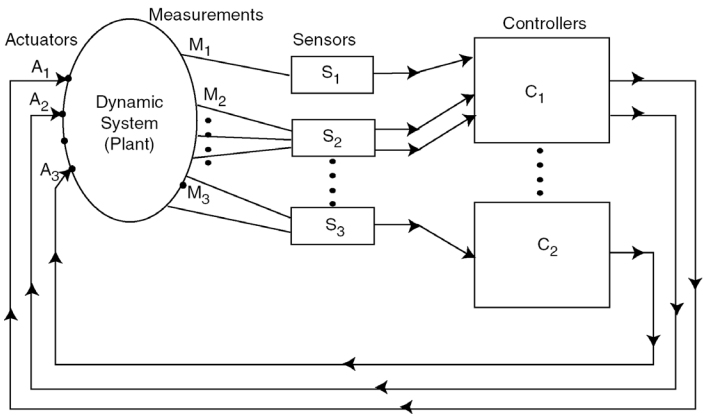


Figure 2.1. Schematic drawing of the ANN environment module and connections to external system

The CFD and ANN codes run in parallel exchanging data as it becomes available. Input data from CFD codes is read by the input Application Programming Interfaces (APIs) and is sent through appropriate transformers called sensors to a data-bus from which it is parceled out to ANN sub-modules, or controllers. The data-bus contents are continually updated at every computational cycle, as new code data is generated. The ANN models, however, read the data at intervals specified by the user. The output from the various ANN modules are then transmitted back to CFD codes or other ANN modules through the output APIs.

The original environment consisted of sensor and controller computational modules. In our implementation, data measurement is a separate process that represents direct polling of the primitive variables (pressure, temperature, velocity, etc.) as output by the plant. The function of the sensor module is to transform the measured signal in such a way as to reflect the **dynamic nature of the measurement process**. The control cycle may thus be represented by the following information flow

$$\text{CFD-CODE} \rightarrow (\text{sensor} \rightarrow \text{controller(ANN)})^{\text{environment}} \rightarrow \text{CFD-CODE}$$

2.2 Simulation Environment: Core Architecture

The computational environment was encapsulated in a stand-alone module and contains a number of internal data processing sub-modules. The sub-modules can represent reduced models as well as system controllers. The environment can therefore be used to represent system components and sub-components, with the ability to automatically control their operation. The environment is written in C++ language and has been tested on a variety of platforms using Unix/Linux and Windows-based operating systems. Its overall function is to transform output data of one code according to a specified schedule using various sub-codes, and feed the results back to another code in a closed computational loop. General environment structure and information flow is shown in Figure 2.2.

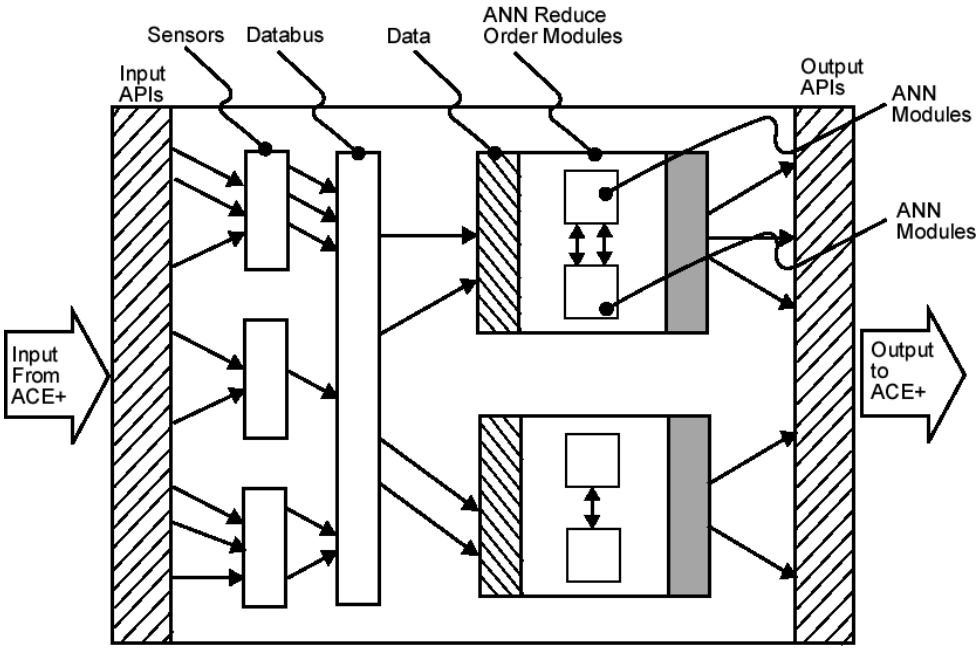


Figure 2.2 Schematic drawing of the environment module

The core of the ANN component model operation resides in the ANN sub-modules. These sub-modules can be used to represent both component reduced models and controllers. The number of ANNs, as the number of sensors, is arbitrary and is specified by the user.

In the current form all the reduced-model computational modules are based on the ANN methodology. A wide variety of ANN architectures are available for use as 0D reduced models ranging from static to dynamic. All the ANNs are based on the feed-forward Multi-Layer Perceptron (MLP) topology, with an arbitrary number of internal connections. A number of training algorithms are available, all based on deterministic or stochastic gradient-based techniques. A number of control/optimization strategies are also available ranging from direct to indirect control. The environment is not restricted to ANN technology however; in this project its primary function will be to provide communication links between the different system components.

2.3 Simulation Environment: Specific Accomplishments

The first six objectives outlined in Section 1.3 were related to the development of software infrastructure enabling controller simulations. In the following sections we list the specific accomplishments associated with these objectives.

2.3.1 Objective 1: Adaptation of Existing Simulation Environment

Simulation environment adaptation consisted of three broad tasks: 1) implementation of actuator models; 2) generalization of the environment structure to enable the interaction of multiple external codes that emulate system component dynamics; and 3) development of appropriate software interfaces that will be used by external codes to exchange data. The last two were adapted from developments associated with a related research sponsored by the Air Force to

develop a computational environment for multi-component system simulations. Selected aspects of that research are directly applicable to and can be synergized with this research.

i) Capabilities Extension. In the original representation the actuators directly manipulated the plant variables, either the primitive variables at specified locations in the plant, selected boundary conditions or sources/sinks. We have extended this definition by defining the actuators as separate computational structures appended as a data structure layer behind the controller. The structures now contain *actuator models* whose function is to transform the controller output signals in order to model the physics of the actuation process. Sensor and actuator functions are therefore analogous in the sense that they both can now represent the dynamics of the sensing and actuation processes, respectively. The simulations data flow can now be expressed in the form:

$$\text{CFD-CODE} \rightarrow (\text{sensor} \rightarrow \text{controller(ANN)} \rightarrow \text{actuator})^{\text{environment}} \rightarrow \text{CFD-CODE}$$

This capability allows us to model all the actuator dynamics in a very compact form, encapsulated in an ANN-based module, and part of the simulation environment. The implementation is general enough to enable a connection of multiple ANNs to compose an actuator system made up of subcomponent building blocks such as valves, pipes, heat exchangers, etc.

ii) Environment generalization. Originally, the environment was designed to interact with only one external (CFD or other) code. During Phase I, we have generalized the environment structure allowing an arbitrary number of codes to interact with the environment and each other, and giving the codes direct access to the individual internal reduced models for further data processing. The new architecture is shown in Figure 2.3 below.

Each sub-environment containing internal reduced models is a replica of the structure shown in Figure 2.2. The sub-environment data structures are independent of each other but can fully communicate and otherwise interact with each other. This structure gives us a very wide flexibility regarding the types of systems that we will be able to analyze. For example, the environment may be used to represent a vehicle airframe (described by a 6-DoF model) with the sub-environments representing components of the propulsion system. This capability will become important for simulations of combustion control under various flight mission profiles.

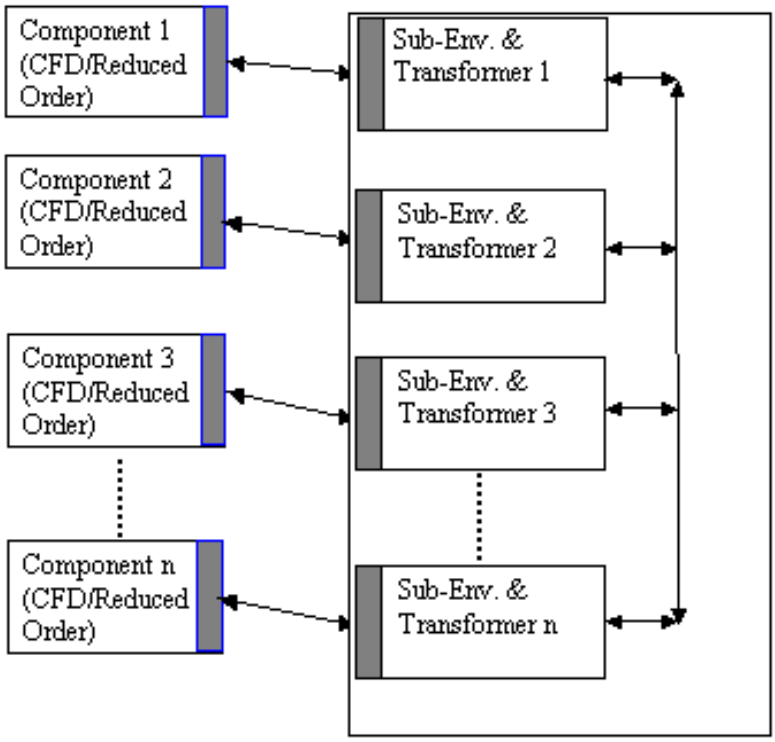


Figure 2.3. Schematic of environment extension

iii) Code communications. Input data from CFD codes such as CFD-ACE+ (CFDRC) and Overflow (NASA), or other component models is read by the input APIs. We have extended the existing APIs to enable communication between codes that use different solution methodologies such as pressure-based and density-based codes. The APIs thus handle not only the logistics of data exchange but also data conditioning to ensure that the data received is compatible with the code that is using it. Additional code was developed to ensure the conservation of mass, energy and momentum during the data exchange process. This issue is discussed in more detail in Section 2.3.4.

2.3.2 Objective 2: Development of a Library of Component Models

In conjunction with Objectives 1 and 3 the existing environment data structures have been generalized to include actuator models library. The GUI is used to access the different model types. Component-to-component connections are also performed by the GUI (Section 2.3.3). The sensors library was already in place. The sensor/actuator dynamics for any variable Y are currently described in terms of time lags, and represented a simple ODE of the form $dY_{sensed}/dt = \alpha(Y_{true}-Y_{sensed})$. More complex, ANNs-based reduced models of the actuators and sub-components are currently being developed. The ANNs used in reduced model and controller formulations (the latter discussed more fully in Sections 2.3.5 and 3.3.1) are given by very general dynamical representation using the NARX (Nonlinear Autoregressive with Exogenous Inputs) architecture as shown in Figure 2.4.

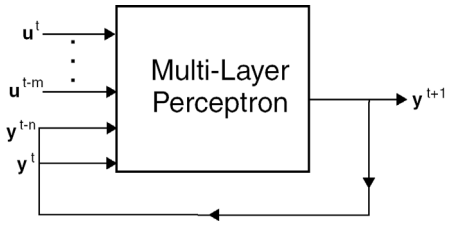


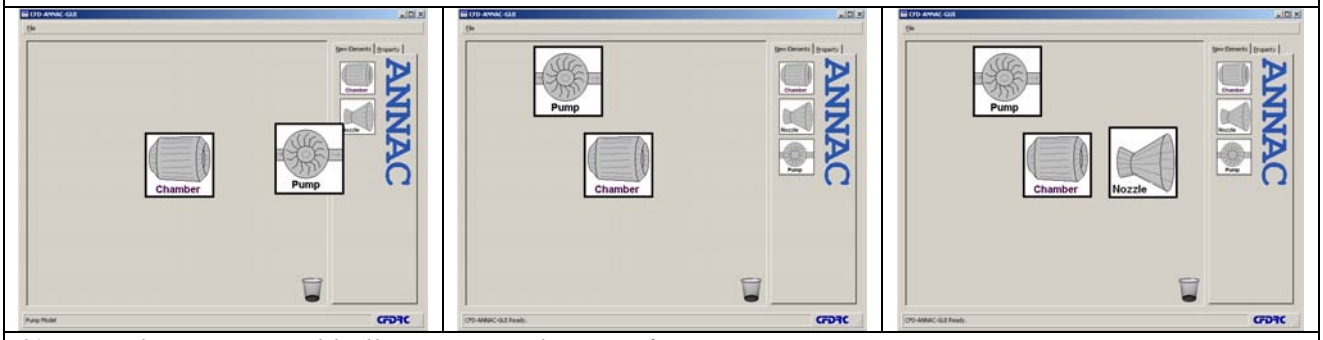
Figure 2.4. A schematic diagram of the NARX network architecture

Despite its simple design, which includes only global feedback, NARX networks exhibit very powerful computational properties under fairly general design conditions, and can be used to represent complex dynamical systems. For example Narendra (1995) has shown that for observable systems, NARX models are equivalent to fully recurrent state-space models. In this context, observability refers to whether "the state of the network can be determined from a finite set of input/output measurements". Theorems of Seigelmann et al. (1997) and Giles (1996) show that under general conditions, NARX models are Touring equivalent. In addition, Lin et al. (1996), have shown that NARX networks have attractive properties regarding the learning of long-range dependencies. System component representations using CFD codes (external models) configured to perform 1D-3D simulations are not formally archived, but are assumed to be provided by the user and available to the GUI. The GUI then displays a 'library' of the different codes available for integration to represent a complete system. The library has an open architecture allowing for the addition of user-specified models. The library currently contains 0D ANN-based reduced models, CFD-ACE+ (developed by CFDRC), and the Overflow (developed by NASA) codes.

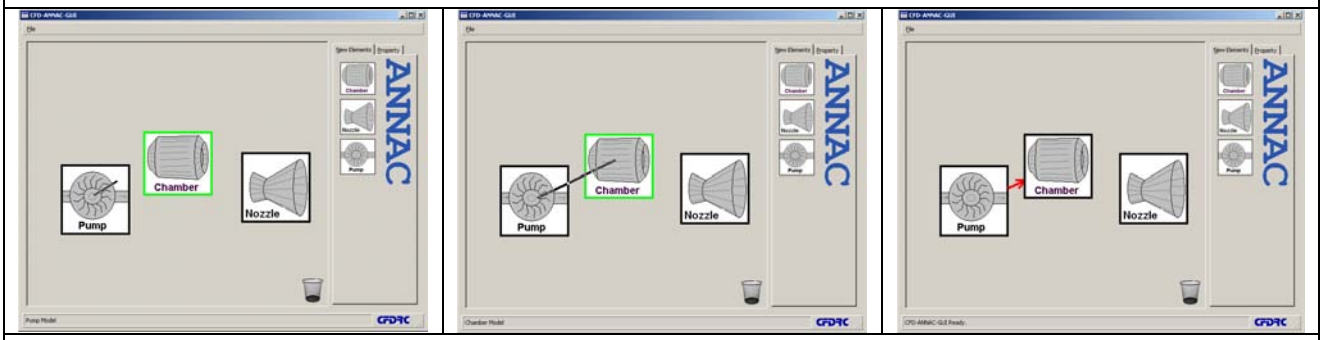
2.3.3 Objective 3: Enhancement of GUI-based System Connectivity

The original JAVA-based and script oriented GUI has been completely rewritten using the FOX GUI-building library, allowing system composition and sub-components definition using on-screen, object-based drag-and-drop-and connect approach. The FOX environment is an open-source library of C++ routines from http://www.fox-toolkit.org, as well as from CFDRC's webpage (http://www.cfdrc.com). The library is completely portable on most platforms, including Windows and Unix/Linux. The GUI developed in Phase I contains a standard and user-specifiable component library, and can be customized using independent *XTM* configuration files. In the GUI, each component is treated as an element with the following attributes: input, output, type, and internal structure. Element type refers to its purpose as a means of transferring data, a component model (reduced or CFD code), or a controller. The elements can be arbitrarily connected subject to restrictions that can be specified in a (hidden) box configuration file. The internal element structure allows elements to be composed of internally connected sub-components. Because the sub-components themselves are allowed internal structure, the overall system configuration allows for *box-within-box* infinite recursion regarding internal system details. This feature directly allows whole system simulation with selective focusing on the details of internal operation of selected sub-components. This is work in progress that will be completed during Phase II. Examples of the GUI's current capabilities are shown in Figures 2.5-a-i, below.

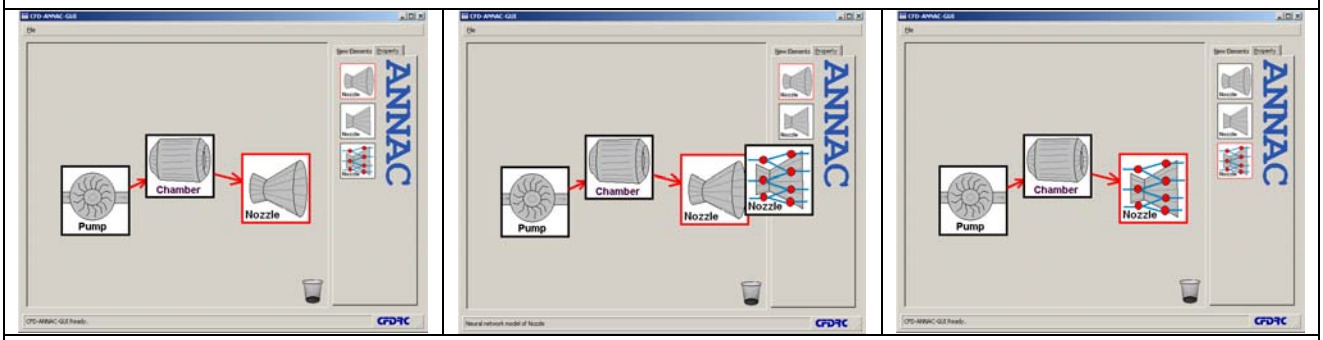
a) Adding new component to project, drag element from library new element bar to project space



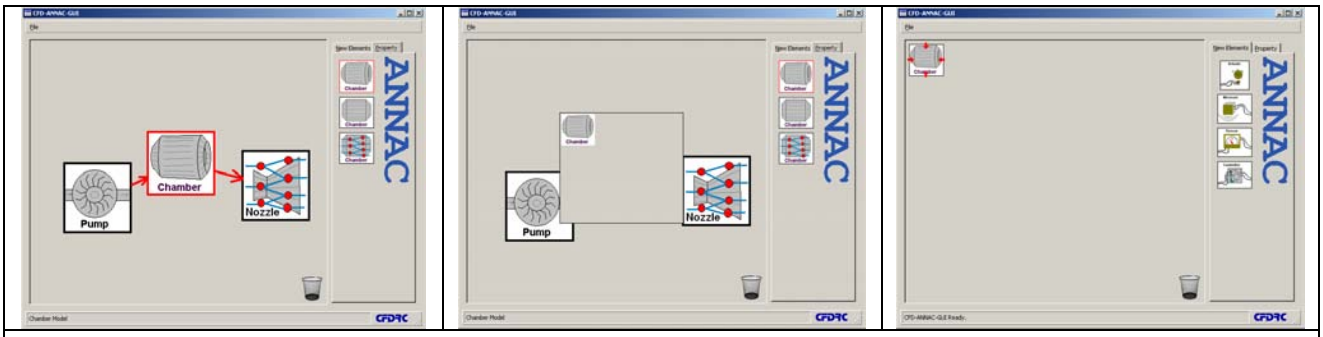
b) In analogous way add all necessary elements for given project



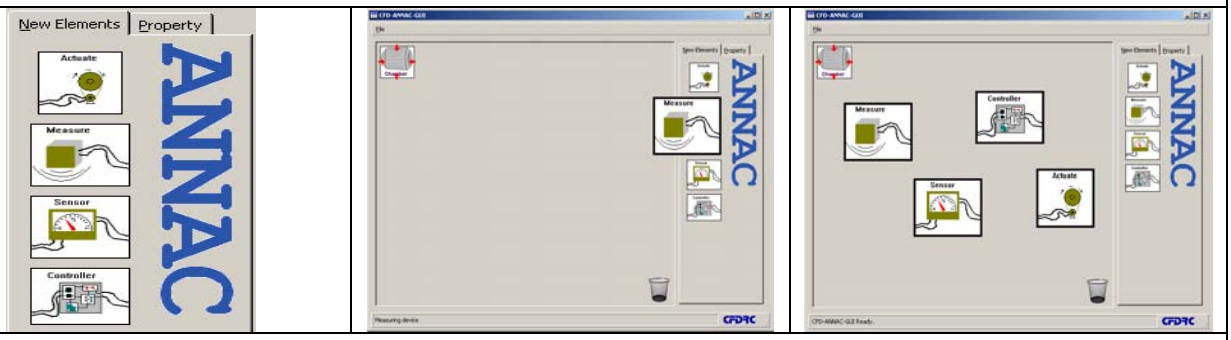
c) Connection between components is done graphically using a mouse. Connection restrictions may apply.



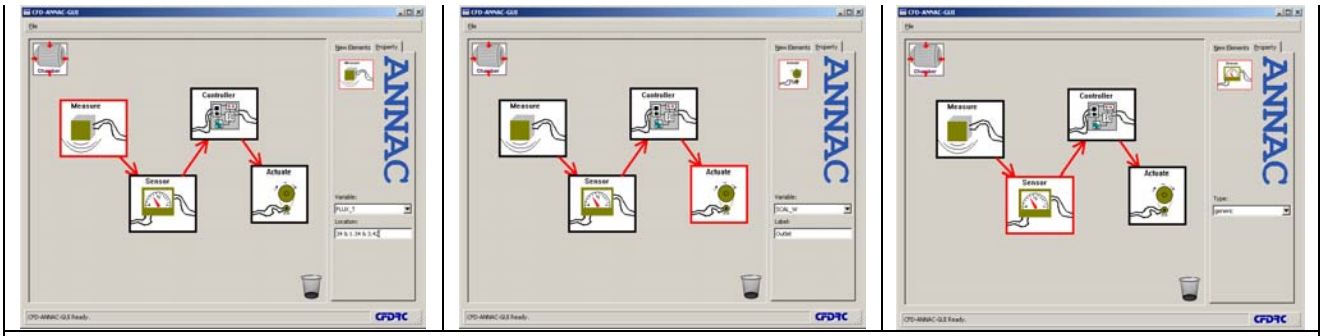
d) Element properties can be changed on-screen. Elements can be represented in simulation by different full 3D, 2D, or ANN models.



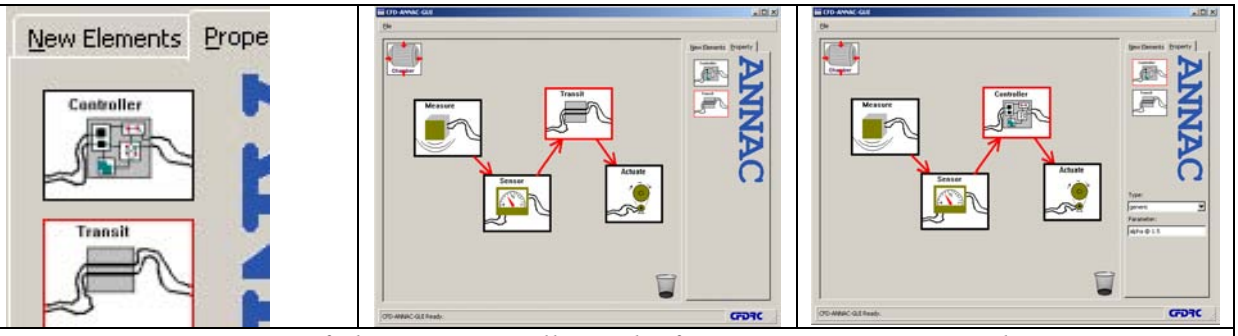
e) Double clicking on element open another window with it internal representation. This window allows defining internal connections inside given element and defining its properties. The symbol of open element is shown in left top corner of new window.



f) The 3D and 2D models of element can contain: Measurement, Sensor, Controller, and Actuator. The internal model of each element is build in the same way as main project through dragging element icons into main window space, and using the mouse to define connectivity.



g) The property panel allows defining for given element all necessary parameters. The Measurement and Actuator elements are defined through providing type of physical value (flux, pressure, etc) and location inside associated 3D/2D element model. The Sensor converts measured signal to one expected by an abstract Controller. The only parameter currently defined for Sensor is its type. Sensors may be used to represent generic data transfer (output=input), analogues of physical sensing processes associated with devices such as time-delayed thermocouples, pressure sensors and others. The latter types may be used in the prototyping of physical control strategies.



h) There are two main types of abstract Controllers. The first transmits input signal to output without modification. The second performs real control functions. Its structure and functionality is specified using additional text windows.

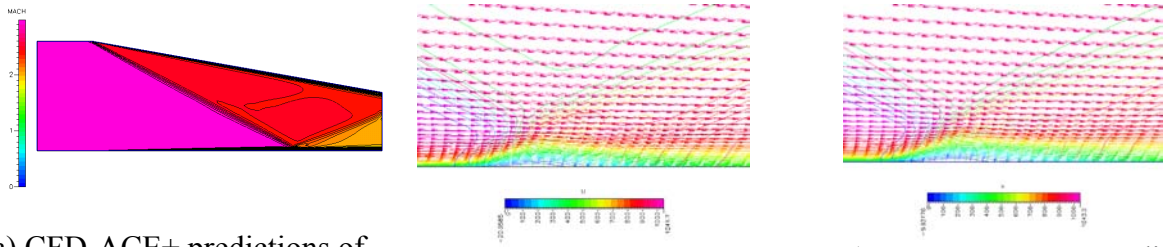
i) Double clicking on open element symbol shown in left top corner of window, closes internal element model window and returns to main project edition.

Figure 2.5. Capabilities outline of the GUI.

2.3.4 Objective 4: Adaptation of CFD-ACE+

We had anticipated performing the simulations using CFD-FASTRAN, a characteristics-based code, developed and commercialized by CFDRC. The objective of this task was to develop and install appropriate communications in this code to enable code-controller data exchange.

In consultation with the Air Force engineers we had amended this objective and used instead CFD-ACE+, a SIMPLEC-based (pressure-based) code developed at CFDRC (2002). For the type of the Mach number flows considered during Phase I, we have found that CFD-ACE+ can perform as well as FASTRAN. Figure 2.6 shows a comparison of predictions of the two codes for a shock wave/boundary layer interaction caused by an oblique shock at Mach 3 freestream conditions interacting with the opposite duct wall. Figure 2.6a showing the overall flowfield was generated by CFD-ACE+. Figures 2.6b,c show details of the interaction region on the lower wall. The predictions of the two codes are seen to be essentially the same. We elected to use CFD-ACE+ because the latter already had some aspects of the required communications available, giving a good base for code generalization.



a) CFD-ACE+ predictions of the overall flowfield

b) CFD-ACE+ predictions in the interaction region

c)CFD-FASTRAN predictions in the interaction region

Figure 2.6. Comparison of code predictions for oblique shock wave/boundary layer interaction at Mach 3 freestream conditions.

i) CFD-ACE Modifications: For high speed or compressible flows, additional data processing and boundary condition treatment is needed to ensure that the mass, momentum and energy fluxes are conserved during the data exchange between boundary patches. Account also has to be made for the predominant flow direction at the boundary patches to ensure that the compressible flow physical conditions are satisfied. We have developed iterative boundary values update and correction procedures to ensure this compatibility. Specifically, at an inlet boundary, we treat the mass flux as a primary conserved variable and iteratively adjust the total/stagnation pressure at the boundary until the transmitted and the calculated values of the mass fluxes match. Once this condition is satisfied, the remaining fluxes (momentum, energy and species) are automatically conserved. The overall mechanics of this boundary condition procedure will allow its use in both pressure as well as density based flow solvers. This procedure will serve as a prototype for the development of generalized API's for joining of codes using unlike solution procedures.

2.3.5 Objective 5: Controller Design

On-line direct control strategy was used for optimization of performance of the system. We had chosen this approach since the controller is simple to implement and requires no plant model. A schematic of a prototypical configuration for direct controller strategy is shown in Figure 2.7. We had chosen the recurrent NARX ANN to model the controller itself.

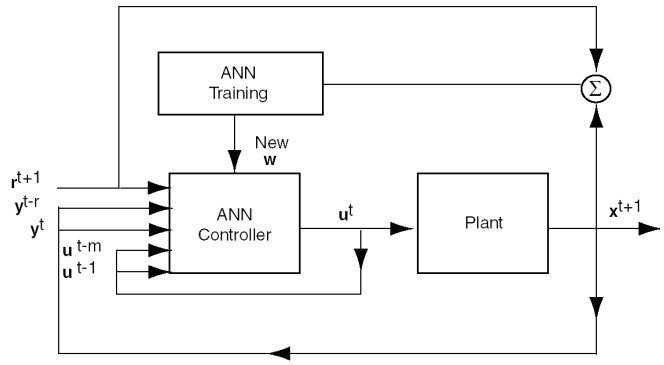


Figure 2.7. Schematic diagram of direct control strategy

Until recently, no general methods existed for direct adjustment of controller parameters to satisfy the specified plant setpoint trajectory in the context of direct control. The difficulty is due to the presence of an unknown system (plant) between the controller and the output error. To a large extent the problems associated with direct control may be addressed using system perturbation methods.

We have used one such approach, SPSA (Simultaneous Perturbation Stochastic Approximation), developed by Spall (1992) for evaluation of the error gradients, for unknown, noisy systems. SPSA has been successfully used for a variety of ANN training and control problems by a number of researchers (Rezayat, 1995; Chin, 1994, 1997; Maeda, 1997; Spall, 1997a,b; 1998a,b; Pindera, 2002). Both the direct control module and the SPSA evaluation algorithm are available in the simulation environment. The design work involved general data flow tests and parametric runs to determine general controller behavior and response to variation of parameters such as ANN topology (number and distribution of neurons), number of time delays of sensed and feedback variables, selection of training parameters, number and location of sensors. This work was done in conjunction with that covered by Objective 6, described below.

2.3.6 Objective 6: Sensor/Actuator Model Design

The sensing and actuation system types had been selected in coordination with the technical monitor for demonstration in Phase I. On a preliminary basis, we have selected the simulation of a fuel injection modulation system for the demonstration for which work has been performed previously at CFDRC for both gas turbine combustor instability control and ANN control system design. Sensing was based on pressure values/distribution in various parts of the domain.

The intent of this Objective was to develop models of sensing and actuation processes that realistically incorporate inertia effects, in order to investigate the influence of these effects on the control process. We had initially intended to use ANNs in the NARX configuration to model sensors and actuators. We have subsequently found that a much simpler, ODE-based approach

can be used. Sensing and actuation dynamics are now modeled using a simple equation of the form

$$dY/dt = \alpha(Y_{true} - Y), \text{ for any variable } Y$$

In the above, α is the device time constant, and is a measure of inertia-induced delay. Figure 2.8 shows a typical device response Y to a specified sinusoidally varying input Y_{true}; the dependence of the system response Y on α is clearly seen in this Figure.

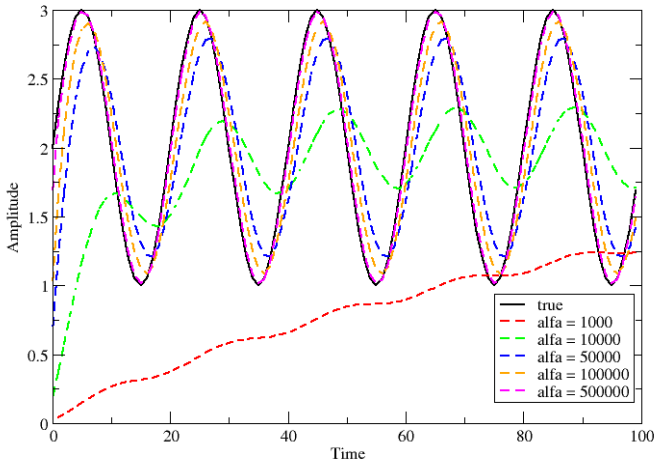


Figure 2.8 System response for selected values of α

For a dynamic system of the form above, a higher response delay also leads to lower overall values. Figure 2.8 and the α values used therein are purely illustrative. Results in later Sections will show that for the flow conditions considered for this project, the values of α were considerably lower.

2.4 Summary of Environment Development

We have presented the details of software developments in the design of a proof-of-concept version of a software environment for prototyping, simulation and control of complex systems composed of interacting internal components. The complete software is composed of a simulation environment, a GUI, and APIs.

The simulation environment is used to couple arbitrary number of external (user supplied) codes that represent system components, both full and reduced scale, and is also responsible for data transfer synchronization. The synchronization is based on the time increment associated with the solution, or on the number of iterations per time increment. The environment also hosts internal computational modules that can be used to define additional reduced scale component models or controllers that control component dynamics. These modules are stored in a library that can be accessed through the GUI.

The GUI is used to define the system of interest, and is responsible for component-to-component connections. The developed GUI prototype can define components with internal structure, expressed in terms of linked sub-components. The level of recursion in such box-within-box construction is arbitrary. Details of full component-to-component connectivities will be completed in Phase II. Component structure is defined using text-based XML files. Users are therefore able to append additional components to the standard component library with minimum effort. In Phase II, standard component templates will be developed to facilitate such developments.

The APIs are responsible for the actual data transfer mechanics and for loading of the transferred data into the appropriate data structures in the various component models/solvers. APIs are appended to solvers of interest, and were designed to minimize the need for source code modification. APIs are code dependent since different codes may have different solution procedures, and may require different forms of data. Limited data conditioning procedures were developed to enable general pressure-based and density-based codes to exchange data in a seamless manner. To date the APIs have been implemented into CFD-ACE+ (pressure-based) and Overflow (density-based).

3. DEMONSTRATION AND VALIDATION SIMULATIONS

3.1 General Problem Description

In consultation with the Air Force engineers we have tested the controller software on a model of a hypersonic combustor system. The controller task was to control the fuel flow rate in order to prevent isolator unstart caused by sudden pressurization of a combustor in a scramjet engine. This may occur under conditions of dumping of fuel in the chamber precipitated by a sudden thrust requirement.

The combustor geometry was obtained from the Air Force and is shown in Figure 3.1. Nearly 6600 cells were used to discretize the initially 2D domain. Two fuel injection strategies were investigated: injection through the top of the mixing chamber (original position) and injection from slender injector holders placed in the middle of the combustion chamber (trial position) as shown.

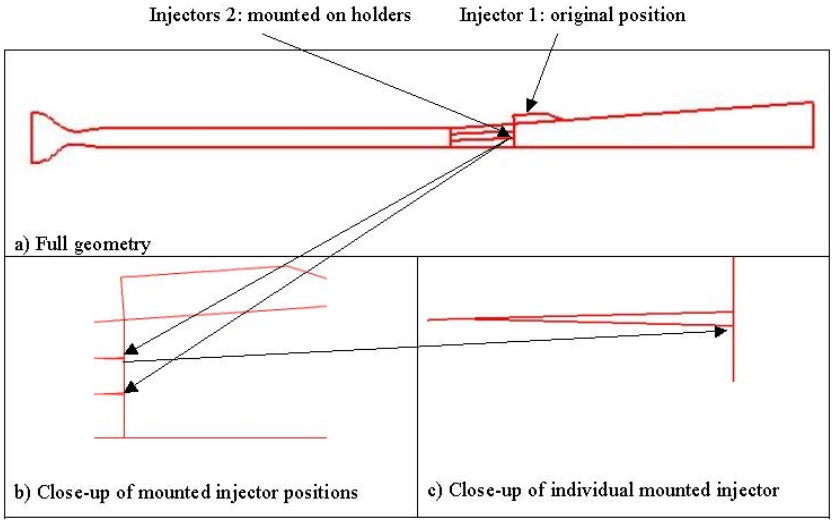


Figure 3.1. Overall domain geometry and injection strategy

We performed simulations for unreacting and reacting flow to in order to choose the best fuel injection strategy that would result in a reasonable combustion process. Subsequently we performed controls and flow response simulations on the chosen configuration. A description of the work performed is given in the following.

3.2 Preliminary Simulations

We have performed 2D simulations to determine the best injector placement and to determine the system response to variations in fuel mass flow rates. All the simulations were performed using CFD-ACE.

Injector placement is a rather critical part of the overall system geometry. The 2D flow that we are considering is expected to provide fuel-mixing characteristics considerably different from its

3D counterpart. In the 2D case the fuel is injected as a sheet that offers a restriction to the surrounding flow. Conversely, a 3D jet offers only partial blockage. The alternate geometry of injectors specified in the middle of the combustor was expected to avoid this blockage problem.

We conducted the first set of simulations for non-reacting flow, in order to determine the influence of the injectors on the general flow field. The boundary conditions were set as follows:

Inlet P_{total} = 200,000 Pa; T=700C
Outlet P = 10,000 Pa.

These conditions ensured supersonic flow in both the isolator, combustor and outlet sections of the engine. We note that the upstream pressure is on the low end of the range suggested by the Air Force, which was 160-800kPa. As will be seen in the following, the reacting flow simulations are therefore rather sensitive to the fuel flow rate. Steady state unreacting flow field for the two-injector configurations are shown in Figures 3.2. We have expanded the color scale to better accentuate the differences.

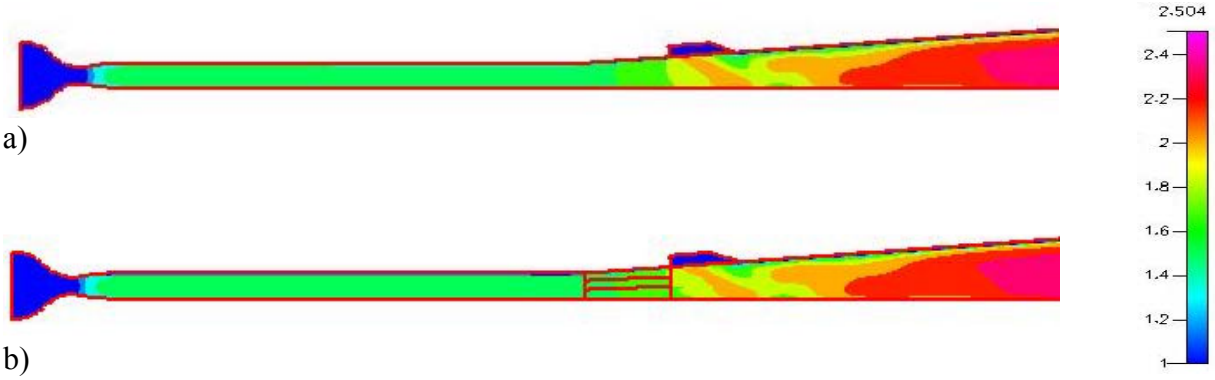


Figure 3.2. Unreacting flow Mach Number fields for the two-injector configurations: a) mixing chamber; and b) injector holder. Note that the lower bound Mach Number has been set to 1 to accentuate subsonic regions (dark blue).

These simulations indicate that the flowfields are virtually identical and that the model injector holders disturb the flowfield only minimally.

We performed a next set of simulations in order to determine the effect of the two injection strategies on the structure of the combustion zone. Methane was used as fuel. At the specified flow conditions the stoichiometric fuel flow rate corresponds to approximately 0.8kg/s. We have started the reacting flow calculations with a lean mixture, arbitrarily setting the fuel mass flow rate to 0.5kg/s.

For quick turnaround time, we assumed instantaneous reaction. For these types of calculations, where the general combustor characteristics are of interest, detailed modeling of chemical reactions is not the primary focus. The results are shown in Figure 3.3.

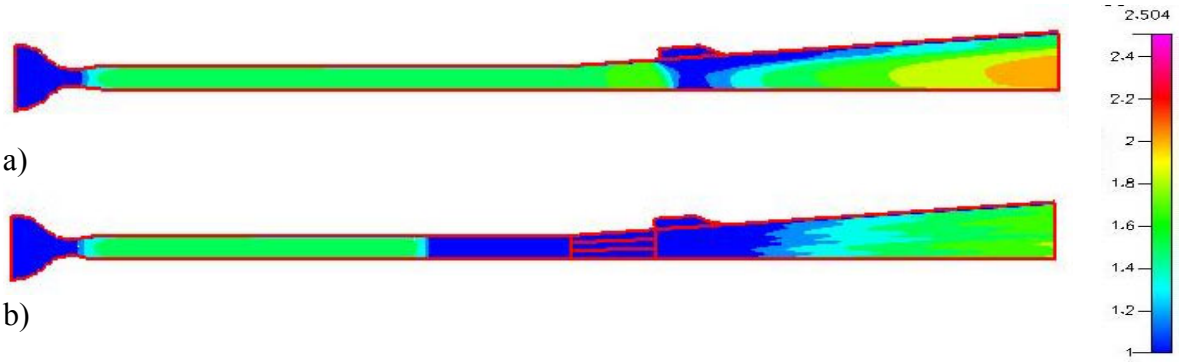


Figure 3.3. Reacting flow Mach Number fields for the two-injector configurations: a) mixing chamber; and b) injector holder. (Note the expanded Mach Number scale)

Figure 3.3a shows that injection from the top of the mixing chamber leads to a very non-uniform and non-realistic combustion zone. Most of the reaction zone is localized to the upper portion of the exit nozzle, as the emerging fuel sheet is swept downstream by the surrounding flow. This flow has reached steady state.

Figure3.3b shows the flowfield structure with the centrally located injector holders. The reaction zone is well mixed and the resulting increased chamber pressure tends to propagate into the isolator. This flow has not reached equilibrium and Figure 3.3b represents a solution snapshot after approximately 3600 time steps. As the solution advances, the subsonic region propagates into the isolator until the flow becomes completely subsonic at steady state conditions. The above results indicate that injection from the top leads to unrealistic results. In the subsequent simulations we have used the central injectors only.

We conducted a final set of preliminary simulations in order to verify the proper response of the flowfield to variations in fuel flow rate. Figures 3.4 show the results after approximately 3600 time steps.

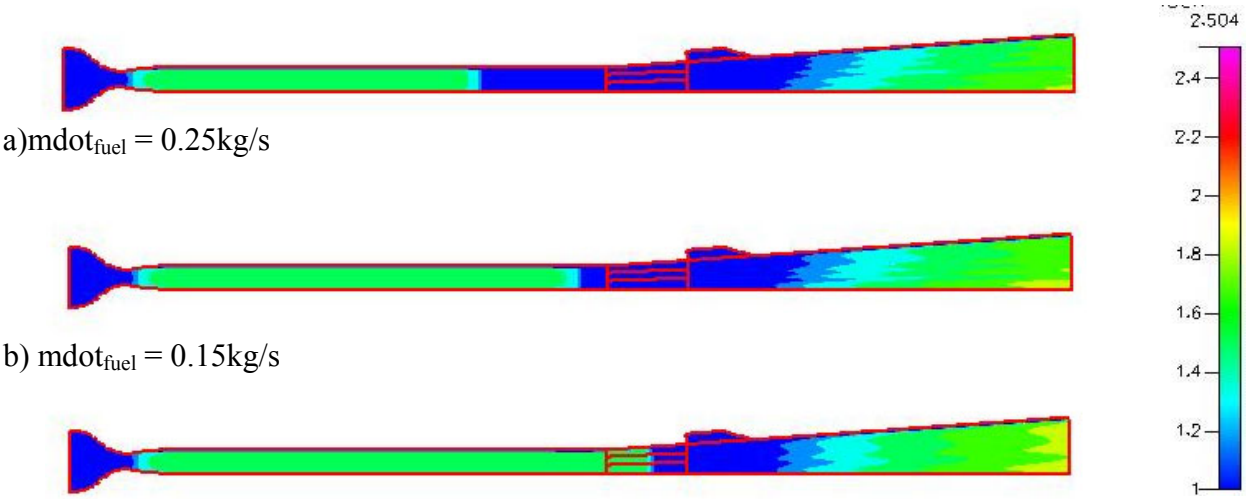


Figure 3.4. Reacting flow Mach Number fields for decreasing fuel flow rates

Figures 3.4 show that that as expected, decreasing the fuel flow-rate reduces the penetration of the subsonic region into the isolator. Note that Figure 3.4c show the flow field at steady state: the isolator is completely supersonic.

The above simulations indicate that the combustor system is responding reasonably to the effects of fuel addition.

3.3 Controls System Arrangement

The controls simulations were performed with the 1-D combustor representation. As described in Section 3.3.2 this was done to maximize the simulation turnaround time. As noted, the salient features of 1D and 2D simulations were very similar.

In the simulations, the controller input was based on the average reading from the pressure sensors in the combustion chamber as shown in Figure 3.5.

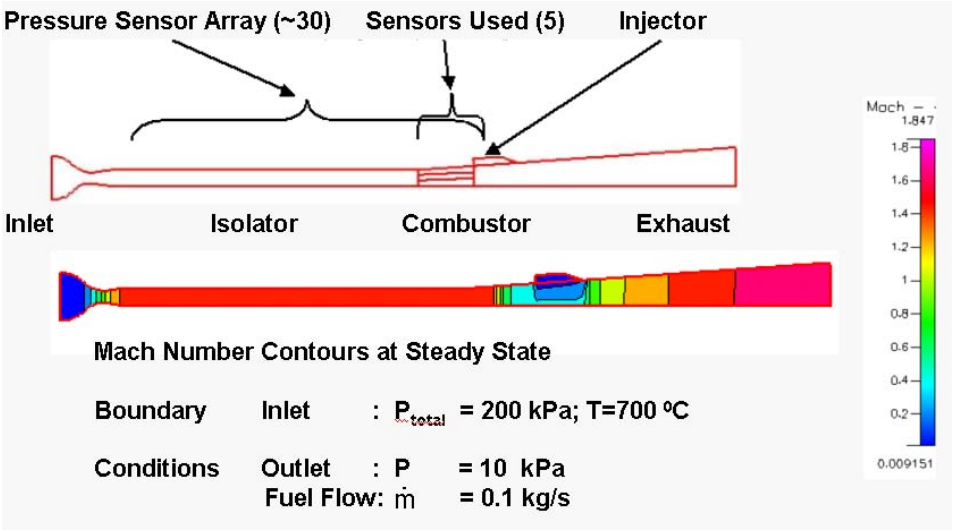
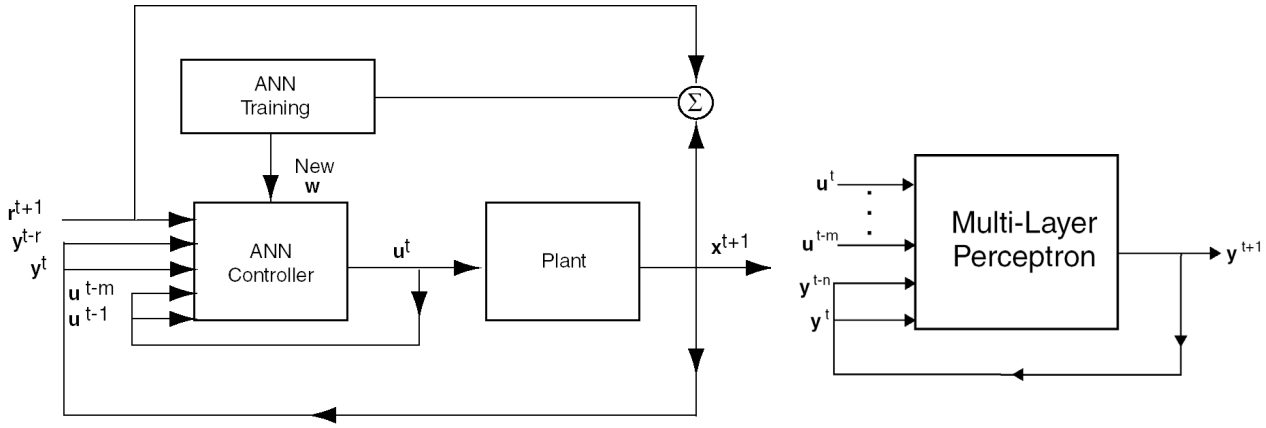


Figure 3.5 A schematic diagram of controls arrangement and steady state flow conditions.

Controls simulations were started by dumping additional fuel into the combustion chamber operating at steady state. The controller task was to adjust the fuel flow rate in order to maintain the sensed regions at a specified pressure level at which the isolator is known to be in the proper operational mode, i.e. that the isolator is not unstarted.

3.3.1 Controller Configuration

We have focused on ANN-based direct control with stochastic training. This type of control design does not require a plant model and the controller learns the controlled system dynamics during the course of operation. A schematic drawing of the control system arrangement, and the ANN controller are shown in Figure 3.6.



a) Direct Control Strategy b) ANN controller with delay taps

Figure 3.6. Schematic diagram of direct control loop.

The controller has several adjustable parameters that must be specified:

a) Internal ANN topology (number of neuron layers and neurons/layer)
b) Number of input-output delay taps
c) Two internal parameters associated with stochastic training.

The number of time delay taps is one of the keys to successful controller operation. The time delay elements endow the controller with a degree of memory such that it in effect becomes a dynamical system that can control other complex dynamical systems. During the course of this work we did not attempt to optimize the controller. We also performed a limited number of parametric tests to illustrate the operation and to show that even un-optimized controller designs performed remarkably well. The work involved general data flow tests and parametric runs to determine general controller behavior and response to variation of parameters such as ANN topology (number and distribution of neurons), number of time delays of sensed and feedback variable, selection of training parameters, number and location of sensors.

In generating the results documented in this report, we used the following controller parameters:

a) Internal ANN topology (number of neuron layers and neurons/layer)
   Input layer - average pressure, instantaneous and delayed (see b, below)
   1st hidden layer - 15 neurons
   2nd hidden layer - 10 neurons
   Output layer - 1 (mass flow rate in the combustor)

b) Number of input-output delay taps: 1/0, 2/1, 4/3, 8/7,12/11

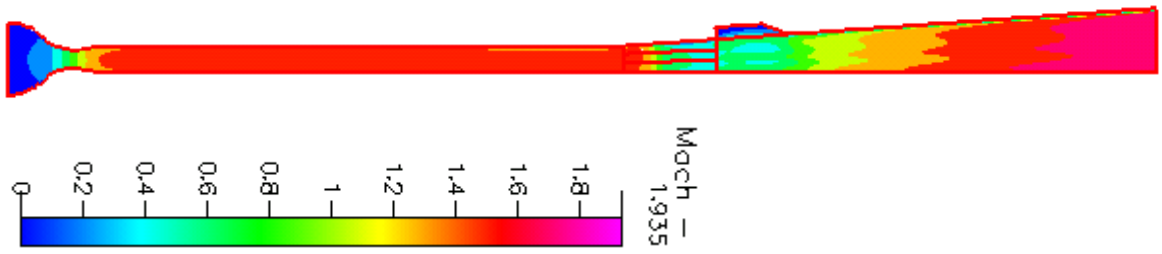
3.3.2 Sensor Configuration and Controller Setpoint

After selected numerical experiments, we connected the controller to (the average of) five pressure sensors located along the extended combustion chamber. We have configured the controller to maintain the average combustor pressure at a specified value.

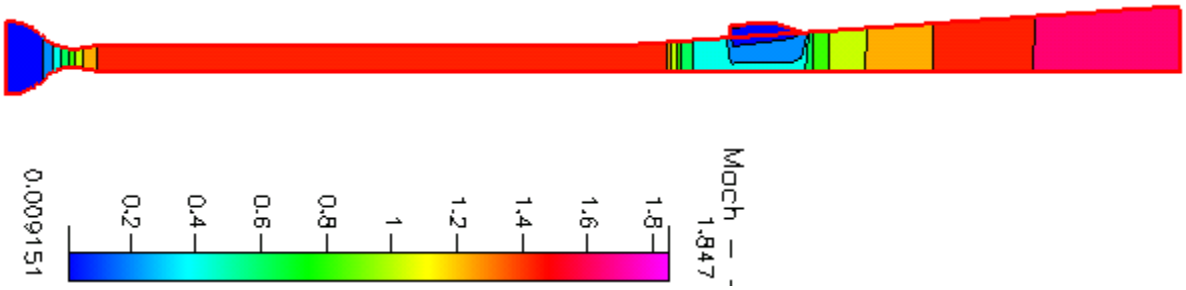
The above choice of a very simple performance function was made only to illustrate the controller operation and dynamics. Effect of a more complex performance function based on the

shock presence and position in the isolator was also investigated, and is discussed in the latter sections.

In order to maintain high computational turnaround time we performed all the simulations assuming 1-D geometry for all the propulsor components. We have performed initial calculations to ensure that our 1-D results were consistent with our previous 2-D runs. Figure 3.7 shows representative results.



a) Two dimensional grid simulation



b) One dimensional grid simulation

Figure 3.7. Comparison between 1D and 2D simulations: Mach number field.

The above Figure indicates that 1-D and 2-D runs are virtually identical. We therefore used 1-D geometry in all our subsequent controller simulations.

3.4 Controls Simulations

The average base combustor pressure under steady state operation with mass fuel flow =0.1kg/s, was on the order 120,000 Pa. The boundary conditions used were:

Inlet : P_{total} = 200,000 Pa; T=700C
Outlet : P = 10,000Pa

All the controls simulations started by suddenly increasing the fuel mass flow rate such that the isolator unstart was initiated. The controller was then required to keep the average combustor pressure to a specified magnitude. The ultimate goal was to control the fuel flow rate in order to prevent isolator unstart caused by sudden pressurization of a combustor in a scramjet engine. This may occur under conditions of sudden increase in the fuel flow rate, due to an increase in the required engine thrust.

We have run the simulations for two large mass flow rate perturbations, 0.15kg/s and 0.2kg/s, corresponding to 50% and 100% of the nominal flow. Two requested combustor pressures (separate simulations) were specified: 125000 Pa (close to steady state operation), and 100,000 Pa (below steady state). We report results for the case 0.15kg/s mass flow and 125000 Pa request pressure. Results for the other conditions are documented in Appendix 1.

The simulations were carried out using time a step of 10^{-5} seconds. Thus the overall simulation length of 10000 time steps corresponds to 0.1 seconds of physical time interval. For the most part, after the initial unstart, the controller attained its objective (request pressure) in a fraction of that time.

The results are shown in Figures 3.8 to 3.12. The Figures show the controller training error history, sensor data history, controller output history and Mach number fields at the beginning and the end of the simulations. The fields show the shock position and are presented to illustrate the controller's ability in preventing the motion of the shock out of the isolator, thus generating unstart conditions. For clarity, the images have normalized such that Mach numbers less than or equal to 1 are depicted in blue. In the figure captions, the term *delays* refers to time-delayed input/output taps, as given below:

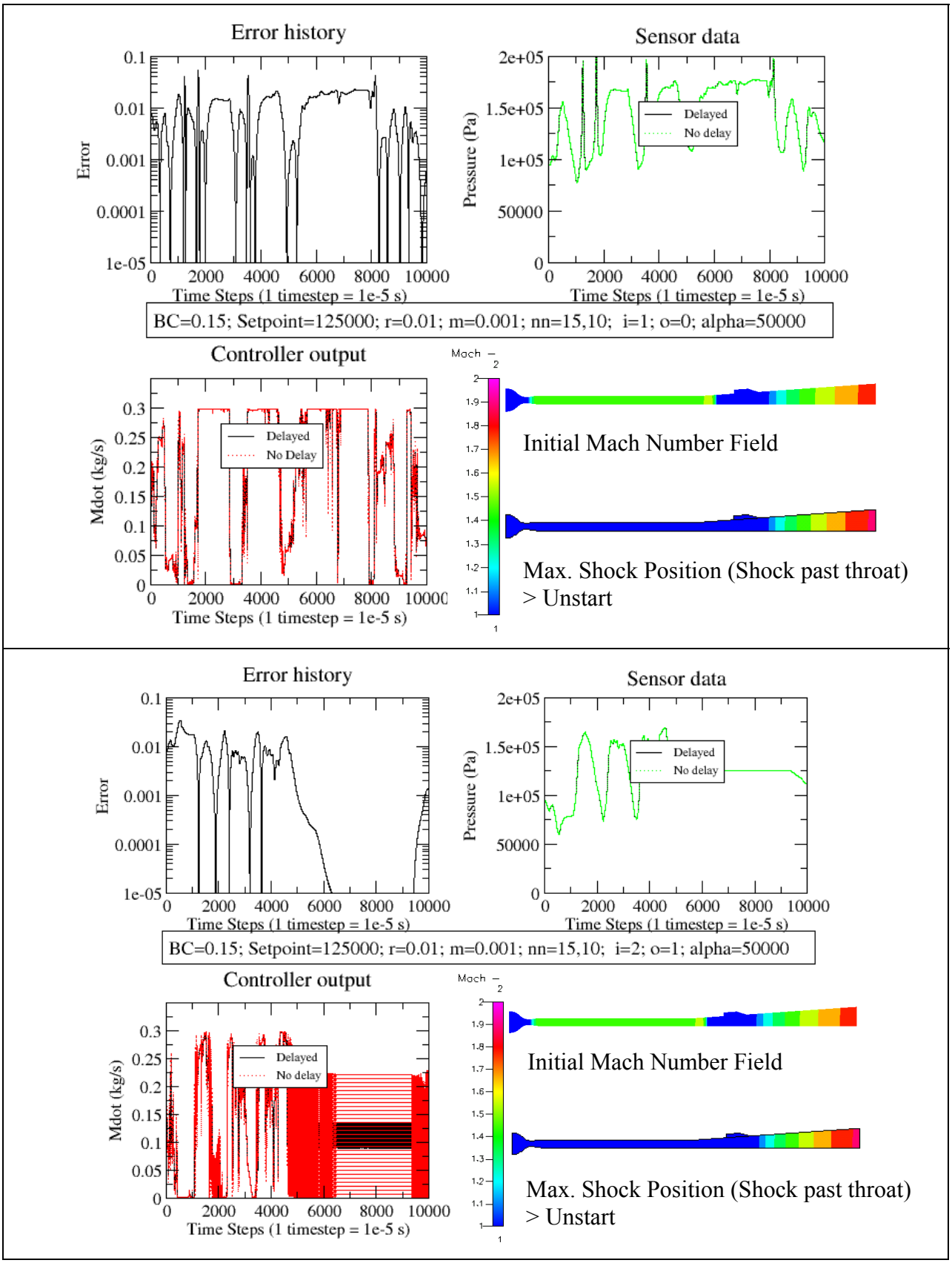
Delays = (#input delay taps - #output delay taps).

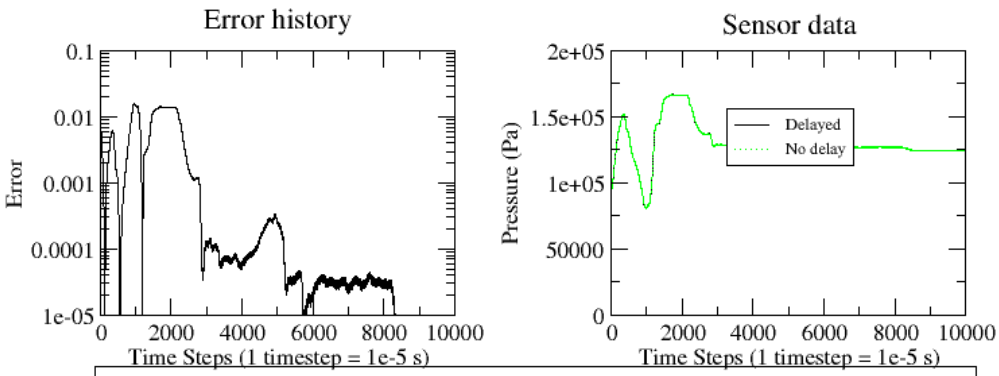
In other words, *Delays* refers to the time steps skipped before the sensor data is used by the controller (refer to the controller configuration Figure 3.6).

3.4.1 Instantaneous Sensor and Actuator Response

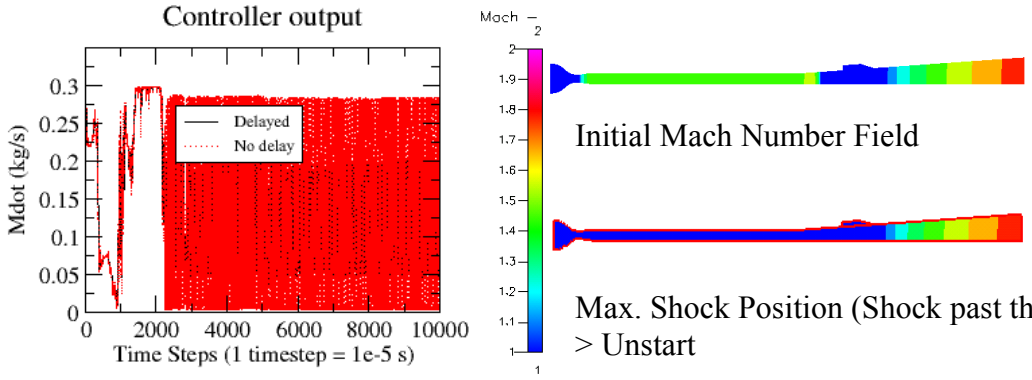
The first set of simulations (Figure 3.8) shows the controller performance as a function of the number of input/output delays. In this case the response parameter α was set to 50,000 Hz so that the sensor and actuator responses were essentially instantaneous.

The results show that the controller behaves very well. The performance however depends on the number of input/output signal delay taps. It is seen that for a small number of taps (i/o delay taps = 1/0, 2/1, 4/3) the controller cannot prevent unstart, although the request pressure of 125 kPa is reasonably well satisfied. The error curves show that the controllers in general train reasonably fast. We note that in these simulations, in all cases of unstart, the controller subsequently was able to restart the isolator (not shown). In general, a higher number of delay taps will better stabilize the isolator flow; however, because of the stochastic training methodology used for the controller, this trend may not be monotonic. Note that in our simulations we have perturbed the mass flow rate by 50% and the controller was able to handle such large perturbations very satisfactorily. Finally we note that all the simulations presented here were performed with an *unoptimized* controller.


Error history
Error
Time Steps (1 timestep = 1e-5 s)
Sensor data
Pressure (Pa)
Delayed
No delay
Time Steps (1 timestep = 1e-5 s)
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=1; o=0; alpha=50000
Controller output
Mdot (kg/s)
Delayed
No Delay
Time Steps (1 timestep = 1e-5 s)
Mach
Initial Mach Number Field
Max. Shock Position (Shock past throat) > Unstart
Error history
Error
Time Steps (1 timestep = 1e-5 s)
Sensor data
Pressure (Pa)
Delayed
No delay
Time Steps (1 timestep = 1e-5 s)
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=2; o=1; alpha=50000
Controller output
Mdot (kg/s)
Delayed
No delay
Time Steps (1 timestep = 1e-5 s)
Mach
Initial Mach Number Field
Max. Shock Position (Shock past throat) > Unstart

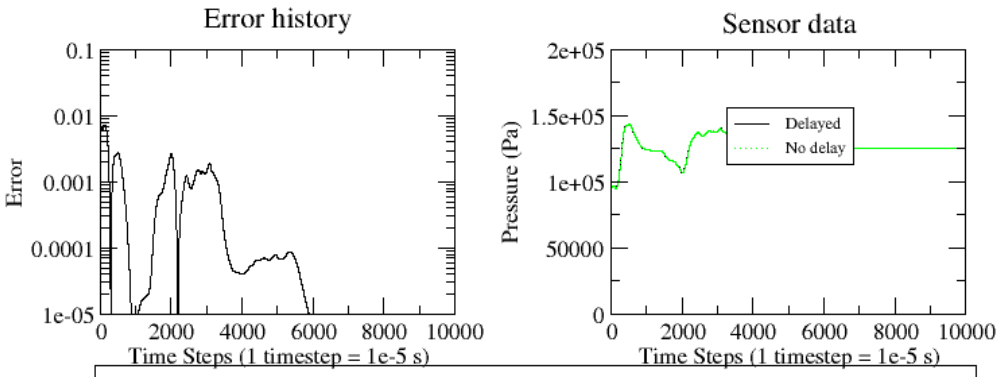


BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=4; o=3; alpha=50000

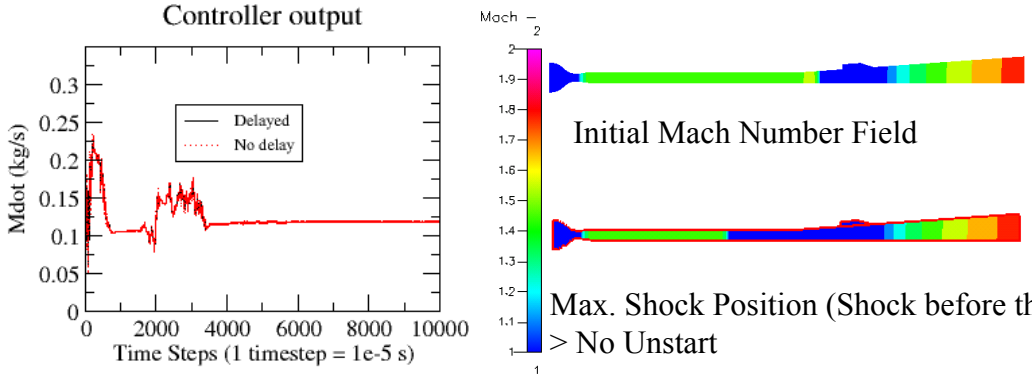


Initial Mach Number Field

Max. Shock Position (Shock past throat)
> Unstart



BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=8; o=7; alpha=50000



Initial Mach Number Field

Max. Shock Position (Shock before throat)
> No Unstart

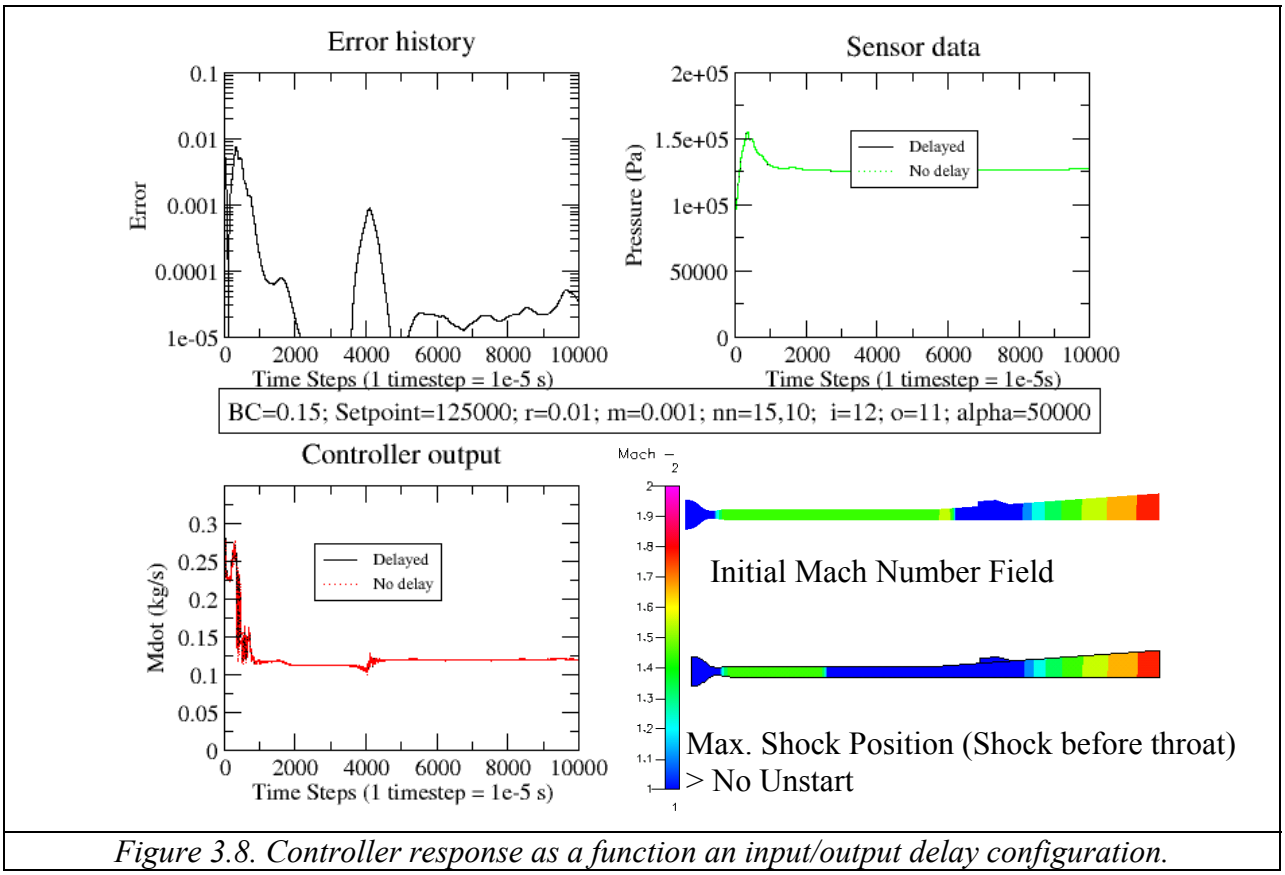


Figure 3.8. Controller response as a function an input/output delay configuration.

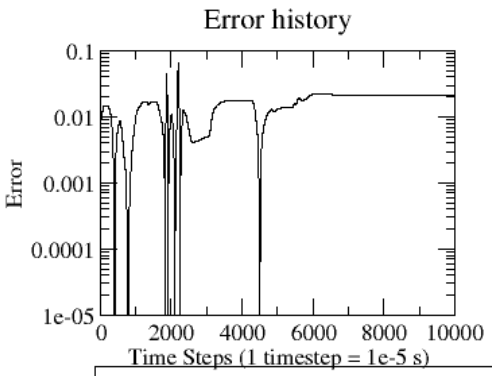
3.4.2 Finite Rate Sensor and Actuator Response

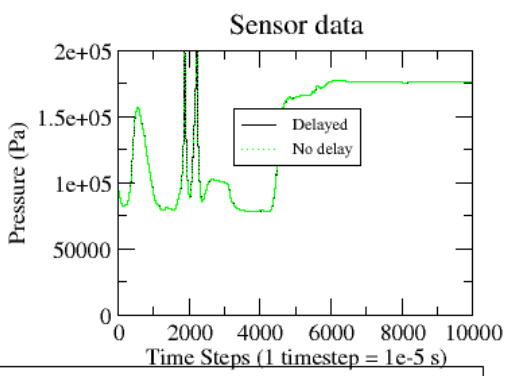
We have represented the sensor/actuator finite response time using a simple finite response mechanism that is based on the solution to an ODE of the form

$$dY/dt = \alpha(Y_{true} - Y),\ Y=\text{pressure, mass flow rate, } \alpha \text{ is the device time constant.}$$

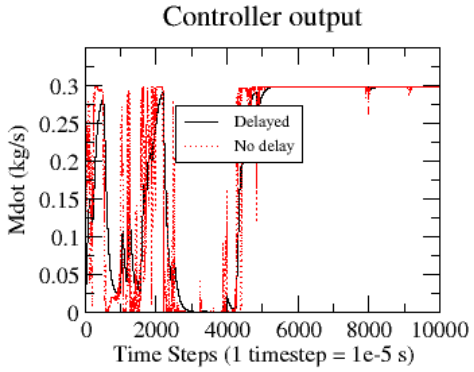
The second set of simulations shows the controller performance as a function of the number of input/output delays and the device response time as measured by α. For the following cases, α was set at10, 100, and 1000 Hz so that the sensor and actuator response times were finite.

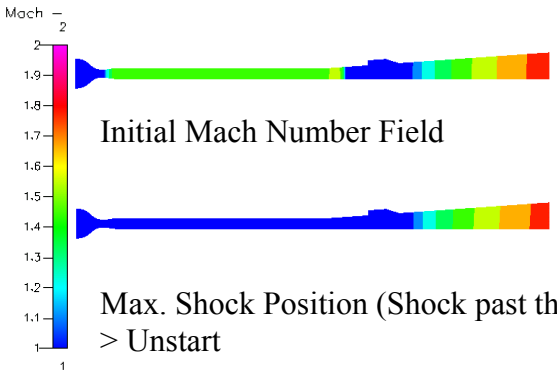
i) Results for α =1000 are shown in Figure 3.9. The actuator and sensor plots indicate little difference between instantaneous (no delay) and finite rate (delayed) responses, but the effect on the flowfield is noticeable. Although the controllers with i/o delay taps (2/1, 4/3, 8/7) were able to satisfy the request setpoint reasonably well towards the end of the simulation, only the controller with the (4/3) i/o delay tap configuration was able to prevent unstart. Except for this case, the error curves indicate little training convergence. The shockwave had traveled close to the nozzle throat; subsequently the controller pulled the shock back to the combustion chamber. These simulations indicate that even small inertia in the actuator and sensor response can have significant effects on controller performance. The simulations also show that performance function based on request pressure may be too simple for robust unstart control. This will be further discussed and illustrated in Section 3.4.3.





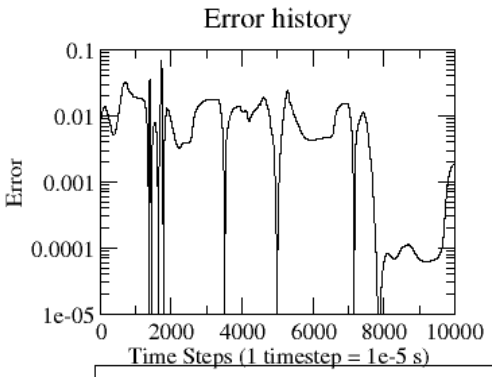
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=1; o=0; alpha=1000

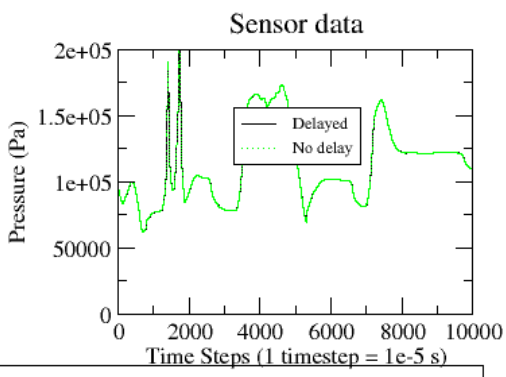




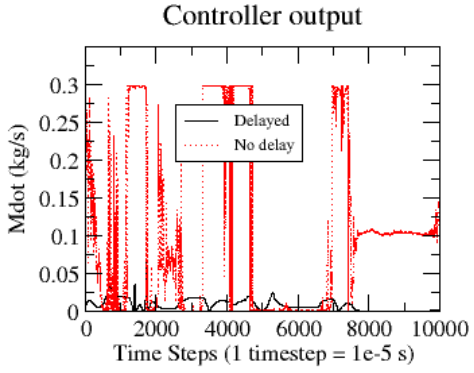
Initial Mach Number Field

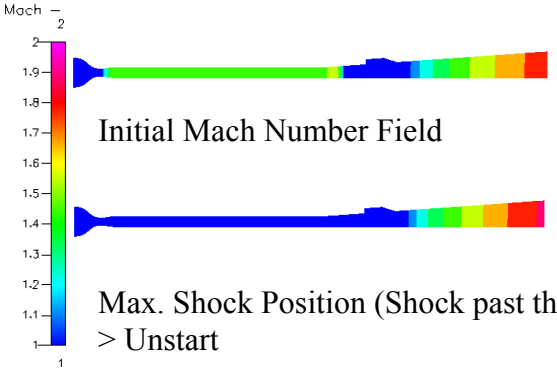
Max. Shock Position (Shock past throat) > Unstart





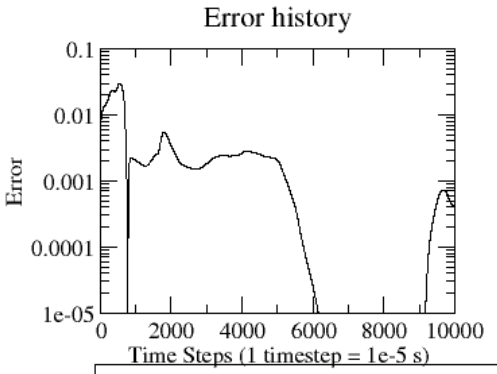
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=2; o=1; alpha=1000

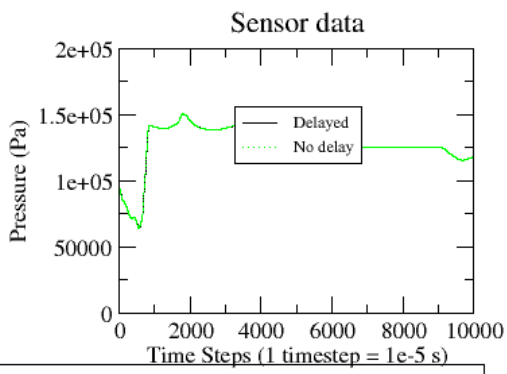




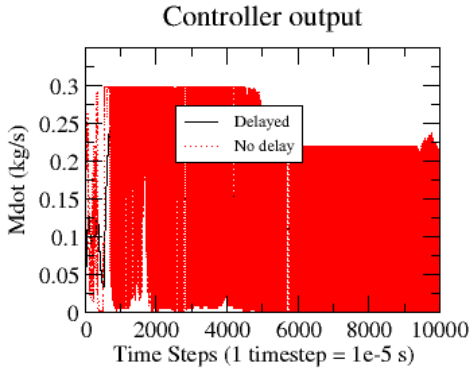
Initial Mach Number Field

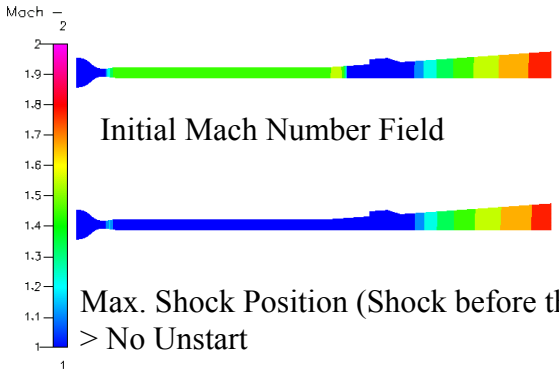
Max. Shock Position (Shock past throat) > Unstart





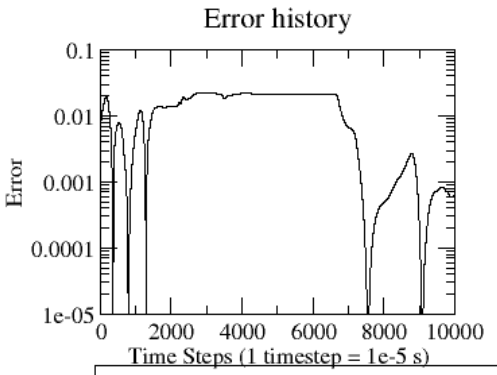
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=4; o=3; alpha=1000

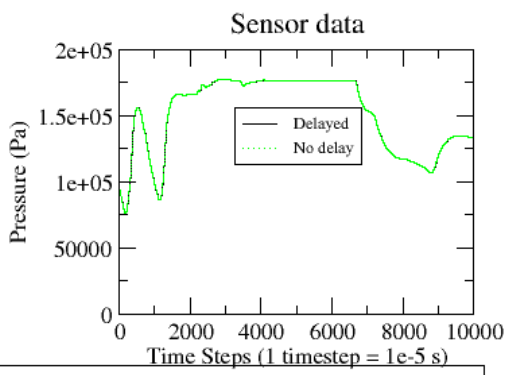




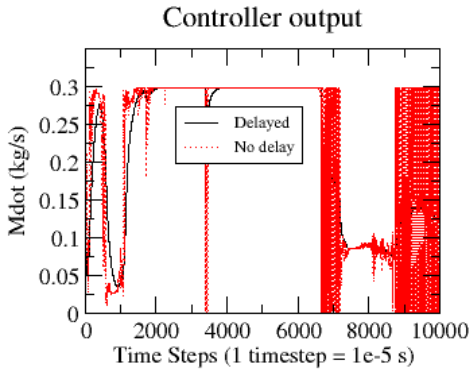
Initial Mach Number Field

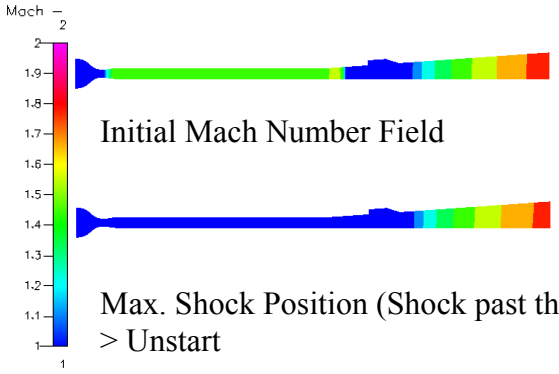
Max. Shock Position (Shock before throat) > No Unstart





BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=8; o=7; alpha=1000





Initial Mach Number Field

Max. Shock Position (Shock past throat) > Unstart

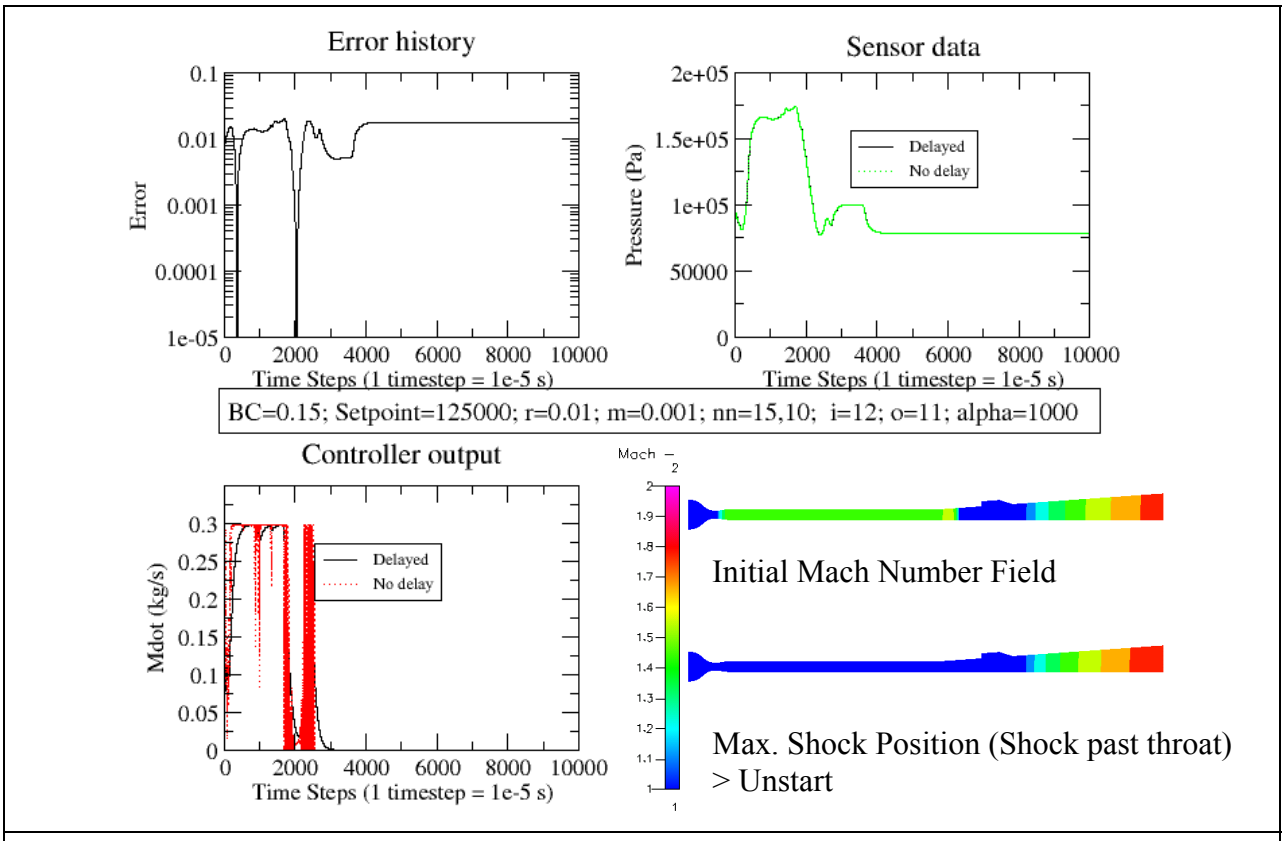
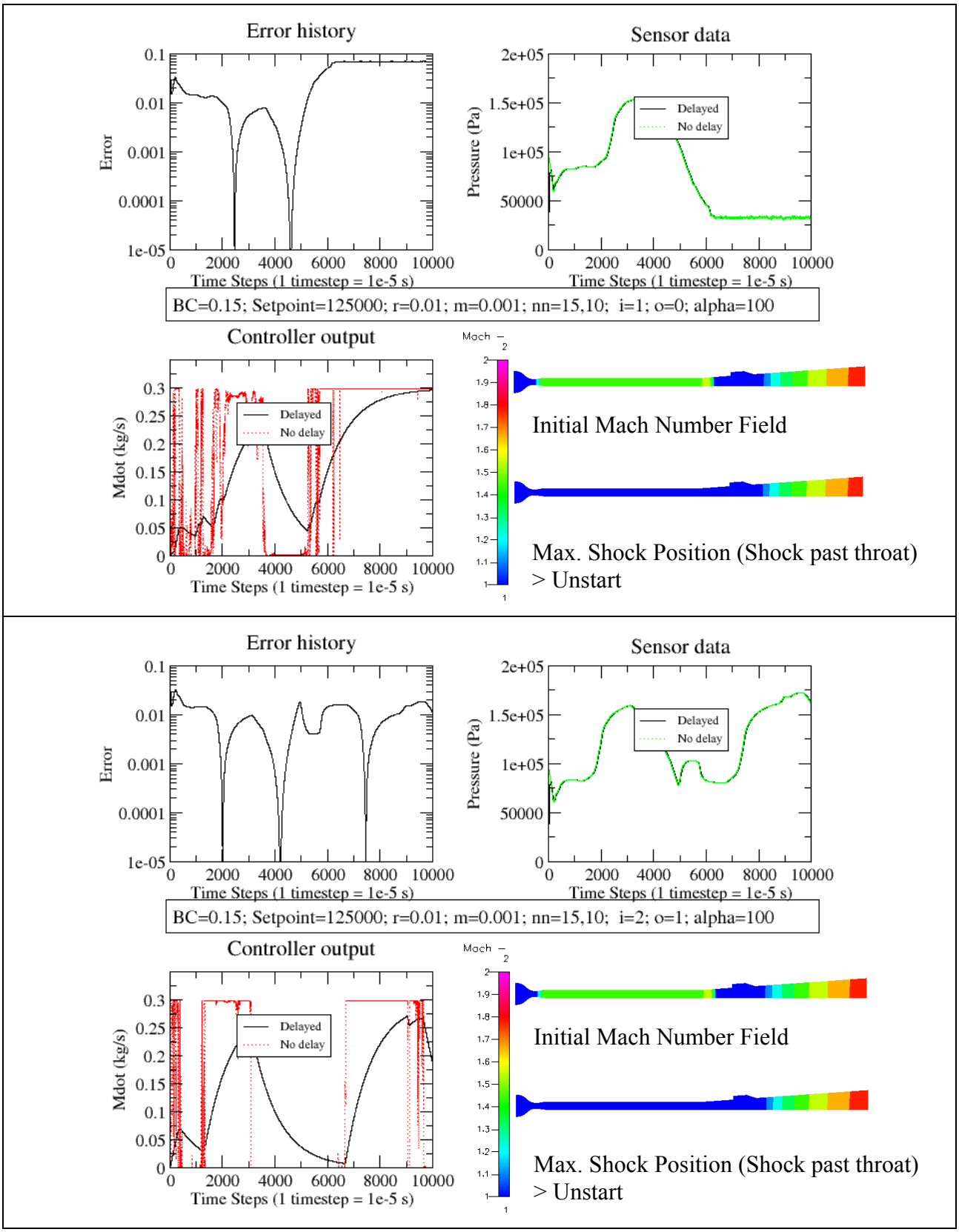
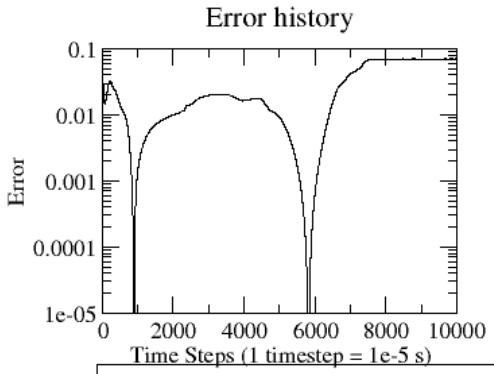
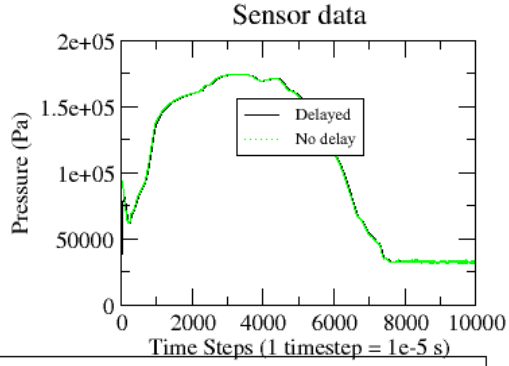


Figure 3.9. Controller response as a function an input/output delay configuration for α=1000.

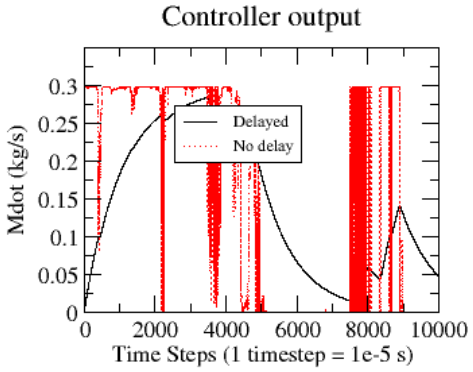
ii) Results for α =100 are shown in Figure 3.10. The actuator plots now indicate considerable difference between instantaneous (no delay) and finite rate (delayed) responses. The effect on the flowfield is also noticeable. There is still very little difference in the instantaneous and finite rate sensor response. This is because the variations in the pressure fluctuations are relatively slow compared to the sensor dynamics, allowing the latter to detect essentially the current pressure state. For this case, all the controllers failed to prevent unstart, although the ones with i/o delay taps (8/7, 12/11) were able to keep the combustor pressure in the vicinity the request setpoint reasonably well towards the end of the simulation. During the time allowed (0.1 s real time) the error curves do not show marked convergence indicating that the controller had not had the time to train properly. These simulations indicate that intermediate inertia in the actuator response can also have significant effects on controller performance. Again, the simulations also show that performance function based on request pressure may be too simple for robust unstart control. This will be further discussed and illustrated in Section 3.4.3.

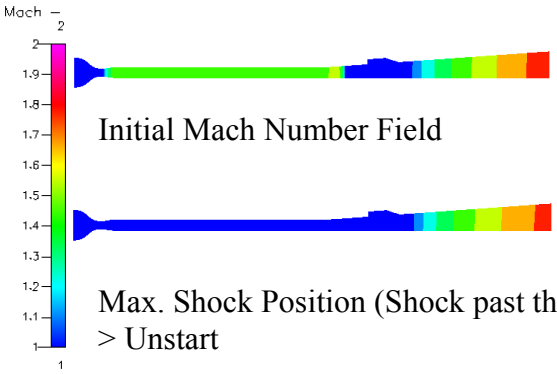

Error history
Error
Time Steps (1 timestep = 1e-5 s)
Sensor data
Pressure (Pa)
Delayed
No delay
Time Steps (1 timestep = 1e-5 s)
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=1; o=0; alpha=100
Controller output
Mdot (kg/s)
Delayed
No delay
Time Steps (1 timestep = 1e-5 s)
Mach
Initial Mach Number Field
Max. Shock Position (Shock past throat) > Unstart
Error history
Error
Time Steps (1 timestep = 1e-5 s)
Sensor data
Pressure (Pa)
Delayed
No delay
Time Steps (1 timestep = 1e-5 s)
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=2; o=1; alpha=100
Controller output
Mdot (kg/s)
Delayed
No delay
Time Steps (1 timestep = 1e-5 s)
Mach
Initial Mach Number Field
Max. Shock Position (Shock past throat) > Unstart





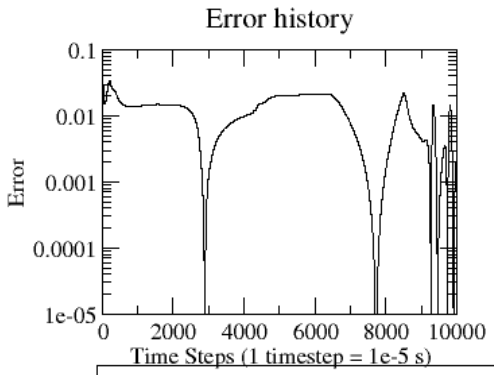
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=4; o=3; alpha=100

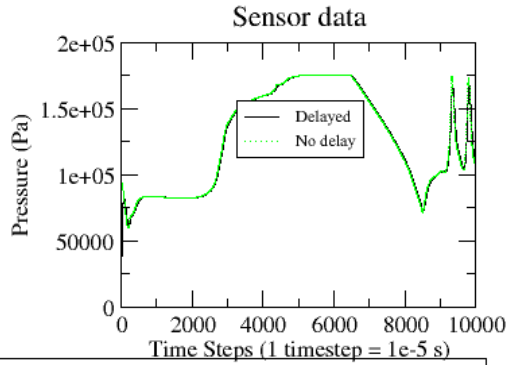




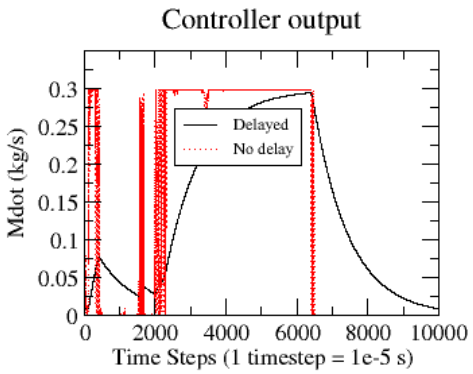
Initial Mach Number Field

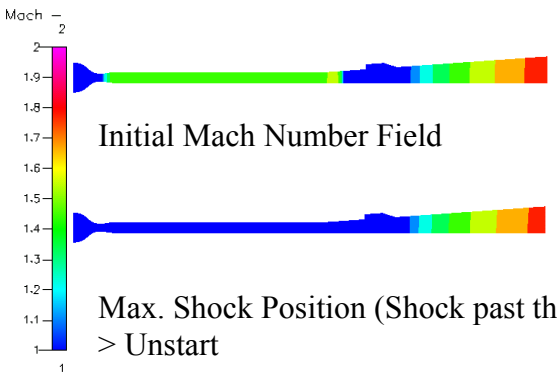
Max. Shock Position (Shock past throat) > Unstart





BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=8; o=7; alpha=100





Initial Mach Number Field

Max. Shock Position (Shock past throat) > Unstart

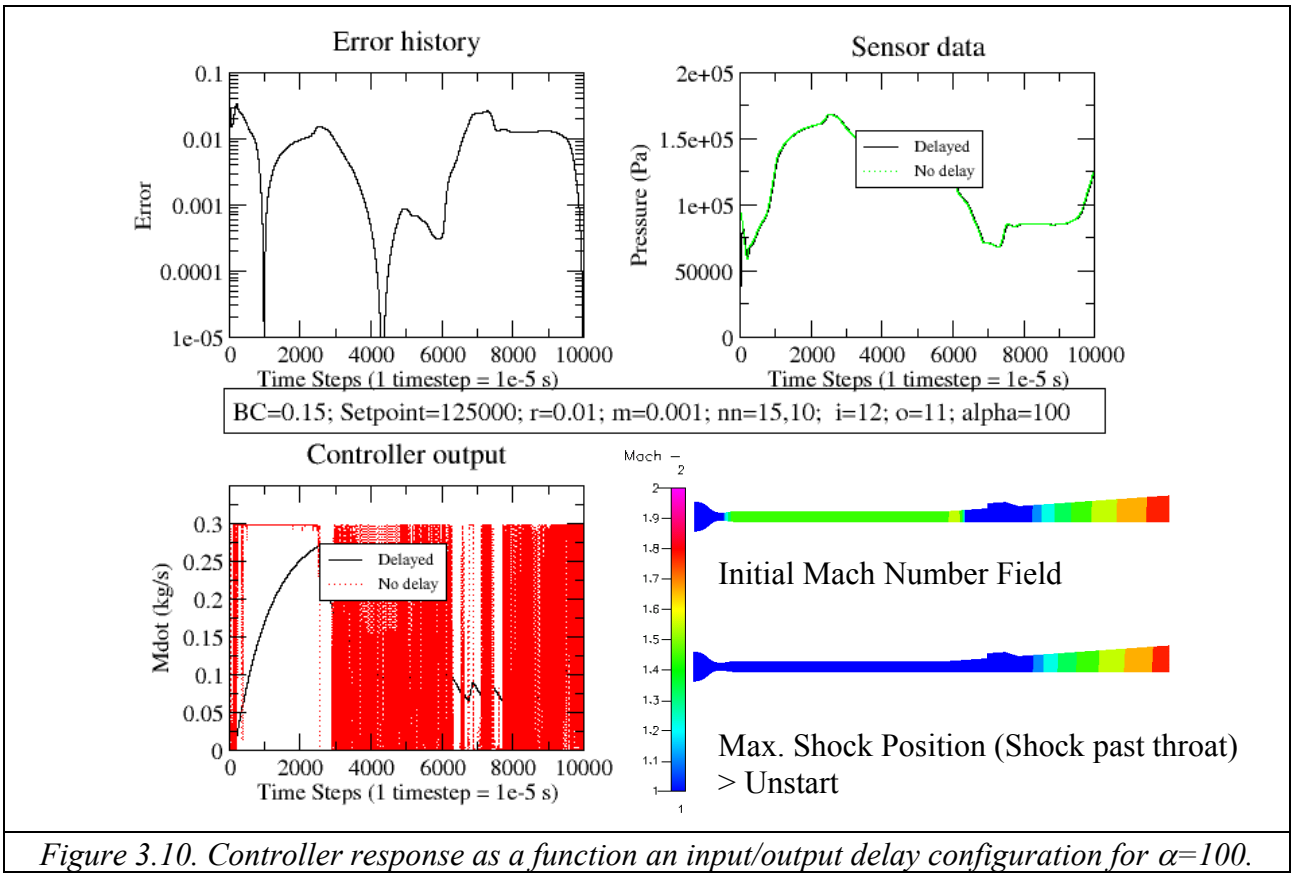
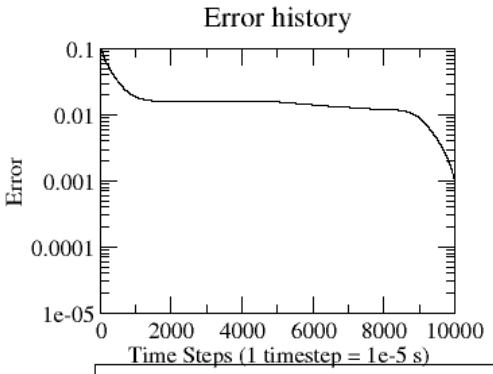
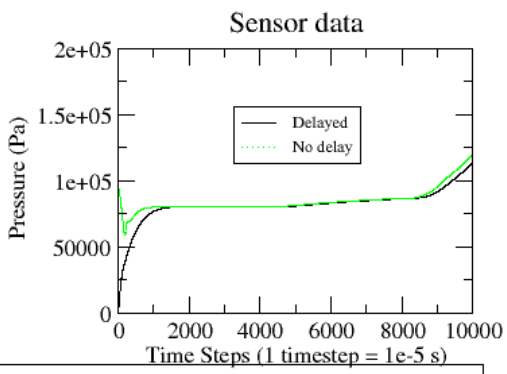


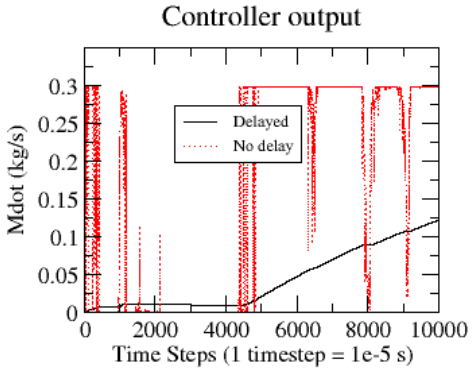
Figure 3.10. Controller response as a function an input/output delay configuration for α=100.

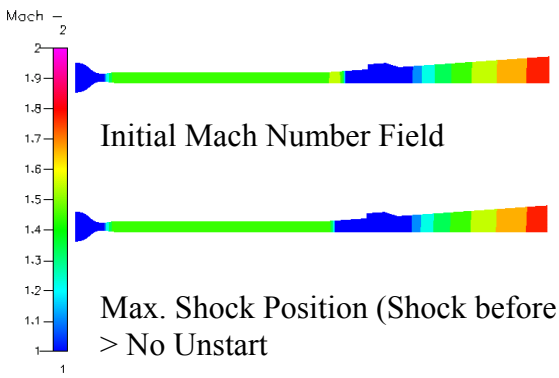
iii) Results for α =10 are shown in Figure 3.11. The actuator plots once again indicate considerable difference between instantaneous (no delay) and finite rate (delayed) responses. The maximum value of the controller output is not reached in the 0.1 sec associated with the simulation. Likewise, the combustor pressure is still increasing at the end of the run. However the pressure increase is smooth and we expect the controller to be able to satisfy the request pressure of 125kPa. The overall effect is that of a gradual increase in the fuel flow rate. In this case the flow dynamics are also changing slowly and the controller has much time to adjust itself to the new conditions. In all the cases the controller was able to prevent isolator unstart. It is seen that in all the cases the training error decreases for all controllers during the course of the simulations.





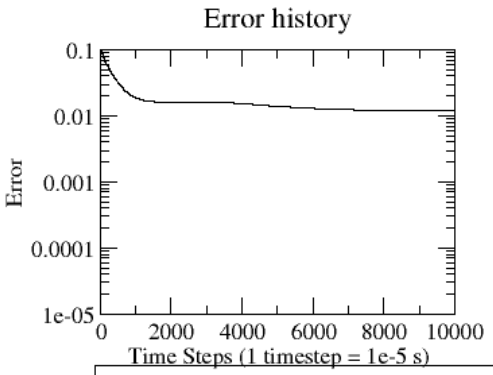
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=1; o=0; alpha=10

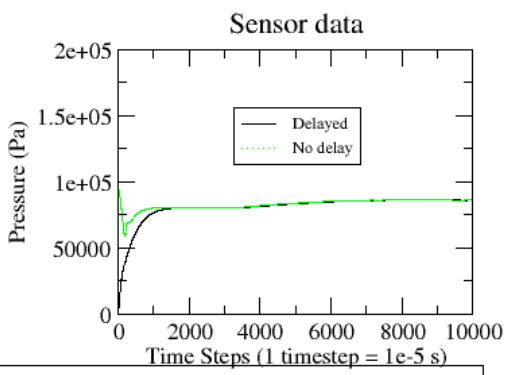




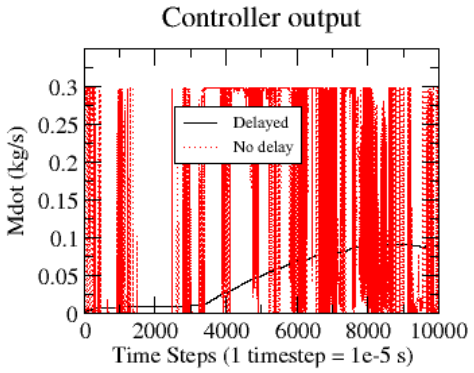
Initial Mach Number Field

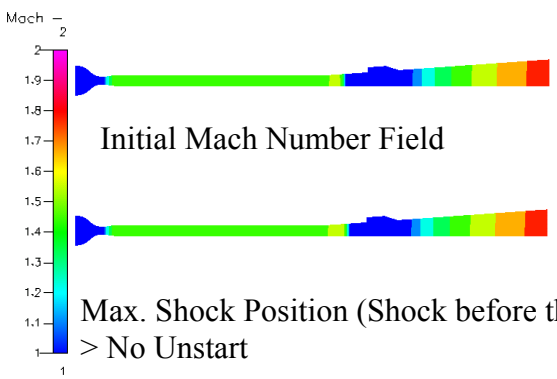
Max. Shock Position (Shock before throat) > No Unstart





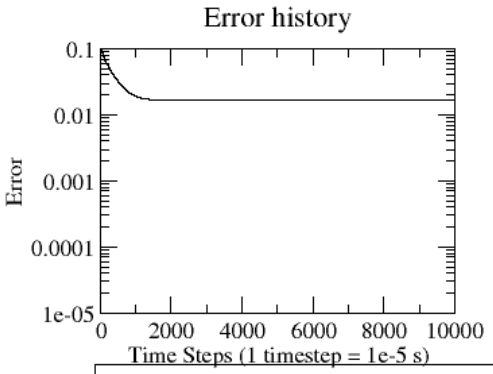
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=2; o=1; alpha=10

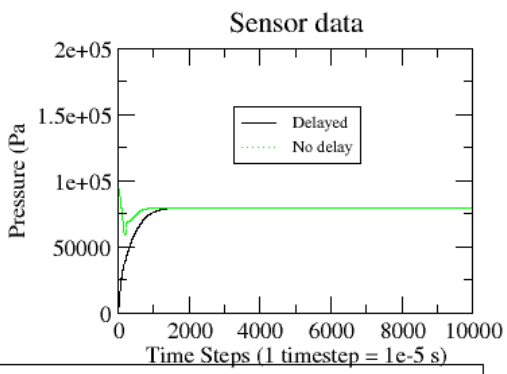




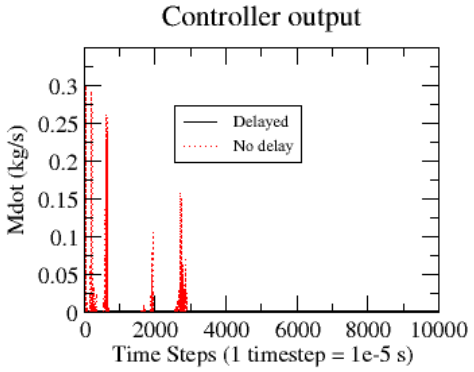
Initial Mach Number Field

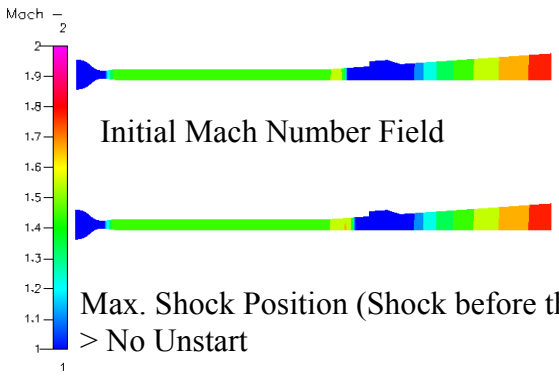
Max. Shock Position (Shock before throat) > No Unstart





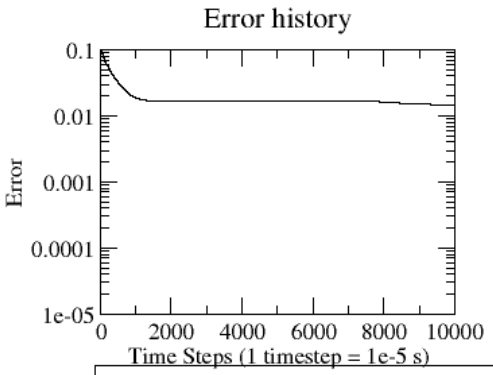
BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=4; o=3; alpha=10

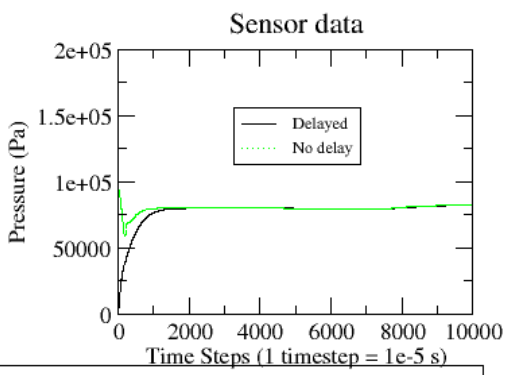




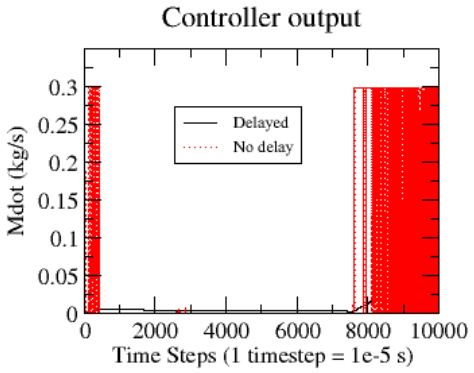
Initial Mach Number Field

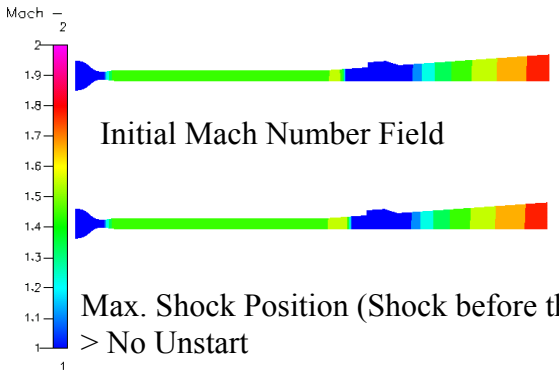
Max. Shock Position (Shock before throat) > No Unstart





BC=0.15; Setpoint=125000; r=0.01; m=0.001; nn=15,10; i=8; o=7; alpha=10





Initial Mach Number Field

Max. Shock Position (Shock before throat) > No Unstart

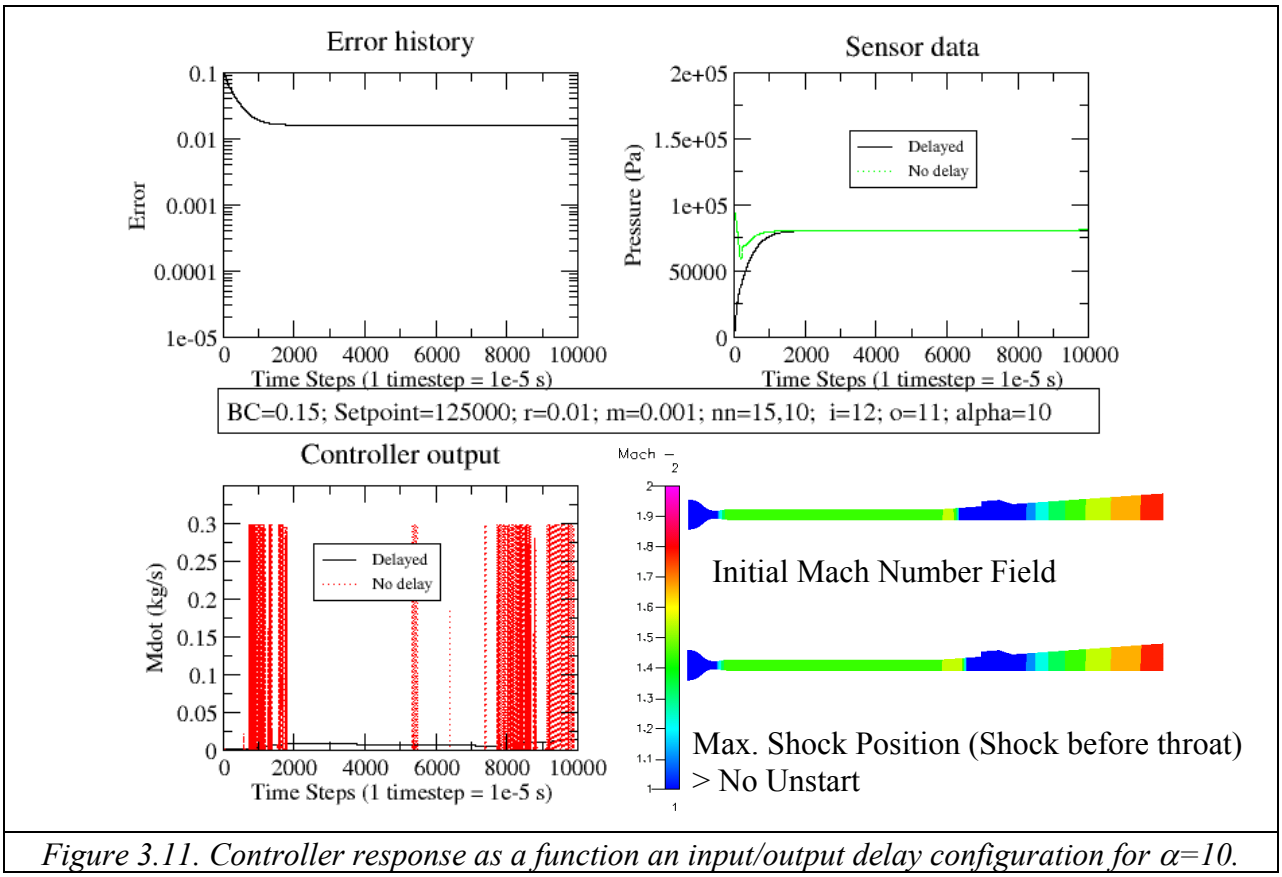


Figure 3.11. Controller response as a function an input/output delay configuration for α=10.

3.4.3 Finite Rate Sensor and Actuator Response with Improved Performance Function

In Section 3.4.2 we have noted that the effect of the response parameter α could have an adverse effect on controller training. We also noted that that the simple performance function requiring the controller to maintain a pressure of 125 kPa may have been too simple for the complex dynamics associated with finite actuator and sensor response time. In particular for α=100 Hz, the selected controller configuration was unable to prevent isolator unstart.

In this section we report results for a simulation performed using a performance function based on a maximum pressure gradient and the associated shock position. In this case the controller was required to control the fuel flow rate such that the maximum pressure gradient (shock position) in the flow was kept at a specified sensor location. The actuator and sensor response parameters were maintained at α=100 Hz. For this simulation we used a controller with 5/4 input/output delay taps. The setpoint was set such that the shock was to be kept in the vicinity of sensor 27, corresponding to its initial, unperturbed position. The results are shown in Figure 3.12. It is seen that in this case the controller performed well. Training errors decrease significantly, and the shock does not move out into the isolator. This simulation illustrates the importance of proper selection of the controller performance function.

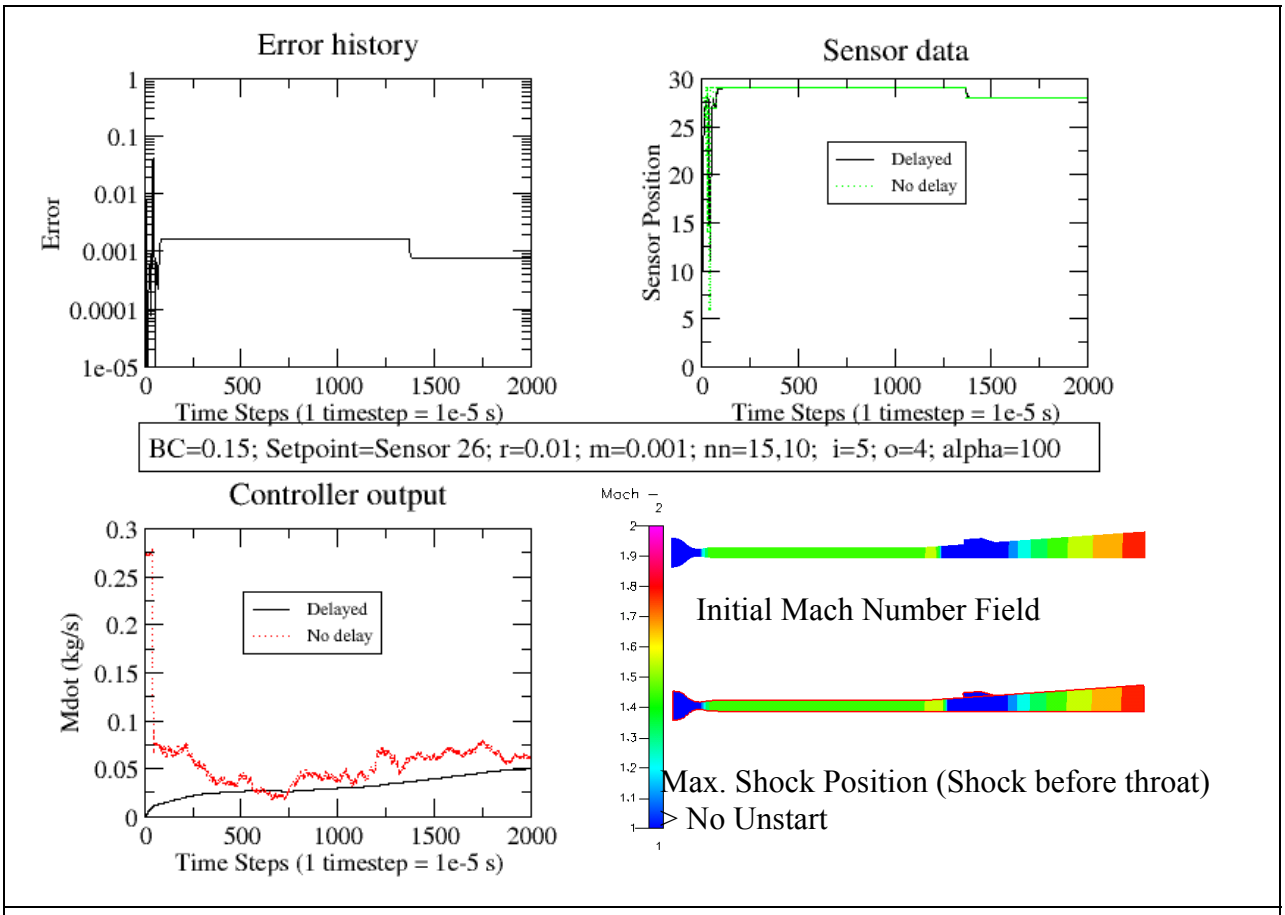


Figure 3.12. Controller response as a function an input/output delay configuration for α=50000 and setpoint request based on shock position.

3.4.4 Effect of Controller Pre-Training

Using the same configuration as previously with α=50000 Hz, we pretrained the controller for the conditions generated by a fuel mass flow rate of 0.15 kg/s. We subsequently used the trained controller to control the combustor subject to 0.20 kg/s in fuel flow perturbation. Comparison of trained and untrained controller response is given in Figure 3.13 below. It is seen that pre-training results in smoother controller operation, with the request pressure of 125,000 Pa being reached much faster. Note also that the pretrained controller was able to completely control the unstart. Conversely, the untrained controller allowed unstart to take place, subsequently recovering the desired supersonic state in the isolator.

The beneficial effects of pre-training can have important consequences in controller design and operation since it can: 1) accelerate the overall controller training procedure; and 2) operate reliably in flow regimes for which it has not been previously trained.

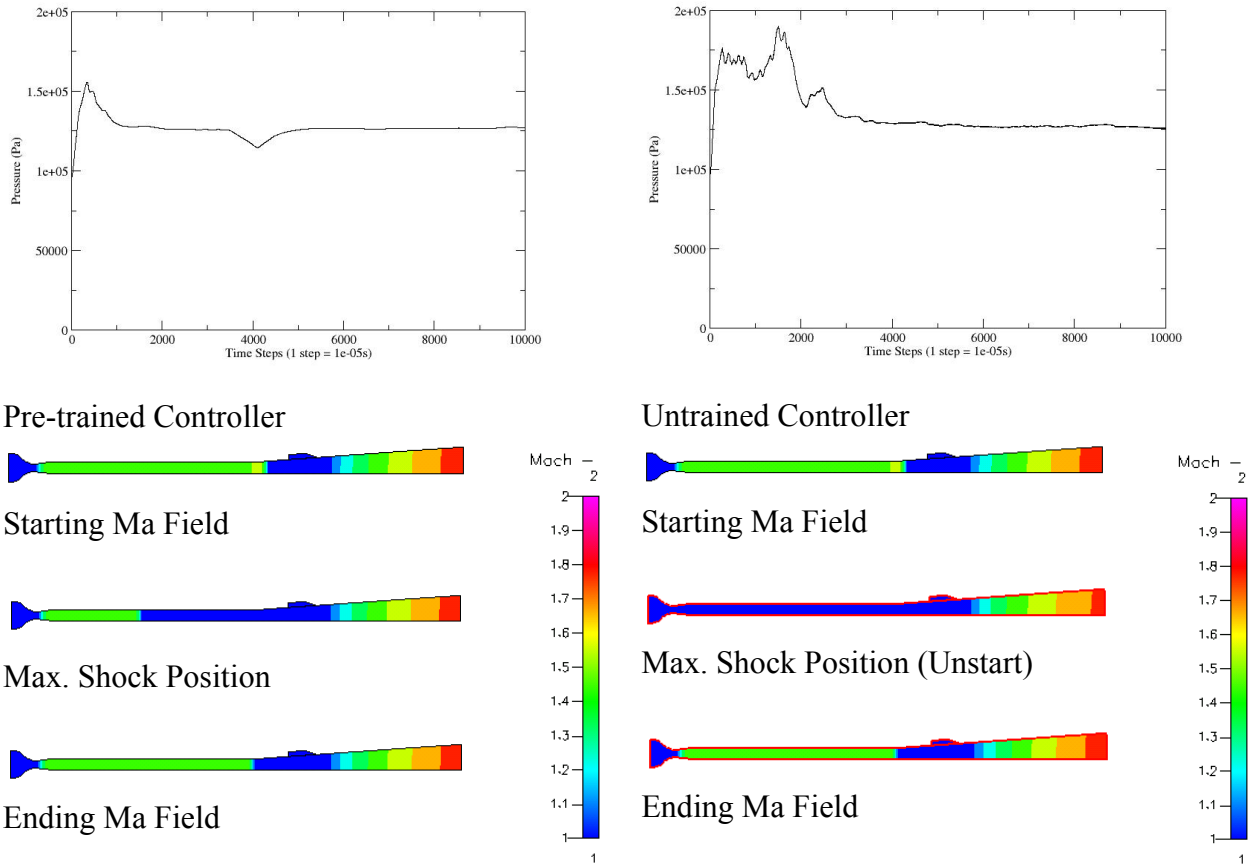


Figure 3.13. Effect of controller pretraining on controller response.

3.5 Summary of Simulations

We have presented simulation results for control of isolator unstart in a hypersonic combustor system. The system was modeled in 1D and verified against selected 2D steady state and transient results. Flow fields were simulated using CFD-ACE+, a SIMPLEC-type, pressure-based code. Selected code results for low Mach number (Ma~3) flows compared favorably against those generated by CFD-FASTRAN, a density based code typically used for compressible flow applications. Several types of controller simulations were performed. The initial set assumed instantanous sensor and actuator response, and was conducted to investigate the effect of controller architecture in controlling isolator unstart. The results showed that rather simple controller configurations were capable of peforming the task without much difficulty. Second set of simulations illustrated the controller performance where sensors and actuators exhibited finite time response. The controller was still able to prevent unstart, although more testing is required in order to determine the optimal controller configuration. There tests have also shown that for more complex flows, the controller requires more complex performance function definitions in order to peform satisfactorily. Finally, we ran a simulation where the controller was trained on one set of flow conditions, and required to control the unstart under

another set. The simulations have indicated that a pre-trained controller responds faster than one that learns on-line, even when the training spanned a different flow regime.

This completes the summary of Phase I work. The results demonstrate the feasibility of constructing a software environment for virtual controls prototyping of complex flow systems using a direct control strategy, and implemented in a simple recurrent ANN that trains on-line during the course of operation. Important findings from this portion of the efforts include:

1. Development of the APIs for different legacy codes must take into account the formulation of the legacy code, and the manner in which the relevant flow equations are solved (characteristics-based, pressure based etc.).

2. Controller structure in terms of input and output delays taps determines to a large degree the controller's ability to handle complex, time dependent flows. Typically, for the conditions tested, increasing the number of delays improves the controller response.

3. The effect of finite response time of sensors and actuators is an important consideration in controller design. There appears to be an optimal number of delays for which the controller has the best response, if finite actuator/sensor response is taken into account.

Pretraining the controller can significantly improve its performance, even if the training was performed in a flow regime different from the one to which the controller is exposed. This is an important characteristic of Artificial Neural Networks that will allow a hardware prototype to be pre-trained using simulations before being mounted in an experimental rig..

4.0 CONCLUSIONS AND RECOMMENDATIONS

4.1 Conclusions

During Phase I research we have successfully demonstrated the feasibility of controlling isolator unstart in a hypersonic combustor using a self-training, model-free, direct control strategy of the fuel flow rate. We have generalized an existing simulation environment to couple concurrent execution of separate user-supplied codes (CFD and other) each modeling the dynamics of individual components comprising the whole system. We have developed API prototypes that can be integrated into external codes for used in code-to-code data transfer. We have also developed a user-extendable GUI prototype for on screen system composition and component connectivity. We have met all the major objectives of Phase I research successfully.

All the simulations were performed using an all-speed pressure-based code, verified against a high fidelity density-based code. The combustor geometry and approximate operating conditions were obtained from WPAFB. We have performed sensitivity analysis with respect to the controller structure, the effects of sensor and actuator dynamics, training strategy, and choice of performance function that the controller minimizes. We have performed the following types of simulations:

1. Preliminary simulations were performed on a 2D grid in order to ascertain the operating conditions that would result in a reasonable flow conditions associated with hypersonic combustors. Parametric tests on the fuel flow rate were performed to obtain the maximum allowable value that would not lead to isolator unstart. We have also conducted numerical experiments to determine the optimal fuel injection strategy. We repeated the calculations for a 1D grid and verified the results by comparison to 2D simulations. Steady state solutions of this former configuration served as a baseline from which all the controls simulations were performed. These simulations showed that 1D combustor geometry can be used to demonstrate controller training at a relatively modest computational cost.

2. Initial unstart conditions were created by suddenly increasing the fuel mass flow rate to create a pressure excursion in the combustion chamber. This may occur under conditions of sudden increase in the fuel flow rate, due to an increase in the required engine thrust. For most of the calculations the controller was then required to manipulate the flow rate in order to keep the average combustor pressure to a specified magnitude.

3. The first set of control simulations was performed assuming instantaneous actuator and sensor response. These simulations showed that controller performance depends its configuration in term of the number of feedback delay taps that it uses. The controller was able to keep the combustor at a request pressure and was able to prevent unstart.

4. The second set of simulations was performed assuming finite time actuator and sensor response. The simulations showed that actuator response is the determining factor in the controller performance. They also showed that the performance depends on the actuator characteristics. Fast actuator (> 1000 Hz) had little effect on the controller performance.

Slower actuator (~100 Hz) affected controller performance significantly and indicated that control criteria based on a specified pressure of the combustion chamber may be inadequate. For this type of an actuator the controller was not able to prevent unstart. For a very slow actuator (~10 Hz) the overall effect was that of a gradual increase in the fuel flow rate. In this case the flow dynamics are also changing slowly, the controller has much time to adjust itself to the new conditions, and had no difficulty in controlling the unstart.

5. The third set of simulations was performed assuming finite time actuator and sensor response (100 Hz), but this time requiring the controller to keep the generated shock wave at a specified position. Using this new criterion, the controller had no difficulty in preventing isolator unstart. The simulation showed that controller performance is sensitive to the control criteria (performance function), especially when complicating dynamic of actuators are taken into account.

6. The fourth set of simulations assumed instantaneous actuator and sensor response, and was conducted to investigate the effect of controller pre-training. We have found that pre-training significantly improves the performance, even if it was done on a set of conditions different from those to which it was applied.

The software development work and the associated software demonstrations show the feasibility of the overall concept of using direct control of fuel mass flow rate to control isolator unstart. In the next section we outline our recommendations for additional work required to satisfy the overall project goals.

4.2 Recommendations For Phase II Research

Phase II work will focus on generalization of the methodologies developed in Phase I, with emphasis on further controller improvements, interfacing with additional legacy codes, GUI finalization, inclusion of system-level sensor/actuator dynamic effects, and hardware implementation and testing. An important part of this research will relate to practical demonstration of the software utility for rapid controls prototyping, and hardware design and testing. Focus again will be on isolator unstart control in a hypersonic combustor. The hardware controller prototype will be tested in an experimental rig, preferably at Wright Patterson facilities. The following issues will be explicitly addressed.

Objective 1. Graphical Prototyping Environment will be extended to enable screen-oriented, specification of sensor, actuator, and controller module designs and connectivities. Final GUI structure will be designed in consultation with the Air Force. The GUI being developed will be a stand-alone construct, and will be largely user-configurable. GUI development to date was being done in conjunction with another Air Force project relating to modeling of systems composed of subsystems being simulated by legacy codes running in parallel. The GUI will be useful for configuring a wide variety of systems composed of well-defined sub-systems. The software will therefore be useful not only for propulsion system design but in other areas such as design of bio-chips.

Objective 2. ANN-Chemistry module will be added to the environment to accelerate the chemistry kinetics calculations. This will be a stand-alone computational module that will be usable in place of very expensive ODE solvers. Its development is desirable since chemistry calculations account for significant portion of the computational burden in reacting flow simulations. This module will contain a library of reaction mechanisms for a number of known fuel-oxidizer combinations, within appropriate operational condition envelopes. It will be implementable into other legacy codes with minimum code modification. The ANN modules will be trained using a class of highly efficient training algorithms available in the existing computational environment.

Objective 3. Sensor and Actuator Device Library will be developed to include the dynamic models of a comprehensive list of sensors and actuators commonly used in propulsion systems. The models will include simple 0D and 1D descriptions as well as ANN forms. Device types to be developed will be selected in consultation with Air Force engineers. Reduced models of devices are of great interest in computational prototyping of combustion systems, since system-level calculations with fully resolved device models could be very expensive. Using ANNs in a general recurrent network topology one can represent a very wide class of complex dynamical systems. One example of this capability stems from our Phase I results where a recurrent ANN was able to handle very complex compressible flow calculations.

Objective 4. General API Design will be completed for coupling of legacy codes and its capability will be demonstrated on selected codes of interest the Air Force. We anticipate focusing our work on CFD-ACE+, CFD++, and Vulcan. The API design will include development of software functions to condition transferred data into form compatible with a particular solution procedure used by legacy codes.

Objective 5. Direct Control Strategy will be refined based on lessons learned during Phase I. Methods for automatic configuring (e.g. LeCun, 1990; Hassibi, 1992; Cowan, 1994; Liu, 1999) of ANN topology in terms of structure, delays specification and acceleration of learning algorithm will be investigated. The primary advantage of direct control strategy is that it does not require an *auxiliary* plant model. A comprehensive controls environment with access to this control technique can be used in any application involving control of complex nonlinear dynamical systems ranging from combustion and flow control, to control of drug delivery based on patient response.

Objective 6. Controller/Flow Solver Interface will be generalized to enable coupling to legacy codes used by Air Force and by OEMs and demonstrated on codes recommended by Air Force engineers. The tendency in the current simulation climate is to perform multidisciplinary calculations, and such interfaces will generalize the applicability of this software to performing of simulations involving the interaction of several legacy codes.

Objective 7. ANN Based Sensor Validation methodology will be implemented to compensate for sensor malfunction or failure during operation. Auto Associative Neural Nets (AANNs) have successfully been used for such tasks, in the context of engine health monitoring and will be used in the present project. Proper sensor operation is essential to successful control.

Objective 7. Device Prototyping and Mission Profile Demonstration Simulations will be performed in consultation with Air Force engineers, in order to validate and demonstrate the performance of the developed control strategies. Comparisons to available experimental data and other controllers will be made where possible for validation. The simulations will demonstrate the role of virtual prototyping of flow control strategies in the setup of guidelines for hardware development and experimental and field-testing.

Objective 8. Hardware Prototype Development and Testing will be performed on the basis of results obtained under Objectives 6 and 7. Hardware-based controllers will be an essential step in the utilization of the controllers in actual engines. Successful development and testing of the hardware prototype will serve as a final proof of the design's viability. Hardware-based controllers in the form of e.g. PCI cards are much more desirable than software-based algorithms for control of real engine hardware, and thus have high utilization potential in engines developed for Air Force, NASA and in commercial engines.

4.3 Technology Challenges

The major technology challenges expected in Phase II are similar as those encountered in Phase I, namely:

1) Realistic incorporation the interaction of reduced and full-scale models. (Alternatives include simulation of component motion, direct manipulation of field variables, or suitable representation of source and sinks).

2) Insuring that the ANN architecture and configurations learn sufficiently well the flowfield dynamics at the points of sensing and actuation.

3) Implementation of self-configuring ANNs and accelerated training algorithms.

4) Specification of a reliable performance function that would facilitate unstart control in minimal time.

5) Selection of time constants associated with input data sampling and controller activation so that the system is not over-controlled.

6) Devising and accelerated and progressively complex controller training scheme based on 1D, 2D, and 3D simulations.

7) Pre-training of PCI-base hardware controller using software-simulations.

5. REFERENCES

Blasco, J.A., et al. (1998), "Modeling the Temporal Evolution of a Reduced Combustion Chemical System with an Artificial Neural Network" *Combustion and Flame,* vol. 113, pp 38-52, 1998.

CFDRC (2002), CFD-ACE+ Users' Manuals, CFD Research Corporation, January 2002.

Chen, C.H. ed. (1996), *Fuzzy Logic and Neural Network Handbook*, New York, McGraw-Hill, 1996.

Chin, D.C. (1994). "A More Efficient Global Optimization Algorithm Based on Styblinski and Tang." *Neural Networks*, Vol. 7, No. 3, pp. 573-574.

Chin, D.C. (1997). "Comparative Study of Stochastic Algorithms for System Optimization Based on Gradient Approximations." *IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics*, Vol. 27, No. 2, pp. 244-249.

Christo, F.C. et al. (1996), "An Integrated PDF/Neural Network Approach for Simulating Turbulent Reacting Systems," 26th Int. Symposium on Combustion/ The Combustion Institute, pp.43-48, 1996.

Cowan, J.D., Tesauro, G. and Alspector, J.R., (1994). "Fast Pruning Using Principal Component Analysis," *Advances in Neural Information Processing*, Vol. 6.

Giles, C.L., (1996). "Dynamically Driven Recurrent Neural Networks: Models, Learning Algorithms, and Applications," Tutorial #4, *International Conference on Neural Networks*, Washington DC.

Hassibi, B., Stork, D.G. and Wolff, G.J., (1992). "Optimal Brain Surgeon and General Network Pruning," *IEEE International Conference on Neural Networks*, vol. 1, pp. 293-299, San Francisco.

Haykin, S., *Neural Networks: A Comprehensive Foundation*, Macmillan (1999), New York, 1999.

Hestenes, M. (1980), Conjugate Direction Methods in Optimization. New York: Springer-Verlag.

LeCun, Y., Denker, J.S. and Solla, S.A., (1990). "Optimal Brain Damage," *Advanced in Neural Information Processing Systems*, vol. 2, pp. 598-605, San Mateo, CA: Morgan Kauffman

Lee, C., King, J., Babcock, D. and Goodman, G., "Application of Neural Networks to Turbulence Control for Drag Reduction," Phys. Fluids vol. 9, no. 6, 1997.

Lin, T., Horne, B.G., Tino, P., and Giles, C.L., (1996). "Learning Long-Term Dependencies in NARX Recurrent Neural Networks," *IEEE Transactions on Neural Networks*, vol, 7, pp. 1329-1338.

Liu, P., Kadirkamanathan, V., and Billings, S.A., (1999). "Variable Neural Networks for Adaptive Control of Nonlinear Systems", *IEEE transactions on Systems, Man , and Cybernetics-Part C: Applications and Reviews*, Vol. 29, No. 1, pp. 34-43, 1999.

Maeda, Y. and De Figueiredo, R.J.P. (1997). "Learning Rules for Neuro-Controller via Simultaneous Perturbation." *IEEE Transactions on Neural Networks*, Vol. 8, No. 5, pp. 1119-1130.

Narendra, K.S., (1995). "Neural Networks for Identification and Control," *NIPS 95* Tutorial Program, pp. 1-46, Denver, CO.

Okuda, H. and Yagawa, G. (1997), "Multi-Color Neural Network with Feedback Mechanism for Parallel Finite Element Fluid Analysis" Chap. 12, Parallel Solution Methods in Computational Mechanics, ed. by M. Papadrakakis, Wiley & Sons, 1997.

Pindera, M.Z., (1998) "Intelligent Solution Optimization Using Artificial Neural Networks Applied to Numerical Flow Solvers", SBIR Phase I Final Report, CFDRC Report:4932/1

Pindera, M.Z. (2001), "Intelligent Solution Optimization Using Artificial Neural Networks Applied To Numerical Flow Solvers"; Phase II Final Report; NASA AMES Contract Number: NAS2-99050

Pindera, M.Z. (2002), "Adaptive Flow Control Strategies Using Simple Artificial Neural Networks"; AIAA-2002-0990.

Rezayat, F. (1995). "On the Use of an SPSA-based Model-free Controller in Quality Improvement." *Automatica*, Vol. 31, No. 6, pp. 913-915.

Rietman, E.A., (1996)"A Neural Network Model of a Contract Plasma Etch Process for VLSI Production," IEEE Trans. of Semiconductor Manufacturing, Vol. 9, No. 1, 1996.

Rumelhart, D.E. and McClelland, J.L., eds. (1986), Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1, Cambridge, MA: MIT Press.

Sadegh, P. (1997). "Constrained Optimization via Stochastic Approximation with a Simultaneous Perturbation Gradient Approximation." *Automatica*, Vol. 33, No. 5, pp. 889-892.

Siegelmann, H.T., Borne, B.G., and Giles, C.L., (1997). "Computational Capabilities of Recurrent NARX Neural Networks," *Systems, Man and Cybernetics*, Part B: Cybernetics, vol. 27, pp. 208-215.

Spall, J.C. (1992). "Multivariate Stochastic Approximation Using a Simultaneous Perturbation Gradient Approximation." *IEEE Transactions on Automatic Control*, Vol. 37, No. 3, pp. 332-341.

Spall, J.C. (1997a). "Accelerated Second-Order Stochastic Optimization Using only Function Measurements." *Proceedings of the 36th Conference on Decision & Control*, San Diego, CA, pp. 1417-1424.

Spall, J.C. (1997b). "A One-measurement From of Simultaneous Perturbation Stochastic Approximation." *Automatica*, Vol. 33, No. 1, pp. 109-112

Spall, J.C. (1998a). "An Overview of the Simultaneous Perturbation Method for Efficient Optimization." *Johns Hopkins Appl Technical Digest*, Vol. 19, No. 4, pp. 482-492.

Spall, J.C. (1998b). "Implementation of the Simultaneous Perturbation Algorithm for Stochastic Optimization." *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 34, No. 3, pp. 817-823.

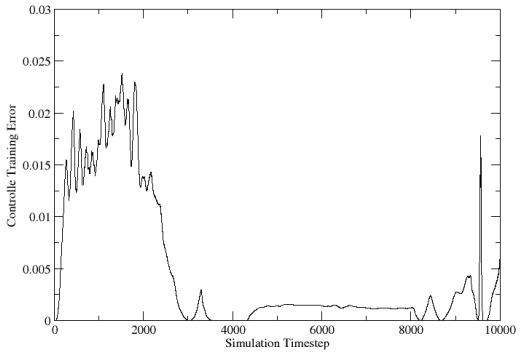
Werbos, P.J., (1974). Beyond Regression: New Tools for Prediction and Analysis in Behavioral Sciences," PhD Thesis, Harvard University, Cambridge, MA.

APPENDIX A ADDITIONAL RESULTS FOR LARGE INITIAL MASS FLOW PERTURBATIONS

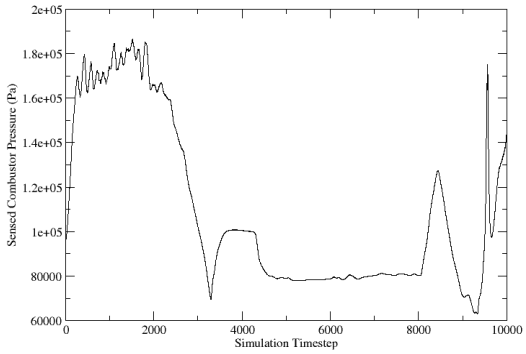
We have performed simulations for larger initial perturbations in order to study the controller behavior under more extreme conditions. This time the mass flow rate perturbation was 0.2kg/s, or 100% of the nominal flow of 0.10 kg/s. We again maintained the same steady flow and boundary conditions, that is:

Mass fuel flow = 0.1kg/s
Inlet : P_{total} = 200,000Pa; T=700C
Outlet : P = 10,000Pa
$P_{request}$ = 100,000Pa

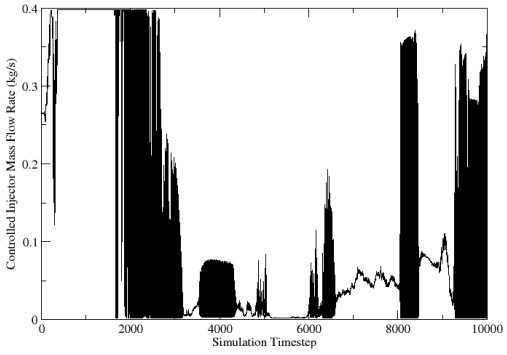
We report the results of two simulations that illustrate the effects of delay taps on controller behavior. Results for delay taps in 3-2 and 12-8 configuration are shown in Figures A1 and A2, respectively.



Controller Training Error



Sensed Averaged Combustor Pressure



Controlled Fuel Flow

Initial Mach Number Field

Max. Shock Position (Shock past throat)>Unstart

Final Mach Number Field

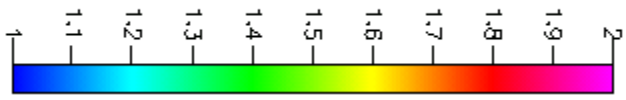
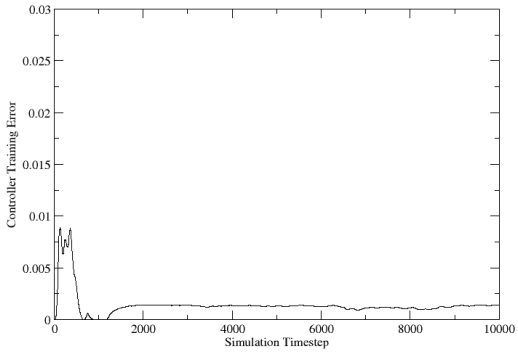
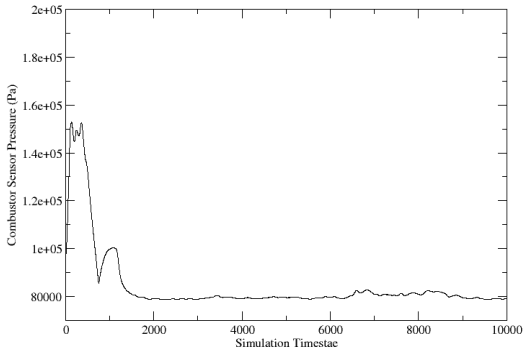


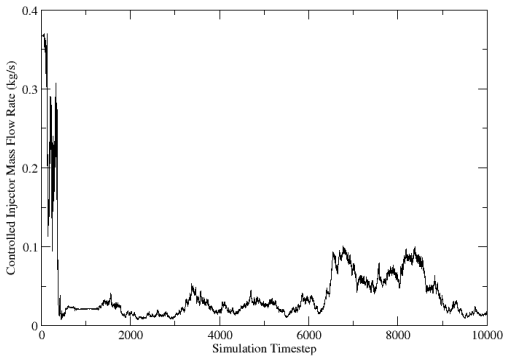
Figure A1) Controller training error, combustor pressure and controlled fuel flow for initial mdot=.2kg/s, delays=3-2. This configuration leads to isolator unstart.



Controller Training Error



Sensed Averaged Combustor Pressure



Controlled Fuel Flow

Initial Mach Number Field

Max. Shock Position (Shock before throat)>No Unstart

Final Mach Number Field

Figure A2) Controller training error, combustor pressure and controlled fuel flow for initial mdot=.2kg/s, delays=12-8. This configuration does not lead to isolator unstart

The results show controller response behavior and sensitivity to the number of input/output delay taps. Even large mass flow perturbations, the controller appears capable preventing isolater unstart provided the number of input/output delay taps was sufficiently large. We again note that all the simulations presented here were performed with an ***unoptimized*** controller.