Design of Reinforcement Learning based Incremental Flight Control Laws for the Cessna Citation II(PH-LAB) Aircraft

Ramesh Konatala

TU Delft

Design of Reinforcement Learning based Incremental Flight Control Laws for the Cessna Citation II(PH-LAB) Aircraft

by

Ramesh Konatala

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday January 30, 2020 at 10:00 AM.

Student number:	4749146	
Project duration:	April 1, 2018 – January 30, 2020	
Thesis committee:	Dr. ir. E. van Kampen,	TU Delft, Supervisor
	Dr. ir. Q.P Chu,	TU Delft, Chairman
	Dr.ir. E. Mooij,	TU Delft, External examiner
	Bo Sun, MSc	TU Delft

Thesis work is partly conducted at DLR, Oberpfaffenhofen

TU Delft

Acknowledgement

This thesis concludes my two years of Masters in Aerospace Engineering at TU Delft,Netherlands. I would like to thank Dr. Erik-Jan Van Kampen and Dr. Gertjan Looye for providing me the opportunity to conduct this thesis at the department of Aircraft System Dynamics, German Aerospace Center(DLR) in Oberpfaffenhofen, Germany.

I would like to thank Dr. Erik-Jan van Kampen for supervising me from the beginning of this project and his patience in arranging the online meetings. His feedback and discussions during the meetings were very helpful in testing different hypothesis to achieve the goal. I thank Dr. G. Looye, head of the Aircraft System Dynamics department at DLR for offering me this opportunity and helping me in dealing with the administrative work at DLR.

I would like to thank Manuel Pusch for introducing me to the DLR through the internship. Work during internship helped to me to gain better understanding of practical aspects in controller implementation and this experience helped me for my thesis work. I would like to thank Paul Acquatella, Daniel Milz, Christian Weiser, Twan Keijzer for the discussions and support during the thesis.

I would like to thank Lisa, Carsten, Alia, Hemjyoti, Juan, Jose, Julianne, Jonas, Phelix whom I met in Munich and for the fun activities. It is a very interesting experience to meet people from different cultural backgrounds. Also I thank fellow students from Control and Simulation group for helping me in many occasions. A big thanks to my friends back home. It was harder than I expected to stay away from home for such long period and I thank them for their support during this phase. I thank my parents and my sister who have been an important part of my life and for their support throughout my study.

Ramesh Konatala
Delft, January 2020

Contents

List of Acronyms

ACD Actor Critic Designs.

ADP Approximate Dynamic Programming.

AI Artificial Intelligence.

ANN Artificial Neural networks.

DASMAT Delft University Aircraft Simulation Model and Analysis Tool.

DP Dynamic Programming.

FBW Fly-By-Wire.

FCC Flight Control Computer.

FCL Flight Control Law.

FCS Flight Control System.

iADP Incremental Approximate Dynamic Programming.

IBS Incremental Backstepping.

IFC Intelligent Flight Control.

INDI Incremental Nonlinear Dynamic Inversion.

LADP Linear Approximate Dynamic Programming.

LOC-I Loss of Control-In Flight.

LQR Linear Quadratic Regulator.

LQT Linear Quadratic Tracking.

LTI Linear-Time-Invariant.

ML Machine Learning.

NDI Nonlinear Dynamic Inversion.

PCH Pseudo Control Hedging.

RL Reinforcement Learning.

List of Symbols

α	Angle of attack
β	Sideslip angle
δ_a	Aileron deflection
δ_e	Elevator deflection
δ_r	Rudder deflection
γ	Discount/forgetting factor
μ	Deterministic policy
ϕ, θ, ψ	Roll, Pitch and Yaw
π	Policy
τ	Time constant
Θ, Cov	Parameter matrix, Covariance matrix
F_t, G_t	State matrix, Input matrix
J	Cost-to-go
P	Kernel matrix
p, q, r	Roll, Pitch and Yaw rate
Q, R	Weighting matrices
R	Reward
U	Control input
V	Value function estimate
V_{tas}	True airspeed
X	State
γ	Flight path angle
h	Altitude
n_z	Load factor

List of Figures

1 Introduction

1.1. Motivation

Last few decades has seen a significant growth in the number of passengers using Air transport from 310 million in 1970 to 4.233 billion in 2018[3] and this number is predicted to reach 8.2 billion by 2037[1]. This surge in the air traffic growth and increased complexity of the modern day aircraft systems made flight safety a priority in the modern day aviation. According to the International Civil Aviation Organization (ICAO), a statistical analysis on risk category effecting the flight safety(Fig. 1.1) show that most number of fatalities are due to the Loss of Control-In Flight (LOC-I). Designing a Flight Control System (FCS) that is "resilient to system failures, external disturbances, inappropriate control inputs by the crew and/or autoflight systems" is one of the ways to ensure flight safety by preventing LOC-I[5].

Modern day FCS is designed using Fly-By-Wire (FBW) system which replaced the manual flight control with mechanical systems by electronic wiring. The FBW system is equipped with a Flight Control Computer (FCC) which interprets the pilot's inputs as the desired outcome and converts these commands to the appropriate control surface actions for the actuators based on a Flight Control Law (FCL). The FBW system together with FCC has enabled to design control laws suitable for complex aerospace systems reducing pilot work load and reduced use of mechanical systems thus enhancing the flight safety. However the underlying FCL should be both adaptive and robust to cope up with unforeseen situations or failure.

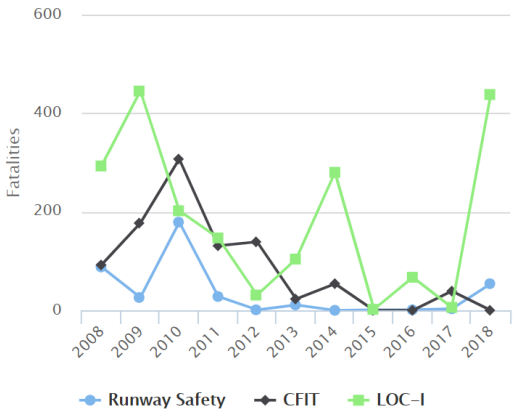


Figure 1.1: Share of Fatalities by Risk Category[2]
Scheduled Commercial flights on airplanes above 5.7t only

Classical control theory based FCL is a popular method to design FCS which can ensure good system performance in nominal conditions. The system to be controlled is linearized around an operating point within the flight envelope and tools from frequency domain approach like Root-locus,bode analysis, Nyquist theory etc,. can be used to design controller with appropriate parameters. To ensure satisfactory controller performance for different flight conditions, techniques like gain scheduling can be used to design linear control laws at different operating points. Some of the limitations of using this approach from flight safety perspective are : dependency on accurate model of the aircraft which is difficult to obtain in reality, performance degradation due to failures, uncertanities and nonlinearities, high development costs involved due to scheduling, complex tuning process due to high number of parameters involved. In the context of control a fundamental difference between others and aerospace systems is that the aerospace systems are typically nonlinear and the nonlinearities in the systems are further accentuated by the faults. To mitigate this issue nonlinear control theory based FCL have become popular. Some of the popular methods include Nonlinear Dynamic Inversion (NDI)[12] or feedback linearization which uses nonlinear feedback to cancel out nonlinearities within the system and then control the linear

system. This approach makes it suitable to use NDI for different operating conditions without using gain scheduling and autoflight control laws based on NDI has been evaluated through flight tests[23] . As perfect cancellation of nonlinearities require accurate modelling of the system another robust control approach based on Lyapunov theory called Backstepping[27] has become popular where additional freedom of excluding nonlinearities which destabilize the system within the closed loop is available. Although NDI and Backstepping methods provide some tolerance to model uncertainites during an event of failure or a fault significant model errors might be introduced which impact the control output.

To reduce the model dependency of the above methods Incremental model based NDI and backstepping versions viz,. Incremental Nonlinear Dynamic Inversion (INDI) and Incremental Backstepping (IBS) are proposed which are found to increase the robustness against model uncertainties[26] [4]. In these incremental based methods, the model to be inverted is written in an incremental form using Taylor series expansion and an incremental control input is evaluated at every time step. FCL based on INDI and IBS have been developed and successfully flight tested at German Aerospace Center(DLR)[14][16][24]. Some of the assumptions involved in the design of INDI and IBS include time scale separation principle where it is assumed that the change in control is significantly faster than the change in states and the availability of the control effectiveness matrix. Although the model dependency has been reduced with the incremental form, these methods still require a model or a partial model to be controlled which is a problem to consider in view of designing fault tolerant FCS.

Researchers have been exploring intelligent,self-learning control methods considering the limitations of above mentioned control methods to design a FCS that is resilient to faults or failures. The Intelligent Flight Control (IFC) is a broader term which has roots in the fields of control theory and artificial intelligence for designing a FCS. Artificial Intelligence (AI) is a term used to describe the intelligent behaviour exhibited by artificial agents like Dynamic Programming (DP) is a tool which can be used to solve for the optimality of an RL problem using Bellman equation[6]. Optimal control is a branch of control systems that is used to achieve an optimal control law such that a defined objective function is optimized over a period of time. In RL based control, a control problem is defined as an objective to achieve and the optimal control law is achieved by solving for optimization using DP techniques[8]. In practice achieving control for real systems using DP is not viable as DP assumes a perfect known model of the system and the high computational expense needed to solve optimization problem for larger state space. The former problem in the context of RL is referred to as *"Curse of Dimensionality"*.

Approximate Dynamic Programming (ADP) combines generalization methods like function approximators with DP techniques rendering these methods suitable to achieve optimal control for larger state space systems. The ADP methods are used to achieve feedback control for dynamical systems using a cost-to-go function with online learning capability using data observed along the system trajectories[22][21]. The ADP method is extended to solve for reference tracking problem[18]. Although these methods are model free they assume a linear time-invariant model of the system to be controlled thus making it difficult to extend these methods for nonlinear aerospace systems. Based on theory from INDI and IBS incremental version of ADP is proposed referred to as iADP for stabilizing control problem using a quadratic cost function approximation. This method which uses an online identified local linearized model using Least squares or Recursive Least squares approach making this method suitable for nonlinear time varying systems. Also iADP does not need knowledge of control effectiveness matrix and does not assume time scale principle separation. This method is further extended to achieve more general reference tracking control[11] and two algorithms with full state feedback and output feedback are proposed and the control approach is verified on a F-16 aircraft model. Another family of ADP methods referred to as ACD's adopt Actor-Critic architectures and use a black box based function approximators like Neural networks for the cost function. Incremental versions of these ACD's are proposed which has online and adaptive learning capability[38][39]. However incremental versions of ADP controllers is yet to be verified on a real system.

The FCL's designed at the Aircraft System Dynamics department, DLR Oberpfaffenhofen are tested and validated through flight tests on a fixed wing Cessna Citation II PH-Lab aircraft together with TU Delft[14][16][24]. The aim of the research project is to extend the RL based controller to Cessna Citation II aircraft and evaluate the viability of using this controller for a real system. To attend this objective additional research has to carried out on integrating the RL based controller within FCS Cessna Citation II aircraft and study the effects of typical aircraft characteristics like sensor, actuator dynamics, time delays on the controller.

1.2. Research objective and Questions

As detailed in Section 1.1 the motivation for designing RL based IFC methods for flight control is clear and these methods have to be tested on a real aircraft for their viability through flight tests. A necessary step in extending these methods to a real system is to assess the controller performance considering real world scenarios. Along these lines the following research objective is formulated.

Research Objective

The objective of this research is to address the practical issues of implementing an online reinforcement learning controller on a fixed wing Cessna Citation II aircraft by investigating the controller performance in a close to realistic simulated environment of Cessna Citation II platform and proposing possible solutions to ensure desired controller performance.

To achieve the formulated research objective the following research questions and sub-questions are framed and are answered during the research study conducted.

Research Question

1. How to design a reinforcement learning based Flight Control System for Cessna Citation II aircraft?
   (a) What is the most suitable reinforcement learning based control method that can be implemented on Cessna Citation II aircraft with online learning capability?
   (b) What are the characteristics and limitations of this control method?
   (c) Which control law architecture is suitable to integrate this controller into the Flight Control System of Cessna Citation II aircraft?
2. What measures are required to ensure desired controller performance in a close to realistic simulated environment of the Cessna Citation II platform?
   (a) What phenomenon are to be considered to emulate reality within simulated environment of Cessna Citation platform?
   (b) What are the relevant criterion to consider to assess the controller performance?
   (c) What is the effect of the selected phenomenon on the controller performance criterion?
   (d) What are the measures required to mitigate the observed performance degradation?

1.3. Outline of the report

The report is structured as follows. Chapter 2 contains the scientific paper which summarizes the main research work and findings. The paper includes derivation of iADP algorithms, Citation II aircraft model, Sensor and actuator models used for realistic simulation, iADP controller based FCS design for Citation II, findings from simulations. Readers are advised to read reminder of the chapters before the paper, if they are not familiar with fundamentals in Reinforcement Learning. Chapter 3 contains an overview of the conducted literature review. It includes fundamentals of Reinforcement learning along with some algorithm implementations, ADP for achieving feedback control, mathematical derivation of iADP algorithms and state of the art review on Intelligent Flight Control methods. Chapter 4 includes simulation results of LADP and iADP algorithm implementation on a Missile System for Longitudinal control. Finally in Chapter 6 the main conclusions drawn from the research project and answers to the research question are provided along with some recommendations for future work.

2

Scientific paper

Reinforcement Learning based Online Adaptive Flight Control for the Cessna Citation II(PH-LAB) Aircraft

Ramesh Konatala*
Delft University of Technology, P.O. Box 5058, 2600 GB Delft, The Netherlands

Online Adaptive Flight Control is interesting in the context of growing complexity of aircraft systems and their adaptability requirements to ensure safety. An Incremental Approximate Dynamic Programming (iADP) controller combines reinforcement learning methods, optimal control and online identified incremental model to achieve optimal adaptive control suitable for Nonlinear Time-Varying systems. The main contribution of this paper is twofold. Firstly, the iADP controller is designed to achieve automatic online rate control to track pilot commands via setpoints provided by the manual outer loop on Citation II Aircraft model. Secondly, to assess the controller performance in the presence of sensor dynamics and actuator dynamics, an analysis is carried out to identify causes of any performance degradation. The simulation results from iADP longitudinal control using full state feedback indicate that the discretization of sensor signals, sensor bias and transport delays did not have any significant effect on the controller performance or on the incremental model identification. However noisy signals and sensors delays are found to cause controller performance degradation. Appropriate filtering of signals resulted in better estimation of the incremental model subsequently improving the controller performance due to noisy signals. Control performance degradation due to sensor delays should be addressed in future before conducting flight tests on Citation II Aircraft.

Nomenclature

α, β	Angle of attack, Sideslip angle
X, U	State, Control input
R	Reward
V	Value function estimate
π	Policy
μ	Deterministic policy
J	Cost-to-go
P	Kernel matrix
$\delta_a, \delta_e, \delta_r$	Aileron, Elevator and Rudder deflections
γ	Discount factor
τ	Time constant
Q, R	Weighting matrices
Θ, Cov	Parameter matrix, Covariance matrix
p, q, r	Roll, Pitch and Yaw rate
ϕ, θ, ψ	Roll, Pitch and Yaw
V_{tas}, h, γ, n_z	True airspeed, Altitude, Flight path angle and Load factor
F_t, G_t	State matrix, Input matrix

I. Introduction

The surge in the air traffic growth and increased complexity of the modern day aircraft systems in recent decades made flight safety a priority in the modern day aviation. According to the International Civil Aviation Organization (ICAO), a statistical analysis on risk category effecting the flight safety show that most number of fatalities are due to the Loss of Control-In Flight (LOC-I). Designing a Flight Control System (FCS) that is resilient to system failures,

*MSc Student, Control and Simulation Division, Faculty of Aerospace Engineering, Delft University of Technology

external disturbances, inappropriate control inputs by the crew and/or autoflight systems is one of the ways to ensure flight safety by preventing LOC-I[1].

Modern day FCS is designed using Fly-By-Wire (FBW) system and Flight Control Computer (FCC) which interprets the pilot's inputs as the desired outcome and converts these commands to the appropriate control surface actions for the actuators based on a Flight Control Law (FCL). However the underlying FCL should be both adaptive and robust to cope up with unforeseen situations or failure. Incremental model based NDI and backstepping versions viz,. Incremental Nonlinear Dynamic Inversion (INDI) and Incremental Backstepping (IBS) are some of the popular control methods designed with the aim of improving flight safety. In these incremental based methods, the model to be inverted is written in an incremental form using Taylor series expansion and an incremental control input is evaluated at every time step. These methods are found to increase the robustness against model uncertainties[2] [3] and similar observations are validated through flight tests in cooperation with the Aircraft System Dynamics department, DLR Oberpfaffenhofen on a fixed wing Cessna Citation II PH-Lab aircraft[4][5][6]. Another interesting approach in designing FCS is using Reinforcement Learning (RL) based FCL. Active research is going on in RL based control to achieve model free nonlinear optimal control with online learning capability. In RL based control, a control problem is defined as an objective to achieve and the optimal control law is achieved by solving for optimization using Dynamic Programming (DP) techniques[7]. In practice achieving control for real systems using DP is not viable as DP assumes a perfect known model of the system and high computational expense needed to solve optimization problem for larger state space. The former problem in the context of RL is referred to as *"Curse of Dimensionality"*.

Approximate Dynamic Programming (ADP) combines generalization methods like function approximators with DP techniques rendering these methods suitable to achieve optimal control for larger state space systems. The ADP methods are used to achieve feedback control for dynamical systems using a cost-to-go function with online learning capability using data observed along the system trajectories[8][9]. This ADP method is further extended to solve for reference tracking problem[10]. Although these methods are model free, they assume a linear time-invariant model of the system to be controlled, thus making it difficult to extend these methods for nonlinear aerospace systems. Based on theory from INDI and IBS, an incremental version of ADP is proposed, which is referred to as Incremental Approximate Dynamic Programming (iADP) for stabilizing control problem using a quadratic cost function approximation. This method which uses an online identified local linearized model using Least squares or Recursive Least squares approach, making this method suitable for nonlinear time varying systems. This method is further extended to achieve more general reference tracking control[11] and two algorithms with full state feedback and output feedback are proposed and the control approach is verified on a F-16 aircraft model. However the iADP controller is yet to be verified on a real system.

The aim of this paper is to extend the RL based controller to Cessna Citation II aircraft and evaluate the viability of using this controller for a real system. To attend this objective, additional research has to carried out on integrating the RL based controller within FCS Cessna Citation II aircraft and study the effects of typical aircraft characteristics like sensor, actuator dynamics, time delays on the controller. The main contributions of this paper are as follows: Firstly, iADP controller is integrated into FCS of Citation II Aircraft to achieve automatic online rate control. Secondly, iADP controller performance is assessed considering sensor and actuator dynamics. The controller performance is evaluated for longitudinal control of the aircraft using full state feedback and output feedback.

The contents of this paper are structured as follows. In Section II, basic concepts of Reinforcement Learning are discussed followed by derivation of the iADP algorithms. In Section III, Cessna Citation II aircraft model along with sensor and actuator models used for simulations is discussed. FCS design of iADP controller for Citation II is explored in Section IV. Section V contains the results from the controller evaluation on the aircraft model. Finally, in Section VI main conclusions from this paper are presented.

II. Reinforcement Learning for optimal adaptive control

Optimal control design involves designing a controller to optimize a cost function that characterizes the desired behaviour of a system. Techniques like Linear Quadratic Gaussian(LQG) are often used to achieve optimal control through a quadratic cost function and a linear model of the system to be controlled. It is desirable to have a controller that does not completely rely on the model of the system as it is difficult to obtain a perfect model of the system due to modelling uncertainties. Optimal adaptive control methods address this issue by redesigning optimal controllers for varying models of the system which are identified using system identification techniques. A direct way to achieve this optimal adaptive control is a model free controller that learns the control scheme online using real time observations along system trajectories[12] and Reinforcement Learning(RL) schemes are found to be useful in designing this direct approach.

Reinforcement Learning

RL process essentially involves an agent interacting in an environment which learns to choose actions such that a certain goal/objective is reached. The RL agent achieves this through a trail and error search method and memorization of situations/states and suitable actions reinforced through the rewards yielded from the environment. Many of the RL algorithms adopt an actor-critic architecture as shown in Fig. (1) which enables online learning through real time observations. In an actor-critic setting, the actor does the job of control policy (mapping from system states to the control action inputs) implementation with the policy updates provided by the critic. The critic evaluates the current policy by updating the value associated with the current state using the cost information provided by the system/environment and updates the control policy for the actor such that the cost associated with the new policy is smaller than the previous one.

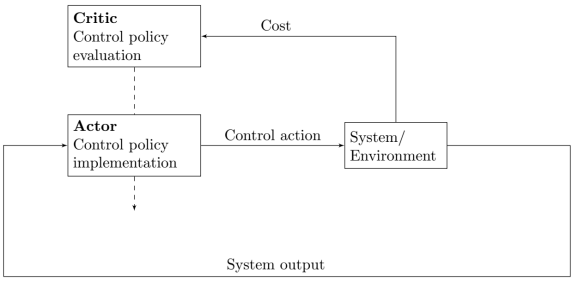


Fig. 1 Actor critic structure of RL agent

The RL problem is formalized mathematically as a Markov Decision Process(MDP) which assumes that the process obeys the memoryless Markov property, which is the concept that a future state is independent of the preceding states given the current state. The MDP is then solved for optimality using techniques like Dynamic Programming(DP).

To solve a DP problem it should have an optimal substructure and overlapping sub problems. The DP algorithms require the complete knowledge of the environment to estimate the state value functions. However it is not practical to have knowledge of the environment in all the cases and thus methods like Monte Carlo(MC) Methods which attain the optimal behaviour through experience can be used to attend RL problem. Monte Carlo refers to the use of random sampling methods to approximate numerical results. Monte Carlo methods in the context of RL refer to the learning of the agent in an environment through experience using sample returns observed. Thus instead of evaluating value function for all the states using a perfect known model of the environment we estimate the value of the states through some policy using the experience gained while visiting those states in an episode. The MC methods differ from DP methods in two ways. Firstly the agent learns from experience instead of state space sweep to estimate value functions and secondly the value functions are estimated directly from returns instead of other value estimates. Temporal Difference(TD) learning combines the advantage of sampling from experience in MC methods and learning from incomplete episodes in DP methods. While in the MC prediction we need to wait till we finish the episode to estimate the value function for a particular state, in a TD prediction we can instead estimate the value of the state by taking one step ahead and then using the value estimate of the new state that we have landed in. Another difference between TD and MC method is that the TD learning algorithm exploits the Markov property by first building an approximate model of the MDP and then converging the solution from the data for the estimated MDP.

Consider a MDP : (X, U, P, R) where X denotes set of states, U denotes set of control actions/inputs. The conditional probability of the MDP to transition from state $x \in X$ to $x' \in X$ by taking action $u \in U$ is $P^u_{xx'} = Pr\{x'|x, u\}$ and the expected immediate cost necessary for the transition is $R^u_{xx'}$. The control policy or action strategy $\pi(x, u) = Pr\{u|x\}$ is the mapping from states X to actions U. The policy can be stochastic $\pi(x, u)$ where there is non zero probability of selecting more than one control u or deterministic $\mu(x)$ policy which admits only one control given state x. The goal of the RL problem is to find the optimal policy π_*(or μ_* for deterministic optimal policy) which minimizes the expected future cost. Extending the MDP framework to a dynamical system which evolves through time we assume that the state transitions happen at discrete time steps : $k, k+1, k+2....$ The one step cost necessary for the transition $x_k \rightarrow x_{k+1}$ by taking action u_k is defined by $r_k = r_k(x_k, u_k, x_{k+1})$. The discounted infinite horizon cost J_k provides measure of sum of future costs incurred by the dynamical system to evolve through time in the future and is given by

$$J_k = \sum_{i=k}^{\infty} \gamma^{i-k} r_i \tag{1}$$

where $0 \leq \gamma \leq 1$ discounts the costs incurred in further in future. Consider the RL agent selects control actions at every time step k following control policy $\pi_k(x_k, u_k)$. The value for a policy V^π is defined as the expected value of the future cost for a dynamical system starting from state x at time k and following the policy $\pi(x, u)$ subsequently, thus providing a measure of the value being in state x with the policy being followed as π. The value function is given by,

$$
\begin{aligned}
V^{\pi}(x) &= E_{\pi}\{J_k | x_k = x\} \\
&= E_{\pi}\left\{\sum_{i=k}^{\infty} \gamma^{i-k} r_i | x_k = x\right\} \\
&= E_{\pi}\left\{r_k + \gamma \sum_{i=k+1}^{\infty} \gamma^{i-(k+1)} r_i | x_k = x\right\}
\end{aligned} \tag{2}
$$

Here E_{π} denotes expectation of the value over all possible transitions conditional on policy π being followed. The Bellman equation is a fundamental concept in solving reinforcement learning problems which helps in arriving at optimal policies using experiences received further in time. The Bellman equation can be obtained from (2) as follows,

$$
V^{\pi}(x) = \sum_{u} \pi(x,u) \sum_{x'} P^{u}_{xx'}[R^{u}_{xx'} + \gamma V^{\pi}(x')] \tag{3}
$$

The value function should satisfy the Bellman equation at all stages of time. This equation can be interpreted as the relation between the current value of state $x = x_k$ and the value of state $x' = x_{k+1}$ whilst following policy $\pi(x,u)$. For an ergodic dynamical system it is proved that the MDP will have an deterministic optimal policy [13] to minimize the expected future cost. *Policy evaluation* is the procedure of arriving at the value of a policy which can be obtained using Bellman equation (3). If we know the value for a given policy $\pi(x,u)$ we can find another policy π' which is at least better than π and this step is referred to as *Policy improvement* which can be written as

$$
\pi'(x,u) = \underset{u}{argmin} \sum_{x'} P^{u}_{xx'}[R^{u}_{xx'} + \gamma V^{\pi}(x')] \tag{4}
$$

The optimality is reached when $\pi'(x,u) = \pi(x,u)$ and according to the Bellman's optimality principle[14] the optimal control policy and the optimal cost can be written as

$$
u^{*} = \underset{u}{argmin} \sum_{x'} P^{u}_{xx'}[R^{u}_{xx'} + \gamma V^{*}(x')] \tag{5}
$$

$$
V^{*}(x) = \underset{\pi}{min} \sum_{x'} P^{u}_{xx'}[R^{u}_{xx'} + \gamma V^{*}(x')] \tag{6}
$$

The objective of the RL problem is to arrive at the optimal control policy and this can be achieved using two iterative algorithms which use mapping between value and policy through policy evaluation and policy improvement steps. *Policy iteration(PI)* is the method of solving the RL problem through repeated sequence of policy evaluation and policy improvement steps until optimal solution is found. *Value iteration(VI)* is a special case of policy iteration method where instead of waiting for the exact convergence of policy evaluation, it is truncated to just one iteration and based on the approximate value function obtained policy improvement is done and the entire process is repeated till convergence. These DP based algorithms require the state transition probabilities $P^{u}_{xx'}$ and the cost $R^{u}_{xx'}$ of the MDP to arrive at the optimal control policy and can only be solved offline. To design optimal adaptive controllers it is desirable to have a method which does not rely on the full knowledge of the system. Temporal Difference(TD) is a model free RL method when applied for control systems has the capability of online learning using observed data measured along the system trajectories, which can be used to design optimal adaptive controllers.

In a TD method, the policy evaluation step is done using observed data collected along one sample path of MDP which the agent follows. The equation (3) now becomes a deterministic equation and the Bellman equation for TD can be written as

$$
V^{\pi}(x_k) = r_k + \gamma V^{\pi}(x_{k+1}) \tag{7}
$$

where the observed data is (x_k, r_k, x_{k+1}) at time step k. The TD error is given by the equation (8) and the objective is to update the value such that the TD error is minimized using PI or VI.

$$
e_k = -V^{\pi}(x_k) + r_k + \gamma V^{\pi}(x_{k+1}) \tag{8}
$$

For discrete systems the TD method provides exact solutions which can be arranged in a n-dimensional lookup table where n is the size of the state vector. In control systems we deal with continuous state and control spaces and when discretized, the state-space increases the number of states in the lookup table exponentially, a phenomenon referred to as "curse of dimensionality". This problem is addressed by approximating the value function using unknown parameters and suitable approximation structure. For a linear system the value function can be approximated to be quadratic in state[15] as shown in equation (9) which benefits from having one local/global minimum

$$V^{\pi}(x_k) = x_k^T P x_k \tag{9}$$

where P is a positive definite symmetric kernel matrix. These methods form class of Approximate Dynamic Programming(ADP) methods. The one-step cost r_k can be constructed based on the requirements of the control to be achieved viz, regulation, tracking with minimum control. A standard form is a quadratic energy function represented as equation (10) where Q,R are state weighting and control weighting matrices which provides trade off between objective to be achieved and the control effort required.

$$r_k = Q(x_k) + u_k^T R u_k \tag{10}$$

Because of the quadratic value function assumption, the ADP methods are suitable for dynamical systems which are Linear Time Invariant(LTI). However as most of the aerospace systems are nonlinear it is desirable to design controller which can deal with system nonlinearities and model uncertainties. Incremental model techniques approximate the original nonlinear dynamical system to linear time varying system around an operating point using first order Taylor series expansion. ADP methods are combined with incremental approach to design optimal controllers suitable for nonlinear systems referred to as Incremental Approximate Dynamic Programming (iADP) controllers. As these iADP controllers use only observed data for achieving the control iADP controllers can be classified as model free methods that has online learning capability.

A. Incremental Approximate Dynamic Programming for Tracking control

Here the methodology of extending iADP controllers to solve more general tracking control problems will be explained considering both the availability of full state observations and partial observability conditions.

1. Incremental model for Nonlinear system

Consider a Non-linear continuous system represented as follows:

$$\begin{aligned} \dot{x}(t) &= f[x(t), u(t)] \\ y(t) &= h[x(t)] \end{aligned} \tag{11}$$

where $f[x(t), u(t)] \in R^n$, $u(t) = R^m$ and output measurements are obtained using the measurement vector $h[x(t)] \in R^p$. As in practice we work with the discrete systems for achieving the control the above nonlinear system is discretized using a high sampling frequency and is represented as (12).

$$\begin{aligned} x_{k+1} &= f(x_k, u_k) \\ y_k &= h(x_k) \end{aligned} \tag{12}$$

The objective is to design the iADP controller such that the system tracks a reference signal. Let the reference trajectory dynamics be represented for a discrete case as

$$\begin{aligned} r_{k+1} &= f_r(r_k) \\ y_k^r &= h_r(r_k) \end{aligned} \tag{13}$$

where $f_r(r_k) \in R^l$. By representing the reference signal in this form one can generate large class of reference trajectories. Augmenting the system dynamics with the reference dynamics we can generate the following augmented nonlinear system

$$X_{k+1} = \begin{bmatrix} x_{k+1} \\ r_{k+1} \end{bmatrix} = \begin{bmatrix} f(x_k, u_k) \\ f_r(r_k) = \end{bmatrix} = t(X_k, u_k) \tag{14}$$

where $t(X_k, u_k) \in R^{n+l}$. The quadratic cost function will now be a quadratic in augmented state X_k. Linearizing the above augmented nonlinear discrete system around X_0, u_0 by taking the first order Taylor series expansion we get

$$X_{k+1} = t(X_k, u_k) \approx t(X_0, u_0) + \left.\frac{\partial t(X_k, u_k)}{\partial X_k}\right|_{X_0,u_0} (X_k - X_0) + \left.\frac{\partial t(X_k, u_k)}{\partial u_k}\right|_{X_0,u_0} (u_k - u_0) \tag{15}$$

As it is assumed that the discretization is done at a high sampling frequency we can consider Δt to be very small and can approximate $X_{k-1} \approx X_k$ and can replace X_0, u_0 with X_{k-1}, u_{k-1} to get (16)

$$\begin{aligned} X_{k+1} - X_k &\approx T(X_{k-1}, u_{k-1})(X_k - X_{k-1}) + G(X_{k-1}, u_{k-1})(u_k - u_{k-1}) \\ \Delta X_{k+1} &\approx T_{k-1}\Delta X_k + G_{k-1}\Delta u_k \end{aligned} \tag{16}$$

where $T_{k-1} = T(X_{k-1}, u_{k-1}) \in R^{(n+l)\times(n+l)}$ is the system matrix and $G_{k-1} = G(X_{k-1}, u_{k-1}) \in R^{(n+l)\times m}$ is the control effectiveness matrix. This regression model represented by F_{k-1}, G_{k-1} can be identified using Recursive Least Squares(RLS) techniques which provides a Linear Time Variant(LTV) approximation to the original model.

2. Full state feedback

For dynamical systems where full state measurements are available the observed measurements can be written as:

$$Y_k = \begin{bmatrix} y_k \\ y_k^r \end{bmatrix} = X_k \tag{17}$$

Using the utility function (10) for achieving tracking control and extending the concept of Bellman equation (7) to the incremental model we get the optimal Value function (18)

$$V^*(X_k) = \min_{\Delta u_k}[(y_k - y_k^r)^T Q(y_k - y_k^r) + (u_{k-1} + \Delta u_k)^T R(u_{k-1} + \Delta u_k) + \gamma V^*(X_{k+1})] \tag{18}$$

Where the optimal control at time step k is given by (19)

$$\Delta u^* = \underset{\Delta u_k}{argmin}[(y_k - y_k^r)^T Q(y_k - y_k^r) + (u_{k-1} + \Delta u_k)^T R(u_{k-1} + \Delta u_k) + \gamma V^*(X_{k+1})] \tag{19}$$

using the quadratic value function approximation (9) we get (20)

$$\begin{aligned} X_k^T P X_k &= (y_k - y_k^r)^T Q(y_k - y_k^r) + u_k^T R u_k + \gamma X_{k+1}^T P X_{k+1} \\ &= (y_k - y_k^r)^T Q(y_k - y_k^r) + (u_{k-1} + \Delta u_k)^T R(u_{k-1} + \Delta u_k) + \\ &\quad \gamma(X_k + T_{k-1}\Delta X_k + G_{k-1}\Delta u_k)^T P(X_k + T_{k-1}\Delta X_k + G_{k-1}\Delta u_k) \end{aligned} \tag{20}$$

For optimal control we can set the derivative of the above cost function with respect to Δu_k to 0 and we get the optimal control law (21)

$$\Delta u_k = -(R + \gamma G_{k-1}^T P G_{k-1})^{-1}[R u_{k-1} + \gamma G_{k-1}^T P X_k + \gamma G_{k-1} P T_{k-1}\Delta X_k] \tag{21}$$

The VI algorithm for iADP-FS is given by the Algorithm (1)

Algorithm 1 iADP for tracking control using Full State Feedback[11]

Initialize a arbitrary control policy $\Delta u_k^0 = \mu(X_k)$
repeat
  Value Update Step: $X_k^T P X_k = (y_k - y_k^r)^T Q(y_k - y_k^r) + u_k^T R u_k + \gamma X_{k+1}^T P X_{k+1}$
  Policy Improvement Step: $\Delta u_k = -(R + \gamma G_{k-1}^T P G_{k-1})^{-1}[R u_{k-1} + \gamma G_{k-1}^T P X_k + \gamma G_{k-1} P T_{k-1}\Delta X_k]$
until Convergence

3. Output feedback

Often in practice, as measurement of full system states is not available, controller design using input-output measurement data over suitable time horizon is desirable. In this method the input output measurements are used to indirectly construct the state information, under the assumption that the system is observable. The measured data is then used to arrive at the optimal control using iADP method.

Consider we measure the data at N time steps between interval $[k-N, k]$, using equations (16) and (25) we can write (22),

$$\Delta X_k = \begin{bmatrix} \Delta x_k \\ \Delta r_k \end{bmatrix} \approx \begin{bmatrix} \tilde{F}_{k-2,k-N-1} & 0 \\ 0 & \tilde{D}_{k-2,k-N-1} \end{bmatrix} \begin{bmatrix} \Delta x_{k-N} \\ \Delta r_{k-N} \end{bmatrix} + \begin{bmatrix} U_N \\ 0 \end{bmatrix} \bar{\Delta u}_{k-1,k-N} \tag{22}$$

where $\tilde{F}_{k-a,k-b} = \Pi_{i=k-a}^{k-b} F_i$ and $\tilde{D}_{k-a,k-b} = \Pi_{i=k-a}^{k-b} D_i$, the input-output measurements captured over the time horizon [k-N,k] and the controllability matrix U_N are given by equations (23) and (24) respectively

$$\bar{\Delta u}_{k-1,k-N} = \begin{bmatrix} \Delta u_{k-1} \\ \Delta u_{k-2} \\ \vdots \\ \Delta u_{k-N} \end{bmatrix} \in R^{mN}, \qquad \bar{\Delta y}_{k,k-N+1} = \begin{bmatrix} \Delta y_k \\ \Delta y_{k-1} \\ \vdots \\ \Delta y_{k-N+1} \end{bmatrix} \in R^{pN} \tag{23}$$

$$U_N = \begin{bmatrix} G_{k-2} & F_{k-2}G_{k-3} \dots \tilde{F}_{k-2,k-N}G_{k-N-1} \end{bmatrix} \in R^{n \times mN} \tag{24}$$

Linearizing the output of the nonlinear system(12) and the reference output of the system (13) using First order Taylor series expansion around x_{k-1} we get (25) and (26) respectively

$$\Delta y_k \approx H_{k-1} \Delta x_k \tag{25}$$

$$\Delta y_k^r \approx H_{k-1}^r \Delta r_k \tag{26}$$

where $H_{k-1} = \frac{\partial h(x)}{\partial x}|_{x_{k-1}} \in R^{p \times n}$ and $H_{k-1}^r = \frac{\partial h^r(x)}{\partial x}|_{x_{k-1}} \in R^{r \times n}$ are the observation matrices. Now using the input-output data from (22) we can write (25) and (26) as follows

$$\begin{aligned} \bar{\Delta y}_{k,k-N+1} &\approx V_N \Delta x_{k-N} + W_N \bar{\Delta u}_{k-1,k-N} \\ \bar{\Delta y}_{k,k-N+1}^r &\approx R_N \Delta r_{k-N} \end{aligned} \tag{27}$$

The matrices V_N, W_N and R_N are given by equations (28), (29) and (30) respectively

$$V_N = \begin{bmatrix} H_{k-1}\tilde{F}_{k-2,k-N-1} \\ H_{k-2}\tilde{F}_{k-3,k-N-1} \\ \vdots \\ H_{k-N}F_{k-N-1} \end{bmatrix} \in R^{pN \times n} \tag{28}$$

$$W_N = \begin{bmatrix} H_{k-1}G_{k-2} & H_{k-1}F_{k-2}G_{k-3} & H_{k-1}\tilde{F}_{k-2,k-3}G_{k-4} & \dots & H_{k-1}\tilde{F}_{k-2,k-N}G_{k-N-1} \\ 0 & H_{k-2}G_{k-3} & H_{k-2}F_{k-3}G_{k-4} & \dots & H_{k-2}\tilde{F}_{k-3,k-N}G_{k-N-1} \\ 0 & 0 & H_{k-3}G_{k-4} & \dots & H_{k-3}\tilde{F}_{k-4,k-N}G_{k-N-1} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 & H_{k-N}G_{k-N-1} \end{bmatrix} \in R^{(p+m) \times N} \tag{29}$$

$$R_N = \begin{bmatrix} H^r_{k-1}\tilde{F}_{k-2,k-N-1} \\ H^r_{k-2}\tilde{F}_{k-3,k-N-1} \\ \vdots \\ H^r_{k-N}F_{k-N-1} \end{bmatrix} \in R^{rN\times l} \tag{30}$$

We can extract Δx_{k-N} and Δr_{k-N} from (27) as follows

$$\begin{aligned} \Delta x_{k-N} &\approx V_N^+(\bar{\Delta y}_{k,k-N+1} - W_N\bar{\Delta u}_{k-1,k-N}) \\ \Delta r_{k-N} &\approx R_N^+\bar{\Delta y}^r_{k,k-N+1} \end{aligned} \tag{31}$$

where $V_N^+ = (V_N^TV_N)^{-1}V_N^T$ and $R_N^+ = (R_N^TR_N)^{-1}R_N^T$ are the pseudo inverses of the respective matrices. Substituting (31) in (22) we get

$$\begin{aligned} \Delta X_k &\approx \begin{bmatrix} \tilde{F}_{k-2,k-N-1}V_N^+ & 0 \\ 0 & \tilde{D}_{k-2,k-N-1}R_N^+ \end{bmatrix}\begin{bmatrix} \bar{\Delta y}_{k,k-N+1} \\ \bar{\Delta y}^r_{k,k-N+1} \end{bmatrix} + \begin{bmatrix} U_N - \tilde{F}_{k-2,k-N-1}V_N^+W_N \\ 0 \end{bmatrix}\bar{\Delta u}_{k-1,k-N} \\ &\approx \begin{bmatrix} U_N - \tilde{F}_{k-2,k-N-1}V_N^+W_N & \tilde{F}_{k-2,k-N-1}V_N^+ & 0 \\ 0 & 0 & \tilde{D}_{k-2,k-N-1}R_N^+ \end{bmatrix}\begin{bmatrix} \bar{\Delta u}_{k-1,k-N} \\ \bar{\Delta y}_{k,k-N+1} \\ \bar{\Delta y}^r_{k,k-N+1} \end{bmatrix} \\ &\approx \begin{bmatrix} M_{\Delta u} & M_{\Delta y} & M_{\Delta y^r} \end{bmatrix}\bar{\Delta Z}_{k,k-N} \end{aligned} \tag{32}$$

Thus augmented state can be reconstructed using the input output data over a certain time horizon using above equation. It can also be shown that the output increments can also be constructed using past data measurements as follows:

$$\begin{aligned} \Delta y_{k+1} &\approx \underline{G}_k\bar{\Delta u}_{k-1,k-N} + \underline{F}_k\bar{\Delta y}_{k-1,k-N} \\ &\approx \underline{G}_{k,11}\Delta u_k + \underline{G}_{k,12}\bar{\Delta u}_{k-1,k-N+1} + \underline{F}_k\bar{\Delta y}_{k,k-N+1} \end{aligned} \tag{33}$$

where $\underline{G}_k \in R^{p\times Nm}$ is the extended control effectiveness matrix, $\underline{F}_k \in R^{p\times Np}$ is the extended system matrix, $\underline{G}_{k,11} \in R^{p\times m}$ and $\underline{G}_{k,12} \in R^{p\times(N-1)m}$ are partitioned matrices from $\underline{G}_k$.
Similarly Δy^r_{k+1} can also be constructed as:

$$\Delta y^r_{k+1} \approx \underline{F}^r_k\bar{\Delta y}_{k-1,k-N} \tag{34}$$

where $\underline{F}^r_k \in R^{r\times Nr}$. We define the quadratic cost to go function using a kernel matrix and the measured data as follows

$$V(\bar{Z}_{k,k-N+1}) = \bar{Z}^T_{k,k-N+1}\bar{P}\bar{Z}_{k,k-N+1} \tag{35}$$

where $\bar{Z}$ contains the input output data given by

$$\bar{Z}_{k,k-N+1} = \begin{bmatrix} \bar{u}_{k-1,k-N} \\ \bar{y}_{k,k-N+1} \\ \bar{y}^r_{k,k-N+1} \end{bmatrix} \in R^{(m+p+r)N} \tag{36}$$

Extending the Bellman equation for tracking control with the incremental model representation, using the input output data we get

$$\bar{Z}^T_{k,k-N+1}\bar{P}\bar{Z}_{k,k-N+1} = (y_k - y^r_k)^TQ(y_k - y^r_k) + (u_{k-1} + \Delta u_k)^TR(u_{k-1} + \Delta u_k) + \gamma\bar{Z}^T_{k+1,k-N+2}\bar{P}\bar{Z}_{k+1,k-N+2} \tag{37}$$

where,

$$\bar{Z}^T_{k+1,k-N+2}\bar{P}\bar{Z}_{k+1,k-N+2} = \begin{bmatrix} u_{k-1}+\Delta u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_k + \Delta y_{k+1} \\ \bar{y}_{k,k-N+2} \\ \bar{y}^r_k + \Delta y^r_{k+1} \\ \bar{y}^r_{k,k-N+2} \end{bmatrix}^T \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} & P_{15} & P_{16} \\ P_{21} & P_{22} & P_{23} & P_{24} & P_{25} & P_{26} \\ P_{31} & P_{32} & P_{33} & P_{34} & P_{35} & P_{36} \\ P_{41} & P_{42} & P_{43} & P_{44} & P_{45} & P_{46} \\ P_{51} & P_{52} & P_{53} & P_{54} & P_{55} & P_{56} \\ P_{61} & P_{62} & P_{63} & P_{64} & P_{65} & P_{66} \end{bmatrix} \begin{bmatrix} u_{k-1}+\Delta u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_k + \Delta y_{k+1} \\ \bar{y}_{k,k-N+2} \\ \bar{y}^r_k + \Delta y^r_{k+1} \\ \bar{y}^r_{k,k-N+2} \end{bmatrix} \tag{38}$$

The optimal control policy in terms of the measured data is now given by (39)

$$\begin{aligned} \Delta u_k &= \underset{\Delta u_k}{argmin}[(y_k - y^r_k)^T Q(y_k - y^r_k) + (u_{k-1}+\Delta u_k)^T R(u_{k-1}+\Delta u_k) + \gamma \bar{Z}^T_{k+1,k-N+2}\bar{P}\bar{Z}_{k+1,k-N+2}] \\ &= -[R + \gamma P_{11} + \gamma(\underline{G}^T_{k,11})P_{33}\underline{G}^T_{k,11} + \gamma P_{13}\underline{G}_{k,11} + \gamma(P_{13}\underline{G}_{k,11})^T]^{-1} \\ &\quad [[R + \gamma P_{11} + \gamma(\underline{G}_{k,11})^T P^T_{13}]u_{k-1} + \gamma[\underline{G}^T_{k,11}P_{33} + P_{13}]y_k \\ &\quad + \gamma[P_{12} + (\underline{G}_{k,11})^T P_{23}]\bar{u}_{k-1,k-N+1} + \gamma[P_{14} + (\underline{G}_{k,11})^T P_{34}]\bar{y}_{k,k-N+2} \\ &\quad + \gamma[(\underline{G}_{k,11})^T P_{33} + P_{13}](\underline{G}_{k,12}\Delta u_{k-1,k-N+1} + \underline{F}_k\bar{\Delta y}_{k,k-N+1})) \\ &\quad + \gamma[P_{15} + (\underline{G}_{k,11})^T P_{35}]y^r_k + \gamma[P_{15} + (\underline{G}_{k,11})^T P_{35}]\underline{F}^r_k\bar{\Delta y}^r_{k,k-N+1})) + \gamma[P_{16} + (\underline{G}_{k,11})^T P_{36}]\bar{y}^r_{k,k-N+2} \end{aligned} \tag{39}$$

The VI algorithm for iADP using output feedback is given by the Algorithm (2)

Algorithm 2 VI algorithm for iADP using output feedback[11]

Initialize a arbitrary control policy $\Delta u^0_k = \mu(\bar{Z}_{k,k-N+1})$
repeat
Value Update Step:
$\Delta u_k = -[R + \gamma P_{11} + \gamma(\underline{G}^T_{k,11})P_{33}\underline{G}^T_{k,11} + \gamma P_{13}\underline{G}_{k,11} + \gamma(P_{13}\underline{G}_{k,11})^T]^{-1}$
$[[R + \gamma P_{11} + \gamma(\underline{G}_{k,11})^T P^T_{13}]u_{k-1} + \gamma[\underline{G}^T_{k,11}P_{33} + P_{13}]y_k$
$+ \gamma[P_{12} + (\underline{G}_{k,11})^T P_{23}]\bar{u}_{k-1,k-N+1} + \gamma[P_{14} + (\underline{G}_{k,11})^T P_{34}]\bar{y}_{k,k-N+2}$
$+ \gamma[(\underline{G}_{k,11})^T P_{33} + P_{13}](\underline{G}_{k,12}\Delta u_{k-1,k-N+1} + \underline{F}_k\bar{\Delta y}_{k,k-N+1}))$
$+ \gamma[P_{15} + (\underline{G}_{k,11})^T P_{35}]y^r_k + \gamma[P_{15} + (\underline{G}_{k,11})^T P_{35}]\underline{F}^r_k\bar{\Delta y}^r_{k,k-N+1})) + \gamma[P_{16} + (\underline{G}_{k,11})^T P_{36}]\bar{y}^r_{k,k-N+2}$
until Convergence

B. Online Incremental Model Identification

The Incremental Model is identified in real time using Recursive Lease Squares (RLS) method assuming high sampling rate. The RLS is a recursive variant of Ordinary Least Squares (OLS) method which consists of simple matrix operations, whereas the OLS has a matrix inversion step[16]. Avoiding matrix inversion is ideal as the online model identification is done through some excitation signal and during phase of no excitation matrix, inversion might lead to numerical instability. The RLS method can also deal with time-varying systems and they demand small computational requirements which makes them suitable for online implementation. The derivation of Incremental model identification using RLS method with Full State measurements is adopted from[17].

1. Full State Measurements

For the implementation of iADP algorithm in section II.A.2, the augmented state transition matrix T_{k-1} and input distribution matrix G_{k-1} has to be identified online. The equation (15) can be segmented by row as follows:

$$\Delta x_{r,k+1} = \begin{bmatrix} \Delta x^T_k & \Delta u^T_k \end{bmatrix} \begin{bmatrix} f^T_{r,k-1} \\ g^T_{r,k-1} \end{bmatrix} \tag{40}$$

where $\Delta x_{r,k+1}$ is the r^{th} state increment yielding $f^T_{r,k-1}$ and $g^T_{r,k-1}$, the r^{th} row elements of F_{k-1} and G_{k-1} respectively. We can construct the parameter matrix Θ_{k-1} as follows

$$\Theta_{k-1} = \begin{bmatrix} F^T_{k-1} \\ G^T_{k-1} \end{bmatrix} \in \mathbb{R}^{(n+m)\times m} \tag{41}$$

The state prediction in terms of parameter matrix is:

$$\Delta\hat{x}^T_{k+1} = W^T_k \hat{\Theta}_{k-1}, \qquad W_k = \begin{bmatrix} \Delta x_k \\ \Delta u_k \end{bmatrix} \in \mathbb{R}^{(n+m)\times 1} \tag{42}$$

The parameter matrix is updated as follows:

$$\begin{aligned} \epsilon_k &= \Delta x^T_{k+1} - \Delta\hat{x}^T_{k+1} \\ \hat{\Theta}_k &= \hat{\Theta}_{k-1} + \frac{Cov_{k-1}W_k}{\gamma^{RLS} + W^T_k Cov_{k-1} W_k}\epsilon_k \\ Cov_k &= \frac{1}{\gamma^{RLS}}\left(Cov_{k-1} + \frac{Cov_{k-1}W_k W^T_k Cov_{k-1}}{\gamma^{RLS} + X^T_k Cov_{k-1} X_k}\right) \in \mathbb{R}^{(n+m)\times(n+m)} \end{aligned} \tag{43}$$

where ϵ_k is the innovation or state prediction error, Cov is the estimation Covariance matrix and $\gamma^{RLS} \in [0, 1]$ is the forgetting factor.

Similar procedure can be adopted for identifying reference dynamics to obtain F^r_{k-1}. Now the system transition matrix can be created for the augmented system as follows

$$T_{k-1} = \begin{bmatrix} F_{k-1} & 0 \\ 0 & F^r_{k-1} \end{bmatrix}$$

2. Output Feedback

The incremental model in input output data in equation (33) can also be constructed using RLS method. Here as the full state measurements are not available, the incremental model is constructed using incremental input output measurements over a time horizon. Equation (40) is now modified to include historical incremental data instead of state measurements as follows :

$$\Delta y_{r,k+1} = \begin{bmatrix} \bar{\Delta y}^T_{k,k-N+1} & \bar{\Delta u}^T_{k,k-N+1} \end{bmatrix} \begin{bmatrix} f^T_{r,k} \\ g^T_{r,k} \end{bmatrix}$$

Using the similar procedure mentioned above RLS can be used to estimate F_k, G_k, F^r_k.

III. Cessna Citation II PH-Lab Research Platform

The Cessna Citation II(Model 550) twin-jet business aircraft is a pressurized, low-wing monoplane that is certified for up to 10 persons including two pilots[18], is jointly operated by TU Delft and National Aerospace Laboratory (NLR). The aircraft has maximum operating altitude 13 km and maximum cruising speed of 710 km/h. The aircraft is modified as a airborne research platform(PH-lab) and flight tests are organized in cooperation with external partners like DLR, Oberpfaffenhofen to test the FCL developed by Aircraft System Dynamics Department. The aircraft has a mechanically linked Flight Control System (FCS), an autopilot system facilitated by FCC, a Flight Test Instrumentation System (FTIS) and an experimental Fly-By-Wire (FBW) system. The control system of the Cessna Citation II consists of cables that are connected to the control surfaces. The movements of the control surfaces are converted to electronic signals and the FCC determines these signals based on expected actuator response and provides them to the servo amplifiers of the actuators that deflect the control surfaces. The FTIS consists of a data acquisition computer and signal conditioning unit that can process information from sensors and can provide measurements at a high sample rate of upto 1000Hz that can be available for controllers[19]. Some of the sensor signals made available for FTIS are Attitude Heading and Reference System (AHRS), Digital Air Data Computer (DADC), air data boom, control surface synchros which measure

deflection angles of the control surfaces. Angle of attack is also available from a body mounted vane sensor. The FBW is developed based on the existing original autopilot system of the aircraft[20] which uses the position setpoint values from the FCL and feed back signals from servo which command the actuators. The overall PH-Lab aircraft diagram integrated with components useful for flight testing is shown in Fig. 2.

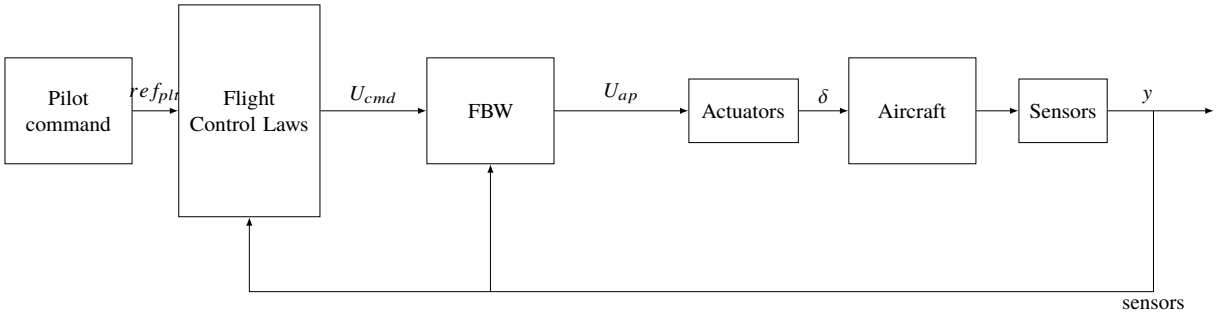


Fig. 2 Overview of PH-Lab integrated with Flight Control Laws and Fly-By-Wire system

Aircraft, sensor and actuator models that are available for testing the FCL's are discussed here.

A. DASMAT Aircraft Model

A simulation model for Cessna 500 aircraft was designed as a standard Flight CAD package referred to as DASMAT [21] and further improvements using this baseline model were made through flight tests on Citation II [22]. The baseline model is based on a generic nonlinear aircraft model with aerodynamic,propulsion and engine models. Models of external conditions like atmospheric wind and turbulence are also available which can be interfaced with the aircraft model. These models use the 6-DOF combined translational and rotational nonlinear equations of motion for the rigid body aircraft. The state vector is constructed using these 6-DOF equations and without the engine model the following 12 state variables and 8 Aerodynamic control inputs are available.

$$x = \begin{bmatrix} p & q & r & V_{tas} & \alpha & \beta & \phi & \theta & \psi & h_e & x_e & y_e \end{bmatrix}$$

$$u = \begin{bmatrix} \delta_e & \delta_a & \delta_r & \delta_{te} & \delta_{ta} & \delta_{tr} & \delta_f & lg_{sw} \end{bmatrix}$$

where δ_f denotes flap control surface, lg_{sw} denotes the landing gear. Further the model also provides observations which will be useful for control design viz,. aircraft states, their derivatives, accelerations, force and moment components from aerodynamic and propulsion models. An accurate mass model was developed and adopted to Citation II which provides aircraft mass, inertia and center of gravity position[4].

B. Sensor model

The sensors instrumentation model of the PH-Lab is identified using flight test data[4][23]. These are modelled taking into account the practical phenomenon like bias, noise, delays, resolution and sampling rate. The noise of sensors are modelled as Gaussian white noise with zero mean. The sensor characteristics available from different sensor systems are shown in the Table 1.

C. Actuator model

A high fidelity actuator model is developed and adopted for use within Citation II with better estimation along elevator and aileron channels[6]. As this model assumes smaller control system movements which cannot be guaranteed during the iADP controller training process, a low fidelity first order actuator model is chosen. This low fidelity actuator model is developed using flight test data to accommodate the dynamics of FBW[4] system. It is modelled as a first order system with a lag component, actuator deflection and rate saturation limits and transport delay as follows:

$$\dot{\delta}(t) = sat_{\dot{\delta}}\{\tau_{act}^{-1}\delta_{com}(t - \lambda_{act}) - \tau_{act}^{-1}sat_{\delta}[\delta(t)]\} \tag{44}$$

Table 1 PH-LAB Sensor characteristics [6]

Signal	Noise (σ^2)	Bias	Resolution	Delay[ms]	Sampling rate [Hz]
$p, q, r, \dot{\phi}, \dot{\theta}, \dot{\psi}[rad/s]$	4.0×10^{-7}	3.0×10^{-5}	6.8×10^{-7}	90	52
$\theta, \phi[rad]$	1.0×10^{-9}	4.0×10^{-3}	9.6×10^{-7}	90	52
$A_x, A_y, A_z[g]$	1.5×10^{-5}	2.5×10^{-3}	1.2×10^{-4}	117	52
$V_{TAS}, V_{CAS}[m/s]$	8.5×10^{-4}	2.5	3.2×10^{-2}	300	16,8
$\delta_a, \delta_e, \delta_r[rad]$	5.5×10^{-7}	2.4×10^{-3}	–	0	100
$\alpha_{boom}, \beta_{boom}[rad]$	7.5×10^{-8}	1.8×10^{-3}	9.6×10^{-5}	100	100
$\alpha_{body}[rad]$	4.0×10^{-10}	–	1.0×10^{-5}	280	1000

Where sat_δ and $sat_{\dot{\delta}}$ represent the saturation function for actuator deflection and rate respectively. τ_{act} is the time lag component identified using a step input response and the delay between FBW and the control surface deflection is modelled as the transport delay λ_{act}. The actuator model characteristics are listed in the Table 2.

Table 2 PH-LAB Actuator characteristics [4]

	$\delta_{max}[^\circ]$	$\delta_{min}[^\circ]$	$\dot{\delta}_{max}[^\circ/s]$	$\lambda_{act}[ms]$	$\tau_{act}[ms]$
Aileron	15	-19			
Elevator	15	-17	19.7	39.8	84
Rudder	22	-22			

IV. iADP Control Law Design

This section presents the control law design for integrating iADP controller within the FCL's of Citation II aircraft. iADP controller is used for automatic control of inner loop while the outer loop is based upon the previously designed manual control[4] laws. The output from the slow outer loop is used as the reference signal to be tracked by the faster inner loop. The outer loop contains Command and reference model and a side-slip controller. The command module ensures safety through attitude flight envelope protection. The reference model is a second order model which converts the commanded signals from pilot into values achievable by aircraft. Smooth reference signals are necessary for iADP controller to ensure any sharp increase on cost function which might result in numerical instability in updating the kernel matrix P. A coordinated flight is desirable due to the absence of FBW for yaw channel. Thus, an outer loop controller for yaw channel is designed which generates reference for yaw rate such that any side slip angle is rejected. Standard flight maneuvers like 3211 for pitch tracking, bank to bank maneuvers for roll are simulated and are provided as input pilot commands to the outer loop.

A coupled longitudinal and lateral rate iADP controller is designed for the automatic inner loop which can learn to control all the available control surfaces at the same time. The advantage of having a combined longitudinal and lateral control is that it can learn control parameters without neglecting any coupling effects.

A. iADP-FS Online Rate Control Design

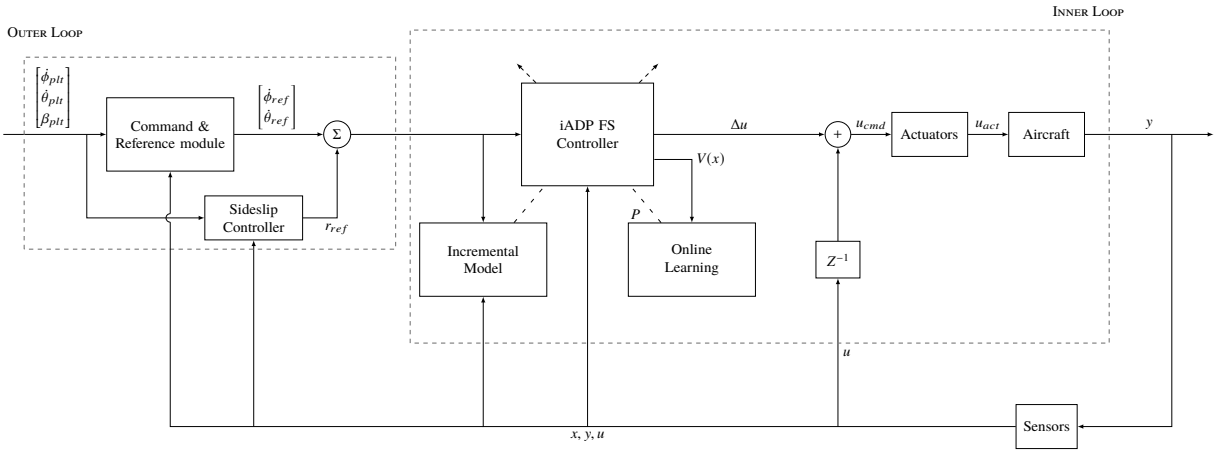


Fig. 3 Controller architecture with online training using Full State Feedback for combined longitudinal and lateral control of Cessna Citation II

The control system architecture with integrated iADP controller using full state feedback is shown in Fig. 3. iADP is used for rate control design as the rate control has least learning complexity. This is due to the fact that the effect of control surfaces input on aircraft angular rates are faster as compared to that of control surface effect on angle of attack, due to time scale principle. However in theory iADP should be able to achieve control involving slower dynamic variables even as it does not assume time scale separation[17]. Assuming actuators are modelled as first order systems the order of learning complexity according to time scale separation is as follows:

$$p, q, r < \phi, \theta, \psi, \alpha, \beta < V, \gamma, \chi \tag{45}$$

The states, control input, output and reference vectors used by the iADP controller are

$$x = \begin{bmatrix} p & q & r & \alpha & \beta & \phi & \theta \end{bmatrix}^T, u = \begin{bmatrix} \delta_a & \delta_e & \delta_r \end{bmatrix}^T$$

$$y = \begin{bmatrix} \dot{\phi} & \dot{\theta} & r \end{bmatrix}^T, y_r = \begin{bmatrix} \dot{\phi}_{ref} & \dot{\theta}_{ref} & r_{ref} \end{bmatrix}^T$$

The air speed information is not included in the state vector due to slower local variations in the airspeed variable which might effect the incremental model identification. The previous implementations are designed using a air speed controller, however this is not possible on Cessna citation because of lack of auto throttle functionality. The effect of variations in air speed on the controller are mitigated by choosing the reference commands to track such that variations in air speed are restricted to smaller values. Also as the controller is implemented with online learning capability this might reduce effects of air speed variations further.

The incremental model for the system is provided with state information, actuator position measurements(x, u). The incremental model for the reference dynamics is identified using the reference signal from the outer loop. Online incremental model for the system as well as the reference dynamics is identified online using RLS approach and the identified model coefficients are provided to the iADP controller. The iADP controller calculates the control increments using the model information from the incremental model and the measurements from the system. For online controller adaptation the kernel matrix P is updated at every time step using a Least Squares method with the data collected along the system trajectory. Recalling equation (20) we can write:

$$\begin{aligned} X_k^T P X_k &= (y_k - y_k^r)^T Q (y_k - y_k^r) + u_k^T R u_k + \gamma X_{k+1}^T P X_{k+1} \\ X_k^T P X_k &= V(X_k) \\ (X_k \otimes X_k)^T \vec{P} &= V(X_k) \\ X^{kr} \vec{P} &= V(X_k) \\ \vec{P} &= X^{kr+}.V(X_k) \end{aligned} \tag{46}$$

where $X^{kr} = (X_k \otimes X_k)^T$ is the Kronecker product, $\vec{P}$ is the kernel matrix reorganized as a vector, X^{kr+} is the pseudo inverse of X^{kr}. The online learning block stores the cost function estimate $V(X)$ and the Kronecker product($X^{kr} = (X_K \otimes X_k)^T \in \mathbb{R}^{(1\times(n+l)^2)}$) of augmented state vector X from the iADP controller over a certain time window $t_{ol} = N_{ol} \times f$, where N_{ol} are the number of samples collected during this window and f is the frequency of simulation. The collected cost estimate and state vector are stacked as follows:

$$\begin{aligned} \bar{V}(x) &= \begin{bmatrix} V(X_k) & \dots & V(X_{k-N_{ol}}) \end{bmatrix}^T \in \mathbb{R}^{(N_{ol}\times 1)} \\ \bar{X}^{kr} &= \begin{bmatrix} X_k^{kr} & \dots & X_{k-N_{ol}}^{kr} \end{bmatrix}^T \in \mathbb{R}^{(N_{ol}\times(n+l)^2)} \end{aligned} \tag{47}$$

Now the kernel matrix can be updated recursively using the data observed over this window as :

$$\vec{P} = \bar{X}^{kr+}.\bar{V}(X_k) \tag{48}$$

It is assumed for the iADP combined control design, clean measurements from sensors are available, thus effects of sensor dynamics like noise,delays, bias and quantization are neglected.

B. iADP-FS Online Longitudinal Rate Control Design

A simple longitudinal control design is considered to analyze the effects of real world phenomenon on the controller performance as this design needs only limited sensor measurements of states/outputs and actuators to study the individual effects of different phenomenon. The control architecture is as shown in Fig. 4 where only variables related to longitudinal rate control are considered. The manual outer loop provides the reference signal to be tracked to the automatic inner rate control loop. The iADP controller is used for the inner rate control with full state feedback. The states, outputs, reference and control vectors used by the iADP controller are

$$x = \begin{bmatrix} q & \alpha \end{bmatrix}^T, u = \delta_e, y = \dot{\theta}, y_r = \dot{\theta}_{ref}$$

.

Further as the primary aim of this analysis is to study the influence of real world phenomenon on controller performance, effects of sensor dynamics like bias, noise, delays, transport delay and quantization effects are considered. Transport delay for citation model is added in the control channel. To mitigate the effect of sensor noise, processing of noisy signals is done through signal filtering as shown in Fig. 4 and filtered signals ($\hat{x}, \hat{y}, \hat{u}$) are used by the iADP controller and also for the incremental model identification. Similar to the combined control approach, the online learning is achieved by updating the kernel matrix P at every time step using the data within a window.

C. iADP-OPFB Longitudinal Rate Control Design

Similar to previous section a simple longitudinal control design is considered to evaluate the output feedback algorithm. The control architecture is as shown in Fig. 5 where it is assumed that the full state feedback is not available and the task of the iADP is to achieve longitudinal rate control using only output measurements over a time horizon. As the effects of sensor dynamics are considered, the noisy sensor measurements are processed through signal filtering. Due to higher learning complexity of the algorithm the controller is trained offline to arrive at a baseline controller P which is used to evaluate the controller. The offline training is done by certain number of episodes where the kernel matrix is updated at the end of every episode. Every episode is initialized with kernel matrix P that is carried on from the previous episode and the aircraft is reset to steady wing level flight condition at the beginning of episode. The incremental model identifies the model coefficients necessary predict the next incremental output. The output,reference and control vectors used by the iADP controller are

$$y = \dot{\theta}, y_r = \dot{\theta}_{ref}, u = \delta_e$$

D. Signal filtering

The iADP controller is a model free controller which is based only on the measurements that are obtained along the system trajectory. For Full State Feedback controller as shown in Fig. 3 the controller needs full state measurement

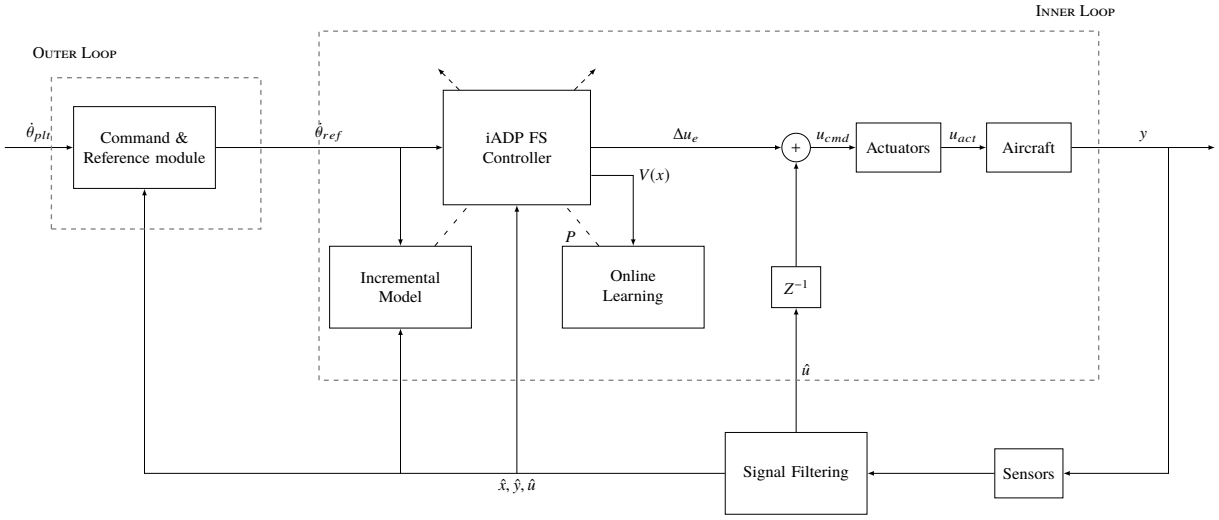


Fig. 4 Controller architecture with online learning using Full State Feedback for longitudinal control of Cessna Citation II

and the actuator deflection measurements. State information is made available through various sensors like AHRS for attitude and angular rates while angle of attack and side slip angle are measured using vane type sensors. Control surface deflection measurements are available for Citation-II aircraft which are measured using synchros. This alleviates the need for an actuator model to estimate the actuator position.

To mitigate the effects of noisy sensor measurements appropriate signal filtering is required. All the signals are filtered using a first order low-pass filter (49) unless specified with a cut off frequency of $\omega_n = 20 rad/s$.

$$H(s) = \frac{\omega_n}{s + \omega_n} \tag{49}$$

E. Implementation issues

The following section provides discussion on some of the implementation issues related to the iADP Controller viz, Incremental model identification, Persistent excitation and parameter tuning.

1. Online Incremental Model Identification

The iADP controller performance is dependent on good identification of the incremental model parameters. As the controller do not have any prior knowledge of the model, procedures similar to online system identification have to be adopted for incremental model identification. Online system identification typically involves exciting the aircraft through specific control inputs along different channels. Some of the common flight maneuvers used for system identification are listed in [16] and some of these maneuvers and necessary control input parameters are adopted here.

(a) For estimation of parameters related to longitudinal motion a short period motion is selected as this motion can provide most information for parameter estimation related to vertical and pitching motion. A multi step input like 3211, which consists of alternative positive and negative steps with relative duration of 3,2,1,1 respectively is chosen . The duration of the time step can be tuned to excite mode of interest.

(b) For lateral motion parameter estimation, banking roll maneuver and dutch roll motion are chosen. The banking maneuver is achieved through multi step aileron pulses while the dutch roll is excited through a doublet input.

The control inputs are chosen such that the aircraft can come back to steady state condition and are skewed in time across the channels such that different dynamic motions across different axes are excited. Another advantage of these control inputs is that the pilot can easily provide these inputs during flight avoiding automated control input generation.

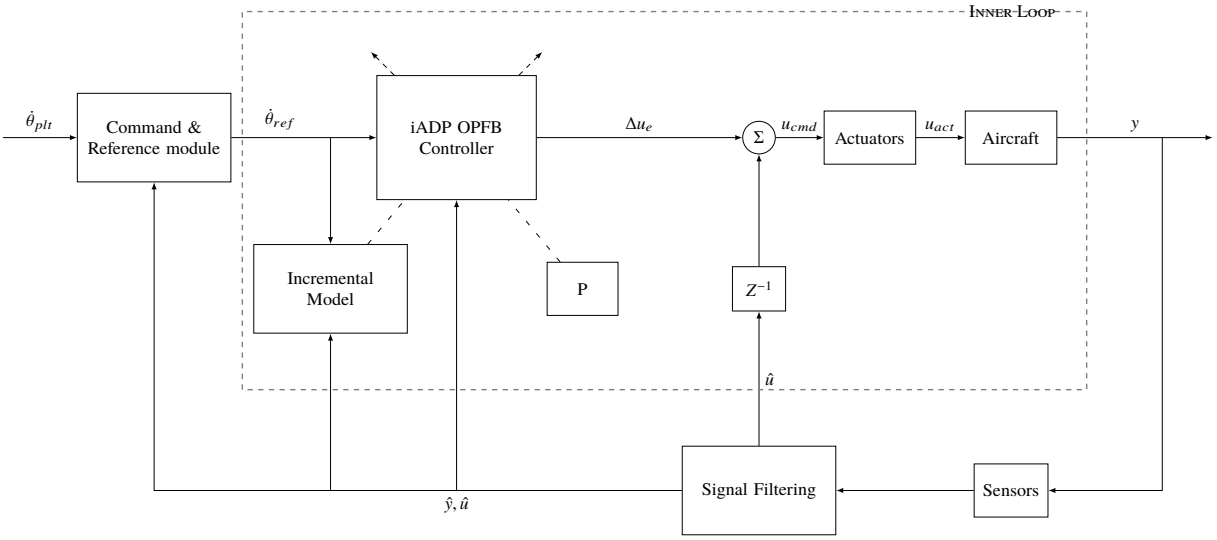


Fig. 5 iADP controller architecture with offline learning using Output Feedback for longitudinal control of Cessna Citation

2. Persistent Excitation

The need for Persistent excitation is two fold. One for online system identification such that all the modes of the system are excited continuously so that the model parameters can be updated. Second it serves for state space exploration which is a necessary condition for RL algorithm to arrive at good policies that can minimize the cost function. Although the input signals mentioned above are useful in identifying the initial model parameters, such signals cannot be implemented continuously. To aid continuous learning for model identification and controller state space exploration, white noise is added to the control input along with the incremental control input across the three channels. Thus the control input becomes

$$u_k = u_{k-1} + \Delta u_k + PE$$

where PE is white noise. During initial mode excitation phase, PE is white noise combined with control input used for model identification mentioned in previous section. The reference tracking problem includes a feedback loop from output y to u. For systems involving feedback loop it is also necessary that the reference signal is also persistently exciting to estimate the model parameters[24].

3. Parameter tuning

For optimal control performance parameter tuning is essential. For online incremental model identification the following hyperparameters are involved viz,. forgetting factor γ_{RLS}, initial covariance matrix Cov_0 and the initial parameter matrix Θ_0. For the *iadp* controller the following hyperparameters are involved viz,. forgetting factor γ, weighting matrices Q and R, initial kernel matrix P_0.

The RLS algorithm for model identification uses forgetting factor γ_{RLS} which provides control over the importance of data and its recency. For time-varying models γ can be chosen to be a value less than 1 to rely on the most recent data. However this makes the parameter updates more sensitive to the noise in the data. This is important if the real time data is obtained from noisy sensors. Typical range of values for the forgetting factor are $0.95 \leq \gamma^{RLS} \leq 1$[24]. The Covariance matrix(Cov) is a measure of confidence in the parameter matrix Θ. It is generally initialized as an identity matrix scaled by a factor. During the phases of poor excitation, covariance matrix parameters might grow exponentially leading to covariance wind up[25] causing numerical instability errors. The initial parameter matrix(Θ_0) contain the control effectiveness matrix and dynamic model information. These are typically initialized as zero matrices.

The hyperparameters of the iADP controller directly influence the controller performance. The discount factor $gamma \in [0, 1]$ is a measure of importance of cost information from future states. Typically, γ is initialized to value smaller than 1 to contain the infinite horizon cost to a finite value. Another advantage of discount factor is that it can reduce the bias effects that is introduced by the white noise during persistent excitation and the effects of improper

initial conditions[12]. The weighting matrices Q and R provide a trade off between the tracking performance and control input energy[11]. The choice of Q and R will also effect the stability and robustness of the controller and hence should be initialized to appropriate values. Initialization of Q and R is done through trial and error such that satisfactory performance can be achieved. Fine tuning of Q and R might be essential for systems involving multiple inputs and outputs. In such scenarios fine tuning of parameters can be achieved through optimization techniques like Multi Objective Parameter Synthesis (MOPS)[26] such that certain control design requirements like overshoot, settling time, rise time and tracking error are met. However tuning control parameters to meet control design requirement is beyond the scope of this paper. Finally the kernel matrix is typically initialized as a identity matrix scaled by a small factor.

V. Control Law Evaluation

This section presents the results of the iADP controller implementation on Cessna Citation II aircraft. The results are simulated in MatLab/Simulink environment using a Cessna Citation model that simulates the data required for the controller evaluation. The simulations are performed at 100 Hz sampling frequency using Heun's second order fixed step solver. As data is sampled from sensors at fixed time steps in real time applications, using a fixed time solver is ideal to evaluate the controller learning performance. Unless specified the simulations are started from a steady straight and level trim flight condition which is a wings-level constant flight path condition. The trimming conditions of the aircraft at the start of the simulation are provided in Tables 3 and 4. PLA stands for power lever angle and an equal constant throttle power is provided to left and right engines.

Table 3 State Trim conditions

State	$p[^\circ/s]$	$q[^\circ/s]$	$r[^\circ/s]$	$V_{tas}[m/s]$	$\alpha[^\circ]$	$\beta[^\circ]$	$\phi[^\circ]$	$\theta[^\circ]$	$\psi[^\circ]$	$h_e[m]$	$x_e[m]$	$y_e[m]$
Value	0	0	0	90	3.76	0	0	3.76	0	2000	0	0

Table 4 Actuator Trim conditions

Actuator	$\delta_a[^\circ]$	$\delta_e[^\circ]$	$\delta_r[^\circ]$	$PLA_{1,2}$
Value	0	-1.727	0	0.6335

A. iADP-FS Online Rate Control

The simulation results for combined Online rate control design using iADP full state feedback controller are presented in this section. The control law is designed based on the procedure discussed in section IV.A and clean sensor measurements are assumed. The hyperparameters are tuned according the principles mentioned before. The parameters for incremental model identification and iADP controller used in this simulation are presented in Tables 5 and 6 respectively. The forgetting factor γ_{RLS} is chosen to be one to provide better convergence of model parameters. The weighting matrices are manually tuned such that a satisfactory controller performance can be achieved. Typically values of weighting matrices R are fixed and Q are varied to achieve satisfactory performance. Higher weightage is given to the yaw rate control to maintain zero side slip to avoid adverse yaw. The task of the controller is two fold. Firstly an incremental model of the aircraft has to be identified online ensuring that the aircraft retains the steady state flying condition. Secondly aircraft should learn the control parameters online and perform a defined flight maneuver and achieve satisfactory tracking performance. The flight maneuver involves a combined longitudinal and lateral motion, where the aircraft performs a bank to bank roll maneuver and tracking a 3211 reference in the longitudinal direction, while ensuring coordinated turn using rudder. The necessary pilot commands to achieve this flight maneuver are simulated which are then fed to the manual control loop. The manual control loop provides the set points of the reference signals to be tracked for the automatic inner loop by ensuring that the aircraft stays within safe flight envelope and the reference commands are achievable by the inner loop.

Figures 6 and 7 shows the time responses and control inputs(u_{act}) acting on the aircraft. During the first 30 seconds, control inputs are generated such that online incremental model can be identified. Firstly, a 3211 input is commanded by elevator to excite short period dynamics. After the aircraft has reached steady state then the lateral dynamics are excited

Table 5 Parameters for Incremental Model Identification

Parameter	γ^{RLS}	Cov_0	Θ_0
Value	1	$1000I$	0

Table 6 Controller tuning parameters

Parameter	$diag(Q_p, Q_q, Q_r)$	$diag(R_{\delta_a}, R_{\delta_e}, R_{\delta_r})$	γ	P_0
Value	$diag(80, 100, 200)$	$diag(1, 1, 0.25)$	0.4	$0.001I$

through a banking motion by commanding a pulsed input through aileron and dutch roll motion is excited through doublet by rudder. As the reference model also needs to be identified online reference signals are provided during this period. The controller is then activated at t = 30 seconds and the parameters of the kernel matrix are updated online from there after. The iADP controller needs data over a window period of t_{ol} before the parameters of the kernel matrix are updated. A window of 20 seconds is chosen in this case where the data is collected during this window viz,. $\bar{V}(x)$, $\bar{X}^{kr}$ and used to update the kernel matrix P at every time step according to equation (48). From the time responses we can see that the controller is able learn the policy online using data collected from just 2000 samples in the 20 second window period and is able to track the reference signals with satisfactory performance. The small oscillations observed at the control surface is due to the persistent excitation which is required for continuous online learning. Also the small peaks at control surfaces are visible at t = 30 seconds due to the kernel matrix update however the deflections are found to be within the actuator limits. High control activity in the rudder can be seen which is due to less weight($R_{\delta r}$) given to rudder. Observing the pitch attitude rate tracking response, we can see that the controller is able to adapt to time varying reference signals. Higher aileron control activity can be seen whenever there is a non zero pitch rate command, implying the controller is able to learn the coupling effects as the designed controller does not assume a decoupled controller for longitudinal and lateral dynamics.

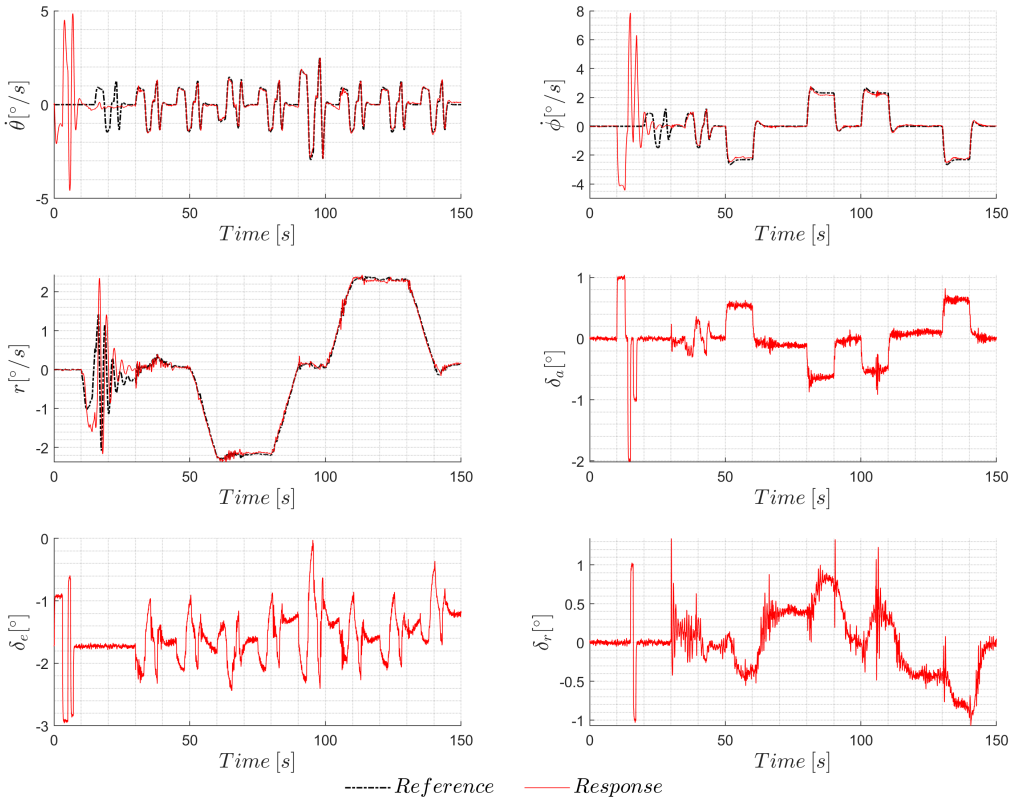


Fig. 6 Time responses of the Cessna Citation with combined iADP Online rate control using Full State Feedback

From Fig. 7 we can see that the aircraft is able to perform bank to bank(±20[°]) manoeuvre while restricting the side slip angle to value ±0.5[°]. Safety critical parameters like α and load factor n_z are found to be within safety limits throughout the flight maneuver including the initial model identification phase. Good longitudinal tracking response is seen even when the velocity conditions are changing which might be due to the online learning aided with persistent excitation. However for large changes in velocity we might have to excite the dynamic modes of the aircraft again if any performance degradation is observed.

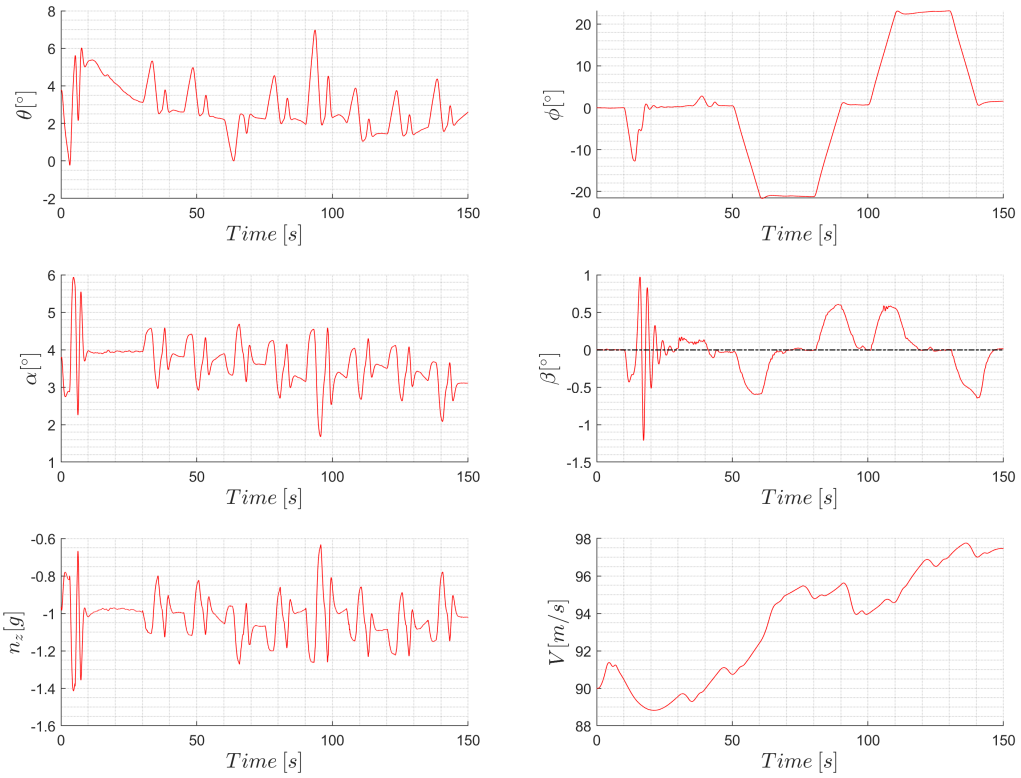


Fig. 7 Time responses of the Cessna Citation with combined iADP Online rate control using Full State Feedback

Fig. 8 shows the evolution of incremental model parameters viz,. state transition matrix F_t, control effectiveness matrix G_t and diagonal values of the kernel matrix parameters. The incremental model parameters have converged after the initial model identification phase of 30 seconds. We can see the effect of different modes of aircraft being excited at different times from the diagram. The kernel matrix parameters have converged within a short time after activating the kernel matrix at 30 seconds. However some of the parameters are fluctuating during the flight maneuver which might be because the controller is finding the need to update its policy for changing flight conditions and the coupling effects not encountered before.

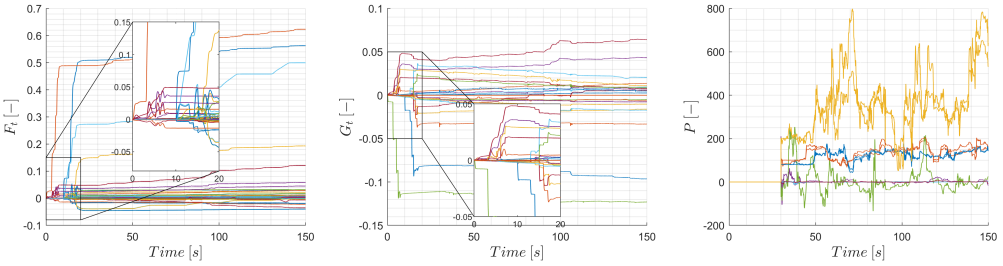


Fig. 8 Evolution of model coefficients and kernel matrix parameters during online learning

B. iADP-FS Online Longitudinal Rate Control

To assess the effect of sensor dynamics a simple longitudinal control task is considered. The time response and the control input is shown in Fig. 9. First 25 seconds is the model identification phase through short period dynamics excitation and the controller is activated after 25 seconds. The evolution of model coefficients and kernel matrix parameters are shown in Fig. 10. The model coefficients have converged after 10 seconds and for the remainder of this section the converged model coefficients will be considered as a measure to evaluate the model identification. Controller performance is assessed by considering three metrics viz, Root Mean Square Error (RMSE) between reference and actual pitch attitude rate, max absolute elevator deflection angle($max(\delta_e)$) and max elevator deflection rate($max(\dot{\delta_e})$). The rate saturation limit of the elevator is 20 deg/s. To minimize the transient effects of controller learning process, these metrics are evaluated between 40 - 80 seconds period.

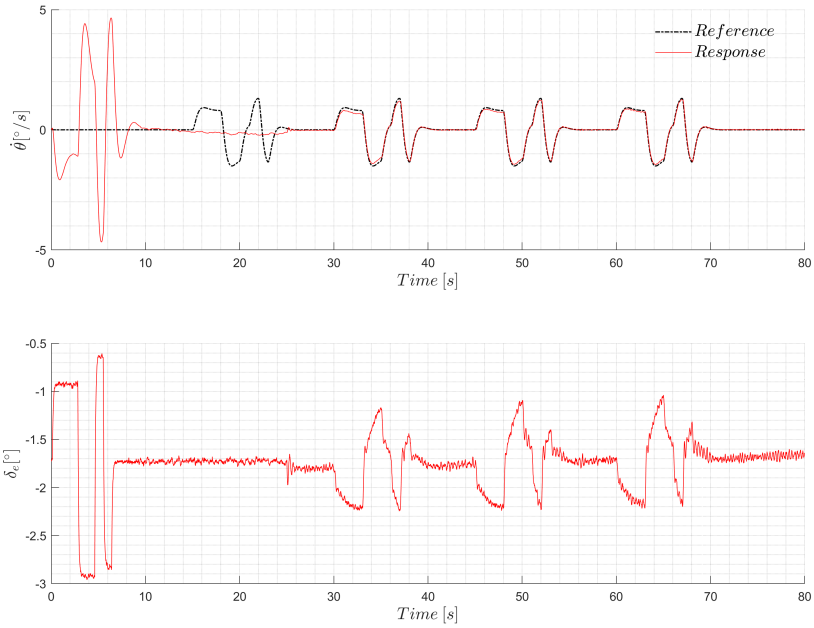


Fig. 9 Time responses of the Cessna Citation with longitudinal iADP Online rate control using Full State Feedback

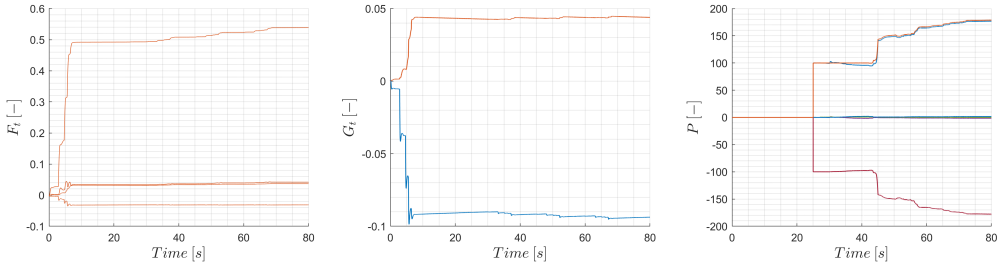


Fig. 10 Evolution of model coefficients and kernel matrix parameters during online learning

1. Selection of Weighting matrices

Before studying the effect of sensor dynamics a robustness analysis of controller is performed to assess the stability of the controller against change in weighting parameters. Fig. 11 shows the variation of the controller performance

metrics against the weighting matrices. It is clear that the tracking error is minimum for higher values of Q and lower values of R. However the tracking error seems to increase after a certain limit indicating oscillatory behaviour after certain threshold. This is confirmed from the elevator deflection rate graph as the actuator rate saturation limits are found to have reached beyond this threshold. For constant values of Q an increase in R will reduce the control activity but will increase the RMSE. For constant values of R an increase in Q results in less RMSE but it is also increasing the control activity. To ensure stability, lower values of Q and higher values of R are preferred.

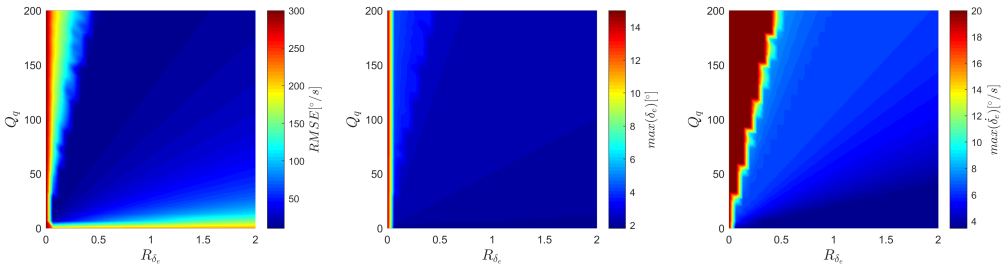


Fig. 11 Effect of weighting matrices on tracking performance and control activity

2. Real world phenomenon investigation

Before studying the effects of sensor dynamics the weighting matrices are readjusted to Q = 20 and R = 1 so that influence of weighting matrices is minimized during this analysis. The effect of sensor dynamics on the performance metric namely RMSE, actuator maximum deflection and rate and converged model parameters are evaluated which is listed in Table 7.

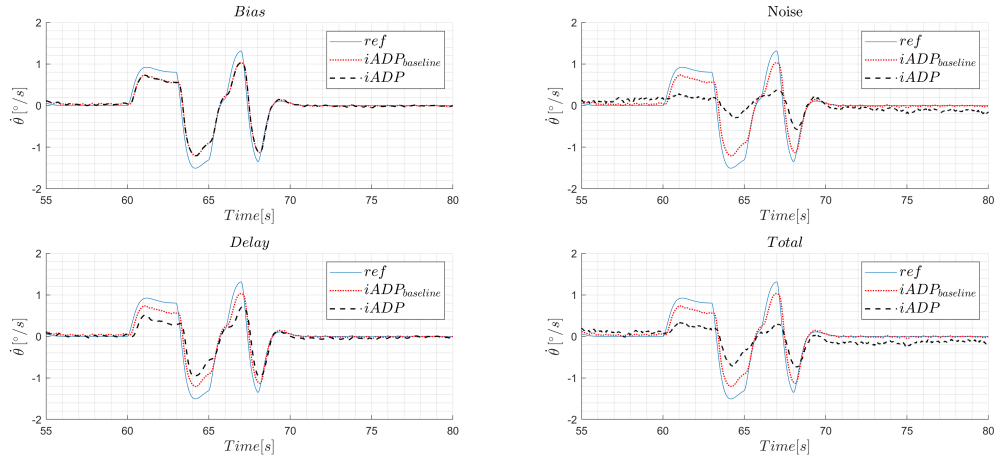


Fig. 12 Effect of sensor dynamics on tracking performance

For a comparison a baseline controller performance is included where sensor dynamics are neglected. Each phenomenon is considered separately based on one factor at a time method. Finally controller performance is evaluated considered combined phenomenon referenced as Total in the table. Comparing with baseline performance, we can see that the controller performance is not effected by discretization. The model parameters are identified without any difference. The effect of sensor bias on controller performance is also minimal, however small changes in the model parameters in the presence of bias is observed. Transport delay has also minimal effect on the controller performance. Noise and delays are found to degrade the controller performance and RLS algorithm is unable to identify the model parameters in the presence of noise/delays. These effects can be visualized from Fig. 12.

Note that the baseline performance is not optimal as the weighting matrices are retuned such that the controller performance is stable. Noise has degraded controller performance considerably. In the presence of delays, although the

Table 7 Effect of Sensor dynamics on Controller performance and Model Identification

	$RMSE[^\circ/s]$	$max(\delta_e)[^\circ]$	$max(\dot{\delta_e})[^\circ/s]$	$Ft[-]$	$Gt[-]$
Baseline	57.45	2.11	3.58	[0.492, -0.032, 0.034, 0.032]	[-0.092, 0.044]
Discretization	57.27	2.11	3.60	[0.492, -0.032, 0.034, 0.032]	[-0.092, 0.044]
Bias	58.73	2.09	3.55	[0.492, -0.034, 0.034, 0.035]	[-0.089, 0.045]
Noise	193.75	1.85	4.47	[0.410, -0.006, 0.016, -0.015]	[-0.046, -0.064]
Sensor Delay	110.82	2.04	3.46	[0.501, -0.011, 0.026, 0.032]	[-0.035, 0.044]
Transport Delay	55.21	2.11	3.50	[0.492, -0.032, 0.034, 0.032]	[-0.093, 0.045]
Total	170.70	1.90	2.40	[0.409, -0.010, 0.017, -0.017]	[-0.049, -0.063]

controller performance has degraded, it is observed that the controller is still trying to track the reference signal but with higher tracking error. Oscillatory behaviour can also be observed.

Table 8 Effect of Sensor dynamics on Controller performance and Model Identification with filtering

	$RMSE[^\circ/s]$	$max(\delta_e)[^\circ]$	$max(\dot{\delta_e})[^\circ/s]$	$Ft[-]$	$Gt[-]$
Baseline	57.53	2.11	4.16	[0.481, -0.030, 0.034, 0.032]	[-0.092, 0.044]
Discretization	57.51	2.11	4.15	[0.481, -0.030, 0.034, 0.032]	[-0.092, 0.044]
Bias	59.19	2.09	4.08	[0.481, -0.032, 0.035, 0.035]	[-0.088, 0.045]
Noise	61.49	2.10	4.03	[0.480, -0.026, 0.034, 0.032]	[-0.091, 0.042]
Sensor Delay	94.77	2.12	3.73	[0.488, -0.014, 0.027, 0.033]	[-0.046, 0.047]
Transport Delay	57.87	2.11	4.05	[0.481, -0.030, 0.034, 0.032]	[-0.092, 0.044]
Total	101.30	2.09	2.94	[0.487, -0.011, 0.027, 0.036]	[-0.045, 0.046]

To mitigate the effect of noise, signals are filtered using a first order low pass filter. Table 8 lists the effect of filtering on the controller performance. Using filtering, the controller performance degradation has been reduced considerably and RLS is able to learn the model parameters with improved accuracy. The improvement in the controller performance with filtering is shown in the time response plots in Fig. 13.

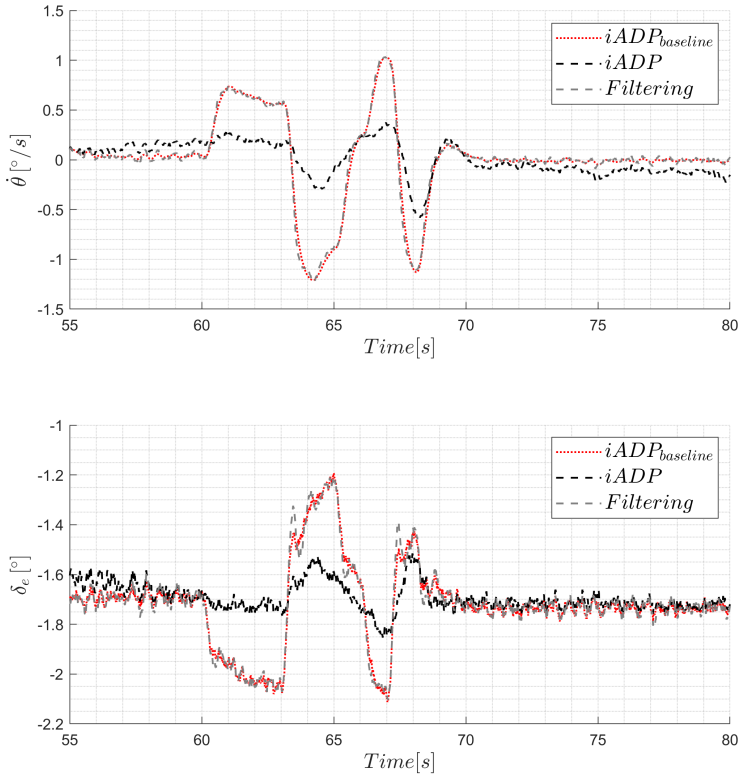


Fig. 13 Effect of Filtering on tracking performance

C. iADP-OPFB Longitudinal Rate Control

This section presents the results of iADP-OPFB controller evaluation to achieve longitudinal rate control of Citation II. As mentioned before, full state information is not provided to the controller and the controller has to achieve longitudinal tracking by using the input output measurements over a certain time horizon. Recalling the methodology from Section II.A.3, N=2 samples is used to construct the state. The weighting matrices($Q = 10, R = 1$) are tuned to achieve satisfactory performances. The results are summarized in Fig. 5. The model is identified using a 3211 maneuver in every episode and the objective of the controller is to track the 3211 signals commanded by the pilot. The evolution of kernel matrix parameters over episodes is shown in Fig. 15. The parameters have converged after 13 iterations. The controller performance using the converged kernel matrix parameters for tracking is shown in Fig. 14. The controller is able to track the reference in the presence of noisy signal measurements and time delays when state information is not provided to the controller. Small steady state errors are visible which might due to the sensor bias and as the full state information is not available for this controller the effects of bias is higher compared to the controller where full state feedback is available.

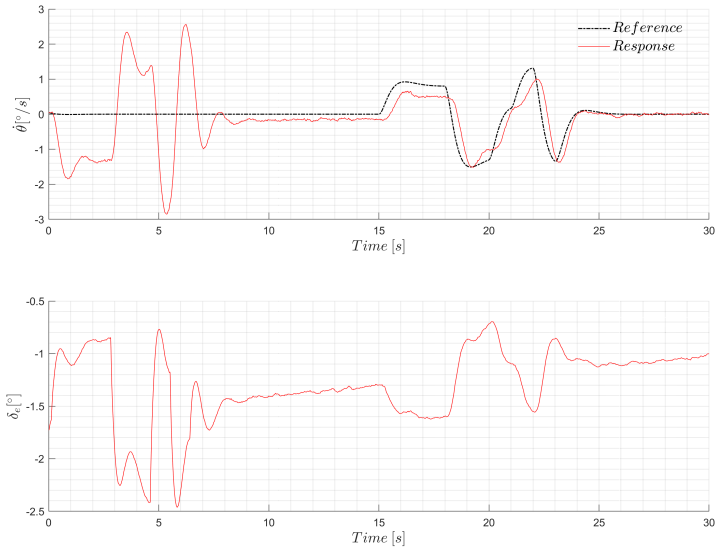


Fig. 14 Time responses of Cessna Citation with longitudinal iADP rate control using Output Feedback

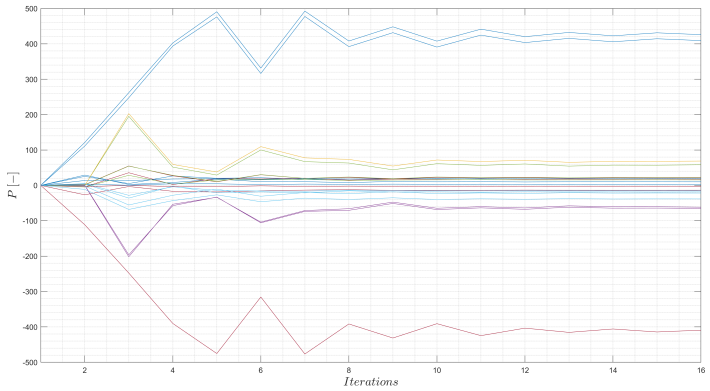


Fig. 15 Evolution of kernel matrix parameters of iADP rate control using Output Feedback

VI. Conclusion

This paper presents design of Incremental Approximate Dynamic Programming based flight control law for Cessna Citation II aircraft. The iADP controller is designed to achieve automatic online rate control to track pilot commands via setpoints provided by the manual outer loop. The incremental model necessary for the iADP controller is identified online through appropriate system excitation using control surfaces. The simulation results show that the combined rate control using full state feedback is able to learn online to control the aircraft without any knowledge of the system but just using the data collected along the system trajectories. To assess the controller performance in the presence of sensor dynamics and actuator dynamics, an analysis is carried out to identify causes of any performance degradation. The simulation results from iADP longitudinal control design using full state feedback indicate that the discretization of sensor signals, sensor bias and transport delays did not have any significant effect on the controller performance or on the incremental model identification while noisy signals and delays in sensors are found to effect the controller performance. The noisy signals resulted in incorrect estimation of incremental model parameters affecting the controller performance.

It is observed that appropriate filtering of signals resulted in better estimation of the incremental model subsequently improving the controller performance. Sensor delays also resulted in incorrect estimation of model parameters, however the controller is found to track the reference inspite of the incorrect incremental model parameters but with reduced tracking performance. Finally an iADP controller using output feedback is designed to achieve longitudinal control in the absence of full state information. The controller is trained offline due to higher learning complexity and performance is evaluated in the presence of sensor and actuator dynamics. The results from output feedback method show the controller can achieve satisfactory tracking control but with reduced tracking performance.

For a successful implementation of iADP controller on Cessna Citation II aircraft, further research needs to be conducted to validate this controller on a real system through flight tests. The effect of sensor delays on controller performance should be investigated in future by conducting stability and robustness analysis as they are found to degrade the controller performance.

References

[1] Belcastro, C. M., Foster, J. V., Newman, R. L., Groff, L., Crider, D. A., and Klyde, D. H., "Aircraft Loss of Control: Problem Analysis for the Development and Validation of Technology Solutions," *AIAA Guidance, Navigation, and Control Conference*, 2016. doi:10.2514/6.2016-0092, URL https://arc.aiaa.org/doi/abs/10.2514/6.2016-0092.

[2] Sieberling, S., Chu, Q. P., and Mulder, J. A., "Robust Flight Control Using Incremental Nonlinear Dynamic Inversion and Angular Acceleration Prediction," *Journal of Guidance, Control, and Dynamics*, Vol. 33, No. 6, 2010, pp. 1732–1742. doi:10.2514/1.49978, URL https://doi.org/10.2514/1.49978.

[3] Acquatella, P., Van Kampen, E.-J., and Chu, Q., "Incremental backstepping for robust nonlinear flight control," 2013.

[4] Grondman, F., Looye, G., Kuchar, R. O., Chu, Q. P., and Kampen, E.-J. V., "Design and Flight Testing of Incremental Nonlinear Dynamic Inversion-based Control Laws for a Passenger Aircraft," *2018 AIAA Guidance, Navigation, and Control Conference*, 2018. doi:10.2514/6.2018-0385, URL https://arc.aiaa.org/doi/abs/10.2514/6.2018-0385.

[5] Keijzer, T., Looye, G., Chu, Q. P., and Kampen, E.-J. V., "Design and Flight Testing of Incremental Backstepping based Control Laws with Angular Accelerometer Feedback," *AIAA Scitech 2019 Forum*, 2019. doi:10.2514/6.2019-0129, URL https://arc.aiaa.org/doi/abs/10.2514/6.2019-0129.

[6] Pollack, T., Looye, G., and Linden, F., "Design and flight testing of flight control laws integrating incremental nonlinear dynamic inversion and servo current control," 2019. doi:10.2514/6.2019-0130.

[7] Bertsekas, D., *Dynamic Programming and Optimal Control*, No. Bd. 2 in Athena Scientific optimization and computation series, Athena Scientific, 2012. URL https://books.google.de/books?id=H-PSMwEACAAJ.

[8] Lewis, F., and Vrabie, D., "Reinforcement Learning and Adaptive Dynamic Programming for Feedback Control," *Circuits and Systems Magazine, IEEE*, Vol. 9, 2009, pp. 32 – 50. doi:10.1109/MCAS.2009.933854.

[9] Lewis, F. L., and Vamvoudakis, K. G., "Reinforcement Learning for Partially Observable Dynamic Processes: Adaptive Dynamic Programming Using Measured Output Data," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, Vol. 41, No. 1, 2011, pp. 14–25. doi:10.1109/TSMCB.2010.2043839.

[10] Kiumarsi, B., Lewis, F. L., Naghibi-Sistani, M., and Karimpour, A., "Optimal Tracking Control of Unknown Discrete-Time Linear Systems Using Input-Output Measured Data," *IEEE Transactions on Cybernetics*, Vol. 45, No. 12, 2015, pp. 2770–2779. doi:10.1109/TCYB.2014.2384016.

[11] Dias, P. M., Zhou, Y., and Kampen, E.-J. V., *Intelligent Nonlinear Adaptive Flight Control using Incremental Approximate Dynamic Programming*, 2019. doi:10.2514/6.2019-2339, URL https://arc.aiaa.org/doi/abs/10.2514/6.2019-2339.

[12] Lewis, F. L., and Vamvoudakis, K. G., "Optimal adaptive control for unknown systems using output feedback by reinforcement learning methods," *2010 8th IEEE International Conference on Control and Automation, ICCA 2010*, 2010, pp. 2138–2145. doi:10.1109/ICCA.2010.5524211.

[13] Bertsekas, D., and Tsitsiklis, J., *Neuro-Dynamic Programming*, Vol. 27, 1996. doi:10.1007/978-0-387-74759-0_440.

[14] Bellman, R., "Dynamic Programming," *Science*, Vol. 153, No. 3731, 1966, pp. 34–37. doi:10.1126/science.153.3731.34, URL https://science.sciencemag.org/content/153/3731/34.

[15] Bradtke, S., Ydstie, B., and Barto, A., "Adaptive Linear Quadratic Control Using Policy Iteration," *Proceedings of the American Control Conference*, Vol. 3, 1994. doi:10.1109/ACC.1994.735224.

[16] Jategaonkar, R., *Flight Vehicle System Identification: A Time Domain Methodology*, Vol. 216, 2006. doi:10.2514/4.866852.

[17] Zhou, Y., van Kampen, E.-J., and Chu, Q., "Incremental Approximate Dynamic Programming for Nonlinear Adaptive Tracking Control with Partial Observability," *Journal of Guidance, Control, and Dynamics*, Vol. 41, No. 12, 2018, pp. 2554–2567. doi:10.2514/1.G003472, URL https://doi.org/10.2514/1.G003472.

[18] "EASA.IM.A.207: Cessna 500, 550, S550, 560 and 560XL," , 2019. URL https://www.easa.europa.eu/documents/type-certificates/aircraft-cs-25-cs-22-cs-23-cs-vla-cs-lsa/easaima207.

[19] Oliveira, J., "DEVELOPMENT OF A FLEXIBLE FLIGHT TEST INSTRUMENTATION SYSTEM," 2006.

[20] Zaal, P., Pool, D., Postema, F., Veld, A., Mulder, M., Van Paassen, M. M., and Mulder, J., "Design and Certification of a Fly-By-Wire System with Minimal Impact on the Original Flight Controls," 2009. doi:10.2514/6.2009-5985.

[21] der Linden, V., "DASMAT-Delft University Aircraft Simulation Model and Analysis tool," 1996.

[22] Hoek, M., De Visser, C., and Pool, D., *Identification of a Cessna Citation II Model Based on Flight Test Data*, 2018, pp. 259–277. doi:10.1007/978-3-319-65283-2_14.

[23] van 't Veld, R., Kampen, E.-J. V., and Chu, Q. P., "Stability and Robustness Analysis and Improvements for Incremental Nonlinear Dynamic Inversion Control," *2018 AIAA Guidance, Navigation, and Control Conference*, 2018. doi:10.2514/6.2018-1127, URL https://arc.aiaa.org/doi/abs/10.2514/6.2018-1127.

[24] Söderström, T., and Stoica, P. (eds.), *System Identification*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[25] Vahidi, A., Stefanopoulou, A., and Peng, H., "Recursive least squares with forgetting for online estimation of vehicle mass and road grade: Theory and experiments," *Vehicle System Dynamics - VEH SYST DYN*, Vol. 43, 2005, pp. 31–55. doi:10.1080/00423110412331290446.

[26] Joos, H.-D., "A methodology for multi-objective design assessment and flight control synthesis tuning," *Aerospace Science and Technology*, Vol. 3, No. 3, 1999, pp. 161 – 176. doi:https://doi.org/10.1016/S1270-9638(99)80040-6, URL http://www.sciencedirect.com/science/article/pii/S1270963899800406.

3

Reinforcement Learning and Optimal Control

3.1. Reinforcement Learning Fundamentals

Reinforcement Learning is a sub class of machine learning where an agent in an environment chooses actions to maximize a reward signal. The term "reinforcement" in behaviour psychology means anything that tends to increase the likeliness of a response to occur. Thus the learning that is acquired through this reinforcement behaviour is called Reinforcement learning. RL process essentially involves a trail and error search method and memorization of situations/states and suitable actions reinforced through the rewards yielded from the environment. The figure 3.1 gives the representation of an RL process. Here we will define some of the common terms used in the context of RL:

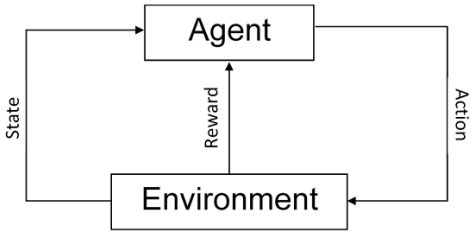


Figure 3.1: Reinforcement Learning Framework

- *Agent* : It is the one that learns from its experience and executes the suitable actions in a situation.
- *Environment* : Environment is anything that surrounds the agent and provides new situations to the agent based on its actions.
- *State(S)* : State/situation represents the information that is perceived by the agent from the environment thus characterizing the environment behaviour that is accessible by the agent.
- *Action(A)* : Actions are taken by the agent based on the state and rewards it receives which will influence the environment.
- *Reward(R)* : Reward signals are the ones responsible for the reinforcement behaviour which are provided by the environment to the agent.
- *Value(V)* : A value can be considered as a measure of accumulation of rewards in the longer run for a given state. Thus it characterizes how good a state is which can be evaluated based on experience.
- *Policy(π)* : A policy can be considered as an association from the states to the actions which helps the agent to act in a particular situation based on its experience thus characterizing the behaviour of the agent.
- *Model* : A model characterizes the behaviour of the environment which provides the information on its future states and rewards based on the present state and actions by the agent.

To define the general RL problem it is important to define the idealized mathematical formulation of the problem known as Markov Decision Process(MDP).

[0]This chapter has already been graded for AE4020.

3.1.1. Finite Markov Decision Process

In an MDP the agent continuously interacts with environment through the actions and the environment responds by creating new states based on the actions and also sends a reward signal. A finite MDP is the one where the states,actions and rewards are finite in number. The available states are denoted by S, actions by A, rewards by R. The probability of landing in state S_t and receiving reward R_t at time t is represented as:

$$p(s', r|s, a) \doteq Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\} \quad \forall \quad s', s \in S, r \in R, a \in A(s) \tag{3.1}$$

As s, r are expected from a probability distribution the following equality holds good:

$$p(s', r|s, a) \doteq Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\} \quad \forall \quad s \in S, a \in A(s) \tag{3.2}$$

The expected return G_t is a function of reward sequence and the objective of the agent is to maximize the expected return which is a function of the reward defined as:

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + ...R_T \tag{3.3}$$

where T is the final time step. The interaction between agent and environment is broken into sub sequences also known as episodes. For a discrete sequence of events each episode ends in a terminal state and equation 3.3 is valid. A discounted return is calculated using a discount rate $\gamma < 1$ and thus 3.3 is modified as:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + ... \quad = \quad \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{3.4}$$

The parameter γ controls the weight that should be given to the past rewards. Policy associates actions with states and is represented as π. $\pi(a|s)$ gives the probability that $A_t = a$ and $S_t = s$ when following policy π. The value of a state $v_\pi(s)$ gives the measure of how good it is to be in that state whilst following a policy π. Thus the term $v_\pi(s)$ is given by :

$$\begin{aligned} v_\pi(s) &= E_\pi[G_t|S_t = s] \\ &= E_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s] \\ &= \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')], \quad \forall \quad s \in S \end{aligned} \tag{3.5}$$

The equation 3.5 is called bellmann equation for the state-value function v_π following policy π. The action value function $q_\pi(s, a)$ gives a measure of how good the state is after taking action a whilst following policy π and is represented as:

$$q_\pi(s, a) = E_\pi[G_t|S_t = s, A_t = a] \tag{3.6}$$

The optimal policy π_* is defined as the one where the agent yields maximum return following policy π_* for all states. The optimal state value function and action value function are the ones which follow the optimal policy and using bellman equation we can define v_* and $q_*(s, a)$ as:

$$v_*(s) = max_a \sum_{s',r} p(s', r|s, a)[r + \gamma v_*(s')] \tag{3.7}$$

$$q_*(s, a) = \sum_{s',r} p(s', r|s, a)[r + \gamma max_{a'} q_*(s', a')] \tag{3.8}$$

The goal of the RL problem is to find optimal policy and optimal state value function. Once $v_*(s)$ is available for all states we can determine the optimal policy and thus the actions to take for a given state which would maximize the expected return. There are several methods that are adopted to solve this RL problem of MDP to find the optimal policy.

3.1.2. Dynamic Programming

Dynamic programming(DP) refers to the methods which involve breaking down a bigger decision making problem into sub problems and solving them[6]. To solve a DP problem it should have an optimal substructure and overlapping sub problems. The RL problem of finding optimal policy of the MDP is solved by using DP algorithms to estimate the value functions.

Policy Evaluation

For a given policy π the policy evaluation estimates the value function using Bellman equation 3.5. Initially the value function is initialized with an arbitrary initial values say zeros for all the states. Then the approximate values for the states are updated iteratively using the Bellman value function equation as:

$$v_{k+1}(s) = \sum_{a} \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_k(s')], \quad \forall\ s \in S \tag{3.9}$$

The value estimates are updated iteratively by considering possible one step transitions while following the policy π. As $k \to \infty$ the value will converge to v_π.

Policy Improvement

Once the value function for a given policy π is evaluated a new policy π' has to be evaluated such that it results in a better value function i.e $v_{\pi'}(s) > v_\pi(s)$. This new policy can be evaluated by considering a greedy action among possible actions at each given state according to:

$$\pi'(s) = argmax_a \sum_{s',r} p(s', r|s, a)[r + \gamma v_k(s')], \quad \forall\ s \in S \tag{3.10}$$

Policy Iteration

Policy iteration is the method of solving the RL problem through repeated sequence of policy evaluation and policy improvement until optimal solution is found[6]. An overview of the policy improvement is shown in figure 3.2. Initially a random policy π_0 is chosen and then policy evaluation is done iteratively to estimate the value function v_{π_0}. Now based on the value function a greedy policy π_1 is evaluated through policy evaluated and the process is repeated until the optimal policy π_* and optimal value function for this policy v_{π_*} is reached. Policy evaluation for a given policy is indicated from left to right and the policy improvement is indicated from top to bottom. The policy iteration method is used to solve the Jack's car rental problem which can be found from the book [30].

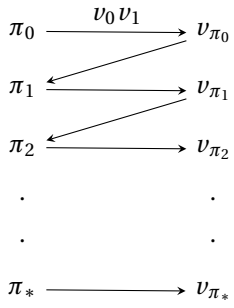


Figure 3.2: Policy Iteration

The solution to the above problem is shown in figure 3.3. The grid represents different states of the problem setting viz. number of cars in location 1 and 2. A policy is the association of possible actions to states. The possible actions are indicated by values from +5 to -5. The final policy π_4 is the optimal policy found and it can be seen that the solution suggests to move cars from its location to other location if one location has higher number of cars relative to the other. As the probability of rental requests at the second location is higher compared to the 1st one, the solution suggests to move less number of cars from location 2 to 1.

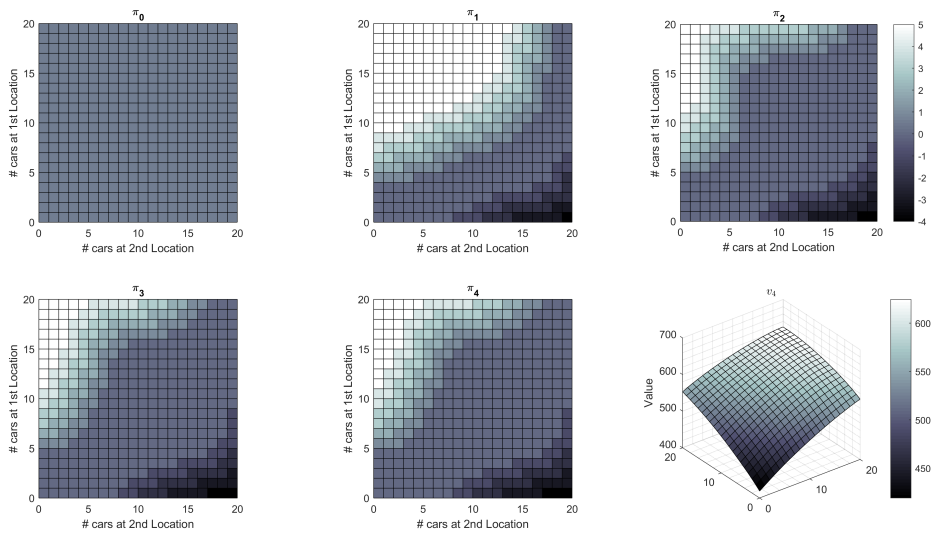


Figure 3.3: Policies and the final Value function for the Jack's car rental problem

Value Iteration

Value iteration is a special case of policy iteration method where instead of waiting for the exact convergence of policy evaluation, it is truncated to just 1 iteration and based on the approximate value function obtained policy improvement is done and the entire process is repeated till convergence. The combined method of policy improvement and one step policy evaluation can be represented by equation 3.11. The value iteration method used to solve the gambler's problem mentioned in the book [30].

$$v_{k+1}(s) = max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_k(s')], \quad \forall\, s \in S \tag{3.11}$$

The solution to the gambler's problem is found using value iteration method and the obtained optimal value function and optimal policy is shown in figure 3.4. The policy here represents the stake(action) for the available capital(state). The found policy is one of the optimal policies among possible optimal policies. This is an example to show that for a unique optimal value function there might exist several optimal policies.

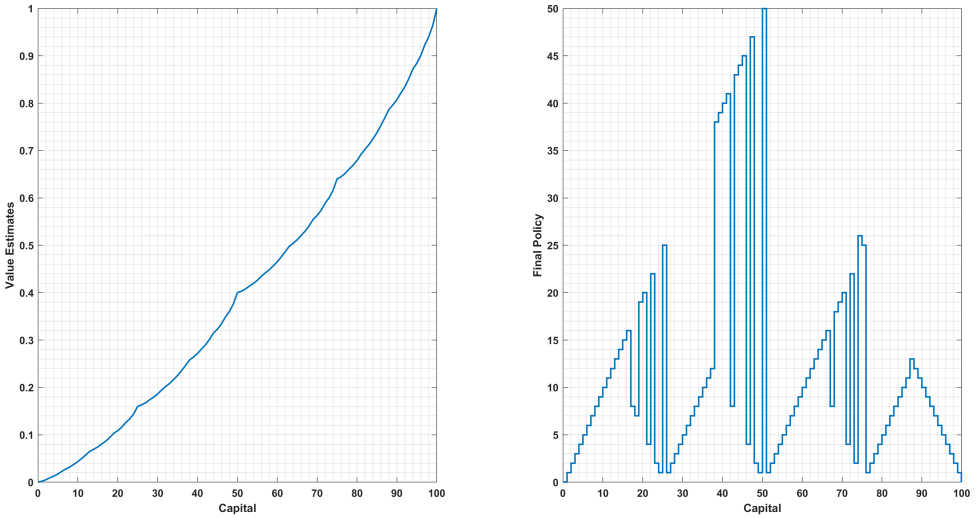


Figure 3.4: Optimal Value function and Policy for the Gambler's problem

The DP algorithms mentioned above can be used if the state space is small where value iteration update can be performed across all the states before proceeding towards policy improvement. For larger state space systems which suffer from the problem of *curse of dimensionality* asynchronous DP algorithms are preferred which gives control over the states for which value should be updated. The DP algorithms require the complete knowledge of the environment to estimate the state value functions however it is not practical to have knowledge of the environment in all the cases and thus methods like Monte Carlo Methods which attain the optimal behaviour through experience can be used to attend RL problem.

3.1.3. Model Free Reinforcement Learning

The DP algorithms discussed above can be used to solve the RL problem only when the model of the MDP process is available through the probability transition matrix $p(s', r|s, a)$. However it is not practical to have the MDP model and in this case we can solve the RL problem through model free methods where the agent learns directly from the experience gained from episodes. There are two different methods that can be categorized under model free RL methods:

- Monte Carlo Learning
- Temporal Difference Learning

The general idea of policy evaluation and improvement which form the basis for policy iteration is applied to solve the model free RL method. Given an unknown MDP the *prediction* problem solves for estimating the value or the action value function for a given policy while *control* problem involves convergence to the optimal policy(π_*) or the optimal value function(v_*).

3.1.4. Monte Carlo Prediction

"Monte Carlo(MC)" refers to the use of random sampling methods to approximate numerical results. Monte Carlo Methods in the context of RL refer to the learning of the agent in an environment through experience using sample returns observed. Thus instead of evaluating value function for all the states using a perfect known model of the environment we estimate the value of the states through some policy using the experience gained while visiting those states in an episode.

The MC methods differ from DP methods in two ways

- learns from experience instead of state space sweep to estimate value functions
- the value functions are estimated directly from returns instead of other value estimates

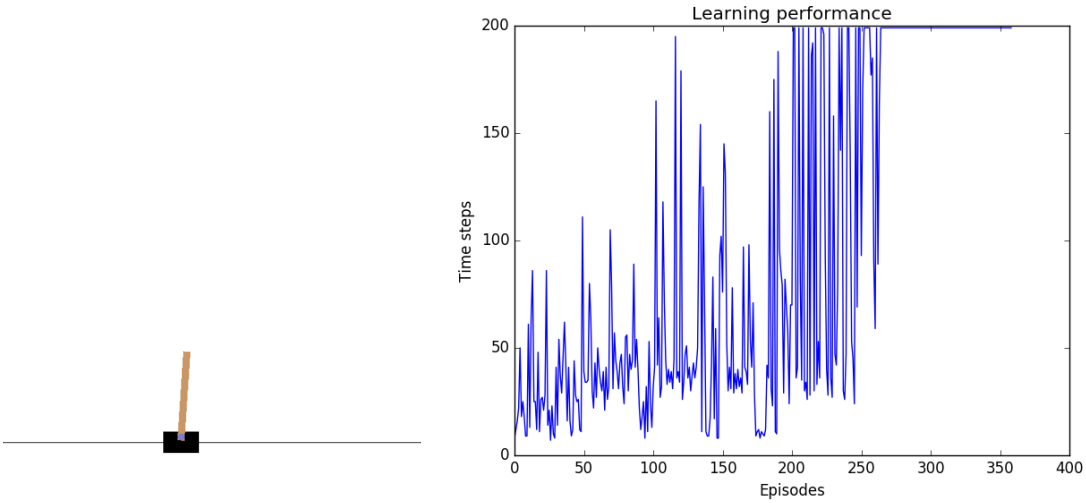
The temporal difference(TD) learning combines the advantage of learning from experience in MC and estimating value function from other value estimates which is known as bootstrapping.

3.1.5. Temporal Difference Learning

Temporal Difference(TD) learning combines the advantage of sampling from experience in MC methods and learning from incomplete episodes in DP methods. In a simple TD learning algorithm TD(0)[29] instead of averaging over the returns collected till the end of the episode to estimate $V(s)$ we will estimate by combining the return by taking a step ahead and the value estimate of the new state s' we have landed in as shown in equation 3.12 where $R_{t+1} + \gamma V(s')$ is called TD target and $\delta_t = R_{t+1} + \gamma V(s') - V(s)$ is called TD error. While in the MC prediction we need to wait till we finish the episode to estimate the value function for a particular state, in a TD prediction we can instead estimate the value of the state by just taking step ahead and then using the value estimate of the new state that we have landed in.

$$V(s) \leftarrow V(s) + \alpha(R_{t+1} + \gamma V(s') - V(s)) \tag{3.12}$$

An important difference between TD and MC method is that the TD learning algorithm exploits the markov property by first building an approximate model of the MDP and then converging the solution from the data for the estimated MDP whereas in an MC method the solution to the value estimate converges directly to the data obtained without exploiting this markov property.



(a) Cart pole system

(b) Learning performance

Figure 3.5: Q-Learning for Cart pole system

ϵ-Greedy exploration

It is not always possible to have exploring starts which is essential for the agent to explore to improve the policy. Instead of exploring starts to achieve the exploration we can use the concept of ϵ-greedy method to select the action in the policy improvement step where a greedy action is chosen with a probability of $1-\epsilon$ and a random action is chosen with ϵ probability as given by equation .

$$\pi'(a|s) \leftarrow \begin{cases} 1-\epsilon+\frac{\epsilon}{|A(s)|} & if \quad a = argmax_a Q(s,a) \\ \frac{\epsilon}{|A(s)|} & Otherwise \end{cases} \tag{3.13}$$

Q-Learning

Q-learning is an off-policy learning algorithm involving action value estimation where target policy π is learnt through samples of experience drawn from behaviour policy b [31]. Here the behaviour policy b can be an ϵ-greedy policy while the target policy π is a greedy policy. We do not consider importance sampling for Q-learning instead we consider bootstrapping of action value function from action a' selected according to the greedy policy π while the action a is selected according the ϵ-greedy policy b. The action value function is updated as shown in equation 3.14.

$$Q(s,a) \leftarrow Q(s,a) + \alpha(R_{t+1} + \gamma max_{a'} Q(s',a') - Q(s,a)) \tag{3.14}$$

The Q-Learning algorithm is applied for a cart pole balancing problem using open AI gym environment. The cart pole problem involves balancing a pole attached to a cart as shown in figure 3.5(a) which can move in a 2-D space either left or right and the reactive forces generated are used to balance the pole. The states viz,. cart position,cart velocity,pole angle, tip angular velocity are discretized into boxes and the q-learning algorithm is used to learn the optimal actions for possible states. The system reaches terminal states whenever the pole is more than 15deg from the vertical or the cart moves more than 2.4 units from the center. A reward of +1 is provided for every-time step the pole remains upright. According to open AI cart pole-v0 environment setting the problem is solved if an average reward of 195.0 is accumulated over 100 consecutive trails. The q-learning algorithms implemented solved the problem in 359 time-steps. The performance graph is shown in 3.5(b).

3.2. Approximate Dynamic Programming

The RL algorithms discussed so far can be classified as tabular solution methods where for every state in the state space the value function is evaluated and optimal policy is found for the problem. However for problems involving large state space or for the continuous state space problems involving infinite states, the tabular methods are not possible to evaluate the value function for all the states. We require methods which work irrespective of the size of the state space. The approximate dynamic programming methods solve this problem where instead of generating a look up table of value function for different states the method estimates the value function using function approximation[7] which is given by equation 3.15 where w represents a parameter vector which fits the value function approximator to the actual value function $v_\pi(s)$(assumed to be available) across the whole state space.

$$\hat{v}(s,w) \approx v_\pi(s) \tag{3.15}$$

Different types of function approximators can be considered for approximating the value function like neural networks, decision tress, multivariate regression or linear combination of features. Some of the considerations for selecting the type of function approximators is to be able to learn online where the agent interacts with its environment and whether the function approximators are able to handle the non stationary target functions.

Value function approximation using Gradient descent

Gradient descent is an optimization algorithm which is used to find the minimum of a function, here in our case we would like to find the local minimum of the function $J(w)$ parameterized by vector "w" where $J(w)$ represents the mean squared error between approximate value function $\hat{v}(s,w)$ and the true value function $v_\pi(s)$ as given by 3.16 over all the states.

$$J(w) = E_\pi[(v_\pi(S) - \hat{v}(S,w))^2] \tag{3.16}$$

The gradient descent algorithm evaluates the gradient of the function $J(w)$ given by 3.17 and the parameterized vector w is adjusted in the direction of the negative gradient given by 3.18. Stochastic Gradient Descent(SGD) methods update the parameterized vector on a single example selected stochastically instead of an expected update.

$$\nabla_w J(w) = \left(\frac{\partial J(w)}{\partial w_1}, \frac{\partial J(w)}{\partial w_2}, \ldots, \frac{\partial J(w)}{\partial w_n}\right)^T \tag{3.17}$$

$$\begin{aligned} \Delta w &= -\frac{1}{2}\alpha \nabla_w J(w) \\ &= \alpha E_\pi[v_\pi(S) - \hat{v}(S,w)]\nabla_w \hat{v}(S,w) \end{aligned} \tag{3.18}$$

Nonlinear function approximation: Artificial Neural Networks

ANN's are inspired from neurons and their connections within brain where the neurons are the represented by the units and connections between them represented by the links as shown in figure 3.6. The ANN structure typically consists of an input layer consisting of certain number of units, a hidden layer(s) and an output layer. Each link is associated with a weight representing the strength of connection between units. The units evaluate the weighted some from previous units and a nonlinear activation function is used to compute the final value for the unit. The ANN tries to fit the output data corresponding to the input data by evaluating the error term between the original output and the output from the ANN output layer and backpropagates the error term across the network. The weights of the links are adjusted such that the error term is minimized generally using stochastic gradient method. In the context of

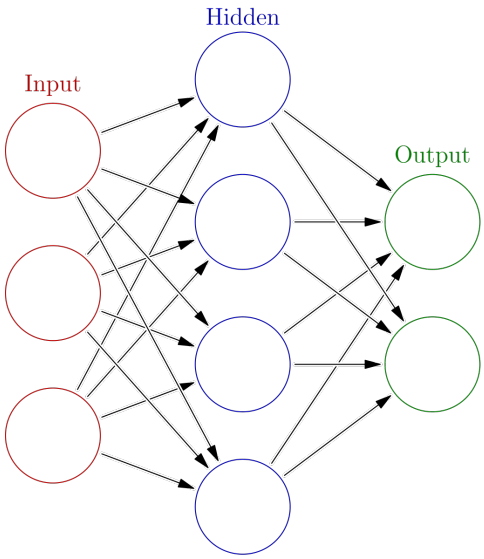


Figure 3.6: Artificial Neural Network

RL ANN's can be used to learn the value functions, or they can be used to maximize the expected reward [33] [34].

3.2.1. Policy gradient methods

So far the final objective of solving the reinforcement learning problem is to solve the control problem by estimating the value functions or action value functions and then deriving the optimal policy from the available value function estimates for optimal action selection. However in policy gradient methods actions are selected without necessarily estimating the action value functions but directly arriving at the optimal policy[35][13]. Similar to how a value function is parameterized as $\hat{v}(s,w)$ using vector w, the policy $\pi(a|s)$ is parameterized using vector θ as given by equation 3.19.

$$\pi(a|s) \approx \pi_\theta(a|s,\theta) \tag{3.19}$$

The reinforcement learning problem is solved by learning the parameter θ which maximizes a performance measure $J(\theta)$. In an episodic environment the performance measure to be maximized can be value of the starting state as per 3.20 where as in a continuous environment it can be the average value over a distribution of states visited as per 3.21 where $\mu_\pi(s)$ gives a measure of distribution of states according to how many times it was visited.

$$J(\theta) = v_{\pi_\theta}(s_0) \tag{3.20}$$

$$J(\theta) = \sum_s \mu_{\pi_\theta}(s) v_{\pi_\theta}(s) \tag{3.21}$$

The optimization problem of solving for best θ can be found using gradient ascent methods as per equation 3.22 where α is the step size.

$$\Delta\theta = \alpha \nabla_\theta J(\theta) \tag{3.22}$$

The policy gradient theorem provides the relationship between performance gradient $\nabla_\theta J(\theta)$ and the policy parameter θ as per equation 3.23.

$$\nabla_\theta J(\theta) \propto \sum_s \mu(s) \sum_a q_\pi(s,a) \nabla_\theta \pi(a|s,\theta) \tag{3.23}$$

3.2.2. Actor-Critic Methods

So far RL methods involve either learning the parameterized value function or learning the parameterized policy function as in policy gradient methods. However in Actor Critic methods both value function and policy function are learnt corresponding to the general policy iteration[35][13]. It is a way of solving the dynamic programming problem introduced before where both policy evaluation and policy improvement happens. The actor performs the policy improvement step while the critic evaluates the policy as shown in figure 3.7. The policy improvement can be done by learning the parameter θ using policy gradient methods while the policy evaluation can be done by learning parameter w used to evaluate the state-value function. To improve the speed of learning in policy gradient method instead of using Monte Carlo methods one can use temporal-difference methods with bootstrapping. A pseudo code for the actor critic algorithm using one-step method is given as shown in psuedo code 1.

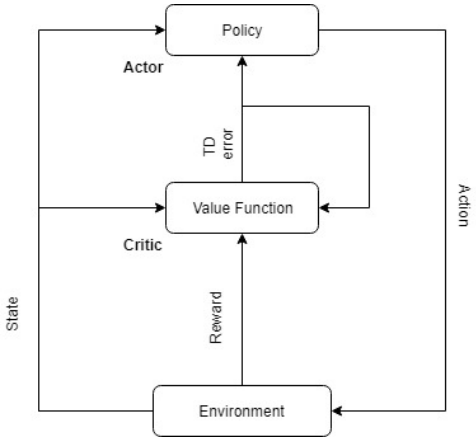


Figure 3.7: Actor-Critic architecture

Algorithm 1 One-Step Actor Critic Algorithm

Initialize $S, w, \theta, \alpha^w, \alpha^\theta$
while S is not a Terminal state **do**
  $A \sim \pi(.|S,\theta)$
  Observe S', R
  $\delta \leftarrow R + \gamma \hat{v}(S', w) - \hat{v}(S, w)$
  $w \leftarrow w + \alpha^w \delta \nabla_w \hat{v}(S, w)$
  $\theta \leftarrow \theta + \alpha^\theta \delta \nabla_\theta \ln \pi(A|S,\theta)$
  $S \leftarrow S'$
end while

3.3. Approximate Dynamic Programming for Feedback control

In this section the methodology of extending RL techniques to control a system will be explained. Firstly the concept of Feedback control and how it is achieved from classical or modern control theory is explained. Then the methodology of achieving feedback control using RL techniques will be explained [22]. As this method involves the knowledge of the system in the form of state representation and can only be used for offline planning a model free online learning approach would be more practical. This method exploits the advantage of value function approximation and TD concepts from ADP and uses the measured input and output data to arrive at the optimal control policy for stabilizing the system.

To achieve the control of a dynamic system they are mathematically represented as a first order ODE of the form $\dot{x} = f(x,u)$ where $x \in R^n$ represents the state and $u \in R^m$ represents the input. The control of the system is achieved through the input to the system whose dynamics are captured by the state. In aerospace applications the system dynamics are captured through identification techniques and then methods from control theory are applied to achieve the control so that we can steer the system to required state through the input. Typically the systems are nonlinear and are continuous in nature. As it is easier to achieve the control of a linear system, first the nonlinear systems are reduced to linear form at different equilibrium points and then control is achieved at these operating points. Also as the computers deal in discrete time, continuous systems are discretized or sampled and take the form as given by the equation 3.24.

$$\begin{aligned} x_{k+1} &= f(x_k) + g(x_k)u_k \\ y_k &= p(x_k) + q(x_k)u_k \end{aligned} \tag{3.24}$$

where x_k is the state vector, y_k is the output vector and u_k is the control input. The control policy is the one which maps the state space to the control input i.e, $u_k = h(x_k)$ and such controllers are called as feedback controllers. $h(x_k)$ can be any transfer function which can achieve desired control and different control approaches like optimal control, classical control can be used to achieve this.

In RL the actor tries to achieve this control policy through DP or MC or TD methods. In model free RL there is no need to identify the system dynamics to represent the system as 3.24 however the actor directly solves for the control policy instead. In an RL setting similar to value function a cost-to-go quadratic function is defined as given by 3.25 and the objective is to find the control policy which minimizes the cost and is called as optimal control policy represented as 3.26.

$$V_h(x_k) = \sum_{i=k}^{\infty} \gamma^{i-k} r(y_i, u_i) \tag{3.25}$$

$$V^*(x_k) = \underset{min}{h(.)} \left(\sum_{i=k}^{\infty} \gamma^{i-k} r(y_i, h(x_i)) \right) \tag{3.26}$$

Here $r(y_i, u_i)$ is the utility function and γ is the discount factor which provides the control over how important value the utility function is in future. This method is called goal directed optimal performance and the idea is similar to the RL problem setting where agent's goal is to find the optimal policy which maximizes the value function which captures the knowledge of sum of future rewards. The utility function can be defined as per the requirements and the following equation 3.27 shows utility function defined as quadratic energy function.

$$r(y_k, u_k) = y_k^T Q y_k + u_k^T R u_k \tag{3.27}$$

Extending the concept of Bellman equation in RL, equation 3.25 can be rewritten as 3.28

$$V_h(x_k) = r(y_k, h(x_k)) + \gamma V_h(x_{k+1}) \tag{3.28}$$

where $V_h(0) = 0$ which leads to the bellman optimality equation 3.29

$$V^*(x_k) = \underset{h(.)}{min}(r(y_k, h(x_k)) + V^*(x_k + 1)) \tag{3.29}$$

The optimality can be solved using DP techniques if model of the system is available in the form of 3.24 and they solve this problem offline. However model free methods are interesting from the perspective of adaptability which can also perform online learning in real time. The concepts of policy and value iteration from RL are extended to solve for the optimal control policy by defining the policy evaluation for a given policy and policy improvement steps as shown in equation 3.30 and 3.31 respectively.

$$V_{j+1}(x_k) = r(y_k, h_j(x_k)) + \gamma V_{j+1}(x_{k+1}) \tag{3.30}$$

$$h_{j+1}(x_k) = \underset{h(.)}{argmin}(r(y_k, h(x_k)) + \gamma V_{j+1}(x_{k+1})) \tag{3.31}$$

While the policy evaluation step converges to the value function for a given policy, the policy improvement step improves the current policy and these steps are repeated until the optimal control policy is achieved. This is similar to policy iteration(PI) algorithm introduced in DP chapter. Now value iteration(VI) algorithm is a special case of PI where the policy evaluation step is not iterated over and again till convergence is achieved instead a one step policy evaluation is performed and policy improvement is done over the improved value function estimate. As most of the practical applications involve discrete time we will see how control is achieved using optimal control techniques like LQR and how these techniques fall inline with the RL methods discussed before.

3.3.1. Discrete Time LQR

For discrete time linear systems equation 3.24 can be written as 3.32 with $A \in R^n$, $B \in R^m$ and $C \in R^p$.

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k \end{aligned} \tag{3.32}$$

The control policy can be a simple linear feedback gain matrix K i.e, $u_k = -Kx_k$ and the objective is to find the gain matrix which minimizes the defined cost function and this is called the linear quadratic regulator(LQR) problem. Substituting this control policy in 3.32 we get $x_{k+1} = (A - BK)x_k$. It can be shown that the optimal value for LQR will be of the form 3.33 for some matrix P.

$$V^*(x_k) = x_k^T P x_k \tag{3.33}$$

Using 3.27 and 3.28 we can rewrite this optimal value function as 3.34

$$\begin{aligned} x_k^T P x_k &= y_k^T Q y_k + u_k^T R u_k + \gamma x_{k+1}^T P x_{k+1} \\ &= x_k^T (C^T Q C + K^T R K + \gamma (A - BK)^T P (A - BK)) x_k \end{aligned} \tag{3.34}$$

As this equation is valid for all x_k we can rewrite 3.34 as 3.35 which is known as Lyapunov equation.

$$\gamma (A - BK)^T P (A - BK) - P + C^T Q C + K^T R K = 0 \tag{3.35}$$

Thus for a fixed gain K we can estimate the cost function $V_K(x_k) = x_k^T P x_k$ where P is obtained from 3.35. We have to find K which minimizes the cost function 3.35. Using 3.32 and 3.34 we can rewrite it as 3.36

$$x_k^T P x_k = y_k^T Q y_k + u_k^T R u_k + \gamma (Ax_k + Bu_k)^T P (Ax_k + Bu_k) \tag{3.36}$$

We can differentiate it with respect to u_k to achieve the optimal control action and we obtain the optimal control action as 3.37.

$$u_k = -(R/\gamma + B^T P B)^{-1} B^T P A x_k \tag{3.37}$$

Extracting K from above equation and substituting this in the Lyapunov equation 3.35 we get equation 3.38 which is quadratic in P and is known as Algebraic Ricatti Equation(ARE).

$$\gamma A^T P A - P + C^T Q C - \gamma A^T P B (R/\gamma + B^T P B)^{-1} B^T P A = 0 \tag{3.38}$$

The DT LQR is solved for optimal control policy using equation 3.37 by estimating the value P from ARE 3.38. The above method can be used to compute the gain matrix K offline. However we can achieve the optimal control policy through online real time computations using RL methods like DP using iterative techniques instead of LQR if the full state information is available. Here we define a Generalized Policy Algorithm(GPI) using LADP with full state feedback where the policy evaluation is done followed by policy improvement step and is given by algorithm 2. To solve the problem we still need the matrices A and B which

Algorithm 2 GPI for LADP-FS

Initialize a arbitrary control policy u_t^0
$P_j^0 = P_j$
repeat
Value Update Step: $x_t^T P^{j+1} x_t = \gamma x_{t+1}^T P^j x_{t+1} + y_t^T Q y_t + (u_t^j)^T R u_t^j$
Policy Improvement Step: $u_t^{j+1} = -(R/\gamma + B^T P^{j+1} B)^{-1} B^T P^{j+1} A$
until Convergence

define the system dynamics. However as we are interested in model free RL we need a method to achieve the optimal control policy without any prior information about the system and which avoids the state measurements. One of the methods to do this is using the input/output data using a technique called output feedback(OPFB).

3.3.2. Optimal control policy using OPFB

Consider the linear DT system as given in 3.32 with $A \in R^n$, $B \in R^m$ and $C \in R^p$. We assume that the system is controllable i.e, we can steer the state of system to a desired state and is dependent on the matrices (A, B). As we will be using the output measured data it is also assumed that the system is observable i.e, the output measurements captures the full state of the system and this is a property of matrices (A, C). Let the input output measurements be recorded at N different instances till k^{th} time step i.e, between interval $[k-N, k]$ and we can construct the following equations 3.39,3.40 using 3.32.

$$x_k = A^N x_{k-N} + \begin{bmatrix} B & AB & A^2B \dots A^{N-1}B \end{bmatrix} \begin{bmatrix} u_{k-1} \\ u_{k-2} \\ u_{k-3} \\ \vdots \\ u_{k-N} \end{bmatrix} \tag{3.39}$$

$$\begin{bmatrix} y_{k-1} \\ y_{k-2} \\ \vdots \\ y_{k-N} \end{bmatrix} = \begin{bmatrix} CA^{N-1} \\ \vdots \\ CA \\ C \end{bmatrix} x_{k-N} + \begin{bmatrix} 0 & CB & CAB & \dots & CA^{N-2}B \\ 0 & 0 & CB & \dots & CA^{N-3}B \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \dots & \dots & CB \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_{k-1} \\ u_{k-2} \\ \vdots \\ u_{k-N} \end{bmatrix} \tag{3.40}$$

These can be simplified as 3.41 where $\bar{y}_{k-1,k-N} \in R^{pN}$ and $\bar{u}_{k-1,k-N} \in R^{mN}$.

$$\begin{aligned} x_k &= A^N x_{k-N} + U_N \bar{u}_{k-1,k-N} \\ \bar{y}_{k-1,k-N} &= V_N x_{k-N} + T_N \bar{u}_{k-1,k-N} \end{aligned} \tag{3.41}$$

where U_N is the controllability matrix and V_N being the observability matrix. Extracting x_{k-N} from the equation 3.41 we get

$$x_{k-N} = V_N^+(\bar{y}_{k-1,k-N} - T_N \bar{u}_{k-1,k-N}) \tag{3.42}$$

where V_N^+ is the pseudo inverse of the observability matrix given by $V_N^+ = (V_N^T V_N)^{-1} V_N^T$. Using the above identity in 3.41 we arrive at the following expression for the full state vector.

$$x_k = A^N V_N^+ \bar{y}_{k-1,k-N} + (U_N - A^N T_N V_N^+) \bar{u}_{k-1,k-N} \tag{3.43}$$

For the ease of notation it can be represented as 3.44 where $\bar{z}_{k-1,k-N} \in R^{(m+p)N}$.

$$\begin{aligned} x_k &= \begin{bmatrix} M_u & M_y \end{bmatrix} \begin{bmatrix} \bar{u}_{k-1,k-N} \\ \bar{y}_{k-1,k-N} \end{bmatrix} \\ &= M \bar{z}_{k-1,k-N} \end{aligned} \tag{3.44}$$

Similar to the approach discussed before we use the cost-to-go quadratic function and substitute the above identity for the state vector represented in input output data.

$$\begin{aligned} V_h(x_k) &= x_k^T P x_k \\ &= \bar{z}_{k-1,k-N}^T M^T P M \bar{z}_{k-1,k-N} \\ &= \bar{z}_{k-1,k-N}^T \bar{P} \bar{z}_{k-1,k-N} \end{aligned} \tag{3.45}$$

Here $\bar{P} \in R^{(m+p)N \times (m+p)N}$ now captures the dynamics and we use the RL techniques like GPI to estimate $\bar{P}$. From 3.34 and above cost equation we arrive at 3.46

$$0 = -\bar{z}_{k-1,k-N}^T \bar{P} \bar{z}_{k-1,k-N} + y_k^T Q y_k + u_k^T R u_k + \gamma \bar{z}_{k,k-N+1}^T \bar{P} \bar{z}_{k,k-N+1} \tag{3.46}$$

This term is also called as TD error where,

$$\bar{z}_{k,k-N+1}^T \bar{P} \bar{z}_{k,k-N+1} = \begin{bmatrix} u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_{k,k-N+1} \end{bmatrix}^T \begin{bmatrix} p_0 & p_u & p_y \\ p_u^T & P_{22} & P_{23} \\ p_y^T & P_{32} & P_{33} \end{bmatrix} \begin{bmatrix} u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_{k,k-N+1} \end{bmatrix} \tag{3.47}$$

and $p_o \in R^{m \times m}$, $p_u \in R^{m \times (m(N-1))}$ and $p_y \in R^{m \times pN}$. The objective is to minimize the TD error which is dependent on u_k which can be differentiated yielding 3.48

$$u_k = -(R/\gamma + p_0)^{-1}(p_u \bar{u}_{k-1,k-N+1} + p_y \bar{y}_{k,k-N+1}) \tag{3.48}$$

Now we can define the GPI algorithm to solve this optimization problem to arrive at the optimal control policy and is given by algorithm 3.

Algorithm 3 GPI for OPFB

Initialize a stabilizing control policy $u_k^0 = h^0(x_k)$
repeat

Value Update Step: Solve for $\bar{P}_{j+1}$

$$0 = -\bar{z}_{k-1,k-N}^T \bar{P}^{j+1} \bar{z}_{k-1,k-N} + y_k^T Q y_k + u_k^T R u_k + \gamma \bar{z}_{k,k-N+1}^T \bar{P}^{j+1} \bar{z}_{k,k-N+1}$$

Policy Improvement Step: Use $\bar{P}_{j+1}$ to improve the policy

$$u_k^{j+1} = -(R/\gamma + p_0^{j+1})^{-1}(p_u^{j+1} \bar{u}_{k-1,k-N+1} + p_y^{j+1} \bar{y}_{k,k-N+1})$$

until Convergence

3.3.3. Optimal tracking control

Similar to the LQR problem discussed above where input output data is used to stabilize the system using online reinforcement learning methods, a more general reference tracking problem can also be addressed using a similar method[18]. The ADP method is used to design the optimal controller which require input,output data along with the reference trajectory data.

Consider a LTI Discrete time system in the same form as 3.32. We define a Linear Quadratic Tracking(LQT) problem where the goal is to arrive at the optimal control input u_k so that the system follows a reference trajectory r_k. The utility function now can be defined in terms of the reference tracking as follows:

$$r(r_k, y_k) = (y_k - r_k)^T Q(y_k - r_K) + u_k^T R u_k \tag{3.49}$$

where the trajectory dynamics are given by $r_{k+1} = F r_k$. The linear DT system is now augmented to include the trajectory dynamics as follows:

$$X_{k+1} = \begin{bmatrix} x_{k+1} \\ r_{k+1} \end{bmatrix} = \begin{bmatrix} A & 0 \\ 0 & F \end{bmatrix} \begin{bmatrix} x_k \\ r_k \end{bmatrix} + \begin{bmatrix} B \\ 0 \end{bmatrix} u_k = T X_k + B_1 u_k \tag{3.50}$$

The optimal value function 3.33 can now be written as 3.51

$$V^*(X_k) = X_k^T P X_k \tag{3.51}$$

The Bellman equation 3.34 is now modified to include the augmented state matrix and the reference trajectory as 3.52

$$X_k^T P X_k = (y_k - r_k)^T Q(y_k - r_k) + u_k^T R u_k + \gamma X_{k+1}^T P X_{k+1} \tag{3.52}$$

If the model of the system is available the above problem can be solved using the full state feedback using a GPI RL method. However as we are interested in a model free approach optimal tracking problem is solved using the input output data along with the reference trajectory data.

Let the input output measurements be recorded at N different instances till k^{th} time step i.e, between interval $[k-N, k]$ and we can construct the following equations 3.53 3.54 using 3.52 and 3.50

$$\begin{bmatrix} x_k \\ r_k \end{bmatrix} = \begin{bmatrix} A^N & 0 \\ 0 & F^N \end{bmatrix} \begin{bmatrix} x_{k-N} \\ r_{k-N} \end{bmatrix} + \begin{bmatrix} B & AB & A^2B \dots A^{N-1}B \end{bmatrix} \begin{bmatrix} u_{k-1} \\ u_{k-2} \\ u_{k-3} \\ \vdots \\ u_{k-N} \end{bmatrix} \tag{3.53}$$

$$\begin{bmatrix} y_{k-1} \\ y_{k-2} \\ \vdots \\ y_{k-N} \end{bmatrix} = \begin{bmatrix} CA^{N-1} \\ \vdots \\ CA \\ C \end{bmatrix} x_{k-N} + \begin{bmatrix} 0 & CB & CAB & \dots & CA^{N-2}B \\ 0 & 0 & CB & \dots & CA^{N-3}B \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \dots & \dots & CB \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_{k-1} \\ u_{k-2} \\ \vdots \\ u_{k-N} \end{bmatrix} \tag{3.54}$$

These can be simplified as 3.55.

$$\begin{bmatrix} x_k \\ r_k \end{bmatrix} = \begin{bmatrix} A^N & 0 \\ 0 & F^N \end{bmatrix} \begin{bmatrix} V_N^+(\bar{y}_{k-1,k-N} - T_N \bar{u}_{k-1,k-N}) \\ r_{k-N} \end{bmatrix} + \begin{bmatrix} U_N \\ 0 \end{bmatrix} \bar{u}_{k-1,k-N} \tag{3.55}$$
$$\bar{y}_{k-1,k-N} = V_N x_{k-N} + T_N \bar{u}_{k-1,k-N}$$

Rewriting the above equation as 3.56

$$\begin{bmatrix} x_k \\ r_k \end{bmatrix} = \begin{bmatrix} U_N - A^N V_N^+ D_N & A^N V_N^+ & 0 \\ 0 & 0 & F^N \end{bmatrix} \begin{bmatrix} \bar{u}_{k-1,k-N} \\ \bar{y}_{k-1,k-N} \\ r_{k-N} \end{bmatrix} \tag{3.56}$$

Which is simplified as 3.57.

$$X_k = M\bar{Z}_{k-1,k-N} \tag{3.57}$$

Now the optimal value function can be rewritten using $\bar{Z}$ as follows:

$$X_k^T P X_k = \bar{Z}_k^T M^T P M \bar{Z}_k = \bar{Z}_k^T \bar{P} \bar{Z}_k \tag{3.58}$$

Here $\bar{P} \in R^{(m+p)N\times(m+p)N}$ now captures the system dynamics along with the reference trajectory dynamics and we use the RL techniques like GPI to estimate $\bar{P}$. Rewriting the Bellman equation 3.52 using the above identity we get 3.59

$$\bar{Z}_{k-1,k-N}^T \bar{P} \bar{Z}_{k-1,k-N} = (y_k - r_k)^T Q(y_k - r_k) + u_k^T R u_k + \gamma \bar{Z}_{k,k-N+1}^T \bar{P} \bar{Z}_{k,k-N+1} \tag{3.59}$$

where,

$$\bar{Z}_{k,k-N+1}^T \bar{P} \bar{Z}_{k,k-N+1} = \begin{bmatrix} u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_{k,k-N+1} \\ r_{k-N+1} \end{bmatrix}^T \begin{bmatrix} p_0 & p_u & p_y & p_r \\ p_u^T & P_{22} & P_{23} & P_{24} \\ p_y^T & P_{32} & P_{33} & P_{34} \\ p_r^T & P_{42} & P_{43} & P_{44} \end{bmatrix} \begin{bmatrix} u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_{k,k-N+1} \\ r_{k-N+1} \end{bmatrix} \tag{3.60}$$

For the policy improvement step we have to minimize the defined value function which is dependent on u_k. Thus differentiating value function expression represented in Bellman equation format with respect to the control input we get

$$u_k = -(R/\gamma + p_0)^{-1}(p_u \bar{u}_{k-1,k-N+1} + p_y \bar{y}_{k,k-N+1} + p_r r_{k-N+1}) \tag{3.61}$$

Now we can define the GPI algorithm to solve this optimization problem to arrive at the optimal tracking control policy and is given by algorithm 4.

Algorithm 4 GPI for Optimal tracking

Initialize a stabilizing control policy $u_k^0 = h^0(x_k)$
repeat

Value Update Step: Solve for $\bar{P}_{j+1}$

$$\bar{Z}_{k-1,k-N}^T \bar{P}^{j+1} \bar{Z}_{k-1,k-N} = (y_k - r_k)^T Q(y_k - r_k) + u_k^T R u_k + \gamma \bar{Z}_{k,k-N+1}^T \bar{P}^{j+1} \bar{Z}_{k,k-N+1}$$

Policy Improvement Step: Use $\bar{P}_{j+1}$ to improve the policy

$$u_k = -(R/\gamma + p_0^{j+1})^{-1}(p_u^{j+1} \bar{u}_{k-1,k-N+1} + p_y^{j+1} \bar{y}_{k,k-N+1} + p_r^{j+1} r_{k-N+1})$$

until Convergence

3.4. Incremental Approximate Dynamic Programming

The proposed Linear Approximate Dynamic Programming(LADP) method learns a linear model of the system online for achieving the optimal control. However as most of the complex systems like aerospace systems are nonlinear especially in the context of FTC as faults introduce non-linearities, it is desirable if the ADP method can be extended to nonlinear systems as well. Techniques to break down a nonlinear system into a time varying linear system using incremental form have been proposed and have been successfully applied for achieving adaptive non-linear control through Incremental nonlinear dynamic inversion(INDI)[26] and Incremental Backstepping(IBS)[27]. Once the incremental version of the nonlinear aircraft model is obtained the ADP techniques are used to achieve the optimal control using full state feedback or input output data [37].

3.4.1. Incremental model

Consider a Non-linear continuous system represented as follows:

$$\begin{aligned} \dot{x}(t) &= f[x(t), u(t)] \\ y(t) &= h[x(t)] \end{aligned} \tag{3.62}$$

where system dynamics are represented by the state vector $f[x(t), u(t)] \in R^n$ and output measurements are obtained using the measurement vector $h[x(t)] \in R^p$. As in practice we work with the discrete systems for achieving the control the above nonlinear system is discretized using a high sampling frequency and is represented as 3.63.

$$\begin{aligned} x_{k+1} &= f(x_k, u_k) \\ y_k &= h(x_k) \end{aligned} \tag{3.63}$$

We can linearize the above system around x^*, u^* by taking the first order Taylor series expansion and we obtain

$$x_{k+1} \approx f(x^*, u^*) + \frac{\partial f(x_k, u_k)}{\partial f(x_k)}|_{x^*,u^*}(x_k - x^*) + \frac{\partial f(x_k, u_k)}{\partial f(u_k)}|_{x^*,u^*}(u_k - u^*) \tag{3.64}$$

As it is assumed that the discretization is done at a high sampling frequency we can consider Δt to be very small and can approximate $x_{k-1} \approx x_k$ and can replace x^*, u^* with x_{k-1}, u_{k-1} to get 3.65

$$\begin{aligned} x_{k+1} - x_k &\approx \frac{\partial f(x_k, u_k)}{\partial f(x_k)}|_{x_{k-1},u_{k-1}}(x_k - x_{k-1}) + \frac{\partial f(x_k, u_k)}{\partial f(u_k)}|_{x_{k-1},u_{k-1}}(u_k - u_{k-1}) \\ \Delta x_{k+1} &\approx F(x_{k-1}, u_{k-1})\Delta x_k + G(x_{k-1}, u_{k-1})\Delta u_k \end{aligned} \tag{3.65}$$

where $F(x_{k-1}, u_{k-1}) = \frac{\partial f(x_k, u_k)}{\partial f(x_k)}|_{x_{k-1},u_{k-1}} \in R^{n\times n}$ is the system matrix and $G(x_{k-1}, u_{k-1}) = \frac{\partial f(x_k, u_k)}{\partial f(u_k)}|_{x_{k-1},u_{k-1}} \in R^{n\times m}$ is the control effectiveness matrix. This regression model represented by F_{t-1}, G_{t-1} can be identified using Recursive Least Squares(RLS) techniques with the assumption that $\Delta x(k), \Delta u(k)$ are measurable for $M \geq (n+m)$ samples. Using the utility function 3.27 and extending the concept of Bellman equation 3.28 for a control policy h to the incremental model we get 3.66

$$\begin{aligned} V_h(x_k) &= y_k^T Q y_k + u_k^T R u_k + \gamma V_h(x_{k+1}) \\ &= y_k^T Q y_k + (u_{k-1} + \Delta u_k)^T R(u_{k-1} + \Delta u_k) + \gamma V_h(x_{k+1}) \end{aligned} \tag{3.66}$$

The optimal Value function is given by 3.67

$$V^*(x_k) = min_{\Delta u_k}[y_k^T Q y_k + (u_{k-1} + \Delta u_k)^T R(u_{k-1} + \Delta u_k) + \gamma V^*(x_{k+1})] \tag{3.67}$$

Where the optimal control at time step k is given by 3.68

$$h*(\Delta x_k) = argmin_{\Delta u_k}[y_k^T Q y_k + (u_{k-1} + \Delta u_k)^T R(u_{k-1} + \Delta u_k) + \gamma V^*(x_{k+1})] \tag{3.68}$$

where the control law(policy) h is a function in the incremental form:

$$\Delta u_k = h(u_{k-1}, x_k, \Delta x_k) \tag{3.69}$$

using the value function representation as quadratic in state 3.33 we can write 3.66 as 3.70

$$\begin{aligned} x_k^T P x_k &= y_k^T Q y_k + u_k^T R u_k + \gamma x_{k+1}^T P x_{k+1} \\ &= y_k^T Q y_k + (u_{k-1} + \Delta u_k)^T R(u_{k-1} + \Delta u_k) + \\ &\quad \gamma(x_k + F_{k-1}\Delta x_k + G_{k-1}\Delta u_k)^T P(x_k + F_{k-1}\Delta x_k + G_{k-1}\Delta u_k) \end{aligned} \tag{3.70}$$

For optimal control we can set the derivative of the above cost function with respect to Δu_k to 0 and we get 3.71

$$\Delta u_k = -(R + \gamma G_{k-1}^T P G_{k-1})^{-1}[R u_{k-1} + \gamma G_{k-1}^T P x_k + \gamma G_{k-1} P F_{k-1} \Delta x_k] \tag{3.71}$$

When the full state of the system dynamics is available we can extend the LADP-FS algorithm to iADP-FS where the optimal control policy is learnt online using ADP methods like PI or VI. The GPI algorithm for iADP-FS is given by the algorithm 5

Algorithm 5 iADP using Full State Feedback

Initialize a arbitrary control policy Δu_k^0
$P_j^0 = P_j$
repeat
Value Update Step: $x_k^T P^{j+1} x_k = \gamma x_{k+1}^T P^j x_{k+1} + y_k^T Q y_k + (u_k^j)^T R u_k^j$
Policy Improvement Step:
$\Delta u_k = -(R + \gamma G_{k-1}^T P G_{k-1})^{-1}[R u_{k-1} + \gamma G_{k-1}^T P x_k + \gamma G_{k-1} P F_{k-1} \Delta x_k]$
until Convergence

3.4.2. Optimal Control using Output Feedback iADP-OPFB

Here we define the methodology for iADP algorithm using output feedback(OPFB) when the full state information is not available. In OPFB method the input output measurements are used to indirectly construct the state information which is then used to arrive at the optimal control using ADP methods.

Consider the nonlinear system 3.63. Linearizing the output of the system using First order Taylor series expansion around x_{t-1} we get 3.72

$$\Delta y_k \approx H_{k-1} \Delta x_k \tag{3.72}$$

where $H_{k-1} = \frac{\partial h(x)}{\partial x}|_{x_{k-1}} \in R^{p \times n}$ is the observation matrix. Consider we measure the data at N timesteps between interval $[k-N, k]$, using equations 3.63 and 3.63 we can write 3.73,

$$\begin{aligned} \Delta x_k &\approx \tilde{F}_{k-2,k-N-1} \Delta x_{k-N} + U_N \bar{\Delta u}_{k-1,k-N} \\ \bar{\Delta y}_{k,k-N+1} &\approx V_N \Delta x_{k-N} + T_N \bar{\Delta u}_{k-1,k-N} \end{aligned} \tag{3.73}$$

where $\tilde{F}_{k-a,k-b} = \Pi_{i=k-a}^{k-b} F_i$,

$$\bar{\Delta u}_{k-1,k-N} = \begin{bmatrix} \Delta u_{k-1} \\ \Delta u_{k-2} \\ \vdots \\ \Delta u_{k-N} \end{bmatrix} \in R^{mN}, \bar{\Delta y}_{k,k-N+1} = \begin{bmatrix} \Delta y_k \\ \Delta y_{k-1} \\ \vdots \\ \Delta y_{k-N+1} \end{bmatrix} \in R^{pN},$$

$U_N = \begin{bmatrix} G_{k-2} & F_{k-2} G_{k-3} \dots \tilde{F}_{k-2,k-N} G_{k-N-1} \end{bmatrix} \in R^{n \times mN}$ is the controllability matrix,

$$V_N = \begin{bmatrix} H_{k-1} \tilde{F}_{k-2,k-N-1} \\ H_{k-2} \tilde{F}_{k-3,k-N-1} \\ \vdots \\ H_{k-N} F_{k-N-1} \end{bmatrix} \in R * pN \times n \text{ is the observability matrix,}$$

$$T_N = \begin{bmatrix} H_{k-1}G_{k-2} & H_{k-1}F_{k-2}G_{k-3} & H_{k-1}\tilde{F}_{k-2,k-3}G_{k-4} & \dots & H_{k-1}\tilde{F}_{k-2,k-N}G_{k-N-1} \\ 0 & H_{k-2}G_{k-3} & H_{k-2}F_{k-3}G_{k-4} & \dots & H_{k-2}\tilde{F}_{k-3,k-N}G_{k-N-1} \\ 0 & 0 & H_{k-3}G_{k-4} & \dots & H_{k-3}\tilde{F}_{k-4,k-N}G_{k-N-1} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 & H_{k-N}G_{k-N-1} \end{bmatrix} \in R^{pN\times mN}.$$

Equation 3.73 can be rewritten as 3.74 by extracting Δx_{k-N}

$$\begin{aligned} \Delta x_k &\approx \tilde{F}_{k-2,k-N-1}V_N^+\bar{\Delta y}_{k,k-N+1} + (U_N - \tilde{F}_{k-2,k-N-1}V_N^+T_N)\bar{\Delta u}_{k-1,k-N} \\ &\approx \begin{bmatrix} M_{\Delta u} & M_{\Delta y} \end{bmatrix} \begin{bmatrix} \bar{\Delta y}_{k,k-N+1} \\ \bar{\Delta u}_{k-1,k-N} \end{bmatrix} \\ &\approx M_{k-1}\bar{\Delta z}_{k,k-N} \end{aligned} \tag{3.74}$$

where $V_N^+ = (V_N^TV_N)^{-1}V_N^T$, $M_{\Delta u} = (U_N - \tilde{F}_{k-2,k-N-1}V_N^+T_N) \in R^{n\times mN}$, $M_{\Delta y} = \tilde{F}_{k-2,k-N-1}V_N^+ \in R^{n\times pN}$.
Writing the output equation using measure input/output data between interval $[k-N, k-1]$ we get:

$$\bar{\Delta y}_{k-1,k-N} \approx \bar{V}_N\Delta x_{k-N} + \bar{T}_N\bar{\Delta u}_{k-1,k-N} \tag{3.75}$$

where $\bar{V_N} = \begin{bmatrix} H_{k-2}\tilde{F}_{k-3,k-N-1} \\ H_{k-3}\tilde{F}_{k-4,k-N-1} \\ \vdots \\ H_{k-N-1} \end{bmatrix} \in R * pN \times n,$

$$\bar{T_N} = \begin{bmatrix} 0 & H_{k-2}G_{k-3} & H_{k-2}F_{k-3}G_{k-4} & \dots & H_{k-2}\tilde{F}_{k-3,k-N}G_{k-N-1} \\ 0 & 0 & H_{k-3}G_{k-4} & \dots & H_{k-3}\tilde{F}_{k-4,k-N}G_{k-N-1} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 & H_{k-N}G_{k-N-1} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \in R^{pN\times mN}.$$

Where $\bar{V_N}^+ = (\bar{V_N}^T\bar{V_N})^{-1}\bar{V_N}^T$. Extracting Δx_{k-N} from 3.75 and substituting it in 3.90 we get:

$$\begin{aligned} \Delta y_k &\approx (H_{k-1}U_N - H_{k-1}\tilde{F}_{k-2,k-N-1}\bar{V_N}^+\bar{T_N})\bar{\Delta u}_{k-1,k-N} + H_{k-1}\tilde{F}_{k-2,k-N-1}\bar{V_N}^+\bar{\Delta y}_{k-1,k-N} \\ &\approx \underline{G}_{k-1}\bar{\Delta u}_{k-1,k-N} + \underline{F}_{k-1}\bar{\Delta y}_{k-1,k-N} \end{aligned} \tag{3.76}$$

The output increment Δy_{k+1} can now be constructed using measured data of N previous steps:

$$\begin{aligned} \Delta y_{k+1} &\approx \underline{G}_k\bar{\Delta u}_{k-1,k-N} + \underline{F}_k\bar{\Delta y}_{k-1,k-N} \\ &\approx \underline{G}_{k,11}\Delta u_k + \underline{G}_{k,12}\Delta u_{\bar{k-1},k-N+1} + \underline{F}_k\Delta y_{\bar{k},k-N+1} \end{aligned} \tag{3.77}$$

where $\underline{G}_k \in R^{p\times Nm}$ is the extended control effectiveness matrix, $\underline{F}_k \in R^{p\times Np}$ is the extended system matrix, $\underline{G}_{k,11} \in R^{p\times m}$ and $\underline{G}_{k,12} \in R^{p\times(N-1)m}$ are partitioned matrices from $\underline{G}_k$.

We define the cost to go function using a kernel matrix $\bar{P} \in R^N(m+p) \times N(m+p)$ which is quadratic in the measured data $\bar{z}_{t,t-N} = \begin{bmatrix} \bar{u}_{t-1,t-N} \\ \bar{y}_{t,t-N+1} \end{bmatrix}$ as follows

$$V(z_{k,k-N}) = \bar{z}_{k,k-N}^T\bar{P}\bar{z}_{k,k-N} \tag{3.78}$$

The optimal control policy in terms of the measured data is now given by 3.79

$$h^*(z_{k,k-N}) = argmin_{\Delta u_k}[y_k^TQy_k + (u_k^TR(u_k) + \gamma\bar{z}_{k+1,k-N+1}^T\bar{P}\bar{z}_{k+1,k-N+1}] \tag{3.79}$$

where

$$\bar{z}^T_{k+1,k-N+1}\bar{P}\bar{z}_{k+1,k-N+1} = \begin{bmatrix} u_{k-1}+\Delta u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_k+\Delta y_{k+1} \\ \bar{y}_{k,k-N+2} \end{bmatrix}^T \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \\ P_{41} & P_{42} & P_{43} & P_{44} \end{bmatrix} \begin{bmatrix} u_{k-1}+\Delta u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_k+\Delta y_{k+1} \\ \bar{y}_{k,k-N+2} \end{bmatrix} \tag{3.80}$$

The policy improvement can be done by differentiating optimal value function expression with respect to Δu_k and we get:

$$\begin{aligned} \Delta u_k = &-[R+\gamma P_{11}+\gamma(\underline{F}^T_{k,11})P_{33}\underline{F}^T_{k,11}+\gamma P_{13}\underline{F}_{k,11}+\gamma(P_{13}\underline{F}_{k,11})^T]^{-1} \\ &[[R+\gamma P_{11}+\gamma(\underline{F}_{k,11})^T P^T_{13}]u_{k-1}+\gamma[\underline{F}^T_{k,11}P_{33}+P_{13}]y_k \\ &+\gamma[P_{12}+(\underline{F}_{k,11})^T P_{23}]\bar{u}_{k-1,k-N+1}+\gamma[P_{14}+(\underline{F}_{k,11})^T P_{34}]\bar{y}_{k,k-N+2} \\ &+\gamma[(\underline{F}_{k,11})^T P_{33}+P_{13}](\underline{F}_{k,12}\Delta\bar{u}_{k-1,k-N+1}+\underline{G}_k\bar{\Delta y}_{k,k-N+1}))] \end{aligned} \tag{3.81}$$

where $\underline{F}_{k,11} \in R^{p\times m} and \underline{F}_{k,11} \in R^{p\times(n-1)m}$ are the matrices partitioned from $\underline{F}_k$. Now we can define the GPI algorithm for iADP-OPFB as given by algorithm 6

Algorithm 6 GPI for iADP-OPFB

Initialize a stabilizing control policy Δu^0_k
repeat

Value Update Step: Solve for $\bar{P}_{j+1}$

$$0 = -\bar{z}^T_{k,k-N+1}\bar{P}^{j+1}\bar{z}_{k,k-N+1}+y^T_k Q y_k+u^T_k R u_k+\gamma\bar{z}^T_{k+1,k-N+2}\bar{P}^{j+1}\bar{z}_{k+1,k-N+2}$$

Policy Improvement Step: Use $\bar{P}_{j+1}$ to improve the policy

$$\begin{aligned} \Delta u_k = &-[R+\gamma P_{11}+\gamma(\underline{F}^T_{k,11})P_{33}\underline{F}^T_{k,11}+\gamma P_{13}\underline{F}_{k,11}+\gamma(P_{13}\underline{F}_{k,11})^T]^{-1} \\ &[[R+\gamma P_{11}+\gamma(\underline{F}_{k,11})^T P^T_{13}]u_{k-1}+\gamma[\underline{F}^T_{k,11}P_{33}+P_{13}]y_k \\ &+\gamma[P_{12}+(\underline{F}_{k,11})^T P_{23}]\bar{u}_{k-1,k-N+1}+\gamma[P_{14}+(\underline{F}_{k,11})^T P_{34}]\bar{y}_{k,k-N+2} \\ &+\gamma[(\underline{F}_{k,11})^T P_{33}+P_{13}](\underline{F}_{k,12}\Delta\bar{u}_{k-1,k-N+1}+\underline{G}_k\bar{\Delta y}_{k,k-N+1}))] \end{aligned}$$

until Convergence

The iADP method is extended to deal with the complex problem of tracking and the methodology of iADP-Tracking is explained in [11].

3.4.3. Full state feedback

For dynamical systems where full state measurements are available the observed measurements can be written as:

$$Y_k = \begin{bmatrix} y_k \\ y^r_k \end{bmatrix} = X_k \tag{3.82}$$

Using the utility function for achieving tracking control and extending the concept of Bellman equation to the incremental model, we get the optimal Value function (3.83)

$$V^*(X_k) = \min_{\Delta u_k}[(y_k-y^r_k)^T Q(y_k-y^r_k)+(u_{k-1}+\Delta u_k)^T R(u_{k-1}+\Delta u_k)+\gamma V^*(X_{k+1})] \tag{3.83}$$

Where the optimal control at time step k is given by (3.84)

$$\Delta u^* = \underset{\Delta u_k}{argmin}[(y_k-y^r_k)^T Q(y_k-y^r_k)+(u_{k-1}+\Delta u_k)^T R(u_{k-1}+\Delta u_k)+\gamma V^*(X_{k+1})] \tag{3.84}$$

using the quadratic value function approximation we get (3.85)

$$\begin{aligned} X_k^T P X_k &= (y_k - y_k^r)^T Q (y_k - y_k^r) + u_k^T R u_k + \gamma X_{k+1}^T P X_{k+1} \\ &= (y_k - y_k^r)^T Q (y_k - y_k^r) + (u_{k-1} + \Delta u_k)^T R (u_{k-1} + \Delta u_k) + \\ &\quad \gamma (X_k + T_{k-1} \Delta X_k + G_{k-1} \Delta u_k)^T P (X_k + T_{k-1} \Delta X_k + G_{k-1} \Delta u_k) \end{aligned} \tag{3.85}$$

For optimal control we can set the derivative of the above cost function with respect to Δu_k to 0 and we get the optimal control law (3.86)

$$\Delta u_k = -(R + \gamma G_{k-1}^T P G_{k-1})^{-1} [R u_{k-1} + \gamma G_{k-1}^T P X_k + \gamma G_{k-1} P T_{k-1} \Delta X_k] \tag{3.86}$$

The VI algorithm for iADP-FS is given by the algorithm (7)

Algorithm 7 iADP for tracking control using Full State Feedback

Initialize a arbitrary control policy $\Delta u_k^0 = \mu(X_k)$
repeat
Value Update Step: $X_k^T P X_k = (y_k - y_k^r)^T Q (y_k - y_k^r) + u_k^T R u_k + \gamma X_{k+1}^T P X_{k+1}$
Policy Improvement Step: $\Delta u_k = -(R + \gamma G_{k-1}^T P G_{k-1})^{-1} [R u_{k-1} + \gamma G_{k-1}^T P X_k + \gamma G_{k-1} P T_{k-1} \Delta X_k]$
until Convergence

3.4.4. Output feedback

Often in practice as measurement of full system states is not available controller design using input-output measurement data over suitable time horizon is desirable. In this method the input output measurements are used to indirectly construct the state information under the assumption that the system is observable. The measured data is then used to arrive at the optimal control using iADP method.

Consider we measure the data at N time steps between interval $[k-N, k]$, using equations (3.65) and (3.90) we can write (3.87),

$$\Delta X_k = \begin{bmatrix} \Delta x_k \\ \Delta r_k \end{bmatrix} \approx \begin{bmatrix} \tilde{F}_{k-2,k-N-1} & 0 \\ 0 & \tilde{D}_{k-2,k-N-1} \end{bmatrix} \begin{bmatrix} \Delta x_{k-N} \\ \Delta r_{k-N} \end{bmatrix} + \begin{bmatrix} U_N \\ 0 \end{bmatrix} \bar{\Delta u}_{k-1,k-N} \tag{3.87}$$

where $\tilde{F}_{k-a,k-b} = \Pi_{i=k-a}^{k-b} F_i$ and $\tilde{D}_{k-a,k-b} = \Pi_{i=k-a}^{k-b} D_i$, the input-output measurements captured over the time horizon [k-N,k] and the controllability matrix U_N are given by equations (3.88) and (3.89) respectively

$$\bar{\Delta u}_{k-1,k-N} = \begin{bmatrix} \Delta u_{k-1} \\ \Delta u_{k-2} \\ \vdots \\ \Delta u_{k-N} \end{bmatrix} \in R^{mN}, \qquad \bar{\Delta y}_{k,k-N+1} = \begin{bmatrix} \Delta y_k \\ \Delta y_{k-1} \\ \vdots \\ \Delta y_{k-N+1} \end{bmatrix} \in R^{pN} \tag{3.88}$$

$$U_N = \begin{bmatrix} G_{k-2} & F_{k-2} G_{k-3} \ldots \tilde{F}_{k-2,k-N} G_{k-N-1} \end{bmatrix} \in R^{n \times mN} \tag{3.89}$$

Linearizing the output of the nonlinear system(3.63) and the reference output of the system using First Order Taylor Series expansion around x_{k-1} we get (3.90) and (3.91) respectively

$$\Delta y_k \approx H_{k-1} \Delta x_k \tag{3.90}$$

$$\Delta y_k^r \approx H_{k-1}^r \Delta r_k \tag{3.91}$$

where $H_{k-1} = \frac{\partial h(x)}{\partial x}|_{x_{k-1}} \in R^{p \times n}$ and $H_{k-1}^r = \frac{\partial h^r(x)}{\partial x}|_{x_{k-1}} \in R^{r \times n}$ are the observation matrices. Now using the input-output data from (3.73) we can write (3.90) and (3.91) as follows

$$\begin{aligned} \bar{\Delta y}_{k,k-N+1} &\approx V_N \Delta x_{k-N} + W_N \bar{\Delta u}_{k-1,k-N} \\ \bar{\Delta y}_{k,k-N+1}^r &\approx R_N \Delta r_{k-N} \end{aligned} \tag{3.92}$$

The matrices V_N, W_N and R_N are given by equations (3.93), (3.94) and 3.95 respectively

$$V_N = \begin{bmatrix} H_{k-1}\tilde{F}_{k-2,k-N-1} \\ H_{k-2}\tilde{F}_{k-3,k-N-1} \\ \vdots \\ H_{k-N}F_{k-N-1} \end{bmatrix} \in R^{pN\times n} \tag{3.93}$$

$$W_N = \begin{bmatrix} H_{k-1}G_{k-2} & H_{k-1}F_{k-2}G_{k-3} & H_{k-1}\tilde{F}_{k-2,k-3}G_{k-4} & \dots & H_{k-1}\tilde{F}_{k-2,k-N}G_{k-N-1} \\ 0 & H_{k-2}G_{k-3} & H_{k-2}F_{k-3}G_{k-4} & \dots & H_{k-2}\tilde{F}_{k-3,k-N}G_{k-N-1} \\ 0 & 0 & H_{k-3}G_{k-4} & \dots & H_{k-3}\tilde{F}_{k-4,k-N}G_{k-N-1} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 0 & 0 & \dots & 0 & H_{k-N}G_{k-N-1} \end{bmatrix} \in R^{(p+m)\times N} \tag{3.94}$$

$$R_N = \begin{bmatrix} H^r_{k-1}\tilde{F}_{k-2,k-N-1} \\ H^r_{k-2}\tilde{F}_{k-3,k-N-1} \\ \vdots \\ H^r_{k-N}F_{k-N-1} \end{bmatrix} \in R^{rN\times l} \tag{3.95}$$

We can extract Δx_{k-N} and Δr_{k-N} from (3.92) as follows

$$\begin{aligned} \Delta x_{k-N} &\approx V_N^+(\bar{\Delta y}_{k,k-N+1} - W_N\bar{\Delta u}_{k-1,k-N}) \\ \Delta r_{k-N} &\approx R_N^+\bar{\Delta y}^r_{k,k-N+1} \end{aligned} \tag{3.96}$$

where $V_N^+ = (V_N^TV_N)^{-1}V_N^T$ and $R_N^+ = (R_N^TR_N)^{-1}R_N^T$ are the pseudo inverses of the respective matrices. Substituting (3.96) in (3.73) we get

$$\begin{aligned} \Delta X_k &\approx \begin{bmatrix} \tilde{F}_{k-2,k-N-1}V_N^+ & 0 \\ 0 & \tilde{D}_{k-2,k-N-1}R_N^+ \end{bmatrix} \begin{bmatrix} \bar{\Delta y}_{k,k-N+1} \\ \bar{\Delta y}^r_{k,k-N+1} \end{bmatrix} + \begin{bmatrix} U_N - \tilde{F}_{k-2,k-N-1}V_N^+W_N \\ 0 \end{bmatrix} \bar{\Delta u}_{k-1,k-N} \\ &\approx \begin{bmatrix} U_N - \tilde{F}_{k-2,k-N-1}V_N^+W_N & \tilde{F}_{k-2,k-N-1}V_N^+ & 0 \\ 0 & 0 & \tilde{D}_{k-2,k-N-1}R_N^+ \end{bmatrix} \begin{bmatrix} \bar{\Delta u}_{k-1,k-N} \\ \bar{\Delta y}_{k,k-N+1} \\ \bar{\Delta y}^r_{k,k-N+1} \end{bmatrix} \\ &\approx \begin{bmatrix} M_{\Delta u} & M_{\Delta y} & M_{\Delta y^r} \end{bmatrix} \bar{\Delta Z}_{k,k-N} \end{aligned} \tag{3.97}$$

Thus augmented state can be reconstructed using the input output data over a certain time horizon using above equation. It can also be shown that the output increments can also be constructed using past data measurements as follows:

$$\begin{aligned} \Delta y_{k+1} &\approx \underline{G}_k\bar{\Delta u}_{k-1,k-N} + \underline{F}_k\bar{\Delta y}_{k-1,k-N} \\ &\approx \underline{G}_{k,11}\Delta u_k + \underline{G}_{k,12}\Delta u_{k-1,k-N+1} + \underline{F}_k\bar{\Delta y}_{k,k-N+1} \end{aligned} \tag{3.98}$$

where $\underline{G}_k \in R^{p\times Nm}$ is the extended control effectiveness matrix, $\underline{F}_k \in R^{p\times Np}$ is the extended system matrix, $\underline{G}_{k,11} \in R^{p\times m}$ and $\underline{G}_{k,12} \in R^{p\times(N-1)m}$ are partitioned matrices from $\underline{G}_k$.

Similarly Δy^r_{k+1} can also be constructed as:

$$\Delta y^r_{k+1} \approx \underline{F}^r_k\bar{\Delta y}_{k-1,k-N} \tag{3.99}$$

where $\underline{F}^r_k \in R^{r\times Nr}$. We define the quadratic cost to go function using a kernel matrix and the measured data as follows

$$V(\bar{Z}_{k,k-N+1}) = \bar{Z}^T_{k,k-N+1}\bar{P}\bar{Z}_{k,k-N+1} \tag{3.100}$$

where $\bar{Z}$ contains the input output data given by

$$\bar{Z}_{k,k-N+1} = \begin{bmatrix} \bar{u}_{k-1,k-N} \\ \bar{y}_{k,k-N+1} \\ \bar{y}^r_{k,k-N+1} \end{bmatrix} \in R^{(m+p+r)N} \tag{3.101}$$

Extending the Bellman equation for tracking control using the incremental model representation using the input output data we get

$$\bar{Z}^T_{k,k-N+1}\bar{P}\bar{Z}_{k,k-N+1} = (y_k - y^r_k)^T Q(y_k - y^r_k) + (u_{k-1} + \Delta u_k)^T R(u_{k-1} + \Delta u_k) + \gamma \bar{Z}^T_{k+1,k-N+2}\bar{P}\bar{Z}_{k+1,k-N+2} \quad (3.102)$$

where

$$\bar{Z}^T_{k+1,k-N+2}\bar{P}\bar{Z}_{k+1,k-N+2} = \begin{bmatrix} u_{k-1}+\Delta u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_k + \Delta y_{k+1} \\ \bar{y}_{k,k-N+2} \\ \bar{y}^r_k + \Delta y^r_{k+1} \\ \bar{y}^r_{k,k-N+2} \end{bmatrix}^T \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} & P_{15} & P_{16} \\ P_{21} & P_{22} & P_{23} & P_{24} & P_{25} & P_{26} \\ P_{31} & P_{32} & P_{33} & P_{34} & P_{35} & P_{36} \\ P_{41} & P_{42} & P_{43} & P_{44} & P_{45} & P_{46} \\ P_{51} & P_{52} & P_{53} & P_{54} & P_{55} & P_{56} \\ P_{61} & P_{62} & P_{63} & P_{64} & P_{65} & P_{66} \end{bmatrix} \begin{bmatrix} u_{k-1}+\Delta u_k \\ \bar{u}_{k-1,k-N+1} \\ \bar{y}_k + \Delta y_{k+1} \\ \bar{y}_{k,k-N+2} \\ \bar{y}^r_k + \Delta y^r_{k+1} \\ \bar{y}^r_{k,k-N+2} \end{bmatrix} \quad (3.103)$$

The optimal control policy in terms of the measured data is now given by (3.104)

$$\begin{aligned} \Delta u_k &= \underset{\Delta u_k}{argmin}[(y_k - y^r_k)^T Q(y_k - y^r_k) + (u_{k-1} + \Delta u_k)^T R(u_{k-1} + \Delta u_k) + \gamma \bar{Z}^T_{k+1,k-N+2}\bar{P}\bar{Z}_{k+1,k-N+2}] \\ &= -[R + \gamma P_{11} + \gamma(\underline{G}^T_{k,11})P_{33}\underline{G}^T_{k,11} + \gamma P_{13}\underline{G}_{k,11} + \gamma(P_{13}\underline{G}_{k,11})^T]^{-1} \\ &\quad [[R + \gamma P_{11} + \gamma(\underline{G}_{k,11})^T P^T_{13}]u_{k-1} + \gamma[\underline{G}^T_{k,11}P_{33} + P_{13}]y_k \\ &\quad + \gamma[P_{12} + (\underline{G}_{k,11})^T P_{23}]\bar{u}_{k-1,k-N+1} + \gamma[P_{14} + (\underline{G}_{k,11})^T P_{34}]\bar{y}_{k,k-N+2} \\ &\quad + \gamma[(\underline{G}_{k,11})^T P_{33} + P_{13}]((\underline{G}_{k,12}\Delta u_{k-1,k-N+1} + \underline{F}_k\Delta y_{k,k-N+1})) \\ &\quad + \gamma[P_{15} + (\underline{G}_{k,11})^T P_{35}]y^r_k + \gamma[P_{15} + (\underline{G}_{k,11})^T P_{35}]\underline{F}^r_k\Delta y^r_{k,k-N+1})) + \gamma[P_{16} + (\underline{G}_{k,11})^T P_{36}]\bar{y}^r_{k,k-N+2} \end{aligned} \quad (3.104)$$

The VI algorithm for iADP using output feedback is given by the algorithm 8

Algorithm 8 VI for iADP using output feedback

Initialize a arbitrary control policy $\Delta u^0_k = \mu(\bar{Z}_{k,k-N+1})$
repeat
Value Update Step:
$\bar{Z}^T_{k,k-N+1}\bar{P}\bar{Z}_{k,k-N+1} = (y_k - y^r_k)^T Q(y_k - y^r_k) + (u_{k-1} + \Delta u_k)^T R(u_{k-1} + \Delta u_k) + \gamma \bar{Z}^T_{k+1,k-N+2}\bar{P}\bar{Z}_{k+1,k-N+2}$

Policy Improvement Step:
$$\begin{aligned} \Delta u_k &= -[R + \gamma P_{11} + \gamma(\underline{G}^T_{k,11})P_{33}\underline{G}^T_{k,11} + \gamma P_{13}\underline{G}_{k,11} + \gamma(P_{13}\underline{G}_{k,11})^T]^{-1} \\ &\quad [[R + \gamma P_{11} + \gamma(\underline{G}_{k,11})^T P^T_{13}]u_{k-1} + \gamma[\underline{G}^T_{k,11}P_{33} + P_{13}]y_k \\ &\quad + \gamma[P_{12} + (\underline{G}_{k,11})^T P_{23}]\bar{u}_{k-1,k-N+1} + \gamma[P_{14} + (\underline{G}_{k,11})^T P_{34}]\bar{y}_{k,k-N+2} \\ &\quad + \gamma[(\underline{G}_{k,11})^T P_{33} + P_{13}]((\underline{G}_{k,12}\Delta u_{k-1,k-N+1} + \underline{F}_k\Delta y_{k,k-N+1})) \\ &\quad + \gamma[P_{15} + (\underline{G}_{k,11})^T P_{35}]y^r_k + \gamma[P_{15} + (\underline{G}_{k,11})^T P_{35}]\underline{F}^r_k\Delta y^r_{k,k-N+1})) + \gamma[P_{16} + (\underline{G}_{k,11})^T P_{36}]\bar{y}^r_{k,k-N+2} \end{aligned}$$
until Convergence

4

iADP for Longitudinal Missile Control Design

In this chapter the proposed LADP and iADP algorithms from the previous chapters viz,. Full state feedback(FS), Output Feedback(OPFB) and Tracking problem will be implemented for a Longitudinal missile system to evaluate the performance characteristics. A model of the system is used in the FS to arrive at the optimal kernel matrix P and in the case of OPFB and Tracking problem the model is used to generate the input-output data useful for online learning. While FS and OPFB algorithms are designed to regulate the states of the system, Tracking algorithm is designed such that the output of the system is made to follow a reference signal.

4.1. Characteristics of the Longitudinal Missile System

The system under consideration is a second-order Nonlinear model of surface to air missile[17]. This model has the advantage of being nonlinear and simple as only the short period dynamics including the angle of attack(α) and pitch rate(q) are considered. For the implementation of LADP algorithms the model is linearized at the initial conditions.

The Nonlinear system is represented as

$$\dot{x} = f(x, \delta_e), \quad x = \begin{pmatrix} \alpha \\ q \end{pmatrix} \tag{4.1}$$

where the equations of motion of the missile are governed by 4.2

$$\begin{aligned} \dot{\alpha} &= q + \frac{\bar{q}S}{mV_t}[C_z(\alpha, M) + b_z(M)\delta_e] \\ \dot{q} &= \frac{\bar{q}Sd}{I_{yy}}[C_m(\alpha, M) + b_m(M)\delta_e] \end{aligned} \tag{4.2}$$

where δ_e is the elevator deflection and $m, V_t, I_{yy}, \bar{q}, S$ and d are mass, airspeed, pitching moment of inertia, dynamic pressure, reference area and reference length respectively. The aerodynamic coefficients are functions of M and α and are approximated such that the model is valid for a valid flight envelope of $-10° < \alpha < 10°$ and $1.8 < M < 2.6$.

For measuring angle of attack(α) vane measurement techniques are considered where the measurement of α is affected by the kinematic errors induced by the angular velocities q[20]. Thus α is set to be a combination of α and q as per the equation .

$$\alpha_{meas} = \begin{pmatrix} 0.9 & 0.1 \end{pmatrix} \begin{pmatrix} \alpha \\ q \end{pmatrix} \tag{4.3}$$

[0]This chapter has already been graded for AE4020.

Linearization of the system

The Nonlinear model of the system is linearized around the operating point by setting $\delta_e = \alpha = q = 0$ using the MatLab linearization tool. The identified approximate linear system is represented as 4.4

$$\begin{aligned} \begin{pmatrix} \dot{\alpha} \\ \dot{q} \end{pmatrix} &= \begin{pmatrix} -0.9118 & 1 \\ -172.3 & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ q \end{pmatrix} + \begin{pmatrix} -0.1957 \\ -241.1 \end{pmatrix} \delta_e \\ \alpha_{meas} &= \begin{pmatrix} 0.9 & 0.1 \end{pmatrix} \begin{pmatrix} \alpha \\ q \end{pmatrix} \end{aligned} \tag{4.4}$$

To open loop analysis of the linearized system is carried out and the results are summarized in the following graphs 4.1. The first figure from the left shows the imaginary poles in the pole zero plot. Frequency analysis of the system is carried out considering the bode plot. Frequency analysis provides necessary information to choose the frequencies for the persistent excitation(PE) vector used in OPFB algorithm. Finally the step response of the system can be visualized by the figure on the right.

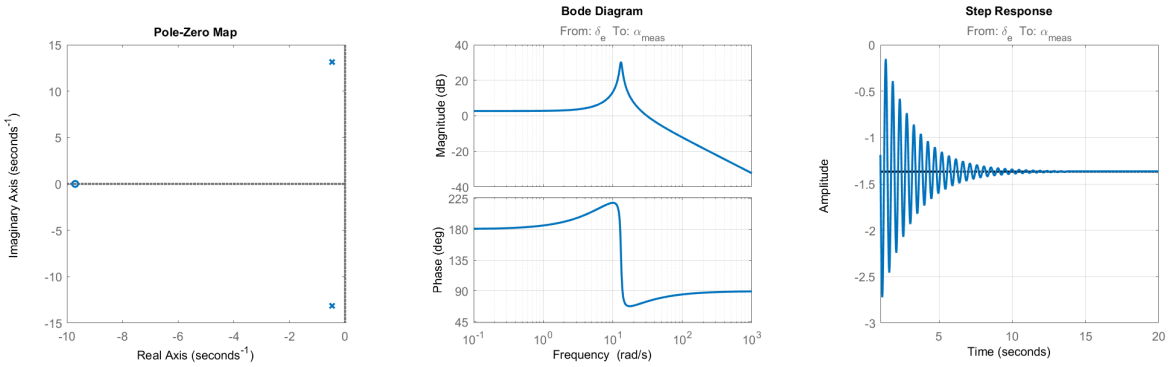


Figure 4.1: Open loop analysis of the Linearized reduced Model of the Longitudinal Missile System

The performance of different algorithms are analyzed by considering two different set of problems with different objectives. Firstly a regulation problem is considered where the objective is to regulate the states to a trimming value and the second one is a tracking problem where the objective is to make output track a reference. Both nonlinear and approximated linear models are considered.

4.2. LADP

Considering the Linearized system firstly a simple regulation problem is considered aimed at stabilizing the states of the system to a trimming value which is 0 in this case for all the states. When the full state information can be fed back considering the availability of the model of the system LADP-FS algorithm can be implemented and in the case of non availability of the full state information a model free LADP-OPFB approach is adopted where the knowledge of the system is indirectly constructed through the input-output measurements. The performance of the LADP-FS and LADP-OPFB at regulating the states can be analyzed through disturbance rejection characteristics of the system and in this case a nonlinear 1-cos disturbance at the elevator as shown in figure 4.2 is considered. To draw a comparison between different algorithms to evaluate the performance the following parameters 4.1 are kept constant for all the algorithms.

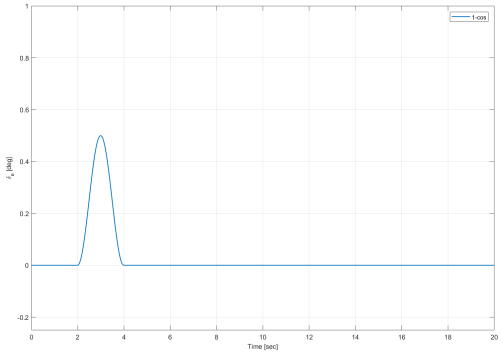


Figure 4.2: 1-Cos Disturbance

Parameter	Value
Q	$100I$
R	I
γ	0.35
Iterations	15
P_0	$0.01I$

Table 4.1: Tuning parameters

The performance of LADP-FS algorithm for regulating the states can be summarized from the figure 4.3. The kernel matrix P is initialized arbitrarily at $0.01I$. It can be observed that the values of P have converged after 4 iterations. The response of the system to the induced 1-cos disturbance is shown in the Angle of attack and Pitch rate plots. Once the LADP algorithm converges to the final policy system has better regulation characteristics as it is able to reject the disturbance at the elevator. As there is the availability of full state there is no need to have the persistent excitation as we are directly using the available model.

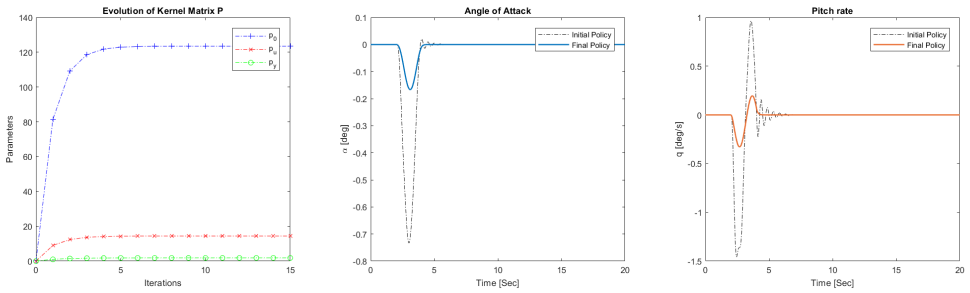


Figure 4.3: Implementation of LADP-FS for regulating the Linear Longitudinal Missile System

The performance of LADP-OPFB algorithm for regulating the states is shown in figure 4.4. The LADP-OPFB does not use the model of the system but instead uses the input-output measurements to arrive at the optimal policy. As the full state information is not available exploration of states is necessary for the algorithm to converge and this exploration is enabled through PE at the elevator. Tuning the frequency and amplitude of the PE signal is found to impact the convergence of the algorithm. It can be seen from 4.4 that the parameters of P kernel matrix initiated with $0.01I$ have converged after 5 iterations thus arriving at the optimal policy. The disturbance rejection characteristics of the system with 1-cos disturbance at the elevator is provided in the state plots where the difference in state response of the system between initial and final policy can be seen clearly. The optimal policy has better disturbance rejection properties compared to the initial policy which is achieved by regulating the states. Although OPFB algorithm has the advantage of being a model free method it is necessary to constantly excite the system through PE such that the input-output measurements contain rich information about the system by sufficient exploration of the states.

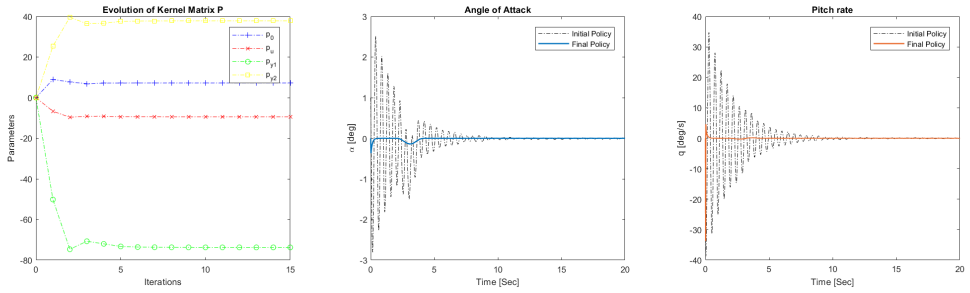


Figure 4.4: Implementation of LADP-OPFB for regulating the Linear Longitudinal Missile System

To draw a comparison between FS and OPFB algorithms and their disturbance rejection properties the state information is plotted for the final policy which is shown in figure 4.5. It is important to note that in case of LADP-FS a direct control of states is possible through defining weights in the Q matrix if the utility function

is defined to include states instead of the output where as in the case of LADP-OPFB states are regulated using output signal instead of state information and thus how output is measured from the states effects the performance of this algorithm. The difference between LADP-FS and LADP-OPFB is minimal however it can be observed that in case of LADP-OPFB algorithm the values of states are taking relatively high values at the beginning of an iteration as there aren't enough samples(input-output measurements) initially for the algorithm to construct the knowledge of the system. Even after convergence states seem to fluctuate in OPFB and this might be because of the lack of full state information in the samples.

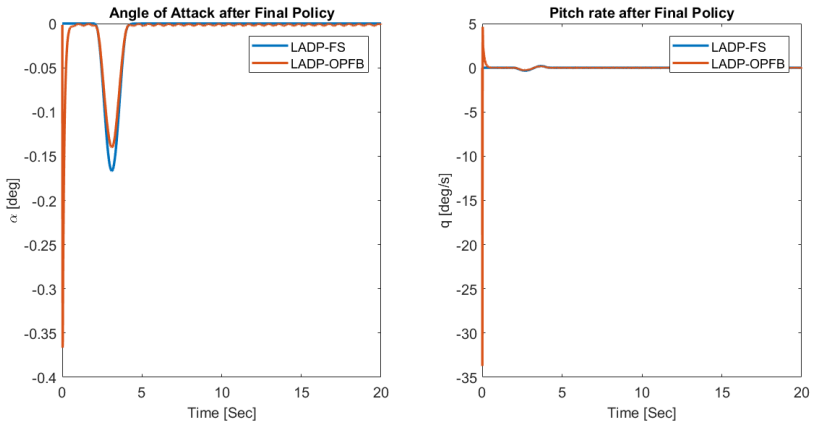


Figure 4.5: Comparison of LADP-FS and LADP-OPFB for regulating the Linear Longitudinal Missile System

While FS and OPFB algorithms are aimed at regulating the states the LADP-Tracking is aimed at solving a general tracking problem where the output of the system is made to track a reference signal. It is assumed that the full state information is not available and similar to OPFB the input-output data is used to arrive at the optimal policy using PE. The reference signal used for this case is a simple sine wave with a 4° amplitude and a frequency of $\frac{2\pi}{10}$ rad/s. The output signal α_{meas} is made to track the reference signal which is obtained using the equation 4.3. The tracking performance of the algorithm can be seen from the figure 4.6. The parameters of P have converged within a few iterations. It can be seen that α_{meas} at the final policy is able to track the reference signal. The pitch rate at the final policy is also plotted. In this algorithm the output of the system α_{meas} is made to track the reference signal while regulating the states. It can be observed that complete regulation of the states to zero might not be possible while making output track a signal in all scenarios as the states might be coupled, however the algorithm still tries to find a satisfactory result although the pitch rate is not completely regulated. Apart from tuning the parameters of PE signal weights in the Q matrix is also observed to be an important parameter to be tuned as it directly leads to the minimization in error between the reference and the output signal.

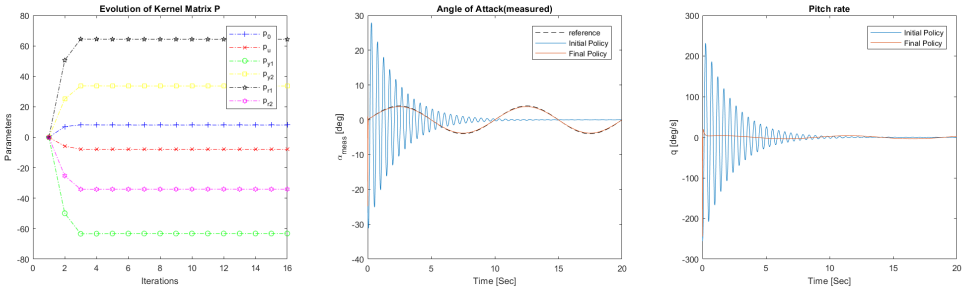


Figure 4.6: Implementation of LADP-Tracking for the Linear Longitudinal Missile System to follow a reference signal

4.3. iADP

In iADP we take advantage of incremental representation of the Nonlinear Missile model instead of the approximated linear model and implement FS and OPFB algorithms for regulation problem and Tracking algo-

rithm for reference tracking problem. The state information is defined using equations of motion according to 4.2.

The performance of the iADP-FS is summarized in the figure 4.7. The same 1-cos disturbance 4.2 was used to analyze the disturbance rejection characteristics of the FS and OPFB algorithms. It can be seen from the results that the parameters of the kernel matrix P have converged after 5 iterations. The optimal policy of the iADP-FS algorithm is clearly able to regulate the states which can be seen by drawing a comparison between the initial arbitrary policy and the final policy. The system adopting the final policy is also able to reject the 1-cos disturbance effectively. As it is assumed that the full state information is available a nonlinear model of the system is necessary for implementing iADP-FS.

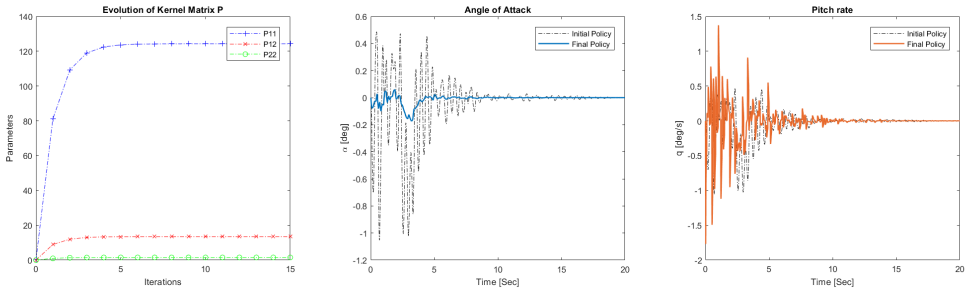


Figure 4.7: Implementation of iADP-FS for regulating the Non Linear Longitudinal Missile System

Considering the same regulation problem iADP-OPFB is implemented for the nonlinear missile system where instead of using the full state information input-output measurements are used to arrive at the optimal control policy for regulating the states and rejecting any input disturbance. The summary of the results of implementation of the iADP-OPFB algorithm is shown in figure 4.8. PE is necessary for sufficient exploration of the states. It can be observed that the initial policy is not able to regulate the states and cannot reject any input disturbance. However as the policy converges to an optimal one the system is better at rejecting the input disturbances while regulating the states.

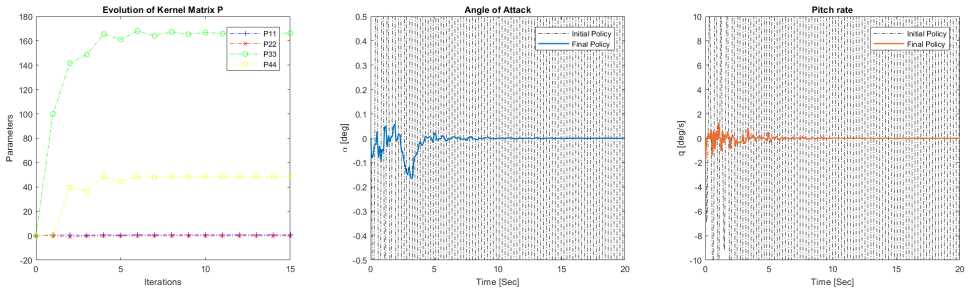


Figure 4.8: Implementation of iADP-OPFB for regulating the Non Linear Longitudinal Missile System

A comparison between iADP-FS and iADP-OPFB is shown in figure 4.9. The difference between FS and OPFB can be clearly seen in the case of iADP algorithms as the FS seems to reject the induced disturbance effectively compared to OPFB.

Now iADP method is implemented for tracking a reference signal in this case a simple sine signal. The iADP-Tracking algorithm uses the data from the input output measurements to arrive at the optimal policy where the output tracks a reference signal and a PE is added at the input for sufficient exploration of states. The tracking performance of iADP-Tracking is shown in figure 4.10.

4.4. iADP Vs LADP

To draw a comparison between iADP and LADP algorithms the performance of the final control policy are analyzed for different algorithms. The optimal policy obtained assuming a linear system is applied to the nonlinear system and are summarized as LADP in this section. This LADP performance is then compared with the results from iADP algorithms. The figure 4.11 shows the regulation and input disturbance rejection

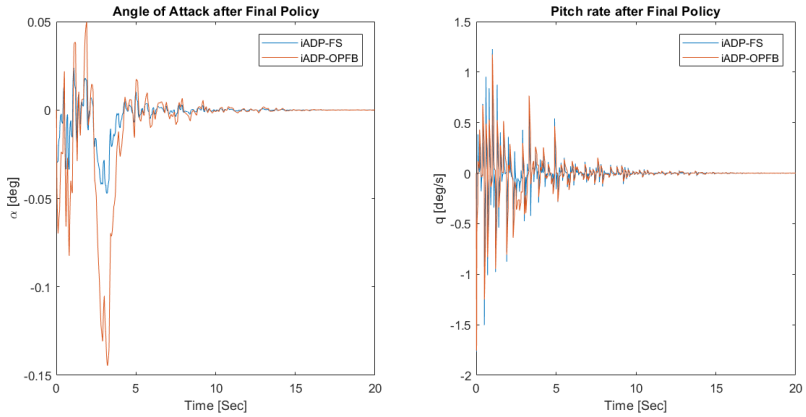


Figure 4.9: Comparison of iADP-FS and iADP-OPFB for regulating the Non Linear Longitudinal Missile System

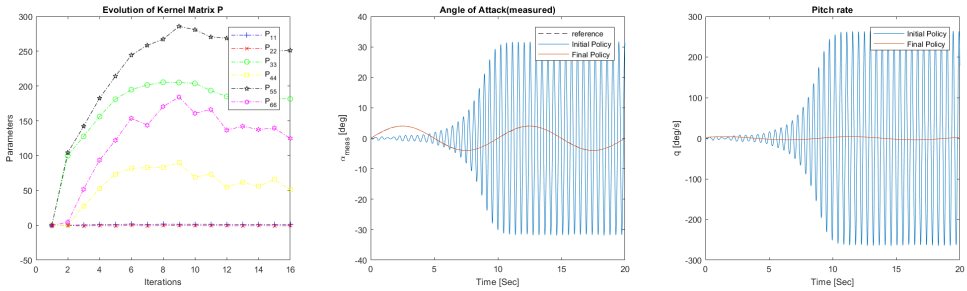


Figure 4.10: Implementation of iADP-Tracking on a Non Linear Longitudinal Missile System for tracking a reference signal

properties of the final control policy using LADP and iADP approaches respectively with full state feedback. Although the differences between LADP and iADP algorithms are observed to be minimal for this missile system as the systems become highly nonlinear where the linear approximation is the model is not valid anymore iADP is expected to yield better results using FS, OPFB or tracking algorithms.

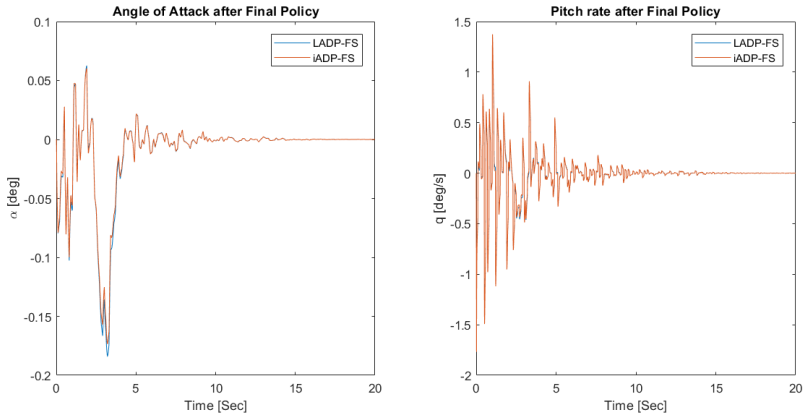


Figure 4.11: Comparison of LADP-FS and iADP-FS implemented on the Longitudinal Missile System

The figure 4.12 shows the regulation and input disturbance rejection properties of the final control policy using LADP and iADP approaches using a model free OPFB approach where input output measurements are used with PE. The difference between LADP and iADP is clearly visible as iADP is able to regulate states effectively.

Although both iADP and LADP approaches have similar performance at regulating states a better performance of iADP method can be expected when a highly nonlinear system is used where LADP methods fail to

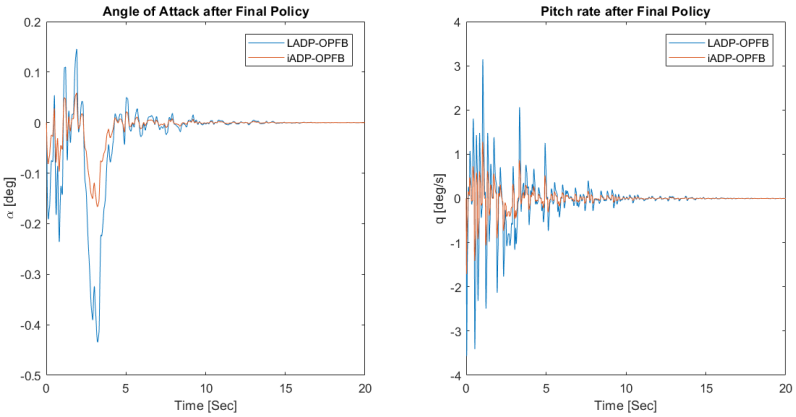


Figure 4.12: Comparison of LADP-OPFB and iADP-OPFB implemented on the Longitudinal Missile System

capture the nonlinear dynamics of the system. The figure 4.13 shows the tracking performance of the final control policy using LADP and iADP approaches. Note that the full state information is assumed to be not available and input-output measurements are used to arrive at the optimal tracking performance for both LADP and iADP.

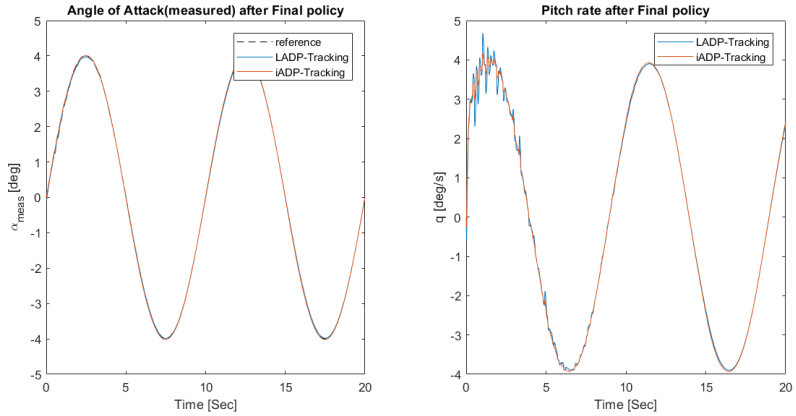


Figure 4.13: Comparison of LADP-Tracking and iADP-Tracking implemented on the Longitudinal Missile System

5

State of the art Review

This chapter presents a summary of the State of the Art review conducted which provides basis for the research project. As mentioned in section 1.2 the objective of this research is to address the practical issues of implementing an online reinforcement learning controller on Cessna Citation II aircraft. The objective is achieved in two stages viz,. designing an RL based FCS for Cessna Citation aircraft and then assessing controller performance in a close to realistic simulated environment. For designing an RL based FCS for Cessna Citation II a review on the State of the Art is conducted in the following areas:

5.1. Intelligent Flight Control

Intelligent Flight Control (IFC) is a broader term referring to flight control methods that can learn and adapt to changes in aircraft system or flying conditions. Researchers are exploring different IFC methods to make aircraft less dependent on human actions, improve performance of aircraft through learning from experience and to increase reliability and safety of flight[28]. The IFC methods are classified into different levels based on goals to be achieved[19] listed in increasing order of complexity viz,. (i) Robust controller for self improvement of tracking error (ii) Robust adaptive controller for self improvement of control parameters for better tracking error (iii) Robust optimal adaptive controller for self improvement of an estimate of some performance measure (iv) Robust optimal adaptive controller with additional self improvement of planning functions. These IFC methods can also be classified based on the methods[36] used to achieve the objective and some of the popular methods include Fuzzy algorithms, Neural Networks, Neuro fuzzy system, Genetic algorithms, Reinforcement learning etc,. Active research is going on application of these IFC methods to achieve different goals as mentioned before each approach having their own advantages and limitations. Machine Learning (ML) is one of the ways to achieve AI where instead of explicitly programming the computers for a given task, they are trained for the task at hand by gaining necessary intelligence through learning and some popular ML based methods from control system perspective are Artificial Neural networks (ANN)'s and RL. ANN's are inspired from biological neural networks which can learn from examples without being explicitly programmed to achieve certain task. In control system design, the application of ANN's can be classified into five categories viz,. supervised control systems, direct inverse control, back-propagation for minimizing/maximizing defined performance criterion, neuro adaptive control and adaptive critics[32]. RL on the other hand is a ML based method where an agent learns from an environment to achieve a goal by maximizing a certain reward. RL based control design is interesting due to its generalizable nature that can solve an optimization problem with real time learning capability whereas ANN's generally require an offline training and large amounts of data for convergence. Based on this, the current research is narrowed down to RL based flight control and thus the state of the Art in this field is considered.

5.2. Reinforcement Learning for Intelligent Flight Control

As discussed before RL is a goal oriented problem solving technique where an agent tries to interact with the environment and optimizes its actions depending on the state information such a certain goal is reached. Dynamic Programming (DP) is a popular technique used to solve the RL problem for optimality using Bellman Optimality equation[6]. DP methods are suitable for systems involving smaller state and action spaces where

the exact optimal value function and optimal policy are obtained[30]. For problems like flight control the state and action spaces are large or continuous which result in an exponential increase of memory size and computations with the states. This is referred to as "Curse of Dimensionality" [6]. In contrast the Approximate Dynamic Programming (ADP) methods use compact representations relying on function approximators so that approximate solutions can be obtained[9]. One of the popular methods for controlling dynamic systems using ADP methods is the design of optimal feedback controllers with reward expressed as a quadratic cost function[22]. The use of quadratic cost function is inspired from LQR problem where the controller is optimized for regulating the states with minimal control effort. An advantage of using quadratic cost function representation for reward is that the optimal policy is linear in state, for a system with transition dynamics linear in state and action variables [9]. The methodology of designing RL based controller for solving LQR problem when full state information of the system is provided and also for the partially observable systems where optimal feedback controller is designed using input output data[21]. The ADP method is extended to deal with Linear Quadratic Tracking (LQT) problem where the state is augmented to include the reference trajectory dynamics[18]. Although these algorithms can deal with tracking/regulation problem without a priory knowledge of the system dynamics, they are suitable for Linear-Time-Invariant (LTI) systems. An incremental model is a time varying linear approximation of the nonlinear system and this incremental model based ADP method is proposed suitable for nonlinear time varying systems for regulation tasks[37]. These are referred to as Incremental Approximate Dynamic Programming (iADP) methods. iADP is extended for tracking control considering full state feedback measurements and using input/output data and the method is implemented on a F-16 aircraft model[11].

5.3. iADP for Flight Control

For this project iADP is considered for FCS design on Cessna Citation II due to the following properties : (i) Online learning capability with online incremental model identification (ii) Applicability for Nonlinear aerospace systems (iii) Quadratic value function approximation making policy evaluation step tractable. (iv) Feedback control design using output measurements when full state measurements are not available.

It is also important to consider some of the limitations/assumptions of the iADP controller to avoid any confounds during the controller design for Cessna Citation II. Some of the observations/remarks made during previous implementations are : (i) The incremental model is identified with the assumption of First order Taylor series expansion. This assumption is not valid for higher sampling times and models which are highly nonlinear. (ii) The system has to be observable(for output feedback algorithm) and controllable. (iii) The input should be persistently excited which is necessary for incremental model identification and also for state space exploration necessary for RL algorithm.

5.4. Flight Control System Design for Cessna Citation II

As the primary of the research project is to design FCS for Cessna Citation II, a review on the Cessna Citation II aircraft and previous FCS design approaches is conducted.

The DASMAT package provides a nonlinear simulation model of the Cessna Citation II suitable for FCS design and evaluation[10]. A new high-fidelity model of the Citation II is identified using flight test data[15]. The flight testing of manual nonlinear flight control laws was conducted on ATTAS and Pseudo Control Hedging (PCH) was integrated within pilot command filtering and reference model to deal with actuator saturation[23]. INDI controller is developed for Cessna Citation II aircraft and was successfully flight tested considering slower actuator dynamics and delays due to sensor and flight control system[14]. Incremental Backstepping (IBS) control laws are flight tested on Citation using angular accelerometer feedback and a first order model for the actuators[16]. Incremental control laws are used to command current instead of control surface position and have been successfully integrated into the FCS architecture using a high fidelity actuator model[25].

For this research project a high fidelity citation II model based on DASMAT package is used for FCS design and to perform necessary simulations. The FCL is designed using manual outer loop control which includes pilot command filtering and reference module further PCH is not considered for this research.

6

Conclusions and Recommendations

This chapter presents the final conclusions drawn from this research project and recommendations for future work. The conclusions section provides an overview of how the research objective is achieved by answering the posed research questions. The recommendations and guidelines for future research is summarized in future work section.

6.1. Conclusions

The objective of this research is *to address the practical issues of implementing an online reinforcement learning controller on a fixed wing Cessna Citation II aircraft by investigating the controller performance in a close to realistic simulated environment of Cessna Citation II platform and proposing possible solutions to ensure desired controller performance*. This research objective is motivated by the need to design an online learning controller that is resilient to faults/failures, thus enhancing the safety of aircraft.

The research objective is addressed by formulating a set of research questions which are answered through this research. The summary of answers to these research questions are as follows. The research questions 1a and 1b are answered through literature review conducted which is summarized in Chapters 3, 4 and 5. An iADP controller is found to be a suitable RL method that can be implemented on Cessna Citation II aircraft. This controller can achieve optimal flight control by interacting with the environment and using the data collected along the system trajectories to minimize a cost-to-go function. This controller does not require a priory information of the system, instead a locally linearized incremental model necessary for the ADP algorithm is identified online. Further iADP uses a simple quadratic cost function which is tractable reducing the learning complexity. The incremental nature involved makes this controller suitable for nonlinear systems like Citation II aircraft with online learning capability. Two iADP algorithms are considered for this research viz,. full state feedback and output feedback for achieving tracking control. A preliminary analysis is done by implementing these algorithms on a longitudinal missile system which provided some insights into the characteristics and limitations of these algorithms. It is observed that full state feedback algorithm is able to achieve optimal policy in fewer iterations compared to output feedback algorithm. Further the system should be persistently excited which is necessary for incremental model identification and exploration necessary for RL algorithm.

Research question 1c is answered in Chapters 2 and 5. Based on previous FCS design for Citation II aircraft, a control law is designed integrating manual outer loop and an automatic inner rate control loop. The manual outer loop includes pilot command filtering and command module which provides setpoints to track for the automatic inner loop. The iADP controller is designed to achieve automatic online rate control to track these references. The simulations results show that the combined rate control using full state feedback is able to track reference signals without any offline training.

Research questions 2a, 2b, 2c and 2d are answered in the scientific paper. To investigate controller performance in a close to realistic simulated environment the effect of sensor and actuator dynamics viz,. discretization effects, sensor bias, sensor noise, signal delay and transport delay are considered. The criterion chosen to assess the controller performance are RMSE, maximum actuator deflections and rates, model coefficients after convergence. The simulation results from iADP longitudinal control design using full state feedback indicate that the discretization of sensor signals, sensor bias and transport delays did not have any

significant effect on the controller performance or on the incremental model identification while noisy signals and delays in sensors are found to effect the controller performance. The noisy signals resulted in incorrect estimation of incremental model parameters affecting the controller performance. It is observed that appropriate filtering of signals resulted in better estimation of the incremental model subsequently improving the controller performance. Sensor delays also resulted in incorrect estimation of model parameters, however the controller is found to track the reference inspite of the incorrect incremental model parameters but with reduced tracking performance. Finally an iADP controller using output feedback is designed to achieve longitudinal control in the absence of full state information. The controller is trained offline due to higher learning complexity and performance is evaluated in the presence of sensor and actuator dynamics. The results from output feedback method show the controller can achieve satisfactory tracking control but with reduced tracking performance.

Through this research it is shown that RL based iADP controller can be used to design FCS for Cessna Citation II fixed wing aircraft. As compared to conventional control methods, iADP controller can achieve optimal control using just the data observed along the trajectories, making it a model free adaptive controller. This online adaptive controller is a feasible alternative that can adapt to unforeseen conditions, thus improving the safety of the aircraft.

6.2. Future Work

Through this research project it is observed that the iADP controller can be used to design FCS for a fixed wing Cessna Citation II aircraft and an investigation of effect of sensor and actuator dynamics on controller performance is carried out. However some challenges has to be addressed before the FCS can be validated through flight tests.

It is observed that sensor noise and delays led to inaccurate online incremental identification. Although the effect of sensor noise is mitigated using filtered signals, another approach would be to use RLS algorithms that can identify incremental model in the presence of noisy signals viz,. regularization, bias compensated RLS. Regarding sensor delays, sensors with smaller delays can be used on PH-Lab like IMU sensors for measuring angular rates. It is observed that synchronization can mitigate effects of sensor delays during previous flight tests of incremental controllers. The effects of synchronization on iADP controller can be investigated.

During implementation it is observed that persistent excitation is necessary for both incremental identification and RL exploration. In this research a white noise is added as PE. Further study can be conducted on tuning the PE signals or using alternative signals for PE like sum of sinusoids.

The controller performance can be improved by including integral and derivative terms in the IADP control loop which provides control over design criterion viz, settling time, overshoot, rise time. During the research controller is manually tuned by adjusting the weight matrices. This controller tuning can be done using optimization algorithms like Multi Objective Parameter Synthesis(MOPS) such that controller meets the design criterion. Before flight testing the controller is checked for robustness against uncertainties and model mismatches. The robustness of IADP against these uncertainties through parameter tuning can be done in future.

A

Additional Results

This chapter presents additional findings that complement the results in the scientific paper(Chapter 2). It is observed that sensor noise and delays are found to degrade the controller performance from iADP implementation for longitudinal control for Citation II aircraft using full state feedback. The following results provides insights into contribution of individual sensor signals to the controller performance degradation.

A.1. Effect of sensor noise

To study the effect of sensor noise, white Gaussian noise is introduced at individual sensor signals and controller performance is evaluated as shown in Fig. A.1. Recalling states, outputs, reference and control vectors used by the iADP controller are

$$x = \begin{bmatrix} q & \alpha \end{bmatrix}^T, u = \delta_e, y = \dot{\theta}, y_r = \dot{\theta}_{ref}$$

Noise at elevator control input is observed to degrade controller performance while the effect of noisy q,α and $\dot{\theta}$ sensor signals is negligible.

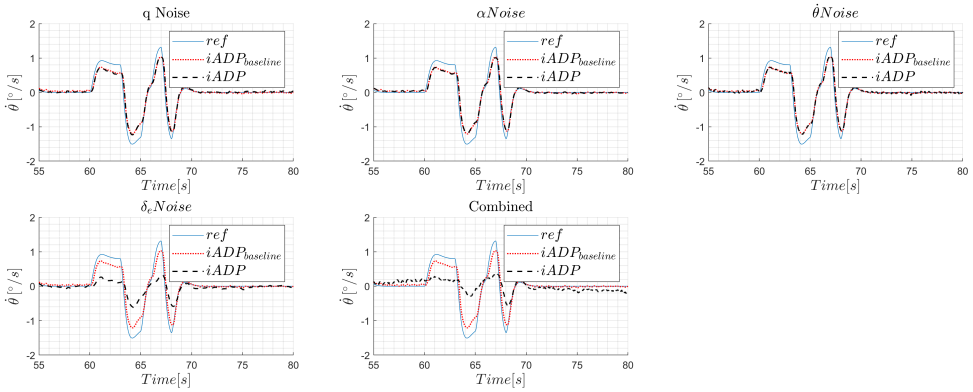


Figure A.1: Effect of sensor noise on tracking performance

A.2. Effect of sensor delay

The contribution of delays in sensor signals to control performance degradation is shown in Fig. A.2. As there is no delay at elevator control input, it is considered in this analysis. Delays in q and $\dot{\theta}$ is found to degrade controller performance while delay in α has negligible effect on tracking performance. This might be due to direct dynamical relation between control surface deflection and angular rate.

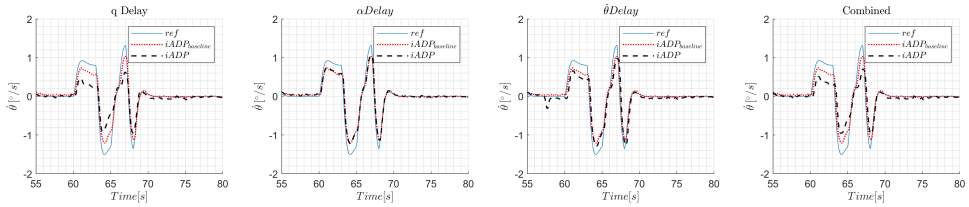


Figure A.2: Effect of sensor delay on tracking performance

Bibliography

[1] International Air Transport Association. https://www.iata.org/pressroom/pr/Pages/2018-10-24-02.aspx, 2018. Accessed: 2019-11-28.

[2] International Civil Aviation Organization Accident Statistics. https://www.icao.int/safety/iStars/Pages/Accident-Statistics.aspx, 2019. Accessed: 2019-11-28.

[3] International Civil Aviation Organization, Civil Aviation Statistics of the World and ICAO staff estimates. https://data.worldbank.org/indicator/IS.AIR.PSGR, 2019. Accessed: 2019-11-28.

[4] Paul Acquatella, Erik-Jan Van Kampen, and Q. Chu. Incremental backstepping for robust nonlinear flight control. *EuroGNC 2013, 2nd CEAS Specialist Conference on Guidance, Navigation Control*, 04 2013.

[5] Christine M. Belcastro, John V. Foster, Richard L. Newman, Loren Groff, Dennis A. Crider, and David H. Klyde. *Aircraft Loss of Control: Problem Analysis for the Development and Validation of Technology Solutions*. 2016. doi: 10.2514/6.2016-0092. URL https://arc.aiaa.org/doi/abs/10.2514/6.2016-0092.

[6] Richard Bellman. The theory of dynamic programming. *Bull. Amer. Math. Soc.*, 60(6):503–515, 11 1954. URL https://projecteuclid.org:443/euclid.bams/1183519147.

[7] Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-dynamic programming*. Athena Scientific, 1999.

[8] D.P. Bertsekas. *Dynamic Programming and Optimal Control*. Number Bd. 2 in Athena Scientific optimization and computation series. Athena Scientific, 2012. ISBN 9781886529441. URL https://books.google.de/books?id=H-PSMwEACAAJ.

[9] Lucian Busoniu, Robert Babuska, Bart De Schutter, and Damien Ernst. *Reinforcement Learning and Dynamic Programming Using Function Approximators*. CRC Press, Inc., Boca Raton, FL, USA, 1st edition, 2010. ISBN 1439821089, 9781439821084.

[10] Van der Linden. Dasmat-delft university aircraft simulation model and analysis tool. 1996.

[11] Pedro Miguel Dias, Ye Zhou, and Erik-Jan Van Kampen. Intelligent nonlinear adaptive flight control using incremental approximate dynamic programming. *AIAA Scitech 2019 Forum*, 2019. doi: 10.2514/6.2019-2339. URL https://arc.aiaa.org/doi/abs/10.2514/6.2019-2339.

[12] DALE ENNS, DAN BUGAJSKI, RUSS HENDRICK, and GUNTER STEIN. Dynamic inversion: an evolving methodology for flight control design. *International Journal of Control*, 59(1):71–91, 1994. doi: 10.1080/00207179408923070. URL https://doi.org/10.1080/00207179408923070.

[13] Andrew G. Barto, Richard Sutton, and Charles Anderson. Neuron like elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man And Cybernetics*, 13:834–846, 01 1970.

[14] Fabian Grondman, Gertjan Looye, Richard O. Kuchar, Q Ping Chu, and Erik-Jan Van Kampen. Design and flight testing of incremental nonlinear dynamic inversion-based control laws for a passenger aircraft. *2018 AIAA Guidance, Navigation, and Control Conference*, 2018. doi: 10.2514/6.2018-0385. URL https://arc.aiaa.org/doi/abs/10.2514/6.2018-0385.

[15] M. Hoek, Coen De Visser, and Daan Pool. *Identification of a Cessna Citation II Model Based on Flight Test Data*, pages 259–277. 01 2018. ISBN 978-3-319-65282-5. doi: 10.1007/978-3-319-65283-2_14.

[16] Twan Keijzer, Gertjan Looye, Q Ping Chu, and Erik-Jan Van Kampen. Design and flight testing of incremental backstepping based control laws with angular accelerometer feedback. *AIAA Scitech 2019 Forum*, 2019. doi: 10.2514/6.2019-0129. URL https://arc.aiaa.org/doi/abs/10.2514/6.2019-0129.

[17] Seung-Hwan Kim, Yoon-Sik Kim, and Chanho Song. A robust adaptive nonlinear control approach to missile autopilot design. *Control Engineering Practice*, 12(2):149 – 154, 2004. ISSN 0967-0661. doi: https://doi.org/10.1016/S0967-0661(03)00016-9. URL http://www.sciencedirect.com/science/article/pii/S0967066103000169.

[18] B. Kiumarsi, F. L. Lewis, M. Naghibi-Sistani, and A. Karimpour. Optimal tracking control of unknown discrete-time linear systems using input-output measured data. *IEEE Transactions on Cybernetics*, 45 (12):2770–2779, Dec 2015. ISSN 2168-2275. doi: 10.1109/TCYB.2014.2384016.

[19] K. Krishnakumar and Karen Gundy-Burlet. Intelligent control approaches for aircraft applications. 05 2002.

[20] M Laban and J.A. Mulder. On-line identification of aircraft aerodynamic model parameters. *IFAC Proceedings Volumes*, 25:199–204, 07 1992. doi: 10.1016/S1474-6670(17)50633-3.

[21] F. L. Lewis and K. G. Vamvoudakis. Reinforcement learning for partially observable dynamic processes: Adaptive dynamic programming using measured output data. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 41(1):14–25, Feb 2011. ISSN 1941-0492. doi: 10.1109/TSMCB.2010.2043839.

[22] F.L. Lewis and Draguna Vrabie. Reinforcement learning and adaptive dynamic programming for feedback control. *Circuits and Systems Magazine, IEEE*, 9:32 – 50, 01 2009. doi: 10.1109/MCAS.2009.933854.

[23] T. Lombaerts and G. Looye. *Design and flight testing of nonlinear autoflight control laws*. 2012. doi: 10.2514/6.2012-4982. URL https://arc.aiaa.org/doi/abs/10.2514/6.2012-4982.

[24] Tijmen Pollack, Gertjan Looye, and Frans Van der Linden. Design and flight testing of flight control laws integrating incremental nonlinear dynamic inversion and servo current control. *AIAA Scitech 2019 Forum*, 2019. doi: 10.2514/6.2019-0130. URL https://arc.aiaa.org/doi/abs/10.2514/6.2019-0130.

[25] Tijmen Pollack, Gertjan Looye, and Frans Linden. Design and flight testing of flight control laws integrating incremental nonlinear dynamic inversion and servo current control. 01 2019. doi: 10.2514/6.2019-0130.

[26] S. Sieberling, Q. P. Chu, and J. A. Mulder. Robust flight control using incremental nonlinear dynamic inversion and angular acceleration prediction. *Journal of Guidance, Control, and Dynamics*, 33(6):1732–1742, 2010. doi: 10.2514/1.49978. URL https://doi.org/10.2514/1.49978.

[27] L. Sonneveldt, Q. Chu, and J.A. Mulder. Adaptive backstepping flight control for modern fighter aircraft. 04 2011. doi: 10.5772/13839.

[28] Robert F Stengel. Toward Intelligent Flight Control. *IEEE Transactions on Systems, Man, and Cybernetics*, 23(6):1699–1717, 1993.

[29] Richard S. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1): 9–44, 1988. doi: 10.1007/bf00115009.

[30] Richard S. Sutton and Andrew Barto. *Reinforcement learning: an introduction*. The MIT Press, 2018.

[31] Christopher Watkins. Learning from delayed rewards. 01 1989.

[32] P. J. Werbos. Neural networks for control and system identification. *Proceedings of the 28th IEEE Conference on Decision and Control,*, pages 260–265 vol.1, Dec 1989. ISSN null. doi: 10.1109/CDC.1989.70114.

[33] Paul Werbos. Building and understanding adaptive systems: A statistical/numerical approach to factory automation and brain research. *Systems, Man and Cybernetics, IEEE Transactions on*, 17:7 – 20, 02 1987. doi: 10.1109/TSMC.1987.289329.

[34] Paul Werbos. The roots of backpropagation: From ordered derivatives to neural networks and political forecasting. 01 1994.

[35] Ian Witten. An adaptive optimal controller for discrete-time markov environments. *Information and Control*, 34:286–295, 08 1977. doi: 10.1016/S0019-9958(77)90354-0.

[36] Wen Yu. *Recent Advances in Intelligent Control Systems*. 01 2009. doi: 10.1007/978-1-84882-548-2.

[37] Ye Zhou, Erik-Jan Van Kampen, and Q. Chu. Nonlinear adaptive flight control using incremental approximate dynamic programming and output feedback. *Journal of Guidance, Control, and Dynamics*, 40:1–8, 11 2016. doi: 10.2514/1.G001762.

[38] Ye Zhou, Erik-Jan Van Kampen, and Q. Chu. Incremental model based heuristic dynamic programming for nonlinear adaptive flight control. 10 2016.

[39] Ye Zhou, Erik-Jan van Kampen, and Qi Ping Chu. Incremental model based online dual heuristic programming for nonlinear adaptive control. *Control Engineering Practice*, 73:13 – 25, 2018. ISSN 0967-0661. doi: https://doi.org/10.1016/j.conengprac.2017.12.011. URL http://www.sciencedirect.com/science/article/pii/S096706611730285X.