Coordinated Wide-Area Control of Multiple Controllers in a Modern Power System

by

Pooja Gupta

A Dissertation Presented in Partial Fulfillment
of the Requirement for the Degree
Doctor of Philosophy

Approved April 2021 by the
Graduate Supervisory Committee:

Anamitra Pal, Co-Chair
Vijay Vittal, Co-Chair
Junshan Zhang
Mojdeh Khorsand Hedman
Meng Wu

ARIZONA STATE UNIVERSITY

May 2021

ABSTRACT

Low frequency oscillations (LFOs) are recognized as one of the most challenging problems in electric grids as they limit power transfer capability and can result in system instability. In recent years, the deployment of phasor measurement units (PMUs) has increased the accessibility to time-synchronized wide-area measurements, which has, in turn, enabled the effective detection and control of the oscillatory modes of the power system. This work assesses the stability improvements that can be achieved through the coordinated wide-area control of power system stabilizers (PSSs), static VAr compensators (SVCs), and supplementary damping controllers (SDCs) of high voltage DC (HVDC) lines, for damping electromechanical oscillations in a modern power system.

The improved damping is achieved by designing different types of coordinated wide-area damping controllers (CWADC) that employ partial state-feedback. The first design methodology uses a linear matrix inequality (LMI)-based mixed H_2/H_∞ control that is robust for multiple operating scenarios. To counteract the negative impact of communication failure or missing PMU measurements on the designed control, a scheme to identify the alternate set of feedback signals is proposed. Additionally, the impact of delays on the performance of the control design is investigated.

The second approach is motivated by the increasing popularity of artificial intelligence (AI) in enhancing the performance of interconnected power systems. Two different wide-area coordinated control schemes are developed using deep neural networks (DNNs) and deep reinforcement learning (DRL), while accounting for the uncertainties present in the power system. The DNN-CWADC learns to make control decisions using supervised learning; the training dataset consisting of polytopic controllers designed with the help of LMI-based mixed H_2/H_∞ optimization. The DRL-CWADC learns to adapt to the system uncertainties based on its continuous interaction with the power system environment by employing an advanced version of the state-of-the-art deep deterministic policy gradient (DDPG)

algorithm referred to as bounded exploratory control-based DDPG (BEC-DDPG).

The studies performed on a 29 machine, 127 bus equivalent model of the Western Electricity Coordinating Council (WECC) system-embedded with different types of damping controls have demonstrated the effectiveness and robustness of the proposed CWADCs.

ACKNOWLEDGMENTS

There are many who helped me along the way on this journey. I want to take a moment to thank them.

First, I would like to express my gratitude to my supervisors Dr. Anamitra Pal and Dr. Vijay Vittal for their guidance, advice and encouragement throughout this research. Their insightful feedback pushed me to sharpen my thinking and brought my work to a higher level. I also greatly appreciate all my committee members, Dr. Junshan Zheng, Dr. Mojdeh Khorsand Hedman, and Dr. Meng Wu for their valuable suggestions and feedback to help me reach my goal.

I would like to acknowledge Dr. Anamitra Pal for providing me with an opportunity to join his team.

To my fellow lab mates, Reetam Sen Biswas and Sameer Nekkalapu: for a cherished time spent together in the lab.

To my parents, Anju and Neeraj; my siblings, Khushboo, Rajat, and Shubham; and my mother-in-law, Meera: for providing me with unfailing support and continuous encouragement throughout my years of study.

To my friend, Gaurav: for providing happy distractions to rest my mind outside of my research.

Most importantly, to my beloved husband, Pankul: for his constant love and understanding that helped me to get through the tough times. Thank you so much for supporting me in everything.

TABLE OF CONTENTS

CHAPTER Page

CHAPTER Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

With the advent of deregulated energy markets, the requirements for energy exchanges have increased resulting in higher loading of transmission systems. Furthermore, the increased penetration of renewable energy sources (RES) and decommissioning of large thermal units have led to increased power flows over long distances and lower inertia in the system [1], [2]. These developments can create additional challenges in damping low frequency inter-area oscillations [3], which are an inherent phenomena of the power system [4]. In [5], it has been shown that these underlying oscillations often become poorly damped during heavy power transfer between different areas connected by weak tie-lines, thus reducing the security margins. As complex conditions evolve within power systems, these oscillations can grow in magnitude and become unstable, which can cause system breakup and eventually lead to a large-scale blackout.

1.1 History of Past Events

One of the most prominent events which occurred as a consequence of unstable oscillations is the splitting of the Western Electricity Coordinating Council (WECC) in the USA into four islanded regions on August 10, 1996, resulting in a loss of 30 GW of load affecting 7.5 million customers [6]. The measurement of line power flow through one of

the three AC lines in the California-Oregon AC Intertie (COI) indicated the presence of an inter-area oscillatory mode as shown in Fig. 1.1, which became highly unstable and led to the tripping of all the three lines that make up the COI [7]. At this point, system collapse became unavoidable.

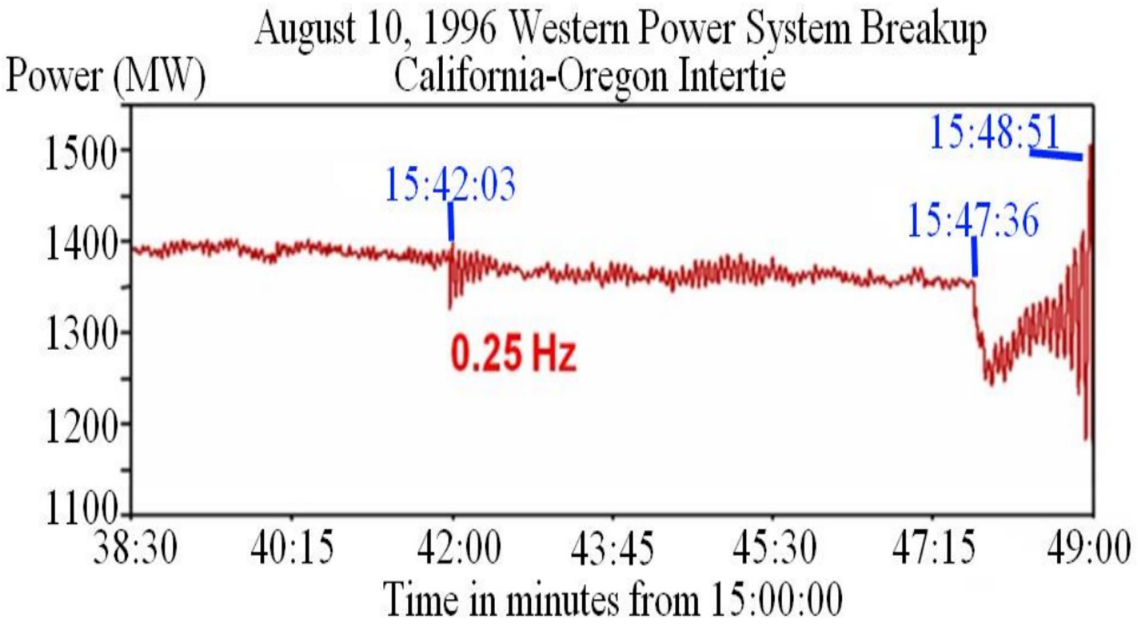

Figure 1.1. Power Flow Through One of the Three AC Lines in COI Leading to System Breakup.

Some of the other noteworthy incidents related to the low frequency oscillations (LFOs) include:

1. The integration of Turkish Grid into the European Network of Transmission System Operators for Electricity (ENSTO-E) triggering inter-area oscillations in the range of 0.15 Hz [8].

2. On July 31^{st}, 2008, within the Mexican Interconnected System, a combination of system events involving operational changes and a weak transmission system caused wide-spread inter-area oscillations [9].

3. The interconnection between Columbia and Venezuela power systems through a single 230 kV transmission line led to sustained power flow oscillation of 0.22 Hz [10].

4. Presence of multiple inter-area oscillation modes in WECC [11]:

    - North-South Mode A(~ 0.25 Hz) dominantly observable in the Alberta Canada area of WECC
    - North-South Mode B(~ 0.4 Hz) with more widespread observability than North-South Mode A
    - Montana Mode (~ 0.55 Hz)
    - British Columbia Mode (~ 0.62 Hz)

To securely transmit the larger amounts of power over long distances, the stabilization of these electromechanical oscillation modes is extremely important. This has led to an increased interest in control schemes that can increase the transmission capability for the systems restricted by oscillatory instability of these modes.

1.2 Previous Research on Damping of Oscillations

Traditionally power system oscillation damping is provided by power system stabilizers (PSSs) installed at generator units [12], [13], [14], [15], [16]. However, PSSs have to be carefully coordinated to damp both local plant modes and inter-area modes. This coordinated tuning is often done using optimization approaches, with genetic algorithms and particle swarm optimization, proving to be popular and effective [17], [18], [19]. Some of the alternative approaches such as use of linear matrix inequalities (LMIs) [20], discrete linear quadratic regulator (DLQR) [21], and linear parameter varying (LPV) systems [22], [23] have also been explored. There has also been extensive research into the use of supple-

mentary damping controllers (SDCs) installed either individually or in combination with [24], [25], [26], [27]:

- series-connected flexible alternating current transmission system (FACTS) devices, such as thyristor controlled series capacitors (TCSCs).
- shunt-connected FACTS device, such as static VAr compensators (SVCs).

Furthermore, in an attempt to improve the performance and robustness, the coordination of SDC action of FACTS devices with that of existing PSSs, is demonstrated in [28], [29], and [30].

In recent years, investigation of the impact of high voltage DC (HVDC) systems on small-disturbance stability has gained considerable interest. Controlled modulation of the active power flow through the line commutated converter (LCC) and voltage source converter (VSC) based HVDC lines has been proposed to improve the capacity of AC transmission lines [31], [32], [33]. However, the possibility of negative interactions of HVDC controllers with the existing controls is usually ignored.

Following the development of phasor measurement units (PMUs), there have been persistent efforts in the development of wide-area damping controllers which employ wide-area signals for damping inter-area oscillation modes. The idea of using a global signal to design a damping controller for PSSs was described in [34]. Later, a multi-agent based supervisory level controller for PSSs using wide-area as well as local measurements was proposed in [35]. The use of PSSs for damping inter-area oscillations based on remote measurements has been explored in [36] and [37]. Similarly, the implementation of FACTS devices using the feedback from remote measurements was investigated in [38] and [39]. An actual application of wide-area measurements to damp inter-area oscillations via modulation of active power in the Pacific DC Intertie (PDCI) in WECC was presented in [40].

In [41] and [42], synthesis of a polytopic controller having mixed feedback control

was proposed using LMIs. A multi-polytope based adaptive control was implemented in [43] and [44], where the selection of a suitable polytopic controller was made based on the current operating point of the system. In [45], a systematic procedure for designing a centralized wide-area control was detailed. In [46], [47], [48], [49], [50], wide-area coordinated controls of energy storage devices (ESDs), DC lines, and FACTS devices were developed to provide robust damping to the oscillatory modes. The capability of doubly-fed induction generators (DFIGs) based wide-area controls in improving the damping of inter-area modes was investigated in [51] and [52].

In recent years, the comprehensive analyses of delays in PMU signals and mitigation techniques for wide-area measurement systems (WAMS) based damping controllers have received increasing interest. The effects of time delay uncertainty on the closed-loop stability of power systems incorporating an SVC and HVDC based supplementary control were examined in [39] and [49]. The techniques proposed in [23], [40], [53], [54], [55], [56], and [57] have focused on improving the robustness of controllers to control signal delays, including both fixed [40], [53], [54] and time-varying delays [23], [53], [55], [56], [57]. In [58], a method to calculate damping factor based delay margins, which constructs the correlations between signal transmission delays and damping factors of inter-area oscillation modes, was presented. The impact of the performance of wide-area signals, latency, and data dropouts in PMU signals on the performance of DFIG-based controls was studied in [59] and [60]. Motivated by the installation of distributed energy resources (DERs), reference [61] utilizes their active power modulation capability to damp the inter-area modes and transient frequency swing. The method is shown to provide an advantage of exerting lesser control effort. Another solution for reducing the number of control signals is suggested in [62], where the online grouping of the coherent generators is considered for identifying the most controllable machines in each group.

1.3 Neural Networks

The recent breakthroughs in artificial intelligence (AI) provide promising approaches to design advanced control schemes for enhancing power system stability. One such approach is the artificial neural networks (ANNs), which can be thought of as non-linear approximators that provide excellent generalization across varying system parameters, superior noise rejection, and faster execution of control actions (most of the computations occur during the off-line training). In the past few years, significant progress has been made in using ANNs to solve a broad range of computational problems both in dynamics and optimization. In [63] and [64], authors have proposed a multi-layered perceptron capable of generalizing previously unencounterd load levels and system topologies, and accurately estimate the critical clearing time (CCT). In [65], ANN is used for predicting and mitigating transient instabilities. A deep learning (DL) based feature extraction framework is proposed for system security assessment in [66]. Reference [67] demonstrates the use of NNs for performing contingency screening and dynamic security ranking. Studies were conducted on two large scale systems of B.C. Hydro and Hydro Quebec. Researchers in [68] have used real data from power systems in Singapore and Australia to perform short-term load and wind power forecasting using NN-based prediction intervals. An overview of different NN applications in power systems such as load forecasting, fault diagnosis, transient stability, security assessment, and economic dispatch are listed in [69] and [70]. Application of NN for non-linear control of DFIGs was demonstrated in [71].

Reference [72] described the use of ANN for optimizing the load-oriented control parameters of static synchronous compensators for damping LFOs. The NN-based model predictive control (NN-MPC) and adaptive neuro-fuzzy inference system are developed for designing and tuning the parameters of PSS in [73] and [74], respectively. The work in [75] and [76] propose the adaptive NN decentralized controllers for damping the oscillations.

An NN-approximated transient energy function (TEF)-based SDC for unified power flow controllers (UPFCs) is designed in [77] for mitigating LFOs. In [78], deep neural network (DNN) is leveraged to enhance the probabilistic small-signal stability.

1.4 Reinforcement Learning

Another AI-based approach that has been deemed promising to address various challenges in power systems is the reinforcement learning (RL) algorithm. Its applications range from the design of a resistive braking controller for stability enhancement [79], [80] to automatic generation control and economic dispatch problems [81], [82], and [83]. In [84] and [85], two different RL techniques, namely Q-learning and least worst action Q-learning (LWA Q-learning), are utilized to design wide-area damping controllers (WADCs). The design of two reduced-dimensional RL-based WADCs is examined in [86]. However, the performance of the RL methods is mainly dependent on the quality of the handcrafted features [87].

Recently deep reinforcement learning (a combination of RL algorithm and DL technologies such as DNNs) has learned to play Atari games using screen pixels as inputs [87]. In 2017, AlphaGo, the Google Deep Mind's DRL-based computer program defeated the world champion at the game of Go in discrete action space [88]. DRL offers the following advantages: a) extraction of high-dimensional features, b) development of different scalable algorithms such as deep Q-network (DQN) [87] and proximal policy optimization (PPO) [89], hence making them suitable for solving different problems for large-scale power systems. Some of the examples of the application of DRL methods in power system analysis include grid emergency controls, autonomous voltage control, composite load modeling, and load frequency control [90], [91], [92] and [93]. A comprehensive review of such applications is presented in [94] and [95].

In [96], a DRL-based deep deterministic policy gradient (DDPG) algorithm is utilized to design a wide-area damping controller. Recently, the use of DRL has also been explored in [97] to address the issue of ultra low frequency oscillations (oscillations below 0.1 Hz) in a hydro-dominant power system.

1.5 Research Motivation

Though considerable research has already been done in the field of damping electromechanical oscillations in the power grid, there is scope for further improvement due to the following reasons:

1. Most of the proposed approaches require changing the configurations of existing controls.

2. In a large-scale interconnected power grid, there are multiple controllers that act based on local measurements for mitigating a specific mode or multiple oscillation modes. However, in the absence of proper coordination, they may act against each other, and thereby, adversely affect the modes.

3. Implemented controller designs had higher-order complexity.

4. The comprehensive assessment of the robustness of different types of controllers based on their detailed dynamic models requires more research. This is particularly important to capture the unfavorable dynamic interactions between the various damping controls and design a suitable mitigation strategy accordingly.

5. The developed wide-area control algorithms require knowledge of the physical dynamics to compute the optimal control signals, which may not always be possible due to the complex nature of the systems and their rapidly changing operating states. Additionally, these methods may not always be scalable to a large-scale system.

6. Despite the efforts made by several authors to develop AI-based stabilizing control methods ([84], [85]), there has not been much attention devoted to the safe use of DRL to improve the small-disturbance stability of the system. Also, many of the existing methods are inhibited by the use of discrete action spaces and large number of training episodes.

7. No studies concerning the use of DRL for the coordination of different types of damping controllers have been conducted.

1.6 Research Objectives and Aims

This research aims to undertake a thorough evaluation of the improvement of small-signal stability of power systems by designing different coordinated wide-area damping controls that can use multiple controls such as PSSs, FACTS devices, and HVDC lines. The primary sub-tasks of this research are described as follows:

1. To develop the two-terminal and multi-terminal HVDC models in PSLF and DSA tools for integration with AC network models [98], [99]. This is one of the prerequisites for designing a SDC for performing small-signal and transient stability studies.

2. To develop a methodology for the selection of suitable stabilizing signals for the coordinated controllers.

3. To design coordinated wide-area damping controllers (CWADCs) that can perform reasonably for a wide range of operating points using:

    - a model-based approach assuming that the dynamic system models are known *a priori*.
    - a model-free design that is more robust to modeling errors and can learn to

perform effectively using the measured input-output signals of the controlled system.

4. To coordinate individual controllers like HVDC-based SDCs, SVCs, and PSSs. This task would incorporate detailed dynamic models of all the aforementioned controls to determine the interactions between them and the network accurately.

5. To thoroughly investigate the robustness of the designed coordinated controllers within meshed AC/DC power systems to varying signal transmission delay and signal loss.

1.7 Contributions of this work

This work aims to address some of the key issues identified in the current body of literature. The main outcome of this research is the design of the control techniques to:

- improve the coordination between different types of damping controllers.
- enhance the damping of LFOs.

The designed methods combine the advantages of local signals with the additional degrees of freedom provided by remote measurements to achieve the targeted damping of LFOs. Additionally, since it is preferable that feedback signals are synthesized from a small set of measurements, the proposed work exploits the dynamic characteristics of oscillatory modes first to choose an optimal number of state feedback signals and then design different types of optimally coordinated wide-area controllers for a range of scenarios using partial state-feedback. The salient contributions of this thesis are summarized as follows:

1. Evaluation of the improvement in small-signal stability that HVDC-based SDC (DC-SDC) can achieve.

2. Use of detailed dynamic models of control devices such as PSSs and SVCs to accurately determine the interactions between the controls and the network.

3. Development of a controller whose design is based on the physical dynamics of a reduced-order polytopic system. The designed control, namely, CWADC, is composed of gains determined by solving a convex optimization problem that minimizes the weighted sum of the LMI-based H_2/H_∞ controls while satisfying the pole-placement constraints.

4. Systematic study of the impact of communication delays on the proposed CWADC.

5. Identification of alternate feedback signals for CWADC to guarantee the system stability in case of a loss of wide-area signal due to failure in PMU or communication channels.

6. Design of a DNN-CWADC that can generate suitable control actions for the diverse operating conditions (OCs) at a fraction of the time required by the modern controls. The learning process involves using a polytopic dataset designed from a reduced-order system using LMI-based mixed H_2/H_∞ optimization with partial state-feedback and regional pole-placement constraints.

7. Development of a DRL-CWADC using a safe continuous control action search method called the *bounded exploratory control* (BEC)-based DDPG (BEC-DDPG), that is capable of adapting itself to evolving operational uncertainties and unanticipated scenarios in real-time. The proposed method is also aimed to accelerate the training time of the DRL-based control.

8. Design of a suitable method for selecting the most relevant type of the AI-based CWADCs based on the current OC detected by real-time measurements.

The remainder of the report is structured as follows:

- Chapter 2 gives a brief theory review of the mathematical concepts relevant to the design of the proposed control methods. The first section provides the theoretical background of the LMI-based H_2/H_∞ control, while the next two sections present the overview of NN and RL.

- Chapter 3 describes the dynamic models of different controllers, such as PSS, SVC, and DC-SDC. A method for reduction in system size and selection of suitable stabilizing signals is also proposed. The selected wide-area signals are then used to demonstrate the technique for designing a coordinated wide-area controller for a reduced order polytopic system. Finally, to ensure the reliability of feedback signals, an alternative feedback selection scheme is proposed.

- Chapter 4 discusses the design and implementation of the two AI-based approaches for creating CWADCs. The first two sections focus on utilizing DNN and DRL for designing the coordinated wide-area controls. Following this, a technique to select a suitable controller based on the current OC is presented.

- Chapter 5 evaluates the effectiveness of the control for two test systems. For the 16-machine, 68 bus system, modal analysis was performed to ensure that the damping of closed-loop eigenvalues is more than a pre-specified value. For the reduced WECC model-a large test system incorporating multi-terminal and two-terminal HVDC lines and wind generation, both eigenvalue analysis and non-linear time-domain simulations are conducted to validate that the control is resilient to communication signal delay and failure of PMUs. In conclusion, a brief discussion is presented based on the findings of this work.

- Chapter 6 demonstrates the application of the two AI-based methods to the reduced-

order model of the WECC. A brief discussion on the advantages of the methods and their applicability is provided.

- Finally, the conclusions of this work are summarized in Chapter 7. Suggestions for future work are also provided.

CHAPTER 2

THEORETICAL BACKGROUND

2.1 Modern Controls

This section gives a brief overview of different concepts related to the LMI-based multi-objective feedback control. The basics of LMI and three different design objectives used to formulate the control problem, i.e., H_2 control, H_∞ control, and pole-placement, are explained here.

2.1.1 Linear Matrix Inequality (LMI)

An LMI has the following form [100]:

$$F(x) := F_0 + \sum_{i=1}^{m} x_i F_i > 0 \tag{2.1}$$

where $x \in \mathbb{R}^m$ is an unknown vector of scalar optimization variables. $F_i \in \mathbb{R}^{n \times n}$ are the known system matrices. The inequality implies that $F(x)$ is positive definite, i.e., $z^T F(x) z > 0$ for all non-zero $z \in \mathbb{R}^n$. The LMI defined in (2.1) is a convex constraint on x. That is, if x and y be two vectors such that $F(x) > 0$ and $F(y) > 0$, then convexity implies that $F(\frac{x+y}{2}) > 0$.

Convexity has an important consequence, since, even though (2.1) has no general so-

lution, it can still be solved numerically. Therefore, if a feasible solution exists, it can be found, and it is guaranteed to be optimal.

Different Representations

a) An advantage of representing control problems with LMIs is that multiple control requirements (expressed as individual LMI constraints) can be regarded as a single LMI problem [101]:

$$F^1(x) > 0; F^2(x) > 0; ...F^q(x) > 0; \tag{2.2}$$

Thus an equivalent LMI can be expressed as:

$$F(x) = F_0 + \sum_{i=1}^{m} x_i F_i = diagF^1(x), F^2(x), ..., F^q(x) > 0 \tag{2.3}$$

where $F_i = diagF^1(x), F^2(x), ..., F^q(x), \forall i = 0, ..., m$ and $diagX_1, X_2, ...X_q$ is a block diagonal matrix with blocks $X_1, X_2, ...X_q$.

b) In some applications, LMIs may not arise in the canonical form (2.1), but rather in the form [100], [102]:

$$L(X_1, X_2, ...X_n) < R(X_1, X_2, ...X_n) \tag{2.4}$$

where $L(.)$ and $R(.)$ are affine functions of the structured variables $X_1, X_2, ...X_n$. An example of using matrices as variables is the Lyapunov inequality:

$$AX + XA' < 0 \tag{2.5}$$

where $A \in \mathbb{R}^{n \times n}$ is given and $X = X^T$ is the decision variable that can be expressed in the form of LMI, (2.1), as follows: Let $P_1, P_2, ...P_m$ be a basis for the symmetric $n \times n$ matrices ($m = n(n+1)/2$), then set $F_0 = 0$ and $F_i = -A^T P_i - P_i A$.

Types of LMI Problems

There are three generic problems related to the study of LMIs [100]:

- Feasibility problem: The test whether or not there exists a solution x for $F(x) > 0$ is called a feasibility problem. If no solutions exist, the LMI is called infeasible.

- Optimization: Minimizing a convex objective function under LMI constraints is also a convex problem. In particular, the linear objective minimization problem defined by (2.6) plays an important role in the the LMI-based design. This problem is called the eigenvalue problem.

$$\textit{Minimize } \; c^T x \; \textit{ subject to } \; F(x) > 0 \tag{2.6}$$

- Generalized eigenvalue problem: Minimize a scalar, λ, subject to:

$$\lambda B(x) - A(x) > 0, B(x) > 0, C(x) > 0 \tag{2.7}$$

  where A, B, and C are symmetric matrices that are affine functions of x.

The strength of LMIs is that the multiple design objectives like H_∞ for the minimization of uncertainties, H_2 for control effort optimization, and pole-placement region can be formulated as individual LMIs. All of these can be combined to form a single multi-objective LMI control problem as explained in Section 3.2.1. The key to this problem lies in evaluating a single Lyapunov matrix that can satisfy all three design criteria. More information about LMIs including their advantages and limitations can be found in [100], [101], [103].

2.1.2 H_2 control

H_2 control is an expansion of the linear quadratic Gaussian (LQG) control problem. If the LQG problem is evaluated in the frequency domain, it is referred to as H_2 control [104]. For a stable linear time invariant (LTI) model of plant P:

$$\begin{aligned} \dot{x} &= Ax + B_1 w + B_2 u \\ z_2 &= C_2 x + D_{22} u \\ y &= C_y x + D_{y1} w + D_{y2} u \end{aligned} \tag{2.8}$$

where x is the state vector, u is the control input vector, w represents the disturbance input, z_2 represents the error output corresponding to H_2 control to be kept small, and y is the output vector. The standard H_2 optimization problem is the problem of choosing a feedback controller, K, such that it [105]:

- stabilizes the closed-loop system, and
- minimizes the H_2 norm of the closed-loop system.

From (2.8), the closed-loop transfer function is formed as:

$$G(s) = C_{cl2}(sI - A_{cl})^{-1} B_{cl} + D_{cl2} \tag{2.9}$$

where $A_{cl} = A + B_2 K$, $B_{cl} = B_1$, $C_{cl2} = C_2 + D_{22} K$, and $D_{cl2} = D_{21} = 0$. The H_2 performance $||G||_2$ is defined as:

$$||G||_2^2 = \frac{1}{2\pi} Trace \int_{-\infty}^{\infty} G^T(-j\omega) G(j\omega) d\omega \tag{2.10}$$

The minimization of the H_2 norm of the closed-loop system implies the minimization of the sensitivity of the output noise z_2 to the white noise input, w. The H_2 norm measures

the complete energy of the system associating the input disturbance to the output response. It is important to note that one of the peculiar things about the H_2 norm is that it can be equal to infinity in a stabilized closed-loop system when the plant D_{21} matrix associated with the input disturbances and output errors is non-zero. Therefore, based on the formulae described in [106], [107] optimal H_2 control implemented by LMI toolbox necessitates the setting of D_{21} to zero. The H_2 norm of the closed-loop transfer function, $||G||_2$, does not exceed the specified maximum bound, ξ_2, where $\xi_2 > 0$ if and only if $D_{21} = 0$ and there exists two symmetric matrices X_2 and Q such that:

$$\begin{bmatrix} A_{cl}X_2 + X_2A_{cl}^{\mathrm{T}} & B_{cl} \\ B_{cl}^{\mathrm{T}} & -I \end{bmatrix} < 0 \tag{2.11}$$

$$\begin{bmatrix} Q & C_{cl2}X_2 \\ X_2C_{cl2}^{\mathrm{T}} & X_2 \end{bmatrix} > 0 \tag{2.12}$$

with $X_2 = X_2^T > 0$ and $Trace(Q) < \xi_2^2$.

2.1.3 $\mathbf{H_\infty}$ control

For any system, the H_∞ norm is the largest magnitude of transfer function in the worst direction over the entire frequency range. Therefore, minimization of the H_∞ norm is a commonly used tool for not only checking the stability of the system, but also for designing the feedback controller as it allows for loop-shaping and robustness when plant modeling

errors are present. For a stable LTI model of plant P:

$$\begin{aligned} \dot{x} &= Ax + B_1 w + B_2 u \\ z_\infty &= C_1 x + D_{11} w + D_{12} u \\ y &= C_y x + D_{y1} w + D_{y2} u \end{aligned} \tag{2.13}$$

where x is the state of system, u is the control, w is the disturbance, y is the output, and output channel z_∞ is associated with H_∞ control performance. From (2.13), the closed-loop transfer function is formed as:

$$T(s) = C_{cl\infty}(sI - A_{cl})^{-1} B_{cl} + D_{cl\infty} \tag{2.14}$$

where $A_{cl} = A + B_2 K$, $B_{cl} = B_1$, $C_{cl\infty} = C_2 + D_{12} K$, and $D_{cl\infty} = D_{11}$. The H_∞ norm of the closed-loop transfer function, T, $||T||_\infty$ is defined as:

$$||T(s)||_\infty = \sup_{\omega > 0} \bar{\sigma}(T(j\omega)) \tag{2.15}$$

where $\bar{\sigma}(T(j\omega))$ is the largest singular value of T. The LTI system given by 2.13 is asymptotically stable and H_∞ norm does not exceed the specified maximum bound, $\xi_\infty > 0$, if and only if there exists a symmetric matrix, X_∞, such that:

$$\begin{bmatrix} A_{cl} X_\infty + X_\infty A_{cl}^{\mathrm{T}} & B_{cl} & X_\infty C_{cl\infty}^{\mathrm{T}} \\ B_{cl}^{\mathrm{T}} & -I & D_{cl\infty}^{\mathrm{T}} \\ C_{cl\infty} X_\infty & D_{cl\infty} & -\xi^2 I \end{bmatrix} < 0 \tag{2.16}$$

$$X_\infty = X_\infty^{\mathrm{T}} > 0 \tag{2.17}$$

2.1.4 Pole-Placement Region

Neither of the two aforementioned controls, i.e., H_2 and H_∞ controls, allow the placement of the closed-loop poles in a particular region. Since the location of the poles is related to the transient behavior and time response of the closed-loop system, it is desirable to ensure additional damping constraints. The closed-loop poles can be forced into some sector of the stable left half-plane to obtain adequate system damping using the regional pole-placement constraints. This will also ensure that the synthesized feedback gains are kept at a reasonable value, which might otherwise lead to controller output saturation, and poor performance to large disturbances [108].

The formulation of the closed-loop region constraints for the systems defined by (2.8) and (2.13) can be expressed using LMIs, which can be solved to obtain the feedback gain matrix, K, for placing the closed-loop poles in any desired LMI region. LMI regions are convex subsets of the complex plane, and can be described by $\boldsymbol{D} = s\epsilon\mathbb{C}\colon V + Ws + W^T\bar{s} < 0$, where $\mathbb{C}$ refers to the complex plane, $V = V^T = \{E_{ij}\}_{1\leq i,j\leq m}$ and $W = \{I_{ij}\}_{1\leq i,j\leq m}$ are fixed real matrices, and $\bar{s}$ is the complex conjugate of s. Different complex regions in the complex planes that are symmetric with respect to the real axis, such as half plane, conic sectors, circles, as well as their combinations, can be expressed as LMI regions. Lastly, the closed-loop system will be *D*-stable, if and only if there exists a positive definite matrix, X_{pol}, such that: [105]

$$[E_{ij}X_{pol} + I_{ij}(A + B_2K)X_{pol} + I_{ij}X_{pol} + I_{ji}X_{pol}(A + B_2K)^T]_{1\leq i,j\leq m+} < 0 \qquad (2.18)$$

where $X_{pol} > 0$.

2.2 Deep Neural Networks

Deep neural networks (DNNs) are non-linear approximators, whose architectures can vary greatly in structure depending on their applications. However, all of the networks have the same basic components. This section will briefly describe those components.

A DNN learns the functional relationship, $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ between the n-dimensional input and the m-dimensional output based on the training done using known input-output datasets. The DNN used in this work has a feedforward structure. It is composed of an input layer, P number of hidden layers, and an output layer. The number of neurons, n, in the input layer is determined by the size of the input features, $x \in \mathbb{R}^n$. The first component of the neurons in each layer is a linear function that computes the weighted sum of the output from the previous layer. An example of a 2-hidden layer architecture with 4 neurons (input features, x_1, x_2, x_3, and x_4) each is shown in Fig. 2.1. The output of the first neuron of hidden layer 1, $z_1^{(1)}$ can be expressed as a linear function as [109]:

$$z_1^{(1)} = x_1\theta_1 + x_2\theta_2 + x_3\theta_3 + x_4\theta_4 + b \tag{2.19}$$

where $\{\theta_1, \theta_2, ...\theta_4\}$ are the weights and b is the bias term. The second component, which represents the final output of each neuron, incorporates an activation function, g, that is tasked with applying a non-linear transformation between the neurons, i.e.,

$$h_1^{(1)} = g(z_1^{(1)}) \tag{2.20}$$

Different types of activation functions, also known as transfer functions are available. Theses activation functions such as linear, sigmoid, and rectified linear unit (ReLU) [109], [110] can be different within each layer. The important property of these functions is that they are differentiable at every value. The most basic transfer function is the linear function,

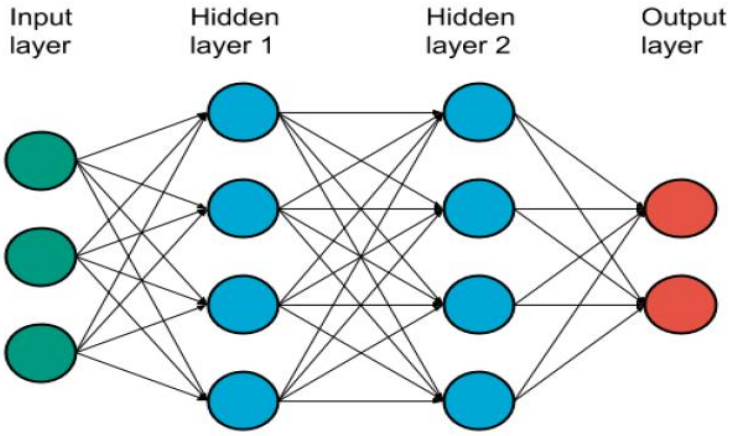

Figure 2.1. Example of a Fully-Connected DNN.

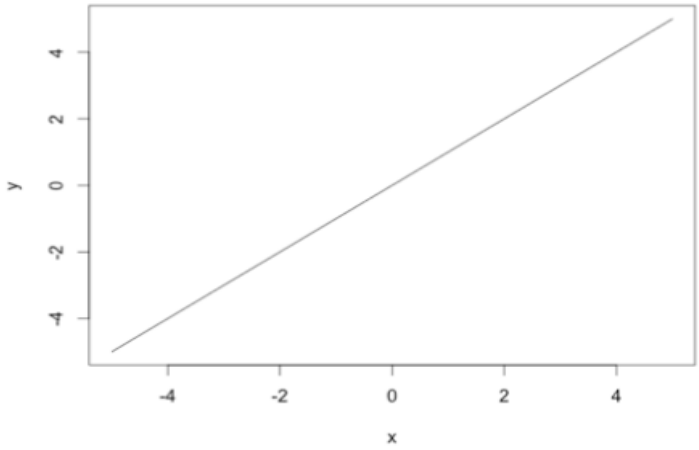

Figure 2.2. Linear Activation Function.

which returns the input value passed to it. The linear function given by (2.21) is shown in Fig. 2.2 [110]. The output layer of the regression-based DNN usually uses this function on its output layer.

$$g(z) = z \tag{2.21}$$

A commonly used activation function for the feedforward DNNs that need to output only

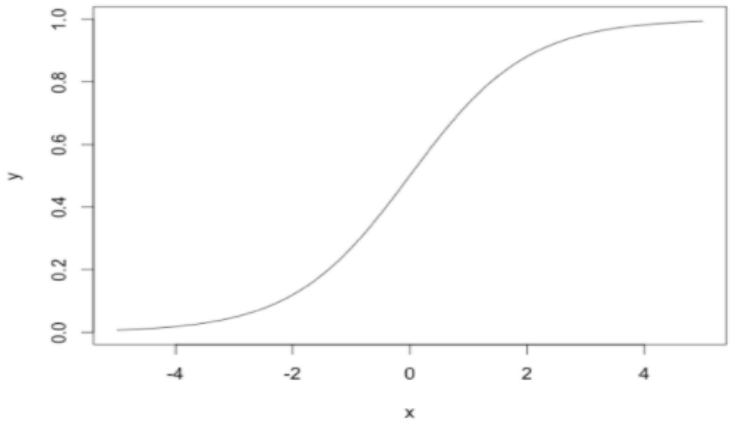

Figure 2.3. Sigmoid Activation Function.

positive values is the sigmoid function given by (2.22) and shown in Fig. 2.3 [110].

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2.22}$$

In recent times, the most suitable choice for the activation function has been the ReLU function as it is resilient against the vanishing gradient problem (refer Fig. 2.4) [109]. The following equation shows the ReLU activation function:

$$g(z) = \max(0, z) \tag{2.23}$$

The loss function (cost function), J, is used to measure how well the DNN model fits the training set. An example of J would be the mean squared error defined as:

$$J = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y} - y)^2 \tag{2.24}$$

where m is the total number of training samples, y is the actual output of a neuron in the

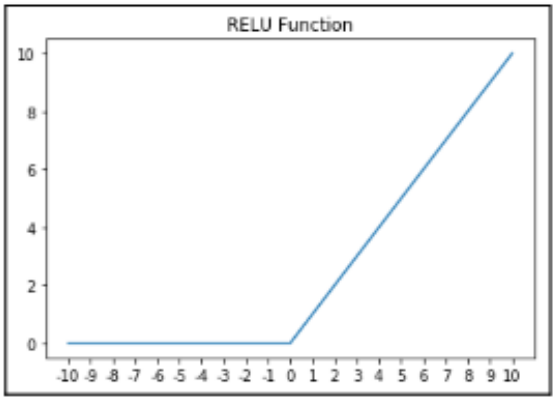

Figure 2.4. ReLU Activation Function.

final layer, and $\hat{y}$ is the predicted value of the target output computed using the forward propagation process (the process of reaching the end of the network starting from the input layer using the weights and activation functions). The training of the DNN is conducted using the back-propagation algorithm, i.e., propagating the error signal backwards through the network. Learning is accomplished by calculating the gradient of the loss function with respect to the network weights using the chain rule. The weights are updated using a learning rate, α, multiplied with the gradient error [109]:

$$\theta_{new} \leftarrow \theta_{old} - \alpha \frac{\partial J}{\partial \theta} \tag{2.25}$$

The stochastic gradient descent algorithm is one of the most popular training algorithms which works by the selection of random batches of training data. An epoch is completed after the complete training data is processed [110].

2.3 Reinforcement Learning

This section presents the theoretical background of reinforcement learning (RL) used to design the wide-area damping control implemented in this report.

RL is an algorithm that learns to make a sequence of decisions through trial and error. The framework consists of an autonomous, self-learning system (referred to as an agent) that learns an optimal or near-optimal control policy dynamically by $interacting$ with its environment in discrete time steps while satisfying a predefined goal. The agent must explore different actions to find more information about the numerical states representing the environment. It must also exploit the known information to eventually learn a behavior that yields maximum positive rewards.

The mechanism of RL is different from other forms of machine learning, i.e., *supervised* learning and *unsupervised* learning. The agent's goal is not to learn from the labeled set of the correct actions (as is done during supervised learning) or to find the general pattern hidden in the unlabeled data (as is done during unsupervised learning). Therefore, RL is considered as a separate category. However, it is to be noted that there are RL algorithms that involve the use of supervised learning.

2.3.1 Markov Decision Process

Both RL and modern controls problems are closely related to each other, particularly when describing system states and actions. Both forms of controls follow the Markov decision process (MDP), i.e., both controls take only the currents states as inputs and generate actions while ignoring the history of states and actions. The formulation of RL as a finite MDP, $\mathcal{M}$, is a 5-tuple [111]:

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathbb{P}, \gamma) \tag{2.26}$$

where $\mathcal{S}$ is the state-space which can be continuous or discrete, $\mathcal{A}$ is the action space which can be continuous or discrete, $\mathcal{R}$ is the distribution of rewards given a state-action pair, $\mathbb{P}$ is the transition probability that maps a state-action pair at time t onto a distribution of states at time, $t+1$, and $\gamma \in [0, 1]$ is the discount factor.

Fig. 2.5 shows the interaction between the agent and the environment for a generic RL model. At each time step, t, the agent receives an observation, $s_t \in \mathcal{S}$, from its environment, takes an action, $a_t \in \mathcal{A}$, and receives a scalar reward, $r_t \in \mathcal{R}$. The process is repeated in episodes. The action-reward feedback loop continues until the training reaches the end of one dynamic simulation (end of one episode).

The agent's behavior is defined by the policy, π, that maps the state space to action space, $\pi : \mathcal{S} \rightarrow \mathcal{A}$. The policy can be either deterministic or stochastic. A deterministic policy always selects the same action given the exact same state every time, and is given by $\pi(s) = a$. This is usually the same approach, followed by the modern controls. On the contrary, a stochastic policy maps the state to the probability distribution over the action space, i.e., $\pi(a) = \mathbb{P}(a)$. This policy can be very useful when one of the PMU sensors is not reliable or noisy.

The goal of the agent is to learn an optimal policy π^* that maximizes the cumulative discounted reward, $\mathcal{R}_t = \sum_{i=t}^{T} \gamma^{i-t} r_{i+1}$ [112]. The γ is a hyperparameter which determines the weight of the future rewards. If $\gamma = 0$, then agent considers only current reward, while for a value of 1, priority is given to the future rewards.

A central tool in the search for the optimal policies is the evaluation of the state-value

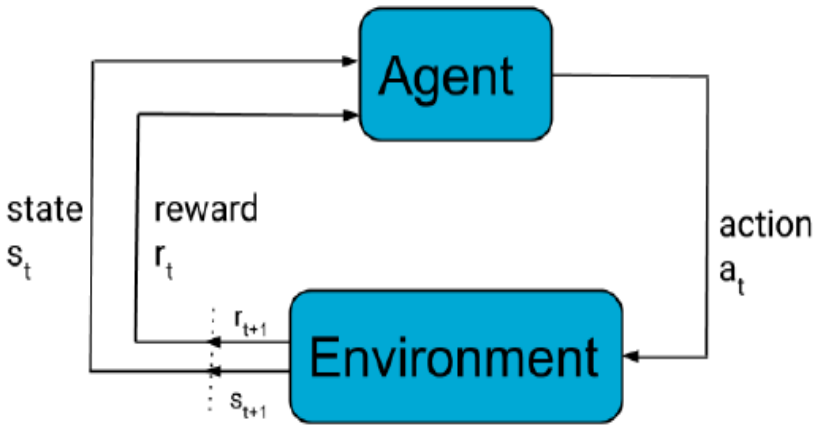

Figure 2.5. Interaction Between RL Agent and Environment.

function, V^{π}, and action-value function, Q^{π}, also called the Q-function. The state-value function determines the goodness of a state and is defined as the expected cumulative future discounted reward in state s from following the policy, π.

$$V^{\pi}(s) = \mathbb{E}_{\pi}[\mathcal{R}_t \mid s_t = s] \tag{2.27}$$

The Q-function quantifies the expected cumulative reward after taking the action, a_t, in state, s_t, and then follows the policy, π [112]:

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{\pi}[\mathcal{R}_t \mid s_t = s, a_t = a] \tag{2.28}$$

The optimal Q-value is determined using the recursive relationship given by the Bellman optimality equation:

$$Q^{*}(s_t, a_t) = \mathbb{E}\{r_{t+1} + \gamma \max_{a_{t+1}} Q^{*}(s_{t+1}, a_{t+1}) \mid s_t = s, a_t = a\} \tag{2.29}$$

The Bellman equation is used to guide the estimates of Q-values close to their true values.

RL can be divided into two main categories: model-based and model-free approaches. A model-based algorithm uses the model dynamics to estimate an optimal policy. For example, a model predictive control based trajectory optimization can be used to direct the policy learning and avoid local optima. This concept is similar to the Guided Policy Search (GPS). Dynamic Programming is another model-based technique, which requires the use of transition probabilities (probability distribution over next states and rewards), and hence is primarily used for planning in a MDP [113], [114].

The second category is the model-free RL algorithm. It learns a policy without needing to use the model or the information of the system dynamics. This model is useful in situations when a transition function is not available to describe the dynamics of the en-

vironment, but instead, experiences can be sampled. It can be further divided into two subcategories: policy-based methods and value-based methods.

2.3.2 Value-Based Methods

The value-based RL methods approximate the Q-function, and use it to take an action. One of the popular value-based algorithms is Q-learning, which finds the optimal value of Q-function using [113]:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \eta[r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{2.30}$$

where η represents the learning rate. The problem with the Q-learning is that it is not scalable, as it is not computationally feasible to compute $Q(s_t, a_t)$ for every state-action pair in a large state space. The solution is to use a function approximator, such as a DNN, to approximate the Q-value; this algorithm is referred to as deep Q-learning (DQL). The benefit of value-based methods is that they can learn off-policy, i.e., learning about policy π (target policy) from experiences sampled from another policy, ν. Another example of value-based method is state action reward state action (SARSA), which, however, learns on-policy (learning about policy π from experiences sampled from the same policy, π). The main disadvantage of these methods is that they are only suitable for discrete action space; they may also have convergence issues.

2.3.3 Policy-Based Methods

Policy-based methods (also referred to as policy gradient) learn policy π directly without using the Q-function to make the decisions. The REINFORCE and the trust region policy optimization (TRPO) are some of the examples of policy-based methods. The advantage of these methods is that they are more effective in high-dimensional and continuous

action spaces, but suffer from high variance. However, there are techniques to reduce the variance, some of which are REINFORCE with baseline and actor-critic algorithms [111].

2.3.4 Actor-Critic: Deep Deterministic Policy Gradient

Both value-based and policy-based methods have their pros and cons. A natural idea is to combine the benefits of both methods. This is called the actor-critic method. The policy π can be thought of as an actor because it chooses the action while the action-value function, Q, can be thought of as a critic because it evaluates the action selected by the actor.

The Deep Deterministic Policy Gradient (DDPG) is also an actor-critic method developed for handling continuous actions space [112]. As the name suggests, it uses a deterministic approximation of the policy function instead of a stochastic function. It is an off-policy method that concurrently learns a policy and a Q-function. It uses off-policy data and Bellman equation to learn Q-function, and uses the Q-function to learn the policy. The DDPG agent uses four function approximators (DNNs) to estimate the policy and value functions: actor (π), target-actor (π^{targ}), critic (μ), and target-critic (μ^{targ}). The target networks are time-delayed copies of their original networks that slowly track the learned networks. Without the use of these target networks, learning can become unstable. The update equations of the critic (Q) network are dependent on the target Q-values (refer 2.30), which in turn depends on the same weight parameters used for training, θ. This may lead to the divergence of the Q network. Therefore, copies of the critic μ^{targ} and the actor π^{targ} networks are creating for calculating the target values. The original μ and π networks are updated after each parameter updates while the weights of both the target networks are updated slowly (soft updates): $\theta^{targ} \leftarrow \rho\theta + (1-\rho)\theta^{targ}$ with ρ. Here ρ is a hyperparameter, referred to as $polyak$ whose value lies between 0 and 1 (usually close to 1).

RL is a sequential process, hence the states and the actions explored by the agent during

the learning would not be independently distributed. This can lead to inefficient learning. To address this problem, DDPG uses a finite-sized cache, replay buffer, $\mathcal{D}$, to sample the experiences to update the DNN parameters. The agent's experiences are stored as a tuple (s_t, a_t, r_t, s_{t+1}) into the replay buffer. During the update of the function approximators, a mini-batch of $\mathbf{B}$ random samples is drawn from $\mathcal{D}$. This ensures that the correlation amongst the drawn samples is low and also provides better sampling efficiency.

There is a trade-off between exploration and exploitation in the RL algorithm. To obtain higher rewards, if an agent always selects the action (exploit) it believes is the best (generate the maximum reward), it will never be able to explore new actions. However, to discover these actions, it must explore new and perhaps better actions. This is referred to as the exploration-exploitation dilemma in RL. To improve the exploration capabilities of DDPG policies, noise sampled from a pre-defined process is added to the actions. For example, in [112], the Ornstein-Uhlenbeck (OU) process is used to generate temporally correlated noise. The algorithm for DDPG is listed in **Algorithm 1** [112].

Algorithm 1: Deep Deterministic Policy Gradient (DDPG)

Randomly initialize critic network $Q(s, a \mid \theta)$, actor network $\mu(s \mid \theta)$ with weights θ and ϕ.
Set the target networks, Q^{targ} and μ^{targ} with weights $\theta^{targ} \leftarrow \theta$ and $\phi^{targ} \leftarrow \phi$.
Initialize the experience replay buffer, $\mathcal{D}$.
for $episode, e \leq M$ **do**
    Observe initial state, s.
    Initialize random noise, $\mathcal{N}$.
    for $timestep, t \leq T$ **do**
        Select action, $a_t = \mu(s_t \mid \theta + \mathcal{N}_t)$.
        Execute a_t and observe the reward, r_t, and next state, s_{t+1}.
        Store the transition $\{s_t, a_t, r_t, s_{t+1}\}$ in $\mathcal{D}$.
        Sample a random minibatch of $\mathbf{B}$ transitions $\{s_i, a_i, r_i, s_{i+1}\}$ from $\mathcal{D}$.
        Compute the critic target $y_i(r_i, s_{i+1}) = r(s_i, a_i) + \gamma Q^{targ}(s_{i+1}, \mu^{targ}(s_{i+1}) \mid \theta^{targ})$.
        Update the Q-function (critic) by one step of gradient descent using
        $$L = \frac{1}{\mathbf{B}} \sum_{i=1}^{\mathbf{B}} (y_i - Q(s_i, a_i \mid \theta))^2.$$
        Update actor policy by one step gradient by
        $$\nabla_\phi \approx \frac{1}{\mathbf{B}} \sum_{i=1}^{\mathbf{B}} \nabla_a Q(s, a \mid \theta) \mid_{s=s_i, a=\mu(s_i)} \nabla_\phi \mu(s \mid \phi) \mid_{s=s_i}.$$
        Soft update the target networks as
        $$\theta^{targ} \leftarrow \rho\theta + (1-\rho)\theta^{targ}$$
        $$\phi^{targ} \leftarrow \rho\phi + (1-\rho)\phi^{targ}$$
    end
end

2.4 Imitation Learning

An alternative to RL for learning sequential decision-making policies is imitation learning (IL). In IL, instead of learning from the reward function, the agent first observes the actions of an expert (demonstrator) and then uses this set of expert's demonstrations to learn the optimal policy by following (imitating) the expert's decisions. This approach is successful in scenarios where a reward function is not available (e.g., teaching a self-driving vehicle), or designing a reward function that can satisfy the desired behavior is highly complicated. This method has also benefited from the recent progress in core learning techniques, increased availability, and fidelity of demonstration data [115]. The IL algorithm is formulated using a MDP framework. However, unlike the RL algorithm, a set of demonstration trajectories, which are sequences of states and actions, are also provided to the environment [116]:

- State-space, $\mathcal{S}$ and action space, $\mathcal{A}$.
- Transition model, $\mathbb{P}(s_{t+1}|s_t, a_t)$ that maps a state-action pair at time t onto a distribution of states at time, $t+1$.
- Optionally, a reward function, $\mathcal{R}$ that the expert is trying to optimize is also available in some problem settings [117].
- Set of one or more teacher demonstrations $(s_0, , a_0, s_1, a_1, ...)$, where actions are drawn from the expert's (optimal) policy, π^*.

The detailed description of this technique can be found in [116], [117].

CHAPTER 3

MODERN CONTROL IMPLEMENTATION

3.1 Controllers and their Interactions

3.1.1 Design of PSSs, SVC and DC-SDC

The aim of both PSSs and DC-SDC is to damp the modes of rotor oscillations by inducing pure damping torques on shafts of generators. For the purpose of addition of DC-SDC, HVDC systems are modeled as described in [98]. The PSS model shown in Fig. 3.1(a) uses shaft speed deviation as an input signal to add the stabilizing signal, V_s^{PSS}, to the exciter input and is of the form,

$$V_s^{PSS} = K_{PSS}(\frac{sT_w}{1+sT_w})(\frac{1+sT_{1PSS}}{1+sT_{2PSS}})(\frac{1+sT_{3PSS}}{1+sT_{4PSS}})\Delta\omega \tag{3.1}$$

The design of DC-SDC used in this work is similar to the conventional structure of a PSS as shown in Fig. 3.1(b). The controller output, P_{mod}^{DC}, given by (3.2), modulates the active power reference setpoint through HVDC link based upon the local signal, i.e., the frequency deviation of the AC bus to which DC converter is connected.

$$P_{mod}^{DC} = K_{DC}(\frac{sT_w}{1+sTw})(\frac{1+sT_{1DC}}{1+sT_{2DC}})(\frac{1+sT_{3DC}}{1+sT_{4DC}})\Delta f \tag{3.2}$$

The first terms in (3.1) and (3.2) are used to washout the compensation effect after a time lag, T_w, which ensures that steady state changes in the input do not modify the output signal. $T_{1PSS} - T_{4PSS}$ are lead time constants for PSS to compensate the phase lag between the input and output signals at oscillation frequencies of interest and $T_{1DC} - T_{4DC}$ is a lead compensation pair for DC-SDC. K_{PSS} represents the gain block for PSS while K_{DC} is the gain for DC-SDC. The third type of controller which is considered in this study is an SVC model whose structure is shown in Fig. 3.1(c). Its primary objective is to provide voltage support to the system, but is equipped with an additional capability to include its own stabilizer model for damping LFOs.

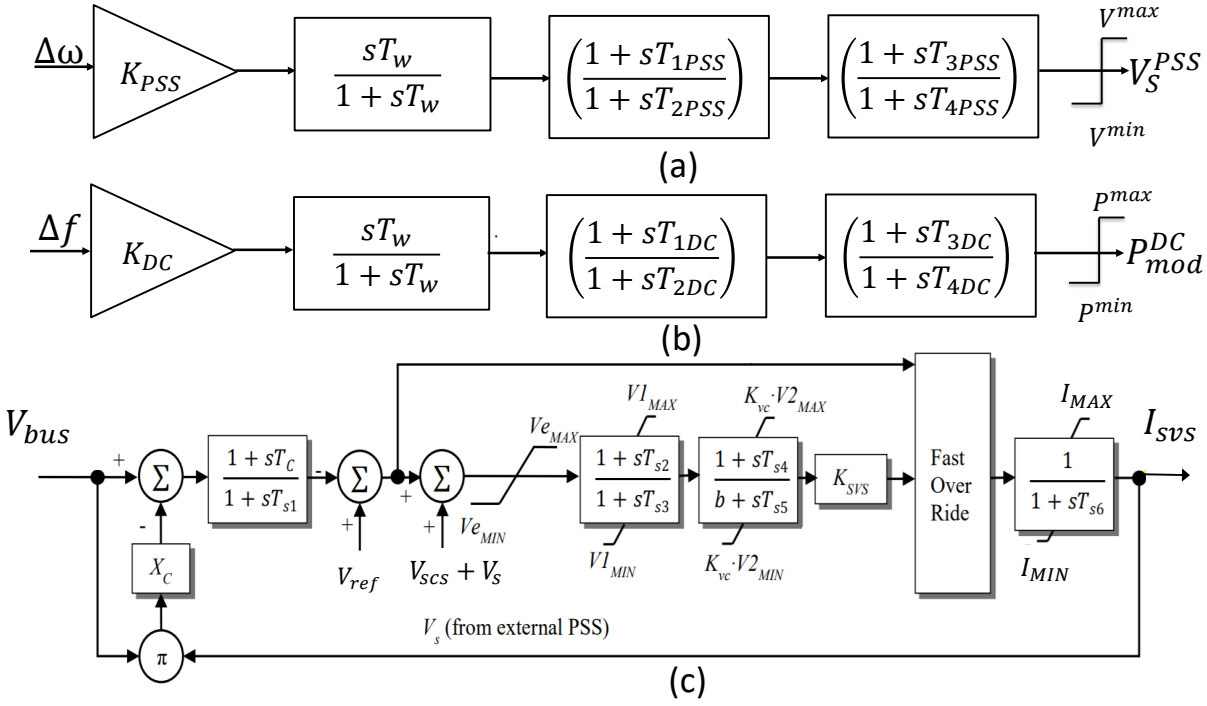

Figure 3.1. Different Types of Controllers.

3.1.2 Types of Control Interactions

To enhance the performance of large interconnected power systems, there has been a considerable increase in the number and complexity of power system controllers. However,

the inter-controller interactions may result in effects that are detrimental to the overall system performance. As there can be many possible combinations and configurations of the controls, the scope of this report is limited to the study of electromechanical interactions between controllers. For the controllers considered in this thesis, these can be categorized into:

a) Interactions between different PSSs,

b) Interactions between PSSs and SVCs,

c) Interactions between PSSs, SVCs, and DC-SDC.

These interactions can manifest themselves in many ways. For example, improvement in the gain of DC-SDC can improve the damping of one mode, but degrade the damping of other modes. Hence, there is a need to coordinate the tuning of these controls. An example of negative interactions among different controls is provided in Table 5.5 of Section 5.2.1.

3.2 Proposed Wide-Area Damping Controller Structure

One approach to provide better observability of the inter-area modes as well as minimize the potential for adverse interactions between the (multiple) existing controllers is by using both local and wide-area signals simultaneously, resulting in a bi-level operational scheme. The schematic of the CWADC design which does so is shown in Fig. 3.2. The selected stabilizing signals are measured by geographically distributed PMUs and sent to CWADC through dedicated communication links. The control signals are then sent to automatic voltage regulators (AVRs) of several selected local machines, SVCs, and SDCs installed on DC lines as shown in Fig. 3.3. On receiving the respective control commands, the local controllers act in unison to increase the damping of oscillation modes. This design process does not require any modification of parameters of the existing controllers; instead, it exerts additional stabilizing signals, u_i, (where $i = \{1, 2..p\}$ and $p <$ total controllers) to

enable them to overcome their tuning deficiencies. Furthermore, if remote signals are lost, the bi-level design ensures that all the local controllers can autonomously work based on the locally available information.

To restrict the output of CWADC within an acceptable range, limiters are also included. Limiter action is mandatory for the system since the proposed controller can create unnecessary violations in the AC and DC voltages in the network through the action of local controllers. In this design process, all PSSs, DC-SDCs, and SVCs are modeled in the open-loop state-space representation, on which design of CWADC is based.

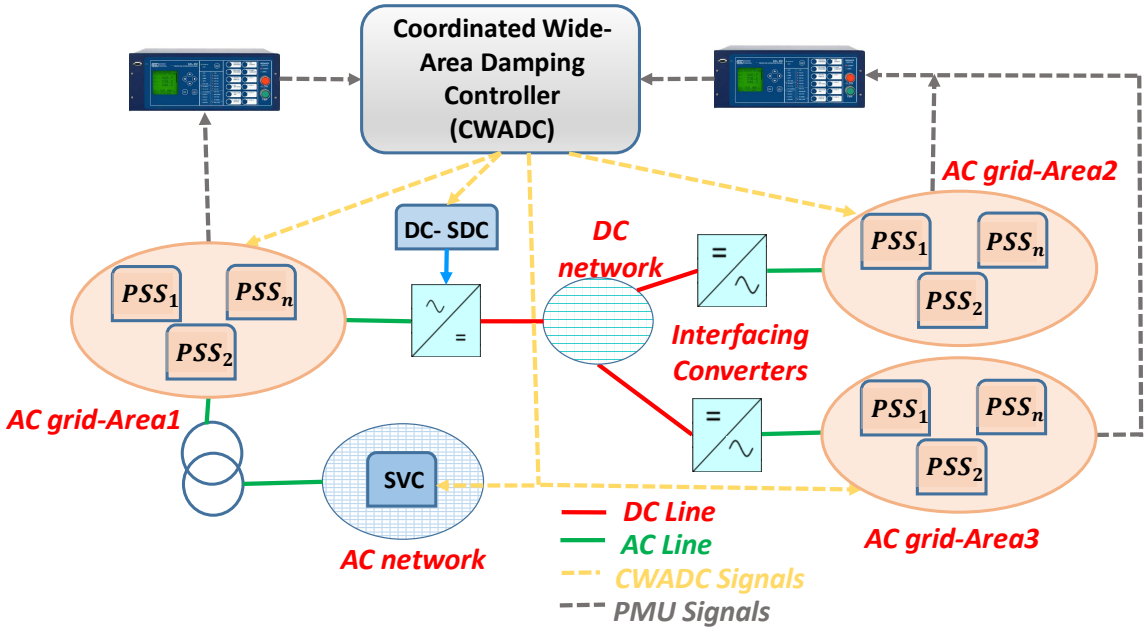

Figure 3.2. Schematic of CWADC.

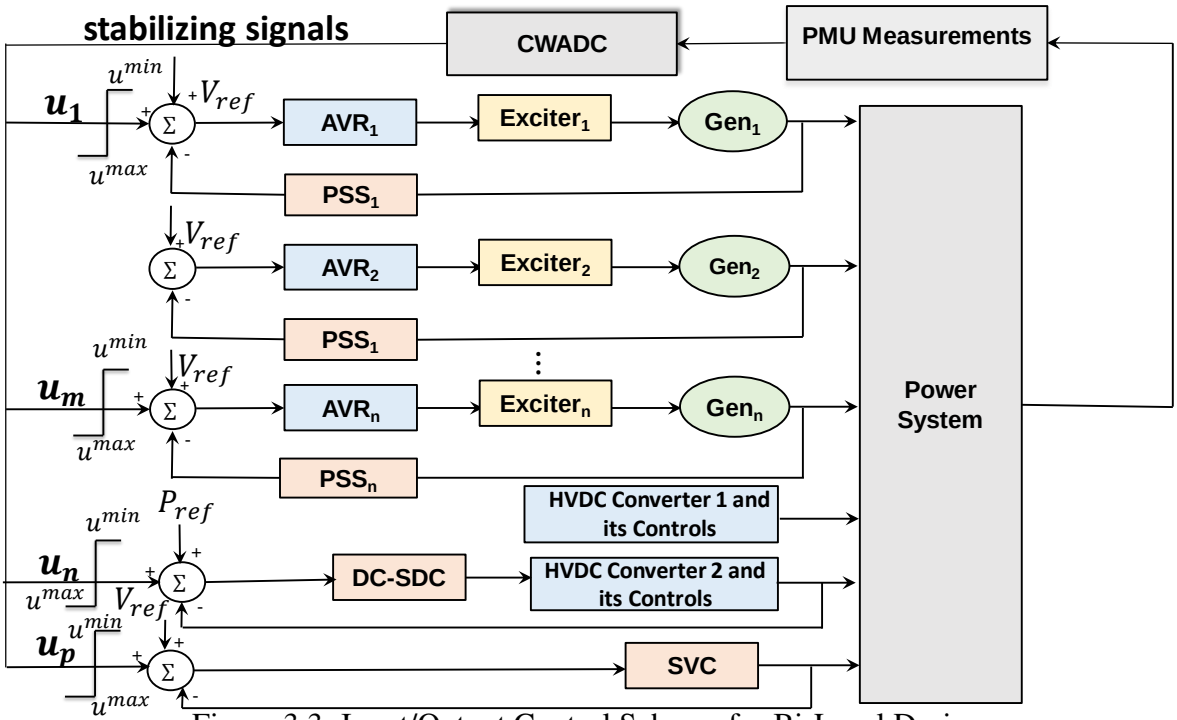

Figure 3.3. Input/Output Control Scheme for Bi-Level Design.

3.2.1 Multi-objective Controller Design

An ideal control design must satisfy a mix of performance and robustness objectives, which can be difficult to realize using a single control criterion [105], [118]. Therefore, the proposed controller is designed based on a multi-objective H_2/H_∞ synthesis technique, that incorporates various criteria including control effort minimization, disturbance rejection, optimal control performance, and control robustness. The H_2 optimization gives more control over the system's transient behavior while minimizing control cost, while the H_∞ control ensures robustness against dynamic uncertainties [119]. To ensure good transient response of the closed-loop system, pole-placement constraints are added to the multi-objective control problem. Such constraints are important as they keep gains at reasonable values, which may otherwise lead to controller output saturation [43]. An LMI framework is used to solve for state feedback with the combination of objectives described above. The

state-space model of the studied system and its associated controls corresponding to an OC posed with H_2/H_∞ formulation is described by:

$$\begin{aligned} \dot{x} &= Ax + B_1 w + B_2 u \\ z_\infty &= C_1 x + D_{11} w + D_{12} u \\ z_2 &= C_2 x + D_{22} u \\ y &= C_y x + D_{y1} w + D_{y2} u \end{aligned} \tag{3.3}$$

where x is the state of the system, u is the control, w is the disturbance, y is the output, A, B_1 , B_2 , C_1 , C_2 , D_{11} , D_{12} , D_{22}, C_y, D_{y1}, D_{y2} are state-space matrices, and z_2 and z_∞ correspond to H_2 and H_∞ controls, respectively. Note that D_{21} is set to zero for computing the optimal H_2 control, as it will otherwise make the H_2 norm infinite. The three sets of design objectives, i.e., H_2, H_∞ optimizations, and pole-placement constraints as defined in Sections 2.1.2-2.1.4 can be combined together, if a single Lyapunov matrix can be found such that:

$$X = X_2 = X_\infty = X_{pol} > 0 \tag{3.4}$$

where X_2, X_∞, and X_{pol} are the Lyapunov matrices satisfying the H_2, H_∞, and pole-placement constraints, respectively (refer Sections 2.1.2-2.1.4). However, the three sets of conditions leads to a non-convex optimization problem. Therefore, with the change of variable $Y = KX$, the suboptimal LMI problem corresponding to multi-objective state-feedback synthesis for the OC defined by 3.3 can be formulated as:

$$\min_{Y,X,Q,\xi^2} (a||T||_\infty^2 + b||G||_2^2) \tag{3.5}$$

satisfying:

$$\begin{bmatrix} AX + XA^{\mathrm{T}} + B_2Y + Y^{\mathrm{T}}B_2^{\mathrm{T}} & B_1 & XC_1^{\mathrm{T}} + Y^{\mathrm{T}}D_{12}^{\mathrm{T}} \\ B_1^{\mathrm{T}} & -I & D_{11}^{\mathrm{T}} \\ C_1X + D_{12}Y & D_11 & -\xi^2 I \end{bmatrix} < 0 \tag{3.6}$$

$$\begin{bmatrix} Q & C_2X + D_{22}Y \\ XC_2^{\mathrm{T}} + Y^{\mathrm{T}}D_2^{\mathrm{T}} & X \end{bmatrix} > 0 \tag{3.7}$$

$$(V \otimes X) + (W \otimes (AX + B_2Y)) + W^{\mathrm{T}} \otimes (AX + B_2Y)^{\mathrm{T}} < 0 \tag{3.8}$$

where X is the Lyapunov matrix, Q is positive-definite matrix, $\otimes$ is a Kronecker product, $\xi^2 < \xi_\infty^2$ and Trace$(Q) < \xi_2^2$. ξ_2 and ξ_∞ are the respective maximum bounds for closed-loop transfer functions of system for H_2 and H_∞ controls.

The optimal state feedback matrix can be obtained as $K = Y(X)^{-1}$. The function $msfsyn$ in LMI toolbox can be used for this purpose. However, for calculating the optimal value of the Lyapunov matrix, X, $mincx$ function is used in LMI which implements Nesterov and Nemirovski's Projective Method as described in [105], [120].

3.2.2 Extension to Multiple Operating Conditions: Polytope Formation

Power system OCs vary with system configuration and load level in a complex manner. As the system's operating point changes, the characteristics of electromechanical modes also vary. Although a controller may be designed for the nominal operating point of a network, real OCs are defined by various factors, including load variations, availability of renewable generation, tie-line outages, and energy market fluctuations. It, therefore, makes more sense to combine different operating points together when designing a single damping

controller that is effective over the whole range. To successfully develop such a control, this report delves into the idea of creating an LMI-based polytopic control design, in which the uncertainty of the system is considered in the design itself. For the implementation of such a model, the system is linearized at a number of typical operating points, that are decided by the scheduling variables measurable in real-time such as load levels, and tie-line flows. The OCs with converged power flows constitute the vertices of the polytope. For a given OC, the vertex, v, of the polytope, Γ, can be expressed as:

$$\Gamma_t = \begin{bmatrix} A_v & B_{v1} & B_{v2} \\ C_{v1} & D_{v11} & D_{v12} \\ C_{v2} & 0 & D_{v22} \\ C_{vy} & D_{vy1} & D_{vy2} \end{bmatrix} \tag{3.9}$$

The convex combination of different vertices representing different system matrices is given by:

$$\Gamma\{\Gamma_1, \Gamma_2, ..\Gamma_v\} = \{\sum_{i=1}^{v} \varphi i \Gamma_i : \sum_{i=1} \varphi_i = 1, \varphi_i \geq 0\} \tag{3.10}$$

Here, φ_i denotes the polytopic coordinate of Γ. The convexity property of the polytope ensures that once the quadratic stability is established for the vertices, the same control extends for the complete polytopic region.

3.2.3 System Reduction

With detailed synchronous machine and control models such as those of exciters, PSSs, and governors, the size of A_v and Γ_v as defined in (3.9) can become large even for a moderately sized system. It is often desirable to obtain a lower order plant model to ensure that it is not too complex for practical implementation. In this work, selective modal analysis (SMA) [121] is used for simplifying the complicated LTI models of the dynamic system,

as it provides the advantage of retaining the physical attributes of the state variables that are strongly linked to the LFOs. Furthermore, as the identity of the state variables is known $a\ priori$, SMA can be used to exploit the state clustering and coherency techniques [122], [123]. A rigorous comparison of the modal analysis of the full and reduced order systems ensures that only those system states that have little effect on the input-output behavior of the system are discarded. The closed-loop reduced order system can be approximated by:

$$\dot{x_1} = (A_1 + M + NK_{LMI-poly})x_1 \tag{3.11}$$

where A_1 represents the portion of the matrix A_v consisting of important system states, M and N are the fixed matrices denoting the non-significant dynamics of the system, and $K_{LMI-poly}$ is the gain matrix obtained using LMI control.

3.2.4 Selection of Control and Stabilizing Signals

Measurements that can retain the good observability of the oscillatory modes of interest (inter-area modes) are good candidates for selection of the input stabilizing signals. The control of the inter-area modes can be effectively provided through generators whose states actively participate in these modes. For indicating the relative contribution of a state variable, x_p, in the i^{th} mode, the *most accepted* solution is the speed (angle) participation factor, pf_i. It is given by the combination of both L_i and R_i, where L_i and R_i are the left and right eigenvectors, respectively, of mode i [13]. Mathematically, pf_i is defined as,

$$pf_i = \begin{bmatrix} pf_{1i} \\ pf_{2i} \\ \vdots \\ pf_{mi} \end{bmatrix} = \begin{bmatrix} R_{1i}L_{i1} \\ R_{2i}L_{i2} \\ \vdots \\ R_{mi}L_{im} \end{bmatrix} \tag{3.12}$$

High values of speed (angle) participation factors (PFs) of generators present in different areas for the same mode indicate the presence of an inter-area mode of oscillation.

3.2.4.1 Limitation of Using PFs Directly

The idea here is to control those generator states which are influencing more number of modes by having higher participation in them. However, the underlying relationship between ensuring stability and minimizing control effort cannot be discovered from PFs directly. This is realized from the PFs of the no-contingency case of the 16 machine, 68-bus system New England-New York Interconnected Power System [4], which are shown in Fig. 3.4. From Fig. 3.4 one can realize that by observing the PFs directly, it is very difficult to draw any definite conclusions regarding the $minimum$ number of states that must be controlled for improving the system's stability. Consequently, a methodology is proposed here that uses PFs as a starting point for control effort minimization.

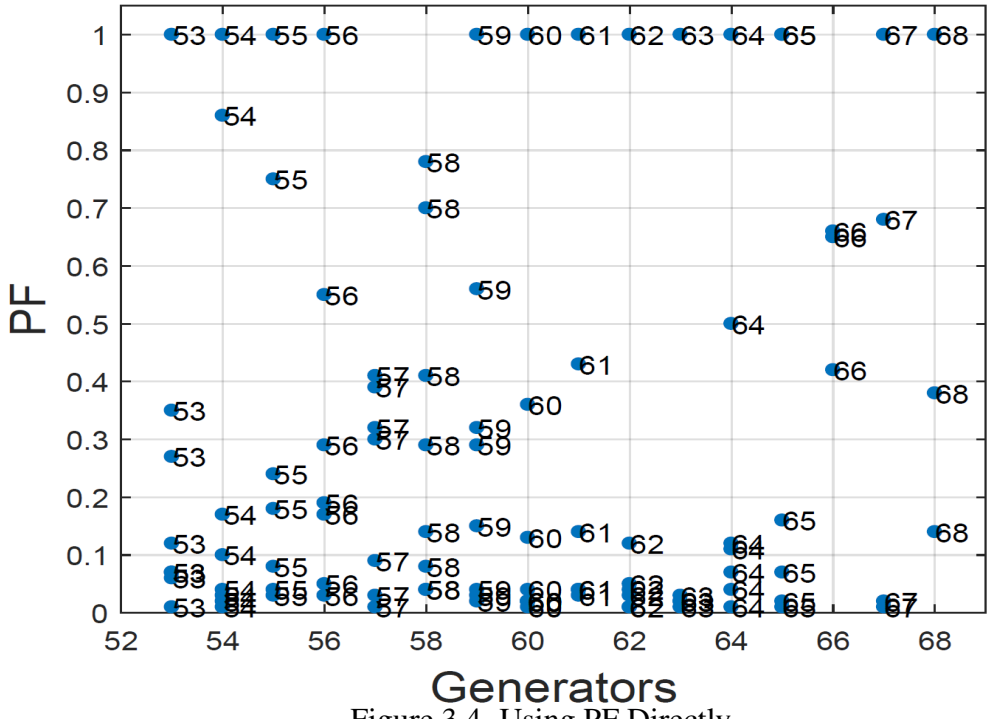

Figure 3.4. Using PF Directly.

3.2.4.2 Methodology for Minimizing Control Effort

In this section, a clustering approach is proposed for allocating minimal control for robust damping of LFOs. Here, clustering implies grouping different generators according to their PFs. Once the clusters are formed, the next step is to identify the set of $non\text{-}critical$ generators, S, which can be dropped from the control group responsible for damping the oscillations. That is, the size of A_1 matrix obtained from SMA is *reduced further* (henceforth referred to as enhanced SMA) and LMI control is applied using the δ and ω of the remaining generators. To ensure quality, a preprocessing step is included before the data is fed to the clustering logic. The schematic of the proposed scheme is shown in Fig. 3.5.

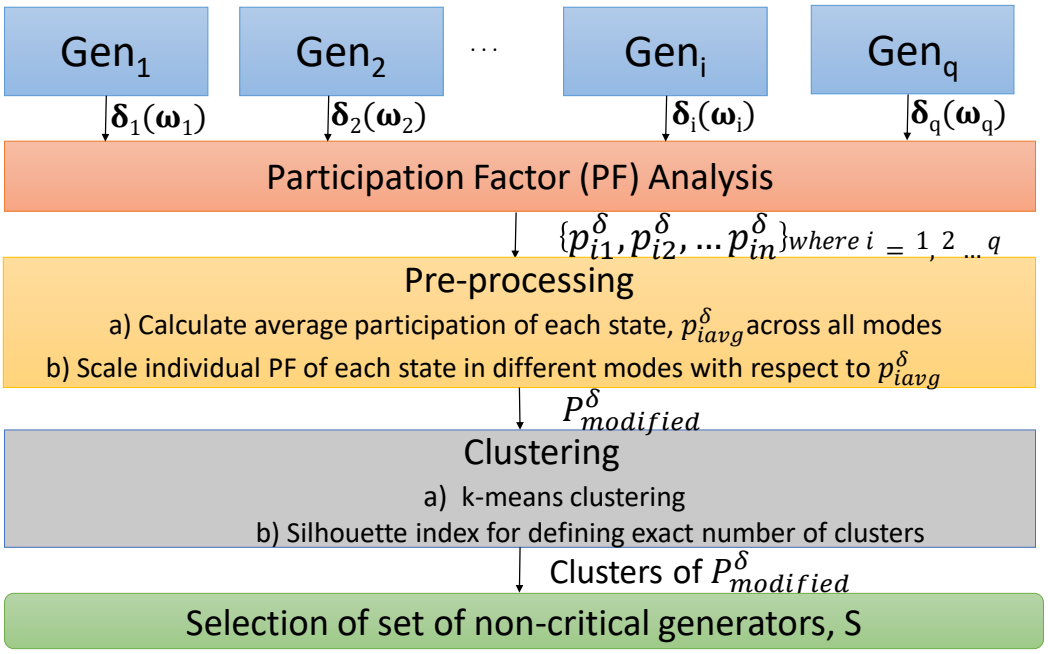

Figure 3.5. Proposed Methodology for Generator Selection.

1)*Pre-processing Step*

This step ensures that an appropriate association is made between the critical modes and the states associated with them.

a) First, the average participation of each state across all the modes (including critical

as well as non-critical) is calculated. If there are a total of q generators in a system with n LFOs, of which Θ are critical modes, and participation of each generator state, i (for instance, speed), in any mode, j, is p_{ij}^{δ}, then

$$p_{iavg}^{\delta} = \frac{p_{i1}^{\delta} + \cdots + p_{ij}^{\delta} + \cdots + p_{in}^{\delta}}{n}, i = 1 \cdots q \tag{3.13}$$

where $q = n + 1$; the relationship between q and n (given by $q = n + 1$) stems from the fact that there is one reference machine in the system.

b) Scaling of the PFs for each state is now done with respect to its calculated average PF as shown below:

$$p_{imodified}^{\delta} = (\frac{p_{i1}^{\delta}, p_{i2}^{\delta}, \cdots p_{in}^{\delta}}{p_{iavg}^{\delta}}), i = 1 \cdots q \tag{3.14}$$

The motivation of using $p_{imodified}^{\delta}$ is to obtain the PFs with respect to the generator's average contribution in all the modes.

2)*Clustering Technique*

After the implementation of the data transformation step, the modified PFs corresponding to different generators are clustered using $k-means$, which classifies them into κ user-defined groups, $\{\Lambda_1, \Lambda_2, ...\Lambda_\kappa\}$. The clustering is done by minimizing the sum of squares of distances (SSE) between the data points, $P_{modified}^{\delta} = \{p_{1modified}^{\delta}, p_{2modified}^{\delta}, ...p_{qmodified}^{\delta}\}$ and the centroids of corresponding clusters, $\{c_1, c_2, ...c_\kappa\}$, as shown in (3.15),

$$SSE = \sum_{\epsilon=1}^{\kappa} \sum_{P_{modified}^{\delta} \subset \Lambda_\kappa} (c_\epsilon - P_{modified}^{\delta})^2 \tag{3.15}$$

where $c_\epsilon = \frac{1}{t_\epsilon} \sum_{P_{modified}^{\delta} \subset \Lambda_\kappa} P_{modified}^{\delta}$, and t_ϵ is the number of datapoints contained in cluster Λ_κ.

The datapoints are then assigned to the clusters with the nearest centroid, after which the centers are recomputed. The process is repeated until the assigned centroids do not change [124]. The output of $k-means$ is influenced by κ and the choice of initial cluster centers. To ensure quality of partition, $silhouette\ index$ is employed [125]. The silhouette ranges from -1 to 1, where a high value guarantees a well-defined clustering result. The idea is to run $k-means$ clustering on dataset for a varying range of values of κ and then for each of its values, calculate SSE. It is then plotted against κ, with the location of the knee point on the curve indicating the optimal number of clusters formed. The set of $non\text{-}critical$ generators, S, are now selected based on the following logic:

a) Highest priority is given to the generator influencing $minimum$ number of system modes.

b) The next priority is given to a generator with $higher\ |p^{\delta}_{imodified}|$, which denotes that contribution of a generator in few modes may be high, but its average contribution in other modes is low.

c) Care is taken to ensure that none of the generators having $high$ value of $p^{\delta}_{ikmodified}$ in $multiple$ critical modes, λ_{crit} are eliminated from the control where $k = 1, 2, ...\Theta$.

The clustering technique is applied to the same system, whose PFs are shown in Fig. 3.4. The validation of the number of clusters has been done by plotting the Silhouette index against the number of clusters, as shown in Fig. 3.6. For this system, $q = 16$, $n = 15$, $\Theta = 4$, and $P^{\delta}_{modified}$ is clustered into $\kappa = 5$ groups (see Fig. 3.7). Sets of first three generators for each critical mode selected on the basis of decreasing order of $p^{\delta}_{i\Theta}$ are: $\Psi_1 = \{g_{67}, g_{66}, g_{68}\}$, $\Psi_2 = \{g_{65}, g_{61}, g_{58}\}$, $\Psi_3 = \{g_{68}, g_{66}, g_{67}\}$, and $\Psi_4 = \{g_{65}, g_{67}, g_{66}\}$. Elements of critical generator set, Ψ_{crit}, are defined by $\Psi_a \cap \Psi_b$ where $a, b \in 1, ..\Theta$. On doing this, the top three machines in Ψ_{crit} come out to be g_{65}, g_{66}, and g_{67}. Combining this knowledge with the modified PFs shown in Fig. 3.7, it is clear that g_{63}, g_{62}, and g_{60} have high values of $p^{\delta}_{imodified}$, but low participation in λ_{crit}. Hence, for the base-case of

the 68-bus system, these are the first three machines that can be dropped from the control feedback.

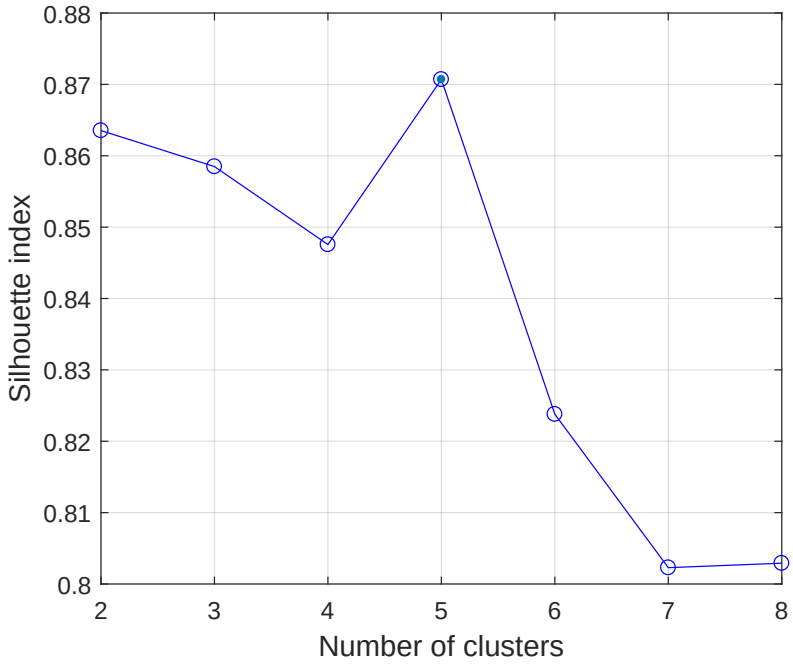

Figure 3.6. Validation of Number of Formed Clusters.

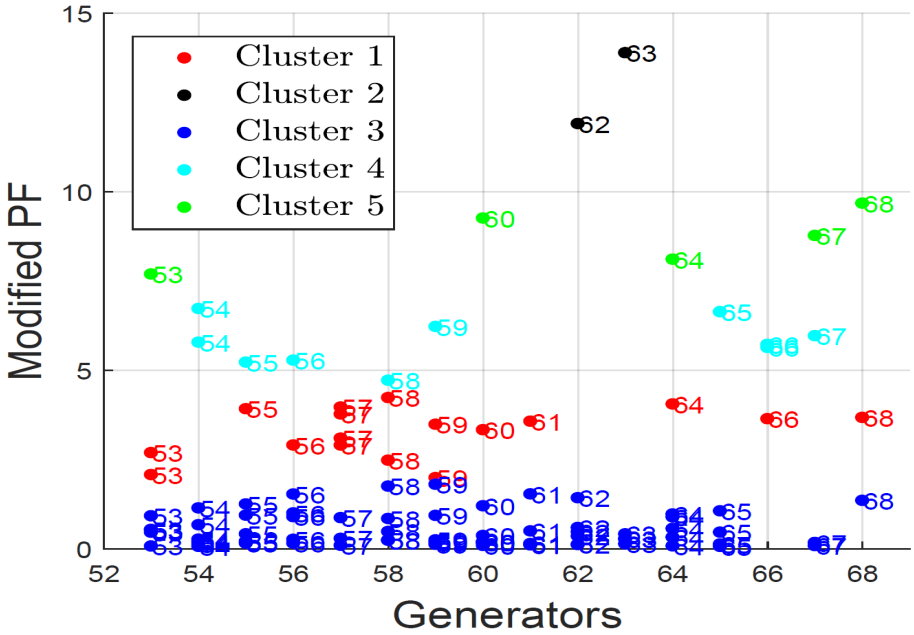

Figure 3.7. Using Modified PF.

3.2.4.3 Proposed Minimal Control Algorithm

The proposed algorithm for minimizing the control effort using the enhanced SMA (a combination of SMA, modified PFs, and $k-means$ clustering technique) is shown in **Algorithm 2**. It is described as follows:

1. Perform the SMA to reduce the size of the system to relevant system states, i.e., speeds and angles of the machines present in the system.

2. Next, identify the critical modes, λ_{crit}, and the generators having maximum participation in these modes using the PFs.

3. Identify the initial set of non-critical generators, $\Psi^{init}_{non-crit}$, with the help of modified PFs and $k-means$ clustering algorithm as explained in Section 3.2.4.2.

4. Drop a single non-critical machine belonging to $\Psi^{init}_{non-crit}$, and check for the following conditions:

   - Perform SMA for the reduced-order system (obtained after dropping the machine) and check for its convergence.
   - If SMA converges, apply a rudimentary linear quadratic regulator (LQR) (which is a special case of H_2 control) to find if the closed-loop system is stable.

5. If the application of the LQR control results in a stable system, repeat step 4) iteratively for all the machines in $\Psi^{init}_{non-crit}$.

6. If SMA fails to converge or the application of LQR destabilizes the system after dropping a machine, the preferred size of the reduced-order system, A_1 would be the stable and converged system obtained in the previous iteration.

7. Repeat the steps 1)-6) for the system operating at different conditions. It is to be noted that with the use of enhanced SMA, the obtained size of the A_1 matrix varies with different OCs. However, as the focus is to design a single $K_{LMI-poly}$ for a range of operating points, similar generator states were retained for each of the vertices (OCs) of the polytope, as a result of which no further reduction in the dimension of A_1 matrix could be achieved. Hence, the system is reduced to minimum state-space representation, which makes it fully controllable and observable.

8. Check if the application of the proposed $K_{LMI-poly}$ results in minimum damping, ζ_{min}, for the closed-loop polytopic system.

Algorithm 2: Minimum (Primary) Control Set Selection

Perform the SMA to reduce the size of the system to speeds and angles of all the machines.
First, identify λ_{crit} and set of generators,Ψ_{crit}, responsible for them.
Segregate the generators into multiple clusters using $k-means$ clustering in conjunction with the $silhouette\ index$.
Define the sequence of generators, $\Psi^{init}_{non-crit}$ consisting of p elements as $\{gen_{non-crit1}, gen_{non-crit2}, ...gen_{non-critp}\}$.
for $d \leq p$ **do**
    Drop $Gen_{non-critd}$ and check for following conditions:
        a) Find if SMA converges, i.e. $\lambda_{A_1+M} \approx \lambda_A$ where λ_A and λ_{A_1+M} are eigenvalues of original and reduced system, respectively. **If** yes, goto step b) **else** goto step c).
        b) Find K by applying a rudimentary LQR control to $A_1 + M$ and check if $\zeta_{A-B*K} \geq \zeta_{min}$.
        c) Set $d = d + 1$.
end
The final set of non-critical generators selected for dropping would be $\Psi^{final}_{non-crit}$ composed of g elements. The preferred size of A_1 would be the stable and converged system obtained in the previous iteration.
Set $\Gamma_{polytope} = \{\Gamma_1, \Gamma_2, ...\Gamma_t\}$, where t = operating condition, and design LMI gain, $K_{LMI-poly}$, for it.
Check if $\{\zeta_{\Gamma_{cl1}}, ..\zeta_{\Gamma_{clb}}...\zeta_{\Gamma_{clt}}\} \geq \zeta_{min}$ where $\zeta_{\Gamma_{clb}}$ is damping of closed-loop system, Λ_b.

3.2.5 Flexibility of Feedback Selection

In this report, flexibility is defined as taking PMU measurements from alternate locations in case the primary locations encounter a problem. The selection of the alternate feedback control signals is performed using **Algorithm 3**, which is briefly summarized below. To select an alternate set of control signals, a set $\Phi = \Psi^{init}_{non-crit} \backslash \Psi^{final}_{non-crit}$ is defined,

where $\Psi^{init}_{non-crit}$ denotes the initial set of non-critical generators identified using $k-means$ clustering and modified PFs, while $\Psi^{final}_{non-crit}$ signifies the primary set of non-critical machines that are dropped from the control group using enhanced SMA (refer to **Algorithm 2**). With the help of an iterative process, a single generator in the list of non-critical generators, $\Psi^{final}_{non-crit}$ is replaced with a generator belonging to Φ to obtain a new sequence, Ψ^{n}_{final}. The system obtained after dropping the generators belonging to Ψ^{n}_{final}, is tested for two conditions, namely, convergence of SMA and a stable closed-loop system after the application of LQR control. If the system satisfies these two conditions, then the *retained* group of machines that result in a stable and converged system form the alternate control set of critical generators, Ψ^{alt}_{crit}. On comparing the primary set of critical generators, Ψ^{prim}_{crit}, with Ψ^{alt}_{crit}, it is observed that the two sets have identical elements except one. The new element in Ψ^{alt}_{crit} serves as an alternate location for taking the feedback in case the PMU at the corresponding generator in Ψ^{prim}_{crit} encounters a problem. Applying this process iteratively, multiple sets of alternate feedback signals can be generated. Based on output of **Algorithm 3**, the coordinated controller can switch between the primary and alternate control sets in case the data quality of the primary set deteriorates (for instance, due to failure of a PMU) [126].

Algorithm 3: Alternate Control Set Selection

1 Define a set, Φ, consisting of h elements as $\Psi^{init}_{non-crit} \setminus \Psi^{final}_{non-crit}$. Here $h = p - g$, where p and g specify the number of elements in $\Psi^{init}_{non-crit}$ and $\Psi^{final}_{non-crit}$, respectively. Set $n = 1$.
2 **for** $f \leq h$ **do**
3 **for** $c \leq g$ **do**
4 Replace the c^{th} element of $\Psi^{final}_{non-crit}$ with f^{th} element of Φ to obtain new sequence Ψ^{n}_{final}.
5 Repeat Steps 6.a)-6.c) mentioned in **Algorithm 2**.
6 Set $n = n + 1$.
 end
end

An example of this switching is shown in Fig. 3.8 for a system with more than 3 generators. Let two solution sets of non-critical generators obtained as the outputs of

Algorithm 2 (primary set) and **Algorithm 3** (alternate set) be given by $\Psi^{prim}_{non-crit} = \{gen_1, gen_2, gen_k\}$ and $\Psi^{alt}_{non-crit} = \{gen_1, gen_2, gen_j\}$, respectively. Now, if the PMU at gen_j is down, then the coordinated controller will use $\Psi^{alt}_{non-crit}$ instead of $\Psi^{prim}_{non-crit}$ for the selection of feedback signals, as gen_j belongs to $\Psi^{alt}_{non-crit}$ and hence, can be dropped from control. Similarly, if the PMU at gen_k is down, the coordinated controller will use $\Psi^{prim}_{non-crit}$, as feedback signal from PMU at gen_k is not required for control purpose. Therefore, from a control perspective, backup observability of gen_j can be provided by PMU at gen_k, and vice-versa. It must be noted here that for the given example, if PMUs at bus k and j fail simultaneously, then the proposed CWADC may not be able to provide the requisite damping. Similarly, it may not be possible to find alternates for *all* non-critical generators. The wide-area input stabilizing signals used in the design of CWADC are the

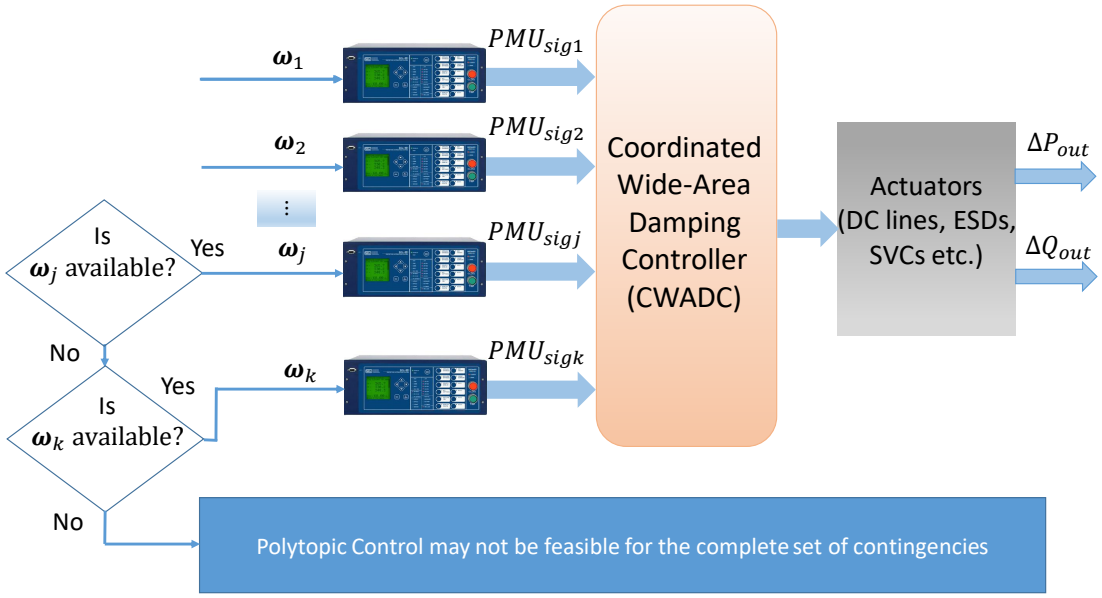

Figure 3.8. CWADC with Flexible Feedback.

angles and speeds of critical generators obtained after performing an enhanced SMA, since they provide a reasonably good approximation to the frequencies of swing modes. For the highest controllability of these modes, CWADC sends the control signals to the SVCs, DC-

SDCs, and exciters of the generators belonging to the critical set, which together constitute the second control level in the bi-level closed-loop control system. The PSSs belonging to the non-critical set of generators remain fully decentralized.

The control law for the LMI polytopic controller with gain matrix, $K_{LMI-poly}$, is given by:

$$u = K_{LMI-poly} x^l \tag{3.16}$$

where

$$K_{LMI-poly} = \begin{bmatrix} K_{11} & K_{12} & K_{13} & .. & K_{1l} \\ K_{21} & K_{22} & K_{23} & .. & K_{2l} \\ .. & .. & .. & .. & .. \\ K_{i1} & K_{i2} & K_{i3} & .. & K_{il} \end{bmatrix} \tag{3.17}$$

where x^l denotes the reduced number of states obtained using the enhanced version of SMA, while i signifies the number of controls present in the system.

CHAPTER 4

AI-BASED IMPLEMENTATION

Fig. 4.1 shows the bi-level design framework for DNN-CWADC as well as DRL-CWADC, which enables the local controls to act based on both local as well as wide-area signals. Both the AI-based control agents are trained offline using the selected stabilizing signals measured by geographically distributed PMUs. Depending upon the current OC, the appropriate controller is put into online operation mode, which then sends the control signals to AVRs of selected machines, SDCs installed on DC lines, and SVCs as shown in Fig. 4.1. On receiving the respective control commands, the controllers act in a coordinated fashion to increase the damping of the oscillatory modes. Limiter action is modeled for every local control (refer Fig. 4.1) to avoid unexpected violations in the system's response in real-time. In addition, H_2/H_∞ optimization constraints and constrained action search method are enforced for the two control schemes (as explained in Sections 4.1 and 4.2.3) to facilitate the inherent safety in the design processes.

4.1 Deep Neural Network (DNN) Implementation

In order to implement a polytopic controller, it is essential to have a mathematical model of the system. However, the model may become unreliable over time, especially due to increased uncertainties in system parameters/operation. Furthermore, for an OC lying outside the designed polytope, the polytopic controller does not guarantee satisfactory per-

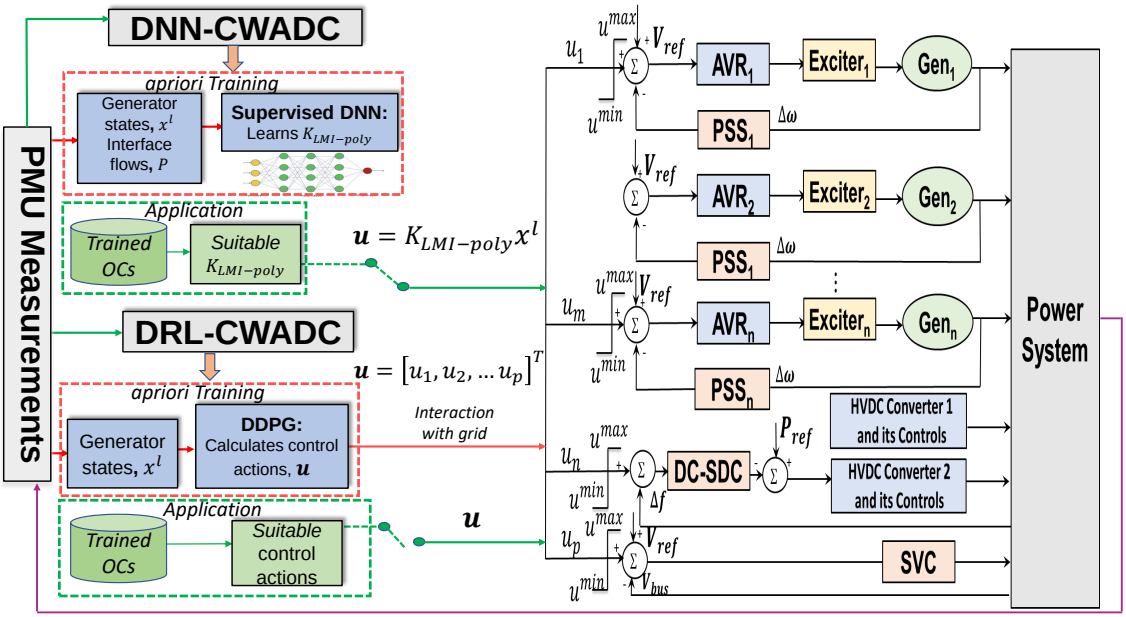

Figure 4.1. DNN-CWADC and DRL-CWADC Based Controls.

formance [127]. A data-driven DNN-CWADC can overcome these issues by handling the changes in the scheduled OCs through a combination of multiple polytopic controllers in an adaptive fashion. The non-linear approximation capabilities of its multi-layer architecture also render it highly effective in the control of dynamic systems without directly using their actual state-space models.

The DNN-CWADC can be seen as a supervised mapping operation from oscillation modes corresponding to the different OCs to the desired coordinated control actions, where the DNN is chosen to realize such a mapping. The resulting controller will provide the desired damping ratio at a much faster rate in comparison to the traditional H_2/H_∞ LMI control [42], since most of the computations occur during the initial offline training. The methodology used to design the proposed DNN-CWADC is as follows:

a) *Stage I-Dataset Creation for DNN-CWADC*: An important aspect of this work that sets it apart from previous AI-based control efforts is the creation of a dataset using *multiple polytopes* [76], [77]. A multi-polytopic control ensures the stability of the entire region

encompassed by the OCs associated with the polytopes using a limited number of controller gains [44]. Here, stability implies that the critical LFO modes have the requisite minimum damping, ζ_{min}.

The methodology for assessment of oscillatory stability of a transfer path or an area is undertaken with a set of appropriately prepared power flow cases by following these steps:

1. Develop a set of power flow cases for the selected path: The first step is to select a suitable base case. Different cases are then developed by modifying the generation and load patterns of the base case such that additional power transfers are created in the interface. The procedure is stopped when the steady state limit of the system is violated. The steady state limit is defined as the maximum amount of power that can be transferred between two or more areas in a power system within the stability margins. The following must be considered when developing these cases:

    - The output of the swing machine accounts for the losses in the system. It should not be allowed to exceed its active and reactive power capability, and instead, other generators should be re-dispatched to pick up the additional load resulting in the increased power flow in the interface.
    - The generation supply should come from the generators (can be synchronous or inverter-based generation) that place the highest stress on the path of interest.

    Some of the additional cases can be created by changing the network topology taking one or more components out of service, i.e., creating $N-1$ and $N-2$ contingencies at a time.

2. Perform post-disturbance eigenvalue analysis: For each of the above operating cases, solve a post-transient power flow and perform eigenvalue analysis. Record the boundary points found for which violations in the steady state limit of the system occurs,

i.e., if any of the resulting eigenvalues for the analyzed case has negative damping.

3. Obtain the reduced-order system for the above cases using the enhanced SMA (refer to Section 3.2.4.3).

4. Create multiple combinations of three-dimensional polytopes for the reduced-order system obtained in the previous step using H_2/H_∞ optimization and pole-placement constraints (refer to Section 3.2.2). Each polytope consists of three system matrices corresponding to three different OCs, reduced to the speeds, ω, and angles, δ, of the critical generators, CG, using enhanced SMA [128]. For the better acquisition of modal information of the critical oscillation modes, these attributes are further augmented by the active power flows on critical transmission paths. The gains of the designed polytopic controllers, $K_{LMI-poly}$ (refer (3.17)), are the outputs of the DNN.

Note that it is possible to create higher-dimensional polytopes (i.e., more than three vertices/OCs). However, doing so would bring additional challenges related to dealing with substantial feature sets, more sophisticated architecture of DNNs, and higher computational costs.The overall dataset is divided into three subgroups: training dataset for model building, validation dataset to measure the overfitting of the model, and testing dataset to evaluate the performance of the model for inputs external to those for which it was trained (extrapolation).

b) *Stage II-Training of DNN*: Training of the DNN is performed using the normalized input dataset consisting of δ and ω of CG, and power flows on important tie-lines. This offers two main advantages: (a) DNNs can be applied to different systems with different ranges of input features, and (b) the back-propagation algorithm used in the DNN can be made to converge faster. Another important consideration for the successful performance of the DNN is an appropriate choice of the different hyperparameters, such as the number of hidden layers and neuron counts, activation function, regularization, batch normalization,

and training parameters. In this work, ReLU is chosen as an activation function because it is resilient against the vanishing gradient problem [129]. During training, the mean-absolute error loss function is minimized using the adaptive moment estimation (ADAM) optimizer. To optimize the number of hidden layers and neurons, and learning rate for the ADAM optimizer, Bayesian optimization is used [130]. Bayesian optimization attempts to find an optimal combination of parameters in a minimum number of steps, and is particularly suited for high cost functions. It works by constructing a posterior distribution of functions (Gaussian processes) that best describe the objective function to be optimized. The optimization algorithm balances exploration and exploitation to direct the sampling to the regions in parameter space where an improvement over the current best observation is likely. The problem of data overfitting is circumvented by employing $L2$ regularization. It works by adding a weight penalty to the objective function using another hyperparameter, χ, that has a value between 0 (no penalty) and 1 (full penalty). If the penalty is high, the model will underestimate the weights and underfit the learning problem. If the penalty is too low, the model will overfit the training data.

4.2 Deep Reinforcement Learning (DRL) Implementation

4.2.1 Proposed Algorithm

The performance of the DNN-CWADC depends on the generation of large training datasets, which albeit done offline, might still be computationally cumbersome to create for high-dimensional dynamic systems. Additionally, the generated training patterns may not always sufficiently capture the non-linear behavior of the power systems operating under highly stressed conditions. One approach to design a real-time wide-area closed-

loop control for systems with unknown dynamics is by using DRL.

The damping of LFOs involves a continuous action space, which cannot be implemented using techniques based on discrete action space such as DQL, since:

- for a large set of actions, computation of Q-value for every action that yields the maximum reward would require optimization at every time step.
- with an increase in the number of controllers, the discretization of a bigger action space will lead to an exponential increase in the number of actions.

To circumvent these issues, in this work, a modified version of DDPG [112] is developed (explained in Section 4.2.3) that can learn policies in a continuous action space.

A DDPG-based CWADC aims to provide continuous control actions so that all critical modes have damping greater than ζ_{min}. The selection of the observation state-space, $\mathcal{S}$, is done using the same logic as presented in Section 4.1, i.e., $\mathcal{S} = \{\Delta\omega^i | i = 1, ..., CG, \Delta\delta^i | i = 1, ..., CG - 1\}$. Note that angles are computed with respect to the reference machine, and therefore, the vector for the angle states does not include the reference machine.

In this work, LFO damping is achieved through modification of the control set-points of the local damping controls on receiving the supplementary stabilizing signals from the designed CWADC. Thus, the action space, $\mathcal{A}$, is expressed as: $\mathcal{A} = \{a^j | j = 1, ..., C\}$, where C corresponds to the number of local controls coordinated using the DRL agent. The proposed architecture of the DNNs used inside the DRL module is as follows: each DNN uses an input layer, two hidden layers, and an output layer. The inputs and outputs to the actor networks are the observation states, $s_t^i \in \mathcal{S}$ and $i = 1, ..., 2CG - 1$ and controller actions, $a_t^j, \forall a \in \mathcal{A}$ and $j = 1, ..., C$ at each time step. For the critic networks, the input layers include the observation states and the action values computed using actor networks, while their output layers output the fitted Q-values.

To improve the exploration capabilities of DDPG policies, noise is added to each action

during the training period. The authors of the original DDPG paper [112] recommended using OU noise (a Markov process), but this procedure led to the saturation of the actions close to their boundary. Hence, independent mean-zero Gaussian noise, $\mathcal{N}(0, \sigma^2)$ is used to overcome this challenge [131]. The decay rate for the sigma, σ_d, is used to balance the exploration and exploitation of actions.

4.2.2 Reward Design

The reward function is aimed at maximizing the damping ratio, while penalizing the agent proportionally for violating the stability margins. The reward, r_t, at each time step, t, is defined as the linear combination over the deviations in speeds and angles of the critical generators being controlled and the cost of applying additional damping action to local control j:

$$r_t = \begin{cases} \text{Huge Penalty} & \text{if power flow diverges, else} \\ -\sum_{i=1}^{CG} d_1 \Delta\omega^i - \sum_{i=1}^{CG-1} d_2 \Delta\delta^i - C_t^j \end{cases} \tag{4.1}$$

where

$$C_t^j = \begin{cases} d_3 & \text{if } T_{cr} < t \leq T_{cr} + t_\alpha \\ d_3 * 2 & \text{if } T_{cr} + t_\alpha < t \leq T_{cr} + t_\beta \\ d_3 * 3 & \text{if } T_{cr} + t_\beta < t \end{cases} \tag{4.2}$$

where T_{cr} is the time instant of removal of the contingency. The choice of time instants, t_α, t_β, and weight factors, d_1, d_2, and d_3 is made based on the expert knowledge of the system as well as trial-and-error selection [90]. Note that the reward function is set to incorporate a huge penalty if any divergence issues related to the power flows are encountered.

4.2.3 Safety Guarantee for DRL Exploration

To ensure the safe control of the power system, it is necessary to constrain the observed states. This is especially important when examining the DRL-based control schemes, which predominantly use exploration for determining improved actions. Without prior knowledge of its environment, the actions of the DRL agent can lead to unnecessary system violations. This is also true even when rewards are carefully designed to penalize unexpected system behavior. This is because in order for the DRL agent to learn to avoid learning an undesirable behavior, it will have to violate the system constraints a few times. Therefore, in this work, a safety mechanism is proposed for constraining the action of the DRL agent to a confined region during the learning process, while circumventing the problem of controller output saturation. The safety mechanism consists of selecting $K_{LMI-poly}$ (refer (3.17)), generated using H_2/H_∞ optimization and pole-placement constraints, as an initial control input for the DRL-CWADC. Thus, the control action of each of the local controls, $a^j \in \mathcal{A}$, is bounded in the exploration phase using (3.16) as: $a^j \in [-u^j, u^j]$ and $j = 1, ..., C$. This method guides the exploratory actions of DRL-CWADC in the direction of feasible and safe operating zones. Consequently, the size of the action space is reduced, thus leading to faster convergence of the DDPG algorithm. The proposed algorithm is referred to as *bounded exploratory control*-based DDPG (BEC-DDPG).

4.2.4 Bounded Exploratory Control-based DDPG (BEC-DDPG)

The overall implementation of the DRL-CWADC is shown in **Algorithm 4**. The selection of the input stabilizing signals is performed using the enhanced SMA explained in **Algorithm 2**, while the setpoints of different local controllers are selected as the outputs. At the beginning of the DRL training process, four DNNs, i.e., actor (π), target-actor (π^{targ}), critic (μ), and target-critic (μ^{targ}) with different set of random weights and the

replay buffer, $\mathcal{D}$, are initialized (refer Section 2.3.4). The purpose of establishing $\mathcal{D}$ is to break the temporal pattern between the training data and increase the robustness of training. For the M episodes, the DDPG agent follows the process listed below.

1. At the start of the episode, a power flow is solved to obtain the initial states ($\Delta\omega^i$ and $\Delta\delta^i$) of CG. Also, the exploration noise from a Gaussian distribution, $\mathcal{N}$, is randomly initialized with zero mean and σ as the standard deviation.

2. Next, a loop for a defined number of steps per episode begins starting with the action predicted by the π network (though the initial outputs of the networks are nearly zero due to the manner in which the weights are initialized [112]) combined with the exploration noise. This, however, implies that the random exploration noise overpowers the output actions at the beginning of the training period. Thus, the limiter action is applied to bound the control action within a safe region using the constrained $K_{LMI-poly}$ generated using H_2/H_∞ optimization and pole-placement constraints. This process is repeated for each of the local controls that are to be coordinated with the help of the DDPG agent.

3. At each timestep, each of the supplementary control actions, $a_t^j \in \mathcal{A}$ is executed on the environment, returning the next generator states, $s_t^i \in \mathcal{S}$, and the reward. This transition is stored in $\mathcal{D}$. The DDPG agent will update the weights of the critic (Q) network followed by those of the actor π network, once $\mathcal{D}$ has a mini-batch size of $\mathbf{B}$ (refer $\mathbf{Algorithm\ 1}$) or else the agent will continue to interact with the environment.

4. The weights of the target networks, μ^{targ}, and the actor, π^{targ}, networks are updated slowly using ρ parameter. The next step is to update the σ using the σ_d (which dictates how fast the exploration noise decays) to balance between the exploration and the exploitation.

The above process is repeated until the training converges according to a convergence criteria discussed in Section 6.2.

Algorithm 4: BEC-DDPG based DRL-CWADC

Input: First, identify the $\mathbf{input}$ generator states using enhanced SMA,
$\mathcal{S} = \{\Delta\omega^i | i = 1, ..., CG, \Delta\delta^i | i = 1, ..., CG - 1\}$.
Output: Select the $\mathbf{outputs}$ as the supplementary stabilizing signals of the different controllers,
$\mathcal{A} = \{a^j | j = 1, ..., C\}$.

```
Initialize the parameters for the BEC-DDPG agent: actor,
target-actor, critic, target-critic networks with random weights.
Initialize the experience replay buffer, 𝒟.
for episode, e ≤ M do
    Initialize the power system environment and obtain initial state
    observations, s_0^i ∈ 𝒮.
    Initialize a random Gaussian noise, 𝒩(0, σ^2).
    Select a suitable K_LMI−poly controller based on the
    domain-knowledge of the system.
    for timestep, t ≤ T do
        for controls, j ≤ C do
            Calculate control action, a_t^j, ∀a ∈ 𝒜 using DDPG.
            Apply limiter action to a_t^j, where a_t^j ∈ [−u_t^j, u_t^j]
            and u_t = K_LMI−poly[s_t^1, s_t^2 ... s_t^{2CG−1}]^T.
        end
        Execute the set of actions and observe the reward, r_t, and
        next state, s_{t+1}, for each of the CG.
        Store the transition {s_t, a_t, r_t, s_{t+1}} in 𝒟 for each of the
        controls and CG.
        Sample a random minibatch of B transitions from 𝒟.
        Update the network parameters.
        Update σ as σ = σ_d * σ.
    end
end
```

4.3 Adaptive Control Design

Fig. 4.2 gives a detailed procedure for selecting the suitable CWADC. Once the *a priori* design of the DNN-CWADC and DRL-CWADC is completed using the methodology described in Sections 4.1 and 4.2, respectively, one of the two controllers can be put into service by following these steps:

Step a) Examine the angles of relevant generator buses from the PMU measurements to check if the current OC represents a small perturbation or a severe transient disturbance [37]; the latter implying inability to perform linearized system analysis. If the current OC

does represent a severe transient disturbance, go to step d), else go to step b).

Step b) If the current OC resembles a small perturbation, and belongs to the database of the trained/test polytopic cases, then put the respective DNN-CWADC (either *trained* or *extrapolated*) into service. The criteria for choosing a suitable polytopic domain are described in [43] and [44]. For an OC that is not part of any of the polytopes created during Stage I (Section 4.1), the representative DNN-CWADC corresponds to the polytope whose distance from the current OC is smallest.

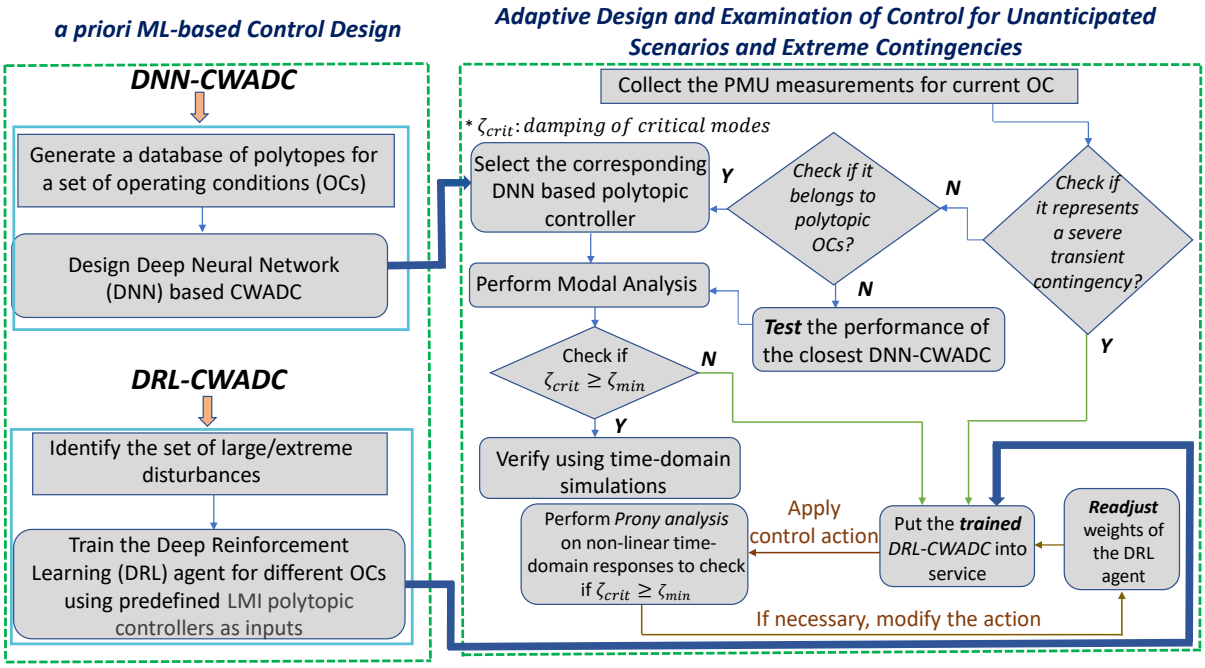

Figure 4.2. Selection of the Desired Controller.

Step c) Check if $\zeta_{crit} \geq \zeta_{min}$ after the application of the selected DNN-CWADC. If yes, then validate it using time-domain simulations.

Step d) If the system is operating under a larger contingency or the performance of the DNN-CWADC is not found to be satisfactory (i.e., $\zeta_{crit} < \zeta_{min}$), the trained DRL-CWADC is selected to provide the optimal control actions for keeping the system stable. The resulting impact of the control actions is verified by performing Prony analysis [132].

Step e) Based on the results obtained in step d), the DRL-CWADC may have to adapt to

the unknown system changes occurring in its environment to increase its operating range. To activate the flexible decision-making process for DRL control, its training parameters, i.e., the weights of the four actor-critic networks can be readjusted (fine-tuned on-the-fly) to keep pace with the changing observations in real-time. The readjustment is done through the continuous interaction between the power system environment and the DRL module. These interactions will continue until the desired damping response is obtained. If the designed DRL control is unable to provide the requisite control action, reformulation of the DRL problem is required. It is to be noted that the two controls will not act simultaneously.

CHAPTER 5

RESULTS FOR A SINGLE POLYTOPIC CONTROL

The technique proposed for the design of a single polytopic controller is illustrated using two test systems: a (smaller) 16-machine, 68 bus system [4], and a (larger) reduced-order model of the WECC system [133]. Numerical experiments on these two test systems were performed using a 64-bit operating system with Intel (R) Core (TM) i7-4510U CPU, 2.6 GHz processing speed and 8 GB RAM. The modeling, power flow, linearization based eigenvalue analysis, and controls were accomplished by writing several routines in MATLAB's Power System Toolbox (PST), DSA Tools, and PSLF software [98], [99], [134]. The scope of application of the proposed method was limited to performing modal analysis for the smaller system, however, for the larger system, both eigenvalue analysis and time-domain simulations were conducted.

5.1 Small-size power system-16 machine, 68 bus system

The 16 machine, 68 bus system represents the reduced-order equivalent model of five separate areas of the New England-New York Interconnected Power System. The generators were represented by their sub-transient reactance models and equipped with manually tuned fast DC Type 1 exciters. The parametric details for the different models used in the cases can be found in [134]. Two types of actuators, namely, DC lines and SVCs were

employed for implementing the control [41]. Three HVDC lines are connected in parallel with the tie-lines between buses 1 and 27, 41 and 42, and 8 and 9. Three SVCs are installed at buses 27, 46, and 51. To test the application of the proposed method, the DC lines are modeled as active power and reactive power injections, while SVC is modeled as a reactive power injection.

5.1.1 Application of the Proposed Control Scheme

For investigating the effect of the proposed control scheme, seven contingency cases listed in Table I were created.

Table 5.1. Test Cases for 16 Machine, 68 Bus System

S.No.	Case Name
1	Base Case
2	Outage in tie-line 1-2
3	Flow in tie-line 50-52 increased from 700 to 900 MW
4	Flow in tie-line 50-52 decreased from 700 to 455 MW
5	10% increase in inertia of machine at Bus 66
6	10% decrease in inertia of machine at Bus 66
7	All loads in the system increased to 103%
8	All loads in the system decreased to 90%

Figs. 5.1-5.2 show the reduction in the size of A_1 matrix obtained after the application of enhanced SMA for two of the operating points, where A_1 refers to the portion of A matrix composed of angles and speeds of machines. The black dots indicate the original eigenvalues of the system while red circles represent the eigenvalues of the reduced-order system obtained after performing 100 iterations of SMA (denoted by M_{100}). It can be seen that eigenvalues of both the reduced-order and full-order system overlap each other. This implies that the reduced-order system is able to mimic the dynamic responses of the full-order system. Similar results were obtained for the other six cases as well.

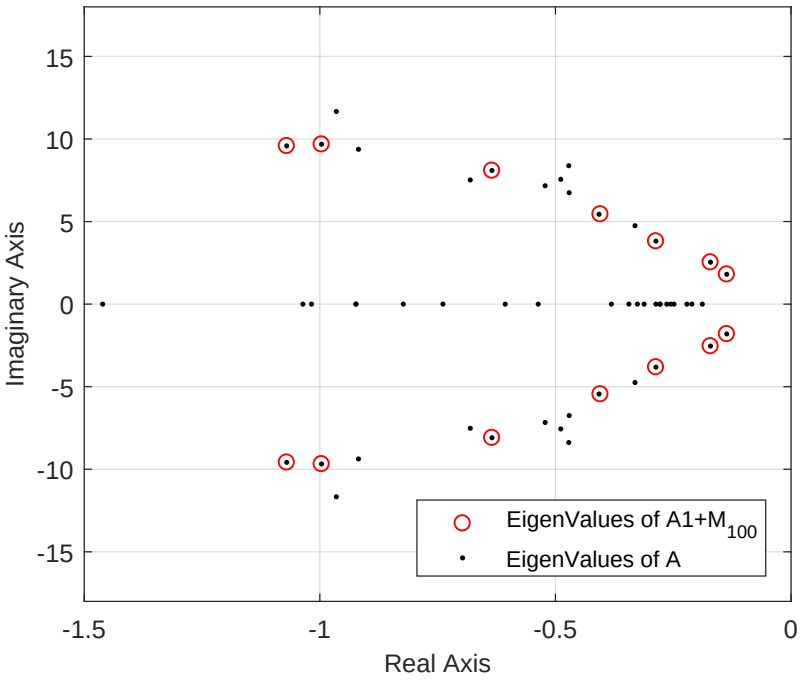

Figure 5.1. SMA Convergence for Case 1.

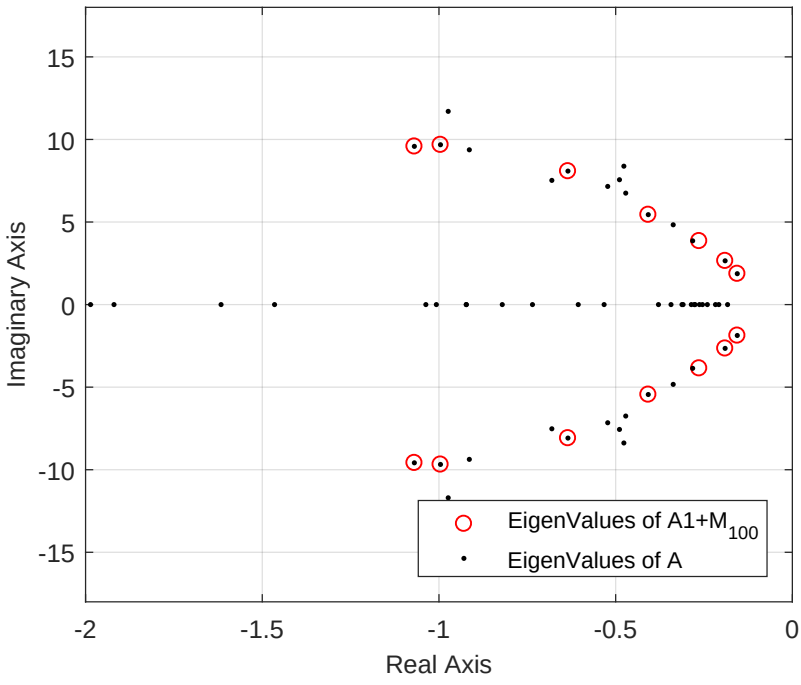

Figure 5.2. SMA Convergence for Case 2.

As detailed in Section 3.2.4.3, the criteria for reducing the number of critical generators (that form the control group) to a certain limit is decided by checking the convergence of SMA as well as the damping of closed-loop eigenvalues obtained after application of LQR control to the reduced system. Fig. 5.3 shows the SMA convergence for Case 1 on dropping the ninth machine from the system. It is evident that one of the eigenvalues of the reduced-order system does not match well with that of the original system. Similarly, the application of LQR control (refer Fig. 5.4), resulted in the movement of one of the eigenvalues to the right. Hence, only eight machines were selected to be dropped from control for the base case (namely, Case 1).

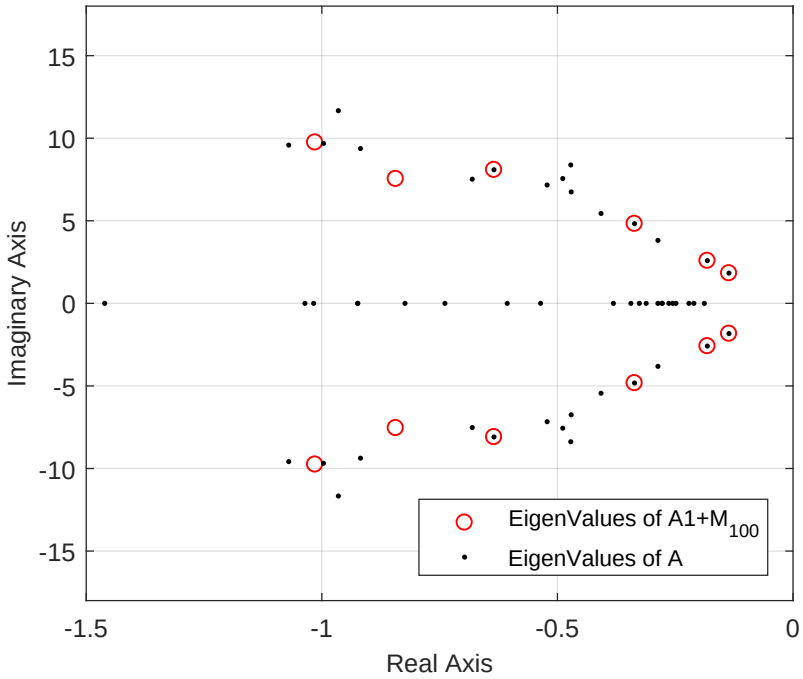

Figure 5.3. SMA on Dropping Ninth Machine for Case 1.

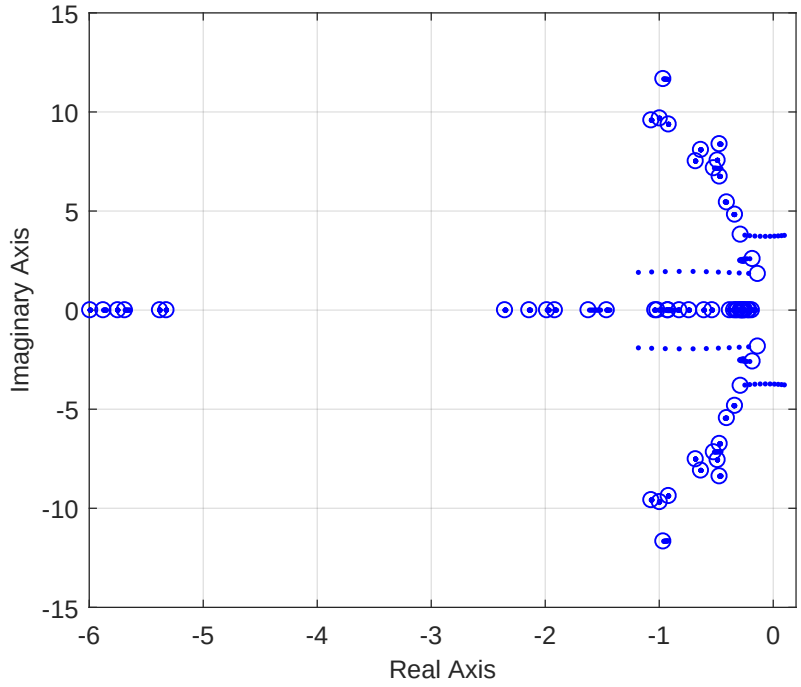

Figure 5.4. Control Check after Dropping Ninth Machine for Case 1.

Table 5.2 provides the sequence in which machines are to be dropped for the eight cases using the methodology described in Section 3.2.4.3. It can be realized that the number of machines which can be dropped varies with the test cases. Since implementing a single polytopic control with different system sizes is not possible, therefore the six (minimum) identical machines were chosen to be dropped from the control group, i.e. $(g_{53}, g_{55}, g_{57}, g_{62}, g_{63}, g_{64})$. This has resulted in the reduction of feedback signals by 37.5%.

In Fig. 5.5, LMI control is designed for a single polytopic system comprising of 18 retained states for each of the eight cases. The overall problem is described by a 29×31 LTI system which resulted in a 29×257 polytopic system and a 29×209 closed-loop system. All the critical eigenvalues shown in black oval, have attained ζ_{min} of more than 15%, whereas all the local modes have damping of at least 7.5%.

Table 5.2. Results for 16 Machine, 68 Bus System

Case No.	Sequence for dropping generators	Reduction in control signals(%)
1	63, 62, 60, 61, 64, 53, 55, 57	50
2	63, 62, 64, 53, 55, 57	37.5
3	63, 62, 60, 64, 53, 55, 57	43.75
4	63, 62, 64, 53, 60, 55, 57	43.75
5	63, 62, 64, 53, 55, 60, 57	43.75
6	63, 62, 64, 53, 55, 57	37.5
7	63, 62, 64, 53, 55, 57	37.5
8	63, 62, 64, 53, 55, 57	37.5

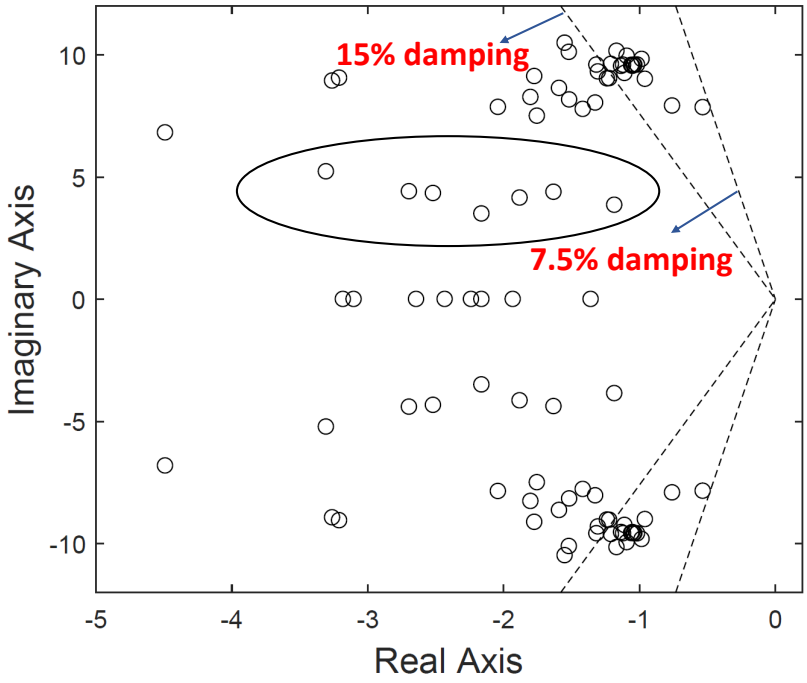

Figure 5.5. LMI Control for 16 Machine, 68 Bus System.

5.2 Large Test System-Reduced-order Model of WECC

A reduced-order model of WECC that includes the modeling features and complexities of the actual system is used to demonstrate the performance of the CWADC; i.e., the reduced model retains the overall inter-area modal properties of the full WECC system. The test system originally introduced in [133] is modified to include the HVDC lines and the

wind farms as shown in Fig. 5.6. This system has 33 synchronous generators, 140 buses, and 2 wind farms. All synchronous machines are represented in detail using a two-axis model, type ST1 excitation control, and general-purpose turbine-governor models. They also have conventional, local PSSs as shown in Fig. 3.2. All loads are assumed to be constant impedance loads. The two wind farms are installed at bus #9991 and bus #8881, and represented using Type 3 wind turbine generator (WTG) generator and electric control modules. The system has three HVDC lines, two of which represent the multi-terminal HVDC lines, namely, the Pacific DC Intertie (PDCI) between Celilo and Sylmar substations transmitting $2,300$ MW from the Northwest to the Southwest; the third one is the Intermountain Power Project (IPP), which transmits $1,750$ MW from the Mid-east to the Southwest. The PDCI and IPP were modeled as simple positive and negative loads in Section 5.1, but they are modeled using detailed dynamic models in this study. SDC is added to one of the two multi-terminal DC lines, whose model along with that of SVC are shown in Fig. 3.2. The rating of the SVC is 100 MVAr. Based on [40], the modulation limit for DC-SDC is set to ± 125 MW. All the existing generator excitation controls along with DC-SDC and SVC in the system are assumed to contain a communication module to accept the control signals from the CWADC.

5.2.1 Setting up of Polytopic Region

The proposed controller is intended to not only prevent negative interaction between different controllers, but also enhance the damping of multiple oscillation modes which may appear in a wide range of OCs.

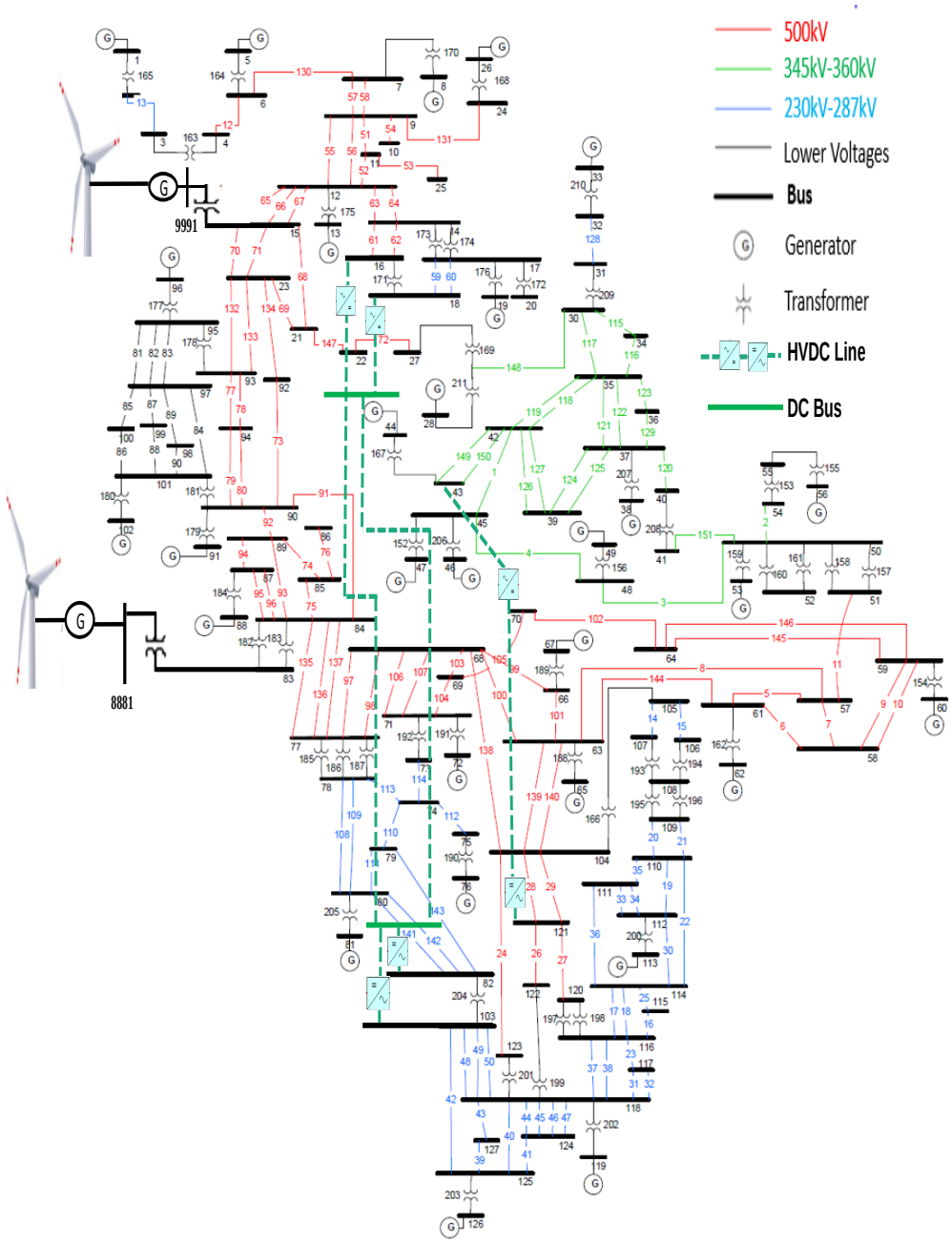

Figure 5.6. Reduced-Order WECC System.

In this study, seven different OCs with converged power flow solutions are analyzed, as shown in Table 5.3, and then combined in a single polytope. To represent these operating points, power generation and load demands are varied at the selected generator and load buses that resulted in a change in power transfers in and around WECC Intertie Path $\#26$. Vertex v_1 represents the base case. Vertices v_2 and v_3 are created by changing the generation of generator $\#91$ and load at bus $\#71$. Vertices v_4 and v_5 are created by changing the generation of a wind farm located at bus $\#8881$ and load at bus $\#69$. Generations of both generator $\#91$ as well as generator $\#8881$ are varied along with the load at bus $\#78$ to create v_6 and v_7. To ascertain if the designed polytopic controller can damp the unstable oscillation modes, one of the vertices chosen (namely, vertex v_6) has a converged power flow solution, but an unstable open-loop mode.

Table 5.3. Test Cases

Case/ Vertex No.	**Parameter Variation**	
	Generation	**Load**
v_1	No Change	No Change
v_2	#91-Increase by 250 MW	#71-Increase by 250 MW
v_3	#91-Decrease by 500 MW	#71-Decrease by 500 MW
v_4	#8881-Increase by 250 MW	#69-Increase by 250 MW
v_5	#8881-Decrease by 500 MW	#69-Decrease by 500 MW
v_6	#91-Increase by 200 MW #8881-Increase by 200 MW	#78-Increase by 400 MW
v_7	#91-Decrease by 250 MW #8881-Decrease by 250 MW	#78-Decrease by 500 MW

Small-signal stability analysis of the full system at each of the seven vertices reveals multiple electromechanical modes with low frequency and poor damping ratios ($< 5\%$). Table 5.4 shows three of the least damped modes obtained for the studied system. The system eigenvalues, damping ratios, and participation factors at each of these OCs are computed using the SSAT software [99].

The impact of negative interactions among different controls on the damping of one of the critical modes of the system is shown in Table 5.5. It can be seen that PSS and SVC which together add at least 3.46% damping to Mode1 for OCs v_2 and v_6, have interacted negatively with the DC-SDC, reducing its damping to less than 1%. In particular, Mode1

Table 5.4. Damping of Selected Modes

Case No.	Damping of critical modes(%)		
	Mode 1	Mode 2	Mode 3
v_1	3.71% @1.737Hz	4.03% @0.934Hz	4.14% @1.115Hz
v_2	0.27% @0.397Hz	2.46% @0.768Hz	2.98% @0.923Hz
v_3	3.74% @1.736Hz	4.05% @0.939Hz	4.07% @1.117Hz
v_4	3.63% @1.742Hz	4.04% @0.926Hz	4.08% @1.114Hz
v_5	3.71% @1.737Hz	4.03% @0.934Hz	4.14% @1.115Hz
v_6	−0.36% @0.399Hz	2.39% @0.766Hz	2.97% @1.743Hz
v_7	3.23% @0.430Hz	3.36% @0.787Hz	3.47% @1.206Hz

has become unstable for operating point v_6 due to the inter-controller interactions. On further investigation, it was found that the gain of the PSS for machine at bus #5, (which has maximum participation in Mode1) was high, and interacted unfavorably with the gain of DC-SDC, resulting in system instability. Therefore, this control interaction highlights the importance of coordinating multiple controllers present in the system.

Table 5.5. Controller Interactions

Case No.	Critical Mode from Table I	Damping(%)	
		PSS + SVC	DC-SDC + PSS + SVC
v_1	Mode1	3.51	3.71
v_2	Mode1	3.47	0.27
v_3	Mode1	3.54	3.74
v_4	Mode1	3.42	3.63
v_5	Mode1	3.51	3.71
v_6	Mode1	3.46	−0.36
v_7	Mode1	3.49	3.23

5.2.2 System Reduction

The size of the A and B matrices for each vertex is 656×656 and 656×35, respectively. Here, 35 refers to the total number of damping controllers present in the system, i.e. 33 PSSs installed at all synchronous machines, 1 DC-SDC, and 1 SVC. Each controller is tuned to provide minimum damping of 3% at the *base OC*. With the help of the enhanced SMA [128], the size of the A matrix for each of the seven operating points is reduced to 43×43 comprising of speeds of the 22 machines (including reference machine) and relative angles of 21 machines computed with respect to reference machine. The performance of the enhanced SMA used for obtaining the reduced A matrices for three operating points

is shown in Figs. 5.7-5.9. The black dots represent the eigenvalues of the complete system, while green circles represent the eigenvalues of the reduced-order system. The size of the B matrix is reduced to 43×24, where 24 signifies the number of controllers that are part of CWADC. The remaining 11 controllers belong to a set of non-critical generators as explained in Section 3.2.4, hence they remain completely decentralized. Each LMI problem with 43 retained states is described by a 66×92 LTI system. The collection of seven vertices results in a 69×652 polytopic system.

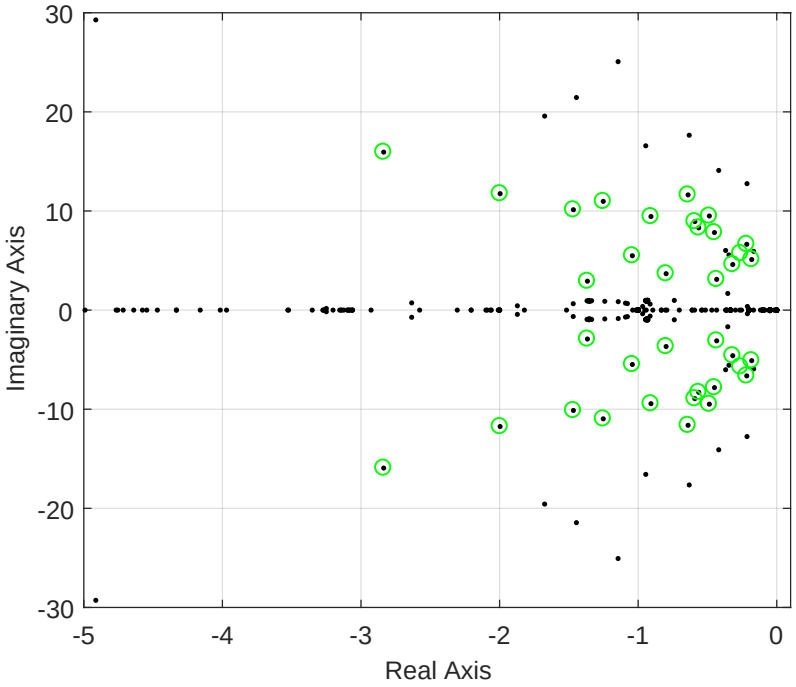

Figure 5.7. SMA Convergence for Case 1.

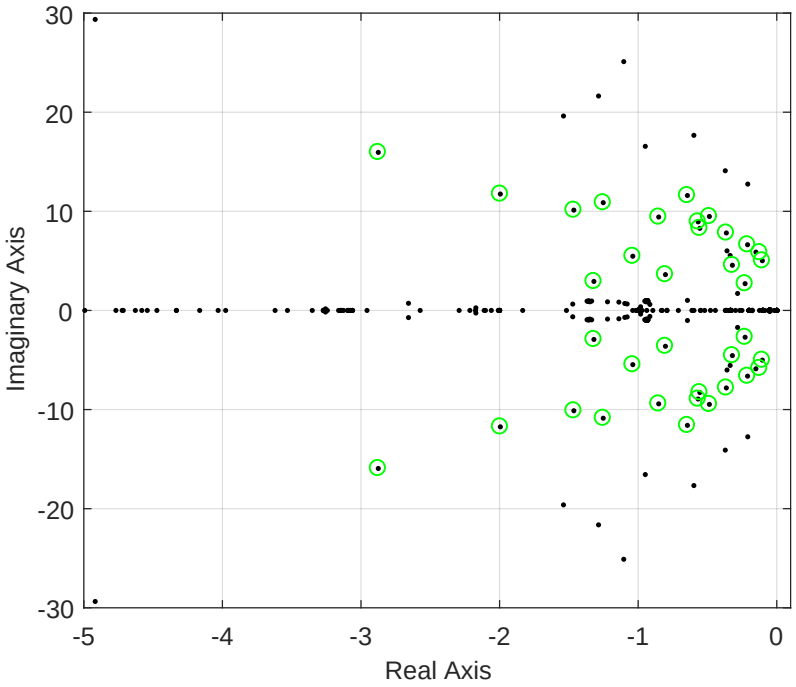

Figure 5.8. SMA Convergence for Case 2.

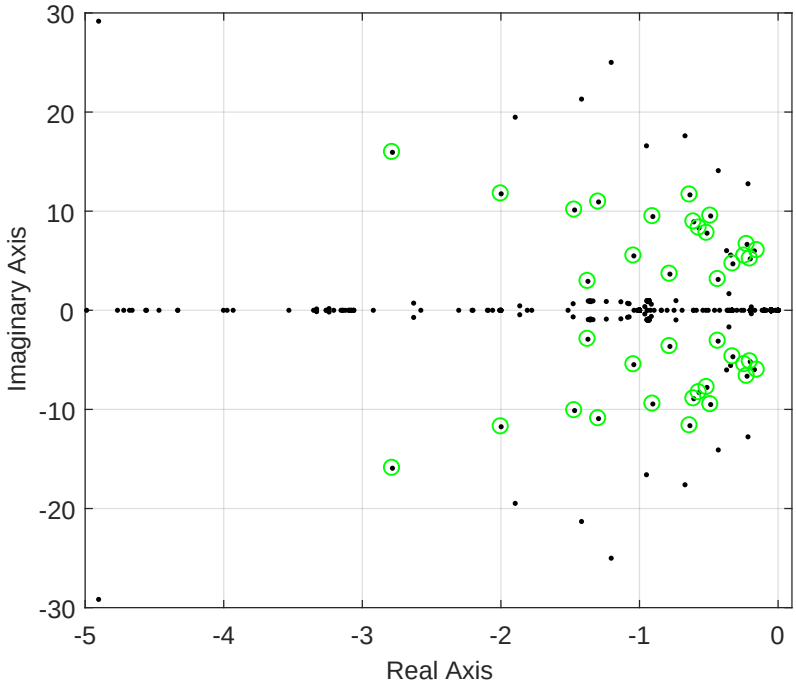

Figure 5.9. SMA Convergence for Case 3.

5.2.3 Synthesis of CWADC

To develop the LMI-based mixed H_2/H_∞ control for the polytopic system with regional pole-placement constraints, the $msfsyn$ function available in the LMI control toolbox of MATLAB was used to design the CWADC. The size of the designed controller gain matrix, $K_{LMI-poly}$ is 24×43, hence CWADC is a 43-input, 24-output system. To guarantee the system's performance over a wide range of operating points, the proposed CWADC gathers system-wide measurements from geographically diverse locations in WECC. This ensures the availability of more system dynamic information contained in remote stabilizing signals. It is to be noted that though CWADC controller is designed for the reduced-order system, its performance is tested using the full-order system. The goal of the CWADC is to provide a minimum damping, ζ_{min} of 5%.

5.3 Application of the Proposed Control Scheme

5.3.1 Performance of the Controller

Table 5.6 shows the application of the LMI control design for the polytopic system. All the critical eigenvalues including those listed in Table 5.4, have attained minimum damping of 5.5%. It was also observed that at higher load conditions, with the CWADC included, the power flow in and around the WECC Intertie Path $\#26$ could be increased. Specifically, the system's stability limit was extended from $3,398$ MW to $3,598$ MW (increase in load at bus $\#71$ from 300 MW to 500 MW), $1,830$ MW to $1,930$ MW (increase in load at bus $\#69$ from 600 MW to 700 MW), and $1,616$ MW to $1,741$ MW (increase in load at bus $\#78$ from 550 MW to 675 MW), respectively, for vertices v_2, v_4, and v_6 specified in Table 5.3. It should be noted that for vertices v_2 and v_4, the limit is imposed by the power flow not converging beyond these values.

Table 5.6. Results of Modal Analysis

Case No.	Damping of critical modes(%)		
	Mode 1	**Mode 2**	**Mode 3**
v_1	9.19% @1.71Hz	5.51% @0.964Hz	5.70% @1.14Hz
v_2	12.54% @0.427Hz	12.5% @0.75Hz	5.68% @0.96Hz
v_3	10.6% @1.71Hz	5.59% @0.96Hz	5.91% @1.15Hz
v_4	13.69% @1.71Hz	5.64% @0.95Hz	6.04% @1.15Hz
v_5	9.23% @1.71Hz	5.57% @0.96Hz	5.96% @1.15Hz
v_6	12.85% @0.41Hz	9.54% @0.76Hz	9.38% @1.707Hz
v_7	11.92% @0.45Hz	11.51% @0.76Hz	9.48% @1.24Hz

To evaluate the performance of the CWADC, results obtained using modal analysis are validated by conducting time-domain simulations in PSLF software. A small disturbance is created by selecting two of the operating points that lie within the designed polytope, i.e. simultaneous increase of load at bus #78 by 200 MW and a decrease in load at bus #71 by 400 MW at time, $t = 0.5sec$. The active power responses of three different generators having higher participation in the inter-area modes (shown in Table 5.4), i.e. generators at buses #5, #76, and #91 are shown in Figs. 5.10-5.12, with CWADC and without CWADC, i.e. in the presence of local controllers (PSSs, DC-SDC, and SVC) only. It can be seen that CWADC designed using a reduced-order system is able to improve the damping of the inter-area modes present in the full-order system. Figs. 5.13-5.14 show the control efforts extended by the DC-SDC and SVC, both of which are within their predefined limits.

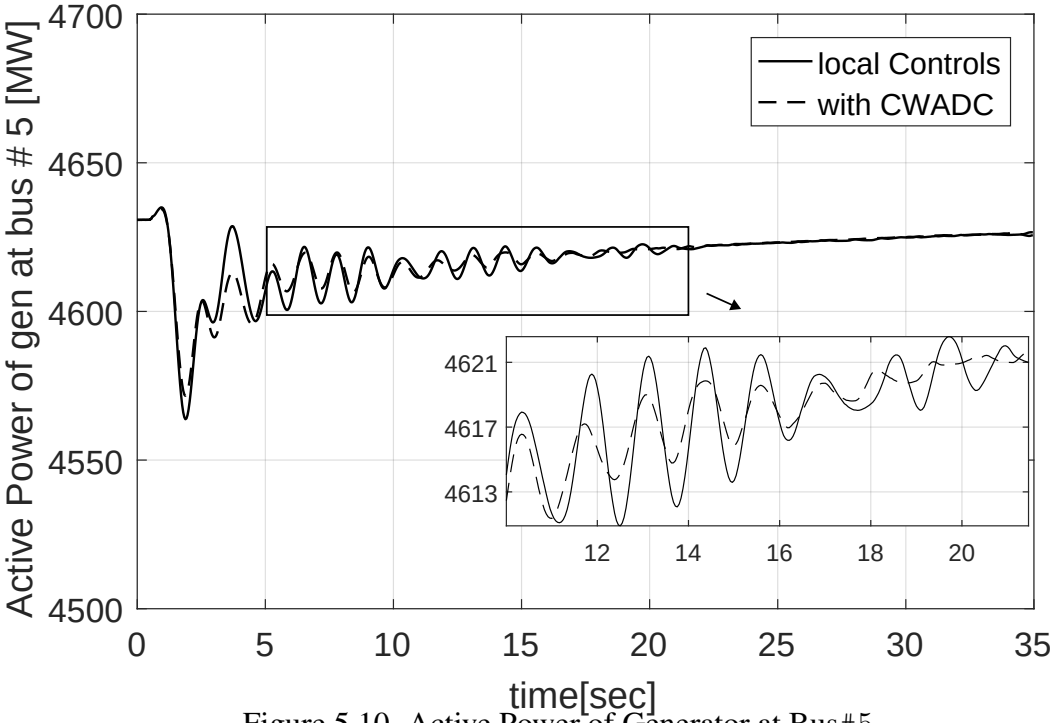

Figure 5.10. Active Power of Generator at Bus#5.

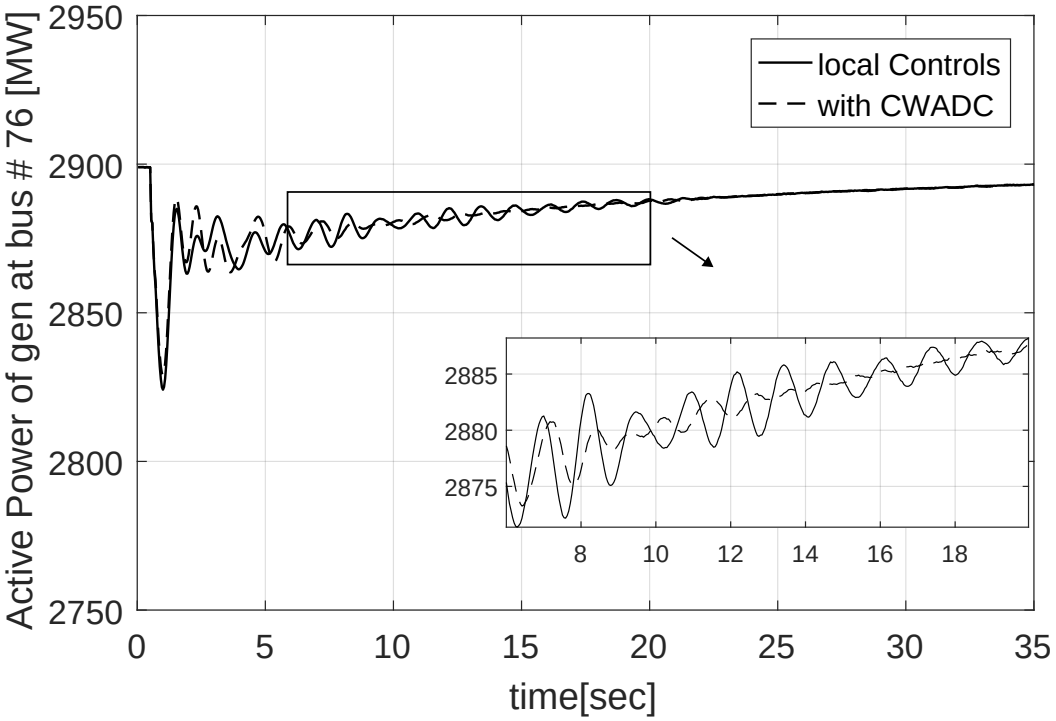

Figure 5.11. Active Power of Generator at Bus#76.

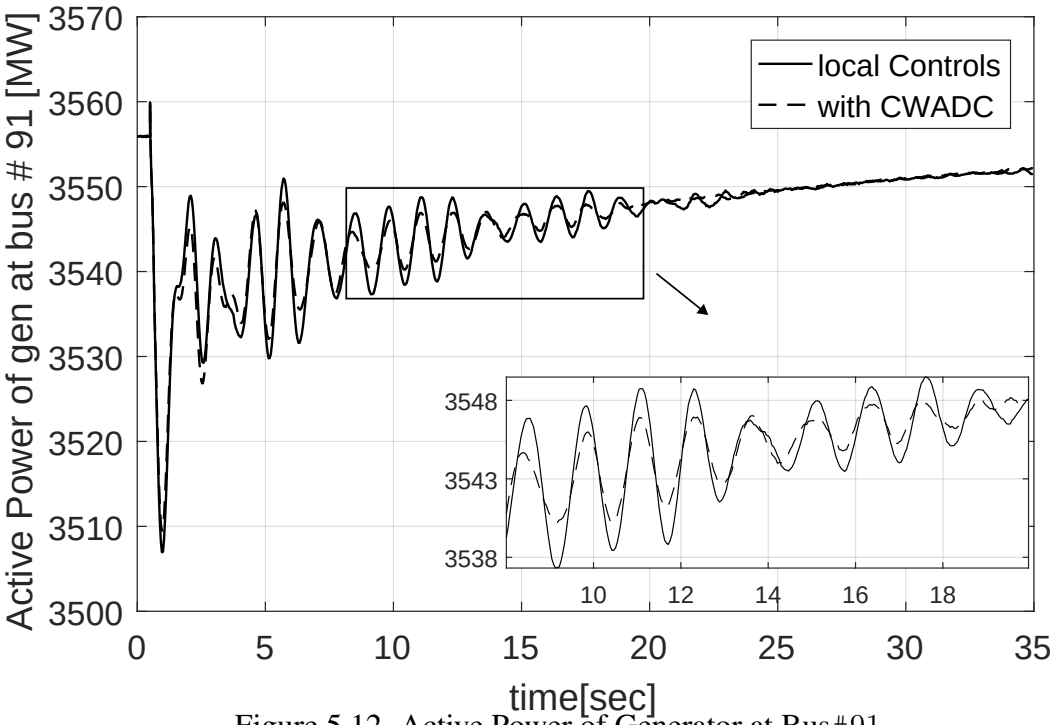

Figure 5.12. Active Power of Generator at Bus#91.

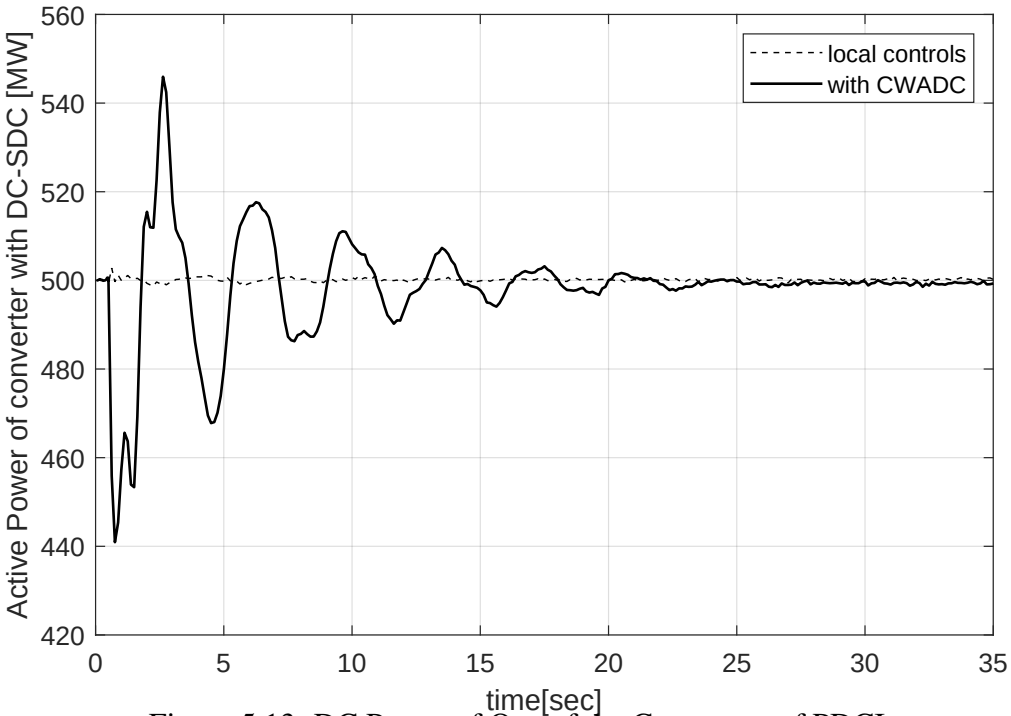

Figure 5.13. DC Power of One of the Converters of PDCI.

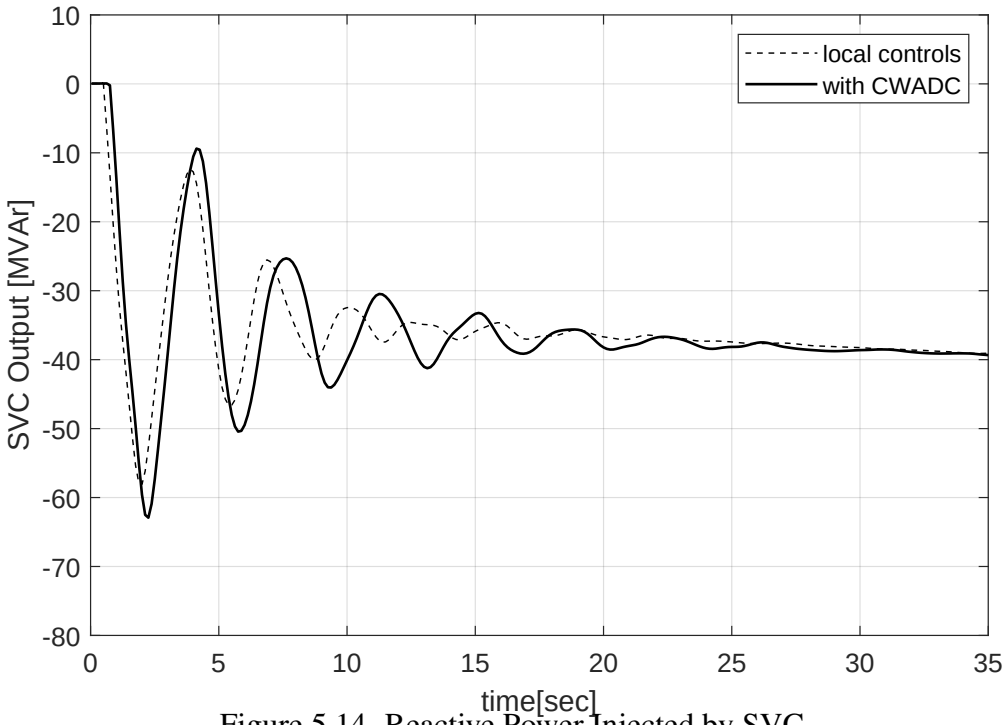

Figure 5.14. Reactive Power Injected by SVC.

5.3.2 Flexibility of Selecting Control Signals

By assessing the performance of the CWADC, it can be realized that the use of wide-area signals has a greater impact on improving the system's stability. However, the proposed controller has a known weakness: its functioning may deteriorate when the quality of PMU data is bad (for instance, due to PMU failure). To ensure reliable access to PMU signals, an alternate feedback selection scheme is integrated into the proposed control design.

The selection of alternate feedback control signals is performed based on **Algorithm 3** of Section 3.2.5. On applying this algorithm to the system under investigation, it is found that the generator present at bus #113 is a reliable alternative to the generator at bus #67. It is to be noted that the generator present at bus #113 is based in Southern California while the generator at bus #67 is located in Nevada (refer Fig. 5.6).

Using this alternate set of feedback signals, the polytopic-CWADC was designed once again for the full range of defined OCs. The size of the controller gain matrix, $K_{LMI-poly}$, is 23×41, which is reduced in comparison to that obtained using the primary set of feedback signals. This is because there are two generator units installed at bus $\#67$, both of which are now dropped from the control set.

Fig. 5.15 provides the result for the LMI control obtained using the alternate feedback signals. It can be realized that all the critical eigenvalues shown in black oval including the ones that are given in Table 5.4, have attained ζ_{min} of 5.5%.

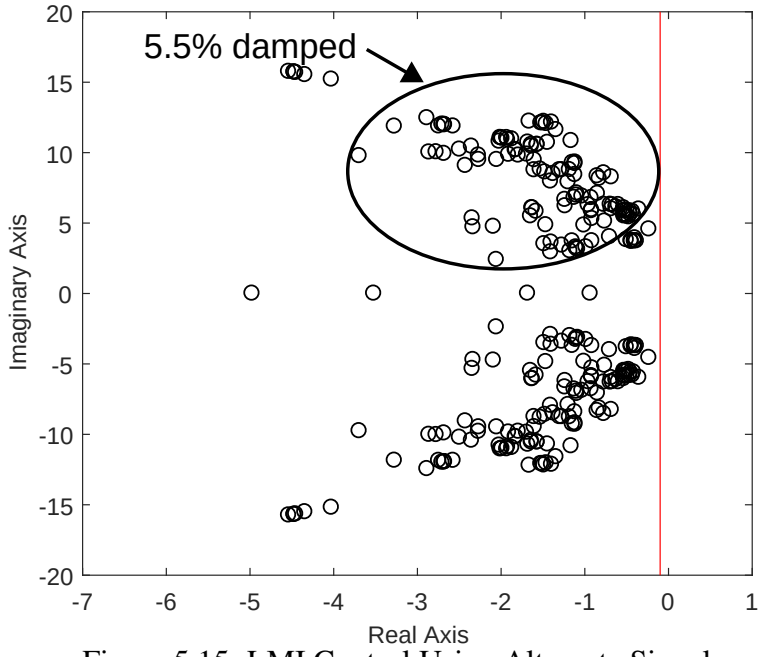

Figure 5.15. LMI Control Using Alternate Signals.

The active power variations for the three generators shown in Section 5.3.1 are shown in Figs. 5.16-5.18. From the plots, it is confirmed that the use of the alternate feedback signals has the same impact on the system's damping as the signals from the primary set. The outputs of the DC line and the SVC, modulated as a result of the supplementary stabilizing signals from the CWADC are shown in Figs. 5.19-5.20.

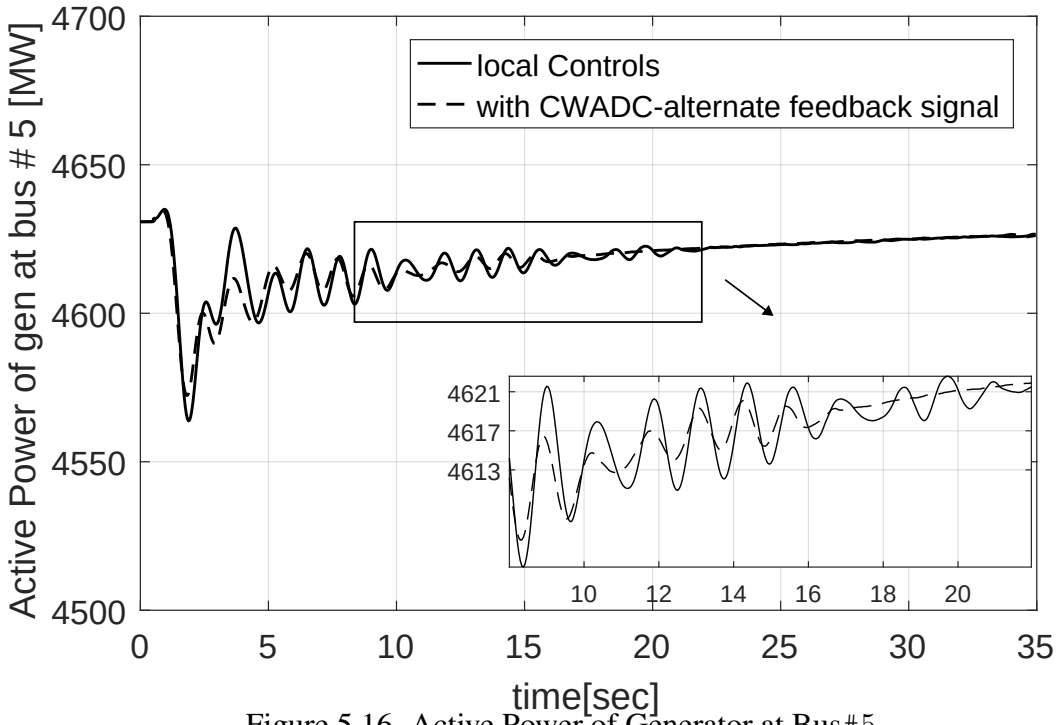

Figure 5.16. Active Power of Generator at Bus#5.

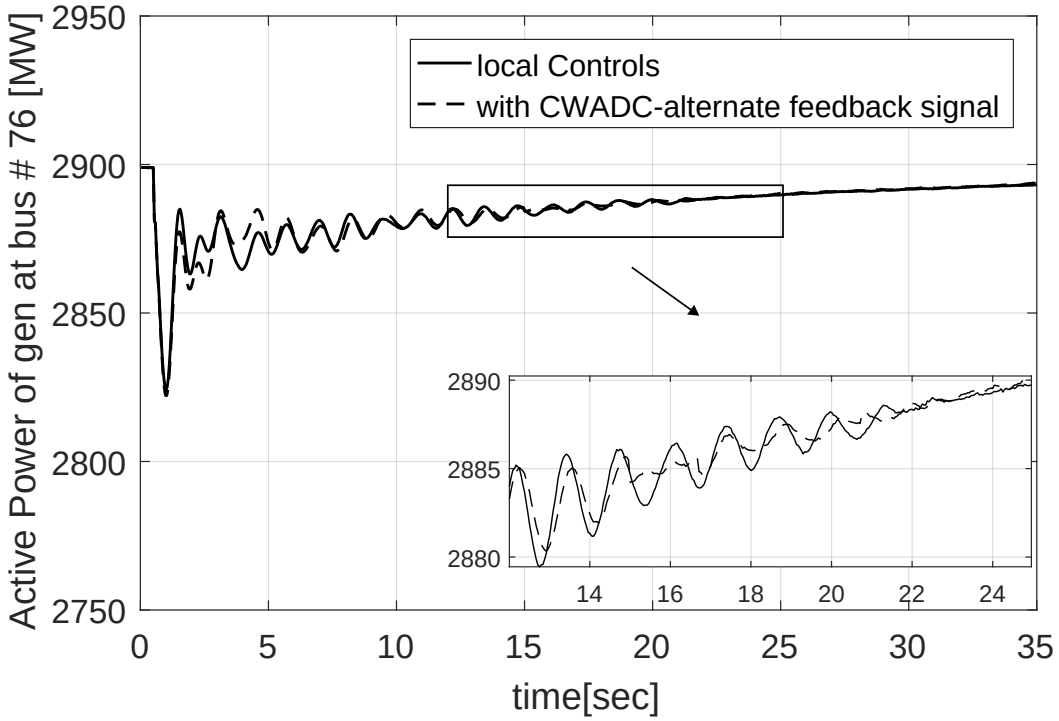

Figure 5.17. Active Power of Generator at Bus#76.

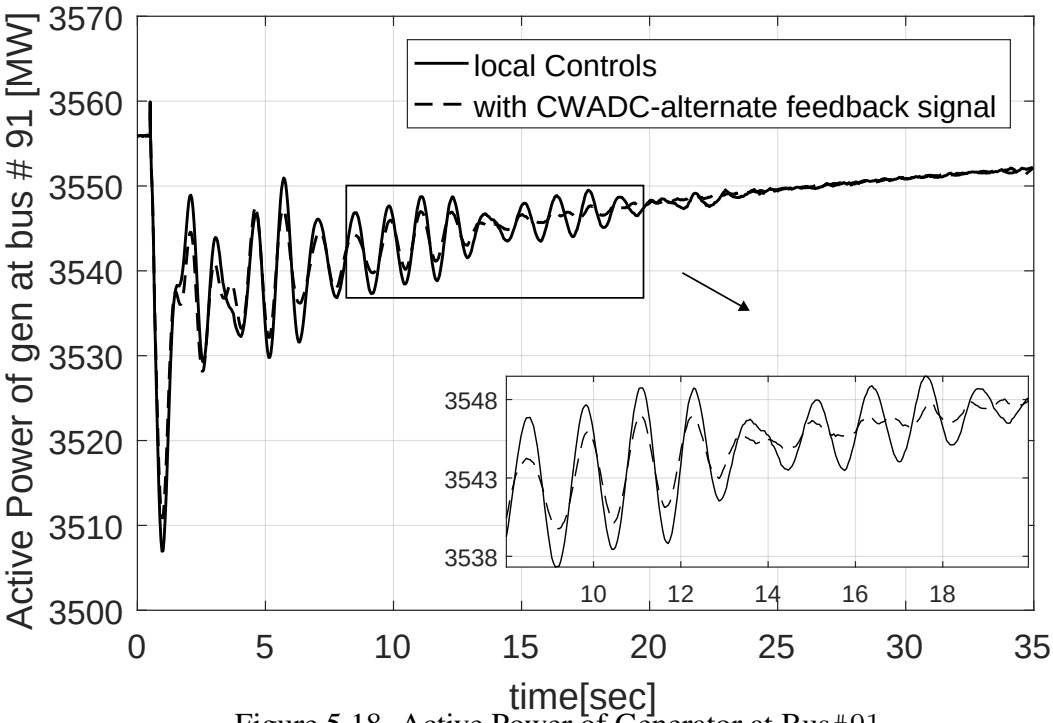

Figure 5.18. Active Power of Generator at Bus#91.

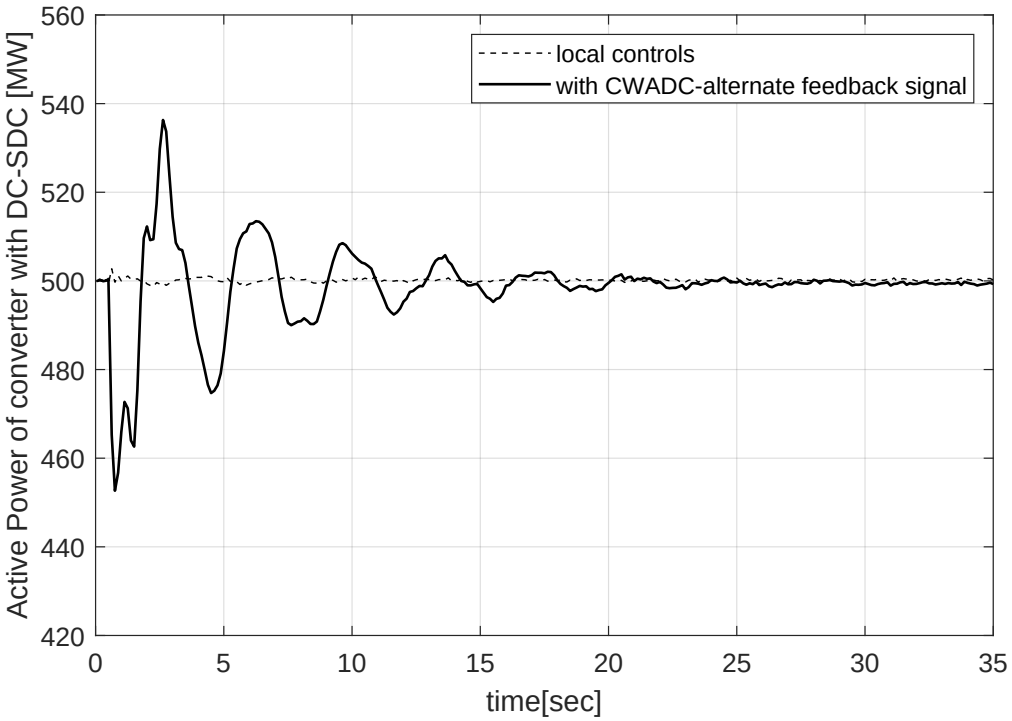

Figure 5.19. DC Output with Alternate Signals.

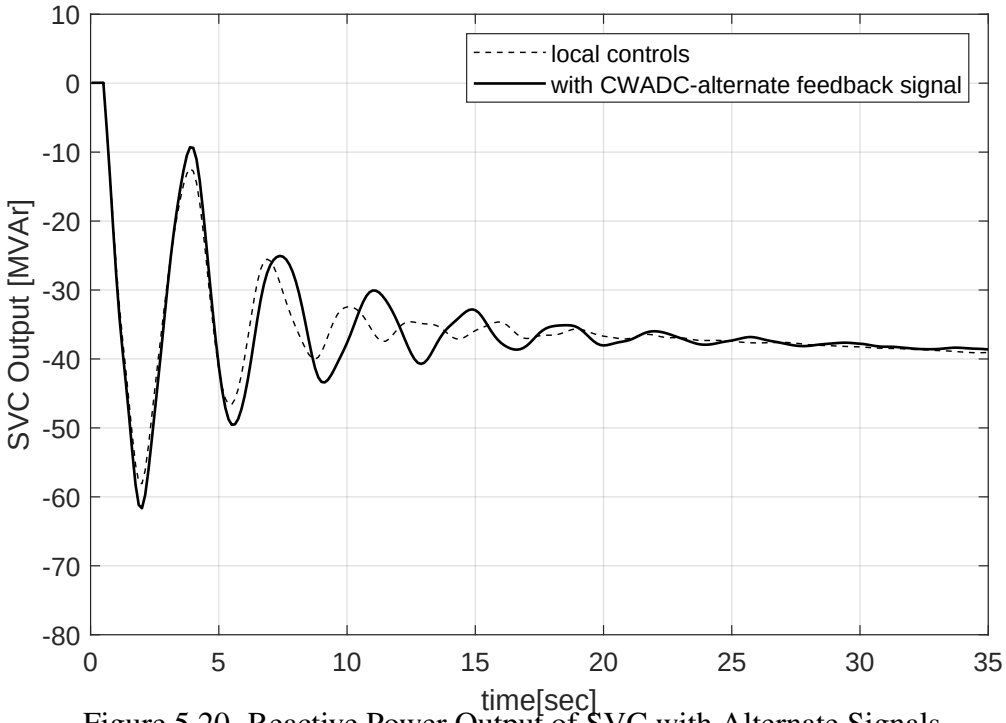

Figure 5.20. Reactive Power Output of SVC with Alternate Signals.

This simulation shows that if the PMU at bus #67 is out of service, the proposed controller can then obtain a feedback signal from the PMU at bus #113 to generate an equivalent feedback control. That is, backup observability of generator at bus #67 (a generating unit in Nevada) is ensured by the PMU at bus #113 (generating station in Southern California), and vice-versa. However, it must again be pointed out that it may not be possible to find a suitable replacement for every critical generator in the primary set. Therefore, if PMUs at multiple locations fail simultaneously, the designed controller may not be able to provide the requisite damping for the complete range of OCs.

5.3.3 Incorporation of Delays

The implementation of the CWADC requires data to be transmitted from the selected PMU locations to the coordinated controller. As detailed in [40], [23], it is possible that the data traveling through different communication channels reach the same destination

at different times. This is because every channel may have a different latency associated with it due to varying network configurations and congestion. Test results obtained in [40] for a real-time damping control designed for the WECC indicate that the total delay, that includes the switching delay of about 11ms, is on average 82 ms with a maximum value recorded at 113 ms. Hence, in this research, non-linear simulations are performed by adding an implicit random delay, that lies between 25-100ms, to the wide-area signals. The gains obtained using the primary set of feedback signals are used for the simulations. It can be observed from Figs. 5.21-5.22, that when the CWADC suffers a communication delay, the performance of the controller degrades slightly, but it still performs better than the local controllers. However, if the delays are too large, it is acceptable to revert to the local controllers. This would remain the case until the latency in the wide-area signals is small enough to ensure a positive effect on the oscillation damping.

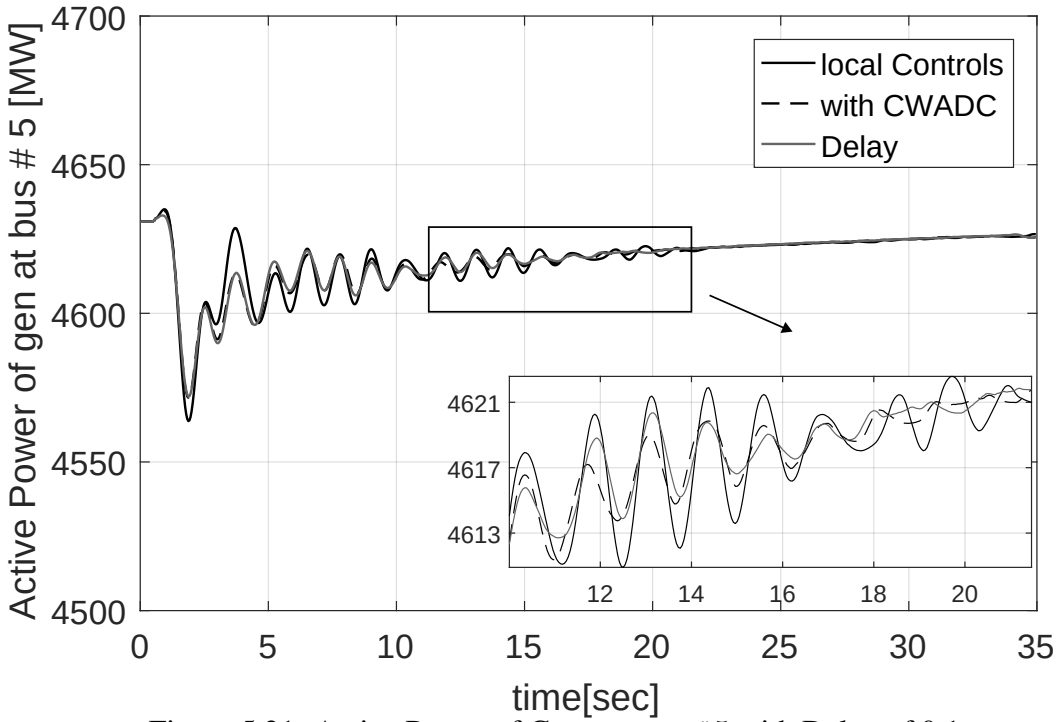

Figure 5.21. Active Power of Generator at #5 with Delay of 0.1s.

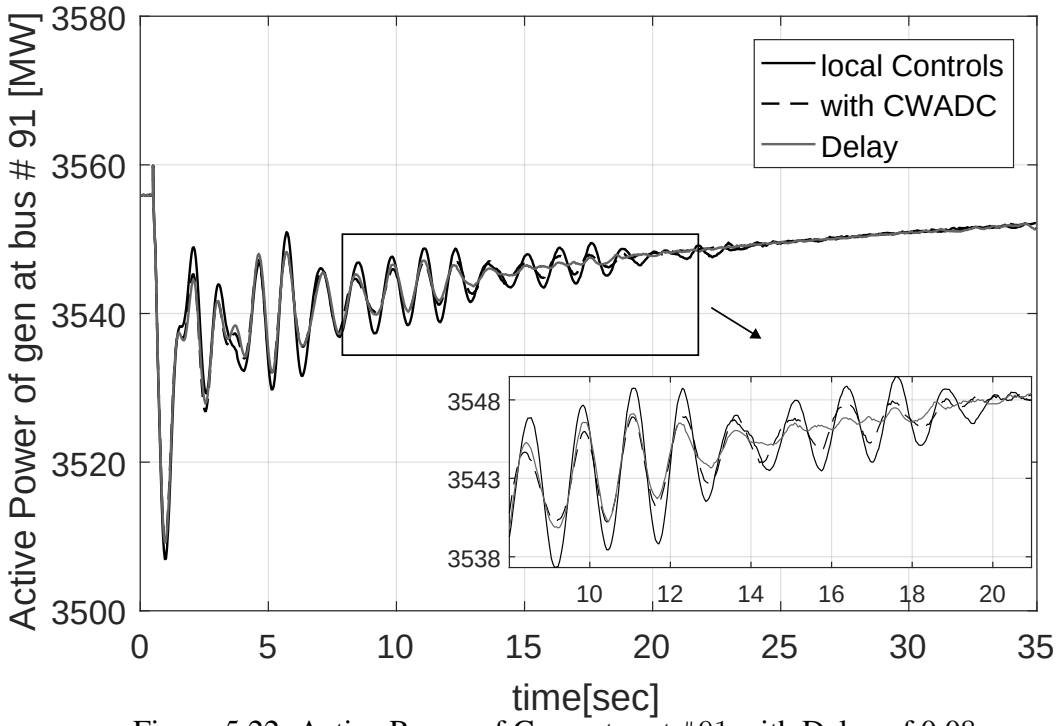

Figure 5.22. Active Power of Generator at $\#91$ with Delay of 0.08s.

5.3.4 Large Disturbance Analysis

By subjecting the test system to large disturbances, the actual non-linear behavior of the system can be observed, and the performance of the CWADC assessed accordingly. For the large disturbance simulated here, the network is subjected to a 100ms three-phase fault at bus $\#105$ at $t = 0.5sec.$ The improved performance realized by employing CWADC is evident in the results of the transient simulation shown in Figs. 5.23-5.25, which depict the active powers of four generators present at bus $\#5$ (a generating unit in Canada), bus $\#26$ (a generating station in Montana), bus $\#76$ (a generating station in Southern California), and bus $\#91$ (a generating unit in Northern California); see Fig. 5.6 for the bus locations. It is clear from Figs. 5.23-5.25 that in the presence of CWADC, the system experiences significantly smaller swings.

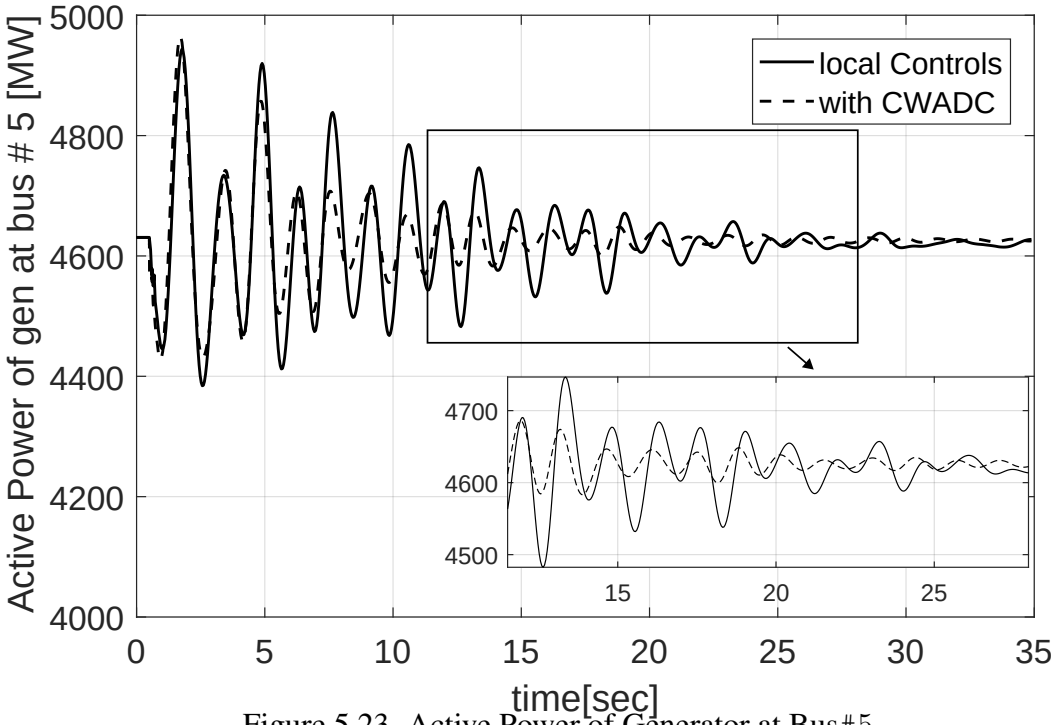

Figure 5.23. Active Power of Generator at Bus#5.

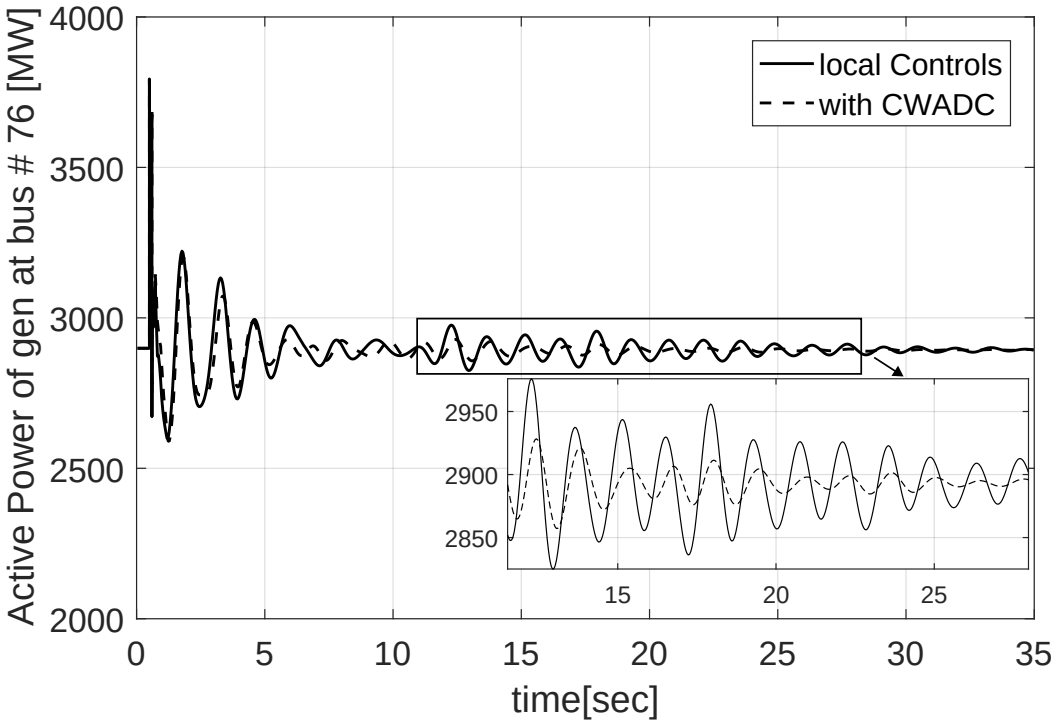

Figure 5.24. Active Power of Generator at Bus#76.

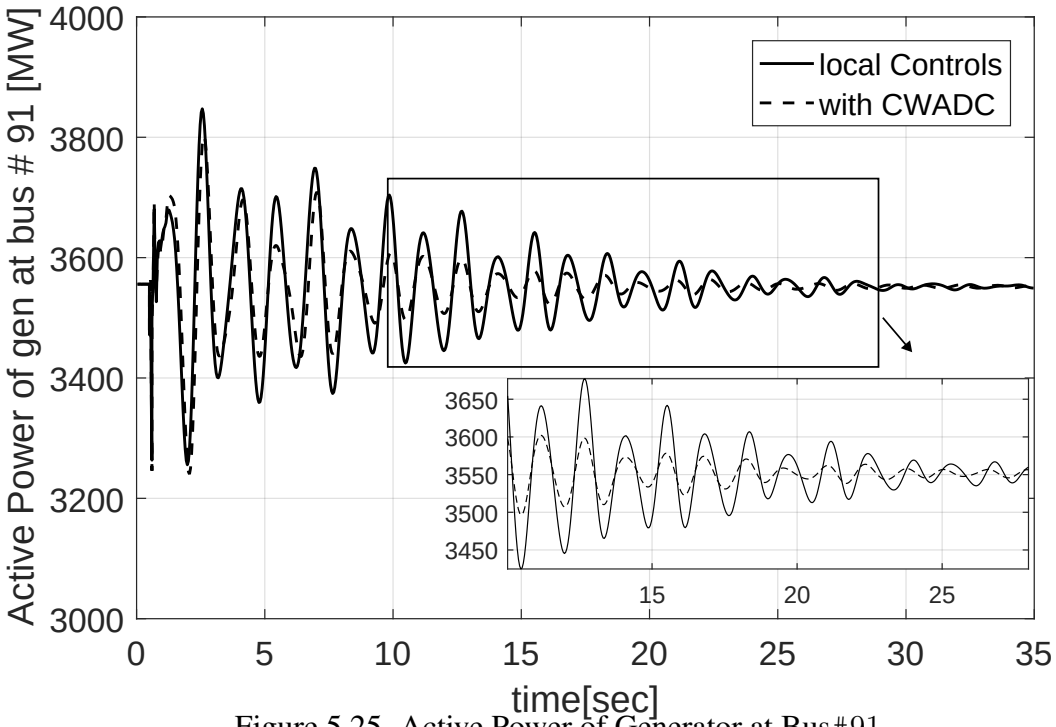

Figure 5.25. Active Power of Generator at Bus#91.

5.3.5 Robustness Comparison

To demonstrate the benefits associated with the proposed method, a comparison is made with another control scheme for a specific operating point that lies inside the designed polytope but is not one of its vertices. The load at bus #69 is increased by 100 MW at $t = 0.5sec$. The analysis is performed by designing two additional non-polytopic controls using LQR: a) the first non-polytopic control corresponds to a controller that is designed for vertex v_1, i.e. base OC, b) the second non-polytopic control corresponds to a controller that is designed for the simulated disturbance, i.e., increase in load at #69 by 100 MW, by treating it as a *known* operating point.

The active powers of two generators, i.e., gen at bus #5 and bus #91 which have maximum participation in the least damped modes for this operating point are shown in Figs. 5.26-5.27. From the two plots, it can be seen that the first non-polytopic controller performs

poorly as it is designed for the base OC. The performance of the second non-polytopic controller is comparable to that of the CWADC; however, designing that controller requires prior knowledge of the OC, which is not likely to happen in reality. Thus, this comparison illustrates the benefit of using a polytopic control, as a single controller can guarantee the desired damping effect for all the OCs lying inside the polytope.

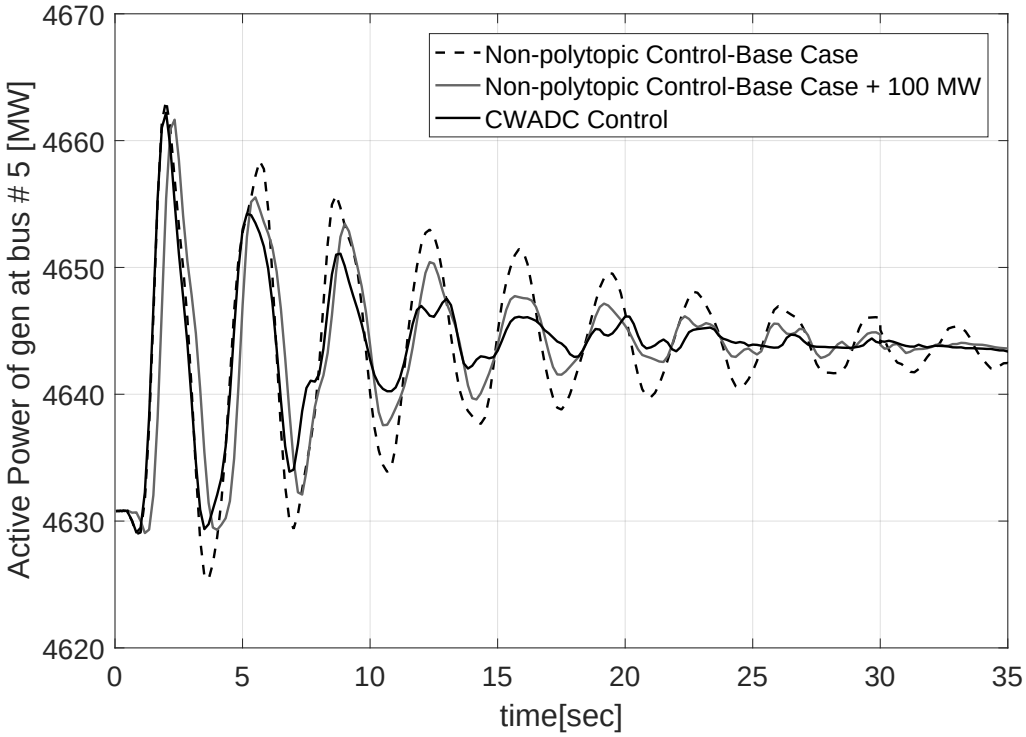

Figure 5.26. Comparative Analysis-Active Power of Generator #5.

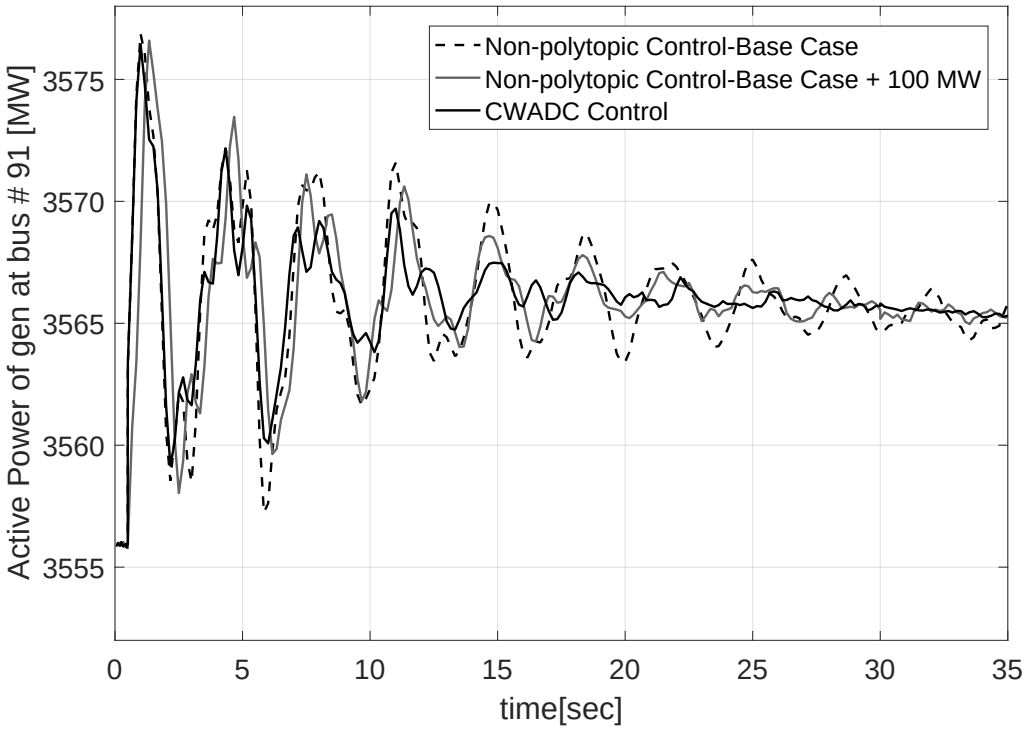

Figure 5.27. Comparative Analysis-Active Power of Generator #91.

5.4 Discussions

The success of the single polytopic controller design for a wide range of OCs depends on two factors: a) diversity of the OCs, and b) number of OCs considered. If the OCs are very diverse, or the number of OCs that are intended to be combined into a single polytope are too many, the controller gain matrix generated after solving the LMI-based control problem can have potentially large magnitudes of gains. Similarly, trying to move the closed-loop poles to the left by a significant margin or an incongruous selection of the weights for H_2 and H_∞ controls during the creation of the polytope can also lead to excessively large controller gains. These scenarios should be avoided as they can lead to controller output saturation.

The proposed approach offers an advantage over adaptive control design schemes as no processing time is required for control selection and actuation depending on the system's

OC. Since all the controller parameters are obtained using off-line case studies and can be stored for online operations, there will be no delay in accessing the controller data. Furthermore, the synthesized control is composed of gains only, as compared to other higher-order controllers that have large numbers of control parameters (such as [22] and [39]). Hence, the designed control is more appropriate for real-time applications, as it is computationally tractable and does not require knowledge of various control parameters.

CHAPTER 6

RESULTS FOR AI-BASED CONTROLS

The performances of the DNN-CWADC and DRL-CWADC are demonstrated using the reduced-order model of the WECC, as shown in Fig. 5.6 of Chapter 5. For this study. ζ_{min} is selected as 5%.

6.1 Results for DNN-CWADC

6.1.1 Training Data Generation

For this study, a diverse set of OCs was generated by creating topological changes (both $N-1$ and $N-2$ contingencies), and changes in the generation and loads of certain generators and load buses, resulting in variations in the power transfers in and around the WECC Intertie Path $\#26$ [135]. A total of 45 different OCs with converged power flow solutions are combined to create 230 polytopes using the LMI-based H_2/H_∞ optimization and pole-placement constraints. Before the synthesis of a polytope, the size of the state matrix and the input matrix corresponding to each OC was reduced to 43×43 (22 speeds of critical machines and 21 relative angles of critical machines calculated with respect to the reference machine) and 43×24 using enhanced SMA [128], respectively, where 24 signifies the total number of local controls that are part of the polytopic control. The size

of every designed controller gain matrix, $K_{LMI-poly}$, is 24×43. The polytopic data, when combined with the heavy transfer flows in the tie-lines connecting the WECC Intertie Path $\#26$ to its neighboring regions, and the three DC lines result in a total of $5,619$ input features and $1,032$ outputs used for training the DNN. A total of $2,600$ simulated samples were created, in which 1% Gaussian noise was added to the input measurements to make them resemble actual data obtained from the field.

The architecture of the DNN used for training is detailed in Table 6.1, which also shows its training progress evaluated using the metric, mean absolute error (MAE). The validation MAE indicates how well the learned model generalizes, while the testing MAE specifies how well the trained model performs for unseen data.

Table 6.1. DNN Architecture and Its Performance

DNN Architecture	
# of neurons in Input Layer	$5,619$
# of neurons in first Hidden Layer	$2,500$
# of neurons in second Hidden Layer	$1,861$
# of neurons in output Layer	$1,032$
Loss Function	Mean Absolute Error (MAE)
Learning rate for ADAM optimizer	10^{-3}
Batch Size	32
L2 Regularization	On (χ =10^{-5})
Errors	
Validation MAE	0.012
Testing MAE	0.017

To illustrate the requirement of the "deep" neural networks, i.e., at least two hidden layers [136], for the design of CWADC, an additional model with a single hidden layer with $2,500$ neurons (while keeping everything else the same) was trained using the same training data. The validation and testing MAE obtained for this model, referred to as NN-CWADC, are 0.215 and 0.14, respectively, which are much higher than those reported in Table 6.1. This clearly demonstrates that a single hidden layer is not sufficient for the neural network to learn the proper mapping between inputs and outputs. The validation of the performance of this model is performed in the next Section.

6.1.2 Performance Evaluation

The impact of the DNN-CWADC is assessed through eigenvalue analysis and non-linear time-domain simulations. The performance is evaluated in three different settings:

a) Scenario 1a-Validation of DNN-CWADC using modal analysis: To demonstrate the benefits of using a polytope-based DNN wide-area control, an OC that lies inside one of the polytopes forming the training dataset, but is not one of the vertices, is selected. This OC corresponds to an increase in the generation of the wind farm connected to bus $\#8881$ and load at bus $\#78$ by 690 MW. For comparison, two additional control schemes are also designed: 1) a polytopic control designed using the classical LMI approach [42] with the OC being one of its vertices (Dedicated H_2/H_∞ polytopic CWADC), and 2) the single polytopic control designed in Chapter 5 (Single Polytopic Controller). From Fig. 6.1, it is realized that all the LFOs (marked by 'o') attained ζ_{min} of 5% using DNN-CWADC. The proposed control also enlarged the stability domain of the system-under-study, since it was stated in Section 5.3.1 that the maximum load that can be increased at bus $\#78$ without causing instability was 675 MW (evident from Fig. 6.1 as the closed-loop poles denoted by '$*$' moved to the RHS of s-plane), while the proposed DNN-CWADC was able to provide requisite damping even when the loading increased to 690 MW.

b) Scenario 1b-Validation of DNN-CWADC using time-domain simulations: The results obtained using modal analysis are validated by conducting non-linear time-domain simulations in PSLF [98] for an increase in load at bus $\#78$ by 200 MW with and without the proposed DNN-CWADC (in the case without DNN-CWADC, the control is provided by local controllers which are *uncoordinated*). The active power responses and rotor angle deviations of two generators (generators at bus $\#5$ and bus $\#26$) having higher participation in the critical LFO modes is shown in Figs. 6.2-6.5. It can be observed that the designed control effectively damps out the oscillations. Additionally, the inclusion of 1% noise in

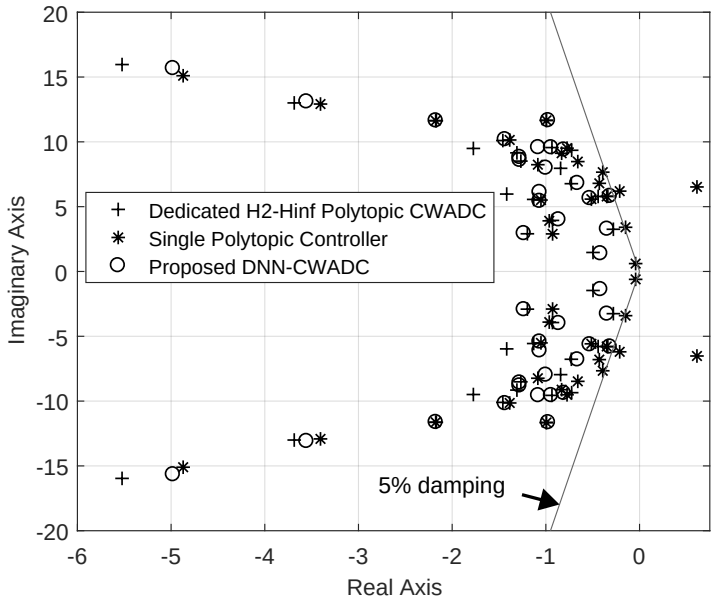

Figure 6.1. Validation of DNN-CWADC: Modal Analysis.

the input measurements does not affect the performance of the learned control.

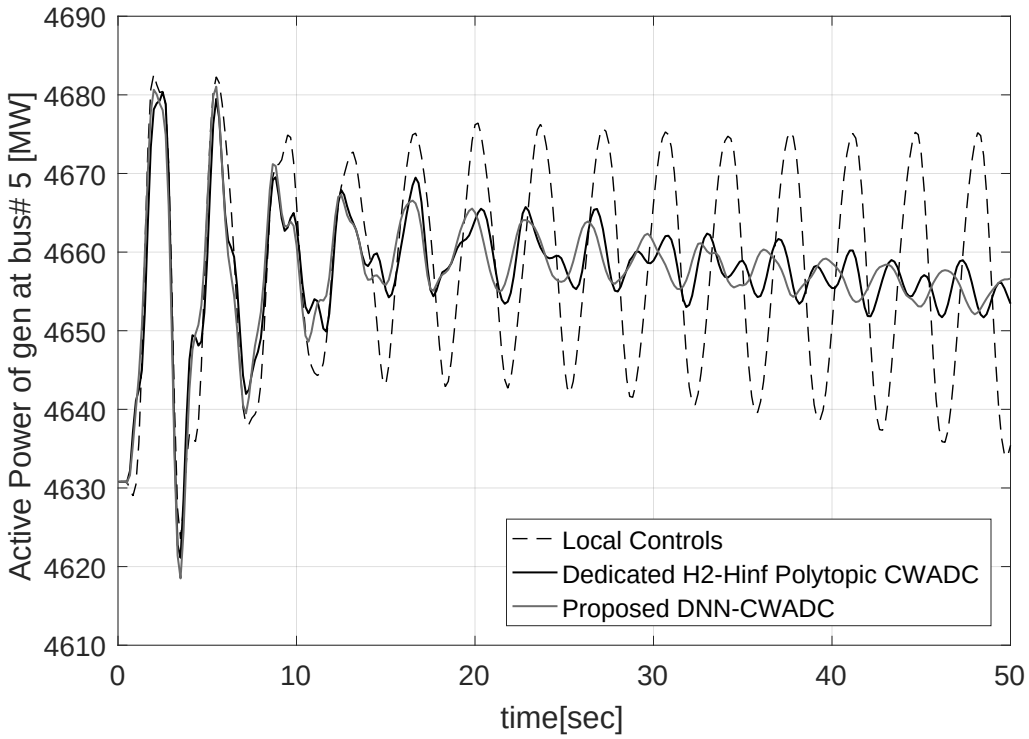

Figure 6.2. Validation of DNN-CWADC: Active Power of Generator #5.

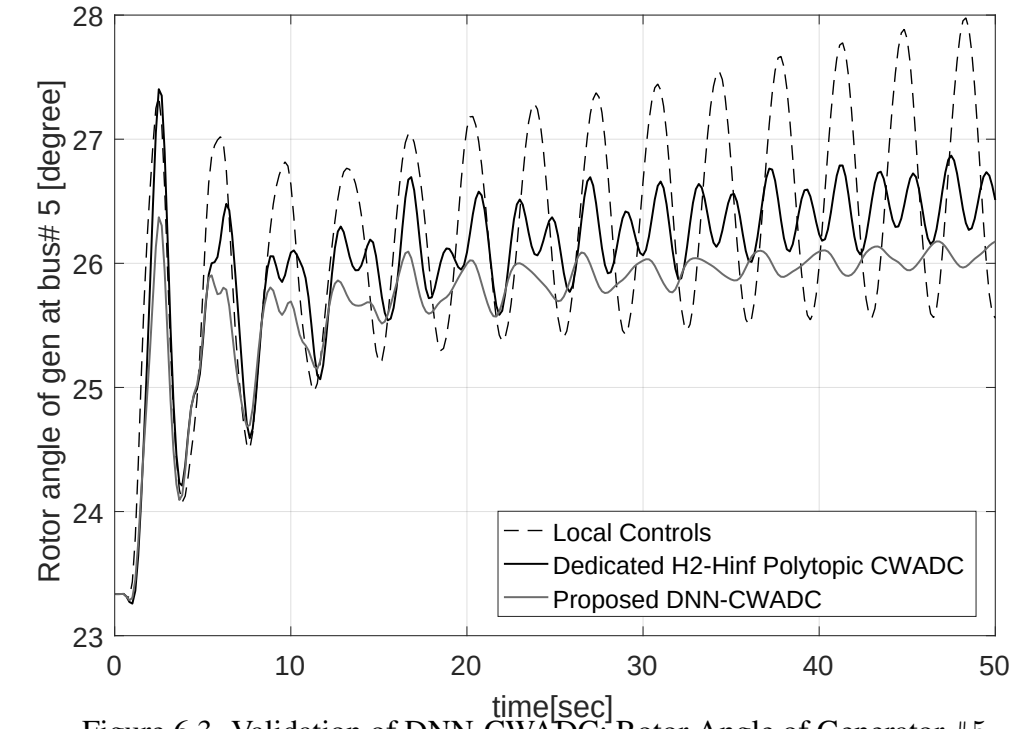

Figure 6.3. Validation of DNN-CWADC: Rotor Angle of Generator #5.

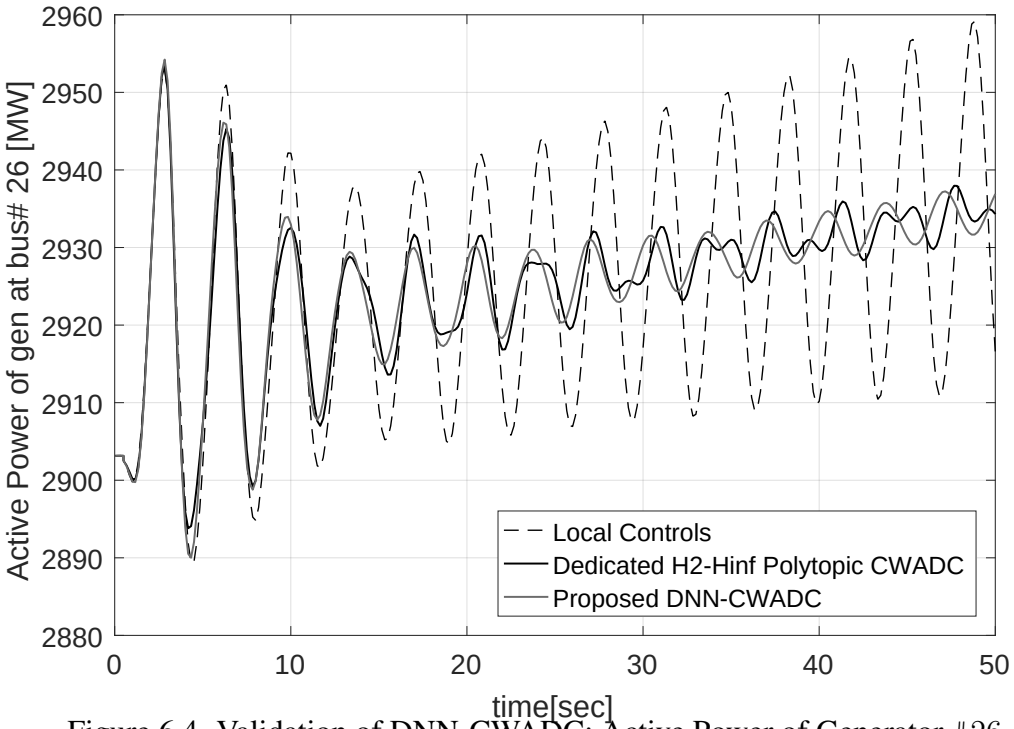

Figure 6.4. Validation of DNN-CWADC: Active Power of Generator #26.

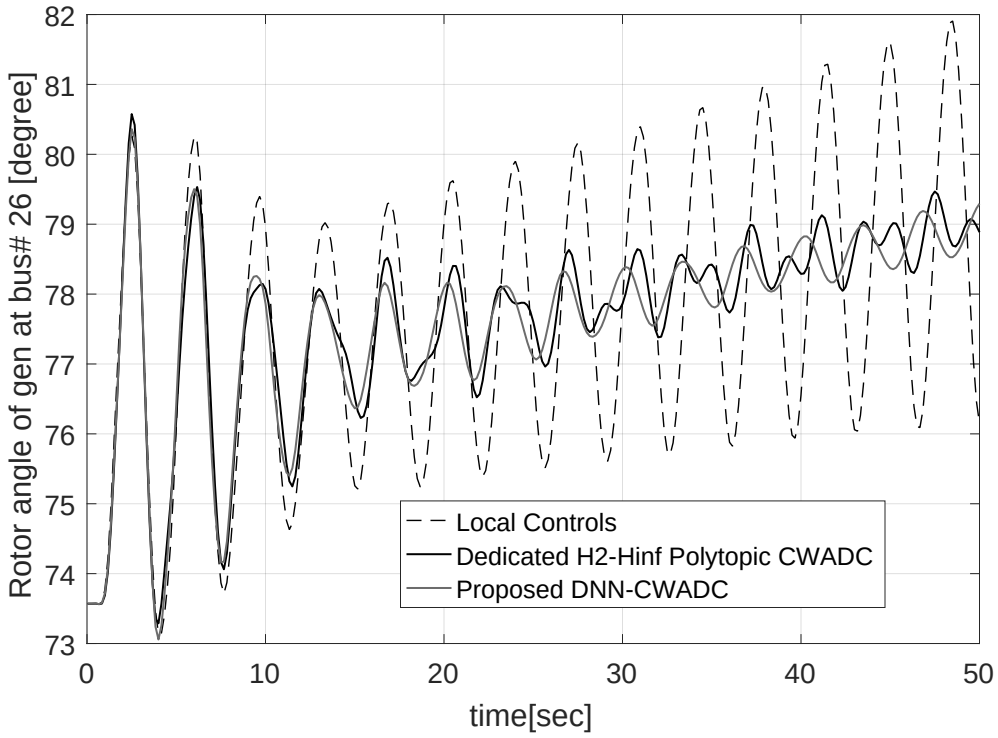

Figure 6.5. Validation of DNN-CWADC: Rotor Angle of Generator $\#26$.

c) Scenario 2-Testing of DNN-CWADC for N-1 contingency: To test the generalization and extrapolation capability of the trained model for DNN-based coordinated control, a test OC is selected that lies outside the training dataset. For this test, the DNN is asked to predict the polytopic controller gains for an outage case (N-1 contingency), i.e., outage of a line connecting bus $\#68$ and bus $\#77$. Fig. 6.6 displays the results for the modal analysis, while Figs. 6.7-6.10 show the time-domain responses of the system performed after applying the extrapolated DNN-CWADC to the system operating at this OC. A comparative analysis is performed by applying the actual polytopic control (designed specifically with this OC being one of its vertices, i.e., the Dedicated H_2/H_∞ Polytopic CWADC) and the predicted gains of DNN-CWADC. The results indicate that the proposed DNN-based control is capable of damping the cases that lie outside those for which it was trained.

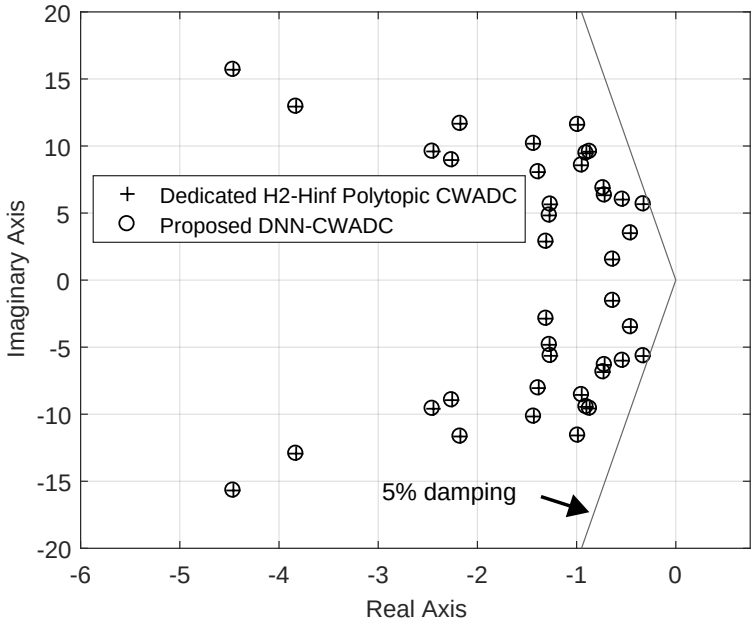

Figure 6.6. Testing of DNN-CWADC for N-1 Contingency: Modal Analysis.

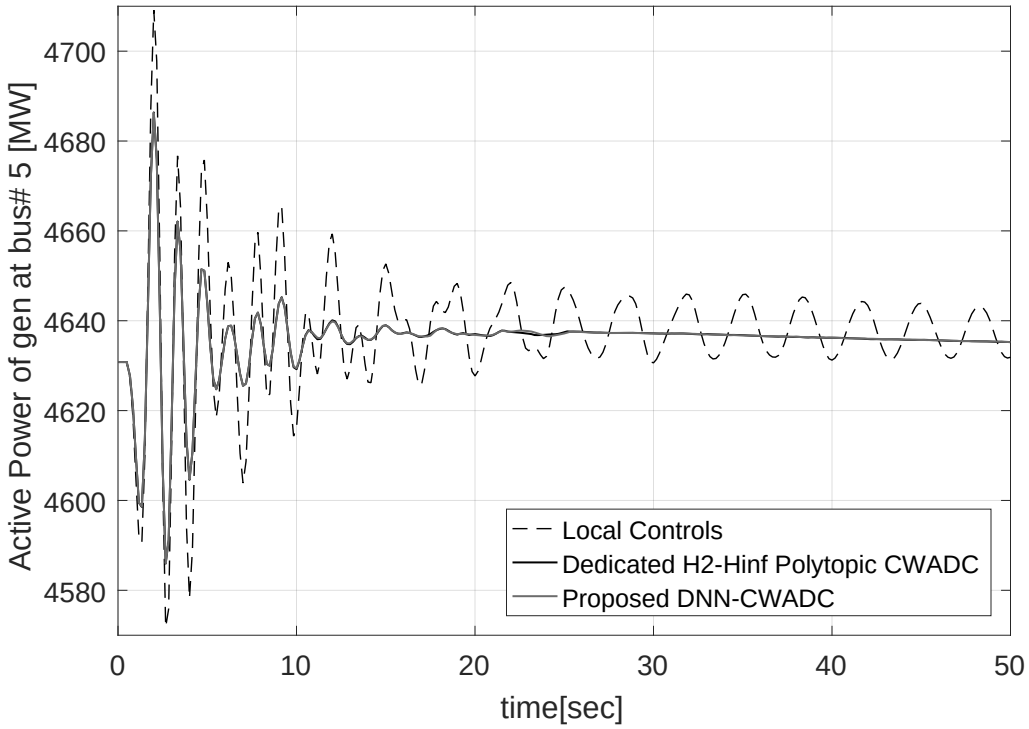

Figure 6.7. Testing of DNN-CWADC for N-1 Contingency: Active Power of Generator #5.

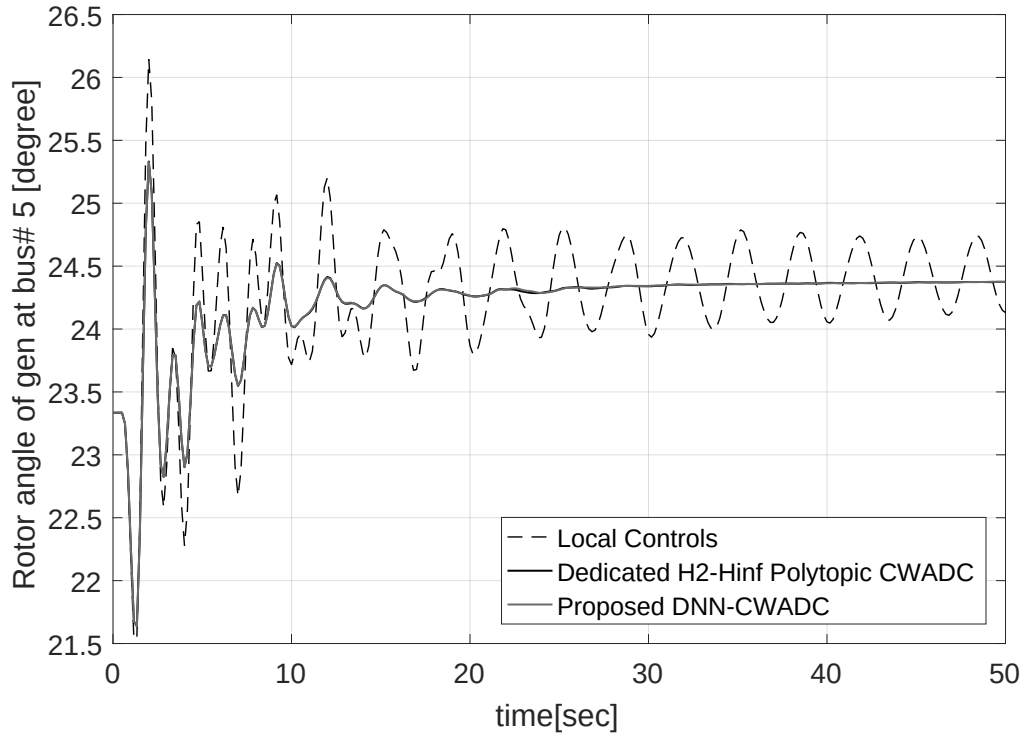

Figure 6.8. Testing of DNN-CWADC for N-1 Contingency: Rotor Angle of Generator #5.

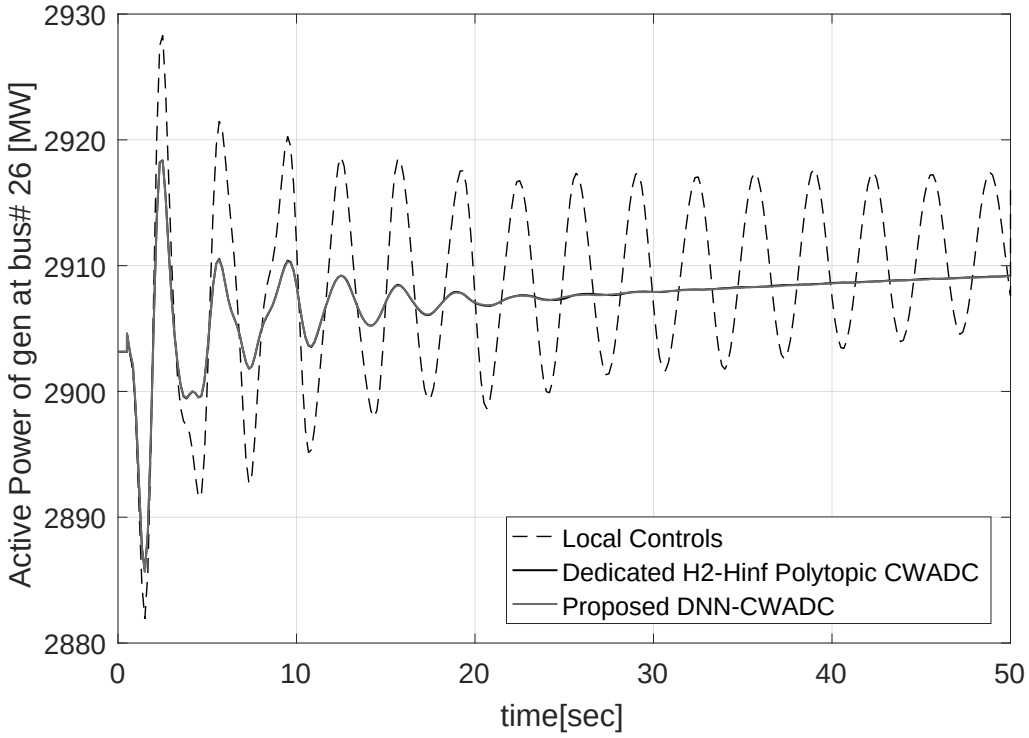

Figure 6.9. Testing of DNN-CWADC for N-1 Contingency: Active Power of Generator #26.

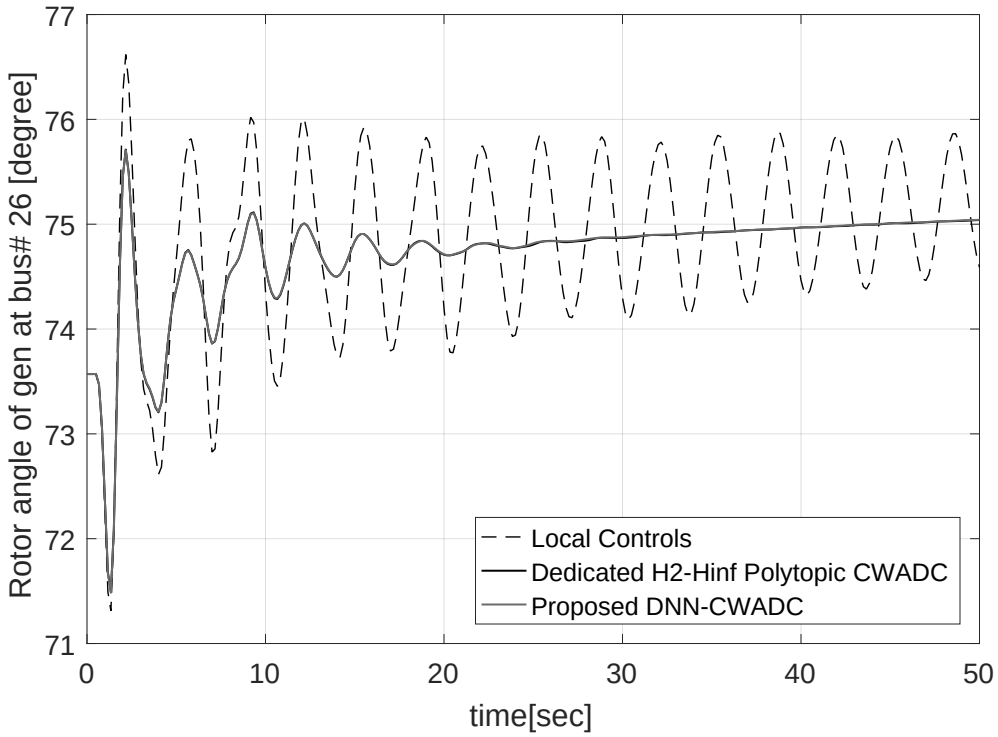

Figure 6.10. Testing of DNN-CWADC for N-1 Contingency: Rotor Angle of Generator #26.

Performance Evaluation for a NN-CWADC with a single hidden layer: The need for a DNN is further confirmed by an additional check conducted for Scenario 2. The modal analysis is performed by applying the gain obtained after training the NN with a single hidden layer having a testing MAE of 0.14 (see last paragraph of Section 6.1). The result shown in Fig. 6.11 establishes the relationship between the choice of the proper architecture of NN and the performance of the trained model. In the absence of two hidden layers, the extrapolated NN-CWADC cannot meet the predefined objective of having minimum damping of 5% for all the eigenvalues of the studied OC.

d) Scenario 3-Testing of DNN-CWADC for N-2 contingencies: This section investigates the stability assessment of the system under a N-2 contingency condition. The outages of two lines connecting bus $\#68$ and bus $\#77$ and bus $\#68$ and bus $\#71$ are selected as the test OC for evaluating the performance of the trained DNN-CWADC. Fig. 6.12

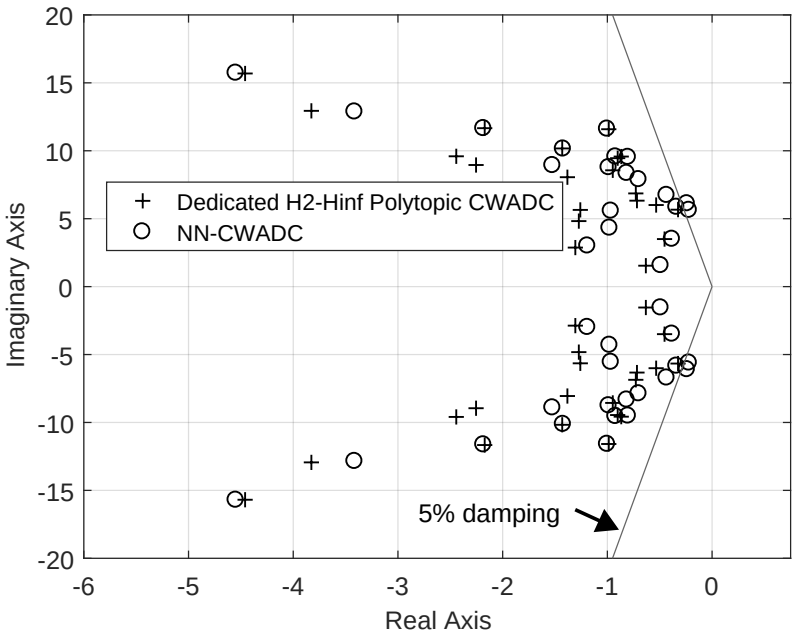

Figure 6.11. Validation of NN-CWADC: Modal Analysis.

shows the results of the eigenvalue analysis. It can be seen that all the modes are at least 5% damped. However, on conducting the time-domain simulations for the studied OC, the system exhibits sustained oscillations of smaller magnitude even in the presence of the designed DNN-based control (as evident from the active powers and branch flows of the most impacted generators and tie-lines shown by Fig. 6.13- Fig. 6.16). For such cases, where the application of the designed DNN-CWADC leads to unsatisfactory performance, DRL-based control can be put into service as mentioned in Section 4.2.

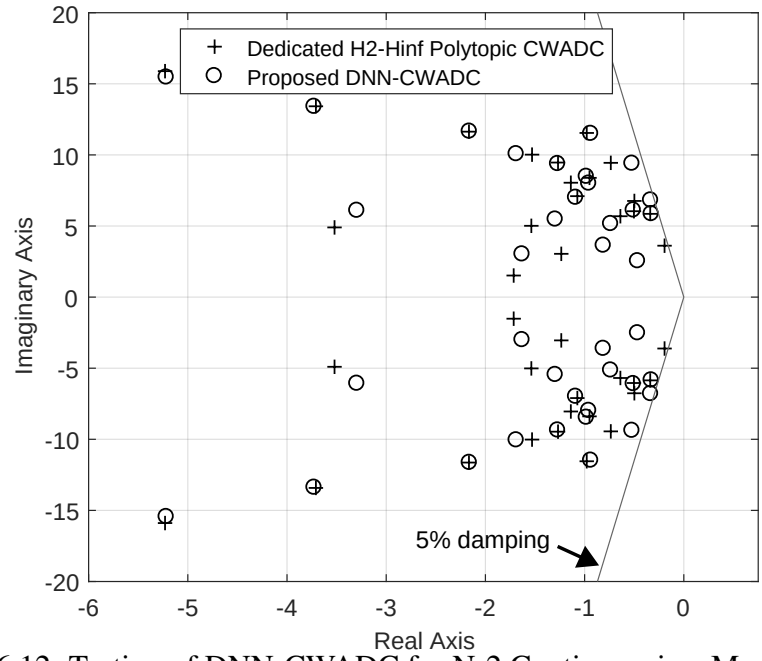

Figure 6.12. Testing of DNN-CWADC for N-2 Contingencies: Modal Analysis.

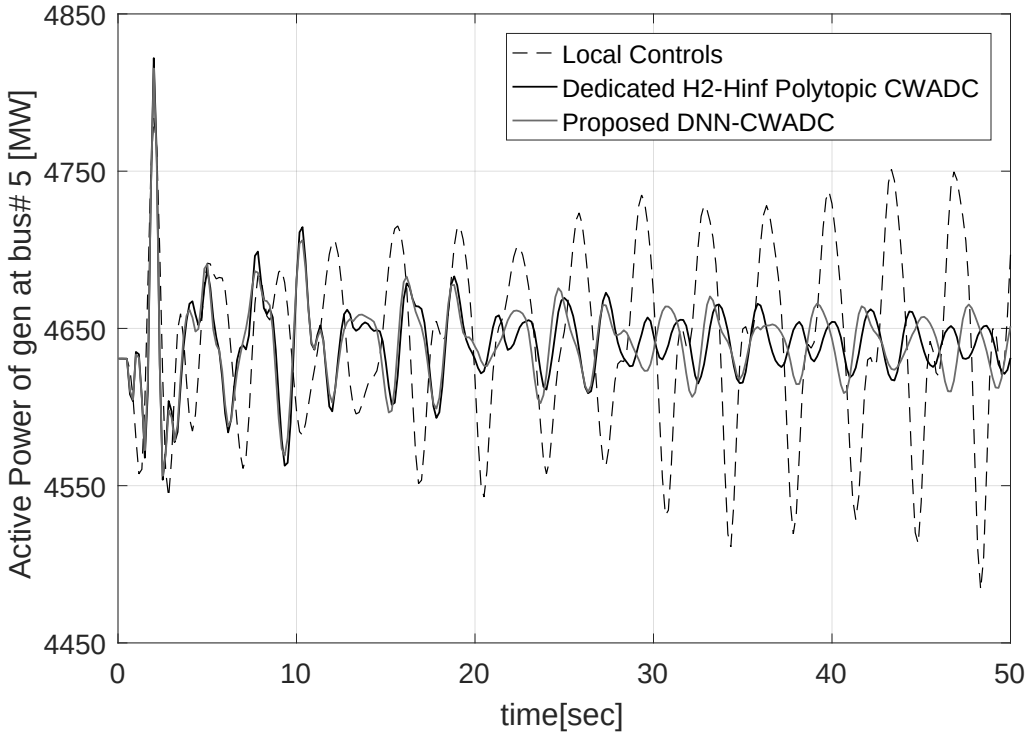

Figure 6.13. Testing of DNN-CWADC for N-2 Contingencies: Active Power of Generator #5.

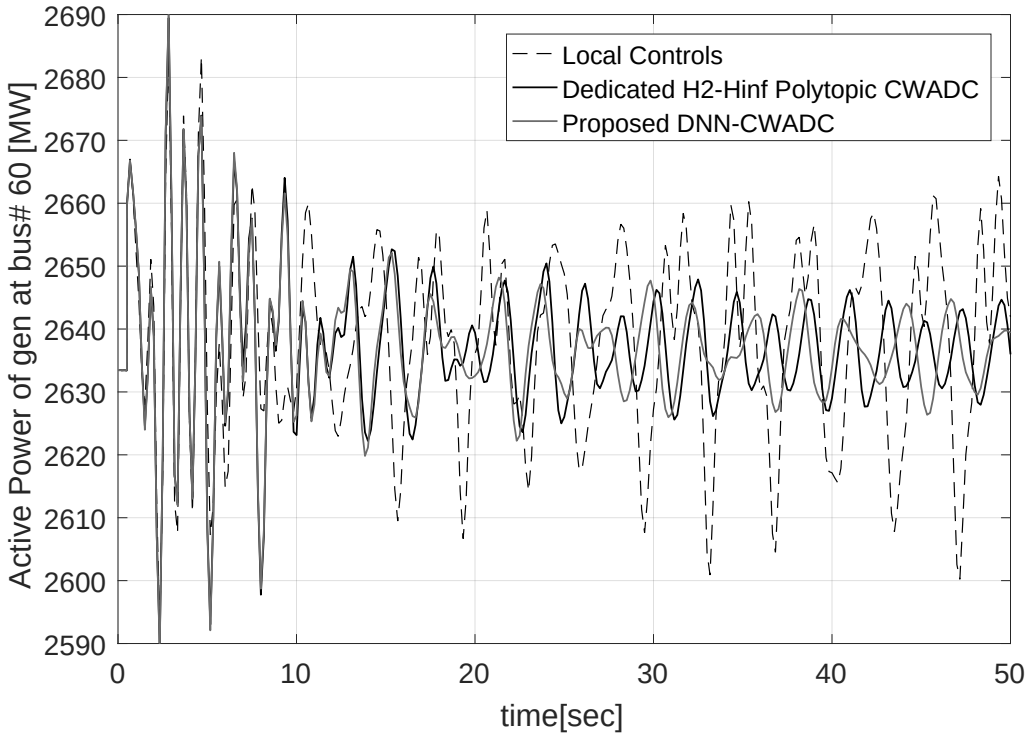

Figure 6.14. Testing of DNN-CWADC for N-2 Contingencies: Active Power of Generator #60.

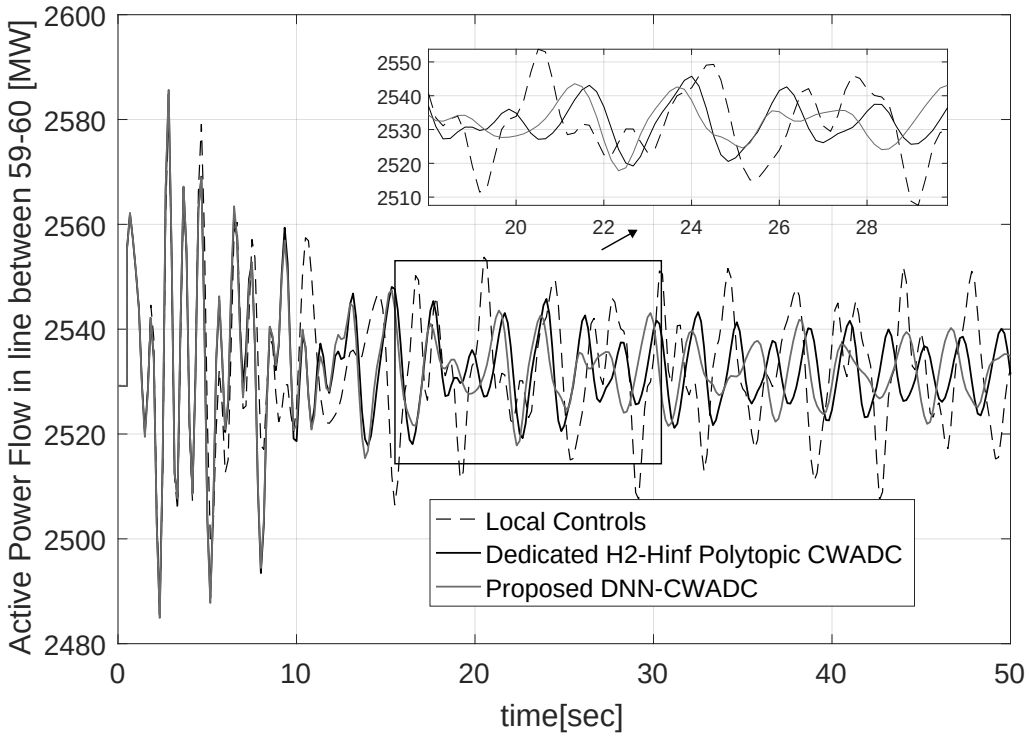

Figure 6.15. Testing of DNN-CWADC for N-2 Contingencies: Power Flow of Line #59-#60.

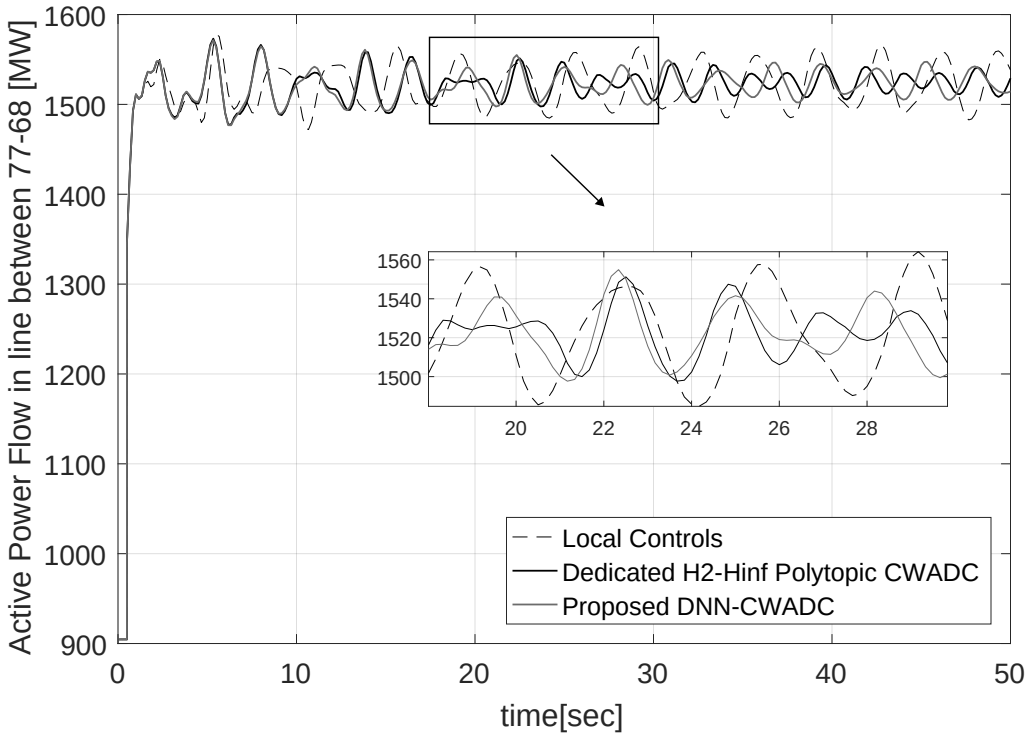

Figure 6.16. Testing of DNN-CWADC for N-2 Contingencies: Power Flow of Line #68-#77

6.2 Results for DRL-CWADC

An interface was created in-house to integrate the PSLF environment [98] with the DRL module. Python libraries were used to implement the BEC-DDPG algorithm for the DRL agent. The control law used by the DRL-CWADC in real-time was initially learned in an offline setting using time-domain simulations. The learning scenarios included short-circuit faults applied to the list of selected buses that connect the high-voltage lines and critical interfaces within and around the WECC Intertie Path #26. The scenario list can also be extended to accommodate some of the other limiting contingencies such as outages of critical transmission lines and generators.

The observation states used as the inputs for DRL-CWADC are the speed and relative rotor angles of 22 and 21 critical generators, respectively, of the reduced-order system iden-

tified using enhanced SMA [128]. Thus, the number of neurons in each of the input layers for the actor and the critic networks forming the DRL agent is 43. The number of nodes in the output layers is 24 (representing the 24 local damping controls that are coordinated using DRL-CWADC). Each of the two hidden layers that are part of the actor-critic networks is composed of 300 neurons. Other important hyperparameters are as follows: learning rate, $\eta = 0.0001$; noise parameters, $\sigma = 0.02$, and, $\sigma_d = 0.995$. The coefficients of the reward function defined by (4.1) and (4.2) are: $t_\alpha = 5$ sec, $t_\beta = 8$ sec, $\mathrm{d}_1 = 10$, $d_2 = 10$, and $d_3 = 1$. Additionally, to avoid the divergence of the power flows, the penalty factor is set to $-5,000$.

The training period of the DRL-CWADC is divided into different episodes. Each episode begins with a flat-start, with a three-phase short-circuit fault applied to the selected buses at time $t = 0.5$ sec, having a clearing time of 4 cycles. Every episode is simulated for 25 sec. During training, the DRL agent interacts with the PSLF simulation environment every 1 sec. The time step for the dynamic simulations in PSLF is set to one-quarter of a cycle. The trained model converges to a satisfactory solution for all the scenarios in 350-400 episodes. The performance of the DRL agent is shown in Fig. 6.17, which depicts the cumulative reward as a function of the number of episodes for one of the trained cases. The performance of the BEC-DDPG agent deteriorates post-episode 250; this can happen due to poor exploration by the agent [90]. However, with an increase in the number of episodes, the training becomes stable, and the BEC-DDPG converges to a reasonable value of the cumulative reward. Finally, it is noted that the proposed method is able to safeguard the system against any violations during the training, as evident from the magnitude of the cumulative reward (penalty factor is set to -5000 if safety constraint is violated)

To illustrate the robustness and adaptability of the learned DRL agent, four different cases are considered:

a) Case I: The system is subjected to a 100 ms three-phase fault at bus $\#68$ at $t = 0.5$

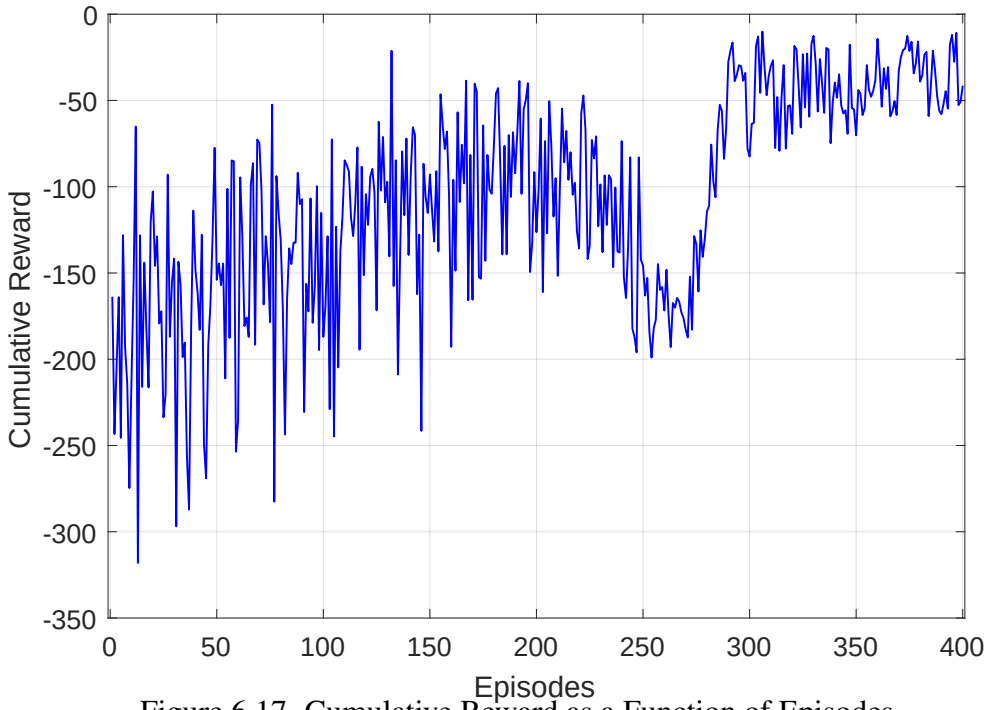

Figure 6.17. Cumulative Reward as a Function of Episodes.

sec. The active power responses of the two generators having higher participation in the critical LFO modes, namely, generators at bus $\#5$ and bus $\#91$ are shown in Fig. 6.18 and Fig. 6.19. It can be seen that the proposed DRL-CWADC is able to provide the requisite damping of at least 5% to the LFOs; for example, oscillation frequencies of 0.57 Hz, 0.7 Hz, and 1.3 Hz identified through Prony analysis show respective damping of 4.69%, 5.63%, and 3.46% in the presence of local controls, while they have damping of 20.09%, 5.23%, and 5.41%, respectively, using the designed DRL-CWADC. To illustrate the impact of the obtained results, rotor angle deviations of one of the generators is also shown in Fig. 6.20.

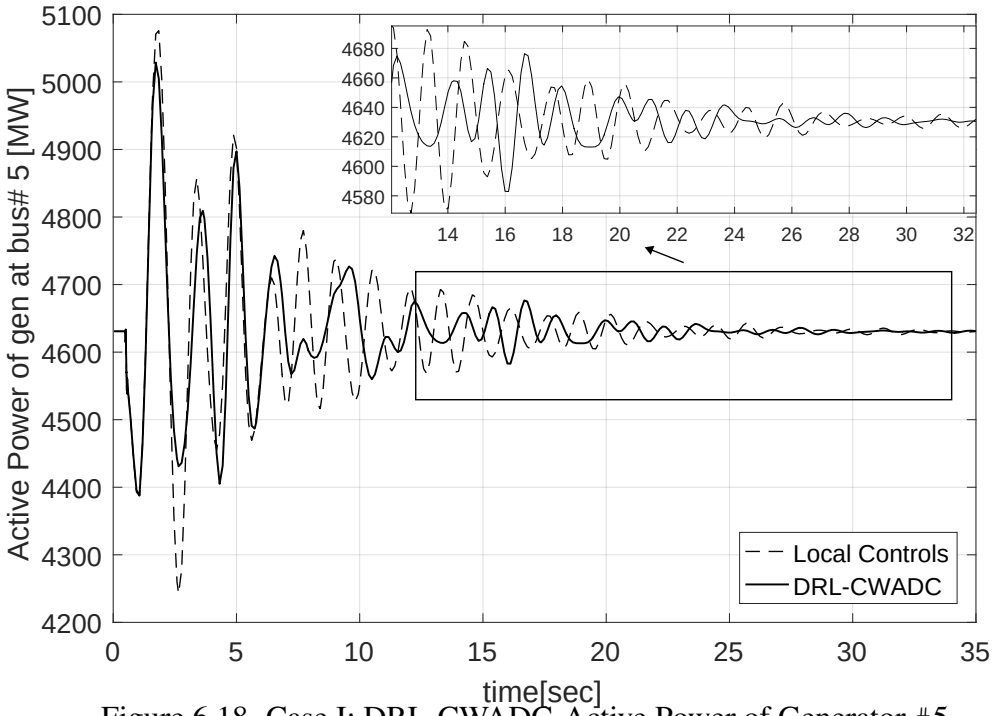

Figure 6.18. Case I: DRL-CWADC-Active Power of Generator #5.

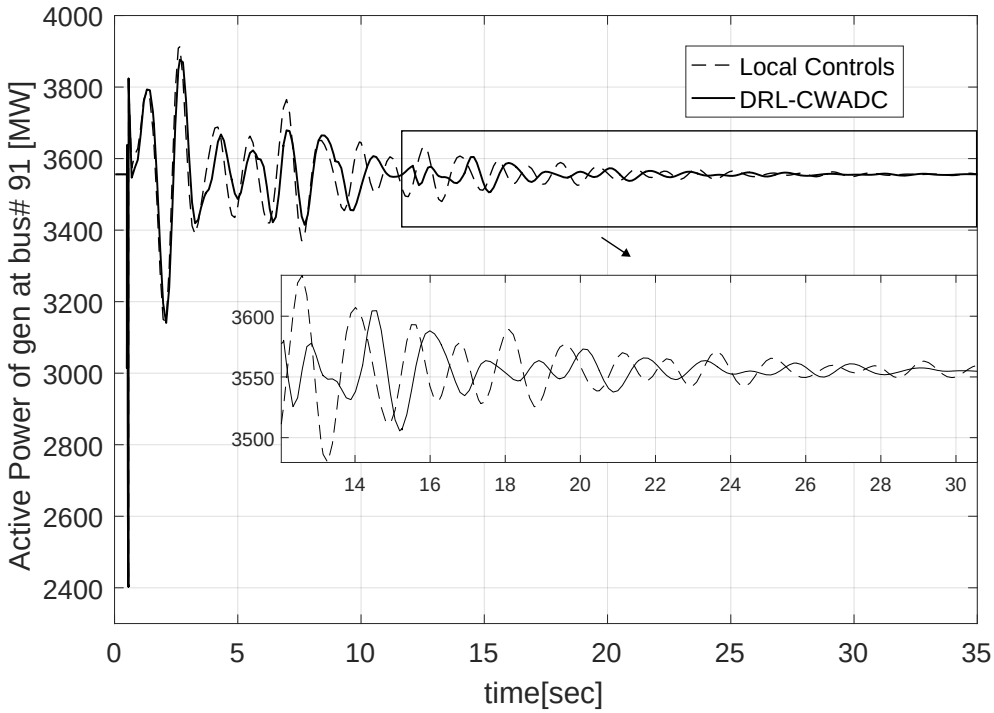

Figure 6.19. Case I: DRL-CWADC-Active Power of Generator #91.

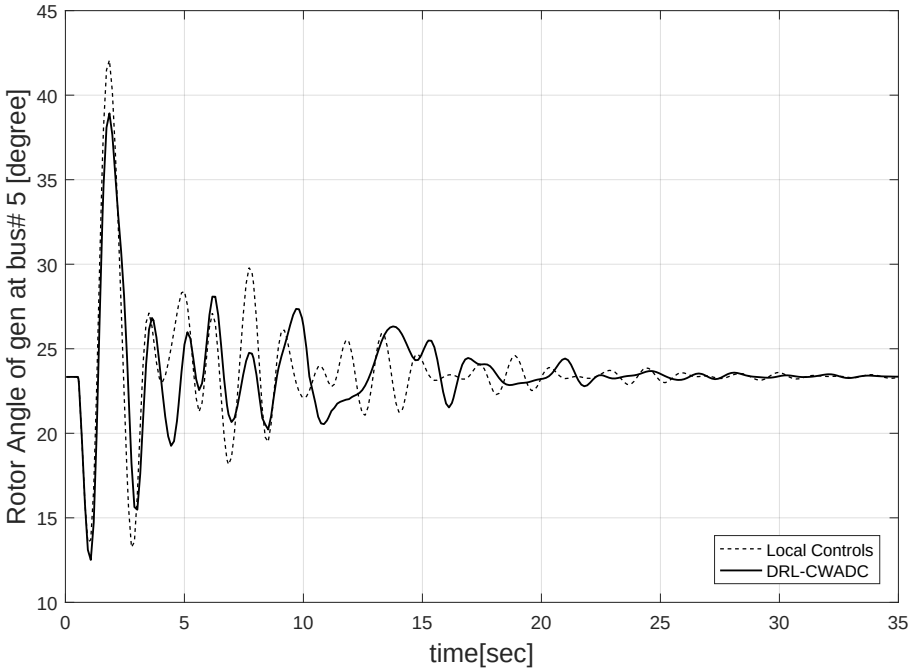

Figure 6.20. Case I: DRL-CWADC-Rotor Angle of Generator #5.

b) Case II: A 100 ms three-phase fault is applied at bus $\#66$ at $t = 0.5$ sec. Compared to Case I, the DRL agent required readjustment of its DNNs' parameters to adapt to this new OC. Figs. 6.21-6.22 show the active power responses of the two generators and rotor angle deviations of one of the generators (similar to Case I). The modal properties of the three dominant modes indicated that ζ_{min} attained by the system in the presence of local controls and DRL-CWADC are 2.81% and 9.82%, respectively. It can be seen that there is an improvement in the system's response after applying the adaptive control action generated in three episodes.

c) Case III: To check the robustness of the trained DRL-CWADC to different types of system uncertainties, the learned control is tested for a scenario that is *very* different from the fault scenarios used in the learning process, namely, loss of generation. The net generation of the generator at bus $\#60$ is reduced by 325 MW at $t = 0.5$ sec. For a better demonstration of the results, the length of the episode is set to 50 sec. The system becomes unstable owing to the depression of the voltage at one of the converter buses form-

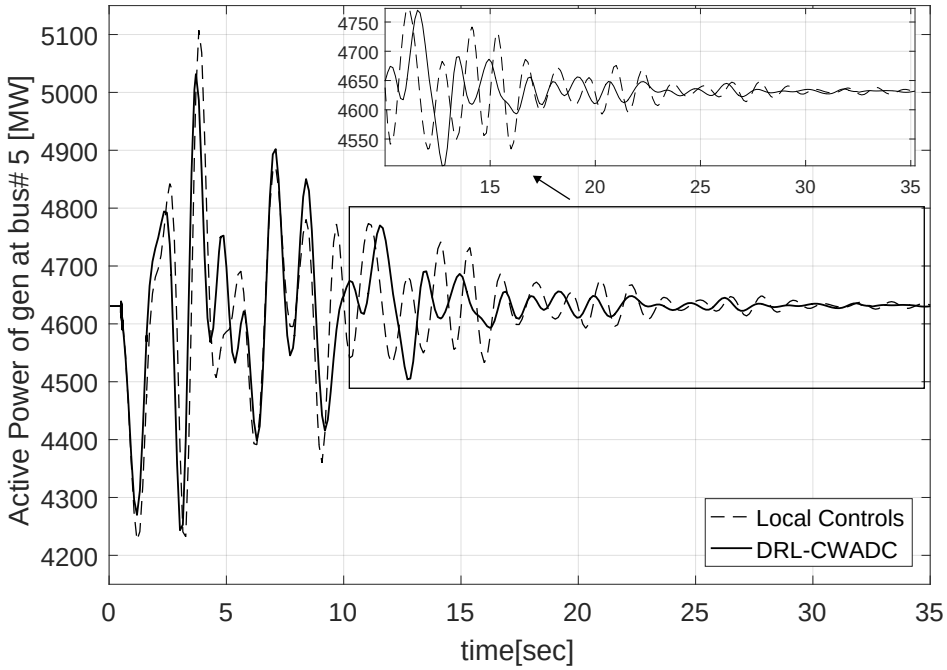

Figure 6.21. Case II: DRL-CWADC-Active Power of Generator #5.

ing the IPP, leading to system-wide low voltages for a longer time period and ultimately network divergence. The improved performance realized by employing two different DRL-CWADCs is evident in the results of the transient simulations shown in Figs. 6.24-6.26, which denote the active power responses of three different generators present at bus $\#60$, bus $\#62$ (a neighboring generating station), and bus $\#5$ (a generating unit in Canada), respectively. It is observed from the figures that when the DRL-CWADC is allowed to adjust its weights (DRL-CWADC-2), then its performance is (a) slightly better than when it is not allowed to adapt (DRL-CWADC-1), and (b) much better than when only the local controllers are present (Local Controls). However, the level of improvement, which depends on the nature and the location of the applied contingency, is less satisfactory than Cases I and II as one of the modes has a damping ratio of only 3.73% with DRL-CWADC-2 and 1.03% with DRL-CWADC-1. To realize a better damping action, the DRL agent may require reformulation of the DRL problem as mentioned in Section 4.2 of Chapter 4, which will be investigated in the future.

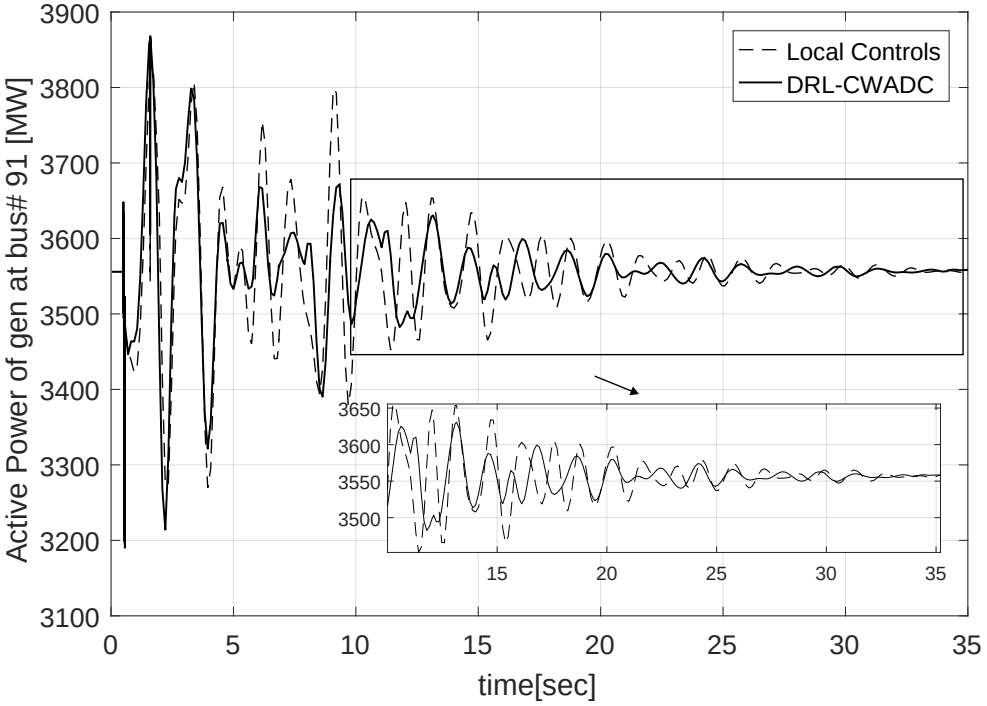

Figure 6.22. Case II: DRL-CWADC-Active Power of Generator #91.

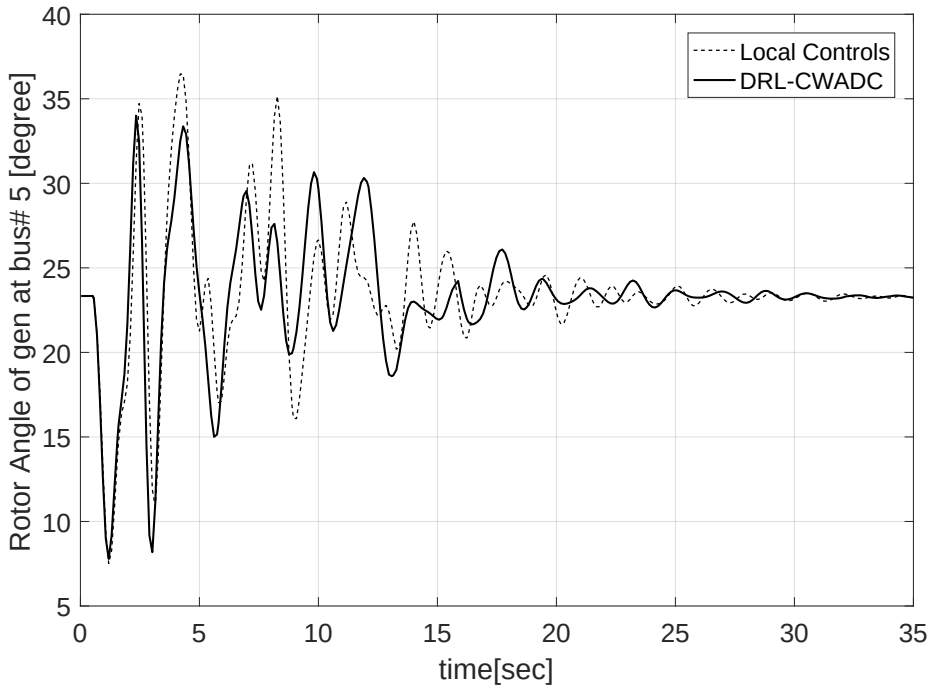

Figure 6.23. Case II: DRL-CWADC-Rotor Angle of Generator #5.

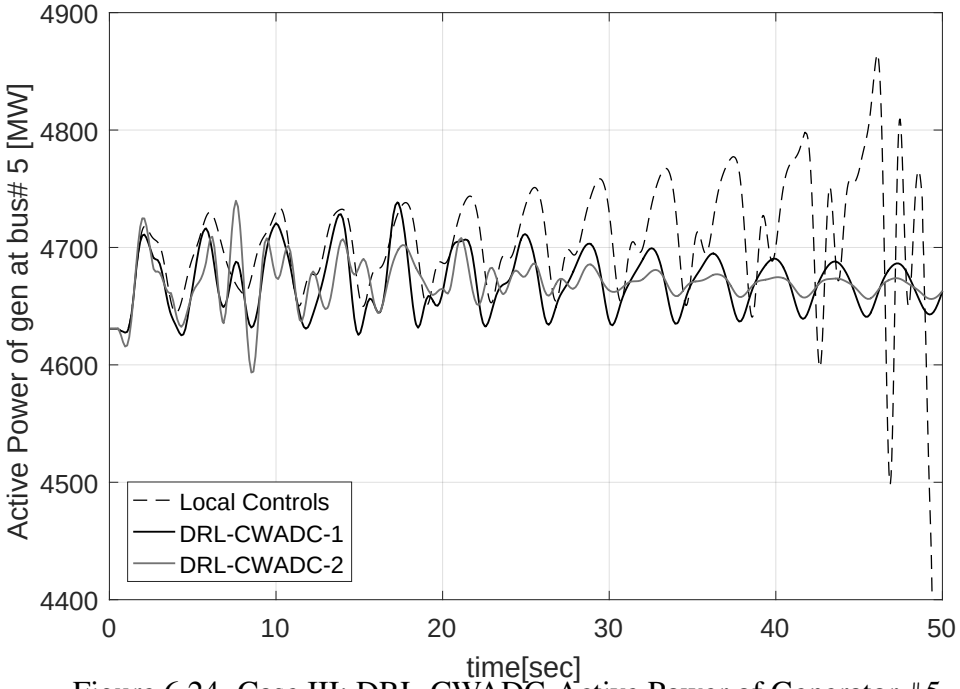

Figure 6.24. Case III: DRL-CWADC-Active Power of Generator #5.

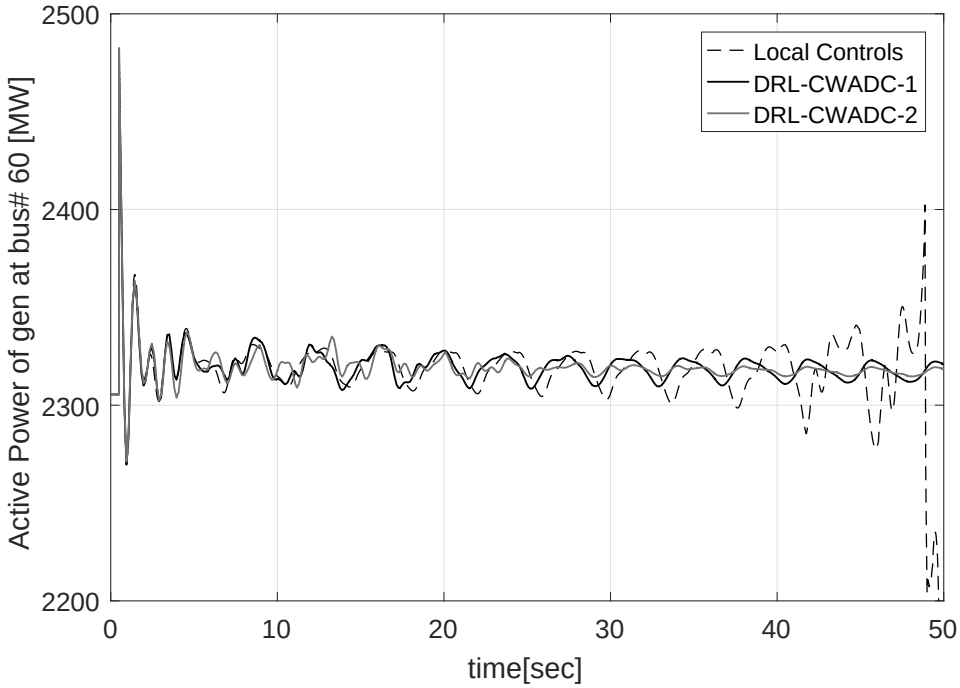

Figure 6.25. Case III: DRL-CWADC-Active Power of Generator #60.

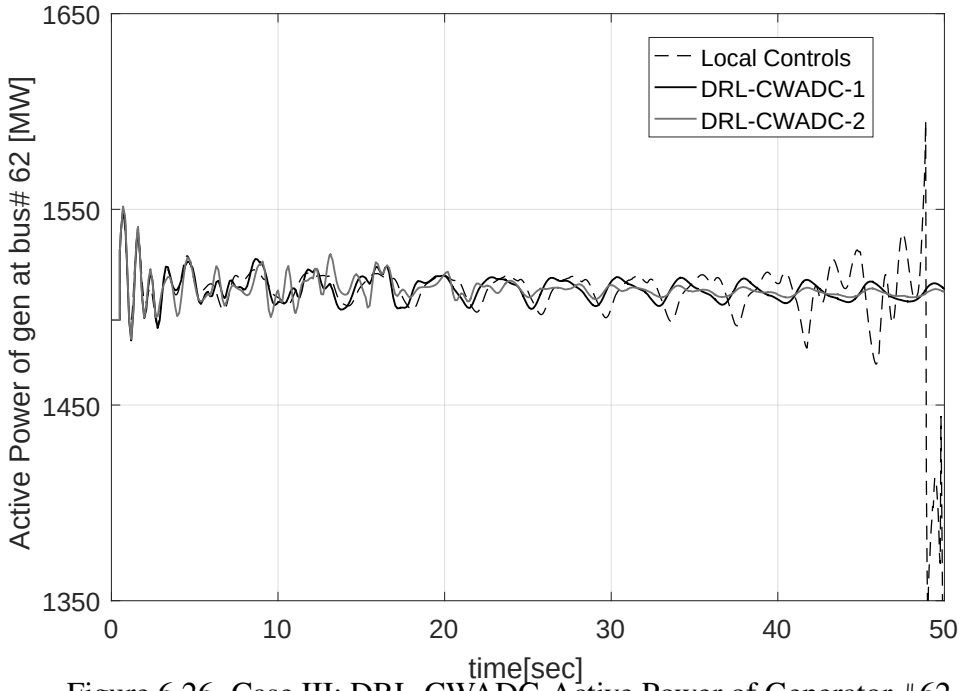

Figure 6.26. Case III: DRL-CWADC-Active Power of Generator #62.

d) Case IV: The effect of reward on improving the learning process for the DRL-CWADC is analyzed in this case. The reward function introduced in Section 4.2.2 is modified to include the gradients of all the branch flows of the system. The analysis is performed for Case I and Case III with two reward functions, i.e., with (DRL-CWADC-Reward2) and without (DRL-CWADC-Reward1) the use of gradients of power flows in 4.1. As shown in Figs. 6.27-6.28, the responses of the system for two different OCs (cases) using a different reward function is comparable to that obtained in Figs. 6.18 and 6.25. However, no further improvement in the results was observed.

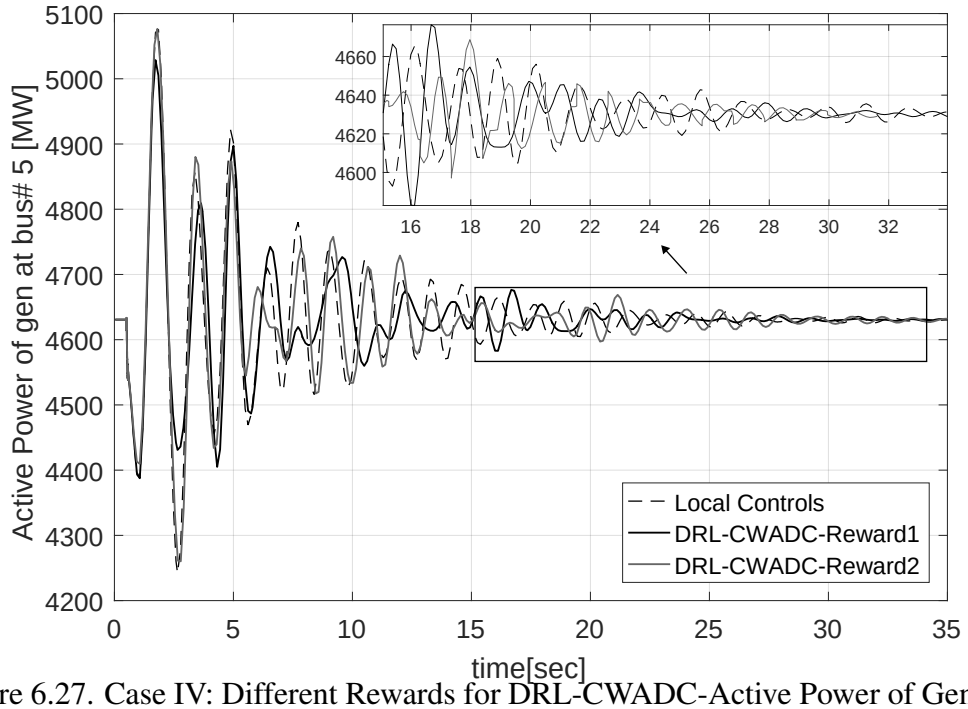

Figure 6.27. Case IV: Different Rewards for DRL-CWADC-Active Power of Generator #5 (Case I).

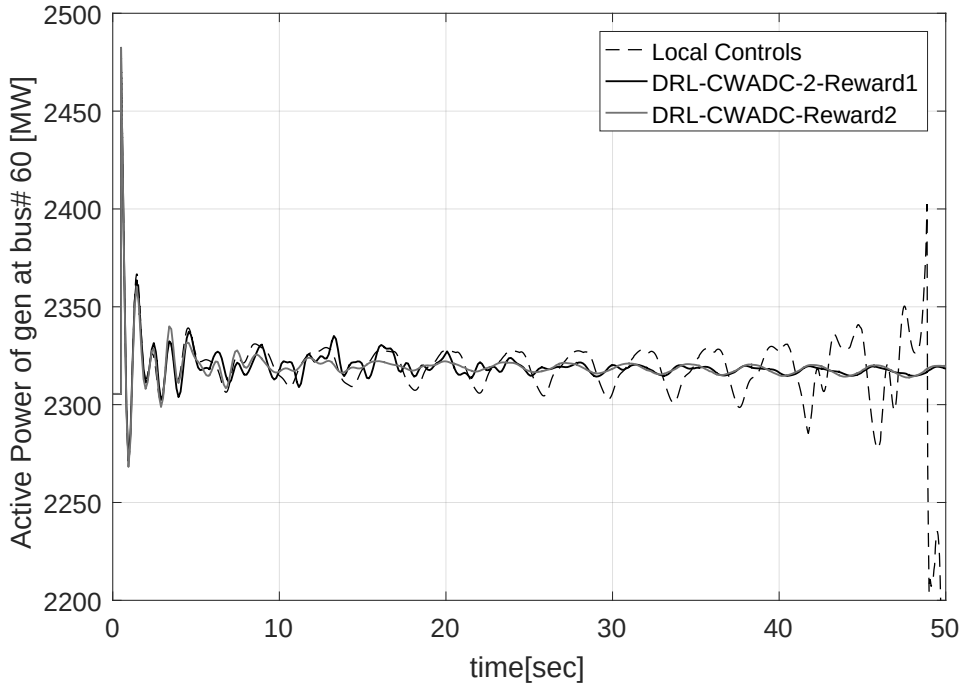

Figure 6.28. Case IV: Different Rewards for DRL-CWADC-Active Power of Generator #60 (Case III).

6.3 Considerations for Real-Time Implementation

The successful application of the two damping control schemes in real-time relies on many practical considerations:

a) Inconsistencies in PMU data: In practice, PMU data may be subjected to data inconsistencies such as measurement noises, loss of communication channel, and delays:

- Measurement noises: The control design of both methods inherently considers the noise/disturbance attenuation using H_∞ control. Furthermore, the robustness of the DNN-CWADC to presence of noise in input measurements has already been demonstrated in Section 6.1.2, while it has already been shown in [90] that the decision-making capability of DRLs is minimally impacted by noisy measurements.

- Failure of a communication channel and delays in wide-area signals: To tackle the problem of missing PMU data, an alternate signal selection scheme is proposed in **Algorithm 3** of Section 3.2.5. This can also be utilized for the proposed control schemes in case of data dropout or failure of a communication channel. Moreover, the proposed bi-level control design ensures that, in case of problems with the wide-area signals, the local controls can work autonomously using the locally available information. Lastly, in the implementation of the DRL-based control, the DRL agent does not require the back-to-back samples of the states (interaction between the DRL agent and power system environment happens every 1 sec.). As such, it can essentially handle the delay and data dropouts in PMU signals for at least 1 sec.

b) Guarantee for safe control: In this paper, bounded exploratory constraints based on H_2/H_∞ control are considered for developing a safe policy for DRL-based control. Similarly, for DNN-based control, a careful selection of weights for H_∞ and H_2 controls is made to ensure that the synthesized feedback gains are kept at reasonable values. There-

fore, the learned DNN and DRL-based controls are expected to not create any violations in the system. As an additional measure, limiters are included for the individual controls to provide both stable as well as secure operation in real-time.

c) Parameter selection for DRL-CWADC: The usefulness of the DRL-based damping control is critically dependent on the MDP problem formulation, including the choice of the appropriate reward functions. One way to resolve this issue is to use imitation learning (refer to Section 2.4) for mapping the operating states of the grid to the effective actions based on the demonstration data (historical operational data in power systems) provided by the system operators and engineers [137]. Another solution is to use inverse RL to infer a better reward function from the same data [117].

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

This research work deals with a thorough assessment of the improvement in small-signal stability of the power grid using wide-area signals. Two different approaches are developed to design the damping controllers that are robust to variable OCs. Both approaches share two similarities a) the same wide-area measurements derived from the enhanced SMA and partial state-feedback are utilized to design the controls, b) the parameters of the existing local controls are not modified, i.e., the improvement in performance was due to the wide-area signals alone.

In the first approach, a wide-area damping controller is developed for mitigating inter-area oscillations using an enhanced SMA and LMI-based polytope. The method facilitates the coordination of existing controls in the system, thereby enabling them to overcome their inability to act based on local measurements to damp the oscillation modes. Additionally, to ensure robustness, the coordinated controller is designed to switch between the primary and alternate sets of feedback signals in case of failure of a communication device or a measurement channel. This resulting controller is a single gain matrix capable of enhancing the damping of all the OCs lying inside the polytope. Modal analysis and time-domain simulations are performed on a 16-machine 68 bus system and the 29 machine, 127 bus

equivalent model of the WECC system (embedded with three HVDC lines) to test the viability of the proposed control. The practical issue of delay in the control signals is analyzed by adding variable latency to different control signals. Simulation results suggest that the use of this scheme can be advantageous for providing additional damping to the critical oscillatory modes despite the presence of reasonable delays.

Unlike the first approach, the second method improves the damping of low frequency oscillations by developing two different AI-based wide-area controllers. The agent administered DNN and DRL control frameworks can identify and track the dynamics of the system and automatically take control actions to increase system damping. Based on the concept of a bi-level control scheme, the robust controllers can also coordinate with the local damping controllers for a wide range of OCs. A selection technique is proposed to apply the appropriate wide-area control based on the nature of the system disturbance.

The developed AI-based techniques are applied to the equivalent model of the WECC system. Test results obtained after performing modal analysis and non-linear time-domain simulations demonstrate the effectiveness of both designs for various OCs, ranging from lowly-stressed to highly stressed network conditions. As the functioning of the proposed controls can be limited by the *a priori* system analysis, it is also shown that the designed DRL-CWADC can adapt to the time-varying real-time system conditions by exercising a safe DRL policy and provide improved performance. Therefore, the proposed techniques can be viewed as powerful tools for assisting grid operators in ensuring the safe operation of the system under practical OCs.

7.2 Suggested Future Work

To further develop the ideas and methods which have been established in this work, some of the potential research areas that were identified are as follows:

1. Due to delays present in the wide-area signals, communication latency may affect the performance of the proposed AI-based CWADCs. Other issues related to data quality, such as missing synchrophasor measurements and bad data, can also negatively influence controller actions. Further study on this topic is therefore necessary and important.

2. The DRL-based control has been developed for providing the requisite control actions for the fault-based scenarios. However, this method is currently unable to account for different types of transient contingencies. This should be investigated to widen the applicability of the DRL-CWADC for mitigating additional instability issues arising due to poorly damped oscillations.

3. As a part of this work, centralized control schemes are designed which require remote feedback signals over long distances. However, the major drawback of this type of architecture is the inherent delay associated with the wide-area signals. An alternative approach would be to use a decentralized framework for damping controllers. In recent years, multi-agent RL (MARL) system has attracted a lot of attention. It is a network of autonomous interacting entities sharing a common environment to solve problems by sensing the environment and acting based on their perceptions [138]. This system offers great potential for designing decentralized damping controllers. The control structures can also take advantage of the wide-area signals to quickly damp out the oscillations after a severe disturbance.

4. The operation of power grids usually has several objectives, such as maximization of stability, security, and/or resiliency. For a DRL-based problem, it can be challenging to balance the various operational objectives in the reward function. This problem can be overcome by imitation learning (IL), where the agent tries to learn the optimal control policy based on the set of demonstration trajectories (historical data in power

systems) provided by the system experts [117, 137]. An alternative strategy is to use inverse RL to model a better reward function from the same data [117].

5. For the implementation of the DRL-based CWADC, an in-house interface was created. However, it is essential to design a more robust interface for the development and benchmarking of DRL algorithms for grid control.

REFERENCES

[1] P. Du, W. Li, X. Ke, N. Lu, O. A. Ciniglio, M. Colburn, and P. M. Anderson, "Probabilistic-based available transfer capability assessment considering existing and future wind generation resources," *IEEE Trans. Sustain. Energy*, vol. 6, no. 4, pp. 1263–1271, 2015.

[2] H. Gu, R. Yan, and T. K. Saha, "Minimum synchronous inertia requirement of renewable power systems," *IEEE Trans. Power Syst.*, vol. 33, no. 2, pp. 1533–1543, 2017.

[3] H. Weber and S. Al Ali, "Influence of huge renewable power production on inter area oscillations in the european ENTSO-E system," *IFAC-PapersOnLine*, vol. 49, no. 27, pp. 12–17, 2016.

[4] G. Rogers, *Power system oscillations*. Springer Science & Business Media, 2012.

[5] M. Klein, G. J. Rogers, and P. Kundur, "A fundamental study of inter-area oscillations in power systems," *IEEE Trans. Power Syst.*, vol. 6, no. 3, pp. 914–921, 1991.

[6] D. N. Kosterev, C. W. Taylor, and W. A. Mittelstadt, "Model validation for the august 10, 1996 WSCC system outage," *IEEE Trans. Power Syst.*, vol. 14, no. 3, pp. 967–979, 1999.

[7] "1996 system disturbances: Review of selected 1996 electricty system disturbances in North America," North American Electric Reliability Council. Avaialble: https://www.nerc.com, 2002.

[8] O. Tor, C. Gencoglu, O. Yilmaz, E. Cebeci, and A. Guven, "Damping measures against prospective oscillations between turkish grid and ENTSO-E system," in *2010 Int. Conf. Power Syst. Technol.* IEEE, 2010, pp. 1–7.

[9] E. Martinez and A. Messina, "Modal analysis of measured inter-area oscillations in

the mexican interconnected system: The july 31, 2008 event," in *2011 IEEE Power & Energy Soc. General Meeting*. IEEE, 2011, pp. 1–8.

[10] F. Aboytes, F. Sanchez, A. M. Cabra, and J. G. Castro, "Dynamic stability analysis of the interconnected colombia-venezuela power system," *IEEE Trans. Power Syst.*, vol. 15, no. 1, pp. 376–381, 2000.

[11] D. Trudnowski, D. Kosterev, and J. Undrill, "PDCI damping control analysis for the western north american power system," in *2013 IEEE Power & Energy Soc. General Meeting*. IEEE, 2013, pp. 1–5.

[12] E. Larsen and D. Swann, "Applying power system stabilizers part I: general concepts, part II: performance objectives and tuning concepts, part III: practical considerations," *IEEE Trans. Power App. Syst.*, no. 6, pp. 3017–3046, 1981.

[13] P. Kundur, N. J. Balu, and M. G. Lauby, *Power system stability and control*. McGraw-hill New York, 1994, vol. 7.

[14] P. Kundur, M. Klein, G. Rogers, and M. S. Zywno, "Application of power system stabilizers for enhancement of overall system stability," *IEEE Trans. Power Syst.*, vol. 4, no. 2, pp. 614–626, 1989.

[15] R. Grondin, I. Kamwa, G. Trudel, L. Gerin-Lajoie, and J. Taborda, "Modeling and closed-loop validation of a new PSS concept, the multi-band PSS," in *2003 IEEE Power & Energy Soc. General Meeting (IEEE Cat. No. 03CH37491)*, vol. 3. IEEE, 2003, pp. 1804–1809.

[16] M. J. Gibbard, D. Vowles, and P. Pourbeik, *Small-signal stability, Control and Dynamic Performance of Power Systems*. University of Adelaide press, 2015.

[17] Y. Abdel-Magid, M. Abido, S. Al-Baiyat, and A. Mantawy, "Simultaneous stabilization of multimachine power systems via genetic algorithms," *IEEE Trans. Power Syst.*, vol. 14, no. 4, pp. 1428–1439, 1999.

[18] M. Abido, "Optimal design of power-system stabilizers using particle swarm optimization," *IEEE Trans. Energy. Conv.*, vol. 17, no. 3, pp. 406–413, 2002.

[19] A. L. Do Bomfim, G. N. Taranto, and D. M. Falcao, "Simultaneous tuning of power system damping controllers using genetic algorithms," *IEEE Trans.Power Syst.*,

vol. 15, no. 1, pp. 163–169, 2000.

[20] R. A. Ramos, L. F. Alberto, and N. G. Bretas, "A new methodology for the coordinated design of robust decentralized power system damping controllers," *IEEE Trans. Power Syst.*, vol. 19, no. 1, pp. 444–454, 2004.

[21] S. Ghosh, M. S. El Moursi, E. F. El-Saadany, and K. Al Hosani, "Online coherency based adaptive wide area damping controller for transient stability enhancement," *IEEE Trans. Power Syst.*, vol. 35, no. 4, pp. 3100–3113, 2019.

[22] W. Qiu, V. Vittal, and M. Khammash, "Decentralized power system stabilizer design using linear parameter varying approach," *IEEE Trans. Power Syst.*, vol. 19, no. 4, pp. 1951–1960, 2004.

[23] Y. Zhou, J. Liu, Y. Li, C. Gan, H. Li, and Y. Liu, "A gain scheduling wide-area damping controller for the efficient integration of photovoltaic plant," *IEEE Trans Power Syst.*, vol. 34, no. 3, pp. 1703–1715, 2018.

[24] M. Ghandhari, G. Andersson, and I. A. Hiskens, "Control lyapunov functions for controllable series devices," *IEEE Trans. Power Syst.*, vol. 16, no. 4, pp. 689–694, 2001.

[25] B. Chaudhuri and B. C. Pal, "Robust damping of multiple swing modes employing global stabilizing signals with a TCSC," *IEEE Trans. Power Syst.*, vol. 19, no. 1, pp. 499–506, 2004.

[26] Q. Liu, V. Vittal, and N. Elia, "LPV supplementary damping controller design for a thyristor controlled series capacitor (TCSC) device," *IEEE Trans. Power Syst.*, vol. 21, no. 3, pp. 1242–1249, 2006.

[27] M. Zarghami, M. L. Crow, and S. Jagannathan, "Nonlinear control of FACTS controllers for damping interarea oscillations in power systems," *IEEE Trans. Power Del.*, vol. 25, no. 4, pp. 3113–3121, 2010.

[28] P. Pourbeik and M. J. Gibbard, "Simultaneous coordination of power system stabilizers and FACTS device stabilizers in a multimachine power system for enhancing dynamic performance," *IEEE Trans. Power Syst.*, vol. 13, no. 2, pp. 473–479, 1998.

[29] L.-J. Cai and I. Erlich, "Simultaneous coordinated tuning of PSS and FACTS damp-

ing controllers in large power systems," *IEEE Trans. Power Syst.*, vol. 20, no. 1, pp. 294–300, 2005.

[30] T. Nguyen and R. Gianto, "Optimisation-based control coordination of PSSs and FACTS devices for optimal oscillations damping in multi-machine power system," *IET Gener., Transmiss. & Distrib.*, vol. 1, no. 4, pp. 564–573, 2007.

[31] W. Juanjuan, F. Chuang, and Z. Yao, "Design of WAMS-based multiple HVDC damping control system," *IEEE Trans. Smart Grid*, vol. 2, no. 2, pp. 363–374, 2011.

[32] Y. Pipelzadeh, B. Chaudhuri, and T. C. Green, "Control coordination within a vsc HVDC link for power oscillation damping: A robust decentralized approach using homotopy," *IEEE Trans. Control Syst. Technol.*, vol. 21, no. 4, pp. 1270–1279, 2012.

[33] P. Agnihotri, A. Kulkarni, A. M. Gole, B. A. Archer, and T. Weekes, "A robust wide-area measurement-based damping controller for networks with embedded multiterminal and multiinfeed HVDC links," *IEEE Trans. Power Syst.*, vol. 32, no. 5, pp. 3884–3892, 2017.

[34] M. E. Aboul-Ela, A. Sallam, J. D. McCalley, and A. Fouad, "Damping controller design for power system oscillations using global signals," *IEEE Trans. Power Syst.*, vol. 11, no. 2, pp. 767–773, 1996.

[35] G. Heydt, C.-C. Liu, A. Phadke, and V. Vittal, "Solution for the crisis in electric power supply," *IEEE Computer Appl. in Power*, vol. 14, no. 3, pp. 22–30, 2001.

[36] H. Ni, G. T. Heydt, and L. Mili, "Power system stability agents using robust wide area control," *IEEE Trans. Power Syst.*, vol. 17, no. 4, pp. 1123–1131, 2002.

[37] G. Sánchez-Ayala, V. Centeno, and J. Thorp, "Gain scheduling with classification trees for robust centralized control of PSSs," *IEEE Trans. Power Syst.*, vol. 31, no. 3, pp. 1933–1942, 2015.

[38] A. Chakrabortty, "Wide-area damping control of power systems using dynamic clustering and TCSC-based redesigns," *IEEE Trans. Smart Grid*, vol. 3, no. 3, pp. 1503–1514, 2012.

[39] S. Zhang and V. Vittal, "Design of wide-area power system damping controllers resilient to communication failures," *IEEE Trans. Power Syst.*, vol. 28, no. 4, pp.

4292–4300, 2013.

[40] B. J. Pierre, F. Wilches-Bernal, D. A. Schoenwald, R. T. Elliott, D. J. Trudnowski, R. H. Byrne, and J. Neely, "Design of the Pacific DC Intertie wide area damping controller," *IEEE Trans. Power Syst.*, 2019.

[41] A. Pal and J. S. Thorp, "Co-ordinated control of inter-area oscillations using SMA and LMI," in *Proc. 2012 IEEE PES Innovative Smart Grid Technologies*, 2012, pp. 1–6.

[42] A. Pal, J. S. Thorp, S. S. Veda, and V. Centeno, "Applying a robust control technique to damp low frequency oscillations in the WECC," *Int.l J.Elect. Power & Energy Syst.*, vol. 44, no. 1, pp. 638–645, 2013.

[43] T. Wang, A. Pal, J. S. Thorp, and Y. Yang, "Use of polytopic convexity in developing an adaptive interarea oscillation damping scheme," *IEEE Trans. Power Syst.*, vol. 32, no. 4, pp. 2509–2520, 2017.

[44] T. Wang, A. Pal, J. S. Thorp, Z. Wang, J. Liu, and Y. Yang, "Multi-polytope-based adaptive robust damping control in power systems using cart," *IEEE Trans. Power Syst.*, vol. 30, no. 4, pp. 2063–2072, 2014.

[45] Y. Zhang and A. Bose, "Design of wide-area damping controllers for interarea oscillations," *IEEE Trans. Power Syst.*, vol. 23, no. 3, pp. 1136–1143, 2008.

[46] N. Mithulananthan, C. A. Canizares, J. Reeve, and G. J. Rogers, "Comparison of PSS, SVC and STATCOM controllers for damping power system oscillations," *IEEE Trans. Power Syst.*, vol. 18, no. 2, pp. 786–792, 2003.

[47] R. Majumder, B. C. Pal, C. Dufour, and P. Korba, "Design and real-time implementation of robust FACTS controller for damping inter-area oscillation," *IEEE Trans. Power Syst.*, vol. 21, no. 2, pp. 809–816, 2006.

[48] Y. Li, C. Rehtanz, S. Ruberg, L. Luo, and Y. Cao, "Wide-area robust coordination approach of HVDC and FACTS controllers for damping multiple interarea oscillations," *IEEE Trans. Power Del.*, vol. 27, no. 3, pp. 1096–1105, 2012.

[49] R. Preece, J. V. Milanovic, A. M. Almutairi, and O. Marjanovic, "Damping of inter-area oscillations in mixed AC/DC networks using WAMS based supplementary con-

troller," *IEEE Trans. Power Syst.*, vol. 28, no. 2, pp. 1160–1169, 2013.

[50] X. Sui, Y. Tang, H. He, and J. Wen, "Energy-storage-based low-frequency oscillation damping control using particle swarm optimization and heuristic dynamic programming," *IEEE Trans. Power Syst.*, vol. 29, no. 5, pp. 2539–2548, 2014.

[51] M. Mokhtari and F. Aminifar, "Toward wide-area oscillation control through doubly-fed induction generator wind farms," *IEEE Trans. Power Syst.*, vol. 29, no. 6, pp. 2985–2992, 2014.

[52] M. Ramirez, G. Calderon, and R. Castellanos, "Effect of PODCs for DFIG based wind farms in the inter-area and torsional oscillation damping," *IEEE Latin America Trans.*, vol. 14, no. 8, pp. 3648–3654, 2016.

[53] W. Yao, L. Jiang, J. Wen, Q. Wu, and S. Cheng, "Wide-area damping controller for power system interarea oscillations: A networked predictive control approach," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 1, pp. 27–36, 2014.

[54] Q. Mou, H. Ye, and Y. Liu, "Nonsmooth optimization-based wadc tuning in large delayed cyber-physical power system by interarea mode tracking and gradient sampling," *IEEE Trans. Power Syst.*, vol. 34, no. 1, pp. 668–679, 2019.

[55] X. Zhang, C. Lu, X. Xie, and Z. Y. Dong, "Stability analysis and controller design of a wide-area time-delay system based on the expectation model method," *IEEE Trans. Smart Grid*, vol. 7, no. 1, pp. 520–529, 2015.

[56] M. E. Bento, "Fixed low-order wide-area damping controller considering time delays and power system operation uncertainties," *IEEE Trans. Power Syst.*, vol. 35, no. 5, pp. 3918–3926, 2020.

[57] M. Beiraghi and A. Ranjbar, "Adaptive delay compensator for the robust wide-area damping controller design," *IEEE Trans. Power Syst.*, vol. 31, no. 6, pp. 4966–4976, 2016.

[58] B. Yang and Y. Sun, "Damping factor based delay margin for wide area signals in power system damping control," *IEEE Trans. Power Syst.*, vol. 28, no. 3, pp. 3501–3502, 2013.

[59] A. Yogarathinam and N. R. Chaudhuri, "Wide-area damping control using multiple

DFIG-based wind farms under stochastic data packet dropouts," *IEEE Trans. Smart Grid*, vol. 9, no. 4, pp. 3383–3393, 2016.

[60] T. Surinkaew and I. Ngamroo, "Hierarchical co-ordinated wide area and local controls of DFIG wind turbine and PSS for robust power oscillation damping," *IEEE Trans. Sustain. Energy*, vol. 7, no. 3, pp. 943–955, 2016.

[61] R. Xie, I. Kamwa, and C. Chung, "A novel wide-area control strategy for damping of critical frequency oscillations via modulation of active power injections," *IEEE Trans. Power Syst.*, 2020.

[62] S. Ghosh, M. S. El Moursi, E. F. El-Saadany, and K. Al Hosani, "Online coherency based adaptive wide area damping controller for transient stability enhancement," *IEEE Trans. Power Syst.*, vol. 35, no. 4, pp. 3100–3113, 2020.

[63] D. J. Sobajic and Y.-H. Pao, "Artificial neural-net based dynamic security assessment for electric power systems," *IEEE Trans. Power Syst.*, vol. 4, no. 1, pp. 220–228, 1989.

[64] Y.-H. Pao and D. J. Sobajic, "Combined use of unsupervised and supervised learning for dynamic security assessment," *IEEE Trans. Power Syst.*, vol. 7, no. 2, pp. 878–884, 1992.

[65] F. Hashiesh, H. E. Mostafa, A.-R. Khatib, I. Helal, and M. M. Mansour, "An intelligent wide area synchrophasor based system for predicting and mitigating transient instabilities," *IEEE Trans. Smart Grid*, vol. 3, no. 2, pp. 645–652, 2012.

[66] M. Sun, I. Konstantelos, and G. Strbac, "A deep learning-based feature extraction framework for system security assessment," *IEEE Trans. Smart Grid*, vol. 10, no. 5, pp. 5007–5020, 2018.

[67] Y. Mansour, A. Chang, J. Tamby, E. Vaahedi, B. Corns, and M. El-Sharkawi, "Large scale dynamic security screening and ranking using neural networks," *IEEE Trans. Power Syst.*, vol. 12, no. 2, pp. 954–960, 1997.

[68] H. Quan, D. Srinivasan, and A. Khosravi, "Short-term load and wind power forecasting using neural network-based prediction intervals," *IEEE Trans. Neural Networks & Learning Systems*, vol. 25, no. 2, pp. 303–315, 2013.

[69] S. Khaitan, "A survey of techniques for using neural networks in power systems," 2017.

[70] M. T. Haque and A. Kashtiban, "Application of neural networks in power systems; a review," *Power*, vol. 2005, 2000.

[71] O. Soares, H. Gonçalves, A. Martins, and A. Carvalho, "Nonlinear control of the doubly-fed induction generator in wind power systems," *Renewable Energy*, vol. 35, no. 8, pp. 1662–1670, 2010.

[72] T. K. Chau, S. S. Yu, T. Fernando, H. H.-C. Iu, and M. Small, "A load-forecasting-based adaptive parameter optimization strategy of STATCOM using ANNs for enhancement of LFOD in power systems," *IEEE Trans. Ind. Informatics*, vol. 14, no. 6, pp. 2463–2472, 2017.

[73] A. M. Yousef, "Neural network predictive control based power system stabilizer," *Res. J. Appl. Sci. Eng. Technol.*, vol. 4, no. 8, pp. 995–1003, 2012.

[74] O. Kahouli, M. Jebali, B. Alshammari, and H. H. Abdallah, "PSS design for damping low-frequency oscillations in a multi-machine power system with penetration of renewable power generations," *IET Renewable Power Gener.*, vol. 13, no. 1, pp. 116–127, 2018.

[75] S. Mehraeen, S. Jagannathan, and M. L. Crow, "Power system stabilization using adaptive neural network-based dynamic surface control," *IEEE Trans. Power Syst.*, vol. 26, no. 2, pp. 669–680, 2010.

[76] S. Kazemlou and S. Mehraeen, "Novel decentralized control of power systems with penetration of renewable energy sources in small-scale power systems," *IEEE Trans. Energy Conversion*, vol. 29, no. 4, pp. 851–861, 2014.

[77] H.-C. Tsai, J.-H. Liu, and C.-C. Chu, "Integrations of neural networks and transient energy functions for designing supplementary damping control of upfc," *IEEE Trans. Ind. Appl.*, vol. 55, no. 6, pp. 6438–6450, 2019.

[78] S. Gurung, S. Naetiladdanon, and A. Sangswang, "A surrogate based computationally efficient method to coordinate damping controllers for enhancement of probabilistic small-signal stability," *IEEE Access*, vol. 9, pp. 32 882–32 896, 2021.

[79] D. Ernst, M. Glavic, and L. Wehenkel, "Power systems stability control: reinforcement learning framework," *IEEE Trans. Power Syst.*, vol. 19, no. 1, pp. 427–435, 2004.

[80] M. Glavic, "Design of a resistive brake controller for power system stability enhancement using reinforcement learning," *IEEE Trans. Control Syst. Technol.*, vol. 13, no. 5, pp. 743–751, 2005.

[81] T. I. Ahamed, P. N. Rao, and P. Sastry, "A reinforcement learning approach to automatic generation control," *Elect. Power Syst. Res.*, vol. 63, no. 1, pp. 9–26, 2002.

[82] T. Yu, B. Zhou, K. W. Chan, L. Chen, and B. Yang, "Stochastic optimal relaxed automatic generation control in non-markov environment based on multi-step (λ) learning," *IEEE Trans. Power Syst.*, vol. 26, no. 3, pp. 1272–1282, 2011.

[83] E. Jasmin, T. I. Ahamed, and V. J. Raj, "Reinforcement learning approaches to economic dispatch problem," *Int. J. Electr. Power Energy Syst.*, vol. 33, no. 4, pp. 836–845, 2011.

[84] J. Duan, H. Xu, and W. Liu, "Q-learning-based damping control of wide-area power systems under cyber uncertainties," *IEEE Trans. Smart Grid*, vol. 9, no. 6, pp. 6408–6418, 2018.

[85] R. Hadidi and B. Jeyasurya, "Reinforcement learning based real-time wide-area stabilizing control agents to enhance power system stability," *IEEE Trans. Smart Grid*, vol. 4, no. 1, pp. 489–497, 2013.

[86] S. Mukherjee, A. Chakrabortty, H. Bai, A. Darvishi, and B. Fardanesh, "Scalable designs for reinforcement learning-based wide-area damping control," *IEEE Trans. Smart Grid*, 2021.

[87] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[88] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[89] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[90] Q. Huang, R. Huang, W. Hao, J. Tan, R. Fan, and Z. Huang, "Adaptive power system emergency control using deep reinforcement learning," *IEEE Trans. Smart Grid*, vol. 11, no. 2, pp. 1171–1182, 2019.

[91] J. Duan, D. Shi, R. Diao, H. Li, Z. Wang, B. Zhang, D. Bian, and Z. Yi, "Deep-reinforcement-learning-based autonomous voltage control for power grid operations," *IEEE Trans. Power Syst.*, vol. 35, no. 1, pp. 814–817, 2019.

[92] X. Wang, Y. Wang, D. Shi, J. Wang, and Z. Wang, "Two-stage wecc composite load modeling: A double deep Q-learning networks approach," *IEEE Trans. Smart Grid*, vol. 11, no. 5, pp. 4331–4344, 2020.

[93] Z. Yan and Y. Xu, "Data-driven load frequency control for stochastic power systems: A deep reinforcement learning method with continuous action search," *IEEE Trans. Power Syst.*, vol. 34, no. 2, pp. 1653–1656, 2018.

[94] M. Glavic, "(Deep) reinforcement learning for electric power system control and related problems: A short review and perspectives," *Annual Reviews in Control*, 2019.

[95] F. Li and Y. Du, "From AlphaGo to power system AI: What engineers can learn from solving the most complex board game," *IEEE Power and Energy Magazine*, vol. 16, no. 2, pp. 76–84, 2018.

[96] Y. Hashmy, Z. Yu, D. Shi, and Y. Weng, "Wide area measurement system-based low frequency oscillation damping control through reinforcement learning," *IEEE Trans. Smart Grid*, 2020.

[97] G. Zhang, W. Hu, D. Cao, Q. Huang, J. Yi, Z. Chen, and F. Blaabjerg, "Deep reinforcement learning based approach for proportional resonance power system stabilizer to prevent ultra-low-frequency oscillations," *IEEE Trans. Smart Grid*, 2020.

[98] GE Technical Staff, "PSLF user's manual," General Electric Int., Inc., 2019.

[99] DSA Tools Technical Staff, "SSAT model manual," PowerTech Labs Inc., 2017.

[100] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear matrix inequalities in system and control theory*. SIAM, 1994.

[101] J. G. VanAntwerp and R. D. Braatz, "A tutorial on linear and bilinear matrix inequalities," *J. Process Control*, vol. 10, no. 4, pp. 363–385, 2000.

[102] B. Erkus and Y. Lee, "Linear matrix inequalities and MATLAB LMI toolbox," in *University of Southern California Group Meeting Report, Los Angeles, California*, 2004.

[103] J. Camino, J. Helton, and R. Skelton, "Solving matrix inequalities whose unknowns are matrices," in *2004 43rd IEEE Conf. Decision and Control (CDC) (IEEE Cat. No.04CH37601)*, vol. 3, 2004, pp. 3160–3166.

[104] K. A. Vance, "Robust control for inter-area oscillations," Master's thesis, Virginia Tech, 2011.

[105] P. Gahinet, A. Nemirovski, A. J. Laub, and M. Chilali, "LMI control toolbox: For use with MATLAB (MathWorks)," 1995.

[106] K. Glover and J. C. Doyle, "State-space formulae for all stabilizing controllers that satisfy an H∞-norm bound and relations to relations to risk sensitivity," *Syst. & Control Lett.*, vol. 11, no. 3, pp. 167–172, 1988.

[107] J. Doyle, K. Glover, P. Khargonekar, and B. Francis, "State-space solutions to standard H2 and H∞ control problems," in *1988 American Control Conf.* IEEE, 1988, pp. 1691–1696.

[108] P. S. Rao and I. Sen, "Robust pole placement stabilizer design using linear matrix inequalities," *IEEE Trans. Power Syst.*, vol. 15, no. 1, pp. 313–319, 2000.

[109] M. Bernico, *Deep Learning Quick Reference: Useful Hacks for Training and Optimizing Deep Neural Networks with TensorFlow and Keras*. Packt Publishing Ltd, 2018.

[110] J. Heaton, "T81-558: Applications of deep neural networks:module 3 and module 4."

[111] D. Silver, "UCL course on reinforcement learning."

[112] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[113] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[114] J. McDowell, "Comparison of modern controls and reinforcement learning for robust control of autonomously backing up tractor-trailers to loading docks," Master's thesis, California Polytechnic State University, 2019.

[115] Y. Yue and H. Le, "Imitation learning tutorial," *Tutorial at ICML*, vol. 2018, 2018.

[116] J. Harrison and E. Brunskill, "CS234: Imitation learning," 2018.

[117] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *arXiv preprint arXiv:1811.06711*, 2018.

[118] J. C. Doyle, "Guaranteed margins for LQG regulators," *IEEE Trans. Autom. Control*, vol. 23, no. 4, pp. 756–757, 1978.

[119] J. Ma, T. Wang, X. Gao, S. Wang, and Z. Wang, "Classification and regression tree-based adaptive damping control of inter-area oscillations using wide-area signals," *IET Gener., Transmiss. & Distrib.*, vol. 8, no. 6, pp. 1177–1186, 2014.

[120] Y. Nesterov and A. Nemirovskii, *Interior-point polynomial algorithms in convex programming*. SIAM, 1994.

[121] I. J. Pérez-Arriaga, G. C. Verghese, and F. C. Schweppe, "Selective modal analysis with applications to electric power systems, part I: Heuristic introduction," *IEEE Trans. Power App. Syst.*, no. 9, pp. 3117–3125, 1982.

[122] A. Pal and J. S. Thorp, "Co-ordinated control of inter-area oscillations using SMA and LMI," in *2012 IEEE PES Innovative Smart Grid Technologies (ISGT)*, 2012, pp. 1–6.

[123] A. Beghi, "An application of selective modal analysis to tokamak modeling and control," *IEEE Trans. Control Syst. Technol.*, vol. 9, no. 4, pp. 574–589, 2001.

[124] P. Tan, M. Steinbach, and V. Kumar, "Introduction to data mining. 1st," 2005.

[125] L. Kaufman and P. J. Rousseeuw, *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons, 2009, vol. 344.

[126] K. D. Jones, A. Pal, and J. S. Thorp, "Methodology for performing synchrophasor data conditioning and validation," *IEEE Trans. Power Syst.*, vol. 30, no. 3, pp. 1121–1130, 2015.

[127] A. Pal and J. S. Thorp, "Co-ordinated control of inter-area oscillations using sma and LMI," Master's thesis, Virginia Tech, 2012.

[128] P. Gupta, A. Pal, C. Mishra, and T. Wang, "Design of a coordinated wide area damping controller by employing partial state feedback," in *Power And Energy Soc. General Meeting (Atlanta), 2019 IEEE PES*. IEEE, 2019, pp. 1–5.

[129] "Activation functions," Avaialble: https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html?highlight=softmax#relu.

[130] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Advances in Neural Inf. Process. Syst.*, 2012, pp. 2951–2959.

[131] S. Fujimoto, H. Van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," *arXiv preprint arXiv:1802.09477*, 2018.

[132] D. J. Trudnowski, J. Johnson, and J. F. Hauer, "Making prony analysis more accurate using multiple signals," *IEEE Trans. Power Syst.*, vol. 14, no. 1, pp. 226–231, 1999.

[133] D. C. Elizondo, "A methodology to assess and rank the effects of hidden failures in protection schemes based on regions of vulnerability and index of severity," Ph.D. dissertation, Virginia Tech, 2003.

[134] J. Chow, G. Rogers, and K. Cheung, "Power system toolbox," *Cherry Tree Scientific Software*, vol. 48, 2000.

[135] "2013 WECC path reports," WECC Staff Western Electricity Coordinating Council. Avaialble: https://www.wecc.biz/Reliability/TAS_PathReports_Combined_FINAL.pdf, 2013.

[136] A. Géron, "Neural networks and deep learning," 2018.

[137] X. Zhou, S. Wang, R. Diao, D. Bian, J. Duan, and D. Shi, "Rethink AI-based power grid control: Diving into algorithm design," *arXiv preprint arXiv:2012.13026*, 2020.

[138] L. Busoniu, R. Babuska, and B. De Schutter, "A comprehensive survey of multiagent reinforcement learning," *IEEE Trans. Syst., Man, & Cybern., Part C (Appl. & Rev.)*, vol. 38, no. 2, pp. 156–172, 2008.

APPENDIX A

LIST OF PUBLICATIONS

1. P. Gupta, A. Pal, C. Mishra, and T. Wang, "Design of a coordinated wide-area damping controller by employing partial state feedback," in *Power & Energy Soc. General Meeting (Atlanta)*, 2019 IEEE PES, pp. 1–5, IEEE, 2019.

2. P. Gupta, A. Pal, and V. Vittal, "Coordinated wide-area control of multiple controllers in a power system embedded with HVDC lines," *IEEE Trans. Power Syst.*, vol. 36, no. 1, pp. 648-658, 2020.

3. P. Gupta, A. Pal, and V. Vittal, "Coordinated wide-area damping control using deep neural networks and reinforcement learning," submitted to *IEEE Trans. Power Syst.*, 2020.

APPENDIX B

MATLAB CODE FOR CWADC

This code is used to create an interface between the SSAT and the LMI toolbox used in MATLAB. The code reads the system matrix corresponding to an OC (a.sma file) generated by SSAT tool after solving the powerflow, extract the states corresponding to different system components including the three controls (PSSs, DC-SDC and SVC), and then form A and B matrices with the help of rearranged states.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Program Name: InterfaceCreation: DSA Tools to MATLAB
% Description: Run this file to form A and B matrices for
%performing the SMA
% Author: Pooja Gupta %
% Arizona State University %
% Last Modified: 10/04/2019 %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc; clear all;
%% Record the information of all the machines from PSLF in the following order: machine number, id, total states of the machine, total states of the system excluing PSS states, mva rating and inertia
nInfoLMIN = [1 4 14 11 15.25 2.95;
            1 3 14 11 15.25 2.95;
            1 2 14 11 15.25 2.95;
            1 1 14 11  15.25 2.95;
            5 1 14 11 90.04 4.34;
            8 1 14 11 130.0  3.46;
            13 1 21 18 54.21 3.67;
            19 1 20 17 11.99 3.0;
            26 1 14 11 30.85 3.38;
            28 1 21 18 17.70 2.32;
            33 1 21 18 8.32  3.01;
            38 1 21 18 22.29 2.82;
            44 1 21 18 19.82 2.88;
            46 1 21 18 14.88 2.61;
            47 1 14 11 30.0  2.45;
            49 1 20 17 20.54 2.63;
            53 1 20 17 24.58 3.42;
            5699 1 20 17 9.14 2.64;
            60 1 20 17 31.17 3.83;
            62 1 20 17 35.4  3.59;
            65 1 20 17 21.04 6.07;
            67 2 20 17 9.09  3.49;
            67 1 20 17 9.09  3.49;
            72 1 14 9 30  2.75;
            76 1 20 17 90  2.82;
            81 2 20 17 12.50 3.03;
            81 1 20 17 12.50 3.03;
            91 1 20 17 68.40 3.82;
            96 1 20 17 10.6 4.39;
```

```
            102  1  20  17  8.95  4.16;
            113  1  12   9  2.70  4.13;
            119  1  20  17  1.13  3.8;
            126  1  20  17  2.5   6.41;];
nInfoLMIN(1,8)  =  nInfoLMIN(1,3);
nInfoLMIN(1,7)  = 1;
for i =2 :length(nInfoLMIN)
    nInfoLMIN(i,8) = nInfoLMIN(i,3) + nInfoLMIN(i-1,8); %% 8th column gives the total number of states till that generator
    nInfoLMIN(i,7) = nInfoLMIN(i-1,3) + nInfoLMIN(i-1,7); %% 7th column gives the starting index of each generator
end
%% specify number of machines and number of modes
nModeN = 656; nUnitN = length(nInfoLMIN); nSVC = 1; nWindUnit = 2; nSDC = 1; var = 0;
%% read the .sma file generated using SSAT after solving the power flow
fName = strcat('C:\pstv3\GriddingPoints\TrainingData\Case7VincDec\300_dec_656_9_17.sma');
fID = fopen(fName,'r');
% skip first 21 lines
for n=1:21
 strTemp=fgets(fID);
end
% read Asys from file
for n=1:nModeN
 for m=1:nModeN
 fTemp1=fscanf(fID,'%g',1);
 A1(m,n)=fTemp1;
 end
end
fclose(fID);
%% End of Read data from files
%% For rearrangement of A and B matrices
nn = size(A1)*[1 0]'; % nn = Size of System
for n = 1:nUnitN %4
   nStateN = nInfoLMIN(n,3);
   nStateEnd = nInfoLMIN(n,8);
   nStateStart = nInfoLMIN(n,7);
  % corresponds to deltas of machines
   r2(n,1)  = nStateStart+1;
   r2_Ref(n,1)  = nStateStart+1;
end
nInfoLMIN(5,11) = 1; nInfoLMIN(5,12) = 14;
var1 = nInfoLMIN(5,11); var2 = nInfoLMIN(5,12);
```

```
%% Excluding the machines that are not part of SMA
for n = 6:nUnitN
  if (nInfoLMIN(n,1) ~=1) & (nInfoLMIN(n,1) ~=19) & (
     nInfoLMIN(n,1) ~=33) & (nInfoLMIN(n,1) ~=96) & (
     nInfoLMIN(n,1) ~=102) & (nInfoLMIN(n,1) ~=113) & (
     nInfoLMIN(n,1) ~=119) & (nInfoLMIN(n,1) ~=126)
      nInfoLMIN(n,11) = var2 + 1;
       nInfoLMIN(n,12) = var2 + nInfoLMIN(n,3);
       var1 = nInfoLMIN(n,11);
       var2 = nInfoLMIN(n,12);
  end
end
%% transformation of A matrix by removing del of reference
    machine
Atrans= A1;
for i =1:length(r2)
    % if reference mc is included first,all machines after
       ref will not have  two non-zero entries for their del
    if r2(i) ~= 86
        % row modification   7th machine is the refernce
           machine
        Atrans(r2(i),:)= Atrans(r2(7),:) - Atrans(r2(i),:);
        % column modification
        Atrans(:,r2(i))= Atrans(:,r2(7)) - Atrans(:,r2(i));
    end
end
% row modification   3rd machine is the refernce machine
Atrans(r2(7),:)= Atrans(r2(7),:) - Atrans(r2(7),:);
% column modification
Atrans(:,r2(7))= Atrans(:,r2(7)) - Atrans(:,r2(7));
%3rd machine is the refernce machine
Atrans(r2(7),:) = [];
Atrans(:,r2(7)) = [];
r2=[]; r3 =[]; r4 =[]; r5 =[];r_link2_1311_1313_comp =[];
   rOmeg =[];rDel =[];c1=1;r_storPSS_ind_rPSS =[];
   r_storPSS_ind_rPSS_nz=[];
for n = 1:nUnitN
       nStateN = nInfoLMIN(n,3);
       nStateEnd = nInfoLMIN(n,8);
       nStateStart = nInfoLMIN(n,7);
       nStateStart_1 = nInfoLMIN(n,11);
       nStateEnd_1 = nInfoLMIN(n,12);
   if nInfoLMIN(n,1) ~=13 & var == 0
      % corresponds to omegas of machines
       r1  = nStateStart;
      % corresponds to deltas of machines
       r2  = nStateStart+1;
```

```
119         % corresponds to remaining states of generators and
                exciters
120         r    = nStateStart+2:nStateStart+8;
121
122         if nInfoLMIN(n,1) ~=72
123             % corresponds to PSS states
124             r_PSS     = nStateStart+9;
125             r_PSS_nz = nStateStart+10:nStateStart+11;
126             if (nInfoLMIN(n,1) ==1) || (nInfoLMIN(n,1) ==19) ||
                    (nInfoLMIN(n,1) ==33) || (nInfoLMIN(n,1) ==96)
                    || (nInfoLMIN(n,1) ==102) || (nInfoLMIN(n,1)
                    ==113) || (nInfoLMIN(n,1) ==119) || (nInfoLMIN(n
                    ,1) ==126)
127
128                 r_storPSS_ind_rPSS = [r_storPSS_ind_rPSS; r_PSS
                        '];
129                 r_storPSS_ind_rPSS_nz = [r_storPSS_ind_rPSS_nz
                        ; r_PSS_nz '];
130                 r_stor_Del =[r_stor_Del;r2 '];
131                 r_stor_Omg =[r_stor_Omg;r1 '];
132             end
133             % corresponds to governor states
134             r_gov     = nStateStart+12:nStateEnd;
135         else
136            % corresponds to PSS states of Miraloma(72) which
                   has 5 PSS states and no governor
137             r_PSS     = nStateStart+9-1:nStateStart+11-1;
138             % 113 also has no governor,but not included
                    specially since it has 3 PSS states like others
                    , which is covered in logic itself
139             r_PSS_nz = nStateStart+12-1:nStateStart+13-1;
140             % corresponds to PSS states of Miraloma(72) which
                    has 5 PSS states and no governor
141              if (nInfoLMIN(n,1) ==1) || (nInfoLMIN(n,1) ==19)
                    || (nInfoLMIN(n,1) ==33) || (nInfoLMIN(n,1)
                    ==96) || (nInfoLMIN(n,1) ==102) || (nInfoLMIN(
                    n,1) ==113) || (nInfoLMIN(n,1) ==119) || (
                    nInfoLMIN(n,1) ==126)
142                 r_storPSS_ind_rPSS = [r_storPSS_ind_rPSS;
                        r_PSS '];
143                 r_storPSS_ind_rPSS_nz = [r_storPSS_ind_rPSS_nz
                        ; r_PSS_nz '];
144                 r_stor_Del =[r_stor_Del;r2 '];
145                 r_stor_Omg =[r_stor_Omg;r1 '];
146             end
147             r_gov     = [];
148         end
```

```
149         rOmeg      = [rOmeg;r1'];
150         rDel       = [rDel; r2'];
151         % corresponds to rem states of generators, exciters
              and governors
152         r3         = [r3;r';r_gov'];
153         % corresponds to nz states of PSS
154         r4         = [r4; r_PSS_nz'];
155         % corresponds to zero states of PSS
156         r5         = [r5;r_PSS'];
157     else
158         var = 1;
159         if nInfoLMIN(n,1) ~=13
160            % corresponds to omegas of machines
161            r1  = nStateStart - 1;
162            r2  = nStateStart+1-1;
163         else
164             % corresponds to omegas of machines
165             r1  = nStateStart;
166         end
167         % corresponds to remaining states of generators and
              exciters
168         r   = nStateStart+2-1:nStateStart+8-1;
169         if nInfoLMIN(n,1) ~=72
170            % corresponds to PSS states
171            r_PSS     = nStateStart+9-1;
172            r_PSS_nz  = nStateStart+10-1:nStateStart+11-1;
173            if (nInfoLMIN(n,1) ==1) || (nInfoLMIN(n,1) ==19) ||
                  (nInfoLMIN(n,1) ==33) || (nInfoLMIN(n,1) ==96)
                 || (nInfoLMIN(n,1) ==102) || (nInfoLMIN(n,1)
                ==113) || (nInfoLMIN(n,1) ==119) || (nInfoLMIN(n
                ,1) ==126)
174              r_storPSS_ind_rPSS = [r_storPSS_ind_rPSS; r_PSS
                    '];
175              r_storPSS_ind_rPSS_nz = [r_storPSS_ind_rPSS_nz
                    ; r_PSS_nz'];
176              r_stor_Del =[r_stor_Del;r2'];
177              r_stor_Omg =[r_stor_Omg;r1'];
178            end
179            % corresponds to governor states
180             r_gov     = nStateStart+12-1:nStateEnd-1;
181         else
182             % corresponds to PSS states of Miraloma(72) which
                  has 5 PSS states and no governor
183             r_PSS     = nStateStart+10-1:nStateStart+12-1;
184             % 113 also has no governor,but not included
                  specially since it has 3 PSS states like others
                  , which is covered in logic itself
```

```
185             r_PSS_nz = nStateStart+13-1:nStateStart+14-1;
186             if (nInfoLMIN(n,1) ==1) || (nInfoLMIN(n,1) ==19) ||
                    (nInfoLMIN(n,1) ==33) || (nInfoLMIN(n,1) ==96)
                   || (nInfoLMIN(n,1) ==102) || (nInfoLMIN(n,1)
                   ==113) || (nInfoLMIN(n,1) ==119) || (nInfoLMIN(n
                   ,1) ==126)
187                 r_storPSS_ind_rPSS = [r_storPSS_ind_rPSS; r_PSS
                        '];
188                   r_storPSS_ind_rPSS_nz = [
                          r_storPSS_ind_rPSS_nz; r_PSS_nz'];
189                   r_stor_Del =[r_stor_Del;r2'];
190                   r_stor_Omg =[r_stor_Omg;r1'];
191               end
192               r_gov     = [];
193           end
194           rOmeg     = [rOmeg;r1'];
195           % corresponds to rem states of generators, exciters
                and governors
196           r3        = [r3;r';r_gov'];
197           % corresponds to nz states of PSS
198           r4        = [r4; r_PSS_nz'];
199           % corresponds to zero states of PSS
200           r5        = [r5;r_PSS'];
201       end
202       % addition of number of governor states to information
            matrix of generators
203       nInfoLMIN(n,9) = length([r_PSS';r_PSS_nz']);
204       % addition of number of governor states to information
            matrix of generators
205       nInfoLMIN(n,10) = length(r_gov);
206   end
207   %nModeN;  % corresponds to SVCs states
208   r_SVC = (nInfoLMIN(nUnitN,8))+1-1:(nInfoLMIN(nUnitN,8))
        -1+(3*nSVC);
209   %corresponds to wind turbines states
210   r_wind = r_SVC(1,end)+1:r_SVC(1,end)+(9*nWindUnit);
211   %% DC link 1 (PDCI-link1)
212   %  with 9 states for DC link 1 -with 656 states
213       r_link1_1312_1314  = r_wind(1,end)+1:r_wind(1,end)+9;
214   %% DC Link 2 with SDC installed on it -has more states than
        DC link 1 (PDCI-link 2)
215   % extract the states apart from the manin controller-SDC
        states
216   r_link2_1311_1313_nonSDC  = [r_link1_1312_1314(1,end)+1:
        r_link1_1312_1314(1,end)+4  r_link1_1312_1314(1,end)+6:
        r_link1_1312_1314(1,end)+8 r_link1_1312_1314(1,end)+10];
217   r_link2_1311_1313_nonSDC_rem  = [r_link1_1312_1314(1,end)
```

```
    +12:r_link1_1312_1314(1,end)+15 r_link1_1312_1314(1,end)
    +17];
% SDC states in DC link appears at positions 13, 14 and 15
    in A matrix, but their impact on rectifiers (position 5
    and 16),master control (position 11) and inverters (
    position 11) appears at different positions
 r_link2_1311_1313_SDC  = [r_link1_1312_1314(1,end)+5
    r_link1_1312_1314(1,end)+9 r_link1_1312_1314(1,end)+11
    r_link1_1312_1314(1,end)+16];
 r_link2_1311_1313_comp = [r_link2_1311_1313_nonSDC';
    r_link2_1311_1313_nonSDC_rem'; r_link2_1311_1313_SDC']
%% DC link 3 (Intermountain-Adelanto)
r_link3_INT_ADL  = r_link2_1311_1313_nonSDC_rem(1,end)+1:
    r_link2_1311_1313_nonSDC_rem(1,end)+5;
%% Rearranging the indices
% Arrangement done such that all other states apart for main
    controls come first followed by wind and DC links, then
    non-zero states of PSS and SVC, finally all controllers
[Lia,Locb] = ismember(r_storPSS_ind_rPSS,r5);
[Lia1,Locb1] = ismember(r_storPSS_ind_rPSS_nz,r4);
r5(Locb,:)=[];
r4(Locb1,:)=[];
[Lia2,Locb2] = ismember(r_stor_Del,rDel);
[Lia3,Locb3] = ismember(r_stor_Omg,rOmeg);
rDel(Locb2,:)=[];
rOmeg(Locb3,:)=[];
r =  [ rDel'  rOmeg' r_SVC(1,2):r_SVC(1,3) r5' r3' r_wind
    r_link1_1312_1314 r_link3_INT_ADL
    r_link2_1311_1313_nonSDC r_link2_1311_1313_nonSDC_rem  r4
    ' r_SVC(1,1) r_link2_1311_1313_SDC ];
%% Rearranging A matrix
A = zeros(length(r),length(r));
nn = size(A)*[1 0]'; % nn = Size of System
A(1:nn,1:nn) = Atrans(r,r);
%% specify number of controls
SDCstates=length(r_link2_1311_1313_SDC);
% Number of states per Control ; here 2, SDCstates, and 1
    corresponds to non-zero states per PSS (total 22), SDC
    and SVC respectively
nStatesControl = 1*nSVC + SDCstates +  (22*2) ;
noOfControl = nSVC + nSDC +  22;
n4 = nn - nStatesControl;
%%%% Begin to set B, C, D matrix
%% B matrix formed only using non-zero states of all
    controls
% number of non-zero states associated with each SVC is 2
    for svswsc PSLF model and SDC is 4
```

```
247 B_PSS_SVC_SDC = zeros(size(Atrans,1),(nSVC*1)+(nSDC*4)+
    (2*22));
248 d = 1;q=1;
249 % PSS states for B
250 for n = 1:22
251         % extracted from SSAT
252         B_PSS_SVC_SDC(r4(q),d) = 0.1;
253         B_PSS_SVC_SDC(r4(q+1),d+1) = 1/0.2;
254         d = d + 2;
255         q = q + 2;
256 end
257 % adding svc states apart from generators-adding at all non-
    zero positions
258 % Here assumed that all SVCs are placed one after other
259 r_PSSrem = [r_storPSS_ind_rPSS;r_storPSS_ind_rPSS_nz];
260 startIndSVC =nInfoLMIN(end,8) - length(r_PSSrem);
261 for n = 1:nSVC
262     % extracted from SSAT- svc states in B
263     B_PSS_SVC_SDC(r_SVC(1,1),d) =16.6666679;
264     d = d + 1;
265 end
266 % adding sdc states apart from generators-adding at all non-
    zero positions
267 % Here assumed that all SDCs are placed one after other
268 for n = 1:nSDC
269     if length(r_link2_1311_1313_SDC) == 4
270         %extracted from SSAT- sdc states in B
271         B_PSS_SVC_SDC(r_link2_1311_1313_SDC(1,1),d) =
              -0.0487427;
272         B_PSS_SVC_SDC(r_link2_1311_1313_SDC(1,2),d+1) =
              0.0487427;
273         B_PSS_SVC_SDC(r_link2_1311_1313_SDC(1,3),d+2) =
              -0.1799416;
274         B_PSS_SVC_SDC(r_link2_1311_1313_SDC(1,4),d+3) =
              0.0487427;
275      d = d + 1;
276 end
277 %% Rearranged B controls
278 for i =1:length(r)
279     BB(i,1:nStatesControl) = B_PSS_SVC_SDC(r(i),1:
        nStatesControl);
280 end
281 %% Transformation matrix to reduce the number of controls
282 % % each row corresponds to a single control;
283 % % each column refers to number of non-zero states in a
    control
284 gamma_RHS = zeros(1,nStatesControl);
```

```
%% PSS states
s=1;
gamma_pss = zeros(1,length(gamma_RHS));
for i =1:22
    gamma_pss(i,s) = 1;
    gamma_pss(i,s+1) = 1;
    s = s + 2;
end
%% SVC states
gamma_svc = zeros(1,length(gamma_RHS));
gamma_svc(1,end) = 1;
%% SDC states
gamma_sdc = zeros(1,length(gamma_RHS));
gamma_sdc(1,45:45+length(r_link2_1311_1313_SDC)-1) = 1;
% defines the exact number of rows in gamma
gamma = vertcat(gamma_pss,gamma_sdc,gamma_svc);
% Number of reduced controls
nnc = size(gamma)*[1 0]';
% % Transforming the A-matrix
An(1:n4,1:n4) = A(1:n4,1:n4);
An(n4+1:n4+nnc,n4+1:n4+nnc) = gamma*A(n4+1:n4+nStatesControl
    ,n4+1:n4+nStatesControl)*gamma';
AA = An;
Bnn = gamma*BB(n4+1:nn,1:nStatesControl)*gamma';
Bn = [ 0*ones(nnc,n4) Bnn ]';
BB = [ Bn Bn ];
% after removal of delta
nModeN =nModeN-1;
save A_C7_300_2  AA
save B_C7_300_2 BB
```

APPENDIX C

PYTHON CODE FOR DNN-CWADC

This code is used for building and training the deep neural network using the input polytopic data generated in LMI toolbox.

```
#########################################################
# Program Name: Training of DNN-CWADC
# Description: Run this file to train deep neural network
# Author: Pooja Gupta %
# Arizona State University %
# Last Modified: 03/04/2021 %
#########################################################
# Import of Python packages
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Activation,
    BatchNormalization,Dropout
from tensorflow.keras.callbacks import EarlyStopping,
    ReduceLROnPlateau
import pandas as pd
import numpy as np
import scipy.io as sio
from sklearn import metrics
from matplotlib import pyplot
from sklearn.metrics import mean_absolute_error
from tensorflow.keras import regularizers
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import layers
from numpy.random import seed
from numpy.random import shuffle
#Length of training neural network
num_epochs = 50
# Reading the input and the output training data generated
    after forming the polytopes in LMI toolbox
dfx_train = pd.read_csv('C:\pstv3\TrainingData\TrainingData\
    InputX_trainAngflowout.csv', header = None)
dfy_train = pd.read_csv('C:\pstv3\TrainingData\TrainingData\
    OutputY_trainAngflowout.csv', header = None)
#convert the read data to numpy
x_train = dfx_train.to_numpy()
y_train = dfy_train.to_numpy()
# Reading the input and output test data generated after
    forming the polytopes in LMI toolbox
dfx_test =  pd.read_csv('C:\pstv3\TrainingData\TrainingData\
    InputX_testAngFlowout.csv', header = None)
dfy_test = pd.read_csv('C:\pstv3\TrainingData\TrainingData\
    OutputY_testAngFlowout.csv', header = None)
#convert the read data to numpy
x_test = dfx_test.to_numpy()
y_test = dfy_test.to_numpy()
## Addition of Gaussian error to the training data
```

```
38 x_train3 = np.concatenate([x_train, n np.random.normal(
       x_train, 0.0002),np.random.normal(x_train, 0.0003),np.
       random.normal(x_train, 0.0004),np.random.normal(x_train,
       0.0005),np.random.normal(x_train, 0.0009),np.random.
       normal(x_train, 0.01)])
39 y_train3 = np.concatenate([y_train,y_train,y_train,y_train,
       y_train,y_train,y_train])
40 input_dim=x_train.shape[1]
41 # Build the neural network
42 model = Sequential()
43 # Neurons in input layer 5169
44 model.add(Dense(5169, input_dim=x_train.shape[1] ,activation
       ='relu'))
45 #model.add(Dropout(0.07))
46 BatchNormalization(axis=-1, momentum=0.99, epsilon=0.001,
       center=True, scale=True, beta_initializer='zeros',
       gamma_initializer='ones', moving_mean_initializer='zeros'
       , moving_variance_initializer='ones', beta_regularizer=
       None, gamma_regularizer=None, beta_constraint=None,
       gamma_constraint=None)
47 # Hidden 1 layer
48 model.add(Dense(2500, activation='relu',
       activity_regularizer=regularizers.l2(1e-5)))
49 BatchNormalization(axis=-1, momentum=0.99, epsilon=0.001,
       center=True, scale=True, beta_initializer='zeros',
       gamma_initializer='ones', moving_mean_initializer='zeros'
       , moving_variance_initializer='ones', beta_regularizer=
       None, gamma_regularizer=None, beta_constraint=None,
       gamma_constraint=None)
50 # Hidden 2 layer
51 model.add(Dense(1861, activation='relu',
       activity_regularizer=regularizers.l2(1e-5))) # Hidden 3
52 BatchNormalization(axis=-1, momentum=0.99, epsilon=0.001,
       center=True, scale=True, beta_initializer='zeros',
       gamma_initializer='ones', moving_mean_initializer='zeros'
       , moving_variance_initializer='ones', beta_regularizer=
       None, gamma_regularizer=None, beta_constraint=None,
       gamma_constraint=None)
53 # Output
54 model.add(Dense(1032, activation='linear'))
55 model.compile(loss='mean_absolute_error', optimizer=Adam(lr
       =1e-3), metrics=['mae'])
56 # monitor = EarlyStopping(monitor='val_loss', min_delta=1e
       -3, patience=5, verbose=1, mode='auto',
       restore_best_weights=True)
57 # ckpointer = ModelCheckpoint(filepath = 'model_zero7.{epoch
       :02d}-{val_loss:.6f}.hdf5',verbose=1,save_best_only=True,
```

```
    save_weights_only = True)
# Reduce learning rate if MAE doesn't decrease
reduce_lr = ReduceLROnPlateau(monitor='
    val_mean_absolute_error', factor=0.1, patience=2, min_lr
    =0.000001, verbose=1)
# Model fitting
history = model.fit(x_train3, y_train3, verbose=1, epochs=
    num_epochs, validation_split=0.2, batch_size=16,
    callbacks=[reduce_lr])
xc = range(1,num_epochs)
loss_train = history.history['loss']
l_t= np.array(loss_train[1:num_epochs])
loss_val = history.history['val_loss']
v_t= np.array(loss_val[1:num_epochs])
# Plotting model loss
pyplot.figure()
pyplot.plot(xc, l_t, label='train')
pyplot.plot(xc,  v_t, label='validation')
pyplot.title('Model loss')
pyplot.ylabel('Loss')
pyplot.xlabel('Epoch')
pyplot.legend(['Train', 'Validation'], loc='upper left')
# Plotting validation error
pyplot.figure()
mae_train = history.history['mean_absolute_error']
m_t= np.array(mae_train[1:num_epochs])
mae_val = history.history['val_mean_absolute_error']
m_v= np.array(mae_val[1:num_epochs])
pyplot.plot(xc,m_t)
pyplot.plot(xc,m_v)
pyplot.title('Model MAE')
pyplot.ylabel('MAE')
pyplot.xlabel('Epoch')
pyplot.legend(['Train', 'Test'], loc='upper left')
pyplot.show()
# Testing of the trained data
x_test1 = np.zeros(x_test.shape)
y_test1 = np.zeros(y_test.shape)
seed(1)
# prepare a sequence
sequence = [i for i in range(15)]
shuffle(sequence)
j = 0
while j < len(sequence):
    x_test1[j,:] = x_test[sequence[j],:]
    y_test1[j,:] = y_test[sequence[j],:]
    j += 1
```

```
100 # predict the gain (output of the the trained model) for the
        testing input
101 pred1 = model.predict(x_test1)
102 score1 = (metrics.mean_absolute_error(pred1,y_test1))
103 print(f"Fold score (MAE): {score1}")
104 # arrangement of the predicted data as accepted by the
       polytopic input -Section 6.1.1
105 K1 =np.zeros([24,43])
106 k=0
107 for j in range(43):
108     for i in range(24):
109         K1[i,j] = pred1[14, k]
110         k += 1
```

APPENDIX D

PYTHON CODE FOR DRL-CWADC

This code is used for creating the power system environment for training DRL-CWADC.

```
#########################################################
# Program Name: Power system environment for DRL-CWADC
# Description: Run this file to create environment for
# training DRL-CWADC
# Author: Pooja Gupta %
# Arizona State University %
# Last Modified: 03/04/2021 %
#########################################################
import logging, time
import math
import gym
from gym import spaces
from gym.utils import seeding
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
from subprocess import call, Popen, PIPE
import math as math
import csv
import random
import os
import shutil
from pathlib import Path
import pandas as pd
#class ActiveEnv(gym.Env):
class ActiveEnv():
    def __init__(self, seed =None):
        self.np_random = None
        self.seed = self.seed(seed)
        self._current_step = 0
        self.gen_data = self.load_gen_data()
        # initilaization of observation space
        self.observation_space = spaces.Box(low=obs[:,0],
            high =obs[:,1], dtype=np.float32)
        # initialization of action space-here only 4 actions
            added due to space constraints
        self.action_high =  np.array([3*np.abs(self.
            get_action()[0]), 1.0*np.abs(self.get_action()
            [1]), 1.0*np.abs(self.get_action()[2]), 1.0*np.
            abs(self.get_action()[3])])
        self.action_low = np.array(-self.action_high)
        self.action_space = spaces.Box(low =  self.
            action_low, high = self.action_high, dtype=np.
            float32)
        self._done = False
        self.reward = 0
```

```
40          self.log_terminalReached(self._done)
41      def seed (self, seed=None):
42          self.np_random, seed = seeding.np_random(seed)
43          return [seed]
44      # function for loading the generator data  - speeds and
           angles of the generators identified using SMA
45      def load_gen_data(self):
46           if os.path.exists('/upslf21/MyPSLF/RL_PSLF/AS1.csv'
                ):
47               try:
48                   ang = pd.read_csv('/upslf21/MyPSLF/RL_PSLF/
                        AS1.csv', delimiter=r"\s+")
49               except pd.errors.EmptyDataError:
50                   print('Note: AS1.csv was empty. Reading AS2
                        .csv.')
51                   ang = pd.read_csv('/upslf21/MyPSLF/RL_PSLF/
                        AS2.csv', delimiter=r"\s+")
52           else:
53              ang = pd.read_csv('/upslf21/MyPSLF/RL_PSLF/AS2.
                    csv', delimiter=r"\s+")
54           return ang.to_numpy()
55      # function for loading the polytopic gains- used as an
           initial input
56      def load_Gains_data(self):
57          #Gains =  pd.read_csv('Gain_Gens1.csv', header =
                None)
58          Gains =  pd.read_csv('Gain_Gens2.csv', header = None
                )
59
60          return Gains.to_numpy()
61      # function for calculating the change in speeds and
           angles of generators
62      def _get_obs(self):
63          state =[]
64          self.ang_speed = self.load_gen_data()
65          j = 0
66          while j < (len(self.ang_speed)):
67              dang = self.ang_speed[j,3]/180 * math.pi - self.
                    ang_speed[j,2]/180 * math.pi
68              dspd = self.ang_speed[j,5]-self.ang_speed[j,4]
69              ang_spd1.append(dang)
70              ang_spd1.append(dspd)
71              j += 1
72          state = (ang_spd1)
73          return np.array(state)
74      # function to find the change in maximum angle and speed
           for finding the maximum bound
```

```
75      def load_maxChange_gen_data(self):
76          max_angspd = pd.read_csv('C:/RLGC-master/RLGC-master
               /Fault_max_delChange.csv', delimiter=r"\s+")
77          return max_angspd.to_numpy()
78      # function to load the powerflows of lines
79      def load_RegPower_data(self):
80          if os.path.exists('C:/upslf21/MyPSLF/RL_PSLF/
               RegPowFlowsPSLF.csv'):
81              try:
82                  RegPower =  pd.read_csv('C:/upslf21/MyPSLF/
                       RL_PSLF/RegPowFlowsPSLF.csv', delimiter=r
                       "\s+", header = None)
83              except pd.errors.EmptyDataError:
84                  print('Note: Reg.csv was empty. Reading Reg3
                       .csv.')
85                  RegPower =  pd.read_csv('C:/upslf21/MyPSLF/
                       RL_PSLF/RegPowFlowsPSLF3.csv', delimiter=
                       r"\s+", header = None)
86          else:
87              RegPower =  pd.read_csv('C:/upslf21/MyPSLF/
                   RL_PSLF/RegPowFlowsPSLF3.csv', delimiter=r"\s
                   +", header = None)
88          return (RegPower[0].to_numpy()-RegPower[1].to_numpy
               ())
89      # reset function for speeds and angles at every timestep
90      def reset(self):
91          aSpd = []
92          j = 0
93          dang = 0.05
94          dspd = 0.025
95          while j < 22 :
96              aSpd.append(dang)
97              aSpd.append(dspd)
98              j += 1
99          high = np.array(aSpd)
100         state = self.np_random.uniform(low=-high, high=high)
101         return state
102     # function to generate the controller actions based on
           different PSLF timesteps
103     def get_action(self):
104         ang = []
105         spd = []
106         Gang = []
107         Gspd = []
108         result_ang = 0
109         result_spd = 0
110         result = 0
```

```
111         GangSVC = []
112         GspdSVC = []
113         result_angSVC = 0
114         result_spdSVC = 0
115         resultSVC = 0
116         GangPSS5 = []
117         GspdPSS5 = []
118         result_angPSS5 = 0
119         result_spdPSS5 = 0
120         resultPSS5 = 0
121         GangPSS8 = []
122         GspdPSS8 = []
123         result_angPSS8 = 0
124         result_spdPSS8 = 0
125         resultPSS8 = 0
126         Gain_Cont =self.load_Gains_data()
127         max_ang_spd =  self.load_maxChange_gen_data()
128         #extract the speeds and angles of generators
129         j = 0
130         while j < (len(max_ang_spd)):
131             if j != 2:
132                 dang = max_ang_spd[j,3]/180 * math.pi -
                        max_ang_spd[j,2]/180 * math.pi
133                 ang.append(dang)
134             j += 1
135         j = 0
136         while j < (len(max_ang_spd)):
137             dspd = max_ang_spd[j,5] - max_ang_spd[j,4]
138             spd.append(dspd)
139             j += 1
140         # Load Gains Data
141         # for DC-SDC
142         j = 2
143         while j < 23:
144             gainDel = Gain_Cont[22,j]
145             Gang.append(gainDel)
146             j += 1
147         j = 23
148         while j < Gain_Cont.shape[1]:
149             gainSpd = Gain_Cont[22,j]
150             Gspd.append(gainSpd)
151             j += 1
152         # for SVC
153         j = 2
154         while j < 23:
155             gainDelSVC = Gain_Cont[23,j]
156             GangSVC.append(gainDelSVC)
```

```
157             j += 1
158         j = 23
159         while j < Gain_Cont.shape[1]:
160             gainSpdSVC = Gain_Cont[23,j]
161             GspdSVC.append(gainSpdSVC)
162             j += 1
163         # for PSSs-shown here for only 2 PSSs
164         j = 2
165         while j < 23:
166             gainDelPSS5 = Gain_Cont[0,j]
167             GangPSS5.append(gainDelPSS5)
168             gainDelPSS8 = Gain_Cont[1,j]
169             GangPSS8.append(gainDelPSS8)
170             j += 1
171         j = 23
172         while j < Gain_Cont.shape[1]:
173             gainSpdPSS5 = Gain_Cont[0,j]
174             GspdPSS5.append(gainSpdPSS5)
175             gainSpdPSS8 = Gain_Cont[1,j]
176             GspdPSS8.append(gainSpdPSS8)
177             j += 1
178         angle   = np.array(ang)
179         speed   = np.array(spd)
180         Gangle   = np.array(Gang)
181         Gspeed   = np.array(Gspd)
182         GangleSVC   = np.array(GangSVC)
183         GspeedSVC   = np.array(GspdSVC)
184         GanglePSS5   = np.array(GangPSS5)
185         GspeedPSS5   = np.array(GspdPSS5)
186         GanglePSS8   = np.array(GangPSS8)
187         GspeedPSS8   = np.array(GspdPSS8)
188         for i in range(len(angle)):
189             result_ang += angle[i] * Gangle[i]
190             result_angSVC += angle[i] * GangleSVC[i]
191             result_angPSS5 += angle[i] * GanglePSS5[i]
192             result_angPSS8 += angle[i] * GanglePSS8[i]
193         for i in range(len(speed)):
194             result_spd += speed[i] * Gspeed[i]
195             result_spdSVC += speed[i] * GspeedSVC[i]
196             result_spdPSS5 += speed[i] * GspeedPSS5[i]
197             result_spdPSS8 += speed[i] * GspeedPSS8[i]
198         result = result_ang + result_spd
199         resultSVC = result_angSVC + result_spdSVC
200         resultPSS5 = result_angPSS5 + result_spdPSS5
201         resultPSS8 = result_angPSS8 + result_spdPSS8
202         return (result, resultSVC, resultPSS5, resultPSS8)
203     # logs actions for PSLF
```

```
204     def log_action(self, action):
205         action_pslf = action
206         myFile = open('/upslf21/MyPSLF/RL_PSLF/Ang_Python.
              csv','w', newline='')
207         with myFile:
208             writer = csv.writer(myFile)
209             writer.writerows(np.transpose(action_pslf))
210     # writes the information for PSLF if the episode is
          terminated
211     def log_terminalReached(self, done):
212         if (done == True):
213             tdone = 1
214         else:
215             tdone = 0
216         tdone1 =[[tdone],[tdone],[tdone],[tdone]]
217         myFile = open('/upslf21/MyPSLF/RL_PSLF/waitinfo.csv.
              csv','w', newline='')
218         with myFile:
219             writer = csv.writer(myFile)
220             writer.writerows(tdone1)
221     # function to calculate the reward
222     def calc_reward(self, action, obtstate):
223         #state_loss = 0
224         #assigned_reward = 0
225         ang_costs = 0
226         spd_costs = 0
227         cont_PSS = 0
228         cont_pen =np.zeros(22)
229         self.angSum = 0
230         action_state_num =  obtstate
231         for i in range(len(self.ang_speed)):
232             self.angSum += np.abs(self.ang_speed[i,3]/180 *
                  math.pi - self.ang_speed[i,2]/180 * math.pi)
233             ang_costs += 10 * np.abs(self.ang_speed[i,3]/180
                   * math.pi - self.ang_speed[i,2]/180 * math.
                  pi)
234             spd_costs += 10 * np.abs(self.ang_speed[i,5]-
                  self.ang_speed[i,4])
235
236         if (self.pslfTime <= 5):
237             if ((np.abs(action[0,0])) > 2*np.abs(self.
                  get_action()[0])) and ((np.abs(action[0,0]))
                  <= 3*np.abs(self.get_action()[0])) :
238                  cont1_pen = 3 * np.abs(action[0,0])
239             elif ((np.abs(action[0,0])) > 1*np.abs(self.
                  get_action()[0])) and ((np.abs(action[0,0]))
                  <= 2*np.abs(self.get_action()[0])) :
```

```
240                 cont1_pen = 2 * np.abs(action[0,0])
241             else:
242                 cont1_pen = 1 * np.abs(action[0,0])
243
244             if ((np.abs(action[0,1])) > (0.95)*np.abs(self.
                    get_action()[1])) and ((np.abs(action[0,1]))
                    <= 1.0*np.abs(self.get_action()[1])):
245                 cont2_pen = 1 * np.abs(action[0,1])
246
247             elif ((np.abs(action[0,1])) > (0.75)*np.abs(self
                    .get_action()[1])) and ((np.abs(action[0,1]))
                     <= (0.95)*np.abs(self.get_action()[1])):
248                 cont2_pen = 2 * np.abs(action[0,1])
249
250             else:
251                 cont2_pen = 3 * np.abs(action[0,1])
252
253             i = 0
254             for i in range(20):
255                 if ((np.abs(action[0,i+2])) > (0.95)*np.abs(
                        self.get_action()[i+2])) and ((np.abs(
                        action[0,i+2])) <= 1.0*np.abs(self.
                        get_action()[i+2])):
256                     cont_pen[i] = 4*np.abs(action[0,i+2])
257                 elif ((np.abs(action[0,i+2])) > (0.75)*np.
                        abs(self.get_action()[i+2])) and ((np.abs
                        (action[0,i+2])) <= (0.95)*np.abs(self.
                        get_action()[i+2])):
258                     cont_pen[i] = 3*np.abs(action[0,i+2])
259                 elif ((np.abs(action[0,i+2])) > (0.5)*np.abs
                        (self.get_action()[i+2])) and ((np.abs(
                        action[0,i+2])) <= (0.75)*np.abs(self.
                        get_action()[i+2])):
260                     cont_pen[i] = 2*np.abs(action[0,i+2])
261                 else:
262                     cont_pen[i] = 1*np.abs(action[0,i+2])
263
264         if (self.pslfTime > 5) and (self.pslfTime <= 8):
265             if ((np.abs(action[0,0])) > 2*np.abs(self.
                    get_action()[0])) and ((np.abs(action[0,0]))
                    <= 3*np.abs(self.get_action()[0])) :
266                 cont1_pen = 4 * np.abs(action[0,0])
267             elif ((np.abs(action[0,0])) > 1*np.abs(self.
                    get_action()[0])) and ((np.abs(action[0,0]))
                    <= 2*np.abs(self.get_action()[0])) :
268                 cont1_pen = 3 * np.abs(action[0,0])
269             elif ((np.abs(action[0,0])) > 0.75*np.abs(self.
```

```
                get_action()[0])) and ((np.abs(action[0,0]))
                <= 1*np.abs(self.get_action()[0])) :
270               cont1_pen = 2 * np.abs(action[0,0])
271           else:
272               cont1_pen = 1 * np.abs(action[0,0])
273
274           if ((np.abs(action[0,1])) > (0.95)*np.abs(self.
                get_action()[1])) and ((np.abs(action[0,1]))
                <= 1.0*np.abs(self.get_action()[1])):
275               cont2_pen = 4 * np.abs(action[0,1])
276           elif ((np.abs(action[0,1])) > (0.75)*np.abs(self
                .get_action()[1])) and ((np.abs(action[0,1]))
                 <= (0.95)*np.abs(self.get_action()[1])):
277               cont2_pen = 3 * np.abs(action[0,1])
278           elif ((np.abs(action[0,1])) > (0.5)*np.abs(self.
                get_action()[1])) and ((np.abs(action[0,1]))
                <= (0.75)*np.abs(self.get_action()[1])):
279               cont2_pen = 2 * np.abs(action[0,1])
280           else:
281               cont2_pen = 1 * np.abs(action[0,1])
282
283           i = 0
284           for i in range(20):
285               if ((np.abs(action[0,i+2])) > (0.95)*np.abs(
                      self.get_action()[i+2])) and ((np.abs(
                      action[0,i+2])) <= 1.0*np.abs(self.
                      get_action()[i+2])):
286                   cont_pen[i] = 4*np.abs(action[0,i+2])
287               elif ((np.abs(action[0,i+2])) > (0.75)*np.
                      abs(self.get_action()[i+2])) and ((np.abs
                      (action[0,i+2])) <= (0.95)*np.abs(self.
                      get_action()[i+2])):
288                   cont_pen[i] = 3*np.abs(action[0,i+2])
289               elif ((np.abs(action[0,i+2])) > (0.5)*np.abs
                      (self.get_action()[i+2])) and ((np.abs(
                      action[0,i+2])) <= (0.75)*np.abs(self.
                      get_action()[i+2])):
290                   cont_pen[i] = 2*np.abs(action[0,i+2])
291               else:
292                   cont_pen[i] = 1*np.abs(action[0,i+2])
293
294       if (self.pslfTime > 8):
295           if ((np.abs(action[0,0])) > 2*np.abs(self.
                get_action()[0])) and ((np.abs(action[0,0]))
                <= 3*np.abs(self.get_action()[0])) :
296               cont1_pen = 4 * np.abs(action[0,0])
297           elif ((np.abs(action[0,0])) > 1*np.abs(self.
```

```
            get_action()[0])) and ((np.abs(action[0,0]))
            <= 2*np.abs(self.get_action()[0])) :
298             cont1_pen = 3 * np.abs(action[0,0])
299         elif ((np.abs(action[0,0])) > 0.75*np.abs(self.
            get_action()[0])) and ((np.abs(action[0,0]))
            <= 1*np.abs(self.get_action()[0])):
300             cont1_pen = 2 * np.abs(action[0,0])
301         else:
302             cont1_pen = 1 * np.abs(action[0,0])
303
304         if ((np.abs(action[0,1])) > (0.95)*np.abs(self.
            get_action()[1])) and ((np.abs(action[0,1]))
            <= 1.0*np.abs(self.get_action()[1])):
305             cont2_pen = 4 * np.abs(action[0,1])
306         elif (np.abs(action[0,1])) > (0.75)*np.abs(self.
            get_action()[1]) and ((np.abs(action[0,1]))
            <= (0.95)*np.abs(self.get_action()[1])):
307             cont2_pen = 3 * np.abs(action[0,1])
308         elif ((np.abs(action[0,1])) > (0.5)*np.abs(self.
            get_action()[1])) and ((np.abs(action[0,1]))
            <= (0.75)*np.abs(self.get_action()[1])):
309             cont2_pen = 2 * np.abs(action[0,1])
310         else:
311             cont2_pen = 1 * np.abs(action[0,1])
312
313         i = 0
314         for i in range(20):
315             if ((np.abs(action[0,i+2])) > (0.95)*np.abs(
                    self.get_action()[i+2])) and ((np.abs(
                    action[0,i+2])) <= 1.0*np.abs(self.
                    get_action()[i+2])):
316                 cont_pen[i] = 4*np.abs(action[0,i+2])
317             elif ((np.abs(action[0,i+2])) > (0.75)*np.
                    abs(self.get_action()[i+2])) and ((np.abs
                    (action[0,i+2])) <= (0.95)*np.abs(self.
                    get_action()[i+2])):
318                 cont_pen[i] = 3*np.abs(action[0,i+2])
319             elif ((np.abs(action[0,i+2])) > (0.5)*np.abs
                    (self.get_action()[i+2])) and ((np.abs(
                    action[0,i+2])) <= (0.75)*np.abs(self.
                    get_action()[i+2])):
320                 cont_pen[i] = 2*np.abs(action[0,i+2])
321             else:
322                 cont_pen[i] = 1*np.abs(action[0,i+2])
323
324     for i in range(22):
325         cont_PSS += cont_pen[i]
```

```
326             costs_tot = ang_costs + spd_costs + cont1_pen +
                    cont2_pen + cont_PSS
327             print("angC, spdC, actC0, actC0", ang_costs,
                    spd_costs, cont1_pen, cont2_pen)
328             return -costs_tot
329         # main step function for the DRL algorithm
330         def step(self, action, time_count):
331             log_action = self.log_action(action)
332             nstate =  self._get_obs()
333             # calculation of the rewards corresponding to the
                    generated actions
334             reward = self.calc_reward (action, nstate)
335             print("angSum", self.angSum)
336             # terminate the episode when PSLF timestep reaches
                    35 sec
337             if (self.pslfTime > 35):
338                 self._done = True
339                 self.log_terminalReached(self._done)
340             return nstate, reward, self._done, {}
```

APPENDIX E

Training of DRL-CWADC

This is the main code used to train DRL-CWADC.

```
#########################################################
# Program Name: Training for DRL-CWADC
# Description: Run this file to train DRL-CWADC
# Author: Pooja Gupta %
# Arizona State University %
# Last Modified: 03/04/2021 %
#########################################################
from ddpg_tf_orig import Agent
import gym
import logging, time
import numpy as np
#from utils import plotLearning
from matplotlib import pyplot
from ActiveEnv_orig import ActiveEnv
env = ActiveEnv()
# Parameters to build deep neural networks for DRL
agent = Agent(alpha=0.0001, beta = 0.001, input_dims=[44],
    tau=0.001, env=env,
                  batch_size=32, layer1_size=400,
                     layer2_size=400, n_actions=24, var =
                     0.25)
score_history = []
ep_rewards = []
ep_avgrewards = []
ep_minrewards = []
ep_maxrewards = []
AGGREGATE_STATS_EVERY = 5  # episodes
counter = 0
np.random.seed(0)
for i in range(500):
    time.sleep(36)
    if i <= 2:
        agent.load_models()
    done = False
    score = 0
    obs = env.reset()
    # step_count represents time_step of PSLF
    for step_count in range(300):
        # agent chooses action based on ddpg algorithm
        act = agent.choose_action(obs, step_count, env)
        # agent takes an action which in turn returns new
            sttaes and rewards
        new_state, reward, done, info = env.step(act,
            step_count)
        print("act, reward",  act, reward)
        # store transition states in replay buffer
```

```
42          agent.remember(obs, act, reward, new_state, int(done
                ))
43          if i > 2:
44              agent.learn()
45          score += reward
46          obs = new_state
47          print('episode ', i, 'score %.2f' % score,
48                  '100 game average %.2f' % np.mean(
                        score_history[-100:]))
49          if i > 1:
50              if i%10 == 0:
51                  agent.save_models()
52          print("pslfTime",env.pslfTime)
53          # terminate the episode when PSLF timestep reaches
                35 sec
54          if (env.pslfTime > 35):
55              i += 1
56              print("i", i)
57              break
58      ep_rewards.append(score)
59      print("score", score)
60      if not i % AGGREGATE_STATS_EVERY:
61          average_reward = sum(ep_rewards[-
                AGGREGATE_STATS_EVERY:])/len(ep_rewards[-
                AGGREGATE_STATS_EVERY:])
62          min_reward = min(ep_rewards[-AGGREGATE_STATS_EVERY
                :])
63          max_reward = max(ep_rewards[-AGGREGATE_STATS_EVERY
                :])
64
65          ep_avgrewards.append(average_reward)
66          ep_minrewards.append(min_reward)
67          ep_maxrewards.append(max_reward)
68  x = [i+1 for i in range(500)]
69  pyplot.plot(x, ep_rewards)
```