Online optimization and learning for the optimal power flow problem with unknown objectives

Anbang.Chai

Supervisors:
Dr.ir. Sergio Grammatico
Ir. Emilio Benenati

TU Delft

Online optimization and learning for the optimal power flow problem with unknown objectives

by

Anbang.Chai

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday March 19, 2024 at 09:30 AM.

Student number:	5375037	
Project duration:	February 15, 2023 – March 19, 2024	
Thesis committee:	Prof. dr. ir. S. Grammatico ,	TU Delft, supervisor
	Prof. dr. ir. P. M. Esfahani,	TU Delft
	Ir. E. Benenati,	TU Delft, supervisor

TU Delft

Preface

I would like to take this opportunity to extend my heartfelt gratitude to all those who supported me throughout the development and completion of this thesis. Their unwavering support, insightful guidance, and continuous encouragement have been pillars of strength and inspiration on this challenging journey.

The inception of this thesis was driven by a compelling motivation to discover and employ a more effective method for operating electricity network systems. This need becomes particularly acute in systems comprising numerous buses, where existing computational resources fall short of performing calculations in a timely manner. Moreover, the challenge is amplified when potential cost functions remain undetermined, especially under criteria that are notoriously difficult to model.

This thesis is a confluence of three pivotal components: the online Primal-Dual-Gradient-Projection (PDGP) OPF control algorithm, which leverages approximate linear power flow models; a voltage measurement feedback strategy that enhances operational accuracy and bypasses the need for power demand data; and the Shape-constrained Gaussian Process, a novel approach for estimating unknown costs with greater precision. Together, these elements embody a comprehensive attempt to address and mitigate the complexities inherent in managing extensive electricity networks.

In closing, I am profoundly grateful for the collective wisdom, patience, and support bestowed upon me by my mentors, peers, and family. Their contributions have not only shaped this academic endeavor but have also facilitated a profound personal and professional growth. It is my sincere hope that this thesis contributes meaningfully to the field of system and control engineering and paves the way for further research in optimizing electricity network systems for the betterment of our communities and beyond.

Thank you once again to everyone who has been a part of this journey. Your belief in my capabilities and your unwavering support have been the wind beneath my wings, propelling me forward to the completion of this significant milestone.

With deep appreciation,

A.Chai
Delft, March 2024

Abstract

The Optimal Power Flow (OPF) problem, a cornerstone of power system operations, has gained increased attention since its inception by Carpentier in 1962. OPF is fundamentally an optimization challenge aimed at enhancing electric power system operations within the bounds of physical and operational constraints. Over the decades, various methodologies have been explored to address the OPF problem, adapting to evolving grid complexities and the integration of distributed energy resources. These advancements have brought to the fore issues related to system randomness, fluctuation, and the need for rapid control mechanisms.

This thesis introduces a comprehensive solution incorporating an online optimization algorithm tailored for real-time OPF applications. This approach, characterized by minimal computation times, integrates a feedback strategy that obviates the necessity for instantaneous power demand information and employs a Shape-constrained Gaussian Process for the estimation of unknown cost functions. The proposed control algorithm demonstrates robust tracking performance and satisfactory computation efficiency, marking a significant improvement towards optimizing future power networks fraught with increasing size and complexity.

Moreover, this work delves into the investigation of various system design parameters, offering insights into potential avenues for enhancing system performance. Through a meticulous examination of these parameters, the thesis sheds light on strategies to refine the integrated system's efficacy, paving the way for more resilient and efficient power networks.

Contents

1 Introduction

The Optimal Power Flow (OPF) problem has been defined variously across different papers and textbooks. However, irrespective of the terminology or descriptions employed, the core challenge inherent in OPF remains consistent: it encompasses any optimization problems aimed at optimizing the operation of an electric power system which is subject to physical constraints dictated by electrical laws and engineering limitations [11]. In other words, any power systems optimization problem that incorporates a set of power flow equations among its constraints can be deemed a form of OPF [12]. The OPF problem is typically a time-variant, constrained, nonlinear, and non-convex optimization problem. It involves a network comprising buses, some of which face time-varying demand loads while others are integrated with distributed power resources. Although different studies may prioritize varying objectives, the overarching goals of OPF typically include minimizing power losses and maximizing economic gains.

1.1. The OPF problem formulation

For the most simplicity, the majority of OPF formulations may be represented using the following standard form [12]:

$$\begin{aligned} \min \quad & f(\boldsymbol{u}, \boldsymbol{x}) \\ \text{s.t.} \quad & g(\boldsymbol{u}, \boldsymbol{x}) = 0 \\ & h(\boldsymbol{u}, \boldsymbol{x}) \leqslant 0 \end{aligned} \tag{1.1}$$

Where $\boldsymbol{u}$ and $\boldsymbol{x}$ denote control variables (Power injections $P_{g,i}$, $Q_{g,i}$, .etc) and dependent system states (uncontrollable voltages V_i, .etc) respectively.

A more specific representation for OPF could be written as follows:

$$\min_{\{u_i\}_{i\in\mathcal{G}}} \quad f(\boldsymbol{u}) \tag{1.2a}$$

$$\text{s.t.} \quad P_i(\boldsymbol{v}, \boldsymbol{\theta}) = P_{g,i} - P_{d,i}, \quad \forall n \in \mathcal{N} \tag{1.2b}$$

$$Q_i(\boldsymbol{v}, \boldsymbol{\theta}) = Q_{g,i} - Q_{d,i}, \quad \forall n \in \mathcal{N} \tag{1.2c}$$

$$V^{min} \leq |\tilde{V}_i| \leq V^{max}, \quad \forall n \in \mathcal{N} \tag{1.2d}$$

$$u_i \in \mathcal{Y}_i, \quad \forall i \in \mathcal{G} \tag{1.2e}$$

Where $\mathcal{N}$ and $\mathcal{G}$ denote all the buses and the buses with control devices (generator, battery, .etc) in the network.

Equation 1.2b and 1.2c are the power flow equations for bus i, while $P_{g,i}$ and $P_{d,i}$ in them are the real power generations and the real power demand respectively at bus i, the same definitions for the reactive power $Q_{g,i}$ and $Q_{d,i}$. Equation 1.2d represents the voltage limits V^{min} and V^{max}, and equation 1.2e denotes the limit $\mathcal{U}_i$ for control variables.

The objective functions employed in addressing the Optimal Power Flow (OPF) problem, as referenced in equation 1.2, exhibit variability due to differing system configurations. These variations ensure that network operations are optimized in alignment with explicitly defined criteria, such as minimizing power losses [7]. Typically, across the majority of scenarios, the objectives hinge on the real and reactive power generation within the system [12].

1.1.1. System settings and objective functions

The general form of the objective function of the OPF problem 1.2 can be written as[7]:

$$f_o(\boldsymbol{v}, \boldsymbol{i}, \boldsymbol{p_g}, \boldsymbol{q_g}) := c_\rho \rho(\boldsymbol{v}, \boldsymbol{i}) + c_\phi \phi\left(\boldsymbol{p_g}, \boldsymbol{q_g}\right) + c_v v(\boldsymbol{v}) \tag{1.3}$$

According to different system settings, the selection of the 3 terms in equation 1.3 could be different. $\rho(\boldsymbol{v}, \boldsymbol{i})$ captures real power losses in the network, it is described in [7] as:

$$\rho(\boldsymbol{v}, \boldsymbol{i}) := \sum_{(m,n)\in\mathcal{E}} \Re\left\{\tilde{V}_m \tilde{I}^*_{mn}\right\} - \Re\left\{\tilde{V}_n \tilde{I}^*_{mn}\right\}. \tag{1.4}$$

$\tilde{I}_{mn} \in \mathbb{C}$ denotes the current flowing on line (m, n), $\mathcal{E}$ consists all the notes (m, n) in the network.

$\phi\left(\boldsymbol{p_g}, \boldsymbol{q_g}\right)$ captures the terms related to the generated real power and imaginary power. In [7], [25] and [6], it is described as the following form:

$$\phi\left(\boldsymbol{p_g}, \boldsymbol{q_g}\right) = \sum_{i\in\mathcal{G}} c_q \left(Q_{g_i}\right)^2 + c_p \left(P^k_{\text{av},i} - P_{g_i}\right)^2 \tag{1.5}$$

Where c_p and c_q are design coefficients, $P^k_{\text{av},i}$ is the total available power generated. The effort of this term is to minimize the amount of real power curtailed and the amount of reactive power injected or absorbed. This is typically used when there is high penetration of photovoltaic (PV) generators, particularly, it is demonstrated in [6] how the proposed controllers can reliably prevent over-voltages that are likely to be experienced during periods when PV generation exceeds the demand [6],[7], while trying to maintain the real power generated by photovoltaic generators as large as possible.

In this case, the controller variables limit at time instant $t = k$ are as follows [6], [25]:

$$\begin{aligned} &0 \leqslant P^k_{g,i} \leqslant P^k_{av,i} \\ &\left|Q^k_{g,i}\right| \leqslant \sqrt{P^{k2}_{av,i} - P^{k2}_{g,i}} \end{aligned} \tag{1.6}$$

The constraints 1.6 describe the situation that a part of the available real power $P^k_{av,i}$ is curtailed so that some available reactive power can be provided in order to enforce voltage regulation, the available reactive power is upper bounded by $\sqrt{P^{k2}_{av,i} - P^{k2}_{g,i}}$.

Alternatively, another typical formation of $\phi\left(\boldsymbol{p_g}, \boldsymbol{q_g}\right)$ in [11], [1], [5], [31]:

$$\phi\left(\boldsymbol{p_g}, \boldsymbol{q_g}\right) = \sum_{i\in\mathcal{G}} (c_{i,p,2} {P_{g,i}}^2 + c_{i,p,1} P_{g,i} + c_{i,p,0}) \tag{1.7}$$

In this formulation, the coefficients $c_{i,p,2}$, $c_{i,p,1}$ and $c_{i,p,0}$ serve as weighting factors. This setup exemplifies scenarios aiming at minimizing the generation fuel cost expressed by a quadratic function [5], [1]. In other words, controllable power injections, denoted by $P_{g,i}$ and $Q_{g,i}$, are sourced from energy storage systems or generators. This approach ensures the fulfillment of all load demands, with the primary aim of ϕ being to minimize the total power injection.

Finally, the last term in equation 1.7 $v(\boldsymbol{v})$ promotes a flat voltage profile[10],[5], [1]:

$$v(\boldsymbol{v}) = \sum_{i\in\mathcal{N}} c_{v,i} \left(\left|\tilde{V}_i\right| - V_{\text{ref}}\right) \tag{1.8}$$

which is optimized by minimizing the load bus voltage deviation (VD) from the pre-defined voltage reference V_{ref}. $c_{v,i}$ denotes weighting coefficients.

The cost terms in this part are some most commonly used objective functions. However, some other terms such as voltages stability enhancement [5], [1], [16] or piece-wise quadratic cost function [1], [5], could also be introduced into the objective function.

1.1.2. Constraints and variables

In the majority of scenarios, the equality constraints for the OPF problem, as delineated in problem 1.2, describe the flow equations that depict the power system's dynamics. Meanwhile, the inequality constraints generally encompass limitations on generators and voltage, as specified in problem 1.2.

Many researchers in the field often categorize the system variables into two groups: control variables $\boldsymbol{u}$, which usually include the power injections from generators ($P_{g,i}$, $Q_{g,i}$), and state variables—sometimes referred to as dependent variables $\boldsymbol{x}$—encompassing voltage magnitudes and angles. Within this framework, the control variables, or independent variables, are described as follows [11]:

$$\boldsymbol{u} = \left[P_{g,i}, Q_{g,i}\right], i \in \mathcal{G} \tag{1.9}$$

while the state variables (dependent variables) can be written as [11]:

$$\boldsymbol{x} = [\tilde{v}_i] \quad i \in \mathcal{N} \tag{1.10}$$

Alternatively, as in [5], [26], the OPF problem 1.2 can be written as:

$$\min_{\boldsymbol{u}} \quad f'^{k}(\boldsymbol{u}, \boldsymbol{x}) \tag{1.11a}$$

$$\text{s.t.} \quad \boldsymbol{u}_i \in \mathcal{U}_i^k, \forall i \in \mathcal{G} \tag{1.11b}$$

Where the new objective function 1.11a: f'^{k} is combined with the original objective function and a quadratic penalty term. The specific formulation of penalty terms can vary based on the system configuration. Following the approach detailed in [5], a penalty term—constituted by the penalty factor λ_v and the squared deviation of the dependent variable from its limit—is appended to the objective function to penalize unfeasible solutions:

$$f_0' = f_0 + \lambda_V \sum_{i=1}^{N} \left(\tilde{v}_i - V_i^{\text{lim}}\right)^2 \tag{1.12}$$

In this optimization framework, control variables are inherently 'self-constrained,' as delineated in equation 1.11b. Additionally, the inequality constraints associated with dependent variables, such as load bus voltage magnitudes, are seamlessly integrated into the objective function through the incorporation of quadratic penalty terms, as outlined in equation 1.12.

In equation 1.12, the voltage $\tilde{v}_i$ is derived from system models, and λ_v denotes the penalty factor. The limit value V_i^{lim} for the dependent variable $\tilde{v}_i$ is defined as follows, ensuring adherence to specified voltage bounds [5]:

$$V_i^{\text{lim}} = \begin{cases} V_i^{\max} & \text{if} \quad \tilde{v}_i > V_i^{\max} \\ V_i^{\min} & \text{if} \quad \tilde{v}_i < V_i^{\min} \end{cases} \tag{1.13}$$

Alternatively, functional constraints can be integrated into the objective function via a logarithmic barrier function, as exemplified in [21]. This method facilitates the handling of constraints within the optimization process, promoting solution feasibility without explicitly enforcing boundary conditions.

1.2. Motivation and main challenges

Based on the particular selection of variables, objective(s), and constraints, OPF formulations differ greatly [12]. Such variable nature leads to different solutions and accuracy. This section describes traditional OPF approaches investigated and their limitations in the past few decades.

Traditionally, classical optimization techniques have predominantly relied on gradient-based optimization algorithms. These methods typically linearize the objective function and system constraints around a specific operating point [1], although in certain instances, a direct application of nonlinear system models is observed. Distinct classifications and methodologies, such as Linear Programming (LP) and Sequential Linear Programming (SLP), Quadratic Programming (QP) and Sequential Quadratic Programming (SQP), along with Gradient Methods and Newton's Method, have been extensively utilized over the past few decades. However, the limitations of these conventional OPF approaches are well-recognized.

Challenge 1) Computation speed: It is well-known that the OPF problem is non-convex and NP-hard [6], which results in significant computational demands for its resolution. A primary challenge is the alignment of the solution time with the rapidly evolving system dynamics [6]. Moreover, as the complexity of the network increases, the time required to solve the OPF task may grow exponentially [11].

In such scenarios, a critical drawback of traditional OPF methods is that one has to wait for iterative convergence before obtaining a usable solution for the network [26] [14]. In the near future, with more integration of renewable power sources, such as distributed wind and solar generation into the electricity grid, the network will face an increasing amount of unpredictable fluctuations[26] [14] [18]. This, along with the growing size and complexity of power networks demanding enhanced control capabilities, necessitates controllers that operate on much faster timescales—seconds [26] [6] or even sub-seconds [30]. Traditional OPF methods, functioning on slower timescales, will fall short in addressing these requirements [26]. Against this backdrop, the concept of real-time optimal power flow (RT-OPF), also known as online OPF or sometimes dynamic OPF, has emerged as a compelling research area in recent years.

challenge 2) Solution accuracy: Even when employing methods known for their faster convergence rates, such as Linear Programming (LP) and Sequential Linear Programming (SLP), the controller's performance may still fall short of expectations. A significant limitation of the solution computed via a a simplified reformulation of the original nonlinear OPF problem is that such solutions might be sub-optimal or even not feasible for the original formulation and might not represent a global optimum or even a feasible solution within the original Nonlinear (NL) OPF problem, especially in complex systems [12].

This limitation is partly addressed by the Sequential Linear Programming (SLP) approach. Unlike relying on a singular linearization, SLP facilitates the optimization of problems with nonlinear attributes through successive linear approximations [12]. Following the resolution of a linear optimization problem, the original NL problem is re-linearized at the new optimal point, and this cycle is repeated until convergence is achieved. The principal benefit of SLP lies in its ability to maintain the expedience of LP while edging closer to the accuracy offered by Nonlinear Programming (NLP) methods. However, unlike LP, SLP constructs the linear program based on the current operating point, thus inherently converging to local optima only [12].

Challenge 3) Uncontrollable demand measurement The requirement for load demand data at each bus for every time instant presents a significant hurdle, which may not be realistic in actual power network operations [6], [19]. The absence of this detailed information hampers the practical implementation of the algorithm.

Challenge 4) Unknown cost functions The cost function $f(\boldsymbol{u})$ in the OPF problem, as referenced in problem 1.2, may sometimes be unknown, particularly in scenarios where the cost functions aim to encapsulate user preferences, comfort, or environmental impacts such as pollution, which are inherently difficult to model [19], [20].

Given the challenges and limitations discussed earlier, the primary objectives of this thesis are to de-

velop an algorithm that provides sub-optimal solutions to the OPF problem while achieving:

- Rapid computation times (within seconds or sub-seconds during sampling)
- Minimal asymptotic errors compared to the actual optimal solutions
- Feasibility without direct observations of the load demands, $\boldsymbol{p_d^k}$, $\boldsymbol{q_d^k}$
- Applicability with unknown cost functions

1.3. Relevant work

To develop controllers capable of managing operations at significantly faster rates, online OPF methods—also referred to as 'real-time' OPF methods—have emerged as an attractive topic of recent research. Despite the diversity in online OPF approaches, their foundational concepts can generally be categorized into previously discussed categories. For instance, [30] presents an online OPF method employing a modified ADMM algorithm originally proposed by [28], along with a linear approximation of the network model from [9]. Similarly, utilizing the linear approximation methodology from [9] and [15], [6] outlines a real-time OPF method based on gradient projection, demonstrating notable tracking performance. In scenarios involving nonlinear network models, [26] evaluates a real-time OPF method utilizing quasi-Newton iterations, illustrating that, with a suitable algorithm, OPF approaches incorporating nonlinear models can also achieve rapid computation times. Moreover, [3], [6], and [17] incorporate a feedback strategy into the online OPF methodology, thereby obviating the need for direct load information and compensating for errors due to model mismatches. This feedback approach shows to enhance convergence performance in numerical simulations.

Furthermore, the exploration of various predictors has been undertaken to enhance the solutions to the OPF problem. Within this context, a predictor is defined as a model designed to forecast the future progression of the sequence of optimization solutions based on current data. [25] explores multiple prediction methods within time-varying optimization frameworks. Specifically, [24] introduces a first-order optimal predictor that leverages first-order derivative information. Meanwhile, [23] expands upon this by presenting a predictor that circumvents the need for computing the inversion of the Hessian. Numerical simulations pertinent to these methodologies indicate that incorporating prediction into the algorithm results in improved tracking accuracy.

1.4. Thesis outline

The primary contribution of this thesis is the development of a controller that integrates an online/real-time Optimal Power Flow (OPF) algorithm with a linearized system model, measurement feedback strategy, and Gaussian process to learn unknown cost functions, along with investigation of the effects of the design parameters and potential improvements.

The thesis is organized as follows: Chapter 1 provides a fundamental introduction to the OPF problem, including the motivation and challenges, as well as relevant work. Chapter 2 describes the mathematical model and its linearization for a power network, alongside test results of linear error using a real-world network that will be utilized in this thesis. Chapter 3 details the online algorithm Primal-Dual Gradient Projection method and the feedback strategy employed. Chapter 4 discusses the shape-constrained Gaussian process designed to learn the unknown cost functions. Chapter 5 presents the simulation results and discusses the implications of the corresponding system design parameters. Chapter 6 concludes the thesis and outlines potential avenues for future work.

2

System model and linearization

This chapter introduces the bus injection model for power systems, which forms the foundation of the analysis conducted in this thesis. We explore the linear approximation of this model as proposed in [9]. Subsequently, we undertake a numerical verification of its accuracy.

In conventional power flow equations, all system buses are assigned to one of three bus types:

- Slack Bus: The voltage magnitude V and angle θ are fixed and the power injections (P for real power, Q for imaginary power) are free. Typically $V = 1.0$ p.u. and $\theta = 0°$. There is only one slack bus in a power system.
- Load Bus: or 'PQ' bus, the power injections P and Q are fixed while the voltage magnitude V and angle θ are free. The symbol $\mathcal{M}$ denotes the indices of PQ buses in a network.
- Voltage-Controlled bus: or 'PV' bus, the real power injection P and voltage magnitude V are fixed while the reactive power injection Q and the voltage angle θ are free. According to the symbols defined before, there are $|\mathcal{N}| - |\mathcal{M}| - 1$ PV buses in the system.

We denote the Complex power as:

$$\tilde{\boldsymbol{s}} = [\tilde{s}_1, \tilde{s}_2, ..., \tilde{s}_n]^T \triangleq \tilde{\boldsymbol{v}} \cdot (\mathbf{Y}\tilde{\boldsymbol{v}})^* \tag{2.1}$$

Where $\mathbf{Y}$ denotes the admittance matrix with typical elements $Y_{i,j}$.
The equation can be also written as:

$$\tilde{s}_i = \tilde{s}_{g,i} - \tilde{s}_{d,i} \quad \Leftrightarrow \quad P_i + jQ_i = \left(P_{g,i} - P_{d,i}\right) + j\left(Q_{g,i} - Q_{d,i}\right). \tag{2.2}$$

Where the superscripts $*_g and *_d$ indicate the generated power and the demand power respectively.

The basic two equations of the conventional power flow are:

$$\begin{aligned} P_i(V, \theta) &= P_{g,i} - P_{d,i} \quad \forall i \in \mathcal{N} \\ Q_i(V, \theta) &= Q_{g,i} - Q_{d,i} \quad \forall i \in \mathcal{N}, \end{aligned} \tag{2.3}$$

Where P_i is the real power injection at bus i, while Q_i is the imaginary power injection at bus i.

Combined with equation 2.1, 2.2 and 2.3: the following form can be derived [26]:

$$P_i(t) + \mathrm{j}Q_i(t) = \sum_{j \in \mathcal{N}} \tilde{V}_i(t)\tilde{V}_j^*(t)Y_{ij}^*$$

$$I_{ij}(t) = -Y_{ij}\left(\tilde{V}_i(t) - \tilde{V}_j(t)\right)$$

2.1. Bus injection model

AC power flow equations can be represented through two distinct models: the Bus Injection Model and the Branch Flow Model [11]. Both models have been extensively investigated by various researchers. The Bus Injection Model, being the most commonly utilized of the two, will be the focus of this report. Specifically, the Bus Injection Model is delineated in [11] in two different forms, distinguished by the coordinate frame adopted for voltage and admittance:

1) Selection of polar coordinates for voltage, $\tilde{V}_i = V_i \angle \theta_i$ and rectangular coordinates for admittance $\mathbf{Y} = \mathbf{G} + j\mathbf{B}$,:

$$\begin{aligned} P_i(V,\theta) =& V_i \sum_{k=1}^{N} V_k \left(G_{ik}\cos(\theta_i - \theta_k)\right. \\ &\left. + B_{ik}\sin(\theta_i - \theta_k)\right) \quad \forall i \in \mathcal{N} \\ Q_i(V,\theta) =& V_i \sum_{k=1}^{N} V_k \left(G_{ik}\sin(\theta_i - \theta_k)\right. \\ &\left. - B_{ik}\cos(\theta_i - \theta_k)\right) \quad \forall i \in \mathcal{N} \end{aligned} \tag{2.4}$$

2) Selection of polar coordinates for voltage $\widetilde{v}_i = V_i \angle \theta_i$, and polar coordinates for admittance,$\widetilde{Y}_{ik} = Y_{ik} \angle \delta_{ik}$:

$$\begin{aligned} P_i(V,\theta) &= V_i \sum_{k=1}^{N} V_k Y_{ik} \cos(\theta_i - \theta_k - \delta_{ik}) \quad \forall i \in \mathcal{N}, \\ Q_i(V,\theta) &= V_i \sum_{k=1}^{N} V_k Y_{ik} \sin(\theta_i - \theta_k - \delta_{ik}) \quad \forall i \in \mathcal{N}. \end{aligned} \tag{2.5}$$

2.2. The admittance matrix

The admittance matrix $\mathbf{Y}$ used in the power system models (equation 2.1, 2.4, 2.5) derives from the application of Ohm's and Kirchoff's laws to a steady-state AC electrical network [11]. Defining y_i^{S} as the shunt admittance associated with the bus itself, the elements of $\mathbf{Y}$ can be computed [11]:

$$\begin{aligned} Y_{ii} &= y_i^{\mathrm{S}} + \sum_{k:(i,k)\in\mathcal{L}} \frac{1}{|a_{ik}|^2}\left(\tilde{y}_{ik} + \frac{1}{2}\tilde{y}_{ik}^{\mathrm{Sh}}\right) + \sum_{k:(k,i)\in\mathcal{L}} \left(\tilde{y}_{ki} + \frac{1}{2}y_{ki}^{\mathrm{Sh}}\right) \\ Y_{ik} &= -\sum_{k:(i,k)\in\mathcal{L}} \frac{1}{a_{ik}^*} y_{ik} - \sum_{k:(k,i)\in\mathcal{L}} \frac{1}{a_{ik}} y_{ki}, \quad i \neq k \end{aligned} \tag{2.6}$$

Where y_{ik} can be computed as:

$$y_{ik} = \frac{1}{R_{ik} + jX_{ik}} = \frac{R_{ik}}{R_{ik}^2 + X_{ik}^2} - j\frac{X_{ik}}{R_{ik}^2 + X_{ik}^2}. \tag{2.7}$$

a_{ik} can be computed as:

$$a_{ik} = T_{ik} e^{j\varphi_{ik}} \tag{2.8}$$

Series impedance $R_{ik} + jX_{ik}$ in equation 2.7 and shunt susceptance $\tilde{y}_{ik}^{\mathrm{Sh}}$ in equation 2.6, voltage ratio T_{ik} and phase angle φ_{ik} in equation 2.8 can be obtained by the system data.

Another form of computing $\mathbf{Y}$ is to separate equation 2.6 into real and imaginary parts using the definition $\mathbf{Y} = \mathbf{G} + j\mathbf{B}$ and the identity $a_{ik} = T_{ik}(\cos\varphi_{ik} + j\sin\varphi_{ik})$:

$$\begin{aligned}
G_{ii} =& g_i^{\mathrm{S}} + \sum_{k:(i,k)\in\mathbf{L}} \frac{1}{T_{ik}^2}\left(g_{ik} + \frac{1}{2}g_{ik}^{\mathrm{Sh}}\right) + \sum_{k:(k,i)\in\mathcal{L}}\left(g_{ki} + \frac{1}{2}g_{ki}^{\mathrm{Sh}}\right),\\
G_{ik} =& -\sum_{k:(i,k)\in\mathcal{L}} \frac{1}{T_{ik}}\left(g_{ik}\cos\varphi_{ik} - b_{ik}\sin\varphi_{ik}\right)\\
&-\sum_{k:(k,i)\in\mathcal{L}} \frac{1}{T_{ki}}\left(g_{ki}\cos\varphi_{ki} + b_{ki}\sin\varphi_{ki}\right), \quad i \neq k\\
B_{ii} =& b_i^{\mathrm{S}} + \sum_{k:(i,k)\in\mathbf{L}} \frac{1}{T_{ik}^2}\left(b_{ik} + \frac{1}{2}b_{ik}^{\mathrm{Sh}}\right) + \sum_{k:(k,i)\in\mathcal{L}}\left(b_{ki} + \frac{1}{2}b_{ki}^{\mathrm{Sh}}\right),\\
B_{ik} =& -\sum_{k:(i,k)\in\mathcal{L}} \frac{1}{T_{ik}}\left(g_{ik}\sin\varphi_{ik} + b_{ik}\cos\varphi_{ik}\right)\\
&-\sum_{k\cdot(k,i)\in\mathbf{I}} \frac{1}{T_{ki}}\left(-g_{ki}\sin\varphi_{ki} + b_{ki}\cos\varphi_{ki}\right), \quad i \neq k.
\end{aligned} \tag{2.9}$$

Where g_{ik} and b_{ik} are the real and imaginary parts of y_{ik} computed by equation 2.7 respectively.

In the simulations of this thesis, the command **makeYbus** in the MATLAB Matpower toolbox is directly used to obtain the admittance matrix $\mathbf{Y}$ by providing a network as the input.

2.3. Linear approximation

In order to develop an online OPF algorithm, one approach involves linearizing the system model, which significantly reduces computation time. More details about the specific online OPF algorithm to be utilized will be discussed in Chapter 3. This section initially describes the linear model proposed in [9], [6].

2.3.1. Linear approximate model

Assume that the temporal domain is discretized as $t = k\tau$, where $k \in \mathbb{N}$ and $\tau > 0$ is a given interval. Assigning $\mathbf{Y}^k$ as the system admittance matrix $(N \times N)$, which can be partitioned into submatrices with the following dimensions: $\overline{\mathbf{Y}}^k \in \mathbb{C}^{(N-1)\times(N-1)}$, $\overline{\mathbf{y}}^k \in \mathbb{C}^{(N-1)}$, $y_{00}^k \in \mathbb{C}$, finally, $\tilde{v_0}^k$ and $\tilde{i_0}^k$ are the slack bus voltage and current, while $\boldsymbol{v}^k \in \mathbb{C}^N$ and $\boldsymbol{i}^k \in \mathbb{C}^N$ are the voltages and currents for the rest of the buses. Using Ohm's and Kirchhoff's circuit laws, the circuit relations can be compactly represented in matrix-vector form as follows ([6]):

$$\begin{bmatrix} \tilde{I_0}^k \\ \boldsymbol{i}^k \end{bmatrix} = \underbrace{\begin{bmatrix} y_{00}^k & \left(\overline{\mathbf{y}}^k\right)^{\mathsf{T}} \\ \overline{\mathbf{y}}^k & \overline{\mathbf{Y}}^k \end{bmatrix}}_{:=\mathbf{Y}^k} \begin{bmatrix} \tilde{V_0}^k \\ \boldsymbol{v}^k \end{bmatrix}, \tag{2.10}$$

The standard form of the OPF problem in [6] can be written as:

$$\begin{aligned}
\left(\mathrm{OPF}^k\right) \min_{\boldsymbol{v},\boldsymbol{i},\left\{P_{g,i},Q_{g,i}\right\}_{i\in\mathcal{G}}} \quad & h^k(\boldsymbol{v}) + \sum_{i\in\mathcal{G}} f_i^k\left(P_{g,i},Q_{g,i}\right)\\
\text{s.t.} \quad & \tilde{V}_i\tilde{I}_i^* = P_{g,i}^k - P_{d,i}^k + \mathrm{j}\left(Q_{g,i}^k - Q_{d,i}^k\right), \quad \forall i \in \mathcal{G}\\
& \tilde{V}_n\tilde{I}_n^* = -P_{d,n}^k - \mathrm{j}Q_{d,n}^k, \quad \forall n \in \mathcal{N}\backslash\mathcal{G}\\
& V^{\min} \le \left|\tilde{V}_i\right| \le V^{\max}, \quad \forall i \in \mathcal{N}\\
& \left(P_{g,i},Q_{g,i}\right) \in \mathcal{U}_i^k, \quad \forall i \in \mathcal{G},
\end{aligned} \tag{2.11}$$

Where $\mathcal{G}$ collects all the buses with generators, $f_i^k\left(P_{g,i}, Q_{g,i}\right)$ is a time-varying function specifying performance objectives for the i th generator, while $h^k(\boldsymbol{v})$captures system-level objectives. The details of the objection function will be discussed later.

We now discuss the Linearization of the model which defines the constraints of problem 2.11:

Define for each bus, the voltage :

$$\tilde{V}_i = \tilde{V}_{e,i} + \Delta\tilde{V}_i \tag{2.12}$$

where $\tilde{V}_{e,i}$ and $\Delta\tilde{V}_i$ are the nominal/equilibrium point and the small deviation, respectively. The desired linear model can be written as:

$$\begin{aligned} \boldsymbol{\Delta v_{re}} &\approx \mathbf{A_{re}}\boldsymbol{p} + \mathbf{B_{re}}\boldsymbol{q} \quad i \in \mathcal{N} \\ \boldsymbol{\Delta v_{im}} &\approx \mathbf{A_{im}}\boldsymbol{p} + \mathbf{B_{im}}\boldsymbol{q} \quad i \in \mathcal{N} \end{aligned} \tag{2.13}$$

Where$\boldsymbol{\Delta v_{re}}$and $\boldsymbol{\Delta v_{im}}$ are the real and imaginary parts of the deviation voltages respectively, $\boldsymbol{p}$, $\boldsymbol{q}$ are the actual real power and imaginary power injections at the i th bus ($\boldsymbol{p} = \boldsymbol{p_g} - \boldsymbol{p_d}$, $\boldsymbol{q} = \boldsymbol{q_g} - \boldsymbol{q_d}$). $\mathbf{A_{re}}$, $\mathbf{A_{im}}$, $\mathbf{B_{re}}$ and $\mathbf{B_{im}}$ are the linear model matrix that we want.

Assigning the power generated by generators and the power consumed by the load demands as $P_{g,i}$, $Q_{g,i}$ and $P_{d,i}$, $Q_{d,i}$ respectively, the net power injection at bus i can be computed as:

$$\begin{aligned} P_i &= P_{g,i} - P_{d,i} \\ Q_i &= Q_{g,i} - Q_{d,i} \\ &\text{for } i \in \mathcal{N} \end{aligned} \tag{2.14}$$

If there is no generator at bus i, then $P_{g,i} = Q_{g,i} = 0$

Note that for computation and derivation convenience, here the form in [9] which defines the relations between control variables $P_{g,i}$, $Q_{g,i}$ and the real part, imaginary part of voltages V_{re}, V_{im} are used instead of the form in [6] where the system outputs are the magnitudes and phase of voltages.

As for the selection of the linearization points, [9] and [15] both provide 2 methods.

1. Flat Voltage: Mainly used in transmission networks, the nominal voltages $\boldsymbol{v_e}$ are set to be:

$$\boldsymbol{v_e} = 1_N \tag{2.15}$$

2. No-load Voltage: Mainly used in distribution networks, the nominal voltages are set to be:

$$\boldsymbol{v_e} = -\overline{\mathbf{Y}}^{-1}\overline{\mathbf{y}}V_0 \tag{2.16}$$

The second selection is used in this thesis since distribution networks will be investigated and tested.

Using equation 2.10, complex-power bus injections can be compactly written as ([9]):

$$\boldsymbol{s} = \mathrm{diag}(\boldsymbol{v})\boldsymbol{i}^* = \mathrm{diag}(\boldsymbol{v})\left(\overline{\mathbf{Y}}^*\boldsymbol{v}^* + \overline{\mathbf{y}}^* V_0^*\right) \tag{2.17}$$

Then replacing $\boldsymbol{v}$ with $\boldsymbol{v_e} + \boldsymbol{\Delta v}$ as equation 2.12 defines, and discarding the second order terms in $\boldsymbol{\Delta v}$:

$$\mathbf{A}'\boldsymbol{\Delta v} + \mathbf{B}'\boldsymbol{\Delta v}^* = \boldsymbol{s} + \boldsymbol{v} \tag{2.18}$$

Where

$$\mathbf{A}' := \mathrm{diag}\left(\overline{\mathbf{Y}}^*\boldsymbol{v_e}^* + \overline{\mathbf{y}}^*\tilde{V_0}^*\right) \tag{2.19}$$

$$\mathbf{B}' := \mathrm{diag}(\boldsymbol{v_e})\tilde{\mathbf{Y}}^* \tag{2.20}$$

$$\boldsymbol{v} := -\mathrm{diag}(\boldsymbol{v_e})\left(\overline{\mathbf{Y}}^*\boldsymbol{v_e}^* + \overline{\mathbf{y}}^*\tilde{V_0}^*\right).$$

Choosing the linearization points as equation 2.16 obtains the linearized power-flow expression [6]:

$$\operatorname{diag}(\boldsymbol{v_e}^*)\,\overline{\mathbf{Y}}\boldsymbol{\Delta v} = \boldsymbol{s}^* \tag{2.21}$$

Which can be rewritten as:

$$\boldsymbol{\Delta v} = \overline{\mathbf{Y}}^{-1}\operatorname{diag}^{-1}(\boldsymbol{v_e}^*)\,\boldsymbol{s}^*$$

The system matrix for 2.13 then can be obtained as:

$$\begin{aligned}
\mathbf{A}_{re} &= \mathbf{R}\operatorname{diag}\left(\frac{\cos\boldsymbol{\theta_e}}{|\boldsymbol{v_e}|}\right) - \mathbf{X}\operatorname{diag}\left(\frac{\sin\boldsymbol{\theta_e}}{|\boldsymbol{v_e}|}\right)\\
\mathbf{B}_{re} &= \mathbf{X}\operatorname{diag}\left(\frac{\cos\boldsymbol{\theta_e}}{|\boldsymbol{v_e}|}\right) + \mathbf{R}\operatorname{diag}\left(\frac{\sin\boldsymbol{\theta_e}}{|\boldsymbol{v_e}|}\right)\\
\mathbf{A}_{im} &= \mathbf{X}\operatorname{diag}\left(\frac{\cos\boldsymbol{\theta_e}}{|\boldsymbol{v_e}|}\right) + \mathbf{R}\operatorname{diag}\left(\frac{\sin\boldsymbol{\theta_e}}{|\boldsymbol{v_e}|}\right)\\
\mathbf{B}_{im} &= -\mathbf{R}\operatorname{diag}\left(\frac{\cos\boldsymbol{\theta_e}}{|\boldsymbol{v_e}|}\right) + \mathbf{X}\operatorname{ding}\left(\frac{\sin\boldsymbol{\theta_e}}{|\boldsymbol{v_e}|}\right)
\end{aligned} \tag{2.22}$$

$\mathbf{R}$ and $\mathbf{X}$ are the resistance and reactance matrix respectively, for all $i \in \mathcal{N}$, defined as:

$$\begin{aligned}
\mathbf{R} &= \operatorname{Re}\left(\overline{\mathbf{Y}}^{-1}\right)\\
\mathbf{X} &= \operatorname{Im}\left(\overline{\mathbf{Y}}^{-1}\right)
\end{aligned} \tag{2.23}$$

$\boldsymbol{\theta_e}$ and $\boldsymbol{v_e}$ are correspond to the chosen linearization points (From [6] and [9]):

$$\boldsymbol{v_e} = -\overline{\mathbf{Y}}^{-1}\overline{\mathbf{y}}\tilde{V}_0 \tag{2.24}$$

The final voltage is:

$$\boldsymbol{v} = \boldsymbol{v_e} + \boldsymbol{\Delta v} \tag{2.25}$$

Assigning $\mathcal{G} \subseteq \mathcal{N}$ to be the indices of the buses with generators. Then the linear surrogate with linear constraints for the target OPF problem is:

$$\min_{\{\boldsymbol{u}_i\}_{i\in\mathcal{G}}} \quad f_o^k(\boldsymbol{u}_i) \tag{2.26a}$$

$$\text{s.t.} \quad g_{1,n}^k\left(\{\boldsymbol{u}_i\}_{i\in\mathcal{G}}\right) \le 0, \forall n \in \mathcal{N} \tag{2.26b}$$

$$g_{2,n}^k\left(\{\boldsymbol{u}_i\}_{i\in\mathcal{G}}\right) \le 0, \forall n \in \mathcal{N} \tag{2.26c}$$

$$\boldsymbol{u}_i \in \mathcal{U}_i^k, \forall i \in \mathcal{G}, \tag{2.26d}$$

Where $\boldsymbol{u}_i := \left[P_{g,i}, Q_{g,i}\right]^{\mathsf{T}}$ are the control variables.

Equation 2.26a is the objective function, which has different forms in different system criteria; the details were discussed in Chapter 1.

The constraints of the optimization problem now become :

$$\begin{aligned}
g_{1,n}^k\left(\{\boldsymbol{u_i}\}_{i\in G}\right) &= V_{\min} - \left|\tilde{V}_n^{\,k}\right|\\
g_{2,n}^k\left(\{\boldsymbol{u_i}\}_{i\in G}\right) &= -V_{\max} + \left|\tilde{V}_n^{\,k}\right|
\end{aligned} \tag{2.27}$$

Where $|\tilde{V}_n^{\,k}|$ is the voltage magnitude of bus n at $t = k$ time instant:

$$\left|\tilde{V}_n^k\right| = \left| \sum_{i\in\mathcal{N}} \left\{ \mathbf{A}_{re}(i,n) \cdot \left(P_{g,i}^k - P_{d,i}^k\right) + \mathbf{B}_{re}(i,n) \cdot \left(Q_{g,i}^k - Q_{d,i}^k\right) \right\} + j \cdot \left\{ \sum_{i\in\mathcal{N}} \left\{ \mathbf{A}_{im}(i,n) \cdot \left(P_{g,i}^k - P_{d,i}^k\right) + \mathbf{B}_{im}(i,n) \cdot \left(Q_{g,i}^k - Q_{d,i}^k\right) \right\} \right\} \right| \tag{2.28}$$

Note that $V^{\max}$ and$V^{\min}$ here are N×1 vectors collecting limits applied on the voltages at each bus. $P_{g,i}$ and $P_{d,i}$ means the real power generation and the real power demand for bus i, the same definition for the imaginary power $Q_{g,i}$ and $Q_{d,i}$.

2.3.2. Linear model error

To meticulously assess the impact of errors introduced by the linearized model, as delineated in equations 2.13 and 2.22, this thesis undertakes an empirical evaluation within the framework of the IEEE-85 electricity network [8]. This particular network encompasses 85 buses, with 8 PQ power sources strategically located at buses 2, 3, 4, 5, 6, 7, 84, and 85. The choice of the IEEE-85 network as a testing ground allows for a comprehensive analysis of linear error manifestations across a complex and representative electrical system.

Combine equation 2.12 and equation 2.13, we can get the complete form of the linear model:

$$\begin{aligned} \boldsymbol{v}_{re} &= \mathbf{A_{re}}\boldsymbol{p} + \mathbf{B_{re}}\boldsymbol{q} + \bar{\boldsymbol{v}}_{e,re} \\ \boldsymbol{v}_{im} &= \mathbf{A_{im}}\boldsymbol{p} + \mathbf{B_{im}}\boldsymbol{q} + \bar{\boldsymbol{v}}_{e,im} \end{aligned} \tag{2.29}$$

where $\boldsymbol{p} = \boldsymbol{p}_g - \boldsymbol{p}_d$, $\boldsymbol{q} = \boldsymbol{q}_g - \boldsymbol{q}_d$.

The linear error is expected to grow as the system inputs $\boldsymbol{p}$ and $\boldsymbol{q}$ deviate further from the linearization point, which is zero. To explore this behavior, we conducted 12 distinct trials, systematically increasing the inputs $\boldsymbol{p}$ and $\boldsymbol{q}$ to observe the impact on linear errors. Specifically, for buses 2, 3, 4, 5, 6, 7, 84, and 85, which are equipped with generators, the power demand components $P_{d,i}$ and $Q_{d,i}$ were kept constant. The variations came from the power injections $P_{g,i}$ and $Q_{g,i}$, which were incrementally raised from 0.1 MW/MVAr to 1.2 MW/MVAr in equal steps. For the remaining buses that lack generators, we increased the power demands incrementally, starting from minor starting values with increments of 0.005 MW/MVAr in each trial.

(5 of the total 85 buses are shown in the figure):

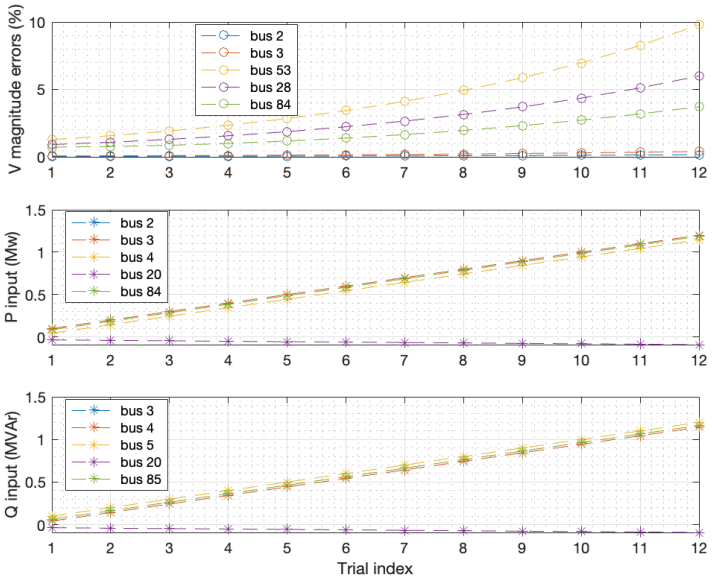


Figure 2.1: The linear errors along with the increasing system inputs

The linear errors depicted in Figure 2.1 are calculated based on the voltages computed by the linear model (equation 2.29) and the actual voltages, where the latter are derived using the Matpower toolbox in MATLAB. The first subplot in Figure 2.1 illustrates the voltage magnitude linear errors for a subset of 5 buses out of the total 85, selected based on the minimum and maximum errors. The second and third subplots present the active (P_i) and reactive (Q_i) power inputs for 5 of the buses, respectively.

Analysis of figure 2.1 reveals that linear voltage errors escalate with increasing system inputs, denoted by $\boldsymbol{p}$ and $\boldsymbol{q}$. Specifically, the maximum linear voltage error across all buses is approximately 2% — as indicated by the yellow line in the first subplot of figure 2.1—when the maximum system inputs from all generators are around 0.2 Mw/MVar. This error escalates to 10% as the system inputs reach approximately 1.3 Mw/MVar. Given that the generator capacity for IEEE-85 is defined as $P_{g,i} \in [0, 1]$ Mw and $Q_{g,i} \in [-1, 1]$ MVar for $i \in \mathcal{G}$ [8], it is inferred that linear errors will not exceed 10%, which is considered acceptable.

3

Online OPF algorithm

While traditional Optimal Power Flow (OPF) methods have been extensively explored over the past few decades, it is only in recent years that online OPF algorithms have begun to attract significant attention. A critical limitation of conventional OPF techniques is the necessity to wait for the iterative process to converge before obtaining a solution that is actionable within the network [26] [14]. This approach is increasingly impractical in the context of modern electricity networks, which are rapidly evolving to incorporate a greater share of renewable energy sources, such as distributed wind and solar generation. These sources introduce significant randomness and fluctuations into the system [26] [14] [18], compounded by the growing complexity and scale of power networks that demand enhanced control capabilities. To effectively manage these rapidly changing system states, controllers must operate at much faster timescales—on the order of seconds [26] [6] or even sub-seconds [30]. Traditional OPF methodologies, which operate on slower timescales, are not capable of addressing these challenges [26]. Against this backdrop, the need for real-time optimal power flow (RT-OPF) solutions, also referred to as online or dynamic OPF, has emerged as a pressing and attractive research topic in recent years.

In this chapter, we introduce the online OPF algorithms that are used in this thesis.

3.1. A primal-dual gradient online method with feedback

3.1.1. Algorithm formulation

This section delves into the Primal-Dual-Gradient-Projection (PDGP) online OPF method, which uniquely incorporates measurement data feedback. The PDGP approach, as detailed in references [6] and [3], builds upon the linear network model previously proposed in [9] and [15] and discussed in Chapter 2.

Recall the linearized power flow model:

$$\boldsymbol{x}^{(k)}(\boldsymbol{u}^k) = \mathbf{C}\boldsymbol{u}^{(k)} + \mathbf{D}\boldsymbol{w}^{(k)} \tag{3.1}$$

With the output $\boldsymbol{x}$ denoting the system states, such as voltages $\boldsymbol{v}$, while $\boldsymbol{w}$ represents the uncontrollable inputs like the power demands $\boldsymbol{p_d}$ and $\boldsymbol{q_d}$. Furthermore, $\boldsymbol{u}^k = (\boldsymbol{u}_i^k, i = 1, 2, ..., N)$ where $\boldsymbol{u}_i^k$ represents the active and reactive power injections $(P_{g,i}, Q_{g,i})$.

It's crucial to understand that equation 3.1 diverges from a dynamical system model, as it does not exhibit a dependency between successive time steps $(k+1)$ and k. Rather, it establishes an algebraic relationship, offering an approximation of the power flow equation at each individual time step.

Using the linearized model 3.1, the OPF problem can be reformulated as follows:

$$\min_{\boldsymbol{u}} \quad f_o^{(k)}\left(\boldsymbol{x}^{(k)}(\boldsymbol{u})\right) + \sum_{i=1}^{N} f_i^{(k)}(\boldsymbol{u}) \tag{3.2a}$$

$$\text{s.t.} \quad \boldsymbol{u}_i \in \mathcal{U}_i^{(k)}, i = 1, \ldots, N \tag{3.2b}$$

$$g_j^{(k)}\left(\boldsymbol{x}^{(k)}\right) \leq 0, j = 1, \ldots, M \tag{3.2c}$$

$$\boldsymbol{x}^{(k)}(\boldsymbol{u}^k) = \mathbf{C}\boldsymbol{u}^{(k)} + \mathbf{D}\boldsymbol{w}^{(k)} \tag{3.2d}$$

In the context of the PDGP online OPF framework, the system is characterized by N buses and M constraints. The formulation of the objective function encompasses both dependent variables $\boldsymbol{x}^{(k)}(\boldsymbol{u})$, such as voltages and currents, and independent variables $\boldsymbol{u}$, like control power injections, namely $\boldsymbol{u_i} = [P_{g,i}, Q_{g,i}]^T$. This relationship is succinctly captured in the expression(see equation 3.2a):

$$f_o^{(k)}\left(\boldsymbol{x}^{(k)}(\boldsymbol{u})\right) + \sum_{i=1}^{N} f_i^{(k)}(\boldsymbol{x}_i) \tag{3.3}$$

where $f_o^{(k)}$ denotes the primary component of the objective function pertaining to dependent variables, and $f_i^{(k)}$ represents the contributions from individual independent variables.

The system's constraints are defined by $g_j^k(\boldsymbol{u}^k)$ (equation 3.2c), which enforce time-varying requirements on the dependent variables $\boldsymbol{x}$. Notably, the constraints are linear as per the discussions in Chapter 2.

To illustrate, a linear constraint can be represented as follows

$$\begin{aligned} g_{1,n}^k\left(\{\boldsymbol{u_i}\}_{i\in G}\right) &= V_{\min} - \tilde{V_n}^k \\ g_{2,n}^k\left(\{\boldsymbol{u_i}\}_{i\in G}\right) &= -V_{\max} + \tilde{V_n}^k \end{aligned} \tag{3.4}$$

In our OPF problem, each g_i denotes the voltage constraint for the certain bus i, so that $N = M$.

The online feedback method we use, which is developed by [3] is based on primal-dual gradient projection, so firstly, the time-varying Lagrangian function can be written as:

$$\mathcal{L}^{(k)}(\boldsymbol{u}, \boldsymbol{\lambda}) := h^{(k)}(\boldsymbol{u}) + \boldsymbol{\lambda}^{\top}\boldsymbol{g}^{(k)}\left(\boldsymbol{x}^{(k)}(\boldsymbol{u})\right) \tag{3.5}$$

$h^{(k)}(\boldsymbol{u})$ is the total objective functions $f_0^{(k)}\left(\boldsymbol{x}^{(k)}(\boldsymbol{u})\right) + \sum_{i=1}^{N} f_i^{(k)}(\boldsymbol{u}_i)$ for simplicity. Denote the vector of dual variables associated with the constraints:

$$\boldsymbol{\lambda}^{\top}\boldsymbol{g}^{(k)}(\boldsymbol{x}^{(k)}(\boldsymbol{u}) \tag{3.6}$$

where $\boldsymbol{g}^{(k)}(\boldsymbol{x}^{(k)}(\boldsymbol{u}) := g_{1,n}^{(k)}\left(\{\boldsymbol{u}_i\}_{i\in\mathcal{G}}\right), g_{2,n}^{(k)}\left(\{\boldsymbol{u}_i\}_{i\in\mathcal{G}}\right), n \in \mathcal{N}$.

It is important to elucidate the concept of linear convergence, as this convergence property of the algorithm allows for an explicit characterization of the error bound in a time-varying setting.

Linear convergence is characterized by comparing the distance between the algorithmically computed sub-optimal solution, $\boldsymbol{u}$, and the actual optimal solution, $\boldsymbol{u}^*$, across iterations. Specifically, if the following condition is met:

$$||\boldsymbol{u}^{k+1} - \boldsymbol{u}^{*,k+1}|| \leq \eta||\boldsymbol{u}^k - \boldsymbol{u}^{*,k+1}||, \quad \eta \in (0, 1) \tag{3.7}$$

where η denotes the rate of convergence or the asymptotic error constant, the algorithm exhibits linear convergence for $q = 1$ and $\eta \in (0, 1)$. Under these conditions, the sequence of sub-optimal solutions, $\tilde{\boldsymbol{u}}_{\boldsymbol{k}}$, is said to converge linearly to the actual optimal solution, $\boldsymbol{u}_{\boldsymbol{k}}^{*}$, as cited in [25].

This principle underscores the importance of ensuring that the cost function within our system is both m-strongly convex and L-smooth. The projected gradient optimization problem's Q-linear convergence on such a cost function is a critical aspect of our methodology [25]. The strong convexity attribute, in particular, is vital for guaranteeing the desired convergence behaviour and thereby justifying the optimization strategy employed in our system.

In order to ensure linear convergence, a regularized Lagrangian function needs to be built based on Equation 3.5 ([6] and [3]):

$$\begin{aligned}\mathcal{L}_{\nu,\epsilon}^{k}(\boldsymbol{u},\boldsymbol{\gamma},\boldsymbol{\mu}) :=& \mathcal{L}^{k}(\boldsymbol{u},\boldsymbol{\gamma},\boldsymbol{\mu}) \\ &+ \frac{\nu}{2}\|\boldsymbol{u}\|_2^2 - \frac{\epsilon}{2}\left(\|\boldsymbol{\gamma}\|_2^2 + \|\boldsymbol{\mu}\|_2^2\right)\end{aligned} \tag{3.8}$$

Within the framework of our PDGP online OPF method, the regularization plays a pivotal role, characterized by the constants $\nu > 0$ and $\epsilon > 0$.

We use these constants as design parameters to introduce regularization, which helps achieve linear convergence in our optimization process. The regularization has two key characteristics: the regularized Lagrangian function 3.8 is strictly convex for the control variables $\boldsymbol{u}$ and strictly concave for the dual variables. This combination of convexity and concavity is crucial for finding the optimal solution $\boldsymbol{u}$ represented as the saddle point in our optimization challenge. Finding this saddle point is central to how our online algorithm works:

$$\max_{\boldsymbol{\gamma}\in\mathbb{R}_{+}^{M},\boldsymbol{\mu}\in\mathbb{R}_{+}^{M}} \min_{\boldsymbol{u}\in\mathcal{U}^{k}} \mathcal{L}_{\nu,\epsilon}^{k}(\boldsymbol{u},\boldsymbol{\gamma},\boldsymbol{\mu}) \tag{3.9}$$

We define the saddle point, which includes both the optimal primal variable $\boldsymbol{u}^{*k}$ and the optimal dual variables $\boldsymbol{\gamma}^{*,k}$, $\boldsymbol{\mu}^{*,k}$, as $\boldsymbol{z}^{*,\boldsymbol{k}} = \left[\left(\boldsymbol{u}^{*,k}\right)^{\top}, \left(\boldsymbol{\gamma}^{*,k}\right)^{\top}, \left(\boldsymbol{\mu}^{*,k}\right)^{\top}\right]^{\top}$. It's crucial to understand that the saddle point $\boldsymbol{z}_{\boldsymbol{\epsilon}}^{*,\boldsymbol{k}}$, derived from the regularized Lagrangian function (referred to as function 3.8), is dependent on ϵ, and in general, means $\boldsymbol{z}_{\boldsymbol{\epsilon}}^{*,\boldsymbol{k}}$ does not coincide with the saddle point of the original Lagrangian function 3.5, making $\boldsymbol{u}_{\boldsymbol{\epsilon}}^{*,k}$ different from $\boldsymbol{u}^{*,k}$. Therefore, our goal is to achieve linear convergence towards a sub-optimal (regularized) solution.

Setting the step size as α, then the following primal-dual gradient projection method to solve the time-varying saddle-point problem 3.9 [6] is updated:

$$\begin{aligned}
\tilde{\boldsymbol{u}}_{i}^{k+1} &= \operatorname{proj}_{\mathcal{U}_{i}^{k}}\left\{\tilde{\boldsymbol{u}}_{i}^{k} - \alpha\nabla_{\boldsymbol{u}_{i}}\mathcal{L}_{\nu,\epsilon}^{k}(\boldsymbol{u},\boldsymbol{\gamma},\boldsymbol{\mu})\big|_{\tilde{\boldsymbol{u}}_{i}^{k},\tilde{\boldsymbol{\gamma}}^{k},\tilde{\boldsymbol{\mu}}^{k}}\right\}, && \forall i \in \mathcal{G} \\
\tilde{\gamma}_{n}^{k+1} &= \operatorname{proj}_{\mathbb{R}_{+}}\left\{\tilde{\gamma}_{n}^{k} + \alpha\left(g_{1,n}^{k}\left(\tilde{\boldsymbol{u}}^{k}\right) - \epsilon\tilde{\gamma}_{n}^{k}\right)\right\}, && \forall n \in \mathcal{M} \\
\tilde{\mu}_{n}^{k+1} &= \operatorname{proj}_{\mathbb{R}_{+}}\left\{\tilde{\mu}_{n}^{k} + \alpha\left(g_{2,n}^{k}\left(\tilde{\boldsymbol{u}}^{k}\right) - \epsilon\tilde{\mu}_{n}^{k}\right)\right\}, && \forall n \in \mathcal{M},
\end{aligned} \tag{3.10}$$

As defined in Equation 1.9, $\boldsymbol{u}$ represents the control variables. The update mechanism, outlined in equations 3.10, illustrates the online OPF algorithm operating in an open-loop fashion. This method is developed based on the principles of primal-dual gradient projection, as extensively discussed in [6] and [3].

However, as highlighted in Chapter 1, we encounter a significant challenge: the measurements of uncontrollable demands, specifically $\boldsymbol{p_d}$ and $\boldsymbol{q_d}$, are often not readily available [19], [25]. Furthermore, the reliance of Equations 3.10 on a linear approximation of the power system introduces a limitation, as this approach does not account for linear errors inherent in such models.

To address these issues, we introduce a feedback strategy into the update equations 3.10. This strategy is designed to mitigate the impact of unmeasurable uncontrollable demands and the limitations of linear approximations. By incorporating real-time feedback, the algorithm adapts to actual system

states, enhancing the accuracy and robustness of the online OPF solution. This feedback mechanism represents a critical advancement in our methodology, enabling the algorithm to better handle the complexities and uncertainties of modern power systems.

The feedback mechanism implemented in this thesis, which was proposed by [6] and [3], utilizes voltage magnitude measurements as the core of its feedback strategy. By denoting these voltage measurements as $\{y_n^k\}_{n\in\mathcal{N}}$, the approach adapts by substituting the theoretical predictions $g_{1,n}^k(\tilde{\boldsymbol{u}}^k)$ and $g_{2,n}^k(\tilde{\boldsymbol{u}}^k)$ with actual voltage measurements from the system. Consequently, this substitution transforms the original open-loop version of the update equations, referenced in equation 3.10, into a closed-loop formulation. Substitute equation 3.4 into the update equations 3.10, and replace the voltage $\tilde{V}_n^{\,k}$ with the measurement voltage y_n^k, the close loop updates can be expressed as follows:

$$\begin{aligned}
\tilde{\boldsymbol{u}}_i^{k+1} &= \mathrm{proj}_{\mathcal{U}_i^k}\left\{\tilde{\boldsymbol{u}}_i^k - \alpha\nabla_{\tilde{\boldsymbol{u}}_i}\mathcal{L}_{\nu,\epsilon}^k(\boldsymbol{u},\boldsymbol{\gamma},\boldsymbol{\mu})\big|_{\tilde{\boldsymbol{u}}_i^k,\tilde{\boldsymbol{\gamma}}^k,\tilde{\boldsymbol{\mu}}^k}\right\}\\
\tilde{\gamma}_n^{k+1} &= \mathrm{proj}_{\mathbb{R}_+}\left\{\tilde{\gamma}_n^k + \alpha\left(V^{\min} - y_n^k - \epsilon\tilde{\gamma}_n^k\right)\right\}\\
\tilde{\mu}_n^{k+1} &= \mathrm{proj}_{\mathbb{R}_+}\left\{\tilde{\mu}_n^k + \alpha\left(y_n^k - V^{\max} - \epsilon\tilde{\mu}_n^k\right)\right\}
\end{aligned} \tag{3.11}$$

This methodological shift to a closed-loop system, through the direct incorporation of real-time voltage measurements, significantly enhances the algorithm's ability to dynamically adjust to the actual conditions of the power system. It embodies a practical solution to the limitations associated with the linear approximations and unavailable demands discussed previously. By aligning the optimization process more closely with the physical realities of the network, this feedback strategy improves the precision and reliability of the online OPF outcomes, ensuring more effective and efficient control over power system operations.

The proposed feedback control strategy can be illustrated as follows [6]:

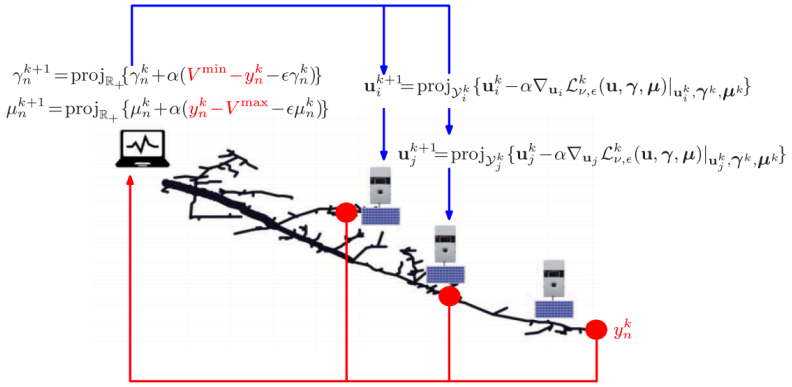


Figure 3.1: Feedback strategy from [6]

In the closed-loop system configuration, real-time measurements from each bus, specifically voltage measurements denoted as y_i^k, are systematically collected at each time instant k (as illustrated in figure 3.1) and subsequently fed into the control algorithm. This process enables the controller to accurately compute the optimal control variables for the forthcoming time instant, $k+1$, namely u_i^{k+1} and u_j^{k+1}, as depicted in figure 3.1. These control actions are then executed locally at each control device, such as inverters and batteries, ensuring that adjustments are both precise and timely.

As the system progresses to time instant $k+2$, a new set of measurements is gathered, feeding back into the controller to inform the next set of control decisions. This iterative process repeats continuously, creating a dynamic feedback loop that allows for the ongoing adjustment of control variables in response to real-time system states. This method ensures that the control strategy is both adaptive and responsive to the time-changing conditions of the power network, enhancing operational efficiency and reliability.

3.1.2. Convergence analyses

The tracking performance is analysed in [6] and [3], firstly, some assumptions should be made [6], [23]:

1. The cost functions $f_i(\boldsymbol{u})$ are convex and continuously differentiable for each $i \in \mathcal{G}$ and $k \geq 0$. Define further the gradient map:

$$\nabla \boldsymbol{f}^k(\boldsymbol{u}) := \left[\nabla^{\top}_{\boldsymbol{u}_1} f_1^k(\boldsymbol{u}_1), \ldots, \nabla^{\top}_{\boldsymbol{u}_{N_{\mathcal{G}}}} f_{N_{\mathcal{G}}}^k\left(\boldsymbol{u}_{N_{\mathcal{G}}}\right)\right]^{\top}. \tag{3.12}$$

   Then, the gradient map $\nabla \boldsymbol{f}^k(\boldsymbol{u})$ is Lipschitz continuous with constant L over the compact set $\mathcal{U}^k$, for all $k \geq 0$:

$$\left\|\nabla \boldsymbol{f}^k(\boldsymbol{u}) - \nabla \boldsymbol{f}^k(\boldsymbol{u}')\right\|_2 \leqslant L \left\|\boldsymbol{u} - \boldsymbol{u}'\right\|_2, \forall \boldsymbol{u}, \boldsymbol{u}' \in \mathcal{U}^k \tag{3.13}$$

2. For $k \geq 0$, there exists a set of feasible power injections $\{\tilde{\boldsymbol{u}}_i\}_{i\in\mathcal{G}} \in \mathcal{U}^k$ such that $g_{1,n}^k\left(\{\tilde{\boldsymbol{u}}_i\}_{i\in\mathcal{G}}\right) \leq 0$ and $g_{2,n}^k\left(\{\tilde{\boldsymbol{u}}_i\}_{i\in\mathcal{G}}\right) \leq 0$ for all $n \in \mathcal{N}$.

3. There exists a constant $\sigma_u \geq 0$ such that $\left\|\boldsymbol{u}^{*,k+1} - \boldsymbol{u}^{*,k}\right\| \leq \sigma_u$ for all $k \geq 0$, together with the dual variables $\boldsymbol{\gamma}$ and $\boldsymbol{\mu}$, upon defining $\boldsymbol{z}^{*,\boldsymbol{k}} = \left[\left(\boldsymbol{u}^{*,k}\right)^{\top}, \left(\boldsymbol{\gamma}^{*,k}\right)^{\top}, \left(\boldsymbol{\mu}^{*,k}\right)^{\top}\right]^{\top}$, and the upper limit constant σ_z such that $\left\|\boldsymbol{z}^{*,k+1} - \boldsymbol{z}^{*,k}\right\| \leq \sigma_z$.

4. There exist constants $\sigma_d \geq 0$ such that $\left|g_n^{k+1}\left(\boldsymbol{u}^{*,k+1}\right) - g_n^k\left(\boldsymbol{u}^{*,k}\right)\right| \leq \sigma_d$ for all $k \geq 0$ and $n \in \mathcal{N}$

5. There exists a scalar $e < +\infty$ such that the other errors including measurement error $\boldsymbol{e_m}$, linear error $\boldsymbol{e_l}$, and possible actuator delay can be bounded as

$$\max\left\{\left\|\boldsymbol{e_m^k}\right\|_2, \left\|\boldsymbol{e_l^k}\right\|_2\right\} \leq e \tag{3.14}$$

With all the 5 assumptions in place, denote $\tilde{\boldsymbol{z}}^k$ the sub-optimal sequence generated by the online feedback OPF algorithm 3.11, by Theorem 1 in [6], if the stepsize $\alpha > 0$ is chosen such that :

$$\rho(\alpha) := \sqrt{1 - 2\eta\alpha + \alpha^2 L_{v,\epsilon}^2} < 1 \tag{3.15}$$

that is $0 < \alpha < 2\eta / L_{v,\epsilon}^2$, then the following convergence bound between the sub-optimal $\tilde{\boldsymbol{z}}^k$ and the actual optimal $\boldsymbol{z}^{(*,\boldsymbol{k})}$ of the OPF problem 3.2 is stated:

$$\operatorname{limsup}_{k\to\infty} \left\|\tilde{\boldsymbol{z}}^k - \boldsymbol{z}^{*,k}\right\|_2 = \frac{1}{1-\rho(\alpha)}\left[\sqrt{2}\alpha e + \sigma_{\boldsymbol{z}}\right] \tag{3.16}$$

with

$$L_{v,\epsilon} = \sqrt{(L + v + 2G)^2 + 2(G + \epsilon)^2} \tag{3.17}$$

Where G signifies the upper bound for the norm of the gradient of the constraint functions with respect to the control variables $\boldsymbol{u}$. Define $\boldsymbol{g}_1^k(\boldsymbol{u})$ and $\boldsymbol{g}_2^k(\boldsymbol{u})$ be a vector stacking all the functions $g_{1,n}^k\left(\{\boldsymbol{u}_i\}_{i\in\mathcal{G}}\right), n \in \mathcal{N}$ and $g_{2,n}^k\left(\{\boldsymbol{u}_i\}_{i\in\mathcal{G}}\right), n \in \mathcal{N}$, respectively, then, given that these functions are linear in $\boldsymbol{u}$, it follows that there exists a constant G such that $\left\|\nabla_{\boldsymbol{u}} \boldsymbol{g}_1{}^k(\boldsymbol{u})\right\|_2 \leq G$ and $\left\|\nabla_{\boldsymbol{u}} \boldsymbol{g}_2{}^k(\boldsymbol{u})\right\|_2 \leq G$ for all $\boldsymbol{u} \in \mathcal{U}^k$ for all $k \geq 0$ [6]

The parameter η is defined as the minimum of v and ϵ, where $v > 0$ and $\epsilon > 0$ are the regularization factors introduced in the Tikhonov regularization terms. The stepsize for the algorithm's iterative process is denoted by α, a critical factor in ensuring convergence. The Lipschitz constant of the cost functions, represented by L, is established under Assumption 1, guaranteeing that the functions' gradients are bounded within a certain rate of change.

The variable e aggregates the maximum value among measurement errors and linear approximation errors, encapsulating the uncertainty and inaccuracies inherent in the system modelling and data acquisition processes, as described in Assumption 5.

Lastly, σ_z represents the upper bound that quantifies the variability of the optimal solutions across iterations, a concept introduced in Assumption 4 to accommodate changes in the system's optimal state over time.

Another important remark should be noted, define:

$$\phi^{(k)} : \boldsymbol{z} \mapsto \begin{bmatrix} \nabla_{\boldsymbol{x}} \mathcal{L}_{v,\epsilon}^{(k)}(\boldsymbol{z}) \\ -\boldsymbol{g}^{(k)}\left(\boldsymbol{y}^{(k)}(\boldsymbol{x})\right) + \epsilon\boldsymbol{\lambda} \end{bmatrix} \tag{3.18}$$

The above linear convergence performance relays on the fact that the map $\phi^{(k)}(\boldsymbol{z})$ strongly monotone over $\mathcal{X}^{(k)} \times \mathcal{D}^{(k)}$ [3], which is also the reason why the regularized Lagrangian function 3.8 is used instead of 3.5. It is discussed in [3] that the map $\phi^{(k)}(\boldsymbol{z})$ is not strongly monotone if the original Lagrangian function 3.5 is used, so that the linear convergence can not be guaranteed.

However, as mentioned before, since the sub-optimizer $\boldsymbol{z}^{(*,k)}$ computed by the online algorithm 3.11 is not in the set of saddle points of the original Lagrangian $\mathcal{L}^{(k)}(\boldsymbol{x}, \boldsymbol{\lambda})$ 3.5 [3], [6]. As a result, larger values of v and ϵ lead to larger perturbations of the solution trajectories of the original time-varying optimization problem 3.2, meaning that optimality and constraint satisfaction are partly sacrificed. On the other hand, small values of v and ϵ lead to a larger asymptotic bound 3.16, so that the values of v and ϵ should be selected numerically (or analytically, whenever possible) based on specific implementation goals[3].

The effects of the design parameters such as α, v, ϵ and system parameters such L will be discussed and tested in Chapter 5.

4

Gaussian process and application to OPF with unknown cost

Supervised learning is the task of inferring input-output mappings based on empirical data, known as the training dataset [29]. Among the different approaches utilized in supervised learning, the Gaussian process (GP) stands out due to its widespread application and flexibility. Unlike a Gaussian probability distribution, which describes the behaviour of random variables that are scalar or vector in nature, a Gaussian process extends this concept to the realm of functions, offering a framework to govern their properties [29].

This chapter is aimed at exploring the Gaussian process in depth. It will commence with an introductory overview of GP, illustrating its basic principles and how it generalizes the notion of probability distributions to function spaces. Following this, we will delve into the kernel functions that play a pivotal role in the GP methodology, influencing the GP's ability to learn and adapt to data. The chapter will conclude with an examination of how Gaussian processes can be extended to estimate functions that are subject to specific shape constraints, showcasing the function and power of GP in solving complex supervised learning problems.

4.1. Gaussian process

This thesis is dedicated to the development of an online OPF algorithm designed for scenarios where the exact cost functions are unknown, such as those involving subjective measures of personal discomfort or environmental concerns like air pollution. These factors, although critically important, present significant modelling challenges due to their inherent complexity and variability. To address this issue, our approach leverages real-time feedback to learn the cost functions dynamically. Specifically, we incorporate feedback from air pollution sensors deployed around generators, which provide instantaneous data on the environmental impact, serving as an approximation for the cost associated with pollution [19] .

To effectively model and incorporate these dynamically learned cost functions into the OPF framework, we employ the shape-constrained Gaussian Process (SC-GP). This method allows us to not only approximate the cost functions with a high degree of accuracy but also to adhere to certain pre-defined shape constraints that reflect the underlying requirements. The shape-constrained GP thus serves as a powerful tool to bridge the gap between the need for precise mathematical models of cost functions and the practical challenges of quantifying subjective or complex costs in real-time.

In the realms of probability theory and statistics, Bayesian analysis lays the groundwork for understanding Gaussian processes. Consider a foundational linear regression model with Gaussian noise:

$$f(\boldsymbol{x}) = \boldsymbol{x}^\top \boldsymbol{w}, \quad y = f(\boldsymbol{x}) + \varepsilon \tag{4.1}$$

where $\boldsymbol{x}$ is the input vector, and $\boldsymbol{w}$ is a vector of unknown weights (parameters) of the linear model.

The function value is denoted by f, with y capturing the observed target value. The Gaussian noise, ε, is characterized by a zero mean and variance σ_n^2:

$$\varepsilon \sim \mathcal{N}\left(0, \sigma_n^2\right) \tag{4.2}$$

The Bayesian framework needs the specification of a prior distribution over the parameters, opting for a zero mean Gaussian prior with covariance matrix Σ_p for the weights, with this prior, one can prove (see [29], eq. 2.8) that $\boldsymbol{w}$ is normally distributed with the distribution :

$$\boldsymbol{w} \sim \mathcal{N}\left(0, \Sigma_p\right) \tag{4.3}$$

Given a dataset $(\boldsymbol{x}, \boldsymbol{y})$, Bayesian inference doesn't merely extrapolate a singular value f^* at a new input x^*; it computes the probability distribution encompassing all possible f^* values, by marginalizing over $\boldsymbol{w}$, and substituting the definition, and by noting that $\boldsymbol{w}$ is normally distributed, one computes:

$$\begin{aligned} p\left(f^* \mid x^*, \boldsymbol{x}, \boldsymbol{y}\right) &= \int p\left(f^* \mid x^*, \boldsymbol{w}\right) p(\boldsymbol{w} \mid \boldsymbol{x}, \boldsymbol{y}) d\boldsymbol{w} = \int x^{*\top} \boldsymbol{w} p(\boldsymbol{w} \mid \boldsymbol{x}, \boldsymbol{y}) d\boldsymbol{w} \\ &= \mathcal{N}\left(\frac{1}{\sigma_n^2} x^{*\top} A^{-1} \boldsymbol{x} \boldsymbol{y}, x^{*\top} A^{-1} x^*\right) \end{aligned} \tag{4.4}$$

Consequently, the predictive distribution of the unknown value f^* is normal following equation 4.4, with a mean given by the posterior mean of the weights multiplied by the test input. Moreover, the explicit computation of $\boldsymbol{w}$ becomes unnecessary.

Gaussian processes allow one to perform Bayesian inference over functions. We define a Gaussian process as a collection of random variables indexed in x, that is, $\{f(x) : x \in \mathbb{R}\}$ such that every finite subset of random variables is a multivariate normal distribution:

$$f \sim \mathcal{N}(\mu(\boldsymbol{x}), \sigma^2(\boldsymbol{x})) \tag{4.5}$$

with $\mu(\boldsymbol{x})$ and $\sigma^2(\boldsymbol{x})$ are mean and covariance function respectively.

let $\boldsymbol{x} = (x_i)_{i=1}^n$and let $y = (f(x_i))_{i=1}^n$ be a finite number of samples of the unknown Gaussian process (training dataset). Let $\boldsymbol{x}*$ be a set of points at which one wishes to extrapolate the values $y^* = (f(x_i^*))_{i=1}^n$. The definition of Gaussian processes allows one to compute y^* via Bayesian inference because they are normally distributed [29]:

$$\begin{bmatrix} f \\ f^* \end{bmatrix} = \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}) \\ \mu^*\left(x^*\right) \end{bmatrix}, \sigma^2\left(\boldsymbol{x}, x^*\right)\right] \tag{4.6}$$

For a detailed derivation of these steps, one can refer to Chapter 2 in [29].

Ultimately, the predictive model for f^* is Gaussian, described by

$$f^* \sim \mathcal{N}\left(\mu\left(x^*\right), \sigma^2\left(x^*\right)\right) \tag{4.7}$$

with the mean and variance:

$$\begin{aligned} \mu &= \boldsymbol{k}_*^T \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \boldsymbol{y} \\ \sigma^2 &= k\left(x_*, x_*\right) - \boldsymbol{k}_*^T \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \boldsymbol{k}_* \end{aligned} \tag{4.8}$$

where

- $\boldsymbol{k}_*$ is the vector of covariances between the new point x_* and all points in $\boldsymbol{x}$
- $\mathbf{K}$ is the covariance matrix computed from all pairs of points in $\boldsymbol{x}$.
- σ_n^2 is the noise variance in the observations.
- σ_*^2 is the variance of the predictive distribution at x_*
- $\boldsymbol{y}$ is the vector of observed outputs.
- $\mathbf{I}$ is the identity matrix.

4.1.1. Covariance functions

In the preceding section, we highlighted that the covariance function, as $\mathbf{K}$ and $\boldsymbol{k}_*$ outlined in equation 4.8, serves as a foundational element in Gaussian process prediction. This function's significance lies in its role in encapsulating the presumptions regarding the function to be learned [29]. This section delves into the specific covariance functions employed, alongside their characteristics.

At its core, a covariance function, also referred to as a kernel, is denoted by k. It is a binary function that maps a pair of inputs, $\boldsymbol{x} \in \mathcal{X}$ and $\boldsymbol{x}' \in \mathcal{X}$, to a real number $\mathbb{R}$ [29]. Within the domain of Gaussian processes, these covariance functions are synonymous with kernel functions.

An essential concept to understand is the covariance matrix $\mathbf{K}$, which can be derived from a set of input points $\{\boldsymbol{x}i \mid i = 1, \ldots, n\}$. The elements of $\mathbf{K}$, given by $K_{i,j} = k(x_i, x_j)$, are determined by the covariance function. Consequently, if k represents a covariance function, then $\mathbf{K}$ becomes the covariance matrix, pivotal for the Gaussian process.

One of the most commonly used covariance functions that will be utilised in this thesis is the squared exponential (SE) covariance function (D-dimensional):

$$k_{\mathrm{SE}}\left(x_i, x_j\right) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}\left(x_{d,i} - x_{d,j}\right)^2 l_d^{-2}\right) \tag{4.9}$$

Where l_d is the characteristic length-scale which will be discussed in detail later, σ_f^2 is another design parameter which can be multiplied in order to get any desired process variance.

For one-dimensional inputs and outputs(input $\boldsymbol{x}$ and output $\boldsymbol{y}$ are one-dimensional vectors), a more compact form of SE function can be written as:

$$k_{\mathrm{SE}}(r) = \sigma_f^2 \exp\left(-\frac{r^2}{2l_d^2}\right) \tag{4.10}$$

Where $r = x_i - x_j$.

This covariance function is infinitely differentiable, which means that the GP with this covariance function has mean square derivatives of all orders and is very smooth; given these ideal properties, it is argued in [29] that it might be unrealistic for modelling many physical processes. However, the SE covariance function is probably the most widely-used kernel within the kernel machines field.

Apart from the SE covariance functions, other typical ones include Linear Kernel, The γ-exponential Covariance function, Rational Quadratic Covariance Function, Matern Kernel, and Periodic Kernel; the details and attributes are outside the scope of this thesis, the interested readers can find the details in [29].

4.1.2. Adaptation of hyperparameters

Among the covariance functions previously discussed, certain design parameters, known as hyper-parameters—such as length scales l_d—remain to be determined. In the realm of GP learning, the selection of a covariance function along with its hyper-parameters is often integrated into the training process [29], [13]. While this approach adds a layer of complexity, it is crucial for tailoring the GP model to specific applications.

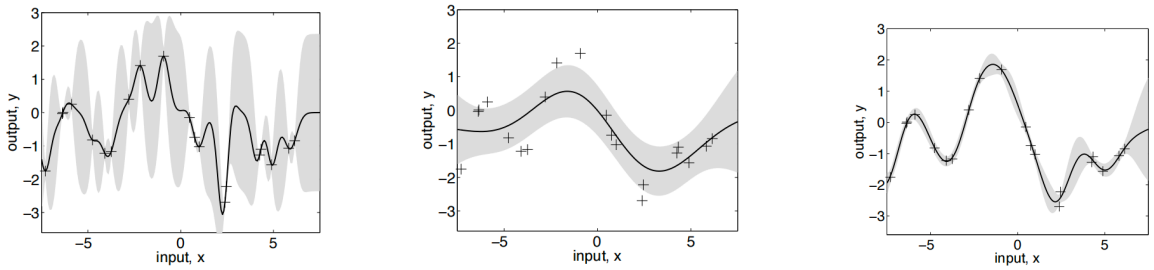
For a majority of models aimed at function learning, the task of integrating over the parameter space to compute the necessary values is analytically infeasible. As a result, finding suitable approximations becomes a challenging task. Nevertheless, Gaussian process regression models that incorporate Gaussian noise stand out as a notable exception. These models not only allow for the analytical tractability of integrals over parameters but also exhibit remarkable flexibility [29].

This section will explore the role of hyper-parameters within the Gaussian process framework, focusing on their influence on model performance. Additionally, we will introduce a method for training these hyper-parameters, using the Squared Exponential (SE) covariance function as a case study.

Take the SE covariance function as an example; recall the SE function in the following form:

$$k_{\mathrm{SE}}(x_i, x_j) = \sigma_f^2 \exp\left(-\frac{(x_i - x_j)^2}{2l_d^2}\right) \tag{4.11}$$

The hyper-parameter l_d here plays the role of characteristic length-scale, loosely speaking, how far it is needed to move in input space for the function values to become uncorrelated [29]. The following plots show the influence of l_d.



(a) $l_d = 0.3$, too narrow (b) $l_d = 3$, too wide (c) $l_d = 1$, good-fit

Figure 4.1: The influence of l_d Source: [29]

For a function tested in [29], '+' in figure 4.1 denotes the observed data points while the grey areas in between represent 95% confidence region for the underlying function. Hyper-parameters $(l_d, \sigma_f, \sigma_n = (0.3, 1.08, 0.00005))$ are used to in figure 4.1c while $(l_d, \sigma_f, \sigma_n = (3, 1.16, 0.89))$ are used in figure 4.1b. l_d is the dominant parameter, and the remaining two parameters are set by optimizing the marginal likelihood and will be discussed later.

Figure 4.1 shows that if the length-scale l_d is too narrow, with the noise parameter reduced to $\sigma_n = 0.00005$, the signal has the greater flexibility and lower noise as figure 4.1c, in this case, the prediction can capture more accuracy but is also over-fit with noise. In contrast, figure 4.1b shows that when l_d is too broad, with the noise parameter σ_n increased to 0.89, the data is explained by a slowly varying function with much noise, so-called under-fit.

In this function, a combination of hyper-parameters of $(l_d, \sigma_f, \sigma_n = (1, 1, 0.1))$ is appropriate for good-fit as figure 4.1c shows.

To ascertain the optimal set of hyper-parameters, such as the length scale l_d and signal variance σ_f within the SE covariance function, the principle of maximal marginal likelihood is commonly employed. This method plays a pivotal role in fine-tuning the SE covariance function to achieve superior model performance.

Nonetheless, insights from simulation tests presented in the subsequent chapter reveal another possibility: the introduction of a shape constraint may skip the necessity for this optimization step. This advancement suggests that under certain conditions, we can simplify the model calibration process, thereby enhancing efficiency without compromising the model's integrity, which is our main goal, considering that a short period of computation time is desired. Detailed discussions and implications of this finding will be further elaborated in the following sections.

4.2. Estimating shape-constrained functions using Gaussian processes

As mentioned in Chapter 3, the efficacy of most online OPF algorithms relies on the presence of a strongly convex objective function. The integration of shape constraints into this framework is feasible through the employment of shape-constrained GPs, as demonstrated by [19] and [20]. This approach leverages the inherent property that the derivative processes of GPs are, in themselves, Gaussian Processes [13], [27]. Consequently, the cornerstone of implementing a shape-constrained Gaussian Process involves utilizing derivative information. This facilitates the construction of joint Gaussian processes that align with the original process, enhancing the algorithm's robustness and efficacy [27].

The use of derivative information with GPs is not new and has variant methods; for this thesis, we focus on the method proposed by [27], which has been tested in online OPF application in [19], [20].

For the covariance function, the squared exponential (SE) correlation function can be used [27], [20] since it has derivative processes of all orders and, hence, is particularly useful for incorporation of shape constraints. For instance, the first-order derivative information can be utilized when the learnt function is monotonic, while the second-order derivative is incorporated for convex functions.

Same definitions as in section 4.1, denote $\boldsymbol{y}$ for noised observed function values, $\boldsymbol{x}$ for inputs, σ_f and l_d for the hyper-parameters, μ for the mean.

For the first-order derivative process, the corresponding variance, covariance, and mean function (joint with the original process) can be written as follows [27]:

$$\begin{aligned} \mathrm{E}\left[\frac{\partial y(x)}{\partial x}\right] &= \frac{\partial \mu(x)}{\partial x} = 0, \\ \operatorname{cov}\left[\frac{\partial y(x)}{\partial x}, \frac{\partial y\,(x')}{\partial x'}\right] &= \sigma_f^2 \exp\left(-\frac{1}{2l_d^2}\,(x-x')^2\right)\frac{1}{l_d^2}\left(1-\frac{1}{l_d^2}\,(x-x')^2\right) \\ \operatorname{cov}\left[\frac{\partial y(x)}{\partial x}, y\,(x')\right] &= \sigma_f^2 \exp\left(-\frac{1}{2l_d^2}\,(x-x')^2\right)\left(-\frac{1}{l_d^2}\,(x-x')\right). \end{aligned} \tag{4.12}$$

Similarly, the second-order derivative process of the GP and its corresponding mean and covariance function (joint with the original process and the first-order derivative process) are [27], [20] :

$$\begin{aligned} \mathrm{E}\left[\frac{\partial^2 y(x)}{\partial x^2}\right] &= \frac{\partial^2 \mu(x)}{\partial x^2} = 0, \\ \operatorname{cov}\left[\frac{\partial^2 y(x)}{\partial x^2}, \frac{\partial^2 y\,(x')}{\partial x'^2}\right] &= \sigma_f^2 \exp\left(-\frac{1}{2l_d^2}\,(x-x')^2\right)\frac{1}{l_d^4}\left(\frac{1}{l_d^4}\,(x-x')^4 - \frac{1}{l_d^2}6\,(x-x')^2 + 3\right), \\ \operatorname{cov}\left[\frac{\partial^2 y(x)}{\partial x^2}, y\,(x')\right] &= \sigma_f^2 \exp\left(-\frac{1}{2l_d^2}\,(x-x')^2\right)\left(\frac{1}{l_d^4}\,(x-x')^2 - \frac{1}{l_d^2}\right), \\ \operatorname{cov}\left[\frac{\partial^2 y(x)}{\partial x^2}, \frac{\partial y\,(t')}{\partial x'}\right] &= -\sigma_f^2 \exp\left(-\frac{1}{2l_d^2}\,(x-x')^2\right)(x-x')\left(\frac{3}{l_d^4} - \frac{1}{l_d^6}\,(x-=x')^2\right). \end{aligned} \tag{4.13}$$

Denote the input $\boldsymbol{x}$ as the points where the noisy observations are obtained, and the $\boldsymbol{d}$ as the points where the derivative constraints will be put.

Denote $\frac{\partial y(d)}{\partial d}$ as $y'(d)$, and n, m for the element numbers of $\boldsymbol{x}$ and $\boldsymbol{d}$, respectively, let $\boldsymbol{y}(\boldsymbol{x}) = [y(x_1), y(x_2), \ldots, y(x_n)]^T$ and $\boldsymbol{y}'(\boldsymbol{d}) = [y'(d_1), y'(d_2), \ldots, y'(d_m)]^T$ be the vectors of corresponding GP and derivative values, it follows that [27]:

$$\begin{bmatrix} \boldsymbol{y}(\boldsymbol{x}) \\ \boldsymbol{y}'(\boldsymbol{d}) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu 1_n \\ 0_m \end{bmatrix}, \begin{bmatrix} \mathbf{K}(\boldsymbol{x},\boldsymbol{x}) & \mathbf{K}^{01}(\boldsymbol{x},\boldsymbol{d}) \\ \mathbf{K}^{10}(\boldsymbol{d},\boldsymbol{x}) & \mathbf{K}^{11}(\boldsymbol{d},\boldsymbol{d}) \end{bmatrix}\right) \tag{4.14}$$

Where the different matrix $\mathbf{K}$ are computed by the following functions:

$$\begin{aligned} K\,(x,x') &= \sigma_f^2 \exp\left(-\frac{1}{2l_d^2}\,(x-x')^2\right) \\ K^{11}\,(d,d') &= \sigma_f^2 \exp\left(-\frac{1}{2l_d^2}\,(d-d')^2\right)\frac{1}{l_d^2}\left(1-\frac{1}{l_d^2}\,(d-d')^2\right), \\ K^{01}(x,d) &= \sigma_f^2 \exp\left(-\frac{1}{2l_d^2}(s-d)^2\right)\left(-\frac{1}{l_d^2}(x-d)\right). \end{aligned} \tag{4.15}$$

When utilizing shape constraints on the second derivatives of the GP, OPF problems with convex objective functions in [20], [19], the joint distribution of the GP and its second-order derivatives is:

$$\begin{bmatrix} \boldsymbol{y}(\boldsymbol{x}) \\ \boldsymbol{y}''(\boldsymbol{d}) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu 1_n \\ 0_m \end{bmatrix}, \begin{bmatrix} \mathbf{K}(]x, \boldsymbol{x}) & \mathbf{K}^{02}(\boldsymbol{x}, \boldsymbol{d}) \\ \mathbf{K}^{20}(\boldsymbol{d}, \boldsymbol{x}) & \mathbf{K}^{22}(\boldsymbol{d}, \boldsymbol{d}) \end{bmatrix}\right) \tag{4.16}$$

With the new k functions for $\mathbf{K}$:

$$\begin{aligned} K^{22}(d, d') &= \sigma_f^2 \exp\left(-\frac{1}{2l_d^2}(d-d')^2\right)\frac{1}{l_d^4}\left(\frac{1}{l_d^4}(d-d')^4 - \frac{1}{l_d^2}6(d-d')^2 + 3\right) \\ K^{02}(x, d) &= \sigma_f^2 \exp\left(-\frac{1}{2l_d^2}(x-d)^2\right)\left(\frac{1}{l_d^4}(x-d)^2 - \frac{1}{l_d^2}\right) \end{aligned} \tag{4.17}$$

A Gaussian Process (GP) can be steered to adhere to specific shape constraints by suitably constraining its derivatives, as elucidated in [27]. For instance, to induce an increasing function, one could ensure that the GP's first-order derivative remains positive at a selected set of inputs, as demonstrated in [27]. Similarly, to achieve convexity, the second-order derivative of the GP could be constrained to non-negative values at chosen inputs, following the approach in [20].

It is crucial to acknowledge, however, that constrained GP realizations might not strictly adhere to these constraints across all input values. This limitation arises because it is impractical to impose constraints universally (e.g., constraints are typically applied to discrete input points as shown in Equation 4.14 and 4.16). Nonetheless, by selecting a sufficiently dense array of constrained inputs—a methodology detailed subsequently—the resulting posterior distribution effectively conforms to the desired constraints for all practical intents and purposes [27].

The following steps aim to predict a convex function using the second-order derivative information.

Defining the matrix that will be used in computations:

$$\begin{aligned} A_1(\boldsymbol{x}, \boldsymbol{d}) =& \mathbf{K^{02}}(\boldsymbol{x}, \boldsymbol{d})\mathbf{K^{22}}(\boldsymbol{d}, \boldsymbol{d})^{-1} \\ A_2(x^\circ, \boldsymbol{d}) =& \mathbf{K^{02}}(x^\circ, \boldsymbol{d})\,\mathbf{K^{22}}(\boldsymbol{d}, \boldsymbol{d})^{-1} \\ B_1(\boldsymbol{x}, \boldsymbol{d}) =& \sigma^2\mathbf{I} + \mathbf{K}(\boldsymbol{x}, \boldsymbol{x}) \\ & - \mathbf{K^{02}}(\boldsymbol{x}, \boldsymbol{d})\mathbf{K^{22}}(\boldsymbol{d}, \boldsymbol{d})^{-1}\mathbf{K^{20}}(\boldsymbol{d}, \boldsymbol{x}) \\ B_2(x^\circ, \boldsymbol{d}) =& \mathbf{K}(x^\circ, x^\circ) \\ & - \mathbf{K^{02}}(x^\circ, \boldsymbol{d})\,\mathbf{K^{22}}(\boldsymbol{d}, \boldsymbol{d})^{-1}\mathbf{K^{20}}(\boldsymbol{d}, x^\circ) \\ B_3(\boldsymbol{x}, x^\circ, \boldsymbol{d}) =& \mathbf{K}(x^\circ, \boldsymbol{x}) \\ & - \mathbf{K^{02}}(x^\circ, \boldsymbol{d})\,\mathbf{K^{22}}(\boldsymbol{d}, \boldsymbol{d})^{-1}\mathbf{K^{20}}(\boldsymbol{d}, \boldsymbol{x}) \\ A(\boldsymbol{x}, x^\circ, \boldsymbol{d})) =& B_2(x^\circ, \boldsymbol{d}) \\ & - B_3(\boldsymbol{x}, x^\circ, \boldsymbol{d})\,B_1(\boldsymbol{x}, \boldsymbol{d})^{-1}B_3(\boldsymbol{x}, x^\circ, \boldsymbol{d})^\top, \end{aligned} \tag{4.18}$$

Assign to $f(.)$ a GP prior, and consider obtaining an estimated function that is L_U-smooth and γ_U-strongly convex, for a given $L_U > 0$ and $\gamma_U > 0$, suppose the goal is to predict $f(.)$ at a new set of n^* inputs points $\boldsymbol{x}^*$, given the current inputs $\boldsymbol{x}$ and resulting observations $\boldsymbol{y}$, according to Lemma 3.1 in [27], the joint distribution $(\boldsymbol{y}(x^*) \mid \boldsymbol{y}''(\boldsymbol{d}), \boldsymbol{x}^*, \boldsymbol{y}^*)$ is a GP with mean, covariance, and standard deviation given by [27], [20]:

$$\begin{aligned} &\bar{\mu}_{p^\circ}(x^*) = \mu\mathbf{1}_{p^\circ} + B_3(\boldsymbol{x}, x^*, \boldsymbol{d})\,B_1(\boldsymbol{x}, \boldsymbol{d})^{-1}\left(\boldsymbol{y}^* - \mu\mathbf{1}_p\right) \\ &+ \left(A_2(x^*, \boldsymbol{d}) - B_3(\boldsymbol{x}, x^*, \boldsymbol{d})\,B_1(\boldsymbol{x}, \boldsymbol{d})^{-1}A_1(\boldsymbol{x}, \boldsymbol{d})\right)\boldsymbol{y}''(\boldsymbol{d}) \\ &\bar{k}_{p^\circ}(x^*, x^{*\prime}) = A(\boldsymbol{x}, x^*, \boldsymbol{d}), \\ &\bar{\sigma}_{p^\circ}(x^*) = \sqrt{A(\boldsymbol{x}, x^*, \boldsymbol{d})}, \end{aligned} \tag{4.19}$$

Where μ is the original mean, and the posterior distribution of $(\boldsymbol{y}''(\boldsymbol{d}) \mid \boldsymbol{x}^*, \boldsymbol{y}^*)$ is given by:

$$(\boldsymbol{y}''(\boldsymbol{d}) \mid \boldsymbol{x}^*, \boldsymbol{y}^*) \propto \mathcal{N}(\boldsymbol{\mu}(\boldsymbol{d}), \mathbf{D}(\boldsymbol{d}, \boldsymbol{d}))\mathbf{1}_{\{\gamma_U \le U''(d_i) \le L_U, i=1,\dots,q\}} \tag{4.20}$$

Note that the inequality $\{\gamma_U \leq U''(d_i) \leq L_U, i = 1, \dots, q\}$ in Quation 4.20 represents the shape constraints for the objective function: L_U-smooth and γ_U-strongly convex, for a given $L_U > 0$ and $\gamma_U > 0$.

The mean and covariance in 4.20 are:

$$\begin{aligned} \boldsymbol{\mu}(\boldsymbol{d}) &= \mathbf{K}^{20}(\boldsymbol{d}, \boldsymbol{x}) \left(\sigma^2 \mathbf{I} + \mathbf{K}(\boldsymbol{x}, \boldsymbol{x})\right)^{-1} \left(\boldsymbol{z}_p - \mu \mathbf{1}_p\right) \\ \mathbf{D}(\boldsymbol{d}, \boldsymbol{d}) &= \mathbf{K}^{22}(\boldsymbol{d}, \boldsymbol{d}) \\ &\quad - \mathbf{K}^{20}(\boldsymbol{d}, \boldsymbol{x}) \left(\sigma^2 \mathbf{I} + \mathbf{K}(\boldsymbol{x}, \boldsymbol{x})\right)^{-1} \mathbf{K}^{02}(\boldsymbol{x}, \boldsymbol{d}) \end{aligned} \tag{4.21}$$

One crucial step in shape constrained Gaussian process is to estimate the truncated distribution $\boldsymbol{y}''(\boldsymbol{d})$ in equation 4.20 that satisfies the limit $\gamma_U \leq U''(d_i) \leq L_U, i = 1, \dots, q$. Here we use the Gibbs sampling procedure [27] to sample from the original second-order distribution $\boldsymbol{y}''(\boldsymbol{d})$ to establish the new distribution $y''(d)_{\gamma_U \leq U''(d_i) \leq L_U}$. In this method, the vector consisting of all the second-order derivatives of the selected constraint points $\boldsymbol{d}$ is treated as a multi-variable normal distribution $\boldsymbol{u}''(\boldsymbol{d}) = U''(d_i), i = 1, 2, \dots, q$. The basic idea of the Gibbs sampler is that the conditional density of a multi-variable normal distribution is multivariate normal again [**gibbs**].At each step j, the $i - th$ element can be calculated as follows [19] [27]:

$$E\left\{P\left(u_i''^{(j)} \mid \boldsymbol{u}_{-i}''^{(j)}\right)\right\} = \mu_i + \Sigma_{i,-i}\Sigma_{-i,-i}^{-1}\left(\boldsymbol{u}_{-i}''^{(j)} - \mu_{-i}\right) \tag{4.22}$$

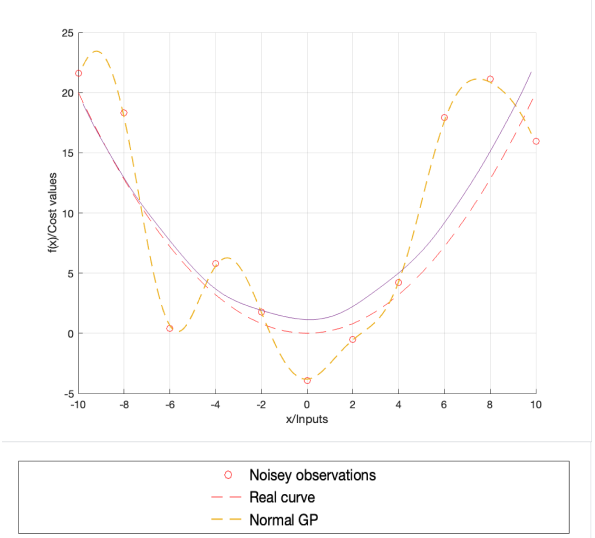
Where $\Sigma_{i,-i}$ is the i th row of the covariance matrix Σ without the entry from the i th column. The approach is to sample from the original second-order derivative $\boldsymbol{u}''(\boldsymbol{d})$ to form a new set of 2-order derivatives that fulfil the limit $\gamma_U \leq U''(d_i) \leq L_U, i = 1, \dots, q$.

Algorithm 1 Gibbs sampler to sample from the second order derivatives

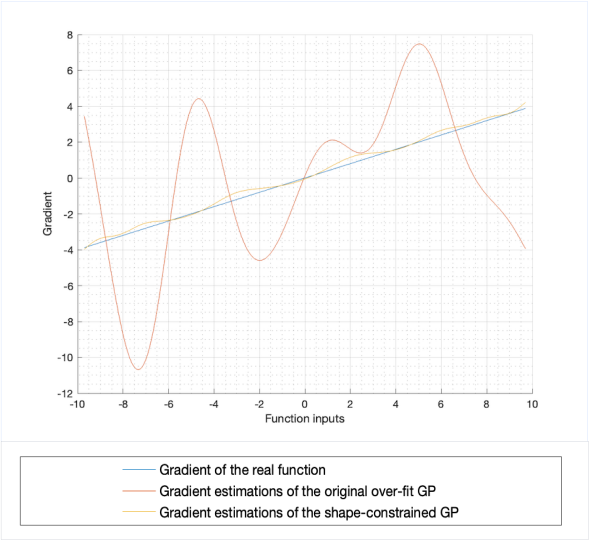
procedure Gibbs sampler for $\boldsymbol{u}''(\boldsymbol{d})$
  for `iteration` $j = 1, 2, 3, \dots$ **do**
    for `element` $U''(d_i) = 1, 2, 3, \dots, q$ **do**
      Calculate the new $U''^{(j+1)}(d_i)$ by $E\{P(U''^{(j)}(d_i) \mid \boldsymbol{u}''^{(j)}_{-i})\}$ Equation 4.22
      if $U''^{(j+1)}(d_i)$ `is inside the limit` $\gamma_U \leq U''(d_i) \leq L_U, i = 1, \dots, q$ **then**
        Pin it
      else if $U''^{(j+1)}(d_i)$ `is outside the limit` **then**
        Go to the next element $U''^{(j)}(d_{i+1})$
      end if
    end for
    if all $U''(d_i)$ are in the limit **then**
      break
    end if
  end for
end procedure

After completing several updates in algorithm 1, one can estimate the multivariate distribution that every element is within the limit.

The following results show the impact of the shape constraint:

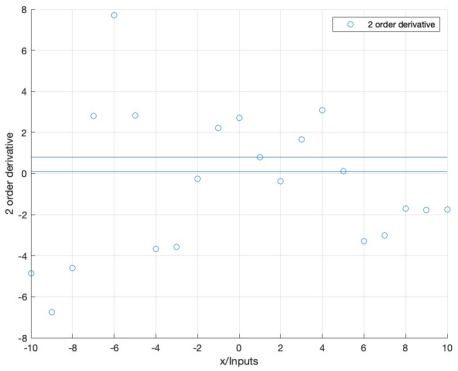


(a) Functions curves for the real functions, the original over-fit GP prediction, the shape-constrained GP prediction

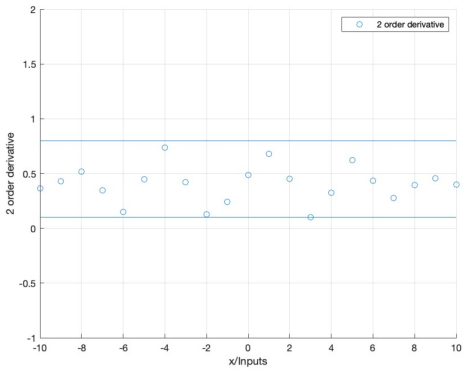


(b) The gradient and its estimations of the 3 function curves

Figure 4.2



(a) The original 2-order derivitive



(b) The new 2-order derivative

Figure 4.3

In Figure 4.2a, an over-fit prediction is illustrated by the orange dashed line, a result of employing a small length scale, $l_d = 1$, in the Gaussian process model. This over-fitting is further evidenced in Figure 4.3a, where it's observed that most of the second-order derivatives of the original vector fall outside the prescribed bounds $\gamma_U \leq U''(d_i) \leq L_U$. However, through multiple iterations of the Gibbs sampler, these derivatives are adjusted to fit within these limits, culminating in the acquisition of a strongly convex function, as depicted by the purple line in Figure 4.2a.

This phenomenon underpins our strategy of employing a consistently small length scale, l_d, within the Squared Exponential (SE) kernel function (referenced in 4.11) to initially generate an over-fit Gaussian prediction. Subsequent application of appropriate limit estimations through shape constraints ensures the derivation of a well-fitted model. This methodology allows us to bypass the computationally intensive step of maximal marginal likelihood hyper-parameter optimization typically required during each Gaussian process learning iteration.

The efficacy and outcomes of employing the Shape-constrained Gaussian Process will be thoroughly examined and discussed in Chapter 5.

5

Results and discussions

This chapter outlines various simulation results, delving into theories and discussions on the impact of different factors on the system. Our analysis employs the data and system specifications of a real-world power system, specifically the IEEE-85 [8], [31], to investigate the effectiveness and efficiency of the proposed algorithm and method.

The IEEE 85 node test feeder, comprising 85 buses and equipped with 8 power storage resources located at buses 2, 3, 4, 5, 6, 7, 84, and 85, serves as the network under investigation. The control variables within this network are defined as:

$$\mathbf{u} = \left\{P_{g,i}, Q_{g,i}\right\}, \text{ for } i \in \mathcal{G}, \mathcal{G} = 2,3,4,5,6,7,84,85 \tag{5.1}$$

For the purpose of testing the formulated controller, a dataset representing load demand is essential. Researchers reveal two prevalent methods for acquiring load data for numerical simulations: one involves utilizing actual electricity consumption datasets from a specified period [6], [22], while the other employs an artificially generated random sequence [5], [11]. This thesis opts for the latter approach, utilizing load demand data generated in Matlab for the subsequent simulations.

To model the variability in demands across the network's buses for our simulations, we generate random sequences that mirror realistic fluctuations in power consumption. The initial demand values for each bus are derived from the baseline data provided in the Matpower case file. Subsequently, for each time step beyond the first $(k > 1)$, the demands are dynamically updated according to the following equations:

$$P_{di}(k+1) = \left(1 + \alpha_{i,k}\right) P_{di}(k), \quad i \in n, k = 2,3,\ldots \; Q_{di}(k+1) = \left(1 + \beta_{i,k}\right) Q_{di}(k), \quad i \in n, k = 2,3,\ldots \tag{5.2}$$

This method ensures that the simulation reflects the operational constraints of the system, particularly the necessity for it to function within its linearization range. Empirical analysis, including insights drawn from actual data in studies such as [6] and [22], indicates that the variation in load demands between two consecutive sampling instances does not exceed 10%. Accordingly, the coefficients $\alpha_{i,k}$ and $\beta_{i,k}$, which dictate the rate of change in power demands, are randomized within the range of $[-0.1, 0.1]$. This approach provides a controlled yet variable environment to simulate realistic operational conditions. Figure 5.1 presents the set of power demand data which will be used in our simulations.

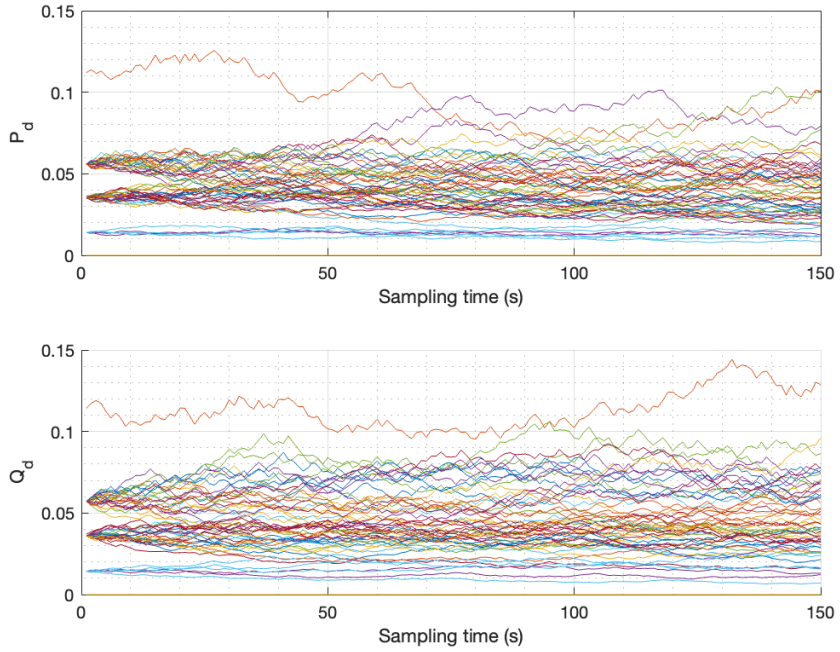


Figure 5.1: The active and reactive power $\boldsymbol{p_d}$ and $\boldsymbol{q_d}$ for all 85 buses used in the simulations

In aligning with methodologies outlined in [11] and [5], the objective function for this segment of our investigation is tailored to emphasize the costs associated with real and reactive power injections. Specifically, the objective function is constructed around the term $c_\phi \phi\left(\boldsymbol{p_g}, \boldsymbol{q_g}\right)$ as delineated in Equation 1.3, prioritizing this component while nullifying the influence of the other two terms, $c_\rho \rho(\boldsymbol{v})$ and $c_v v(\boldsymbol{v})$, by setting them to zero.

We use quadratic functions to represent the objective functions in the following form:

$$f_o = \sum_{i \in N} C_i \left(P_{i,g}^2 + Q_{i,g}^2\right) \tag{5.3}$$

This quadratic formulation effectively encapsulates the power injection costs, with terms for both the square and linear components of power injections by the generators within the network. Such a formulation is particularly advantageous in OPF problems where the primary goal is to minimize the total power injections, aligning with the studies set in [7], [11], and [5].

The system settings and limits are as:

$$\begin{aligned} &\overline{P_g} = 1MW, \quad \underline{P_g} = 0MW, \quad \overline{Q_g} = 1MWAr, \quad \underline{Q_g} = -1MWAr \\ &0.9 \leqslant |V_i| \leqslant 1.1 \text{ (p.u)}, -10 \leqslant \theta_i \leqslant 10 \text{ (degree)} \end{aligned} \tag{5.4}$$

The coefficients C_i in the quadratic objective function, as delineated in Equation 5.4, play a pivotal role in representing the relative impact associated with each generator's operation within the network. Essentially, these coefficients are indicative of the cost implications of utilizing a particular generator; a higher coefficient value signifies a higher cost for power generation, thereby guiding the optimization algorithm to allocate generation responsibilities across the network. For the purposes of our analysis, these parameters have been chosen arbitrarily, with the understanding that the control variables include both $P_{g,i}$ and $Q_{g,i}$ for each generator i within the set $\mathcal{G} = 2, 3, 4, 5, 6, 7, 84, 85$.

This arbitrary selection allows us to explore the impact of different cost structures on the overall optimization strategy and the resulting power flow solutions. It underscores the flexibility of the OPF

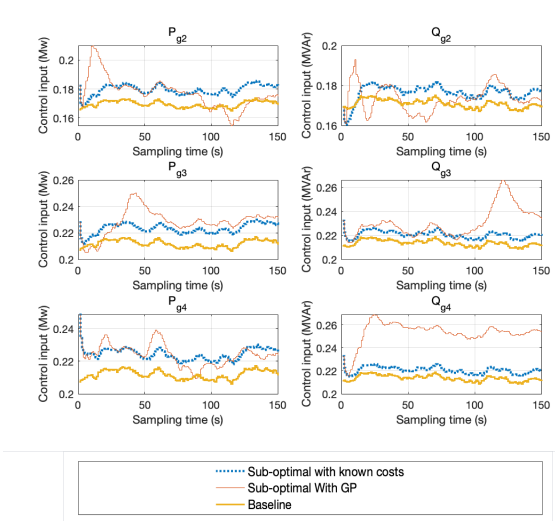
model in accommodating various scenarios and provides insights into how changes in the cost coefficients can influence the optimization outcome, prioritizing efficiency and cost-effectiveness in power distribution.

In our investigation, firstly, we start with a scenario characterized by low Gaussian noise, selecting the parameter α appropriately to illuminate the integrated system's performance under (relatively) optimal conditions. This initial setup allows us to showcase the potential of our algorithm in achieving superior control outcomes. The parameters for this phase of our analysis are specified as follows:

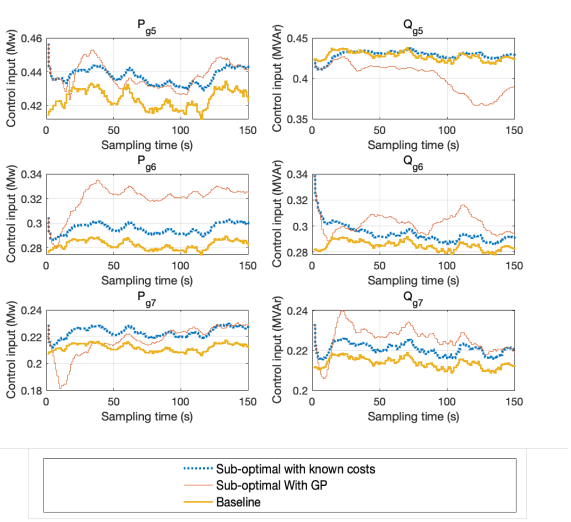
$$l_d = 1.5, \quad \sigma_n = 0.001, \quad \alpha = 0.4, \quad \gamma = \epsilon = 0.01 \tag{5.5}$$

To establish a benchmark for evaluating the effectiveness of the control inputs generated by our algorithm, we compute baselines reflecting the ideal scenario where complete information about the uncontrollable power demand at each bus is available as perfect predictions. This computation leverages the CVX optimization solver in Matlab, ensuring precision and reliability in the baseline establishment process.

With these carefully chosen parameters and baselines in place, we proceed to examine the simulation results derived from our system under study. These results, obtained under the aforementioned conditions, serve as a critical foundation for our subsequent analysis, wherein we will delve into the impacts of various factors that could potentially degrade performance. This structured approach enables a comprehensive understanding of the system's dynamics and the algorithm's robustness across different scenarios.

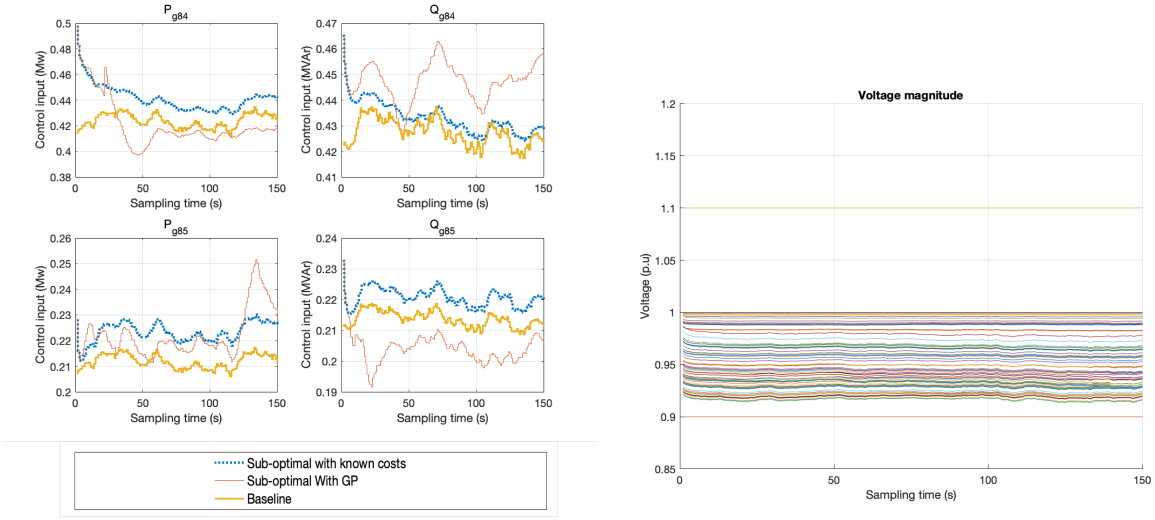


(a) Controller inputs for generators 2, 3, and 4



(b) Controller inputs for generators 5, 6 and 7

Figure 5.2: Controller inputs comparison



(a) Controller inputs for generator 84 and 85

(b) Voltage magnitude for all buses

Figure 5.3

An examination of the voltage measurements in Figure 5.3b across the system reveals a consistent adherence to voltage constraints, underscoring the effectiveness of both control strategies in maintaining system stability and operational integrity. To assess the efficacy of the algorithm developed in this thesis, we adopt a performance metric inspired by the methodologies outlined in [19] and [4]. Specifically, we utilize the 'Cost value error' as the overarching metric for evaluation:

$$f_o\left(\tilde{\boldsymbol{u}}^k\right) - f_o\left(\boldsymbol{u}^{*\boldsymbol{k}}\right) \tag{5.6}$$

Here, $f_o^k\left(\tilde{\boldsymbol{u}}^k\right)$ represents the total cost value associated with the sub-optimal solution $\tilde{\boldsymbol{u}}$ at time instant k, while $f_o\left(\boldsymbol{u}^{*,\boldsymbol{k}}\right)$ denotes the total cost value of the actual optimal solution (the baseline) $\boldsymbol{u}^*$, also at time k. This metric effectively quantifies the deviation of the algorithm's performance from the optimal scenario, providing a clear measure of the algorithm's accuracy and effectiveness over time.

With the system factors carefully selected and low noise conditions ensured, the metric yields insightful results under the best scenario, which are as follows:

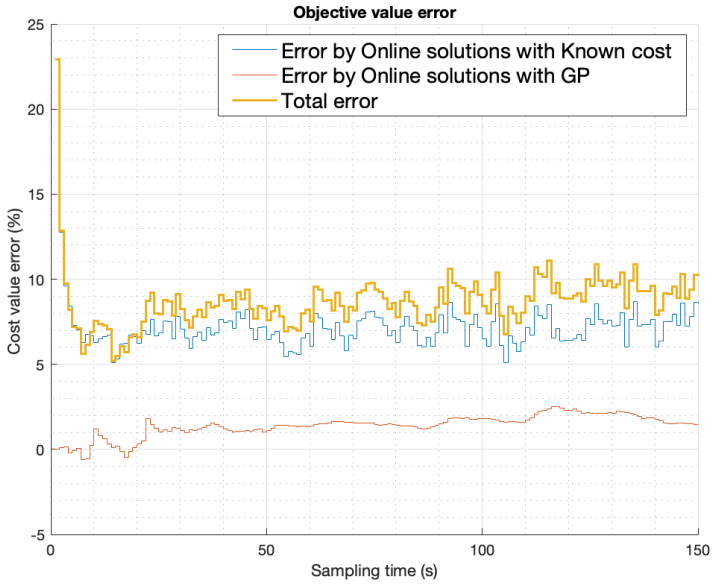


Figure 5.4: Cost value errors

Within the context of control variables, Figure 5.2 illustrates the performance of different control strategies. The blue dashed lines depict the control inputs computed by the Primal-Dual-Gradient-Projection online algorithm, which assumes known cost functions, whereas the orange thick lines represent the baseline for comparison. This comparison reveals that the Primal-Dual-Gradient-Projection online algorithm is capable of tracking closely with the actual optimal solution; figure 5.4 shows a clearer comparison that the discrepancies between the computed and optimal solutions diminish to nearly zero over time.

The cost value error, as depicted in Figure 5.4, exhibits a significant reduction at the outset, a result attributed to the relative large stepsize α employed in the initial stages of the algorithm. Despite subsequent fluctuations, the error stabilizes and consistently hovers around zero, aligning with our expectations for the algorithm's performance. This initial drop and subsequent stability underscore the algorithm's capacity to rapidly adjust towards an optimal solution, even in the presence of dynamic conditions.

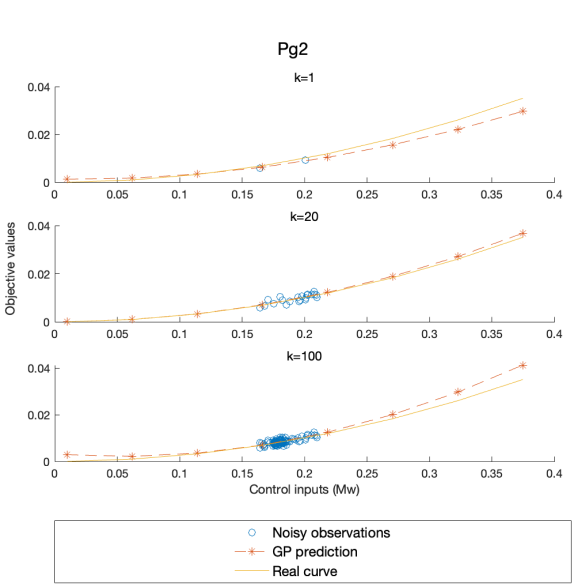
Moreover, Figure 5.4 further elucidates the sources of error within the system. Under conditions of low Gaussian noise, denoted as $\sigma_n = 0.001$, it becomes apparent that the primary source of the cost value error in the system's performance—represented by the blue line in Figure 5.4—accounts for approximately 6% to 8% of the total discrepancy. In contrast, the error attributable to the Gaussian Process, illustrated by the red line in Figure 5.4, constitutes a smaller fraction, ranging from 2% to 3%. This differentiation in error contribution highlights the relative impact of the online algorithm and the Gaussian Process in shaping the system's overall accuracy and reliability.

Furthermore, when incorporating the Shape-constrained Gaussian Process for the inference of unknown cost functions, the red lines in Figure 5.2 demonstrate the system's behaviour under this scenario. Despite some fluctuations, the control strategy leveraging the Shape-constrained Gaussian Process aligns with expectations, indicating a robust capacity to handle uncertainty in cost function parameters. The specifics regarding the metric of comparison and convergence performance of these approaches will be elaborated upon in subsequent discussions.

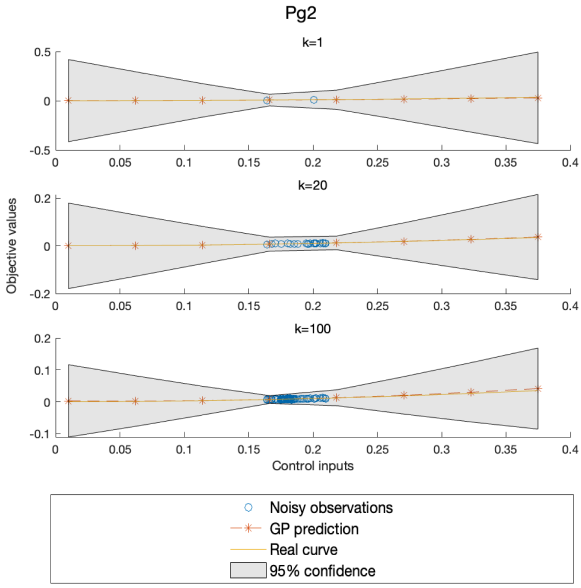
In our investigation of known function learning within the context of optimal power flow (OPF) control, we direct our focus towards understanding the cost functions associated with generators 2 and 5. These generators serve as primary examples to illustrate the methodology and effectiveness of our learning process. For the purposes of our simulations, we start with a scenario where only two noisy observations for each generator's cost function are available at the onset of the learning process.

This initial condition is designed to mimic the practical challenges often encountered in real-world systems, where complete and accurate information about cost functions may not be readily accessible. The presence of noise in these observations further replicates the uncertainty and measurement errors that typically characterize operational data in power systems. Through this simulation setup, we aim to demonstrate how our algorithm can effectively learn and adapt to the given cost functions, despite the limited and imperfect initial data.

The subsequent sections will delve into the specifics of the learning algorithm's performance, detailing how it navigates the challenges posed by sparse and noisy data to achieve accurate and reliable cost function estimations. This approach not only highlights the robustness of the learning process but also showcases its potential applicability in enhancing the efficiency and reliability of OPF solutions under realistic operational constraints.

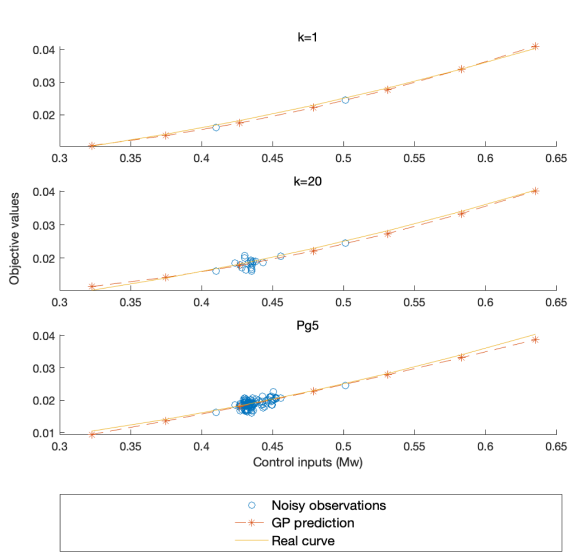


(a) Cost function learning for Generator 2

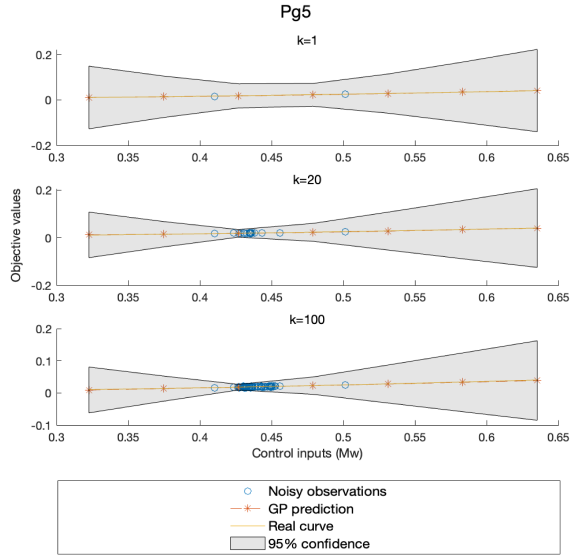


(b) Cost function learning for generator 2 with 95% confidence interval

Figure 5.5: Cost function learning for Generator 2 $P_{g,2}$ at sampling time $k = 1, 20 and 100$



(a) Cost function learning for Generator 5



(b) Cost function learning for generator 5 with 95% confidence interval

Figure 5.6: Cost function learning for Generator 5 $P_{g,5}$ at sampling time $k = 1, 20 and 100$

The core objective of the Shape-constrained Gaussian Process within our framework is the precise estimation of the cost function gradient, which plays a critical role in updating control variables. This estimation process is pivotal, as the accuracy of gradient estimation directly influences the effectiveness and efficiency of the controller's performance. The precise estimation enables a more informed and accurate adjustment of control variables, leading to improved system optimization and performance.

To elucidate the relationship between gradient estimation accuracy and controller performance, we conduct an in-depth analysis focusing on generators 2, 3, and 5. This analysis includes plotting both the estimated gradient values and their associated errors over time. Through this approach, we aim to provide a clear visual representation of how gradient estimation accuracy evolves throughout the control process and to quantify its impact on the overall system performance.

This examination not only sheds light on the technical capabilities of the Shape-constrained Gaussian Process in capturing and utilizing cost function gradients but also underscores the importance of accurate gradient information in the context of optimal power flow control. By analyzing the gradient estimation and its error dynamics for selected generators, we gain valuable insights into the strengths and potential areas for improvement within our control strategy.

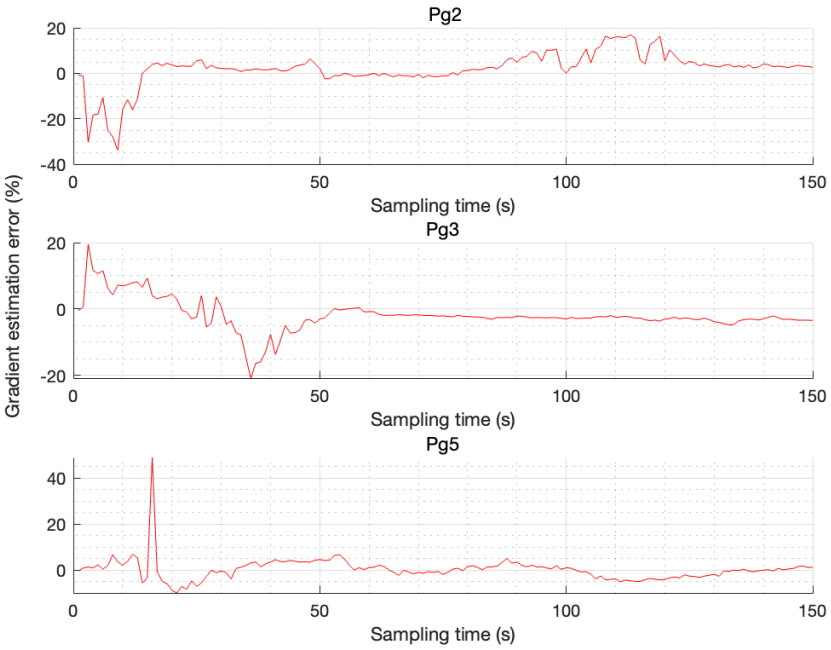


Figure 5.7: Cost function gradient estimation and its error for $P_{g,2}$, $P_{g,3}$ and $P_{g,5}$

An initial review of the simulation results, particularly with a focus on the early stages of the learning process, reveals that gradient estimation errors for the cost functions start off relatively large. This is clearly illustrated in Figure 5.7, where the initial phase is marked by significant errors in gradient estimation. Correspondingly, Figures 5.5b and 5.6b depict a high level of uncertainty (indicated by the grey shaded areas) at the outset of the learning process. This initial uncertainty underscores the challenges faced when beginning with a sparse dataset, in this case, only two observations.

Despite these early challenges, the imposition of shape constraints plays a pivotal role in guiding the learning process towards the expected outcomes. Even with limited initial observations, the learned prediction curves exhibit strong convexity, aligning with the anticipations of the online algorithm. This demonstrates the efficacy of shape constraints in steering the learning process in the desired direction, ensuring that the essential characteristics of the cost functions are preserved and accurately captured, even in the face of data scarcity.

As the learning process progresses, marked by the inclusion of additional observations into the training dataset, a notable improvement in the accuracy of gradient estimation is observed. Figure 5.7 highlights a consistent reduction in gradient estimation errors over time, with errors stabilizing within a minimal range around zero. This trend signifies the growing precision of the algorithm in capturing the true gradient of the cost functions as more data becomes available.

Moreover, the uncertainty associated with the Gaussian Process, as evidenced in Figures 5.5b and 5.6b, diminishes significantly with the accumulation of more observations. This reduction in uncertainty is particularly pronounced in regions of the function input range that witness a higher density of observations, such as the intervals $[0.4, 0.45]$ and $[0.15, 0.2]$ for generators 5 and 2, respectively. The decreased uncertainty in these figures underscores the Gaussian Process's ability to refine its predictions and reduce ambiguity as it is fed more data, enhancing the reliability and robustness of the learning outcomes.

5.1. Computation time

Addressing the challenge of insufficient computation time is essential; an evaluation of our algorithm's performance speed within the specified system is necessary. Simulations were conducted on a MacBook Pro equipped with 16GB of RAM and an Apple M2 chip:

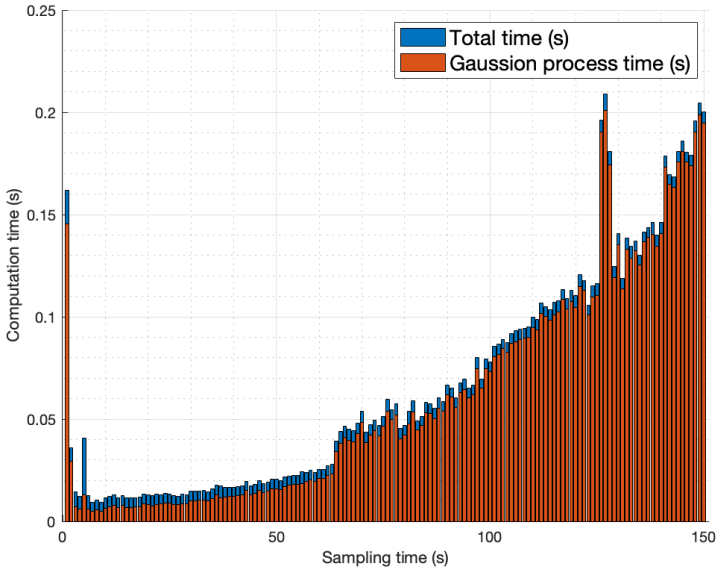


Figure 5.8: Computation time for the online OPF algorithm

Figure 5.8 shows that the time required for the online algorithm (the blue part in figure 5.8) remains similar during the whole program, while the computation time for the shape-constrained Gaussion procee increases. This is due to the more observations added into the training data. Consider that the sampling time in our system is 1 second, the keep increasing total computation time could lead to issues when there are more observations added. However, from figure 5.7 it can be seen that when there are about 100 or 150 observations available, ($k = 100 or k = 150$ in figure 5.7), the gradient estimation accuracy doesn't improve much with small errors of 5% to 10%. So the strategy here can be set to be, limit the total observations in the training data set, in this system, 200 for instant, and the gradient estimation with this amount of observations is accuary enough, while the total computation time only cost about 20% of the sampling time.

As depicted in Figure 5.8, the computation time attributed to the online algorithm component (illustrated by the blue section in Figure 5.8) remains relatively constant throughout the simulation. In contrast, the computation time associated with the shape-constrained Gaussian process escalates as the volume of training data observations increases. Given that our system operates with a sampling interval of 1 second, the cumulative increase in total computation time may pose challenges with the introduction of additional observations. However, as indicated in Figure 5.7, the gradient estimation accuracy stabilizes with minimal improvement beyond 100 to 150 observations, exhibiting small errors within the range of 5% to 10%. Consequently, a strategic approach involves capping the number of training data observations, for instance, at 200. This limitation ensures sufficient gradient estimation accuracy while maintaining the total computation time at approximately 20% of the sampling interval.

5.2. Numerical sensitivity analysis

5.2.1. Objective's smoothness constant

One pivotal factor that significantly impacts the convergence performance of optimization algorithms is the Lipschitz constant, denoted as L. A function is considered Lipschitz continuous if its variation is bounded by the one of an affine function with slope $L > 0$, known as the Lipschitz constant. The formal mathematical expression for this concept is provided as follows [2]:

$$|f(x_1) - f(x_2)| \leq L\,|x_1 - x_2| \tag{5.7}$$

Here, $f : \mathcal{R} \to \mathcal{R}$ represents a real-valued function. A function f is termed Lipschitz continuous if there exists a positive real constant L satisfying equation 5.7 for all real numbers x_1 and x_2. The assumption of Lipschitz continuity forms a cornerstone of the theoretical framework discussed in Chapter 3.

The Lipschitz constant L is crucial not merely as a foundational assumption but also for its significant influence on the asymptotic error bounds and, consequently, the convergence performance of the algorithm. It has been articulated in previous discussions that the asymptotic error bound is critically dependent on L, underscoring the importance of accurately determining or estimating this constant to ensure robust and efficient algorithmic performance.

The expression for the asymptotic tracking error in equation 3.15, 3.16 and 3.17, shows that an increase in the Lipschitz constant L consequently elevates the value of $L_{\nu,\epsilon}$ in Equation 3.17. This elevation directly influences $\rho(\alpha)$ in Equation 3.15, driving it closer to 1. As a result, the error bound delineated in equation 3.16 experiences an increase, provided that L does not reach a magnitude that propels $\rho(\alpha)$ beyond 1.

To elucidate this relationship with greater clarity, we have conducted an analysis focusing on the cost-value error across four distinct generators. Each generator is characterized by differing cost function coefficients(C_i in Equation 5.4 offering a varied perspective on how L and its associated parameters impact performance. The results of this analysis are presented graphically, showcasing the cost value error trends for each generator:

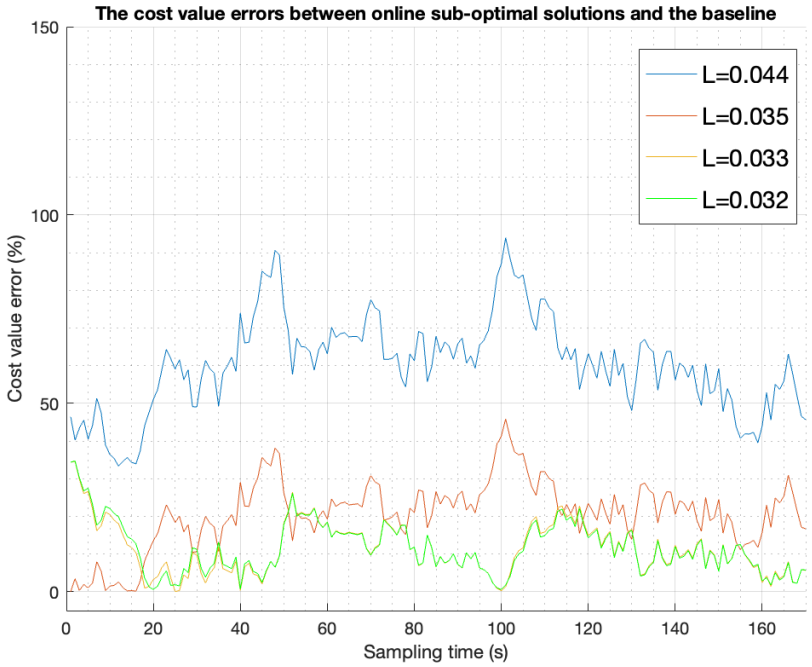


Figure 5.9: Cost value errors for 4 different cost functions and their L

In Figure 5.9, the observed trends corroborate our theoretical discussion on the influence of the Lipschitz constant L on cost value errors. Specifically, the blue line, representing the generator with the highest L exhibits the most significant error among all generators. Conversely, the green line, associated with the lowest L, demonstrates the ability to achieve the lowest cost value error, even starting from a high error level. This behaviour underscores the inverse relationship between L and the algorithm's performance in minimizing cost value errors. The errors associated with the other two generators display minimal variance, attributed to their similar L values, further validating the theoretical framework established in this section.

It is crucial to recognize the potential for instability when the Lipschitz constant L of any cost function becomes excessively large, pushing $\rho(\alpha)$ in Equation 3.15 above 1. Given that the cost functions in this study are intended to model air pollution levels as measured by sensors, the variation in hardware settings across different equipment providers can lead to significantly different L values. Consequently, raw data from sensors might yield cost functions with an L large enough to compromise controller stability.

In practical applications, conducting preliminary experiments and simulations is vital for identifying potentially 'dangerous' cost functions—those with an L so high that they risk destabilizing the controller. A practical solution involves applying a common normalization factor across all cost value measurements, effectively reducing L across all cost functions by scaling down the cost penalty coefficients C_i. This approach, akin to the regularization strategies discussed in earlier chapters, introduces a trade-off: while it enhances stability by lowering L, it may also perturb the original time-varying optimization problem, affecting the balance between stability and tracking accuracy.

5.2.2. Algorithm step size

The selection of the stepsize parameter α plays a critical role in optimizing the tracking performance of our controller. As delineated by Equations 3.15 and 3.16, an increased α value contributes to a reduction in $\rho(\alpha)$, which, in turn, leads to a decrease in the error bound. This relationship suggests that larger α values are generally favourable for enhancing tracking accuracy by minimizing the deviation from the optimal trajectory.

However, it is imperative to approach the selection of α with caution, as excessively large values can push $\rho(\alpha)$ beyond 1 (the optimal selection for convergence is $\alpha = \eta/L_{v,\epsilon}^2$ [6]), a condition that risks destabilizing the controller. This delicate balance between achieving optimal tracking performance and maintaining system stability underscores the importance of careful parameter tuning.

The theoretical considerations mentioned above find practical validation in our simulation results, which illustrate the impact of different α values on the system's behaviour:

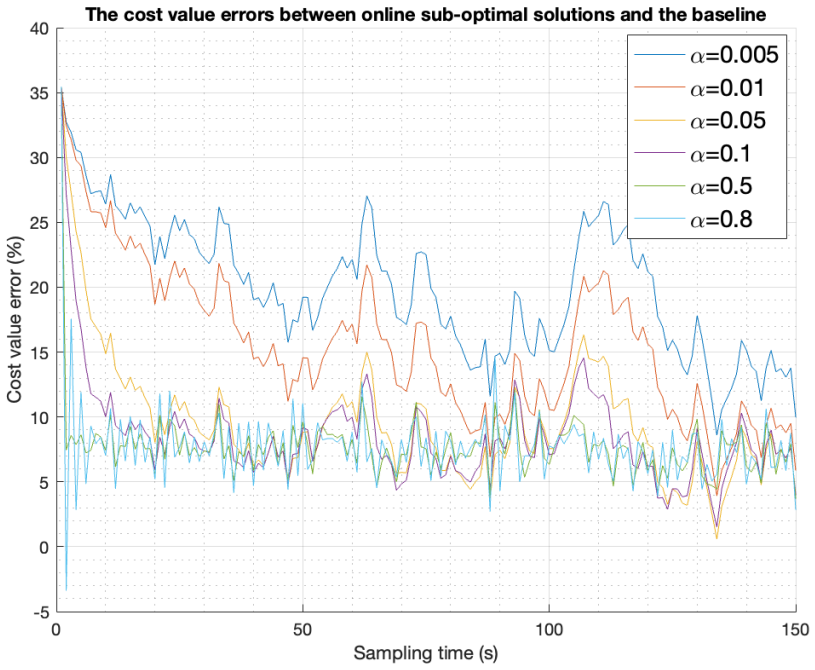


Figure 5.10: Cost value errors with different stepsize α

Incorporating a Gaussian Process (GP) into the system introduces additional considerations, particularly concerning the update equations:

$$
\begin{aligned}
\gamma_n^{k+1} &= \text{proj}_{\mathbb{R}_+}\left\{\gamma_n^k + \alpha\left(V^{\min} - y_n^k - \epsilon\gamma_n^k\right)\right\} \\
\mu_n^{k+1} &= \text{proj}_{\mathbb{R}_+}\left\{\mu_n^k + \alpha\left(y_n^k - V^{\max} - \epsilon\mu_n^k\right)\right\} \\
\boldsymbol{x}_i^{k+1} &= \text{proj}_{\mathcal{X}_i^k}\left\{\boldsymbol{x}_i^k - \alpha\nabla_{\boldsymbol{x}_i}\mathcal{L}_{v,\epsilon}^k(\boldsymbol{x},\boldsymbol{\gamma},\boldsymbol{\mu})\big|_{\boldsymbol{x}_i^k,\boldsymbol{\gamma}^k,\boldsymbol{\mu}^k}\right\}
\end{aligned}
\tag{5.8}
$$

A larger stepsize α increases the sub-optimal solution's sensitivity to gradient estimation errors. This sensitivity manifests as follows:

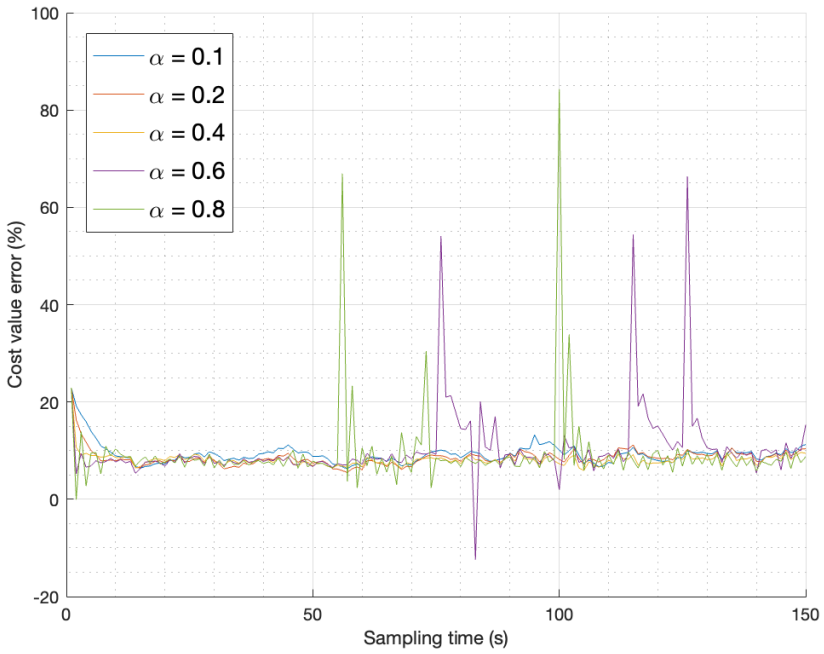


Figure 5.11: Cost value error caused by Gaussian process with different stepsize α

Figure 5.11 delineates the cost value error - the discrepancy between the sub-optimal solutions derived with the GP and those obtained with known cost functions. As α escalates to 0.6 and 0.8, the figure exhibits pronounced peaks. These peaks are indicative of situations where the gradient estimation from the GP deviates slightly, but the elevated value of α significantly exacerbates this discrepancy.

5.2.3. Noise variance

The critical role of gradient estimation in the update of control variables, as highlighted by Equation 5.11, underscores the necessity of accurately estimating the cost function. This estimation, when learned through a shape-constrained Gaussian process, is susceptible to Gaussian noise, potentially originating from equipment measurement inaccuracies, among other sources. Thus, a thorough examination of Gaussian noise's influence on convergence performance is essential.

In this study, a noise level represented by $\varepsilon \sim \mathcal{N}\left(0, \sigma_n^2\right)$ was employed, revealing that the error attributable to Gaussian noise at this magnitude is relatively insignificant when compared to the overall error. Subsequent analyses will explore the effects of elevated noise levels:

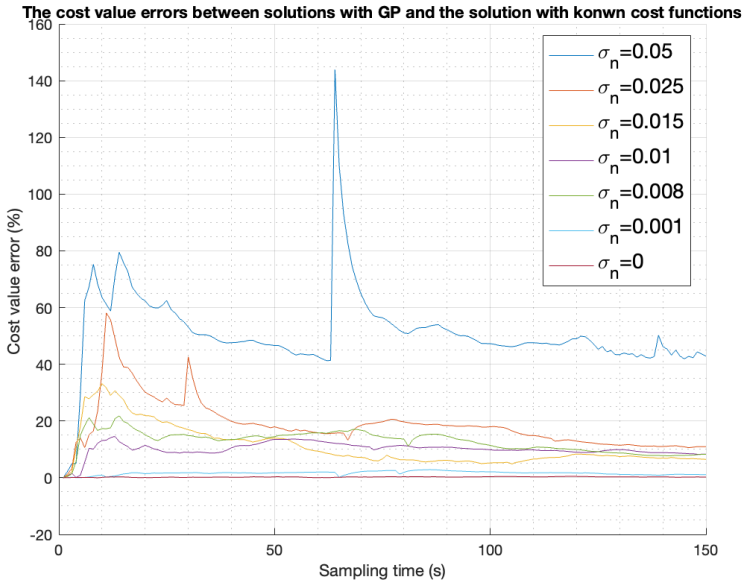
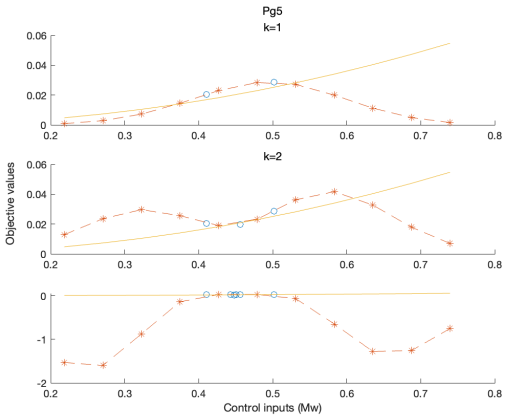


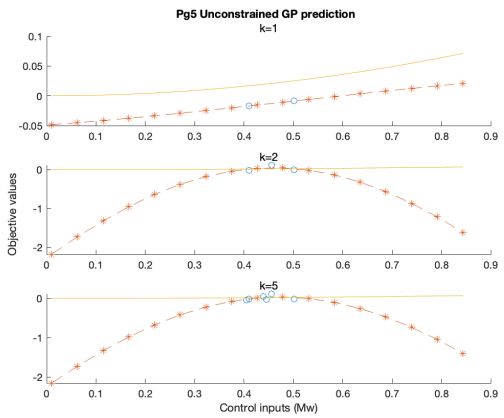
Figure 5.12: Cost value error by different levels of Gaussian noise

Observations from Figure 5.12 indicate that for small σ_n values, the error introduced by the Gaussian process remains inferior. Conversely, an escalation in noise levels proportionally amplifies the cost value error. Notably, all trends within Figure 5.12 converge towards a maximal error initially, eventually stabilizing near zero. This pattern is attributed to the more observations added to the training dataset, a phenomenon previously discussed in figures 5.5, 5.6, and 5.7.

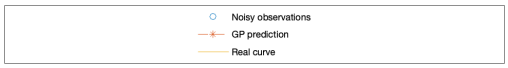
The significance of the shape constraint escalates with the increase in noise variance. Without this constraint, the learned function curve risks overfitting, like figure 5.13a shows, particularly when the Gaussian process's length-scale hyper-parameter l_n is intentionally minimized, bypassing the optimization of the maximal marginal likelihood hyper-parameter, as detailed in chapter 4.1. Moreover, even with an optimal length-scale l_n that facilitates accurate Gaussian predictions, the absence of a shape constraint might yield a concave learned function curve during the initial stages with limited observations, as illustrated in Figure 5.13b.



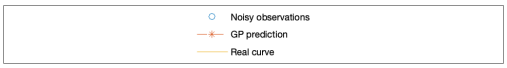
(a) function learning without shape constraint for $P_{g,5}$ with $\sigma_n = 0.005$



(b) A concave learnt function without shape constraint



(c)



(d)

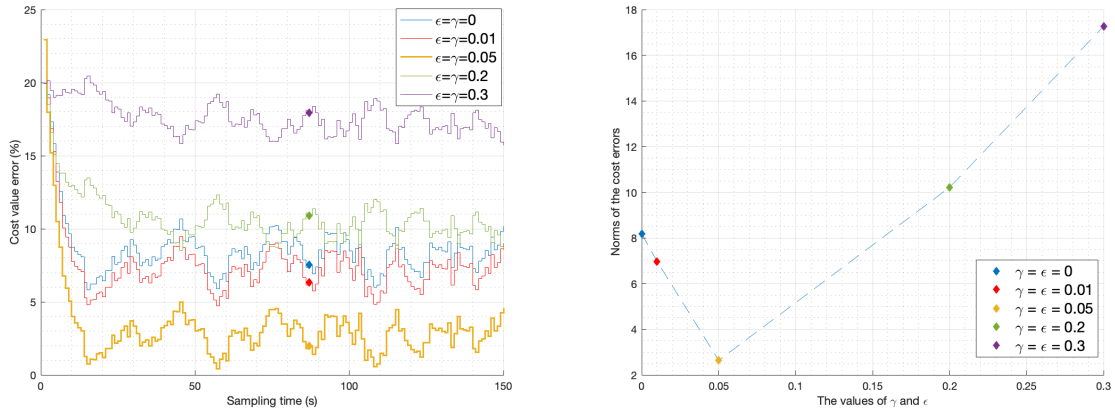
Figure 5.13: Function learning without shape constraint

In summary, the application of a shape constraint not only obviates the need for maximal marginal likelihood hyper-parameter optimization at each learning iteration but also enables precise convex function learning from the outset, with minimal available data. The estimation error diminishes rapidly as the dataset expands, highlighting the shape constraint's efficacy in enhancing model accuracy and convergence performance.

5.2.4. Regularization factors

The calibration of regulation factors ν and ϵ is critical, as their inappropriate magnitudes can adversely affect the solution to the original time-varying optimization problem. Numerical selection is essential, as larger values of ν and ϵ induce more significant perturbations due to the adjustments in the 'regulated' Lagrangian function. Conversely, these larger values contribute to reducing the asymptotic error, as evidenced by equations **??**, **??**, and **??**. In this context, a higher set of ν and ϵ results in a smaller $\rho(\alpha)$, given that $\eta = min(\nu, \epsilon)$. On the contrary, smaller values of ν and ϵ bring equation **??** closer to 1, leading to a larger asymptotic bound. Therefore, an optimal selection of the regulation factors ν and ϵ necessitates a balanced trade-off to minimize both perturbation error and asymptotic error.

This theory is illustrated by the following experimental results:



(a) Cost value error with different regulation factors ν and ϵ

(b) Norms of cost value error with different regulation factors

Figure 5.14: Cost value errors with different regulation factors ν and ϵ

Figures 5.14 and 5.14b demonstrate that as ν and ϵ increase from 0, the cost value errors initially decrease, reaching a minimum at $\nu = \epsilon = 0.05$ (indicated by the orange line/dot in figures 5.14 and 5.14b). This minimum primarily results from asymptotic error influences (equation **??**). Beyond this optimal point, further increases in ν and ϵ lead to an escalation in error, with perturbation effects introduced by the regulation becoming predominant.

5.3. Reference tracking

Taking practical considerations into account, the system's design was augmented to include reference tracking as a crucial component. Previously, we posited that the cost functions associated with each generator could be represented as quadratic functions, exemplified by the equation for bus 1:

$$f_{o,1} = c_{p,1}P_1^2 + c_{q,1}Q_1^2 \tag{5.9}$$

However, in real-world electricity distribution networks, the output from bus 1, also referred to as the reference bus, cannot be directly controlled. The terms P_1 and Q_1 usually represent the power supplied by a higher-level network, such as a transmission electricity system, where this power is often a predetermined value for a specific duration. Consequently, rather than treating P_1 and Q_1 as variables subject to control, we reinterpret them through the lens of reference tracking behavior:

$$f_0^{k'} = \beta\left(\left|P_1^k - P_{ref}^k\right|^2 + \left|Q_1^k - Q_{ref}^k\right|^2\right) + \sum_{i\in\mathcal{G}} C_i\left(\left|P_{g,i}^k\right|^2 + \left|Q_{g,i}^2\right|^2\right) \tag{5.10}$$

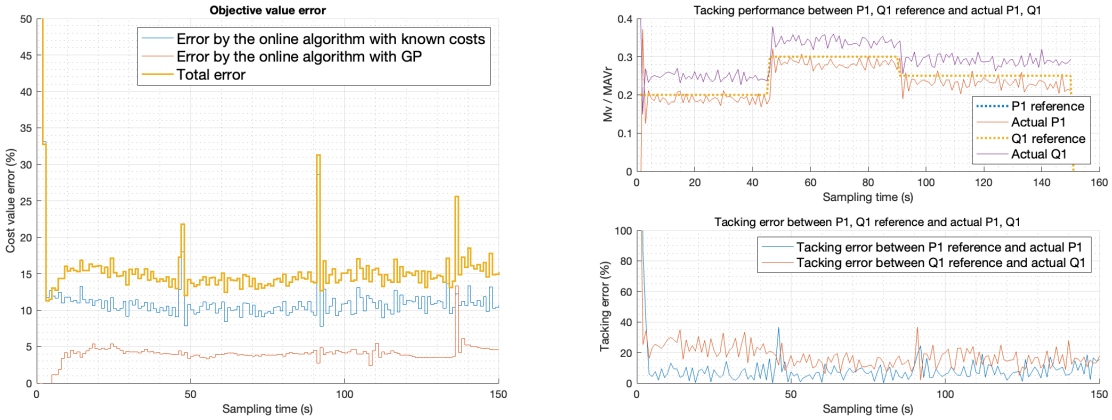
This modification underscores the necessity of adjusting the system to align with the inherent constraints and behaviors of actual power distribution frameworks. By shifting focus to reference tracking, the model better accommodates the operational realities where the reference bus's power output is a given, rather than a variable under our control.

Understanding the roles of P_1 and Q_1 is essential, as they can be calculated once the internal control variables, $P_{g,i}$ and $Q_{g,i}$, are defined. The power sourced from bus 1 is designed to equilibrate the entire system. In essence, once the network's control variables are established, P_1 and Q_1 become pivotal in soliciting the requisite power from the higher-level system to achieve system equilibrium. Consequently, a linear mapping from the control vector $\boldsymbol{u}$ to P_1 and Q_1 is feasible, given the system's linear model. With this framework, the revised cost function 5.10 is articulated as:

$$f_0^{k\prime} = \beta\left(\left|P_1\left(u^k\right) - P_{ref}^k\right|^2 + \left|Q_1\left(u^k\right) - Q_{ref}^k\right|^2\right) + \sum_{i=c} C_i\left(\left|P_{g,i}^k\right|^2 + \left|Q_{g,i}^2\right|^2\right) \tag{5.11}$$

Here, β denotes the penalty coefficient associated with reference tracking. The primary objective of this cost function is to optimize the utilization of power supplied by the higher-level system while concurrently minimizing the total costs associated with power injections within the network.

The simulation results, employing the revised cost function, are presented below:



(a) Cost value error for reference tracking

(b) Reference tracking error

Figure 5.15

Analysis of Figure 5.15a reveals cost value error spikes corresponding to abrupt changes in the reference signal. Despite these fluctuations, both the cost value error and the tracking error remain within acceptable bounds, demonstrating that the inclusion of reference tracking behavior does not significantly detriment the overall performance compared to simulations conducted without this behavior.

The determination of the penalty coefficient, β, was empirically based for this study. The analytical derivation of an optimal β value represents a promising avenue for future research, potentially leading to even more refined control strategies.

6

Conclusion and future work

This thesis presents the development and integration of an online optimization algorithm known as the Primal-Dual-Gradient-Projection (PDGP) method. This novel algorithm is designed to compute sub-optimal solutions that closely track the actual optimal solution with Q-linear convergence while requiring minimal computational time. A key innovation within this method is the introduction of a feedback strategy that bypasses the need for explicit information about power demands at each bus. Additionally, the incorporation of a shape-constrained Gaussian process enables the learning of unknown cost functions, further enhancing the system's efficiency.

The seamless integration of the PDGP algorithm, the feedback strategy, and the shape-constrained Gaussian process results in a complete system. Simulation results demonstrate commendable tracking performance, with a cost value error margin of 5% to 10%, and a computation time that is merely 20% of the sampling interval for a system comprising 85 buses. The study also delves into the impact of system design parameters, revealing that a larger Lipschitz constant L or Gaussian noise intensity results in a larger asymptotic error. Conversely, an increased step size contributes to reducing the asymptotic error. Furthermore, it is observed that regulation parameters that are either too large or too small can be detrimental; optimal parameter selection is attainable through empirical experimentation.

Nonetheless, this research is not without its limitations, which open avenues for future exploration. For instance, another performance metric known as 'dynamic regret'—which evaluates not only the instantaneous sub-optimal error but also incorporates the cumulative effect of all past performance—is also defined and tested by some OPF researchers. The influence of system design parameters on dynamic regret could prove to be substantial and needs further investigation.

Moreover, while heuristic guidelines for stepsize selection have been established, the determination of regulation parameters currently relies on trial-and-error. An analytical exploration into this aspect could streamline the process, leading to a more straightforward approach for identifying optimal regulatory parameters.

Finally, the methodologies developed and tested in this thesis are tailored specifically to electricity distribution networks. For transmission networks, where the choice of linearization points and other system configurations differ significantly, the applicability of our methods remains to be verified. Future work should include validating these methods within the context of transmission networks to ensure broader applicability.

Bibliography

[1] AA Abou El Ela, MA Abido, and SR Spea. “Optimal power flow using differential evolution algorithm”. In: *Electric Power Systems Research* 80.7 (2010), pp. 878–885.

[2] Yoav Benyamini and Joram Lindenstrauss. *Geometric nonlinear functional analysis*. Vol. 48. American Mathematical Soc., 1998.

[3] Andrey Bernstein, Emiliano Dall’Anese, and Andrea Simonetto. “Online primal-dual methods with measurement feedback for time-varying convex optimization”. In: *IEEE Transactions on Signal Processing* 67.8 (2019), pp. 1978–1991.

[4] Andrey Bernstein, Emiliano Dall’Anese, and Andrea Simonetto. “Online primal-dual methods with measurement feedback for time-varying convex optimization”. In: *IEEE Transactions on Signal Processing* 67.8 (2019), pp. 1978–1991.

[5] A.E. Chaib et al. “Optimal power flow with emission and non-smooth cost functions using backtracking search optimization algorithm”. In: *International Journal of Electrical Power Energy Systems* 81 (2016), pp. 64–77. issn: 0142-0615.

[6] Emiliano Dall’Anese. “Optimal power flow pursuit”. In: *2016 American Control Conference (ACC)*. 2016, pp. 1767–1767.

[7] Emiliano Dall’Anese, Sairaj V. Dhople, and Georgios B. Giannakis. “Optimal Dispatch of Photovoltaic Inverters in Residential Distribution Systems”. In: *IEEE Transactions on Sustainable Energy* 5.2 (2014), pp. 487–497.

[8] Do Das, DP Kothari, and A Kalam. “Simple and efficient method for load flow solution of radial distribution networks”. In: *International Journal of Electrical Power & Energy Systems* 17.5 (1995), pp. 335–346.

[9] Sairaj V. Dhople, Swaroop S. Guggilam, and Yu Christine Chen. “Linear approximations to AC power flow in rectangular coordinates”. In: *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. 2015, pp. 211–217.

[10] Masoud Farivar et al. “Optimal inverter VAR control in distribution systems with high PV penetration”. In: *2012 IEEE Power and Energy Society General Meeting*. 2012, pp. 1–7.

[11] Stephen Frank and Steffen Rebennack. “An introduction to optimal power flow: Theory, formulation, and examples”. In: *IIE Transactions* 48.12 (2016), pp. 1172–1197.

[12] Stephen Frank, Ingrida Steponavice, and Steffen Rebennack. “Optimal power flow: A bibliographic survey I: Formulations and deterministic methods”. In: *Energy systems* 3 (2012), pp. 221–258.

[13] Giovanni Franzese. *Intelligent Control System [Lecture slide].* 2021. url: `https://www.dcsc.tudelft.nl/`.

[14] Lingwen Gan and Steven H Low. “An online gradient algorithm for optimal power flow on radial networks”. In: *IEEE Journal on Selected Areas in Communications* 34.3 (2016), pp. 625–638.

[15] Swaroop S. Guggilam et al. “Scalable Optimization Methods for Distribution Networks With High PV Integration”. In: *IEEE Transactions on Smart Grid* 7.4 (2016), pp. 2061–2070.

[16] P Kessel and H Glavitsch. “Estimating the voltage stability of a power system”. In: *IEEE Transactions on power delivery* 1.3 (1986), pp. 346–354.

[17] Yun Liu et al. “Distributed real-time optimal power flow control in smart grid”. In: *IEEE Transactions on Power Systems* 32.5 (2016), pp. 3403–3414.

[18] Erfan Mohagheghi et al. “A survey of real-time optimal power flow”. In: *Energies* 11.11 (2018), p. 3142.

[19] Ana M Ospina, Andrea Simonetto, and Emiliano Dall'Anese. "Time-varying optimization of networked systems with human preferences". In: *IEEE Transactions on Control of Network Systems* 10.1 (2022), pp. 503–515.

[20] Ana M Ospina, Andrea Simonetto, and Emiliano Dall'Anese. "Personalized demand response via shape-constrained online learning". In: *2020 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE. 2020, pp. 1–6.

[21] William Rosehart, Claudio Cañizares, and Victor Quintana. "Optimal power flow incorporating voltage collapse constraints". In: *1999 IEEE Power Engineering Society Summer Meeting. Conference Proceedings*. Vol. 2. IEEE. 1999, pp. 820–825.

[22] Andrea Simonetto and Emiliano Dall'Anese. "Prediction-Correction Algorithms for Time-Varying Constrained Optimization". In: *IEEE Transactions on Signal Processing* 65.20 (2017), pp. 5481–5494.

[23] Andrea Simonetto and Emiliano Dall'Anese. "Prediction-correction algorithms for time-varying constrained optimization". In: *IEEE Transactions on Signal Processing* 65.20 (2017), pp. 5481–5494.

[24] Andrea Simonetto et al. "A class of prediction-correction methods for time-varying convex optimization". In: *IEEE Transactions on Signal Processing* 64.17 (2016), pp. 4576–4591.

[25] Andrea Simonetto et al. "Time-Varying Convex Optimization: Time-Structured Algorithms and Applications". In: *Proceedings of the IEEE* 108.11 (2020), pp. 2032–2048.

[26] Yujie Tang, Krishnamurthy Dvijotham, and Steven Low. "Real-Time Optimal Power Flow". In: *IEEE Transactions on Smart Grid* 8.6 (2017), pp. 2963–2973.

[27] Xiaojing Wang and James O Berger. "Estimating shape constrained functions using Gaussian processes". In: *SIAM/ASA Journal on Uncertainty Quantification* 4.1 (2016), pp. 1–25.

[28] Ermin Wei and Asuman Ozdaglar. "Distributed alternating direction method of multipliers". In: *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE. 2012, pp. 5445–5450.

[29] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*. Vol. 2. 3. MIT press Cambridge, MA, 2006.

[30] Yijian Zhang, Emiliano Dall'Anese, and Mingyi Hong. "Dynamic ADMM for real-time optimal power flow". In: *2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*. IEEE. 2017, pp. 1085–1089.

[31] Ray Daniel Zimmerman, Carlos Edmundo Murillo-Sánchez, and Robert John Thomas. "MATPOWER: Steady-State Operations, Planning, and Analysis Tools for Power Systems Research and Education". In: *IEEE Transactions on Power Systems* 26.1 (2011), pp. 12–19. doi: 10.1109/TPWRS.2010.2051168.