Multiple Constraints and Non-regular Solution in Deep Declarative Network

Suikei Wang

A thesis submitted for the degree of
Master of Machine Learning and Computer Vision
The Australian National University

November 2020

© Suikei Wang 2020

Except where otherwise indicated, this thesis is my own original work.

Suikei Wang
8 November 2020

to my parents, for their unconditional love

Suikei Wang – 8 November 2020

Acknowledgments

The past two years at the Australian National University have been an invaluable experience for me. When I started my Master of Machine Learning and Computer Vision at the beginning of 2019, I could barely understand lectures, knew little about the country, and had never heard of the term "convex optimization". It is unbelievable that I have been doing a research project on this topic for a whole year. ANU has the top tier research group in this realm and how honored I am to be a postgraduate student here.

First and foremost, I would like to express my deep and sincere gratitude to my supervisor, Stephen Gould. I knew Stephen before the admission of the program and how privileged I am to work and learn with one of the most brilliant minds in our field. He always has an insightful and high-level view on this topic. More importantly, Stephen is extremely kind and patient as he is always willing to discuss and share ideas with us during the weekly meetings. Although sometimes I stuck in some problems that he has given me constructive suggestions.

I would like to thank Dylan Campbell and Miaomiao Liu – another two giants of the Australian Centre for Robotic Vision – for offering me supervision and being my second examiner on my thesis. Dylan is always enthusiastic and knowledgeable, and his ideas and works inspire me a lot in my research. Although I have not worked with Miaomiao before, her works in the computer vision area are very impressive, which makes me work hard to complete this research. I want to thank her for her time examining my thesis seminar.

During my research, I have very fun collaboration and discussion with collaborators in our research group: Jiteng Ma, Jingyi Zhang, Yuchong Yao, Ruikai Cui, and Wenbo Du. Although we almost never meet in person due to the COVID-19, we still share many ideas related to our research through the social network. I greatly enjoyed the teamwork with them.

Outside the research group, I am very lucky to be surrounded by many great friends. Yao Liu, Ce Wang and Nuocheng Tian – We almost study together every single day during the academic year, even on the public holidays. I am impressed by their insightful ideas and I really appreciate the brainstorming with them. Mingrui Zhao – As my closest partner, a charming Engineering & Mathematics double-degree undergraduate student and my math "teacher", he always encourages me to pursue my academic ambition and I learned a lot from him.

I thank my parents: Shiwu Wang and Hongyin Shen. As traditional Asian parents, they never restrict my life planning, and always encourage me to fulfill my dream. I hope that they will a little proud of me.

Abstract

Constructing an end-to-end differentiable network to solve constrained optimization problems is one of the most elusive and long-standing challenges in Artificial Intelligence. In this thesis, we focus on the deep declarative network: a new class of end-to-end models with implicit defined differentiable nodes. Compared to traditional explicit defined neural networks, this general differentiable network constructs each layer as a constrained optimization problem.

We tackles the problem of multiple constraints nodes in the deep declarative network: how to build a general implicit differentiable network to solve constrained optimization problems and deal with the non-regular solution. On the one hand, adopting the solution of constrained optimization problems as the output of the node or layer provides a distinct pathway to implement the neural network. On the other hand, if we can obtain a heuristic solution or explore different optimality conditions for non-regular solution, it would be a crucial technology for more general and broad applications.

This thesis consists of two parts. In the first part, we aim to cover the essence of numerical optimization and present our efforts at implementing deep declarative nodes. More importantly, several examples of constrained optimization problems and their corresponding gradient solutions are provided. We also summarize recent advances in the differentiable network and discuss future research directions in the deep declarative network.

In the second part of this thesis, we investigate how to solve the non-regular solution in the deep declarative network. In particular, we proposed two different research directions: 1) how to approximate the heuristic solution for the overdetermined and underdetermined system; and 2) how to find the exact solution based on the different optimality conditions. We experimented and analyzed these ideas in solving non-regular points. We believe that applying these approaches to the deep declarative network holds great promises for future optimization technologies.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Constructing an end-to-end differentiable network to solve constrained optimization problems is one of the most elusive and long-standing challenges in Artificial Intelligence. Modern deep learning models have produced many insightful achievements in various domains with different functional layers and nodes. Most state-of-the-art network structures perform well in solving computer vision tasks [Simonyan and Zisserman, 2014; He et al., 2016; Sandler et al., 2018] such as image recognition, segmentation and reconstruction. These neural network models are constructed with parametrized processing layers [Gould et al., 2019], which pass the data through explicitly defined nodes. These nodes are usually composed of a differentiable forward function and the output is well-defined. Also, the input of these tasks is usually the image, which can be analyzed in matrix form and the transformation between layers is continuous. However, what if the problems have discrete logical relationships? How do we capture these constraints? Figure 1.1 demonstrates several 9 $\times$ 9 Sudoku puzzles and their corresponding solutions from the Kaggle competition dataset [Rao, 2019], where unknowns are zeros in the puzzles. Obviously, the relationship between each value in the puzzle is discrete, since all rows, columns and blocks are constrained with different rules. To optimize such logic-based puzzle games, many researchers have put decades of efforts to approach constrained optimization problems from various aspects, including:

- **Existence of optimizers.** The solution of the problem is defined based on the existence of optimizers, which requires our model to verify. For example, a normal 9 $\times$ 9 Sudoku puzzle has at least one solution satisfying all rules. That means if we transfer it to a mathematical constrained optimization problem, there is a global optimizer or multiple local optimizers.

- **Condition of the optimality.** The model should also understand if the solution satisfies both sufficient and necessary conditions of the optimality. In the 9 $\times$ 9 Sudoku puzzle, given a partially-filled grid of numbers, all remaining empty grid cells are to be filled with number 1 to 9. Each number must appear only once in each row, each column, and each of nine 3 $\times$ 3 subgrids. We should recognize the optimal point under these constraints in some specific conditions.

sudoku

Puzzle
070000043040009610800634900094052000358460020000800530080070091902100005007040802
301086504046521070500000001400800002080347900009050038004090200008734090007208103
048301560360008090910670003020000935509010200670020010004002107090100008150834029
008317000004205109000040070327160904901450000045700800030001060872604000416070080
040890630000136820800740519000467052450020700267010000520003400010280970004050063
561092730020780090900005046600000427010070003073000819035900670700103080000000050
310450900072986143906010508639178020150090806004003700005731009701829350000645010
Solution
679518243543729618821634957794352186358461729216897534485276391962183475137945862
371986524846521379592473861463819752285347916719652438634195287128734695957268143
748391562365248791912675483421786935589413276673529814834962157296157348157834629
298317645764285139153946278327168954981453726645792813539821467872634591416579382
142895637975136824836742519398467152451328796267519348529673481613284975784951263
561492738324786195987315246659831427418279563273564819135928674746153982892647351
318457962572986143946312578639178425157294836284563791425731689761829354893645217

Figure 1.1: A sample 9 $\times$ 9 Sudoku puzzle with corresponding solution from the Kaggle 9 Million Sudoku Puzzles and Solutions dataset [Rao, 2019]

- **Nodes in the neural network.** Further more, our model even need to solve more complex problems through adopting the approach as nodes in the neural network. Hence, some related functionality such as learning progress and back-propagation should be implemented. Also, the model needs to be flexible for different application tasks.

There are various recent interests in the differentiable neural network for solving the constrained optimization problem [Yang et al., 2017; Rocktäschel and Riedel, 2017; Wang et al., 2019; Amos and Kolter, 2017]. However, these approaches are targeting a specific problem with the special network structure. Most of them are also not end-to-end learnable models since they focus on the use of continuous parameters for integrating relationships. Is there a general and single end-to-end framework for solving the differentiable optimization in neural network nodes? We argue that the deep declarative network [Gould et al., 2019] – solving the constrained optimization problem – is a proper and efficient way to approach that. Just as we solve the constrained problem by hand, we believe that it can play the same role in determining the optimal solution. This is the theme of the first part of this thesis.

Let us take a closer look at the more complex problem. Figure 1.2 shows the possible moves of a Knight in chess, which is not unique and more difficult to determine the optimal solution. Also, the moves are constrained by the positions of other pieces. Therefore, the optimal solution is not guaranteed at this stage. We also have no idea

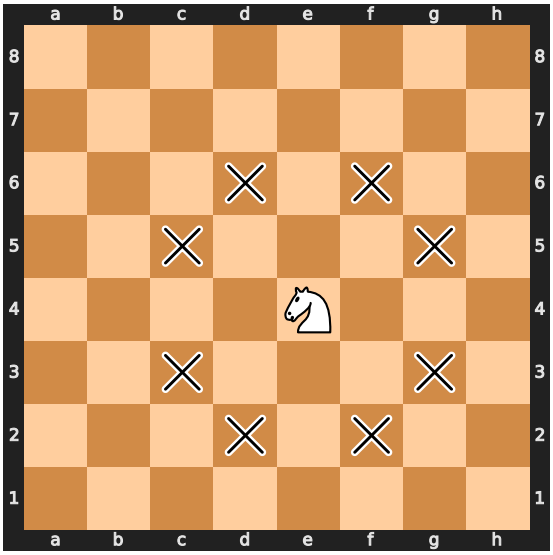

Figure 1.2: Moves of a Knight, produced from *chess* module in Python

what constraints determine the solution at each step. Is the feasible move restricted to a single point? Or do we have enough information to find the optimal solution? These are the questions we cared about, the regularity of the solution, which is also the theme of the second part of this thesis.

Therefore, in this thesis, we explore two research directions which employ the deep declarative network as a key component:

- **Multiple constrained declarative nodes** define a general form for different constrained optimization problems and aim to find the gradient solution for them as the output of the layer.
- **Non-regular solutions in deep declarative nodes** combine and analyze the solution to deal with non-regular points in the solution of the problem. For example, how to approximate the heuristic solution, or solve the constrained optimization problem under different optimality conditions.

1.2 Thesis Outline

Following the two central themes we mentioned above, this thesis consists of two parts – PART I Deep Declarative Network: Multiple Constrained Declarative Nodes and PART II Deep Declarative Nodes: Non-regular Solution.

PART I focus on the regular points of multiple constrained declarative nodes in the deep declarative network, with some basic examples of different constraints nodes.

In Chapter 2, we first give an overview of the theory of optimization, with the discussion of the optimality. Next, we formally define the unconstrained, equality

constrained and inequality constrained problems. We then discuss the related works on the solutions to these problems. Finally, the differentiable neural network, an application of numerical optimization in machine learning, is briefly discussed with various modern deep learning algorithms.

In Chapter 3, which is based on works of Gould et al. [2019], we present the details of the deep declarative network. We begin with the overview structure and how it works through the specific declarative nodes. We then introduce the learning progress of the deep declarative network. We analyze the backpropagation through the declarative nodes in different constrained problems. Finally, we give multiple examples of the implementation of the deep declarative nodes in both equality constrained and inequality constrained optimization problems. Also, we point out the limitation of current deep declarative nodes and address some foreseeable ideas for future improvements in the optimization process. We also look to the practical application of the deep declarative network in computer vision tasks.

PART II is an extension of the deep declarative nodes in non-regular solution problems, which cannot be solved through the traditional numerical optimization methods. Detailedly, we focus on the approximate solution of the non-regular points with different approaches, which can solve the problem more efficiently.

In Chapter 4, we give an overview of the non-regular solution problems. We list the general non-regular solution cases: overdetermined system, rand deficiency, and non-convex feasible set. We also introduce various related works regarding problems with non-regular gradient results.

In Chapter 5, we demonstrate various possible solutions for each non-regular point case. Since for non-regular solution problems, there are no exact solutions, we can only approximate the closed result. We also compare and discuss the results of each method on minimizing the final loss. Finally, we point out some future works for solving non-regular point problems.

We will finally present the conclusion in Chapter 6. Proofs for the important theorems and definitions are given in the Appendix.

1.3 Contribution

The contributions of this thesis are summarized as follows:

- We implemented the deep declarative nodes with equality and inequality constraints. In particular, we developed nodes for both linear and non-linear constraints based on regular solution points. Examples of each case are also given.
- We made effort to tackle the gradient solution for the non-regular point in declarative nodes. For the overdetermined system, rank deficient problems and non-convex cases, we adequately cover various insightful solutions with simple examples.

Part I

Deep Declarative Network: Multiple Constrained Declarative Nodes

Chapter 2

An Overview of Numerical Optimization

In this chapter, we aim to provide readers an overview of numerical optimization. We begin with the theory of optimization (Section 2.1), from the existence of optimizers, to the optimality conditions for both unconstrained and constrained problems with duality. This theoretical background of optimization provides a solid base for algorithm development.

We then formally define the optimization of unconstrained and constrained problems in Section 2.2 and describe the general regular solution for these problems based on the gradient calculation.

Next, we briefly discuss the differentiable optimization in the neural network, which is a novel end-to-end network structure involving optimization problems in each layer with and without constraints. (Section 2.3). Finally, we give a summary of the numerical optimization for solving unconstrained and constrained problems in Section 2.4.

2.1 Theory of Optimization

2.1.1 Existence of Optimizers

In optimization, a basic question is to determine the existence of a global optimizer. In this thesis, we focus on the optimizer in general minimization problems. Supposed we want to find the minimizer for a given function f. There are several sufficient conditions on f to guarantee the existence, and the optimizer falls in the feasible set of solutions. For a feasible set, some related definitions are following:

Definition 2.1. [Nocedal and Wright, 2006] A subset $\Omega \in \mathbb{R}^n$ is called

- *bounded* if there is a constant $R > 0$ such that $\|x\| \leq R$ for all $x \in \Omega$
- *closed* if the limit point of any convergent sequence in Ω always lies in Ω
- *compact* if any sequence $\{x_k\}$ in Ω contains a subsequence that converges to a point in Ω

The following result gives a characterization of compact sets in $\mathbb{R}$. When we find the minimum or maximum solution for the problem, there exist a lower or upper bound but not necessarily an optimal solution. Therefore, we have some additional requirements.

Firstly, we give the definition of compact sets in Lemma 2.2. [Oman, 2017] gives a brief proof.

Lemma 2.2 (Bolzano-Weierstrass theorem)**.** A subset Ω in $\mathbb{R}^n$ is *compact* if and only if it is bounded and closed.

We also assume that the function f is continuous and *coercive*, where *coercive* means "$+\infty$ at infinity". More precisely, $f(x) \to +\infty$ if $|x| \to +\infty$. [Nocedal and Wright, 2006] Then the problem can be restricted to a bounded set and existence of a global minimum x^* is guaranteed: a continuous function has a minimum on a compact set. This theorem is defined as follows and the proof is given in Appendix A.1.1.

Theorem 2.3. *[Nocedal and Wright, 2006] If f is a continuous function defined on a compact set Ω in $\mathbb{R}$, then f has a global minimizer x^* on Ω i.e. there exists $x^* \in \Omega$ such that $f(x^*) \leq f(x)$ for all $x \in \Omega$.*

More general, based on the definition of coercive function f, we can give following theorem. Proof is given in Appendix A.1.2.

Theorem 2.4. *[Nocedal and Wright, 2006] If $f : \mathbb{R}^n \to \mathbb{R}$ is a continuous coercive function, then f has at least one global minimizer.*

Theorem 2.4 requires the continuity of f which is slightly restrictive for applications. However, we can replace it by the lower semi-continuity of f which is a rather weaker condition.

Definition 2.5. [Nocedal and Wright, 2006] Let $f : \mathbb{R}^n \to \mathbb{R} \cup \{\pm\infty\}$. Then f is called *lower semi-continuous* at a point $x_0 \in \mathbb{R}^n$ if for any sequence $f(x_k)$ converging to x_0 here holds $f(x_0) \leq \lim_{k\to\infty} f(x_k)$. f is called *lower semi-continuous* if f is lower semi-continuous at every point.

Recall our assumptions on function f, it is a continuous function, which is always lower semi-continuous. Notably, lower semi-continuous functions are not necessarily continuous. For instance, a binary function equals to 0 when $x \leq 0$ and equals to 1 when $x > 0$ is not continuous at $x_0 = 0$. However, since it is greater than 0 for all x and $f(0) = 0$, we have $f(0) = 0 \leq \liminf_{x\to 0} f(x)$ and it is lower semi-continuous at $x_0 = 0$.

The theorem of the existence of the optimizer of lower semi-continuous function is given as follows and the proof is given in Appendix A.1.3

Theorem 2.6. *[Nocedal and Wright, 2006] Let $f : \mathbb{R}^n \to \mathbb{R}$ be a lower semi-continuous function. If f has a nonempty, compact sublevel set $D := \{x \in \mathbb{R}^n : f(x) \leq \alpha\}$, then f achieves a global minimizer on $\mathbb{R}$.*

Also, we introduce the definition of convex function and convex set which are important in regular optimization problems.

Definition 2.7. [Nocedal and Wright, 2006] A function f is convex when

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha)f(y) \quad \text{for all } x, y, \text{ and } \alpha \in]0,1[$$

A set $C \subset \mathbb{R}^n$ is convex when

$$\alpha x + (1-\alpha)y \in C \quad \text{for all } x, y \text{ in } C, \text{ and } \alpha \in]0,1[$$

The objective function of the optimization problem we discussed in this part is convex and regular, which means its gradient can be computed and the solution exists. However, although the existence of the optimizer is sufficient, for unconstrained and constrained problems, the optimality conditions are different. In the next two sections, we will give necessary and sufficient conditions for both these cases.

2.1.2 Optimality Conditions for Unconstrained Problems

Firstly, we consider the unconstrained minimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^n} f(x), \tag{2.1}$$

where f is the objective function on $\mathbb{R}^n$.

In order to determine the minimizer, it is important to understand what can happen at a minimizer, and at what conditions a point must be a minimizer. Now we have to recognize the optimal point. There are two necessary conditions and one sufficient condition given below [Nocedal and Wright, 2006]. The proof is given in Appendix A.1.4.

Theorem 2.8. *Necessary and Sufficient Conditions. Let $f : \Omega \to \mathbb{R}$ be a funcion defined on a set $\Omega \subset \mathbb{R}^n$ and let x^* be an interior point of Ω that is a local minimizer of f.*
Necessary conditions:

- *(NC1) If f is differentiable at x^*, then x^* is a critical point of f, i.e. $\nabla f(x^*) = 0$.*
- *(NC2) If f is twice continuous differentiable on Ω, then the Hessian $\nabla^2 f(x^*)$ is positive semidefinite.*

Sufficient condition (SC1): if x^* is such that $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite, then x^* is a local minimum. (i.e. $f(x) \geq f(x^*)$ for x close to x^*)

Any point satisfying (NC1) as the minimizer of f is called a *critical* or *stationary* point of f. If the objective function f is convex, (NC1) is also the sufficient condition for the global minimum of the solution.

Below is an example of unconstrained minimization problem. Supposed we have to determine the minimum of function

$$f(x, y) = x^4 - 4xy + y^4$$

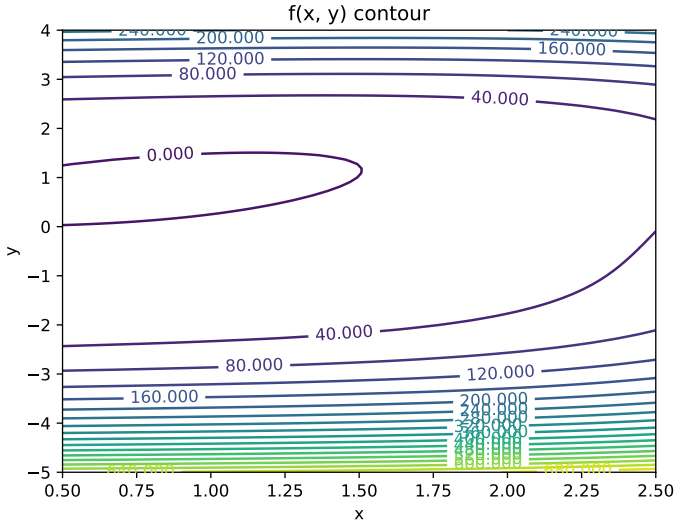

Figure 2.1: Contour Graph of $f(x,y) = x^4 - 4xy + y^4$

From the definition of function f, it is clear that f is continuous, then we can expand f by writing

$$f(x,y) = \left(x^4 + y^4\right)\left(1 - \frac{4xy}{x^4 + y^4}\right)$$

we can see f is coercive. Also, we show the contour graph of function f in Figure 2.1. Therefore f has global minimizers which are critical points. According to (NC1), we can find the global minimizer through solving the derivative of f equaling to zero:

$$0 = \nabla f(x,y) = \begin{pmatrix} 4x^3 - 4y \\ -4x + 4y^3 \end{pmatrix}$$

Thus, $y = x^3$ and $x = y^3$. Consequently $y = y^9$, i.e.

$$0 = y - y^9 = y\left(1 - y^8\right) = y\left(1 - y^4\right)\left(1 + y^4\right) = y(1-y)(1+y)\left(1 + y^2\right)\left(1 + y^4\right)$$

This implies $y = 0, 1, -1$. Thus f has three critical points $(0,0), (1,1), (-1,-1)$. Then we can evaluate f as these points since they may be local minimizer:

$$f(0,0) = 0, \quad f(1,1) = -2, \quad f(-1,-1) = -2$$

It achieves the same global minimum value on $(1,1)$ and $(-1,-1)$. Therefore, they are both global minimizers of f. Figure 2.2 shows the function $f(x,y)$ at these two optimal points.

From this example, we verify that we can find the global minimizer through (NC1). However, not all continuous functions with critical points have a global maxi-

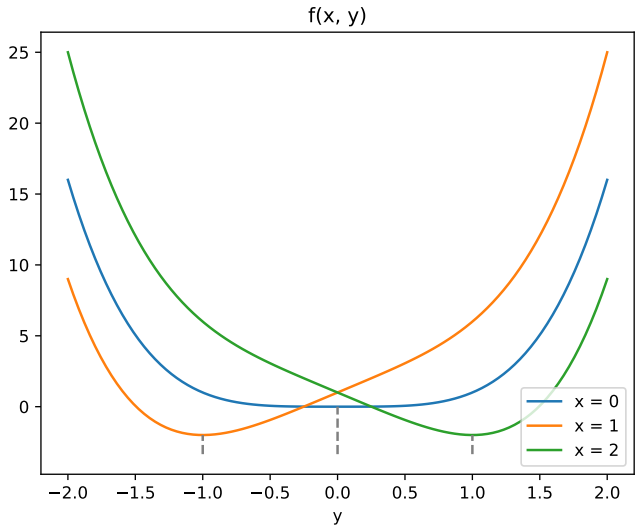

Figure 2.2: Function $f(x, y)$ at $x = 0$, $x = -1$ and $x = 1$

mizer or minimizer. If the function goes to infinity along its axes or a line, it does not have any maximizer or minimizer although it has a critical point, such as the cubic function. The condition of the minimizer as the critical point is that the function f should be a convex function with continuous first partial derivatives.

Let us move to the sufficient condition (SC1). The result obtained under this theorem is best possible for general functions. Specifically, for a convex function f that is defined on a convex set $\Omega \subset \mathbb{R}^n$, any local minimizer of f is also a global minimizer. Moreover, if a function f is strictly convex, it has at most one global minimizer.

2.1.3 Optimality Conditions for Constrained Problems

A general formulation for constrained optimization problems is as follows:

$$\begin{aligned} &\text{minimize } f(x) \\ &\text{subject to } \begin{cases} c_i(x) = 0 & \text{for } i = 1, \cdots, m_e, \\ c_i(x) \leq 0 & \text{for } i = m_e + 1, \cdots, m \end{cases} \end{aligned} \tag{2.2}$$

where f and c_i are smooth real-valued functions on $\mathbb{R}^n$, and m_e and m are nonnegative integers with $m_e < m$. We set

$$\mathscr{E} := \{1, \cdots, m_e\} \quad \text{and} \quad \mathscr{I} := \{m_e + 1, \cdots, m\}$$

as index sets of equality constraints and inequality constraints, respectively.

Here, f is so-called the objective function, and $c_i, i \in \mathscr{E}$ and $\mathscr{I}$ are equality constraints and inequality constraints respectively.

To solve the optimization problem (2.2), we define the feasible set of it to be

$$\mathscr{F} := \{x \in \mathbb{R}^n : c_i(x) = 0 \text{ for } i \in \mathscr{E} \text{ and } c_i(x) \leq 0 \text{ for } i \in \mathscr{I}\}$$

Any point $x \in \mathscr{F}$ is called a feasible point of (2.2) and we call (2.2) infeasible if $\mathscr{F} = 0$. Also, in this feasible set, a feasible point $x^* \in \mathscr{F}$ is called a local minimizer of (2.2) if it is the minimum solution in a neighborhood (strict local minimizer if it is the only one minimum solution). The definition of the global minimizer and strict global minimizer is similar, whose neighborhood is the whole feasible set.

Let us move to the constraints in this problem. For equality constraints, they are strictly equivalent. However, for inequality constraints, there are some exceptions. Let x^* be a local minimizer of (2.2). If there is an index $i \in \mathscr{I}$ such that $c_i(x^*) < 0$, then, x^* is still the local minimizer of the problem obtained by deleting i-th constraint. In this situation, we say that the i-th constraint is inactive at x^* since it does not have any effect on the solution. A general definition of active and inactive inequality constraints is as follows:

Definition 2.9. [Nocedal and Wright, 2006] At a feasible point $x \in \mathscr{F}$, the index $i \in \mathscr{I}$ is said to be *active* if $(x) = 0$ and *inactive* if $c_i(x) < 0$.

In the next chapter, we will give different processes for different cases of active or inactive inequality constraints in the deep declarative nodes. In this chapter, we only focus on the necessary and sufficient conditions for a feasible point x to be a local minimizer of (2.2). These conditions will be derived by considering the change of f on the feasible set along with certain directions. We give the lemma for the condition of local minimizer $x^* \in \mathscr{F}$ as follows, which can be proved through Taylor's formula in Appendix A.1.5.

Lemma 2.10. [Nocedal and Wright, 2006] If $x^* \in \mathscr{F}$ is a local minimizer of (2.2), then

$$d^T \nabla f(x^*) \geq 0 \quad \text{for all } d \in T_{x^*}\mathscr{F}$$

where $T_{x^*}\mathscr{F}$ is the set of all vectors tangent to $\mathscr{F}$.

However, we may not be able to extract useful results from this lemma, since $T_{x^*}\mathscr{F}$ depends only on the geometry of $\mathscr{F}$ but not on the constraints functions c_i. Not all local minimum falls on the boundary of the constraint function, which is a part of $T_{x^*}\mathscr{F}$. Therefore, it is necessary to introduce linearized feasible directions to give a characterization of $T_{x^*}\mathscr{F}$ in terms of c_i.

Definition 2.11. [Nocedal and Wright, 2006] Given $x \in \mathscr{F}$, we define

$$\text{LFD}(x) := \left\{d \in \mathbb{R}^n : d^T \nabla c_i(x) = 0 \text{ for } i \in \mathscr{E}; d^T \nabla c_i(x) \leq 0 \text{ for } i \in \mathscr{I} \cap \mathscr{A}(x)\right\}$$

and call it the set of linearized feasible directions of $\mathscr{F}$ at x.

Heuristically, for $i \in \mathscr{E}$ we should travel along directions d with $d^T \nabla c_i(x) = 0$ in order to stay on the curve $c_i(x) = 0$; for $i \in \mathscr{I}$ we should travel along directions with

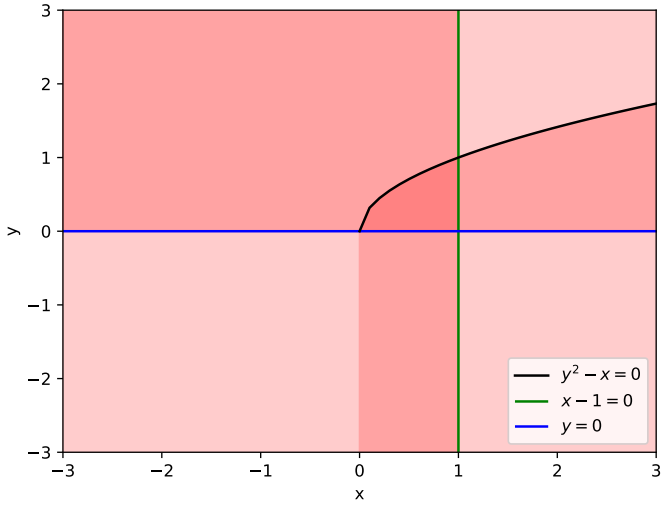

Figure 2.3: Feasible set of constraints c_1, c_2 and c_3

$d^T \nabla c_i(x) \leq 0$ in order to stay in the region $c_i(x) \leq 0$. Let us see an example of the linearized feasible directions and the tangent. Supposed we are considering a set $\mathscr{F}$ with variables $(x, y) \in \mathbb{R}^2$ and three inequality constraints functions:

$$\begin{aligned} c_1(x, y) &= x - 1 \leq 0 \\ c_2(x, y) &= -y \leq 0 \\ c_3(x, y) &= y^2 - x \leq 0 \end{aligned}$$

We can illustrate the feasible set of constriants c_1, c_2 and c_3 in Fig 2.3. The active set of $0 = (0, 0)$ is $\{2, 3\}$, since $c_1(0) = -1 < 0$, which is inactive. And we can get the derivative of c_2 and c_3 at 0:

$$\nabla c_2(0) = (0, -1)^T \quad \text{and} \quad \nabla c_3(0) = (-1, 0)^T$$

Then we have the linearized feasible directions on $x = 0$:

$$\begin{aligned} \text{LFD}(0) &= \left\{ d \in \mathbb{R}^2 : d^T \nabla c_2(0) \leq 0 \text{ and } d^T \nabla c_3(0) \leq 0 \right\} \\ &= \left\{ d \in \mathbb{R}^2 : d \geq 0 \right\} \end{aligned}$$

which equals the set of all vectors tangent to the feasible set $T_0 \mathscr{F}$.

Unlike the unconstrained optimization problem, the first order necessary condition of the existence of the optimizer is different since we should consider its linearized feasible directions and constraints feasibility. This is so-called the Karush-Kuhn-Tucker theorem:

Theorem 2.12 (Karush-Kuhn-Tucker Theorem)**.** *[Nocedal and Wright, 2006] Let* $x^* \in$

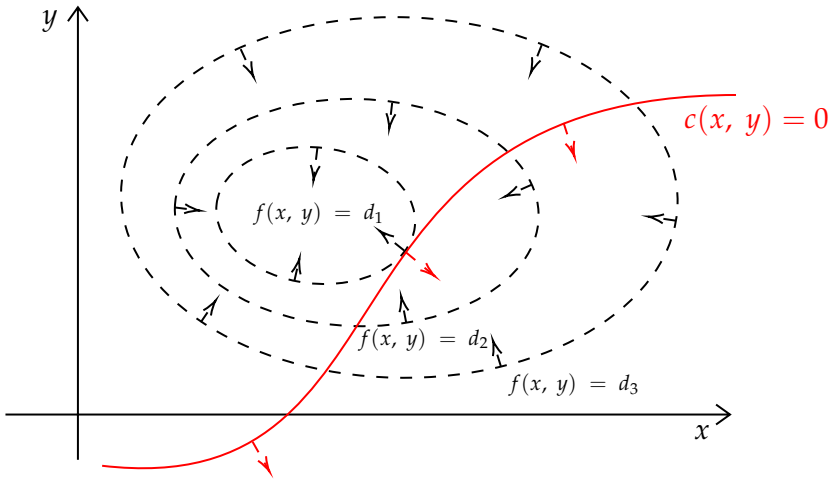

Figure 2.4: Contour of a constrained problem

$\mathscr{F}$ be a local minimizer of problem (2.2). If

$$T_{x^*}\mathscr{F} = \mathrm{LFD}(x^*),$$

then there exists $\lambda^* = (\lambda_1^*, \cdots, \lambda_m^*)^T \in \mathbb{R}^m$ such that

$$\nabla f(x^*) + \sum_{i \in \mathscr{E} \cup \mathscr{I}} \lambda_i^* \nabla c_i(x^*) = 0, \quad (\textit{Lagrangian stationary})$$

$$\left.\begin{array}{ll} c_i(x^*) = 0 & \textit{for all } i \in \mathscr{E}, \\ c_i(x^*) \leq 0 & \textit{for all } i \in \mathscr{I}, \end{array}\right\} \quad (\textit{primal feasibility})$$

$$\lambda_i^* \geq 0 \quad \textit{for alli} \in \mathscr{I}, \quad (\textit{dual feasibility})$$

$$\lambda_i^* c_i(x^*) = 0 \quad \textit{for alli} \in \mathscr{E} \cup \mathscr{I}. \quad (\textit{complementary slackness})$$

This set of equations are Karush-Kuhn-Tucker (KKT) conditions and a point x^* is called a KKT point if there exists λ^* such that (x^*, λ^*) satisfies the KKT conditions.

For constrained optimization problem, the classic solution is using Lagrange multipliers [Bertsekas, 2014]. Figure 2.4 shows an example of constrained problem, where $f(x,y) = d_i, i = 1,2,3$ are the contours of different solution of the objective function and $c(x,y) = 0$ is the equality constraint of the problem.

For multiple constraints, this method introduces the function

$$\mathscr{L}(x,\lambda) := f(x) + \sum_{i \in \mathscr{E} \cup \mathscr{I}} \lambda_i c_i(x)$$

which is called the Lagrange function. x is the primal variables and $\lambda_i, i = 1, \ldots, m$ are the Lagrange multipliers or the dual variables. According to the Lagrange multipliers method, we can solve this problem through the gradient of the Lagrange

function:

$$\nabla_x \mathscr{L}(x, \lambda) = \nabla f(x) + \sum_{i \in \mathscr{E} \cup \mathscr{I}} \lambda_i \nabla c_i(x)$$

Therefore, the first equation in KKT conditions can be written as

$$\nabla_x \mathscr{L}\left(x^*, \lambda^*\right) = 0$$

2.2 Solution of Unconstrained and Constrained Optimization Problems

According to the sufficient conditions for unconstrained optimization problems, we can easily compute the optimal solution through the first and second derivative of the objective function. For equality and inequality constrained problems, the introduction of Lagrangian $\mathcal{L}$ is useful for their closed-form solution. Gould et al. [2016] collected both argmin and argmax bi-level optimization results with and without constraints, which also provide insightful examples of these cases. Amos and Kolter [2017] also presents a solution for exact, constrained optimization within a neural network. In this thesis, we only focus on argmin problems, but the argmax problems have similar results.

In this section, we are going to provide some background for the solution of both unconstrained and constrained optimization problems, which is based on the gradient of the regular point.

2.2.1 Unconstrained Optimization

For unconstrained optimization problems, the solution is easy to obtain since we only need to focus on the optimality of the objective function. We consider an objective function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$:

$$y(x) \in \operatorname{argmin} f(x, y)$$

The derivative of $y(x)$ with respect to x is

$$\frac{dy(x)}{dx} = -[\frac{\partial^2 f}{\partial y(x)^2}]^{-1} \frac{\partial^2 f}{\partial x \partial y(x)} \tag{2.3}$$

which can be proved through differentiating and chain rule. [A.2.1]

A very classic example of the unconstrained minimization problem based on a closed convex nonempty set is the L2 norm $\|\cdot\|_2$. Let $\Omega \in \mathbb{R}^n$ be a closed convex nonempty set. For any $x \in \mathbb{R}^n$, the minimization problem is defined as follows:

$$\min_{y \in \Omega} \|y - x\|_2^2$$

This problem has a unique minimizer, which can be denoted by $P_\Omega(x)$, the Euclidean projection of x onto Ω.

Proof. Let $m := \inf_{y\in\Omega} \|y - x\|_2^2$. Since $\Omega \neq \emptyset$, we have $0 \leq m < \infty$. Let $\{y_k\} \subset \Omega$ be a minimizing sequence such that $\|y_k - x\|_2^2 \to m$ as $k \to \infty$. Thus $\|y_k - x\|_2^2 \leq m + 1$ for large k which implies that $\|y_k\|_2 \leq \|x\|_2 + \sqrt{m+1}$ for large k. Therefore $\{y_k\}$ is a bounded sequence. Consequently $\{y_k\}$ has a convegent subsequence $\{y_{k_l}\}$ with limit y^*. Since Ω is closed, we have $y^* \in \Omega$, Thus

$$m = \lim_{l\to\infty} \|y_{k_l} - x\|_2^2 = \|y^* - x\|_2^2$$

which means that m is achieved at y^*, i.e. the given minimization problem has a solution.

Next we show that the given minimization problem has a unique solution by contradiction. If the solution is not unique, let y_0 and y_1 be two distinct solutions. Then for $0 < t < 1$ we set $y_t = ty_1 + (1-t)y_1$. Since Ω is convex, we have $y_t \in \Omega$. Thus

$$\begin{aligned}
\|y_0 - x\|_2^2 = \|y_1 - x\|_2^2 \leq \|y_t - x\|_2^2 &= \|t\,(y_1 - x) + (1-t)\,(y_0 - x)\|_2^2 \\
&= t^2 \|y_1 - x\|_2^2 + (1-t)^2 \|y_0 - x\|_2^2 + 2t(1-t)\,\langle y_1 - x, y_0 - x\rangle \\
&= t\|y_1 - x\|_2^2 + (1-t)\|y_0 - x\|_2^2 - \left(t - t^2\right)\|y_1 - x\|_2^2 \\
&\quad - \left(1 - t - (1-t)^2\right)\|y_0 - x\|_2^2 + 2t(1-t)\,\langle y_1 - x, y_0 - x\rangle \\
&= t\|y_1 - x\|_2^2 + (1-t)\|y_0 - x\|_2^2 \\
&\quad - t(1-t)\left(\|y_1 - x\|_2^2 + \|y_0 - x\|_2^2 - 2\,\langle y_1 - x, y_0 - x\rangle\right) \\
&= \|y_0 - x\|_2^2 - t(1-t)\,\|y_1 - y_0\|_2^2
\end{aligned}$$

where $\langle\cdot,\cdot\rangle$ denotes the inner product on $\mathbb{R}^n$. Therefore $t(1-t)\,\|y_1 - y_0\|_2^2 \leq 0$ for $0 < t < 1$ and thus $\|y_1 - y_0\|_2^2 \leq 0$. So $y_1 = y_0$ which is a contradiction.

Overall, the minimization problem defined above has a unique minimizer. □

There are two most classical methods to solve the unconstrained optimization problem: Newton method [Newton and Colson, 1736] and the Method of Steepest Descent [Debye, 1909]. The former one, the Newton method starts from an initial guess x_0 and defines a sequence $\{x_k\}$ iteratively according to some rules. It uses the tangent line of the objective function f at x_k to replace f and uses the root of $L(x) = 0$, where $L(x)$ is the updated $f(x)$ as the next iterate x_{k+1}. Finally, the iteration is terminated as long as the difference between x_k and x_{k+1} less than a preassigned small number. The later one, steepest descent is a basic gradient method, which decreases the value of the objective function in a direction of most rapid change. The change rate of a function f at x in the direction u, a unit vector in $\mathbb{R}$ is determined by the directional derivative. Therefore, at x the value of f decrease fastest in the direction $u = -\nabla f(x)/\|\nabla f(x)\|$, which leads to the gradient method: we update the x through the direction with the step length.

2.2.2 Equality Constrained Optimization

Constrained problems are usually more complicated since the solution is restricted on a boundary or in a feasible region. For equality constraints, the basic case is the linear equality constraints $A\boldsymbol{y} = \boldsymbol{b}$. Again, we consider an objective function $f : \mathbb{R}^m \times \mathbb{R}^n \to \mathbb{R}$. Let $A \in \mathbb{R}^{p \times m}$ and $b \in \mathbb{R}^p$. A is a set of p linear equations as constraints $A\boldsymbol{y} = \boldsymbol{b}$. The problem is defined as follows:

$$\begin{aligned} y(x) \in \arg\min_{y \in \mathbb{R}^m} \quad & f(x, y) \\ \text{subject to} \quad & A\boldsymbol{y} = \boldsymbol{b} \end{aligned}$$

The derivative of $y(x)$ with respect to x is

$$\frac{dy(x)}{dx} = \left(H^{-1}A^T \left(AH^{-1}A^T \right)^{-1} AH^{-1} - H^{-1} \right) B \tag{2.4}$$

where $H = \partial^2 f(x, y)/\partial y(x)^2$ and $B = \partial^2 f(x, y)/\partial x \partial y(x)$.

The solution in 2.4 can be proved through the Lagrange multipliers [Bertsekas, 2014] in A.2.2. More generally, constraints can be non-linear. That means we cannot use A as a weight matrix for constrained parameters anymore. Therefore, we define the equality constraints problem using a set of m constraints functions $c(x, y)$:

$$\begin{aligned} y(x) \in \arg\min_{y \in \mathbb{R}^m} \quad & f(x, y) \\ \text{subject to} \quad & c_i(x, y) = 0, \quad i = 1, \ldots, m \end{aligned}$$

Solution for general multiple non-linear equality constraints is discussed in the chapter of deep declarative network nodes. Here, we give a simple example of non-linear equality constrained optimization problem.

For any given nonzero vector $y \in \mathbb{R}^n$, we define the minimization problem as follows:

$$\begin{aligned} \text{minimize} \quad & -x^T y \\ \text{subject to} \quad & \|x\|_2^2 = 1 \end{aligned}$$

From the constraint defined above, we can write the constraint function as $c(x) = \|x\|_2^2 - 1$ as illustrated it in Fig 2.5. Differentiating $c(x)$ with respect to x, we get $\nabla c(x) = x \neq 0$. Therefore, it follows the definition of LFD 2.11 and the theorem of KKT 2.12, which means that every local minimizer of this problem is a KKT point. Now we can write the Lagrangian function:

$$\mathcal{L}(x, \lambda) = -x^T y + \lambda \left(\|x\|_2^2 - 1 \right)$$

and the KKT conditions are:

$$\nabla_x \mathcal{L} = -y + 2\lambda x = 0, \quad \|x\|_2^2 = 1$$

From $-y + 2\lambda x = 0$ and $y \neq 0$ defined in the question, we must have $\lambda \neq 0$ and

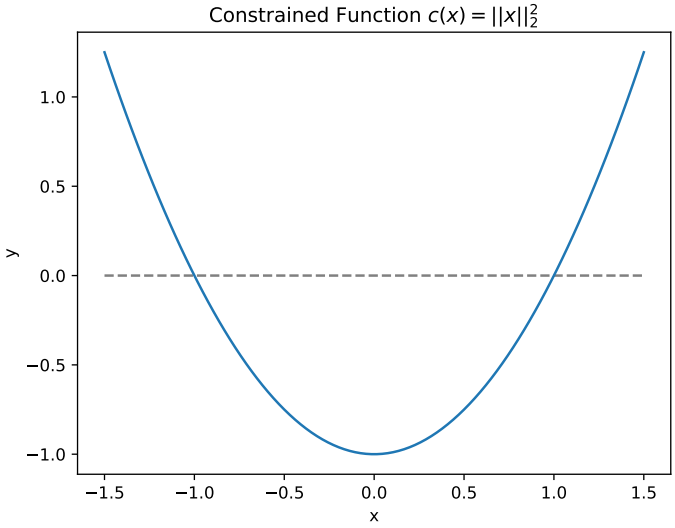

Figure 2.5: Constrained function $c(x) = \|x\|_2^2 - 1$

$x = y/2\lambda$. Combined with $\|x\|_2^2 = 1$, we get

$$4\lambda^2 = \|y\|_2^2 \Leftrightarrow \lambda = \pm\frac{\|y\|_2}{2}$$

Consequently, we have $x = \pm\frac{y}{\|y\|_2}$. For each x, we can compute its corresponding value of the objective function:

$$x = \frac{y}{\|y\|_2}, -x^T y = -\|y\|_2$$

$$x = -\frac{y}{\|y\|_2}, -x^T y = \|y\|_2$$

Obviously, the minimum is achieved with $f = -\|y\|_2$ at $x = \frac{y}{\|y\|_2}$.

Algorithms for solving constrained problems are various. For basic linear programming, which means that all functions involved are linear, we can transform it into standard form with matrix A, then solve the problem using Lagrangian function based on the KKT condition.

Penalty method[Yeniay, 2005], a function determining when a point x is feasible or not, is used to replace the constrained problem with an unconstrained one. For a minimization problem $f(x)$, the penalty function $P(x)$ associated with a penalty parameter are introduced to combine with $f(x)$ and now we are going to solve a series of unconstrained problems. These problems have converged solutions of the original constrained problem.

2.2.3 Inequality Constrained Optimization

Similar to equality constrained problems, inequality constrained problem usually defined the solution in a feasible set. In general, the standard form of inequality constrained problem is negative constraints:

$$\begin{aligned} y(x) \in \arg\min_{y \in \mathbb{R}^m} \quad & f(x,y) \\ \text{subject to} \quad & c_i(x,y) \leq 0, \quad i = 1, \ldots, m \end{aligned}$$

Solutions for inequality constrained problems are various. According to the properties of inequality constraints, active and inactive constraints have different criteria. We aim to find the gradient of the optimal solution, $f\prime(x)$, based on the inequality constrained argmin function. Panier et al. [1988] proposed a globally convergent algorithm for solving the minimization of smooth objective function based on smooth inequality constraints. This algorithm is based on the Quasi-Newton [Dennis and Moré, 1977] iteration for the solution of the first order condition of the optimality in KKT. An updated verision of this algorithm, a new QP-free method demonstrated by Qi and Qi [2000], emphasizes the feasibility of all iterates. It reformulates the KKT optimality condition Fischer–Burmeister function [Jiang, 1999] for nonlinear complementarity problems. The classical solution is still based on the Lagrange multipliers. Bertsekas [2014] proposed that for one-sided inequality constrained problems, it cannot be converted to equality constrained problem. Therefore, it introduced the method minimizing the augumented Lagrangian with respect to x for various value of the Lagrange parameters, which is presented by Powell [1969] and Hestenes [1969]:

$$\begin{aligned} \bar{L}_c(x,z,\lambda,\mu) =& f(x) + \lambda' h(x) + \frac{1}{2} c |h(x)|^2 \\ & + \sum_{j=1}^{r} \left\{ \mu_j \left[g_j(x) + z_j^2 \right] + \frac{1}{2} c \left| g_j(x) + z_j^2 \right|^2 \right\} \end{aligned}$$

The minimization of this augumented Lagrangian can be found through computing the first order derivative with respect to z explicitly for each fixed x.

More recently, Gould et al. [2016] introduced a method approximating the gradient of the inequality constrained problem based on ideas from interior-point methods [Boyd et al., 2004]. It gives a demonstration of log-barrier function, which transforms the original constrained problem into a unconstrained minimization problem.

$$\phi(x,y) = \sum_{i=1}^{m} \log\left(-c_i(x,y)\right) \tag{2.5}$$

$$\text{minimize}_y \quad t f(x,y) - \sum_{i=1}^{m} \log\left(-c_i(x,y)\right) \tag{2.6}$$

Equation 2.5 is the log-barrier function, which takes the sum of the logrithm for all constraints. Then substracting it in the unconstrained minimization problem approximates the original inequality constrained problem. t in Equation 2.6 is a scaling

factor for duality gap control if the solution set is convex.

Similar to the solution for unconstrained and equality constrained problem, minimizing Equation 2.6 is based on the gradient and hessian of the log-barrier function. Therefore, we can compute the approximation of the inequality constrained objective function.

2.3 Differentiable Neural Network

If a problem is differentiable, then the solution of this problem can be back-propagated. In neural networks, back-propagation [Goodfellow et al., 2016] is widely used to train the feedforward neural network, especially for supervised learning. Therefore, in deep neural networks, we can treat constrained optimization as an individual layer. Recently, there are several works on end-to-end differentiable convex optimization in the neural network, since this type of layer provides inductive bias for different problems, which is very practical.

OptNet [Amos and Kolter, 2017] is a very classical differentiable layer neural network. Each layer in the end-to-end deep neural network is intergraded into optimization problems, which can capture and encode complex dependencies and constraints between hidden variables. It specifically considers the quadratic programs, which are general convex optimization problems. Similarly, SATNet [Wang et al., 2019], a differentiable maximum satisfiability solver, is also intergraded into end-to-end deep learning systems. Besides, it combines the solver with the traditional convolutional network. Both OptNet and SATNet are applied to solving the Sudoku puzzles, which is a very basic constrained logical problem. To make it more general and efficient, Agrawal et al. [2019] demonstrate an approach based on disciplined convex programs, which is a subclass of the classical convex optimization problems. The affine map introduced in this paper represents the disciplined parametrized program.

A popular application in the differentiable network is the Perspective-n-Points (PnP) solver. Chen et al. [2020] present BPnP based on PnP solver, performing geometric optimization in computer vision tasks. It back-propagates gradient through PnP accurately and effectively since there is a differentiable function in the optimizer block. Besides, for blind PnP problems in the 3D computer vision task, Campbell et al. [2020] propose an end-to-end network based on the differentiating optimization solutions, which is robust and outperforming.

Apart from the above, the differentiable neural network has many practical and powerful applications. Amos and Yarats [2019] introduce a differentiable variant cross-entropy method for non-convex optimization objective function. Again, due to the differentiable feature of the network, the output of the cross-entropy method is differentiable with respect to the parameters in the objective function, even it is non-convex. In 3D reconstruction tasks, some implicit shape and texture are difficult to represent. Hence, Niemeyer et al. [2020] introduce a differentiable rendering formulation, which makes the network learn them from input images directly since implicit differentiation can learn the depth gradients. Also, some research has been

done to simplify the differentiable neural network since the computational cost and complexity of the differential operators can be very high in different tasks. The architecture proposed by Chen and Duvenaud [2019] is cheap and efficient, which sets the Jacobian matrix into diagonal and hollow. It also changes the backward progress into automatic differentiation, which is more effective and lightweight.

2.4 Summary

In Chapter 2, we first briefly introduce the numerical optimization with some necessary conditions and theorems. For the general convex optimization problem, the existence of the local or global optimizers can be determined by the feasible set. Next, the optimality of both unconstrained and constrained problems are discussed. For unconstrained problems, we only need to follow the necessary and sufficient conditions to find the global minimum of the solution. For constrained problems, it should also satisfy the KKT condition. Then we compare existing algorithms on the solution to these problems. Here, for constrained optimization problems, we have to consider the linearity of constraints. Specifically, the activity of inequality constraints can also be solved with different algorithms. Finally, since this thesis is based on the end-to-end differentiable network, some related works of the application are described since these works inspire the deep declarative network in the next chapter. In the next chapter, we are going to describe the deep declarative network in detail with examples.

Chapter 3

Deep Declarative Network

In this chapter, we will cover the structure and nodes in the deep declarative network: from its learning process to the back-propagation.

Before delving into the details of the back-propagation in different constraints cases, we give an overview of the deep declarative network in Section 3.1. In particular, the basic structure of the network and the details of declarative nodes are described according to Gould et al. [2019]. The learning progress of the network is also given. We hope this will give readers a better sense of what is the deep declarative network and how it works.

In Section 3.2, we present the details of the back-propagation in different constrained problems. The gradient computation results are based on the implicit differentiation and different in constrained problems. We discuss this part based on the regular solution and compare it with the general solution in the previous chapter.

Next we present the examples of constrained optimization problems with both linear and non-linear, equality and inequality constraints in Section 3.3. We also provide more implementation details of the deep declarative nodes.

Finally, we summarize the deep declarative network and its solution in different constrained problems under the regular point.

3.1 An Overview of Deep Declarative Network

3.1.1 Declarative Node

In deep declarative network, it defines the solution of a constrained optimization problem with parameter $x \in \mathbb{R}^n$ as the output of each node $y \in \mathbb{R}^m$. The general optimization problem can be defined as

$$y \in \underset{u \in C}{\arg\min} f(x, u) \tag{3.1}$$

where f is the objective function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, and $C \in \mathbb{R}^m$ is the set of constraints parameterized by x.

Apart from the traditional forward processing mapping node, deep declarative node does not explicitly define the transforming function from the input to the out-

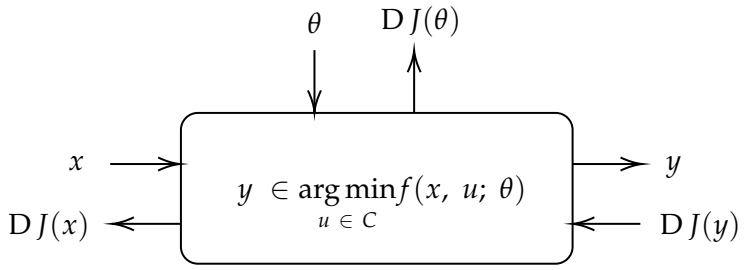

Figure 3.1: End-to-end learnable declarative node [Gould et al., 2019]

put. It defines the input-output relationship implicitly by a constrained optimization problem with an objective function, where the solution of the problem is the output.

Figure 3.1 shows the forward and backward pass of the declarative node. In the forward evaluation pass, the output of the declarative y is computed as the solution of some minimization problem $f(x, u; \theta)$. We use D to denote the total derivative with respect to the independent variables. Therefore, in the backward pass, the gradient of the global objective function with respect to the output $\mathrm{D}\, J(y)$ is back-propagated. Its value is computed through the chain rule based on the gradients with respect to the input $\mathrm{D}\, J(x)$ and parameters $\mathrm{D}\, J(\theta)$.

Since the definition of deep declarative nodes is very general, it can be embedded within another network for solving subproblems such as robust fitting. However, we may not be able to find the gradient when the feasible set is discrete, or the declarative node is low efficiency to evaluate. As non-regular solution cases, the nonexistent gradient problem will be discussed in the next part.

3.1.2 Learning

In declarative nodes, there are no explicitly defined forward functions, we can directly compute the optimal solution y through some algorithms. Under this assumption, when we performing the back-propagation, we can compute the gradient of the output from each node with respect to the corresponding input through the implicit differentiation. This can be treated as a bi-level optimization problem[Bard, 1998] where the parameterized constraints as a lower-level problem blinds variables in the objective function, an upper-level problem. Combining the schematic illustration in Figure 3.1, the problem can be defined formally as

$$\begin{aligned} &\text{minimize} && J(x, y) \\ &\text{subject to} && y \in \arg\min_{u \in C} f(x, u) \end{aligned} \tag{3.2}$$

We may have additional layers to make the objective function $J(x, y)$ depend on y, which is a function of x. In general, they are the sum of loss terms and regularization terms. We can solve this minimization problem by the gradient descent method as

follows:

$$\mathrm{D}\, J(x,y) = \mathrm{D}_X\, J(x,y) + \mathrm{D}_Y\, J(x,y)\, \mathrm{D}\, y(x) \tag{3.3}$$

where $\mathrm{D}_X\, J(x,y)$ is the partial derivatives of $J(x,y)$ with respect to x and $\mathrm{D}_Y\, J(x,y)$ is the partial derivatives of $J(x,y)$ with respect to y. We decompose the total derivatives of $J(x,y)$ as the sum of the partial derivatives with the chain rule. In application, we can consider it as the sum of gradients for losses on training examples.

The lower-level objective function f can be simplied. If it is the only term involving y in the upper-level objective function J. It means $J(x,y) = g(x, f(x,y))$ and the lower-level problem is actually unconstrained with $u \in C = \mathbb{R}^m$. Under this condition, the calculation of the gradient can be expanded using chain rule through both $\mathrm{D}_X\, J(x,y)$ and $\mathrm{D}_Y\, J(x,y)$:

$$\begin{aligned} \mathrm{D}\, J(x,y) &= \mathrm{D}_X\, g(x,f) + \mathrm{D}_F\, g(x,f)\, (\mathrm{D}\, f + \mathrm{D}_Y\, f\, \mathrm{D}\, y) \\ &= \mathrm{D}_X\, g(x,f) + \mathrm{D}_F\, g(x,f)\, \mathrm{D}\, f \end{aligned} \tag{3.4}$$

where $\mathrm{D}_Y\, f(x,y) = 0$ since y is the minimum of $f(x,y)$ and $f(x,y)$ is an unconstrained problem, its partial derivative should be zero.

In addition, for the solution y, we should verify its regularity.

Definition 3.1 (Regular Point)**.** [Gould et al., 2019] A feasible point u is said to be *regular* if the equality constraints gradient $D_U h_i$ and the active inequality constraints gradients $D_U g_i$ are linearly independent, or there are no equality constraints and the inequality constraints are all inactive at u.

Therefore, in unconstrained problems, we can consider the solution is regular by default. However in constrained problems, especially the inequality constrained problems, we should be aware of the continuity of the feasible set.

3.2 Back-propagation Through Declarative Nodes

Let us focus back on the more general case with y involving in different terms. The backward pass is different in different sub-classes of declarative nodes. We consider three common cases based on Equation 3.1.

3.2.1 Unconstrained

Firstly, the most basic case is the unconstrained problem. Consider a function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, we have

$$y \in \arg\min_{u \in C} f(x,u) \tag{3.5}$$

We make the assumption that the solution of this problem, $y(x)$ exists, and in the neighborhood of the point $(x, y(x))$, f is second-order differentiable. Therefore, we can compute the derivative of y with respect to x

$$\mathrm{D}\, y(x) = -H^{-1}B$$

where $H = \mathrm{D}^2_{YY} f(x, y(x)) \in \mathbb{R}^{m \times m}$ is the second-order derivative of f with respect to y, and it is a non-singular matrix. $B = \mathrm{D}^2_{XY} f(x, y(x)) \in \mathbb{R}^{m \times n}$ is the second-order derivative of f with respect to y and x (the derivative of $D_Y f(x, y)$ with respect to x).

The proof of this solution is similar to the proof of Equation 2.3: setting the partial derivative of $f(x, y)$ with respect to the optimal y as 0, then transposing and differentiating both sides according to the implicit function theorem:

$$\begin{aligned} 0_{m \times n} &= \mathrm{D}\left(\mathrm{D}_Y f(x, y)\right)^T \\ &= \mathrm{D}^2_{XY} f(x, y) + \mathrm{D}^2_{YY} f(x, y)\, \mathrm{D}\, y(x) \end{aligned} \tag{3.6}$$

Rearranging the equation, we get

$$\mathrm{D}\, y(x) = -\left(\mathrm{D}^2_{YY} f(x, y)\right)^{-1} \mathrm{D}^2_{XY} f(x, y) \tag{3.7}$$

Since this is the unconstrained case, for any stationary point of $f(x, y)$, the result is valid. In the following constrained cases, the optimal solution is actually the stationary point of the Lagrangian.

3.2.2 Equality Constrained

Secondly, we consider the equality constrained problem, that means the feasible set is defined by p nonlinear equality constraints. Consider functions $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ and $h : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$, we have

$$\begin{aligned} y(x) \in \arg\min_{u \in \mathbb{R}^m} & f(x, u) \\ \text{subject to} \quad & h_i(x, u) = 0, i = 1, \ldots, p \end{aligned} \tag{3.8}$$

where $h = [h_1, \ldots, h_p]^T$ are a set of constraints.

Again, we make the assumption that the solution of this problem, $y(x)$ exists and both f and all constraints in h are second-order differentiable. Also, we should consider the Jacobian matrix of h with respect to y is full rank since we need the optimal point is regular. Then we can calculate the derivative of y with respect to x is

$$\mathrm{D}\, y(x) = H^{-1} A^T \left(A H^{-1} A^T\right)^{-1} \left(A H^{-1} B - C\right) - H^{-1} B$$

where

$$\begin{aligned} A &= \mathrm{D}_Y h(x, y) \in \mathbb{R}^{p \times m} \\ B &= \mathrm{D}^2_{XY} f(x, y) - \sum_{i=1}^{p} \lambda_i \mathrm{D}^2_{XY} h_i(x, y) \in \mathbb{R}^{m \times n} \\ C &= \mathrm{D}_X h(x, y) \in \mathbb{R}^{p \times n} \\ H &= \mathrm{D}^2_{YY} f(x, y) - \sum_{i=1}^{p} \lambda_i \mathrm{D}^2_{YY} h_i(x, y) \in \mathbb{R}^{m \times m} \end{aligned}$$

and $\lambda \in \mathbb{R}^p$ can be solved through the system $\lambda^T A = \mathrm{D}_Y f(x, y)$.

Similar to the solution of linear equality constrained problem in Equation 2.4, we can form the Lagrangian by the method of Lagrange multipliers[Bertsekas, 2014]:

$$\mathcal{L}(x, y, \lambda) = f(x, y) - \sum_{i=1}^{p} \lambda_i h_i(x, y) \tag{3.9}$$

We introduce the Lagrange multipliers λ to set the stationary point of this Lagrangian is (y, λ). Then we differentiate $\mathcal{L}$ with respect to y and λ separately, which are both resulting in 0 since y is the optimal solution. We have two cases at the optimal point y. The first one is that the partial derivatives of f with respect to y equals to zero: $\mathrm{D}_Y f(x, y) = 0 \in \mathbb{R}^{1\times m}$. That means we transfer this problem to an unconstrained problem since its solution satisfies the constraints directly. Under this case, λ can be set as 0 directly. The second one is that the partial derivatives of f with respect to y is non-zero vector, and it is orthogonal to the constraint surface, which is controlled by the set of equality constraints $h(x, y) = 0$. In this case, from the derivative of $\mathcal{L}$ with respect to y equaling to zero, we have

$$\mathrm{D}_Y f(x, y) = \sum_{i=1}^{p} \lambda_i \mathrm{D}_Y h_i(x, y) = \lambda^T A \tag{3.10}$$

where A is the same as the defined above.

To explicitly solve this equation for λ, there is a unique analytic solution $\lambda = (AA^T)^{-1}A(\mathrm{D}_Y f)^T$.

Next, we compute the second-order derivative of $\mathcal{L}$ with respect to x. It still equals to zero in both functions. Solving the equation through variable elimination [Boyd et al., 2004] we have

$$\mathrm{D}\lambda(x) = \left(AH^{-1}A^T\right)^{-1}\left(AH^{-1}B - C\right) \tag{3.11}$$

$$\mathrm{D}y(x) = H^{-1}A^T\left(AH^{-1}A^T\right)^{-1}\left(AH^{-1}B - C\right) - H^{-1}B \tag{3.12}$$

which is our result.

The problem and solution defined in Section 2.2.2 is the simplier case compared to the general constrained problems. Also, if we only have one constraint, the matrix A and vector λ can be simplifed as vector and scalar separately. Moreover, for linear constraints, the repsentation of matrix H and B are also simplier since $\lambda = 0$.

3.2.3 Inequatlity Constrained

Lastly, inequality constrained problems are more complicated since we have to consider the solution for active and inactive constraints. Also, previous works metioned in Section 2.2.3 only give approximation results. Here, the declarative nodes consider the the problem containing both equality and inequality constraints and resulting in the exact solution. Consider functions $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ and $h : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^p$, and

$g : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^q$, we have

$$\begin{aligned} y(x) \in & \arg\min_{u \in \mathbb{R}^m} f(x,u) \\ & \text{subject to} \quad \begin{matrix} h_i(x,u) = 0, i = 1,\dots,p \\ g_i(x,u) \leq 0, i = 1,\dots,q \end{matrix} \end{aligned} \tag{3.13}$$

where $h = [h_1,\dots,h_p]^T$ are still a set of equality constraints, and $g = [g_1,\dots,g_q]^T$ are a set of inequality constraints.

In declarative nodes, it present a more general result based on activity of all inequality constraints at the optimal point $y(x)$. Also, assumptions of the existence of the optimal point $y(x)$ and second-order differentiation of f, g, h remain true. We combine the equality constraints and inequality constraints together in $\tilde{h} = [h_1,\dots,h_p,g_1,\dots,g_q]$. The derivative of $\tilde{h}$ with respect to y is a full rank matrix to keep the regularity. We have

$$\mathrm{D}\, y(x) = H^{-1}A^T \left(AH^{-1}A^T\right)^{-1} \left(AH^{-1}B - C\right) - H^{-1}B \tag{3.14}$$

where

$$\begin{aligned} A &= \mathrm{D}_Y \tilde{h}(x,y) \in \mathbb{R}^{(p+q)\times m} \\ B &= \mathrm{D}^2_{XY} f(x,y) - \sum_{i=1}^{p} \lambda_i \mathrm{D}^2_{XY} \tilde{h}_i(x,y) \in \mathbb{R}^{m\times n} \\ C &= \mathrm{D}_X \tilde{h}(x,y) \in \mathbb{R}^{(p+q)\times n} \\ H &= \mathrm{D}^2_{YY} f(x,y) - \sum_{i=1}^{p+q} \lambda_i \mathrm{D}^2_{YY} \tilde{h}_i(x,y) \in \mathbb{R}^{m\times m} \end{aligned}$$

and $\lambda \in \mathbb{R}^{p+q}$ satisfies $\lambda^T A = \mathrm{D}_Y f(x,y)$ with $\lambda_i \leq 0$ for $i = p+1,\dots,p+q$, which is almost the same as the solution of the equality constrained problem.

However, in inequality constraints, the gradient is discontinuous since the Lagrange multipliers λ for active inequality constraints are zero. We separate the solution of inequality constrained declarative nodes into 3 scenarios, which is illustrated in Figure 3.2. For any inequality constraints $g(x,u) \leq 0$, the constraint can be active or inactive at the optimal solution y. In the first scenario, the constraint is inactive at the solution y, which means $g(x,y) < 0$ and it is completely in the feasible set (y_1). Therefore we have $\mathrm{D}_Y f(x,y) = 0$ and we can consider it as an unconstrained problem. In the second scenario, the constraint is active at y, but it is orthogonal to the constraint surface (y_2). Then we have $\mathrm{D}_Y f(x,y) \neq 0$ and $\lambda \neq 0$. Then we can consider this inequality constraint as an equality one since the solution falls on the boundary of the feasible set, but the gradient of negative and pointing outside the set. The last scenario, the constraint is active and $\mathrm{D}_Y f(x,y) = 0$ when the solution is on the boundary and it is a local minimum (y_3). For the backward propagation, in this case, we can choose both solutions of unconstrained and equality constrained gradient.

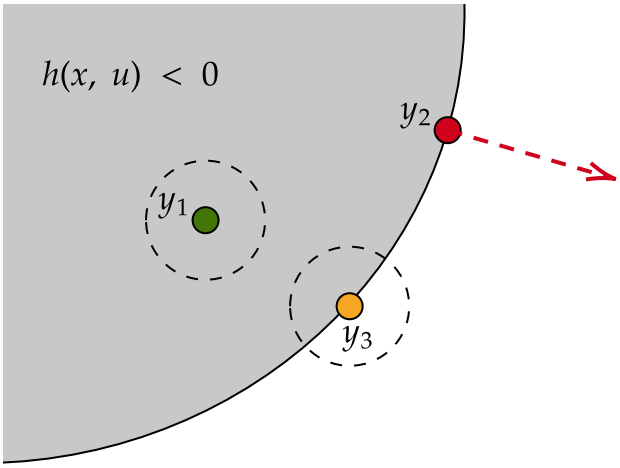

Figure 3.2: Different scenarios for the solution to inequality constrained nodes [Gould et al., 2019]

3.3 Examples of Declarative Nodes

3.3.1 Implementation Details

We give the implementations of both equality and inequality constraints with linear and nonlinear cases in raw Python under the gradient calculation package Autograd[Maclaurin et al., 2015]. In this subsection, we are going to show the implementation details of the basic deep declarative nodes.

Constraints definition. In the multiple equality constraints node and inequality constraints node, equality constraints and inequality constraints are defined in two functions separately. If there are more than one constraint, they are stored in a 1-d array. We define the general form for all constraints that they equal to or less than zero. All parameters in the equations are normalized to the left-hand side and the right-hand side is always zero. There is a specific checking function defined to check this. In the linear equality constraints node, the set of linear constraints are defined as a single matrix A and its corresponding vector b in the initialization directly.

Gradient computation. To compute the gradient of the solution under the constraints, we need to calculate the Jacobian and Hessian matrix, which are the first-order and second-order derivative matrix of the constraints. For matrix H in the general solution of the constrained problem, it is costly to compute the inverse of it. Therefore, we transfer the problem to solving a linear system $Hx_1 = A^T$ and $Hx_2 = B$ where $x_1 = H^{-1}A^T$ and $x_2 = H^{-1}B$. Cholesky decomposition, an efficient algorithm for solving the linear system, is applied to this problem. It decomposes a Hermitian positive-definite matrix into a unique product of a lower triangular matrix, of which the diagonal entries are real and positive, and its conjugate transpose. The solution of the Lagrange multipliers λ similar, which is the solution of a linear system $\lambda^T A = \mathrm{D}_Y f(x, y)$ through the least-square solution solver.

Optimality checking. For constrained optimization problems, according to the optimality conditions defined in Section 2.1.3, we should check if the first-order optimality condition is satisfied. We define a function in all cases for checking the optimality: the gradient of constraint is zero at optimal point ($\mathrm{D}_Y h(x,y) = 0$), or the gradient of objective function equals to the product of the Lagrange multipliers and the gradient of constraints ($\mathrm{D}_Y f(x,y) = \lambda \mathrm{D}_Y h(x,y)$).

Exception handling. Since our assumptions are based on the regular solution, and not all problems have a regular optimal point, we define the exception cases for this problem. When we solve the linear system for the Lagrange multipliers λ, if the solution does not fall as a regular point, we may get a null value in λ due to the non-existence of the gradient on a non-regular solution. We are going to discuss the solution for this specific case in the next part. For here, we just throw the exception once there is a null value in the Lagrange multipliers λ.

3.3.2 Equality Constrained

Firstly, we consider a single linear equality constrained problem, minimizing the KL-divergence between the input x and output y subject to the output forming a valid probablility vector, which can be formally defined as

$$\begin{aligned} y = \quad & \underset{u}{\operatorname{argmin}} \quad -\sum_{i=1}^{n} x_i \log u_i \\ & \text{subject to} \quad \sum_{i=1}^{n} u_i = 1 \end{aligned} \tag{3.15}$$

where the positivity constraint on y is automatically satisfied by the domain of the log function.

A nice feature of this problem is that we can solve it in closed-form as

$$y = \frac{1}{\sum_{i=1}^{n} x_i} x.$$

Now we are going to solve this problem via an iterative method with derivative of deep declarative node. Set $n = 5$ and $m = 5$, which means the input $x \in \mathbb{R}^5$ and the output $y \in \mathbb{R}^5$. We begin from a random feasible solution, which is the normalization of the value y, then perform the gradient update. Figure 3.3 shows the function and gradient sweeping of the first component of the input x_1 from 0.1 to 10.0 while holding the other elements of x constant.

Now we consider a multiple non-linear equality constrained problem defined as

$$\begin{aligned} y = \quad & \underset{u}{\operatorname{argmin}} \quad \sum_{i=1}^{n} x_i u_i^2 \\ & \text{subject to} \quad \sum_{i=1}^{n-1} u_i^2 = 1 \\ & \qquad\qquad\quad \sum_{i=1}^{n} u_i = 0 \end{aligned} \tag{3.16}$$

Same as the previous example, we instantiate the multiple equality constriants nodes in deep declarative nodes with $n = 3$ and $m = 3$. We begin from a feasible solution, $\cos^2 x + \sin^2 x - \sin x - \cos x = 0$, performing the gradient descent until the

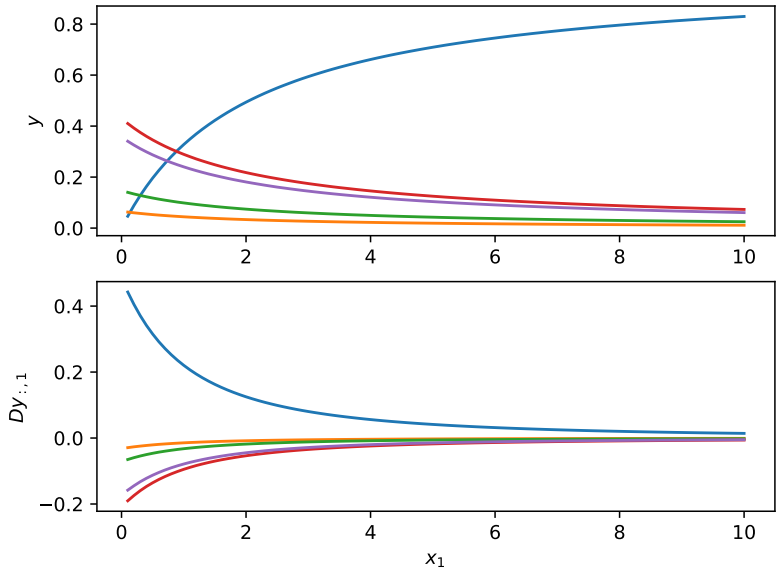

Figure 3.3: Plots of the function y (top) and the gradient (bottom) sweeping the first component of the input x_1 while holding the other elements of x constant

convergence. Figure 3.4 shows the function y and its gradient changes.

3.3.3 Inequatlity Constrained

For inequality constrained problems, we consider a similar problem based on the multiple equality constrained problem above. The only difference is that we restrict the first parameter u_1 to be smaller than u_2. Therefore, the problem is defined as

$$\begin{aligned} y = \quad & \underset{u}{\text{argmin}} \quad \sum_{i=1}^{n} x_i u_i^2 \\ & \text{subject to} \quad \sum_{i=1}^{n-1} u_i^2 = 1 \\ & \qquad\qquad\quad \sum_{i=1}^{n} u_i = 0 \\ & \qquad\qquad\quad u_1 - u_2 < 0 \end{aligned} \tag{3.17}$$

The deep declarative node is the instantiation of inequality constraints nodes with $n = 3$ and $m = 3$. Same as the feasible solution as the beginning, we set $x = \pi/6$, which satisfies the inequality constraints since $\sin(\pi/6) < \cos(\pi/6)$. Figure 3.5 shows the function y and its gradient changes. The gradient flattened to some extreme value at the beginning of the iteration then converges to zero.

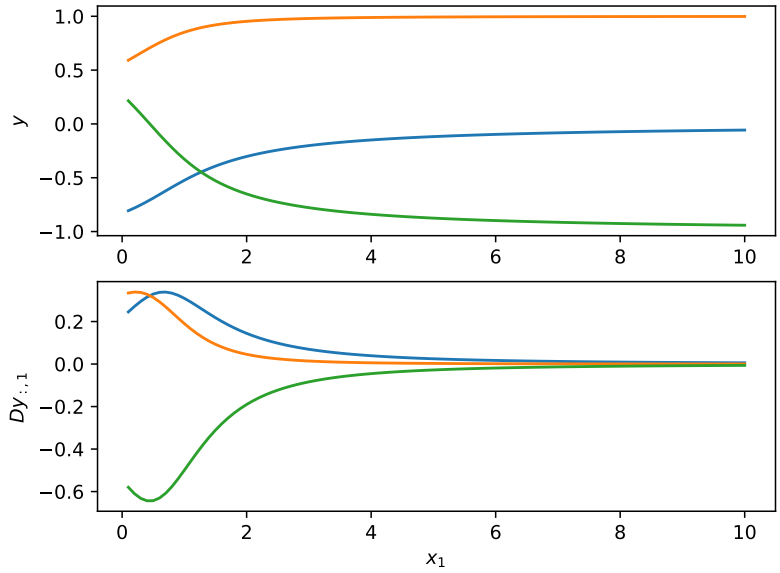

Figure 3.4: Plots of the function y (top) and the gradient (bottom) sweeping the first component of the input x_1 while holding the other elements of x constant

3.4 Future Work of the Deep Declarative Network

There are various possible challenging extensions of the deep declarative network in both theories and applications. As a differentiable network, it combines the exact functionality solution with the iterative gradient method. In this section, we give several future foreseeable extensions of the declarative network comparing with other state-of-the-art differentiable models in both theory optimization and applications in computer vision tasks.

We give the solution to the deep declarative nodes of constrained problems under the assumption that the solution of the problem $y(x)$ exists, and it should be a regular point whose gradient can be computed. Also, both objective function and constraints are required to be first and second-order differentiable, which also means that they are continuous instead of discrete. Therefore, this method may not able to solve discrete functions such as binary constrained classification problems in computer vision. Some relevant extension methods are discussed in PART II. Also, the robustness of the declarative nodes can be improved by introducing attention to specific constraints.

In solving the real computer vision tasks, compared to other differentiable models, the deep declarative network can be applied to the visual Sudoku problem and other similar puzzles. Also, many constrained optimization problems in the graph can be solved through this method. Future work in applications can focus on providing a more specific algorithm for different tasks, and more applications can be explored.

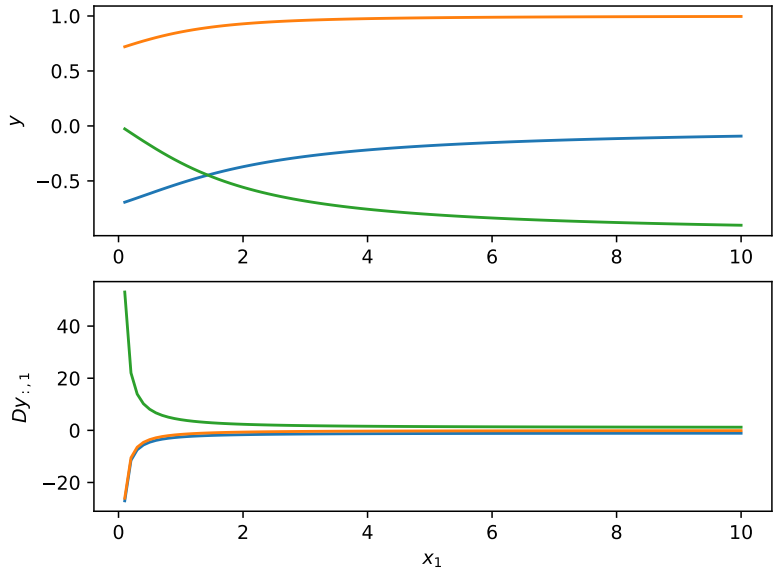

Figure 3.5: Plots of the function y (top) and the gradient (bottom) sweeping the first component of the input x_1 while holding the other elements of x constant

3.5 Summary

In Chapter 3, we describe the deep declarative network from its structure to its implementation with examples of different constrained problems. As a differentiable neural network, the deep declarative network defines its nodes to process input implicitly through the optimization problem, which can find the optimal solution directly without the verification of the local or global minimum. Its learning progress is also based on the derivative of the solution and the gradient of its constraints. We also introduce the back-propagation through deep declarative nodes in three subclasses: unconstrained, constrained, and inequality constrained problems. All representations of the gradient in declarative nodes have assumptions of the existence of the optimal point, its regularity, and its first and second-order derivative. In the next chapter, we are going to give an overview of the non-regular solution with related previous works.

Part II

Deep Declarative Nodes: Non-regular Solution

Chapter 4

An Overview of Regular and Non-regular Solution

In PART I, we described the solution for both unconstrained and constrained problems in deep declarative nodes: its theoretical background of numerical optimization, the general solution for unconstrained and constrained optimization problems, and the details of the back-propagation in deep declarative nodes. However, the solutions given before are based on the assumption that the solution exists, and both objective functions and constraints are second-order differentiable. In PART II, we discuss the solution for different non-regular points which can approximate the gradient of constrained problems.

In this chapter, we will give an overview of the non-regular point for deep declarative nodes supported by related previous works. We focus on the solution that is not regular, which means we cannot use the solutions we proposed in the previous chapter to solve them.

We first discuss possible problems with non-regular solutions in the deep declarative nodes in Section 4.1 with corresponding examples. Next, we briefly discuss the related works in solving non-regular points problems in Section 4.2. We also give some comparisons between these existed approaches.

We finally give a summary of the non-regular solution cases and our literature review findings.

4.1 Problems in Regular Deep Declarative Nodes

4.1.1 Assumptions and Problems

In Section 3.2, we provided general solutions for unconstrained and constrained optimization problems in deep declarative nodes. For both problems, we assume that the solution of the problem, $y(x)$ exists in the neighborhood of the point $(x, y(x))$. Also, the objective function is supposed to be second-order differentiable. In particular, for constrained optimization problems, all constraints are also assumed to be second-order differentiable. These assumptions are made to guarantee that the optimal point is regular and strictly minimum.

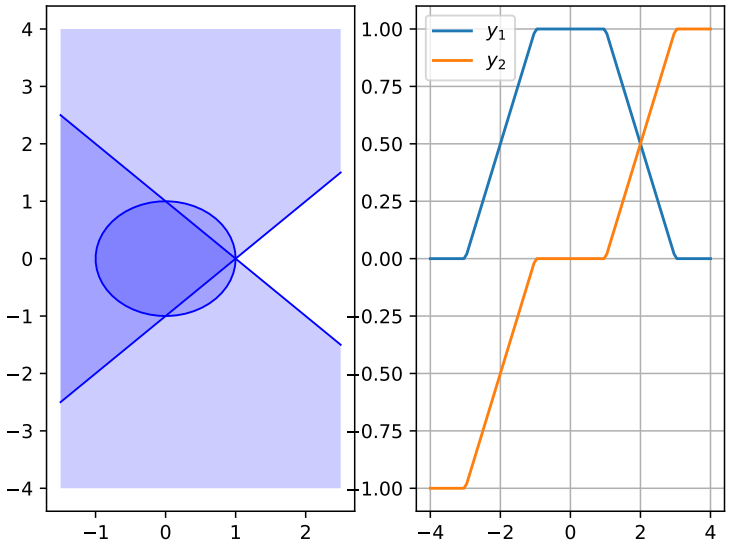

Figure 4.1: The feasible solution set and the results of the overdetermined system [Gould et al., 2019]

However, not all problems have differentiable constraints and the Jacobian or Hessian matrix of constraints are full-rank. Also, the first-order derivatives of the equality constraints and active inequality constraints may not linearly independent. The dimension of the output may also various from the number of active constraints. According to the solution of equality constraints and inequality constraints in Equation 3.12 and Equation 3.14 with their corresponding representations of matrix H, we have to compute the inverse of the matrix H to get the solution. If H is singular or very sparse, we are not able to get the exact solution since we cannot apply the Cholesky decomposition to this matrix.

There are many possible problems of non-regular solutions in the deep declarative nodes. We classify them into three scenarios: Overdetermined system, rank deficiency problems, and non-convex cases.

4.1.2 Overdetermined System

If the number of constraints is more than the number of unknowns to be solved, this system of equations is considered as an overdetermined system. In declarative nodes, we require at least one degree of freedom for the back-propagation, since we have to update the gradient with the direction in a feasible solution set. Therefore, when the number of active constraints exceeds the dimensionality of the output, the system is overdetermined.

We give the first example constrains the solution to a circle with two segments removed in Figure 4.1 from the tutorial by Gould et al. [2019]. Also, we formulate

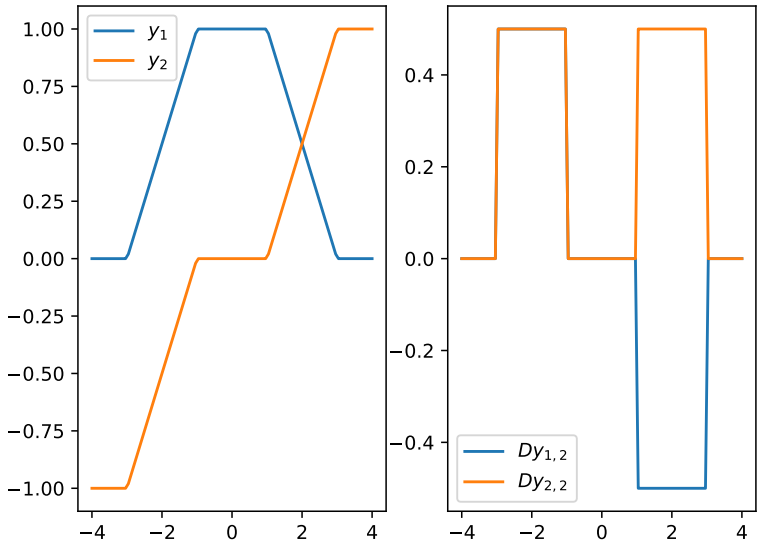

Figure 4.2: The results and corresponding gradient solution of the overdetermined system [Gould et al., 2019]

this as the intersection of a (solid) circle constraint with two half-spaces. The problem is defined officially as

$$\begin{aligned} y \in \quad & \operatorname{argmin}_u \quad \tfrac{1}{2}\|u-x\|^2 \\ & \text{subject to} \quad u_1^2+u_2^2-1 \leq 0 \quad (h_1) \\ & \qquad\qquad\quad u_1-u_2-1 \leq 0 \quad (h_2) \\ & \qquad\qquad\quad u_1+u_2-1 \leq 0 \quad (h_3) \end{aligned} \tag{4.1}$$

From Figure 4.1, there is only one intersection point among three constraints, where all three constraints are active. We calculate the graident of it using the implicit differentiation of the KKT optimality conditions as discussed in Equation 3.14 and their corresponding formulas. Figure 4.2 shows the value of y and the corresponding gradient. Using the solution for inequality constraints problems, we can get

$$\begin{aligned} A &= \mathrm{D}_Y h(y) &&= \begin{bmatrix} 2y_1 & 2y_2 \\ 1 & -1 \\ 1 & 1 \end{bmatrix} \quad \text{for active } h_i \\ B &= \mathrm{D}^2_{XY} f(x,y) - \textstyle\sum_{i=1}^3 \lambda_i \mathrm{D}^2_{XY} h_i(y) &&= -I \\ C &= \mathrm{D}_Y h(y) &&= 0 \\ H &= \mathrm{D}^2_{YY} f(x,y) - \textstyle\sum_{i=1}^3 \lambda_i \mathrm{D}^2_{YY} h_i(y) &&= (1-2\lambda_1)I \end{aligned}$$

where inactive constraints, and corresponding Lagrange multipliers, are first removed. Thus, the matrix A may have between zero and three rows, which is not

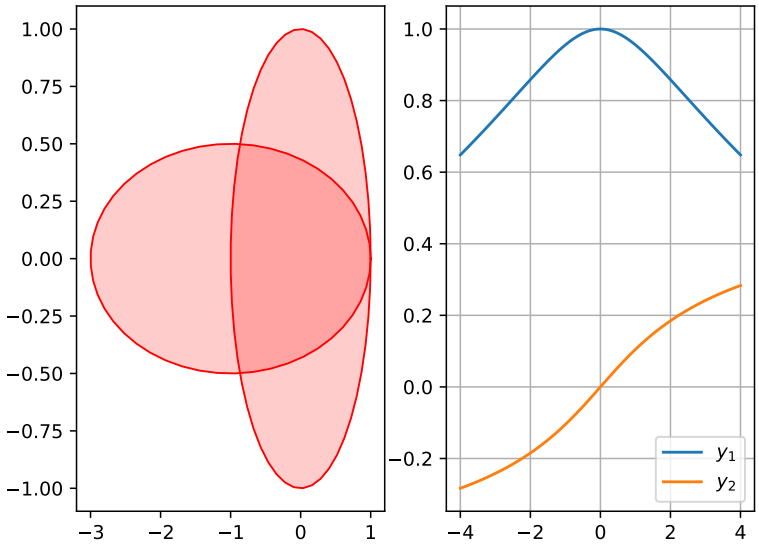

Figure 4.3: The feasible solution set and the results of the rank-deficiency problem [Gould et al., 2019]

regular. This equation can be solved to give different results:

$$\mathrm{D}y(x) = \begin{cases} I & \text{if all constraints are inactive} \\ 0 & \text{if all constraints are active (since } C = 0) \\ I - A^T(AA^T)^{-1}A & \text{if } h_1 \text{ is inactive} \\ \frac{1}{1-2\lambda_1}\left(I - A^T(AA^T)^{-1}A\right) & \text{otherwise.} \end{cases}$$

where the gradient is zero if all constraints are active. In this scenario, we can not perform back-propagation in our deep declarative network.

4.1.3 Rank Deficiency Problems

Contrary to the overdetermined system, rank deficiency means that there are insufficient equations to determine the solution, which is the so-called underdetermined system. More generally, we do not have enough information to estimate the desired model in this case.

In declarative nodes, if the first-order derivatives of the equality constraints and active inequality constraints are linearly dependent, the solution points are not regular since the KKT conditions may still be satisfied but there are still some degrees of freedom in the Lagrange multipliers. The direct result is that matrix H is sparse and singular.

We give the example that the constraint set is defined as the intersection between

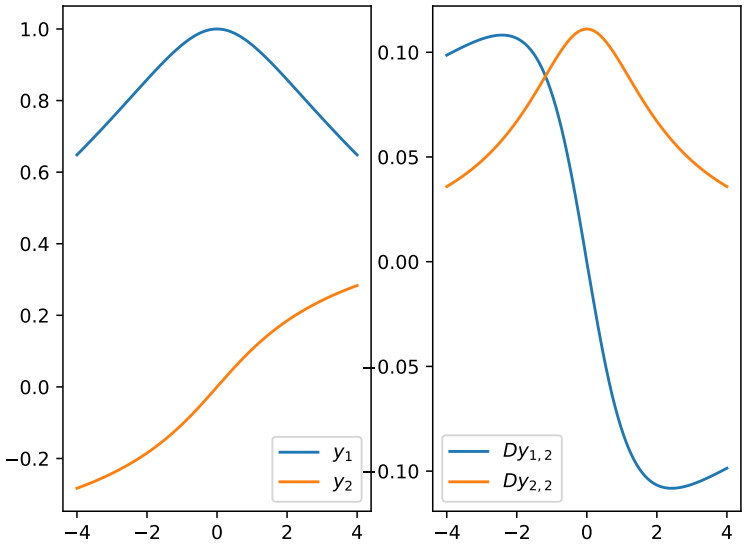

Figure 4.4: The results and corresponding gradient solution of the rank-deficiency problem [Gould et al., 2019]

a circle and an ellipse in Figure 4.3 from the tutorial by [Gould et al., 2019]. The problem is defined officially as

$$\begin{aligned} y \in \quad & \operatorname{argmin}_u \quad \tfrac{1}{2}\|u-x\|^2 \\ & \text{subject to} \quad u_1^2 + u_2^2 - 1 \leq 0 \qquad (h_1) \\ & \qquad\qquad \tfrac{1}{4}(u_1+1)^2 + 4u_2^2 - 1 \leq 0 \qquad (h_2) \end{aligned} \tag{4.2}$$

where the solution $y \in \mathbb{R}^2$ is a function of x and at $x_2 = 0$, both constraints are active. This results in A being rank deficient.

Similar to the previous overdetermined system, we can compute the gradient $\mathrm{D}\,y(x)$ of this rank deficient problem with each matrix as follows

$$\begin{aligned} A &= \begin{bmatrix} 2y_1 & 2y_2 \\ \frac{1}{2}(y_1+1) & 8y_2 \end{bmatrix} && \text{for active } h_i \\ B &= -I \\ C &= 0 \\ H &= \begin{bmatrix} 1-2\lambda_1 - \frac{1}{2}\lambda_2 & 0 \\ 0 & 1-2\lambda_1 - 8\lambda_2 \end{bmatrix} \end{aligned}$$

where A is rank deficient at $y = (1, 0)$.

Some columns are linearly dependent in A, so we need to remove one of the rows of A before solving for $\mathrm{D}\,y(x)$. One strategy is to keep those constraints where the rate of change of the objective is steepest relative to the curvature induced by the

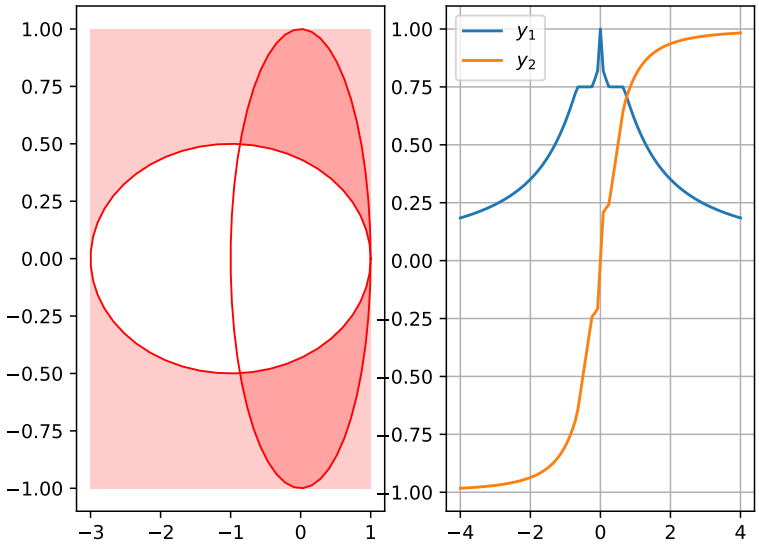

Figure 4.5: The feasible solution set and the results of the non-convex case [Gould et al., 2019]

constraint surface. That is, remove from A rows that are linearly dependent on other rows and with the smaller $\mathrm{D}_Y f(\mathrm{D}^2_{YY} h_i)^{-1} \mathrm{D}_Y f^T$. Figure 4.4 shows the solution of y with their corresponding gradient.

4.1.4 Non-convex Cases

The last scenario is the non-convex cases, which is also a rank deficient problem. In general, solving a non-convex problem is NP-hard since it can have many local minima or its solutions are in very flat regions, which is hard to update the gradient.

Similarly, we give the example that the constraint set is defined as the area in the circle that is not within the ellipse from the tutorial by Gould et al. [2019],

$$\begin{aligned} y \in \quad & \operatorname{argmin}_u \quad \tfrac{1}{2}\|u - x\|^2 \\ & \text{subject to} \quad u_1^2 + u_2^2 - 1 \leq 0 \qquad (h_1) \\ & \qquad\qquad \tfrac{1}{4}(u_1 + 1)^2 + 4u_2^2 - 1 \geq 0 \quad (h_2) \end{aligned} \tag{4.3}$$

Figure 4.5 shows the feasible solution set, which is non-convex of this problem and the results y with fixed $x_1 = 0.75$ and x_2 sweeping from -4 to 4. When $x = (0.75, 0)$, the result $y = (1, 0)$ and both constraints h_1 and h_2 in Equation 4.3 are active. Thus at this point, the matrix A is deficient.

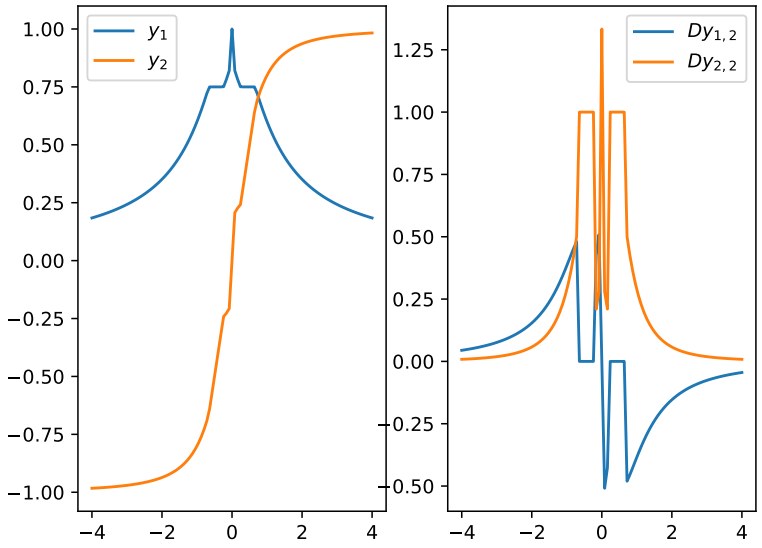

Figure 4.6: The results and corresponding gradient solution of the non-convex case [Gould et al., 2019]

Now we use the same method to calculate the gradient $\mathrm{D}\,y(x)$, we have

$$\begin{aligned} A &= \begin{bmatrix} 2y_1 & 2y_2 \\ -\frac{1}{2}(y_1+1) & -8y_2 \end{bmatrix} && \text{for active } h_i \\ B &= -I \\ C &= 0 \\ H &= \begin{bmatrix} 1-2\lambda_1+\frac{1}{2}\lambda_2 & 0 \\ 0 & 1-2\lambda_1+8\lambda_2 \end{bmatrix} \end{aligned}$$

where A is rank deficient at $y = (1, 0)$. Figure 4.6 shows the solution of y with their corresponding gradient.

We give an example for each non-regular solution scenario, which is not able to compute the gradient directly. In the next section, we discuss several previous related works in solving these problems.

4.2 Related Work in Non-regular Solution

Most previous works in the differentiable network only focus on the regular and convex solution. However, to some extent, dealing with non-regular solutions is solving non-regular linear systems. Therefore, we present several related works in discussing the solution of the overdetermined system, rank deficient problems, and non-convex cases in linear equations. Most of these approaches are based on theoretical optimization only, but some of them are problem-based. It means that in solving real

computer vision tasks, we need to consider the actual problems then find the best solution.

Overdetermined system. There are various approximations of the solution for the overdetermined system. The most classical approach is the ordinary least squares, which minimize the L_2 residual between Ax and b [Anton and Rorres, 2013]. However, if we want to have more numerically accurate solutions, QR factorization can give better results [Trefethen and Bau III, 1997]. Besides, Barrodale and Roberts [1974] proposed the solution in the L_1 norm for calculating those data contains wild points, which are unstable with L_2 norm. They also introduced an algorithm to solve the linear Chebyshev data fitting problem, which is also overdetermined [Borrodale and Phillips, 1975]. Apart from them, Watson [1979] demonstrated a minimax solution for the overdetermined system of nonlinear equations based on the minimax norm. All these methods are similar since they are implemented to minimize the error between the exact solution and the approximation.

Rank deficiency problems. As an underdetermined system, many previous methods are provided through a similar approximation as the overdetermined system for the sparse matrix A. Donoho [2005] demonstrated a method for finding the unique sparse solution by L_1 minimization and neighborliness of convex polytopes, which transferring the problem to a convex optimization problem. They also proved that the minimal L_1-norm solution is also the sparsest solution [Donoho, 2006]. Some other methods such as the successive overrelaxation method introducing the augmented coefficient matrix for finding the least square solution [Darvishi and Khosro-Aghdam, 2006] and transforming the matrix into a smaller full rank as sparse as possible one [Wu et al., 2004] are also effective. Apart from the traditional methods, Wang [1997] proposed an approach using recurrent neural networks to calculate the pseudoinverses of the rank deficient matrix, which is novel and practical.

Non-convex cases. Solving NP-hard problems directly is expensive. Methods for solving convex optimization problems can also be applied to non-convex cases such as stochastic gradient descent [Robbins and Monro, 1951], although the convergence is not guaranteed. Variance reduction is designed for nonconvex problems to improve the convergence of non-convex problems. Reddi et al. [2016] analyzed the stochastic variance reduced gradient method for non-convex problems, which achieved faster convergence than the traditional stochastic gradient method. Allen-Zhu and Hazan [2016] also demonstrated an algorithm based on the variance reduction trick with the first-order minibatch stochastic method to accelerate the training.

4.3 Summary

In Chapter 4, we raise the problems from the original deep declarative nodes under the assumptions it makes. For these non-regular solution points, we discuss them in three scenarios: overdetermined system, rank deficient problems and non-convex cases, where the later two cases are underdetermined systems. All of these solutions are not able to compute the gradient directly since we cannot solve the linear system

with traditional methods, and there is no exact solution. In addition, we introduce some related works in solving these irregular linear systems, which are approximating the closest solution to the problem by minimizing the residual norm. In the next chapter, for each scenario, we provide two efficient approaches and give a detailed comparison between these methods.

Chapter 5

Solutions of Non-regular Points

In last chapter, we discussed various previous works in solving the non-regular point problem. However, not all these approaches are suitable for deep declarative nodes and they may not be able to optimize each parameter in the node. In this chapter, we set out to solve this problem efficiently in the extension of the deep declarative network.

For each scenario, we provide two practical solutions in Section 5.1 (overdetermined system), Section 5.2 (rank deficiency problems) and Section 5.3 (non-convex cases) separately. More specifically, we introduce the least-squares method and conjugate gradient preconditioning approach for the overdetermined system, which are practical and classical approximation algorithms. For rank deficient problems, we firstly propose a greedy strategy, the orthogonal matching pursuit algorithms to recover the matrix as a full rank one. We also present that in the application of the deep declarative network, how to avoid the non-regular point and minimize the approximation error. In the last scenario, we provide a theoretical method based on the optimality conditions of the optimization problem, which is the extension of the traditional Lagrange multipliers approach. We finally prospect some future improvements for the non-regular solution in the optimization problem in Section 5.4. The summary of this chapter is given at the end.

5.1 Overdetermined System

5.1.1 Least-Squared Method

As a classical approach to approximate the solution of the overdetermined system in linear analysis, the least-squares method is powerful and empirical in many prediction problems by minimizing the sum of the squares of the residuals. [Datta, 2010]

We begin with the linear system of equations. Supposed we have such a linear system

$$\mathbf{A}\mathbf{x} = \mathbf{b} \tag{5.1}$$

where $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $m > n$, $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{b} \in \mathbb{R}^m$. Therefore, there are more equations than unknowns, which is an overdetermined system and there is no solution making $\mathbf{A}\mathbf{x} = \mathbf{b}$ for all $\mathbf{x}$. $\mathbf{b}$ is also not in the column subspace of $\mathbf{A}$, hence $\mathbf{b}$ is not

a linear combination of the column vectors of $\mathbf{A}$. With the least-square method, we want to find an $\mathbf{x}$ which makes the residual vector $\mathbf{r} \in \mathbb{R}^m$ approaching zero:

$$\mathbf{r} = \mathbf{b} - \mathbf{A}\mathbf{x}, \text{for each element in } \mathbf{r} : r_i = b_i - \sum_{j=1}^{n} a_{ij}x_j, \quad i = 1, \ldots, m \tag{5.2}$$

The solution $\mathbf{x}$ in Equation 5.2 is obtained by the least-squares method minimizing $\|\mathbf{r}\|_2 = \|\mathbf{b} - \mathbf{A}\mathbf{x}\|_2$, which is also the sum of errors:

$$\|\mathbf{r}\|_2 = \mathbf{r}^T\mathbf{r} = (\mathbf{b} - \mathbf{A}\mathbf{x})^T(\mathbf{b} - \mathbf{A}\mathbf{x}) = \mathbf{b}^T\mathbf{b} - \mathbf{x}^T\mathbf{A}^T\mathbf{b} - \mathbf{b}^T\mathbf{A}\mathbf{x} + \mathbf{x}^T\mathbf{A}^T\mathbf{A}\mathbf{x} \tag{5.3}$$

We aim to find the optimal approximation $\mathbf{x}$ that minimizes $\|\mathbf{r}\|_2$ in Equation 5.3. Therefore, we compute the derivative of Equation 5.3 with respect to $x_k, k = 1, \ldots, n$ and set it to zero

$$\frac{\partial\|\mathbf{r}\|}{\partial x_k} = \frac{\partial}{\partial x_k}\left[\sum_{i=1}^{m}\left(b_i - \sum_{j=1}^{n} a_{ij}x_j\right)^2\right] = 2\sum_{i=1}^{m}\left(b_i - \sum_{j=1}^{n} a_{ij}x_j\right)(-a_{ik}) = 0) \tag{5.4}$$

where a_{ij} are elements in $\mathbf{A}$.

From Equation 5.4, we can find

$$\sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}a_{ik}x_j = \sum_{j=1}^{n}\left[\sum_{i=1}^{m} a_{ij}a_{ik}\right]x_j = \sum_{i=1}^{m} b_i a_{ik} \tag{5.5}$$

which is equivalent to

$$\mathbf{A}^T\mathbf{A}\mathbf{x} = \mathbf{A}^T\mathbf{b} \tag{5.6}$$

Consequently, we can solve $\mathbf{x}$

$$\mathbf{x} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b} = \mathbf{A}^{-}\mathbf{b} \tag{5.7}$$

where $\mathbf{A}^{-}$ is the pseudoinverses of $\mathbf{A}$ and $\mathbf{A}^T\mathbf{A}$ is supposed to be invertible since $\operatorname{rank}\mathbf{A} < \min(m, n)$.

In deep declarative nodes, if constraints are not second-order differentiable, matrix $B = \mathrm{D}^2_{XY}f(x,y) - \sum_{i=1}^{p+q}\lambda_i\mathrm{D}^2_{XY}\tilde{h}_i(x,y) \in \mathbb{R}^{m\times n}$ and matrix $H = \mathrm{D}^2_{YY}f(x,y) - \sum_{i=1}^{p+q}\lambda_i\mathrm{D}^2_{YY}\tilde{h}_i(x,y) \in \mathbb{R}^{m\times m}$ are undefined. Therefore, there is no exact solution for the linear system $Hx_1 = A^T$ and $Hx_2 = B$ since it is overdetermined. In this circumstance, using the least-squares method to approximate the solution of the linear system is a classical approach. Figure 5.1 shows an example of overdetermined system using the least-squares method to approximate the solution of the gradient, where the gradient in one of three variables converges to zero and others are almost zero.

The least-squares method is very basic and practical since the residual between the ground truth and approximation is minimized, which usually leads the optimal solution. Many methods are based on this in solving different non-regular solution

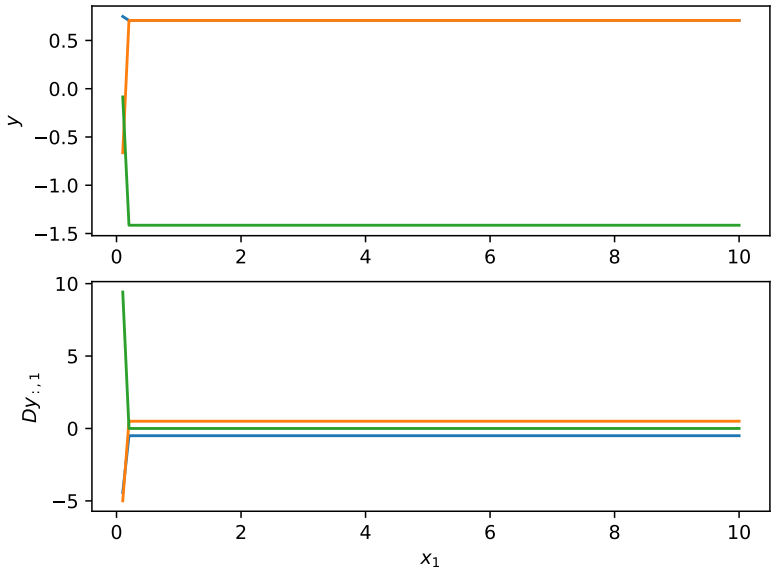

Figure 5.1: Plots of the function y (top) and the gradient (bottom) sweeping the first component of the input x_1 while holding the other elements of x constant with the least-squares method

problems, which will be discussed in Section 5.2 and Section 5.3.

5.1.2 Conjugate Gradient and Preconditioning

We mentioned the steepest descent method in solving unconstrained optimization problems in Section 2.2. In general, the steepest gradient descent takes several steps in the same direction. In the conjugate gradient method, it only takes one step in each direction from a set of orthogonal search directions [Hestenes et al., 1952].

Supposed we have such a set of orthogonal search directions $\mathbf{d}_{(0)}, \mathbf{d}_{(1)}, \cdots, \mathbf{d}_{(n-1)}$, and we update the gradient $\mathbf{x}$ n steps

$$\mathbf{x}_{(i+1)} = \mathbf{x}_{(i)} + \alpha_{(i)}\mathbf{d}_{(i)} \tag{5.8}$$

where α_i is the step size. To determine the value of α_i for each movement under the orthogonal relation between the residual $\mathbf{e}_{(i+1)}$ and the gradient direction $\mathbf{d}_{(i)}$, we take different direction for each update

$$\begin{aligned} \mathbf{d}_{(i)}^T \mathbf{e}_{(i+1)} &= 0 \\ \mathbf{d}_{(i)}^T \left(\mathbf{e}_{(i)} + \alpha_{(i)}\mathbf{d}_{(i)} \right) &= 0 \\ \alpha_{(i)} &= -\frac{\mathbf{d}_{(i)}^T \mathbf{e}_{(i)}}{\mathbf{d}_{(i)}^T \mathbf{d}_{(i)}} \end{aligned} \tag{5.9}$$

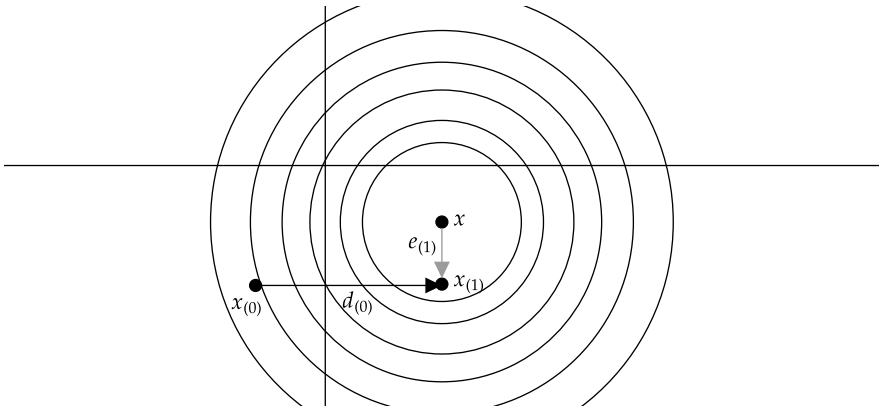

Figure 5.2: Directions of the orthogonal gradient method

where the residual $\mathbf{e}_{(i+1)}$ is the difference between the ground truth x and the current updated x_{i+1}.

Figure 5.2 shows the gradient update from $x_{(0)}$ to x_1 using the orthogonal gradient method, where x is the ground truth and $x_{(1)}$ is the updated gradient. From this figure, it is clear that the direction of vector $d_{(0)}$ and the residual vector $e_{(1)}$ are orthogonal. However, we cannot decide the $\mathbf{e}_{(i)}$, so we used to make these two vectors **A**-orthogonal, which means

$$\left\langle \mathbf{d}_{(i)}, \mathbf{e}_{(i+1)} \right\rangle_A := \mathbf{d}_{(i)}^T \mathbf{A} \mathbf{e}_{(i+1)} = 0 \tag{5.10}$$

Figure 5.3 is an example of the A-orthogonality between two vectors. They are not exactly orthogonal, but A-orthogonal.

From the Equation 5.10, we get the step size

$$\alpha_{(i)} = \frac{\mathbf{d}_{(i)}^T \mathbf{r}_{(i)}}{\mathbf{d}_{(i)}^T \mathbf{A} \mathbf{d}_{(i)}} \tag{5.11}$$

where in this case, the two vectors are not absolutely orthogonal and it depends on the matrix **A**. This **A**-orthogonality is applied to the direction set $\left\{ \mathbf{d}_{(i)} \right\}$. We decide the search directions through n linearly independent vectors **u**

$$\mathbf{d}_{(i)} = \mathbf{u}_i + \sum_{k=0}^{i-1} \beta_{i,k} \mathbf{d}_{(k)}, \quad i > 0 \tag{5.12}$$

where

$$\beta_{i,j} = \frac{-\mathbf{u}_i^T \mathbf{A} \mathbf{d}_{(j)}}{\mathbf{d}_{(j)}^T \mathbf{A} \mathbf{d}_{(j)}} \tag{5.13}$$

which the second direction based on the **A**-orthogonality.

Now from above theorems, we can choose the conjugate directions that are con-

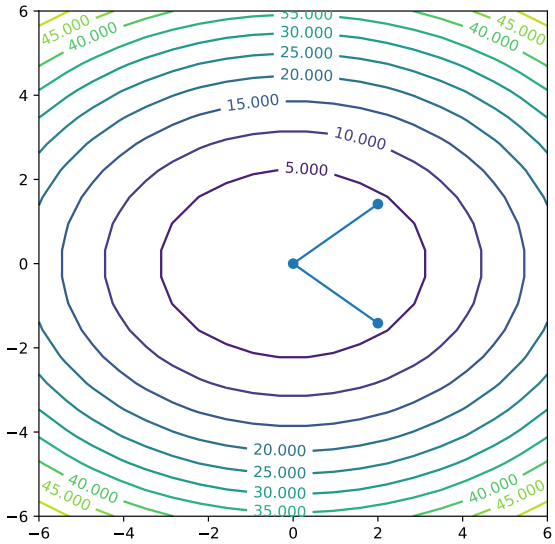

Figure 5.3: A-orthogonality between two vectors

structed by conjugation of the residuals. Since the residuals are orthogonal to the previous search directions, it is graranteed to produce a new linearly independent search direction until the residual is zero. Therefore, the new search direction only depends on the previous one.

The completed conjugate gradient method can be concluded as follows:

- Initialize the conjugate direction $\mathbf{d}_{(0)} := \mathbf{u}_{(0)} = \mathbf{b} - \mathbf{A}\mathbf{x}_{(0)}$ from the linear system.
- Determine the step size based on the $\mathbf{A}$-orthogonality from Equation 5.11.
- Update the gradient from Equation 5.8.
- Update the set of linearly independent vectors $\mathbf{u}_{(i+1)} := \mathbf{u}_{(i)} - \alpha_{(i)}\mathbf{A}\mathbf{d}_{(i)}$
- Determine the current step size β based on the previous one α through Equation 5.13.
- Determine the next conjugate direction based on the new linearly independent vectors, the current step size, and the previous conjugate direction:

$$\mathbf{d}_{(i+1)} := \mathbf{u}_{(i+1)} + \beta_{(i+1)}\mathbf{d}_{(i)}$$

Figure 5.4 shows a simple example of the conjugate gradient method, which achieves the ground truth with two steps.

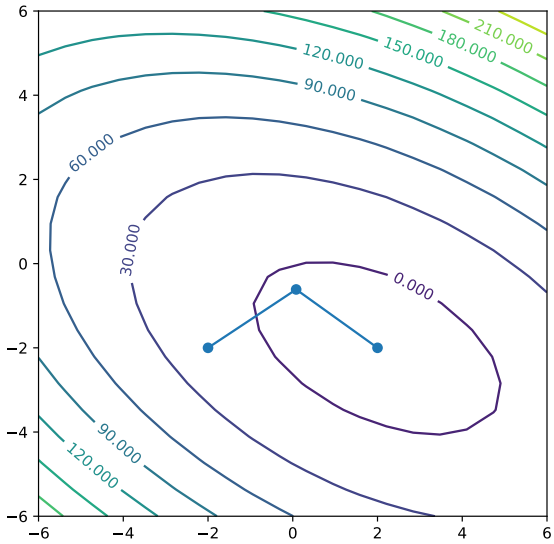

Figure 5.4: Conjugate gradient method

Obviously, for overdetermined linear systems, the traditional conjugate gradient method does not work. We introduce the method of preconditioning [Shewchuk et al., 1994] to solve $\mathbf{Ax} = \mathbf{b}$ indirectly. In the preconditioning method, it demonstrates a positive definite invertible preconditioner $\mathbf{M}$ as the approximation of $\mathbf{A}$, which can convert the problem to solving

$$\mathbf{M}^{-1}\mathbf{Ax} = \mathbf{M}^{-1}\mathbf{b} \tag{5.14}$$

where $\mathbf{M}^{-1}\mathbf{A}$ is not necessarily symmetric. Hence, matrix decomposition $\mathbf{EE}^T = \mathbf{M}$ is performed to transform the problem as $\mathbf{M}^{-1}\mathbf{A}$ and $\mathbf{E}^{-1}\mathbf{AE}^{-T}$ share the same eigenvalues

$$\mathbf{E}^{-1}\mathbf{AE}^{-T}\hat{\mathbf{x}} = \mathbf{E}^{-1}\mathbf{b} \quad \hat{\mathbf{x}} = \mathbf{E}^T\mathbf{x} \tag{5.15}$$

Combining the result of the linear system in Equation 5.15 and the conjugate gradient method, we get this transformed conjugate gradient method [Shewchuk et al., 1994]. The choices of the preconditioner are various. The perfect one is that $\mathbf{M} := \mathbf{A}$ but it is not anymore useful. The diagonal preconditioner, which is easy to calculate in finding a solution, is also a reasonable choice. Another preconditioner is the incomplete Cholesky factor, turning $\mathbf{A} = \mathbf{LL}^T$ with partial Gaussian elimination.

Using the transformed conjugate gradient method to solve linear systems helps to approximate the solution of the overdetermined system. In deep declarative nodes, we can adopt this approach to solve the linear system $Hx_1 = A^T$ and $Hx_2 = B$ iteratively. Comparing with the least-squares method, the preconditioned conjugate gradient method is not so direct since it takes iterations to converge. However, it can

be more stable to get the ideal approximation.

5.2 Rank Deficiency

5.2.1 Orthogonal Matching Pursuit Algorithm

Rank deficiency is also known as the underdetermined system, which means the system has less equations than the unknowns. In a linear system $Ax = b$, there are two problems if A is rank deficient. The first problem is that there may not be an exact solution at all, because the range of A does not span the entire $\mathbb{R}^n$. In this case, we can solve exactly for x only if b is in this subspace. The second problem is that there are actually infinity solutions, since A has a non-empty null space, which means that there is at least an 1-dimensional subspace of vectors x that gives $Ax = 0$. If we use the least-squares solution, the pseudoinverses of A multiply by b, it provides a solution for both problems. For the first problem, it gives the closest x, which is also the one with the smallest $\|b - Ax\|$. At the same time, for the second problem, out of all solutions that share the smallest approximation error and it gives the one with the smallest norm, which is the minimum-norm solution. Therefore, applying the least-squares method to rank deficient may not be able to obtain the best solution for all columns in the matrix.

We consider the orthogonal matching pursuit algorithm (OMP), which is a greedy compressed approach widely used in signal recovery [Mallat and Zhang, 1993]. The goal of this algorithm is to recover the sparse signal vectors from a small number of noisy linear measurements. Supposed we consider a K-sparse signal vector $\mathbf{x}$, which is an n-dimensional vector with at most K non-zero elements, transformed into a smaller m-dimensional $\mathbf{y}$ based on matrix $\mathbf{\Phi}$

$$\mathbf{y} = \mathbf{\Phi x} \tag{5.16}$$

In compressive sensing, we use to set $n > m$. Therefore, the system of Equation 5.16 is underdetermined and the matrix $\mathbf{\Phi}$ is rank deficient. Also, there is no exact solution can be obtained. The OMP algorithm provides a simple but competitive solution for solving this underdetermined system by selecting the best fitting column of the sensing matrix iteratively. Besides, a least-squares optimization is performed in the subspace spanned by all previously picked columns.

We give the details of the OMP algorithm with the classical linear system $Ax = b$ where $A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m$. An optional operation before the OMP iteration is the normalization of columns in the matrix A, which transforms them into unit vectors

$$a_i \leftarrow \frac{a_i}{\|a_i\|_2} \tag{5.17}$$

where a_i are the columns of A. This normalization step is to make sure any correlations between two columns of A is within the range $[-1, 1]$, which also bound the absolute value within 1. In the official OMP algorithm, the first step is the initial-

ization. We set the counter k which keeps track of the number of iterations, i.e. the "column extraction" has occurred. The residual r_k is set as b which is the key in extracting the "important columns" of A. Important columns are defined as the column in A that has the largest absolute value of correlation with the residue vector r_k. We also define the index set Λ_1 at this stage as $\varnothing$. Secondly, we complete the iteration in the main loop. The important column is extracted through $\lambda_k = \arg\max_j \left|\left\langle a_j, r_k \right\rangle\right|$ for general k, where λ is the column index of A and a_j is the j-column of A. Since it can produce multiple results due to duplicate columns in A, some extra preprocessing steps have to be taken before the main loop. Meanwhile, the index set is augmented with the newly selected column. Now the estimation x_i is obtained by solving the least-squares problem

$$x_k = \arg\min_x \left\| A_{\Lambda_k} x - b \right\|_2 \tag{5.18}$$

which minimizes the error between the selected important columns of A and b. Next we update the residual r_{k+1} by $r_k - \hat{b}_k$, where $\hat{b}_k = A_{\lambda_k} x_k$ is the approximation of the given b using the basis A_{λ_k} and the coefficients x_k. In this step, we aim make sure that the extracted columns are not be selected again in the next iteration.

Overall, the completed OMP algorithm can be represented as follows:

Algorithm 1: The OMP algorithm

Result: $\mathbf{x}_{k_{\text{Max}}}$
Initialization: $\mathbf{r}_1 = \mathbf{b}, \Lambda_0 = \varnothing$;
Remove duplicate columns in $\mathbf{A}$ (make $\mathbf{A}$ full rank) ;
normalize all columns of $\mathbf{A}$ to unit L_2 norm;
for $k = 1$ *to* k_{Max} **do**
  $\lambda_k = \arg\max_i \left|\left\langle \mathbf{a}_j, \mathbf{r}_k \right\rangle\right|$;
  $\Lambda_k = \Lambda_{k-1} \cup \{\lambda_k\}$;
  $\mathbf{x}_k = \arg\min_{\mathbf{x}} \left\| \mathbf{A}_{\Lambda_k} \mathbf{x} - \mathbf{b} \right\|_2$;
  $\hat{\mathbf{b}}_k = \mathbf{A}_{\lambda_k} \mathbf{x}_k$;
  $\mathbf{r}_{k+1} = \mathbf{r}_k - \hat{\mathbf{b}}_k$
end

After a fixed number of iterations, the algorithm stops and is converged to the optimal solution. In deep declarative nodes, when we get the underdetermined system we can apply this effective algorithm to approximate the gradient. Moreover, a more stable and faster extension proposed by Donoho et al. [2012] allows more than one coefficients and takes a fixed number of stages to converge. Both algorithms perform well in solving the underdetermined system theoretically.

5.2.2 Strategies in Applications

Rank deficiency problems occur when the constraints are not all second-order differentiable or we do not have enough constraints to determine the solution. Hence in the application of the deep declarative network, it is essential to build the nodes with sufficient constraints from the problem.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

Figure 5.5: An example of the Sudoku puzzle [Commons, 2020]

A very classical constrained task is the Sudoku problem, which involves multiple rules as constraints. Figure 5.5 is a simple example of the 9×9 Sudoku puzzle. In general, the model treats the puzzle as a matrix and constraints between rows and columns, which are represented as rows and columns interactions. Also, some rules in each 3×3 block are defined as blocks interactions. Simonis [2005] provides an understanding of the Sudoku puzzle from constraint problems, which constructs a constraint model with the extension of redundant constraints for better performance. Therefore, building an approximate constraint model with sufficient constraints is important. Adding extra redundant constraints may help the model produce more accurate results. However, we still need to trade off the constraints between overdetermined and underdetermined solutions.

5.3 Non-convex Problems

Non-convex problems are usually NP-hard to solve due to the potentially various local minima, saddle points, very flat regions and widely varying curvature [De, 2017]. Generally, non-convex problems have multiple local minima, but we aim to find the global optimal solution. There are plenty of methods are proposed based on different non-convex cases. For example, if the problem is non-convex but quasiconvex, it can be solved by transforming it as an equivalent convex problem using geometric programming [Boyd et al., 2007]. Otherwise, a dual problem with guaranteed convexity will be demonstrated to indirectly solve the original non-convex problem. The essence of these methods is solving a convex problem which is equivalent to the non-convex one. Apart from this, we introduce the method based on the optimality conditions of the optimization, the non-linear Lagrangian.

There are multiple efficient solution algorithms for constrained optimization problems based on the optimality conditions we mentioned in Section 2.1.3. We usually use the classic Lagrange multipliers method [Bertsekas, 2014] and the penalty function methods [Courant, 1962], which are under the assumption of convex situation.

For non-convex problems, several extensions on non-linear Lagrangian theory are proposed Li [1995, 1997]; Goh and Yang [1997]; Xu [1997]. In this section, we represent the fundamental properties of non-linear Lagrangian in solving non-convex constrained optimization problems.

In non-linear Lagrangian, the Lagrange function makes the duality gap always zero regardless of convexity and it is defined based on the weighted Chebyshev norm [Goh and Yang, 1997]. Chebyshev distance is a metric based on the uniform norm, which is also known as the L_∞ metric [Cantrell, 2000]. In the weighted version, a weight vector is introduced and the whole function is a mapping from $\mathbb{R}^{m+1}$ to $\mathbb{R}_+$. For a vector $\mathbf{z} \in \mathbb{R}^{m+1}$, the weighted Chebyshev norm can be represented as

$$\xi_{\mathbf{e}}(\mathbf{z}) = \max_{0 \leq i \leq m} \left\{ \frac{z_i}{e_i} \right\} \tag{5.19}$$

where $\mathbf{e}$ is the weight vector.

Now the Lagrangian of the constrained optimization problem defined in Equation 2.2 can be represented as

$$\mathcal{L}(\mathbf{x}, \mathbf{e}) = \xi_{\mathbf{e}}(\mathbf{f}(\mathbf{x})) = \max_{0 \leq i \leq m} \left\{ \frac{f_i(\mathbf{x})}{e_i} \right\} \tag{5.20}$$

where $\mathbf{f}(\mathbf{x})$ is the set of both the objective function and constraints.

Same as the traditional Lagrangian, we transfer the constrained optimization problems to unconstrained. Also, we present the equivalent optimality conditions in terms of the unconstrained problems, which leads to the algorithm handling both convex and non-convex constrained optimization problems.

Definition 5.1. (Sufficient and Necessary condition for optimality without convexity)Goh and Yang [1997]
Let x^* be the global optimal solution of the constrained optimization problems defined in the form of Equation 2.2, and let the weight vector $\theta_0 = f(x^*) > 0$. Then the solution x^0 solves the problem if and only if it solves the unconstrained problem

$$\text{minimize}\, \mathcal{L}(\mathbf{x}, \hat{\boldsymbol{\theta}}) = \text{minimize} \max_{0 \leq i \leq m} \left\{ \frac{f_i(\mathbf{x})}{\theta_i} \right\}$$

where $\hat{\boldsymbol{\theta}} = [\theta_0, \theta_1, \cdots, \theta_m]$

The above definition can be proved through the properties of the Chebyshev norm, which has the minimum value 1 in the Appendix B.1. In deep declarative nodes, applying this new non-linear Lagrangian in solving the gradient for constrained optimization problems can avoid the non-regular points in the solution. Apart from the solutions of other non-regular points scenarios, the non-linear Lagrangian solves the gradient exactly instead of approaching an approximation.

5.4 Future Work of the Solution for Non-regular Points

It is obvious that there are still many different non-regular solution scenarios, and not all of them can be solved perfectly through the approximation. In our scenarios above, the intersection of constraints is a single point. Therefore, we can easily find various approximations around this point as our solution. However, if the intersection of constraints is a flat region that is filled with non-regular points, it is difficult to select an appropriate approximation under this circumstance since the closest regular solution may still have a large residual error. Future work can explore various possibilities of the intersection and consider algorithms for the exact solution.

There is still no general approach for all these non-regular cases, but in the construction of the network, we need to fit different scenarios with a regularization representation. We hope that a general but efficient algorithm can be addressed in the future.

5.5 Summary

In Chapter 5, we discuss the solution for different non-regular solution scenarios. We can separate these methods into two classes: heuristic solution and exact solutions. The former focuses on transferring the problem into a regular one with the solution which approximates the original one based on the least-squares method. The later defines a new type of non-linear Lagrangian for solving the gradient regardless of the convexity and changes the optimality conditions. Both classes are useful and effective in solving the non-regular solutions and many insightful future works can be extended based on them.

Chapter 6

Conclusion

In this thesis, we provided readers a thorough overview of constrained optimization in the deep declarative network: the multiple constraints nodes (PART I) and solutions for non-regular points (PART II).

In Chapter 2, we walked through the theory of numerical optimization, which is the theoretical background of the deep declarative network. A system explanation of the optimality conditions of constrained optimization problems under the assumption of convexity: the objective function is differentiable at the optimal point and twice differentiable on the definition set. Solutions to unconstrained and constrained optimization problems are discussed, which are obtained based on the optimality conditions with Lagrange multipliers. Apart from the traditional deep neural network, we introduced the related works in the differentiable network. It uses the properties of back-propagation with differentiable optimization problems to construct the nodes.

In Chapter 3, we covered all contents of the deep declarative network. More specifically, the properties and functions of deep declarative nodes. We introduced how the deep declarative network works and its learning progress based on the end-to-end learnable declarative nodes. We later presented the back-propagation and corresponding gradient solution of regular points through declarative nodes in different scenarios: unconstrained, equality constrained and inequality constrained. We also showed various examples of our implementation of deep declarative nodes in different scenarios. We hope our works in this new type of differentiable network will lead to a new generation of deep neural networks.

In PART II, the key questions we wanted to answer are: Is the solution of the deep declarative nodes always regular? If the solution is non-regular, how do we find the gradient then perform the back-propagation?

In Chapter 4, we addressed the problems in regular deep declarative nodes, which is based on the assumption of convexity in the optimality conditions. We separated the problems into three scenarios: the overdetermined system, rank deficiency problems and non-convex cases. We also provided the corresponding example for each scenario, which is not able to obtain the gradient directly. Lastly, we discussed multiple previous works in solving non-regular solution problems.

In Chapter 5, we showed that we can deal with the different non-regular solutions with both approximation and theoretical based optimality methods. For overdeter-

mined systems and rank deficiency problems, we introduced approaches based on the least-squares method and iterative gradient descent. Both of these methods try to approximate the closest solution with minimum residual errors. For non-convex cases, we proposed a different algorithm for solving the constrained optimization problems based on the optimality conditions regardless of the convexity. This technique can solve the problem exactly with the weighted Chebyshev norm and non-linear Lagrangian. We believe these methods are useful for handling all range of solutions in the deep declarative network.

All together, we are really excited about the progress that has been made in this field for the past years and are glad to contribute to this field. At the same time, we also hope to encourage more researchers to work on the deep declarative network applications, or apply the deep declarative network to new domains. We believe that it will lead us towards building better constrained problem solvers and hope to see these ideas implemented and developed in industry applications.

Bibliography

Agrawal, A.; Amos, B.; Barratt, S.; Boyd, S.; Diamond, S.; and Kolter, J. Z., 2019. Differentiable convex optimization layers. In *Advances in neural information processing systems*, 9562–9574. (cited on page 20)

Allen-Zhu, Z. and Hazan, E., 2016. Variance reduction for faster non-convex optimization. In *International conference on machine learning*, 699–707. (cited on page 44)

Amos, B. and Kolter, J. Z., 2017. Optnet: Differentiable optimization as a layer in neural networks. *arXiv preprint arXiv:1703.00443*, (2017). (cited on pages 2, 15, and 20)

Amos, B. and Yarats, D., 2019. The differentiable cross-entropy method. *arXiv preprint arXiv:1909.12830*, (2019). (cited on page 20)

Anton, H. and Rorres, C., 2013. *Elementary linear algebra: applications version*. John Wiley & Sons. (cited on page 44)

Bard, J. F., 1998. *Practical Bilevel Optimization: Algorithms and Applications*, vol. 30. Springer Science & Business Media. (cited on page 24)

Barrodale, I. and Roberts, F., 1974. Solution of an overdetermined system of equations in the l 1 norm [f4]. *Communications of the ACM*, 17, 6 (1974), 319–320. (cited on page 44)

Bertsekas, D. P., 2014. *Constrained optimization and Lagrange multiplier methods*. Academic press. (cited on pages 14, 17, 19, 27, and 55)

Borrodale, I. and Phillips, C., 1975. Solution of an overdetermined system of linear equations in the chebyshev norm [f4](algorithm 495). *ACM Trans. Math. Software*, 1, 3 (1975), 264–270. (cited on page 44)

Boyd, S.; Boyd, S. P.; and Vandenberghe, L., 2004. *Convex optimization*. Cambridge university press. (cited on pages 19 and 27)

Boyd, S.; Kim, S.-J.; Vandenberghe, L.; and Hassibi, A., 2007. A tutorial on geometric programming. *Optimization and engineering*, 8, 1 (2007), 67. (cited on page 55)

Campbell, D.; Liu, L.; and Gould, S., 2020. Solving the blind perspective-n-point problem end-to-end with robust differentiable geometric optimization. *arXiv preprint arXiv:2007.14628*, (2020). (cited on page 20)

Cantrell, C. D., 2000. *Modern mathematical methods for physicists and engineers*. Cambridge University Press. (cited on page 56)

Chen, B.; Parra, A.; Cao, J.; Li, N.; and Chin, T.-J., 2020. End-to-end learnable geometric vision by backpropagating pnp optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8100–8109. (cited on page 20)

Chen, R. T. and Duvenaud, D. K., 2019. Neural networks with cheap differential operators. In *Advances in Neural Information Processing Systems*, 9961–9971. (cited on page 21)

Commons, W., 2020. File:sudoku-by-l2g-20050714.svg — wikimedia commons, the free media repository. https://commons.wikimedia.org/w/index.php?title=File:Sudoku-by-L2G-20050714.svg&oldid=473124111. [Online; accessed 6-November-2020]. (cited on pages xiv and 55)

Courant, R., 1962. *Calculus of Variations: With Supplementary Notes and Exercises, 1945-1946*. Courant Institute of Mathematical Sciences, New York University. (cited on page 55)

Darvishi, M. and Khosro-Aghdam, R., 2006. Symmetric successive overrelaxation methods for rank deficient linear systems. *Applied mathematics and computation*, 173, 1 (2006), 404–420. (cited on page 44)

Datta, B. N., 2010. *Numerical linear algebra and applications*, vol. 116. Siam. (cited on page 47)

De, S. C., 2017. Lecture notes in advanced machine learning systems. (cited on page 55)

Debye, P., 1909. Näherungsformeln für die zylinderfunktionen für große werte des arguments und unbeschränkt veränderliche werte des index. *Mathematische Annalen*, 67, 4 (1909), 535–558. (cited on page 16)

Dennis, J. E., Jr and Moré, J. J., 1977. Quasi-newton methods, motivation and theory. *SIAM review*, 19, 1 (1977), 46–89. (cited on page 19)

Donoho, D. L., 2005. Neighborly polytopes and sparse solutions of underdetermined linear equations. (2005). (cited on page 44)

Donoho, D. L., 2006. For most large underdetermined systems of linear equations the minimal l1-norm solution is also the sparsest solution. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 59, 6 (2006), 797–829. (cited on page 44)

Donoho, D. L.; Tsaig, Y.; Drori, I.; and Starck, J.-L., 2012. Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. *IEEE transactions on Information Theory*, 58, 2 (2012), 1094–1121. (cited on page 54)

Goh, C. and Yang, X., 1997. A sufficient and necessary condition for nonconvex constrained optimization. *Applied Mathematics Letters*, 10, 5 (1997), 9–12. (cited on page 56)

Goodfellow, I.; Bengio, Y.; Courville, A.; and Bengio, Y., 2016. *Deep learning*, vol. 1. MIT press Cambridge. (cited on page 20)

Gould, S.; Fernando, B.; Cherian, A.; Anderson, P.; Cruz, R. S.; and Guo, E., 2016. On differentiating parameterized argmin and argmax problems with application to bi-level optimization. *arXiv preprint arXiv:1607.05447*, (2016). (cited on pages 15 and 19)

Gould, S.; Hartley, R.; and Campbell, D., 2019. Deep declarative networks: A new hope. *arXiv preprint arXiv:1909.04866*, (2019). (cited on pages xii, xiii, 1, 2, 4, 23, 24, 25, 29, 38, 39, 40, 41, 42, 43, and 70)

He, K.; Zhang, X.; Ren, S.; and Sun, J., 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778. (cited on page 1)

Hestenes, M. R., 1969. Multiplier and gradient methods. *Journal of optimization theory and applications*, 4, 5 (1969), 303–320. (cited on page 19)

Hestenes, M. R.; Stiefel, E.; et al., 1952. Methods of conjugate gradients for solving linear systems. *Journal of research of the National Bureau of Standards*, 49, 6 (1952), 409–436. (cited on page 49)

Jiang, H., 1999. Global convergence analysis of the generalized newton and gauss-newton methods of the fischer-burmeister equation for the complementarity problem. *Mathematics of Operations Research*, 24, 3 (1999), 529–543. (cited on page 19)

Li, D., 1995. Zero duality gap for a class of nonconvex optimization problems. *Journal of Optimization Theory and Applications*, 85, 2 (1995), 309–324. (cited on page 56)

Li, D., 1997. Saddle point generation in nonlinear nonconvex optimization. *Nonlinear Analysis: Theory, Methods & Applications*, 30, 7 (1997), 4339–4344. (cited on page 56)

Maclaurin, D.; Duvenaud, D.; and Adams, R. P., 2015. Autograd: Effortless gradients in numpy. In *ICML 2015 AutoML Workshop*, vol. 238, 5. (cited on page 29)

Mallat, S. G. and Zhang, Z., 1993. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on signal processing*, 41, 12 (1993), 3397–3415. (cited on page 53)

Newton, I. and Colson, J., 1736. *The Method of Fluxions and Infinite Series; with Its Application to the Geometry of Curve-lines... Translated from the Author's Latin Original Not Yet Made Publick. To which is Subjoin'd a Perpetual Comment Upon the Whole Work... by J. Colson*. (cited on page 16)

Niemeyer, M.; Mescheder, L.; Oechsle, M.; and Geiger, A., 2020. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3504–3515. (cited on page 20)

Nocedal, J. and Wright, S., 2006. *Numerical optimization*. Springer Science & Business Media. (cited on pages 7, 8, 9, 12, and 13)

Oman, G., 2017. A short proof of the bolzano-weierstrass theorem. *The College Mathematics Journal*, (2017). (cited on page 8)

Panier, E. R.; Tits, A. L.; and Herskovits, J. N., 1988. A qp-free, globally convergent, locally superlinearly convergent algorithm for inequality constrained optimization. *SIAM Journal on Control and Optimization*, 26, 4 (1988), 788–811. (cited on page 19)

Powell, M. J., 1969. A method for nonlinear constraints in minimization problems. *Optimization*, (1969), 283–298. (cited on page 19)

Qi, H.-D. and Qi, L., 2000. A new qp-free, globally convergent, locally superlinearly convergent algorithm for inequality constrained optimization. *SIAM Journal on Optimization*, 11, 1 (2000), 113–132. (cited on page 19)

Rao, R., 2019. 9 million sudoku puzzles and solutions. https://www.kaggle.com/rohanrao/sudoku. [Online; accessed 6-November-2020]. (cited on pages xiii, 1, and 2)

Reddi, S. J.; Hefny, A.; Sra, S.; Poczos, B.; and Smola, A., 2016. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, 314–323. (cited on page 44)

Robbins, H. and Monro, S., 1951. A stochastic approximation method. *The annals of mathematical statistics*, (1951), 400–407. (cited on page 44)

Rocktäschel, T. and Riedel, S., 2017. End-to-end differentiable proving. In *Advances in Neural Information Processing Systems*, 3788–3800. (cited on page 2)

Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; and Chen, L.-C., 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 4510–4520. (cited on page 1)

Shewchuk, J. R. et al., 1994. An introduction to the conjugate gradient method without the agonizing pain. (cited on page 52)

Simonis, H., 2005. Sudoku as a constraint problem. In *CP Workshop on modeling and reformulating Constraint Satisfaction Problems*, vol. 12, 13–27. Citeseer. (cited on page 55)

Simonyan, K. and Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, (2014). (cited on page 1)

Trefethen, L. N. and Bau III, D., 1997. *Numerical linear algebra*, vol. 50. Siam. (cited on page 44)

Wang, J., 1997. Recurrent neural networks for computing pseudoinverses of rank-deficient matrices. *SIAM Journal on Scientific Computing*, 18, 5 (1997), 1479–1493. (cited on page 44)

Wang, P.-W.; Donti, P. L.; Wilder, B.; and Kolter, Z., 2019. Satnet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver. *arXiv preprint arXiv:1905.12149*, (2019). (cited on pages 2 and 20)

Watson, G. A., 1979. The minimax solution of an overdetermined system of non-linear equations. *IMA Journal of Applied Mathematics*, 23, 2 (1979), 167–180. (cited on page 44)

Wu, X.; Silva, B.; and Yuan, J., 2004. Conjugate gradient method for rank deficient saddle point problems. *Numerical Algorithms*, 35, 2-4 (2004), 139–154. (cited on page 44)

Xu, Z., 1997. Local saddle points and convexification for nonconvex optimization problems. *Journal of Optimization Theory and Applications*, 94, 3 (1997), 739–746. (cited on page 56)

Yang, F.; Yang, Z.; and Cohen, W. W., 2017. Differentiable learning of logical rules for knowledge base reasoning. In *Advances in Neural Information Processing Systems*, 2319–2328. (cited on page 2)

Yeniay, Ö., 2005. Penalty function methods for constrained optimization with genetic algorithms. *Mathematical and computational Applications*, 10, 1 (2005), 45–56. (cited on page 18)

Appendix

A An Overview of Numerical Optimization

A.1 Theory of Optimization

A.1.1 Proof of Theorem 2.3

Proof. Let

$$m := \inf\{f(x) : x \in \Omega\}$$

By the definition of m we may pick a sequence $\{x_k\} \subset \Omega$ with $f(x_k) \to m$ as $k \to \infty$. Because Ω is compact, we can extract a convergent subsequence $\left\{x_{k_j}\right\}$ from $\{x_k\}$. Let $x^* \in \Omega$ denote the limit point of $\left\{x_{k_j}\right\}$. Since f is continuous, $f(x^*) = \lim_{j\to\infty} f\left(x_{k_j}\right) = m$. Thus m is finite and x^* is a global minimizer of f on Ω.

When $\Omega = \mathbb{R}^n$, we need to impose conditions on f at infinity to guarantee the existence of a global minimizer. □

A.1.2 Proof of Theorem 2.4

Proof. Let $m := \inf\{f(x) : x \in \mathbb{R}^n\}$, and take a sequence $\{x_k\}$ such that

$$f(x_k) \to m \quad \text{as } k \to \infty.$$

Since f is coervice, $\{x_k\}$ must be bounded; otherwise it has a subsequence $\left\{x_{k_j}\right\}$ with $\left\|x_{k_j}\right\| \to \infty$ as $j \to \infty$, and hence $m = \lim_{j\to\infty} f\left(x_{k_j}\right) = +\infty$, a contradition.

Thus there is $r > 0$ such that

$$\{x_k\} \subset \{x \in \mathbb{R}^n : \|x_k\| \le r\}.$$

Because $\{x \in \mathbb{R}^n : \|x\| \le r\}$ is compact, $\{x_k\}$ has a convergent subsequence $\left\{x_{k_j}\right\}$

with $x_{k_j} \to x^*$ as $j \to \infty$. In view of the continuity of f, we have

$$f(x^*) = \lim_{j\to\infty} f\left(x_{k_j}\right) = m$$

Therefore m is finite and f achieves its minimum on $\mathbb{R}^n$ at x^* □

A.1.3 Proof of Theorem 2.6

Proof. We may assume that $\alpha > f_* := \inf\{f(x) : x \in \mathbb{R}^n\}$. Let $\{x_k\}$ be a minimizing sequence for f, i.e.

$$f(x_k) \to f_* \quad \text{as } k \to \infty$$

Then there is an N such that $f(x_k) \leq \alpha$ for all $k \geq N$, that is, $x_k \in D$ for all $k \geq N$. Since D is compact, $\{x_k\}_{k=N}^{\infty}$ has a convergent subsequence $\left\{x_{k_j}\right\}$ with $x_{k_j} \to x_* \in D$ as $j \to \infty$. In view of the lower semi-continuity of f, we have

$$f(x_*) \leq \lim_{j\to\infty} f\left(x_{k_j}\right) = f_*$$

By the definition of f_* we must have $f(x_*) = f_*$. Therefore f achieves its minimum on $\mathbb{R}$ at x_*. □

A.1.4 Proof of Theorem 2.8

Proof. (NC1): First, recall that for any $v \in \mathbb{R}^n$ there holds

$$v^T \nabla f(x^*) = D_v f(x^*) = \lim_{t \searrow 0} \frac{f(x^* + tv) - f(x^*)}{t}.$$

Since x^* is a local minimizer, we have

$$f(x^* + tv) - f(x^*) \geq 0 \quad \text{for small } |t|.$$

Therefore

$$v^T \nabla f(x^*) \geq 0 \quad \text{for all } v \in \mathbb{R}^n.$$

In particular this implies $(-v)^T \nabla f(x^*) \geq 0$ and thus

$$v^T \nabla f(x^*) \leq 0 \quad \text{for all } v \in \mathbb{R}^n.$$

Therefore $v^T \nabla f(x^*) = 0$ for all $v \in \mathbb{R}^n$. Taking $v = \nabla f(x^*)$ gives $\|\nabla f(x^*)\|^2 = 0$ which shows that $\nabla f(x^*) = 0$ □

Proof. (NC2): Recall that for any $v \in \mathbb{R}^n$ and small $t > 0$ there is $0 < s < 1$ such that

$$f(x^* + tv) = f(x^*) + tv^T \nabla f(x^*) + \frac{1}{2} t^2 v^T \nabla^2 f(x^* + stv)\, v.$$

Since x^* is a local minimizer of f, we have $f(x^* + tv) \geq f(x^*)$ and $\nabla f(x^*) = 0$ by (NC1). Therefore

$$\frac{1}{2}t^2 v^T \nabla^2 f(x^* + stv)\, v = f(x^* + tv) - f(x^*) \geq 0.$$

This implies that

$$v^T \nabla^2 f(x^* + stv)\, v \geq 0.$$

Taking $t \to 0$ gives

$$v^T \nabla^2 f(x^*)\, v \geq 0 \quad \text{for all } v \in \mathbb{R}^n$$

i.e. $\nabla^2 f(x^*)$ is semi-definite. □

Proof. (SC1): Since $\nabla^2 f(x$ is continuous and $\nabla^2 f(x^*) \geq 0$, we can find $r > 0$ such that

$$B_r(x^*) \subset \Omega \quad \text{and} \quad \nabla^2 f(x) > 0 \text{ for all } x \in B_r(x^*).$$

By Taylor's formula we have

$$f(x) = f(x^*) + \nabla f(x^*) \cdot (x - x^*) + \frac{1}{2}(x - x^*)^T \nabla^2 f(\hat{x})(x - x^*)$$

where $\hat{x} := x^* + t(x - x^*)$ for some $0 < t < 1$.

It is clear that $\hat{x} \in B_r(x^*)$ and hence $\nabla^2 f(\hat{x}) > 0$ which implies that

$$(x - x^*)^T \nabla^2 f(\hat{x})(x - x^*) > 0 \quad \text{for } x \neq x^*$$

Consequently

$$f(x) > f(x^*) + \nabla f(x^*) \cdot (x - x^*)$$

for all $x \in B_r(x^*)$ with $x \neq x^*$.

Since $\nabla f(x^*) = 0$, we can obtain $f(x) > f(x^*)$ for all $x \in B_r(x^*)$ with $x \neq x^*$. □

A.1.5 Proof of Lemma 2.10

Proof. For $d \in T_{x^*}\mathscr{F}$, we have $z_k \subset \mathscr{F}$ and t_k such that

$$z_k \to x^*, \quad 0 < t_k \to 0 \quad \text{and} \quad \frac{z_k - x^*}{t_k} \to d$$

as $k \to \infty$. As $f(x^*) \leq f(z_k)$, by Taylor's formula we have

$$\begin{aligned} f(x^*) \leq f(z_k) &= f(x^* + (z_k - x^*)) \\ &= f(x^*) + (z_k - x^*)^T \nabla f(x^*) + \frac{1}{2}(z_k - x^*)^T \nabla^2 f(\hat{z}_k)(z_k - x^*) \end{aligned}$$

where $\hat{z}_k$ is a point on the line segment joining x^* and z_k. This implies that

$$0 \leq \left(\frac{z_k - x^*}{t_k}\right)^T \nabla f\left(x^*\right) + \frac{1}{2}\left(z_k - x^*\right)^T \nabla^2 f\left(\hat{z}_k\right)\left(\frac{z_k - x^*}{t_k}\right)$$

Letting $k \to \infty$ gives $d^T \nabla f\left(x^*\right) \geq 0$ □

A.2 Solution of Unconstrained and Constrained Optimization Problems

A.2.1 Proof of Equation 2.3

Proof. Firstly, for any optimal y, according to the first-order optimality condition, we have

$$\frac{df(x,y)}{dy} = \mathbf{0} \in \mathbb{R}^{1\times m}$$

Then from the implicit function theorem, rearranging and differentiating both sides we have

$$\begin{aligned} \mathrm{D}(\frac{df(x,y)}{dy})^T &= \mathbf{0} \in \mathbb{R}^{m\times n} \\ &= \frac{\partial^2}{\partial x \partial y} f(x,y) + \frac{\partial^2}{\partial y^2} f(x,y) \frac{dy(x)}{dx} \\ \frac{dy(x)}{dx} &= -[(\frac{\partial^2}{\partial y^2}) f(x,y)]^{-1} (\frac{\partial^2}{\partial x \partial y}) f(x,y) \end{aligned}$$

□

A.2.2 Proof of Equation 2.4 [Gould et al., 2019]

Proof. According to the definition of Lagrange multipliers, we can define the Lagrangian:

$$\mathcal{L}(x,y,\lambda) = f(x,y) - \sum_{i=1}^{p} \lambda_i (A_i y_i - b_i)$$

We are going to find the stationary point (y,λ) for this lagrangian. Therefore, we calculate the derivative of $\mathcal{L}$ with respect to y and λ separately:

$$\frac{\partial}{\partial y} f(x,y) - \sum_{i=1}^{p} \lambda_i \frac{\partial}{\partial y}(A_i y_i - b_i) = 0 \tag{1}$$

$$A\boldsymbol{y} - \boldsymbol{b} = 0 \tag{2}$$

Since y is the optimal point, we have $\frac{\partial}{\partial y} f(x,y) = 0$, which can be an unconstrained problem or it is orthogonal to the constraint surface. For unconstrained cases, we can

set $\lambda = 0$ directly. For the orthogonal case, from Equation 1, we have

$$\frac{\partial}{\partial y} f(x,y) = \sum_{i=1}^{p} \lambda_i \frac{\partial}{\partial y}(A_i y_i - b_i) = \lambda^T A$$

Now we are going to calculate the derivative of the Lagrangian with respect to x for both 1 and 2:

$$\frac{\partial^2}{\partial x \partial y} f(x,y) + \frac{\partial^2}{\partial y^2} f(x,y) \mathrm{D}y - \frac{\partial}{\partial y}(A\boldsymbol{y} - \boldsymbol{b})^T \mathrm{D}\lambda = 0 \tag{3}$$

$$\frac{\partial}{\partial x}(A\boldsymbol{y} - \boldsymbol{b}) + \frac{\partial}{\partial y}(A\boldsymbol{y} - \boldsymbol{b}) \mathrm{D}y = 0 \tag{4}$$

Solving 3 and 4, we get:

$$\mathrm{D}y(x) = \left(H^{-1} A^T \left(A H^{-1} A^T \right)^{-1} A H^{-1} - H^{-1} \right) B$$

where

$$H = \frac{\partial^2}{\partial y^2} f(x,y), \quad B = \frac{\partial^2}{\partial x \partial y} f(x,y)$$

□

B Solutions of Non-regular Points

B.1 Proof of Definition 5.1

Proof. It is clear that the minimum value of the unconstrained optimization problem in this definition is

$$\min_{\mathbf{x} \in X} \max_{0 \le i \le m} \left\{ \frac{f_i(\mathbf{x})}{\theta_i} \right\} = 1 \tag{5}$$

If x^0 solves the original problem in Equation 2.2, it must be in the feasible set and $f(x^0) = f(x^*)$. Hence

$$\max_{0 \le i \le m} \left\{ \frac{f_i\left(x^0\right)}{\theta_i} \right\} = 1 \tag{6}$$

Conversely, if x^0 does not solve the original problem in Equation 2.2, then either x^0 is infeasible, or x^0 if feasible and $f(x^0) > f(x^*)$. If x^0 is infeasible, then $\exists j \in \{1, 2, \ldots, m\}$ such that $f_j\left(\mathbf{x}^0\right) > \theta_j$ where f_j is the j-th constraints, implying that

$$\max_{0 \le i \le m} \left\{ \frac{f_i\left(\mathbf{x}^0\right)}{\theta_i} \right\} > 1 \tag{7}$$

If x^0 is feasible then $f(x^0) > f(x^*)$, and the inequality in Equation 7 still holds. Thus in both cases, x^0 does not solve the unconstrained problem in this definition. □