FINDING THE EXTREMA OF CONTINUOUS PIECEWISE LINEAR FUNCTIONS

by

Özge Arslan

A Thesis Submitted to the
Graduate School of Engineering
in Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in

Industrial Engineering

Koç University

January, 2011

Koç University

Graduate School of Sciences and Engineering

This is to certify that I have examined this copy of a master's thesis by

Özge Arslan

and have found that it is complete and satisfactory in all respects,
and that any and all revisions required by the final
examining committee have been made.

Committee Members:

_____

Metin Türkay, Assoc. Prof. (Advisor)

_____

Sibel Salman, Asst. Prof.

_____

Serdar Kozat, Asst. Prof.

Date: _____

This thesis is dedicated to my parents, grand parents, sister and Can

ABSTRACT

Development of accurate models and efficient solution algorithms for piecewise linear functions (PWL) attracted a lot of attention due to wide range of application areas. There have been several attempts to develop exact and efficient optimization models, including mixed integer programming formulations, branching based algorithms, and linear programming formulations for PWL functions with special structure. All of these attempts provide useful insights; however, these formulations are either complex or target a very specific class of PWL functions.

In this thesis, we present a novel linear programming formulation to find the extrema of continuous PWL functions. We first prove that our formulation finds the extrema of any continuous PWL function (convex, concave, or non-convex) exactly. While developing this formulation we make use of two facts: first, simplex algorithm moves along the extreme points of the feasible region while searching for the optimal solution and extrema of any continuous PWL function lies at its break points. We develop a linear programming formulation with a special feasible region such that the extreme points of this region overlap with the break points of the corresponding PWL function. This property enables the simplex algorithm to find the extrema of a PWL function exactly. Our formulation can be solved with a general purpose LP solver, since binary variables are not needed. Furthermore, our formulation has less number variables and constraints than existing formulations in the literature. As a result, all of these properties decrease the complexity of our formulation and the CPU time to find the optimal solution. We support our findings by computationally benchmarking our formulation with the most common formulations in the literature. Finally, we show that our formulation can also be used to find the extrema of separable piecewise continuous linear functions.

ÖZETÇE

Geniş uygulama alanları olması nedeniyle parçalı doğrusal fonksiyonların etkin tekniklerle modellenmesi her zaman dikkat çeken bir konu olmuştur. Bu amaçla çeşitli modelleme teknikleri geliştirilmiştir; tamsayı karışık programlama modelleri, dallara ayırma odaklı yöntemler, özel yapıda parçalı doğrusal fonksiyonlar için lineer programlama modelleri bu tekniklere örnek olarak gösterilebilir. Bütün bu çalışmalar faydalı olmakla birlikte, ya kompleks formüller olmuş veya kısıtlı bir sınıf parçalı doğrusal fonksiyonları modellemeye yoğunlaşmıştır.

Biz bu çalışmada sürekli parçalı doğrusal fonksiyonların uç değerinin başka kısıtlar olmadığı hallerde bulunmasını sağlayan yeni bir lineer programlama modeli sunmaktayız. Öncelikle modelimizin sürekli parçalı doğrusal fonksiyonların uç değerini tam bir şekilde bulduğunu gösteriyoruz. Modelimizi geliştirirken iki gerçekten faydalanıyoruz. Bunlardan ilki simplex algoritmasının en iyi sonucu ararken olanaklı bölgenin uç noktalarında gezdiği, ikincisiyse sürekli parçalı doğrusal bir fonksiyonun uç değerinin her zaman kırılma noktalarında bulunduğudur. Buradan yola çıkarak, geçerli bölgesinin uç noktaları tanımladığı fonksiyonun kırılma noktalarına tamamen denk gelen bir lineer programlama modeli geliştirdik. Böylelikle simplex algoritmasının parçalı doğrusal fonksiyonun uç değerini tam olarak bulmasını sağladık. Modelimiz ikili değişkenler içermediğinden genel amaçlı bir lineer programlama çözümleyicisiyle çözülebilmektedir. Ayrıca modelimiz literatürde yer alan modellerden daha az kısıt ve değişken içermektedir. Sonuç olarak, bütün bu özellikler modelimizin kompleksitesini azaltırken çözüme ulaşma süresini de kısaltmaktadır. Modelimizi literatürdeki en yaygın modellerle hesaplamalı olarak kıyaslayarak da bulgularımızı desteklemekteyiz. Son olarak, modelimizin ayrılabilir parçalı sürekli doğrusal fonksiyonların uç değerlerinin bulunması için kullanılabileceğini de gösteriyoruz.

ACKNOWLEDGMENTS

First of all, I would like to thank Assoc. Prof. Metin Türkay for his help, guidance, trust and gentle efforts throughout my thesis study. I am grateful to Asst. Prof. Sibel Salman and Asst. Prof. Serdar Kozat for taking part in my thesis committee. I would also like to thank Koç University for the financial support and opportunities provided during my graduate education. I am grateful to my "CAS130" friends "Kübra, Naciye, Emin and Çagri" for being there whenever I needed their help and joy they added to my life at Koç University. Finally, special thanks to my parents, grand parents, sister and Can for being with me in both good and bad times and for their endless love. Without them everything I achieved would not be possible. I would like dedicate this thesis to them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

PWL	Piecewise Linear
MIP	Mixed Integer Programming
LP	Linear Programming
NLP	Nonlinear Programming
IP	Integer Programming
t	Number of Break Points of PWL Function
j	Index of Break Points of PWL Function, $j = 1, .., t$
x_j	Break Points of PWL Function
a_j	Slope of the PWL Function in interval j
b_j	Cost Intercept of PWL Function in Interval j
$f(x_j)$	PWL Function Value at Break Point j
λ_j	Multipliers of Each Break Point in Convex Combination Formulation
δ	Binary Decision Variables in Convex Combination Formulation
z_j	Total Load Amount of x in Interval j in Multiple Choice Model
q_j	Load Amount in Interval j in Incremental Cost Model
y_j	Load Amounts in Interval j in Our Formulation
x	Domain value of PWL function in Our Formulation
y	Value of PWL function at x in Our Formulation

Chapter 1

INTRODUCTION

A function which consists of a number of linear sections, with different slopes is defined as a piecewise linear function. We encounter piecewise linear functions in many different fields of science [33] [11] [45] [34] [22] [2] and engineering [50] [29] [44] [23] [51] [8] [48] [10], [9]. They are often used to approximate nonlinear functions in order to make their analysis easy. For example, Osowski approximated an electrical circuit to a convex piecewise linear function for optimization [41], similarly Dantzig, Johnson and White used piecewise linear approximation while minimizing a chemical equilibrium function [19]. Figure 1.1 illustrates a nonlinear function's(blue) approximation to a piecewise linear function(red). Additionally, in industrial engineering (especially in supply chain management) we encounter optimization problems with piecewise linear cost functions. These problems include network loading problems [38] [5] [27] [28], facility location problems [31] [32], merge in transit problems [15] and network flow problems [12] [1] [3] [14]. Their wide range of application areas make modeling and analysis of piecewise linear functions an important subject to study and the history of these studies dates back to 1950s [16], [7], [40].

1.1 Piecewise Linear Functions

Piecewise linear functions have many different definitions and the followings are some examples to these definitions:

Definition 1 In mathematics, a piecewise linear function is a piecewise-defined function whose pieces are linear. [6]

Definition 2 A function which consists of a number of linear sections, with different slopes

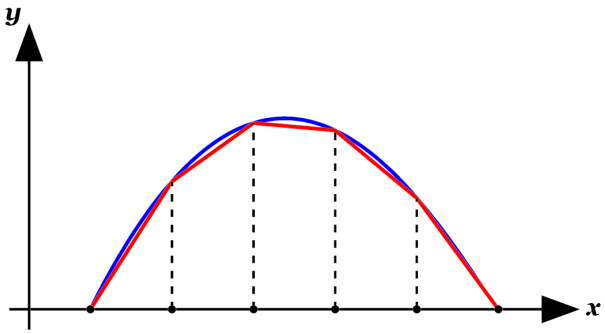

Figure 1.1: Piecewise Linear Approximation of a Nonlinear Function

is defined as a piecewise linear function. [6]

Definition 3 A function f on the interval $[a, b]$ is called a piecewise linear function if there is a subdivision $a = x_0 \leq x_1 \leq ... \leq x_n = b$ such that f is linear on each interval $[x_j, x_{j+1}]$. [6]

Definition 4 If $f(x)$ is a piecewise linear function on the domain $[x_1,x_t]$ then it is defined as follows [20]:

$$f(x) = \begin{cases} a_1x + b_1 & \text{if } x_1 \leq x \leq x_2 \\ a_2x + b_2 & \text{if } x_2 \leq x \leq x_3 \\ ... & ... \ ... \\ ... & ... \ ... \\ a_{t-1}x + b_{t-1} & \text{if } x_{t-1} \leq x \leq x_t \end{cases} \tag{1.1}$$

where

x_j : break points on domain j=1,...,t

a_j : the slope of the cost function for interval j $\forall$ j=1,...,t-1

b_j : cost intercept of the function for interval j $\forall$ j=1,...,t-1

It is assumed that $x_j \leq x_{j+1}\ \forall\ j$=1,...,$t$-1 and the domain is bounded, i.e. $x_t < \infty$.

Note the strict inequality when defining a function:

$$f(x) = \begin{cases} a_1 x + b_1 & \text{if } x_1 \leq x < x_2 \\ a_2 x + b_2 & \text{if } x_2 \leq x < x_3 \\ \quad ... & \quad ... \ ... \\ \quad ... & \quad ... \ ... \\ a_{t-1} x + b_{t-1} & \text{if } x_{t-1} \leq x < x_t \end{cases} \tag{1.2}$$

We can consider $<$ as $\leq$ by implicitly subtracting ϵ from x_j such that $x_{j-1} \leq x \leq (x_j - \epsilon)$. The rest of the thesis refers to the last definition of piecewise linear functions given above. As an example, Figure 1.2 illustrates a piecewise linear function(red) defined by Eq. (1.3). Piecewise linear function defined by Eq. (1.3) has a bounded domain from -7 to 50, 7 break points and is divided into 6 intervals. The break points of this function are $x_1 = -7, x_2 = 1, x_3 = 20, x_4 = 28, x_5 = 30, x_6 = 40$ and $x_7 = 50$. The slopes on each interval are $a_1 = 1, a_2 = 1.5, a_3 = a_4 = 0, a_5 = -1, a_6 = 1$ and the cost intercepts are $b_1 = 10, b_2 = 3, b_3 = 33, b_4 = 0, b_5 = 20, b_6 = -60$.

$$f(x) = \begin{cases} x + 10 & \text{if } -7 \leq x \leq 1 \\ 1.5x + 3 & \text{if } 1 \leq x \leq 20 \\ 33 & \text{if } 20 \leq x \leq 28 \\ 0 & \text{if } 28 \leq x \leq 30 \\ -x + 20 & \text{if } 30 \leq x \leq 40 \\ x - 60 & \text{if } 40 \leq x \leq 50 \end{cases} \tag{1.3}$$

Different criteria are used to classify piecewise linear functions depending on the purpose of classification. One commonly used measure to classify PWL functions is to use their structural properties. This approach has two main considerations:

1. Continuity at the break points

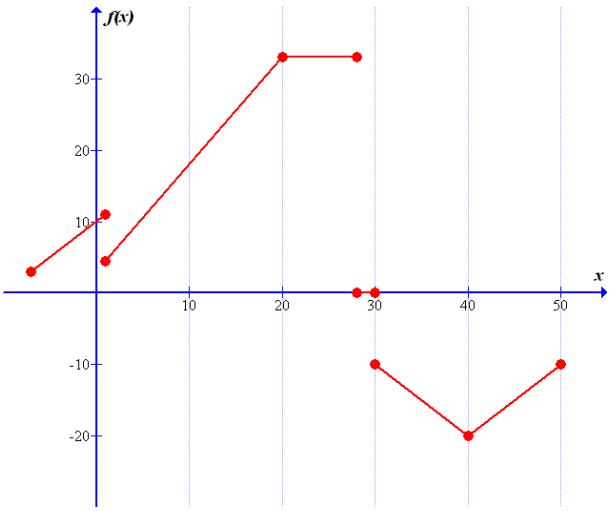

Figure 1.2: A Piecewise Linear Function

2. Convexity of the overall function

The following classes of piecewise linear functions are derived using these considerations:

- Continuous Piecewise Linear Functions
  - Convex Piecewise Linear Functions
  - Concave Piecewise Linear Functions
  - Non-Convex Piecewise Linear Functions
- Discontinuous Piecewise Linear Functions

1.2 Problem Definition

In this thesis we develop a linear programming formulation to find the extrema of a piecewise linear function $f(x)$, where $f(x)$ is defined with Eq. (1.1). We focus on formulating *continuous* piecewise linear functions in all classes *(convex, concave, non-convex)* and we also extend our research to cover the formulation of *separable piecewise continuous linear functions*. We propose a linear programming formulation that can find the extrema of any

continuous piecewise linear function in the existence of no other constraints. Our model is superior to the existing models in terms of complexity and solution time. The general problem structure is:

$$\begin{aligned} min \quad & f(x) \\ st \quad & x \in X \end{aligned} \tag{1.4}$$

where $f(x)$ is a continuous piecewise linear function and $x \in X$ represents the set of equations that we define to find the extrema of $f(x)$.

The remainder of this thesis is organized as follows. In Chapter 2, an overview of the previous studies about modeling and optimization of PWL functions is given. Chapter 3 addresses our novel linear programming formulation to find the extrema of continuous piecewise linear functions and its extension to separable piecewise continuous linear functions. This chapter also includes illustrative examples to better visualize our model and prove its superiority over the existing formulations from literature. Chapter 4 includes benchmark studies of these formulations with our novel formulation in terms of cpu time usage. Results of benchmark studies supported with our findings from previous chapters are also provided in Chapter 4. The thesis concludes in Chapter 5 with summarizing and emphasizing our work and giving initiatives for future research. Additionally, data and sample GAMS code used during benchmarks, and some definitions related to classification of PWL functions are given in Appendixes.

Chapter 2

LITERATURE REVIEW

Piecewise linear functions are difficult to model, because their shape changes in different parts of the domain. Hence, complex formulations are needed to model such functions exactly. Several approaches have been developed so far to model piecewise linear functions. Based on their modeling and solution strategies we can group these approaches into three categories. The first group of studies base on mixed integer programming (MIP) formulations. The second group consists of a combination of linear programming formulations and special modeling techniques. And the third group focuses on piecewise linear functions with special properties.

The first category of work includes the use of mixed integer programming (MIP) models. Several MIP formulations are proposed to model piecewise linear functions; such as convex combination [17], multiple choice [12], incremental cost models [40] and bounded variable method [18]. The first three of these formulations are the ones that are widely used in textbooks and literature.

- Convex Combination Model

  Convex combination model makes use of the fact that any point on the curve defined by PWL function $f(x)$ can be represented as a weighted average of the two successive break points [17] . x and $f(x)$ are defined by convex combinations of break points x_j and function values at break points ($f(x_j)$) respectively (Figure 2.1):

$$\begin{aligned} x &= \lambda_1 x_1 + \lambda_2 x_2 + ... + \lambda_t x_t \\ f(x) &= \lambda_1 f(x_1) + \lambda_2 f(x_2) + ... + \lambda_t f(x_t) \\ 1 &= \lambda_1 + \lambda_2 + ... + \lambda_t \\ \lambda_j &\leq 1 \ \forall \ j \end{aligned}$$

$$\lambda_j \geq 0 \ \forall \ j \tag{2.1}$$

then impose the conditions that all $\lambda_j = 0$ except for one pair λ_j and λ_{j+1} as follows :

$$\begin{aligned}
\lambda_1 &\leq \delta_1 \\
\lambda_2 &\leq \delta_1 + \delta_2 \\
&.... \\
\lambda_{t-1} &\leq \delta_{t-2} + \delta_{t-1} \\
\lambda_t &\leq \delta_{t-1} \\
1 &= \delta_1 + \delta_2 + ...\delta_{t-1} \\
\delta_j &= 0, 1 \ \forall \ j = 1, .., t-1
\end{aligned} \tag{2.2}$$

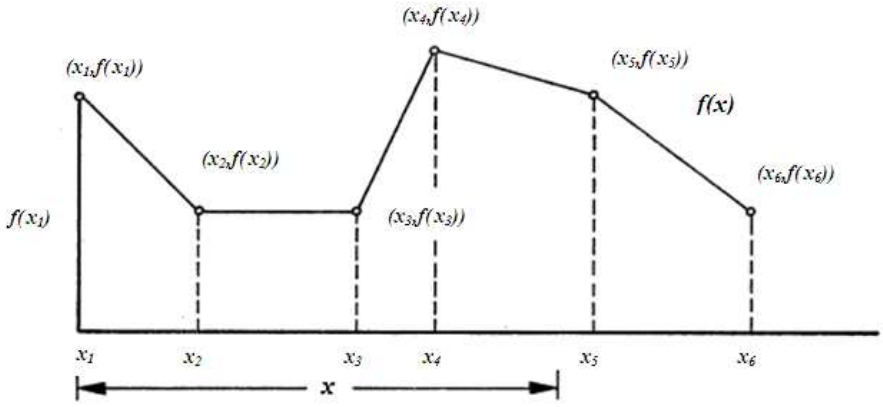

Figure 2.1: Convex Combination Model [17]

Let $f(x)$ be a piecewise linear function with j break points. Then modeling $f(x)$ with Convex Combination Model results in $2j + 1$ variables and $4j + 3$ constraints.

- Multiple Choice Model

  Multiple choice model defines a new variable z_j as the total load of x in interval j [12]. If the total load is equal to $\hat{x}$ and $\hat{x}$ is in the interval $\hat{j}$, then $z_{\hat{j}} = \hat{x}$ and $z_j = 0$ for

all intervals $j \neq \hat{j}$. Additionally, binary variable y_j is introduced as $y_j = 1$ if $z_j > 0$ and $y_j = 0$ otherwise. Additionally the condition that at most one y_j is equal to one is enforced. Based on these rules $f(x)$ and x is defined as follows:

$$\begin{aligned} f(x) &= \sum_j a_j z_j + b_j y_j \\ x &= \sum_j z_j \end{aligned} \tag{2.3}$$

and the constraints

$$\begin{aligned} x_j y_j &\leq z_j \leq x_{j+1} y_j \ \forall \ j = 1, .., t-1 \\ \sum_j y_j &= 1 \\ y_j &= 0, 1 \ \forall \ j = 1, .., t-1 \end{aligned} \tag{2.4}$$

Let $f(x)$ be a piecewise linear function with j break points. Then modeling $f(x)$ with Multiple Choice Model results in $2j$ variables and $3j$ constraints.

- Incremental Model

  As mentioned in [40], the incremental model introduces the component variables q_j as the load amount in interval j. The summation of q_j gives the total load amount x. The following equations defines the $f(x)$ and x.

$$\begin{aligned} f(x) &= b_1 + a_1 x_1 + \sum_{j=1}^{t-1} a_j q_j \\ x &= x_1 + \sum_{j=1}^{t-1} q_j \end{aligned} \tag{2.5}$$

  The interval $k+1$ cannot be loaded unless the interval k is fully loaded. To provide this condition an additional binary variable y_j is introduced as $y_j = 1$ if $q_j > 0$ and

$y_j = 0$ otherwise. The following are the related constraints:

$$\begin{aligned} y_j &\geq (\frac{1}{x_{j+1} - x_j})q_j \;\; j = 1, .., t-1 \\ y_{j+1} &\leq (\frac{1}{x_{j+1} - x_j})q_j \;\; j = 1, .., t-2 \\ y_j &= 0, 1 \;\; \forall \;\; j = 2, .., t \\ q_j &\geq 0 \end{aligned} \tag{2.6}$$

Let $f(x)$ be a piecewise linear function with j break points. Then modeling $f(x)$ with Incremental Cost Model results in $2j$ variables and $4j$ constraints.

Comparing MIP formulations with different criterion have became a popular subject to study. Researchers evaluated MIP formulations in order to find best performing formulation in terms of solution time and complexity. Majority of these evaluations are conducted by analyzing the strength of LP relaxations of MIP formulations. For example, Padberg studied incremental cost and convex combination models by comparing their LP relaxations [42]. In this study he showed that incremental cost formulation is far better than convex combination formulation. He also showed that incremental cost formulation is locally ideal. (A linear programming formulation is locally ideal if in the absence of other constraints its LP relaxation models the problem perfectly [43]).

Similarly, Croxton, Gendron and Magnanti compared incremental cost, convex combination and multiple choice models by studying their LP relaxations [12]. They showed that all these three formulations are equal in the sense that any feasible solution to one model is also feasible to another model with the same cost. Besides, using the relationship of the models to lagrangian duality, they showed that LP relaxations of all models approximate the original function with its lower convex envelope .

Another comparison was conducted by Keha, Farias and Nemhauser [35]. They compared incremental cost and convex combination formulations by investigating their LP relaxations with and without binary variables. LP relaxations without binary variables of both models gave the same bound as those with binary variables. Hence, they suggested

that modeling without binary variables (with the help of SOS2 or SOSX approach) is advantageous because the size of the formulation reduces and the problem structure becomes compact. Additionally, they showed that neither formulation is superior to other when the binary variables are removed.

Likewise, Croxton, Gendron and Magnanti studied MIP formulations for multi-commodity network flow problems with non-convex piecewise linear costs and extended those formulations by applying disaggregation methods [13]. They proved that disaggregation techniques improve the LP lower bounds for multi-commodity network flow problems when the objective is a PWL function.

Ho also conducted a benchmark study and compared 4 different formulations to formulate separable convex PWL functions [30]. The first two of these formulations (δ and λ) formulates the function (γ and σ) by representing $f(x)$. He showed that the decomposition dual of δ and λ is equal to the LP dual of σ. Besides, γ and σ has the same extremal equivalent which corresponds to the formulation λ with an elimination of variables.

The second group of work focuses on modeling PWL functions without binary variables and enforcing nonlinearities with different methods. For this purpose, Beale and Tomlin suggested a formulation similar to convex combination, except that no binary variables are included in the model [4]. Instead, they focused on developing special branch and bound strategies to minimize sums of non-convex functions. They utilized the special structure of such problems and showed that special ordered sets (SOS) can be used during the branch and bound process. They described the procedure and performed tests indicating a better performance than conventional MIP formulations. Their approach has been utilized by different researchers to solve PWL functions' optimization problems. For example; in their paper Keha, Farias and Nemhauser introduced an algorithm to solve continuous separable non-convex piecewise linear optimization problems [36]. Their algorithm relies on using special ordered sets of two variables (SOS2) in branch and bound algorithm. In this paper, they also suggested ways of creating strong cuts to be used in branch and cut algorithm. Similarly, Farias and Zhao used SOS to solve discontinuous piecewise linear optimization problems [21] and they called the method as "SOSD".

Fourer published three papers complementing each other to introduce a modified simplex algorithm for piecewise linear programming [24] [25] [26]. He developed and analyzed a general and computationally practical simplex algorithm for minimization of convex separable piecewise linear objective subject to linear constraints. In his first paper, he introduced and justified a PWL simplex algorithm under three assumptions (finiteness, feasibility and non-degeneracy) [24]. In his second paper, he showed how these assumptions can be relaxed to allow the effective use of the algorithm and he also studied the theory of PWL programming in detail in this second paper [25]. In his third paper, he argued the advantages of using a direct piecewise linear simplex implementation over an indirect approach that transforms the function into a linear program [26]. For example, he proposed that the algorithm introduced in first paper is 2-6 times faster than a comparable linear algorithm.

In addition to these studies researchers have also developed new MIP formulations with several advantages over the existing ones. For example, Sherali developed a formulation which is a modification of the convex combination problem [46]. His formulation has totally unimodularity property and hence is locally ideal. It can be applied to lower semi-continuous PWL functions and when used to formulate continuous PWL functions it reduces to the Padberg's formulation [42]. Likewise, Li, Lu, Huang and Hu proposed a way of representing PWL functions with few binary variables [37]. The numerical examples indicate that their formulation is computationally efficient compared to existing formulations (especially multiple choice) and performs much better when the number of break points is large.

The third group of studies related to piecewise linear functions focus on modeling piecewise linear functions with special properties. In 1956 Dantzig proposed a formulation that can be used to find the minimum of a convex separable PWL function [16]. The minimum is attained by loading each segment up to its upper bound until the desired domain value is reached. He showed that even though this formulation increases the number of variables, the number of constraints remain the same. He illustrated his formulation with an example. Similarly, Markowitz and Manne observed that while minimizing a convex piecewise linear function, there is no need to binary variables and minimum is obtained automatically [40]. On the other hand, when the PWL function to be minimized is concave binary variables

should be used. Hence, they introduced Incremental Cost Formulation that we mentioned among the first group of studies as well [40].

Dantzig, Johnson and White approximated chemical equilibrium to a PWL function [19]. They transferred free energy function into a convex separable PWL function, which can be formulated with linear programming tools. Finally, they illustrated how to formulate a convex PWL function as a linear programming problem by applying it on the sample problem. Similar to their study, Osowski approximated a nonlinear function (from circuit theory) to a PWL function [41]. The resulting PWL function was convex and hence he formulated it without binary variables as follows:

Let $f(x)$ be a convex piecewise linear function defined by Eq. (1.1) and satisfying conditions in Eq. A.4. Then the following LP formulation models the minimization problem of $f(x)$ exactly:

$$\begin{aligned} min \quad & a_1 y_1 \quad + a_2 y_2 + ... + a_{t-1} y_{t-1} \\ x \quad & = \quad y_1 + y_2 + ... + y_{t-1} \\ 0 \quad & \leq \quad y_j \leq \Delta_j \ \ \forall \ \ j = 1, .., t-1 \end{aligned} \tag{2.7}$$

where

$$\Delta_j = x_{j+1} - x_j \tag{2.8}$$

defines the length of each interval.

Osowski explained this model as follows; "... according to the convexity of the function $a_1 \leq a_2 \leq ... \leq a_{t-1}$, it is obvious that the way to find the minimum is to choose y_1 as large as possible until it hits its upper bound Δ_1, then take y_2 in turn etc, until y_k is reached such that by setting $y_k = \Delta_k$ the value of x is exceeded - in which case y_k is reduced and chosen in such a way that second equation above holds. It is evident that this process simply generates the curve $f(x)$, section after section, from zero up to the given value of x. " [41]

Chua and Kang worked on sectionwise Piecewise Linar functions and their canonical representations [9]. They observed that when the Piecewise Linear functions are continu-

ous and monotonically increasing or decreasing the formulation changes to a simpler form and binary variables may be removed. They also worked on approximations of nonlinear functions using sectionwise PWL representation. Additionally, Snyder studied the maximization of concave piecewise linear separable objectives subject to linear constraints [47]. He observed that such problems may have a special structures that are not utilized by simplex method. He proposed that division of variables into special ordered sets to control the pivoting operations can reduce the number of basis changes in the simplex algorithm.

Charnes and Lemke studied the PWL approximation of nonlinear problems [7]. They covered convex and separable PWL functions and developed an extended simplex method for them. They illustrated the method with an example problem from the literature and outlined the procedure on this instance. They achieved notable improvements in the computational tableau by taking advantage of the "uncompressed matrix of the problem". First they convert the nonlinear function to a PWL function and then using the duality they transformed the approximation into a simplex problem. Similar to Charnes and Lemke's study, Mangarasian, Rosen and Thompson used PWL functions to estimate the global minimum of a non-convex function in $\mathbb{R}^n$ with multiple local minimums [39]. They approximated it from below via concave minimization with PWL convex function using sample points from the original function. The method provides accurate estimate of the global minimum and applied to non-convex piecewise quadratic functions that model protein docking problems.

Although, modeling of piecewise linear functions is extensively studied in literature, effective models for representing continuous piecewise linear functions had not been developed so far. They were either complex or targeted to a very limited classes of PWL functions. The objective of this study is to fulfill this gap by introducing a novel LP formulation without binary variables, for finding the extrema of continuous piecewise linear functions. Our novel formulation is superior to existing formulations in terms of complexity and simplicity. There are locally ideal MIP formulations; however, their complexity is high or they are used with special branching strategies. This becomes a serious issue especially when the PWL function has too many break points and/or component functions, because this increases the cpu time spend until finding the extrema of PWL functions. To the contrary, our formu-

lation has much lower complexity than any other existing formulation in literature. It has less number of constraints and variables, and it is free from binary variables. These factors decrease the cpu time for finding the extrema of PWL functions when our formulation used. Other than complexity and simplicity, our formulation can be applied to a wide range of PWL functions. There are formulations designed for specially structured PWL functions; however they have limited application areas, such as only convex PWL functions etc. Our formulation, however, can be applied to all types of continuous PWL functions (convex, concave, non-convex) and separable piecewise continuous linear functions.

Chapter 3

A NOVEL LINEAR PROGRAMMING FORMULATION TO FIND THE EXTREMA OF CONTINUOUS PIECEWISE LINEAR FUNCTIONS

We divided this chapter into four sections. The first section gives the model as a whole. The second section mentions some observations related to piecewise linear functions. The third section validates our novel formulation with a proof and models a piecewise linear functions with our novel formulation. Section four shows how we extended our formulation to separable piecewise continuous linear functions and illustrates this extension on a separable piecewise continuous linear function. Throughout this section we refer to the PWL definition in Chapter 1.

3.1 Novel Formulation

In our novel formulation, we defined decision variables y_j, as the load amounts in each interval j. These decision variables are the same as the incremental cost model of Markowitz and Manne, Eq. (2.5 - 2.6) [40]. Using these decision variables we developed the following novel formulation for finding the extrema of continuous piecewise linear functions:

$$
\begin{aligned}
min \quad & y & & \\
st \quad & y & = a_1 x_1 + b_1 + \sum_{j=1}^{t-1} a_j y_j & \\
& x & = x_1 + \sum_{j=1}^{t-1} y_j & \\
& 0 & \leq y_j \leq x_{j+1} - x_j & \forall \ j = 1, ..., t-1 \\
& y_{j+1} & \leq (x_{j+2} - x_{j+1}) y_j / (x_{j+1} - x_j) & \forall \ j = 1, ..., t-2
\end{aligned}
$$

(3.1)

The objective function in Eq. (3.1) can be replaced by max to find the maximum of the piecewise linear function. The first constraint defines the PWL function's value y and the second constraint defines the domain's value x that corresponds to y. The third and fourth constraints are actually the set of constraints that define the load amounts in each interval and the relationships among them respectively.

3.2 Observations Related to Continuous Piecewise Linear Functions

In order to model and optimize a continuous piecewise linear function, the following observations should be taken into account.

- For a continuous piecewise linear function with $a_j \neq 0 \; \forall \; j = 1,...,$ t-1 extrema always lies at kink (break)points

  Proof : Assume to the contrary that minima of a piecewise linear function ($f(x^*)$) is in the s^{th} segment where $x^* \neq x_S$ and $x^* \neq x_{s+1}$.

$$\begin{aligned} f(x^*) &= a_s x^* + b_s, \;\; x \in (x_s, x_{s+1}) \\ f(x_s) &= a_s x_s + b_s \\ f(x_{s+1}) &= a_s x_{s+1} + b_s \end{aligned} \tag{3.2}$$

  Then,

$$\begin{aligned} f(x_s) - f(x^*) &= a_s(x_s - x^*) \\ f(x_{s+1}) - f(x^*) &= a_s(x_{s+1} - x^*) \end{aligned} \tag{3.3}$$

  Since $x^* \in (x_s, x_{s+1})$ we obtain $(x_s - x^*) < 0$ and $(x_{s+1} - x^*) > 0$.

  If $a_s < 0$ then,

$$f(x_s) - f(x^*) > 0$$

$$f(x_{s+1}) - f(x^*) < 0 \tag{3.4}$$

Hence, $f(x_s) > f(x^*) > f(x_{s+1})$ when $x^* \in (x_s\ ,\ x_{s+1})$ which is a contradiction to the previous assumption.

If $a_s > 0$ then,

$$\begin{aligned} f(x_s) - f(x^*) &< 0 \\ f(x_{s+1}) - f(x^*) &> 0 \end{aligned} \tag{3.5}$$

Therefore, $f(x_s) < f(x^*) < f(x_{s+1})$ when $x^* \in (x_s\ ,\ x_{s+1})$ which is contradiction to the previous assumption. Hence, the minima of a piecewise linear function is located always at break points when $a_j \neq 0\ \forall\ j = 1,..,\ t\text{-}1$. □

When $\exists\ a_j = 0$, the above assumption can be restated as: " At least one extreme point corresponds to the extrema of the piecewise linear function represented ".

- Markowitz and Manne stated in their paper the following: "While modeling piecewise linear functions; feasibility requires that the load on the interval $k+1$ must be zero unless the load on interval k is equal to its upper bound" [40]. This observation can be restated as follows:

$$y_{k+1} > 0 \ \ only \ \ if \ \ y_k = x_{k+1} - x_k \tag{3.6}$$

3.3 Proof of the Formulation

Theorem 1 *If a piecewise linear function is continuous, then it can be represented with the following system of linear equations exactly (given that load amounts in each interval meet the condition in Eq. (3.6)):*

$$y \ = \ a_1 x_1 + b_1 + \sum_{j=1}^{t-1} a_j y_j$$

$$x = x_1 + \sum_{j=1}^{t-1} y_j \tag{3.7}$$

where each y_j is a continuous variable satisfying :

$$0 \leq y_j \leq x_{j+1} - x_j \quad \forall \ j = 1, ..., t-1 \tag{3.8}$$

Proof : Consider an arbitrary value $\overline{x}$ in the interval [x_k,x_{k+1}]. Then the value of piecewise linear function is:

$$f(\overline{x}) = a_k \overline{x} + b_k \tag{3.9}$$

Assume to the contrary that the value of piecewise linear function at this arbitrary $\overline{x}$ is not equal to the value of y at $\overline{x}$, where y and $\overline{x}$ are defined by (3.7). Which is :

$$f(\overline{x}) \neq y \tag{3.10}$$

Then second equation of (3.7) for this chosen $\overline{x}$ can be written as:

$$\overline{x} = x_1 + \sum_{j=1}^{k-1} y_j + y_k + \sum_{j=k+1}^{t-1} y_j \tag{3.11}$$

Assume that

y_j: at its upper bound, which is $y_j = x_{j+1}$ - x_j $\forall$ j = 1,..., k-1.

y_j: at its lower bound, which is $y_j = 0$ $\forall$ j = k+1,..., t-1.

y_k: in the interval of its boundaries.

Then, Eq(3.11) becomes :

$$\begin{aligned} \overline{x} &= x_1 + \sum_{j=1}^{k-1} (x_{j+1} - x_j) + y_k + \sum_{j=k+1}^{t-1} 0 \\ \overline{x} &= x_k + y_k \end{aligned} \tag{3.12}$$

Since $0 \leq y_k \leq x_{k+1}$ - x_k, when we add x_k to the both sides of this inequality and combine it with Eq(3.12) we obtain :

$$\begin{aligned} x_k &\leq x_k + y_k \leq x_{k+1} \\ x_k &\leq \overline{x} \leq x_{k+1} \end{aligned} \tag{3.13}$$

Eq(3.13) shows that functions in Eq(3.7) represents $\overline{x}$ exactly. Then, from the first equation in (3.7), we can write y at $\overline{x}$ as follows :

$$\begin{aligned} y &= a_1x_1 + b_1 + \sum_{j=1}^{t-1} a_jy_j \\ &= a_1x_1 + b_1 + \sum_{j=1}^{k-1} a_jy_j + a_ky_k + \sum_{j=k+1}^{t-1} 0 \\ &= a_1x_1 + b_1 + \sum_{j=1}^{k-1} a_j(x_{j+1} - x_j) + a_ky_k \end{aligned}$$

$$\begin{aligned} y &= a_1x_1 + b_1 + a_1(x_2 - x_1) + a_2(x_3 - x_2) + ... + a_{k-2}(x_{k-1} - x_{k-2}) + a_{k-1}(x_k - x_{k-1}) + a_ky_k \\ y &= a_1x_1 + b_1 + a_ky_k + a_1x_2 - a_1x_1 + \sum_{j=2}^{k-1} a_j(x_{j+1} - x_j) \\ y &= a_ky_k + b_1 + a_1x_2 + \sum_{j=2}^{k-1} a_j(x_{j+1} - x_j) \end{aligned} \tag{3.14}$$

From Eq(A.1) we know;

$$\begin{aligned} a_1x_2 + b_1 = a_2x_2 + b_2 &\Rightarrow b_2 = a_1x_2 + b_1 - a_2x_2 \\ a_2x_3 + b_2 = a_3x_3 + b_3 &\Rightarrow b_3 = a_2x_3 + b_2 - a_3x3 \\ ... &\Rightarrow ... \\ a_{k-2}x_{k-1} + b_{k-2} = a_{k-1}x_{k-1} + b_{k-1} &\Rightarrow b_{k-1} = a_{k-2}x_{k-1} + b_{k-2} - a_{k-1}x_{k-1} \\ a_{k-1}x_k + b_{k-1} = a_kx_k + b_k &\Rightarrow b_k = a_{k-1}x_k + b_{k-1} - a_kx_k \end{aligned}$$

$$b_k = b_1 + a_1x_2 + a_2(x_3 - x_2) + a_3(x_4 - x_3) + ... + a_{k-1}(x_k - x_{k-1}) - a_kx_k \tag{3.15}$$

After inserting b_k into Eq(3.9) and rearranging we obtain :

$$\begin{aligned} f(\overline{x}) &= a_k\overline{x} + b_k \\ &= a_k\overline{x} + b_1 + a_1x_2 + a_2(x_3 - x_2) + a_3(x_4 - x_3) + ... + a_{k-1}(x_k - x_{k-1}) - a_kx_k \\ &= a_k(\overline{x} - x_k) + b_1 + a_1x_2 + \sum_{j=2}^{k-1} a_j(x_{j+1} - x_j) \\ &= a_ky_k + b_1 + a_1x_2 + \sum_{j=2}^{k-1} a_j(x_{j+1} - x_j) \end{aligned} \tag{3.16}$$

From equations Eq(3.14) and Eq(3.16) it is observed that $y = f(\overline{x})$. Hence, assumption $f(\overline{x}) \neq y$ does not hold and $f(x)$ is exactly represented. □

Theorem 2 *The following LP models the minimization of any continuous piecewise linear function exactly.*

$$\begin{aligned} min \quad & y \\ st \quad & y = a_1x_1 + b_1 + \sum_{j=1}^{t-1} a_jy_j \\ & x = x_1 + \sum_{j=1}^{t-1} y_j \\ & 0 \leq y_j \leq x_{j+1} - x_j \quad \forall\ j = 1, ..., t-1 \\ & y_{j+1} \leq (x_{j+2} - x_{j+1})y_j/(x_{j+1} - x_j) \quad \forall\ j = 1, ..., t-2 \end{aligned} \tag{3.17}$$

Proof : We proved the validity of the first and second constraints in Theorem 1. When we examine the last constraint of our model we observe the following:

For the interval (x_k,x_{k+1}) let y_k be the load amount in this interval, then:

- If $y_k = 0$, then the last constraint enforces the following :

$$y_{k+1} \leq (x_{k+2} - x_{k+1})y_k/(x_{k+1} - x_k)$$

$$\begin{aligned} y_{k+1} &\leq (x_{k+2} - x_{k+1})0/(x_{k+1} - x_k) \\ y_{k+1} &\leq 0 \end{aligned} \tag{3.18}$$

and combining this with the third constraint of Eq (3.31) we have :

$$0 \leq y_{k+1} \leq 0 \;\; \Rightarrow \;\; y_{k+1} = 0. \tag{3.19}$$

Hence; if y_k=0, then y_j=0 $\forall$ j=k+1, ..., t-1.

- If $y_{k+1} = x_{k+2}$ - x_{k+1}, then the last constraint enforces the following :

$$\begin{aligned} y_{k+1} &\leq (x_{k+2} - x_{k+1})y_k/(x_{k+1} - x_k) \\ (x_{k+2} - x_{k+1}) &\leq (x_{k+2} - x_{k+1})y_k/(x_{k+1} - x_k) \\ x_{k+1} - x_k &\leq y_k \end{aligned}$$

and combining this with the third constraint of Eq (3.31) we have :

$$x_{k+1} - x_k \leq y_k \leq x_{k+1} - x_k \;\; \Rightarrow \;\; y_k = x_{k+1} - x_k \tag{3.20}$$

Therefore; if $y_{k+1} = x_{k+1}$ - x_k, then $y_j = x_{j+1}$ - x_j $\forall$ j=1, ..., k.

The last two constraints construct a feasible region where each corner point of that feasible region corresponds to a break point of the piecewise linear function modelled. The total load and its corresponding function value is calculated based on the y_j values at the extreme points of that feasible region. Moreover, for any break point k represented in this feasible region, our second observation holds as follows:

1. All the intervals up to the break point k are loaded up to their upper bound. That is:

$$y_j = x_{j+1} - x_j \quad \forall \;\; j = 1, ..., k-1 \tag{3.21}$$

2. All the intervals after the break point k are not loaded. That is:

$$y_j = 0 \quad \forall \ j = k, ..., t-1 \tag{3.22}$$

In previous section we observed and proved that minima of any continuous piecewise linear function is located at an extreme point. Our model constructs a feasible region with the properly defined extreme points each corresponding to the break points of the piecewise linear function. Moreover, we know that simplex algorithm moves among the extreme points of the feasible region while searching the optimal solution. Hence, the extreme point at which the simplex algorithm finds the optimal solution corresponds to the break point where the piecewise linear function has its minima. Therefore, linear programming formulation in Eq (3.31) models the minimization of continuous piecewise linear function exactly. □

The formulation in Eq (3.31) can be easily converted into a formulation which finds the maxima of any continuous piecewise linear function by changing the objective function from *min* to *max*. Consequently, we can claim that this formulation finds the extrema (minima or maxima) of any continuous piecewise linear function exactly.

Illustrative Example

Let f(x) be a continuous piecewise linear function defined by Eq. (3.23) and represented in Fig. 3.1 :

$$f(x) = \begin{cases} 8 + 2x & \text{if } 0 \leq x \leq 3 \\ 20 - 2x & \text{if } 3 \leq x \leq 7 \end{cases} \tag{3.23}$$

Where $x_1 = 0$, $x_2 = 3$ and $x_3 = 7$; $a_1 = 2$ and a_2= -2; $b_1 = 8$ and $b_2 = 20$.

When we model PWL function in Eq. (3.23) according to the formulation in Theorem (2) the following model is obtained :

$$\begin{aligned} min \quad & y & \\ st \quad & y & = 8 + 2y_1 - 2y_2 \\ & x & = x_1 + y_1 + y_2 \end{aligned}$$

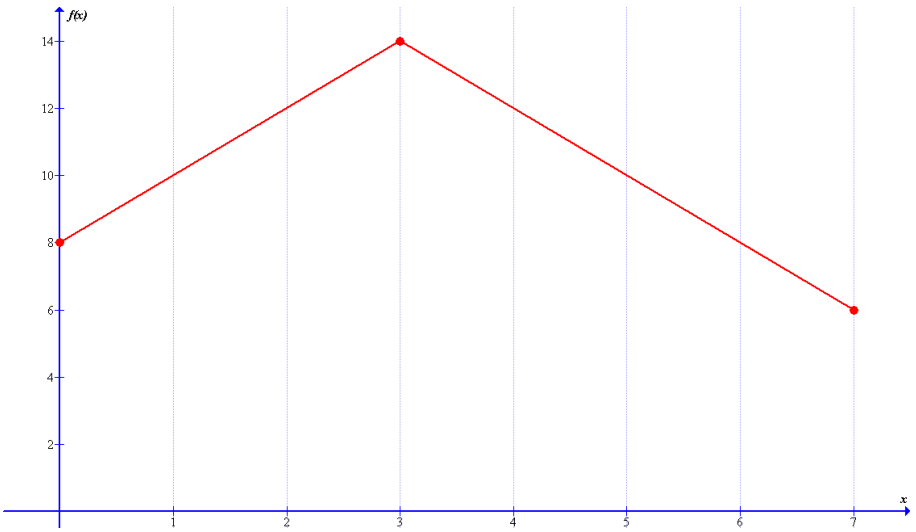

Figure 3.1: Graph of $f(x)$

$$\begin{aligned} y_1 &\leq 3 \\ y_2 &\leq 4 \\ 3y_2 &\leq 4y_1 \\ y_1 &\geq 0 \\ y_2 &\geq 0 \end{aligned} \tag{3.24}$$

LP formulation in Eq. (3.24) has 4 variables and 7 constraints and yields the feasible region in Fig. (3.2), where the corner point feasible solutions correspond to the break points and represent them exactly. First corner point (0,0) corresponds to the first break point x_1=0, where none of the intervals are loaded, that is y_1=y_2=0. The second corner point (3,0) corresponds to the second break point x_2=3, where the first interval is fully loaded and the second interval is not loaded, that is y_1=3 and y_2=0. The third corner point (3,4) corresponds to the third break point x_3=7, where all the intervals are loaded up to their upper bounds, that is y_1=3 and y_2=4.

When Eq(3.24) is solved with a linear programming solver, the optimal solution is obtained where y_1=3, y_2=4 and x^*=7 with the objective function value y^*=6. This solution

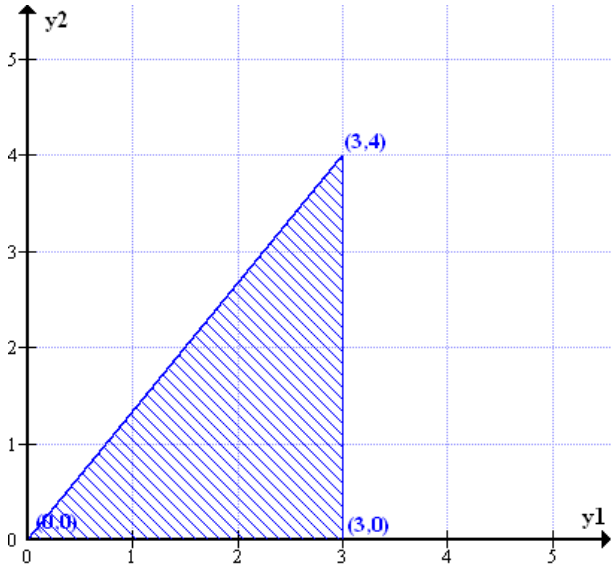

Figure 3.2: Feasible region of the model

represents the minima of piecewise linear function in Fig. 3.1 exactly.

We can model the same problem with three text book formulations:

- Convex Combination Model
- Multiple Choice Model
- Incremental Cost Model

Convex Combination Formulation:

$$
\begin{aligned}
min \quad & f(x) \\
st \quad & f(x) = \lambda_1 f(x_1) + \lambda_2 f(x_2) + \lambda_3 f(x_3) \\
& x = \lambda_1 x_1 + \lambda_2 x_2 + \lambda_3 x_3 \\
& \lambda_1 \leq 1 \\
& \lambda_2 \leq 1 \\
& \lambda_3 \leq 1
\end{aligned}
$$

$$\begin{aligned}
\lambda_1 \quad & +\lambda_2 + \lambda_3 = 1 \\
\lambda_1 \quad & \leq \delta_1 \\
\lambda_2 \quad & \leq \delta_1 + \delta_2 \\
\lambda_3 \quad & \leq \delta_2 \\
\delta_1 \quad & +\delta_2 = 1 \\
\lambda_1 \quad & \geq 0 \\
\lambda_2 \quad & \geq 0 \\
\lambda_3 \quad & \geq 0 \\
\delta_1 \quad & = 0, 1 \\
\delta_2 \quad & = 0, 1
\end{aligned} \tag{3.25}$$

Multiple Choice Formulation:

$$\begin{aligned}
min \quad & f(x) \\
st \quad & f(x) = a_1 z_1 + b_1 y1 + a_2 z_2 + b_2 y_2 \\
& x = z_1 + z_2 \\
& x_1 y_1 \leq z_1 \\
& x_2 y_2 \leq z_2 \\
& z_1 \leq x_2 y_1 \\
& z_2 \leq x_3 y_2 \\
& y_1 +y_2 = 1 \\
& y_1 = 0, 1 \\
& y_2 = 0, 1
\end{aligned} \tag{3.26}$$

Incremental Cost Formulation:

$$min \quad f(x)$$

$$
\begin{aligned}
st \quad f(x) &= b + a_1 x_1 + a_1 q_1 + a_2 + q_2 \\
x &= x_1 + q_1 + q_2 \\
y_1 &\geq \frac{q_1}{x_1 - x_2} \\
y_2 &\geq \frac{q_2}{x_3 - x_2} \\
y_2 &\leq \frac{q_1}{x_2 - x_1} \\
q_1 &\geq 0 \\
q_2 &\geq 0 \\
y_1 &= 0, 1 \\
y_2 &= 0, 1
\end{aligned} \tag{3.27}
$$

Formulating the same problem with different methods yield programming models with different number of variables and constraints. For example, when we model PWL function in Eq. (3.23) with our formulation and with three other formulations, we get programming models with different complexities, see Table 3.1. Among these formulations, our formulation has both least number of variables and constraints. This result can be generalized to all continuous PWL functions as follows: Let $f(x)$ be a PWL function with t number of break points. Then, table 3.2 represents the number of variables and constraints of different modeling methods (and their LP relaxations) used to formulate $f(x)$.

Table 3.1: Number of Variables and Constraints for Piecewise Linear Function in Fig. 3.1

	variables	constraints (LP)	constraints (MIP)
Our Formulation	4	7	
Convex Combination	6	11	9
Multiple Choice	7	17	15
Incremental Cost	6	11	9

Table 3.2: Number of Variables and Constraints for a Piecewise Linear Function with t Break Points

	variables	constraints (LP)	constraints (MIP)
Our Formulation	$t+1$	$2t+1$	
Convex Combination	$2t+1$	$5t+2$	$4t+3$
Multiple Choice	$2t$	$4t-1$	$3t$
Incremental Cost	$2t$	$5t-4$	$4t$

3.4 Extension to Separable Piecewise Continuous Linear Functions

Separable Piecewise Continuous Linear Function "A real-valued function $f(x)$ defined over $x = (x_1, x_2, ..., x_n)^T \in \Re^n$ is said to be a *seperable piecewise continuous linear function* if and only if it can be written as

$$f(x) = f_1(x_1) + f_2(x_2) + ... + f_n(x_n) \tag{3.28}$$

where each component function $f_i(x_i)$ depends only on one variable x_i, and is piecewise continuous linear with respect to x_i" [20].

From definition it is observed that, component functions $(f_i(x_i))$ of a separable piecewise linear function are defined independent from each other. Therefore, in the absence of other equations that link component functions to each other, minimizing/maximizing a separable piecewise continuous linear function $f(x)$ is equal to minimizing/maximizing its component functions $f_i(x_i)$ separately [12].

Let $f(x)$ be a separable piecewise continuous linear function defined by Eq.3.29,

$$f(x) = f_1(x_1) + f_2(x_2) + ... + f_n(x_n) = \sum_{i=1}^{n} f_i(x_i) \tag{3.29}$$

where each $f_i(x_i)$ is continuous PWL function defined by Eq. 3.30:

$$f_i(x_i) = \begin{cases} a_{i1}x_i + b_{i1} & \text{if } x_{i1} \le x_i \le x_{i2} \\ a_{i2}x_i + b_{i2} & \text{if } x_{i2} \le x_i \le x_{i3} \\ \ldots \quad \ldots \ldots & \\ \ldots \quad \ldots \ldots & \\ a_{it_i-1}x_i + b_{it_i-1} & \text{if } x_{it_i-1} \le x_i \le x_{it_i} \end{cases} \tag{3.30}$$

where

x_{ij} : break points on domain $\forall$ j=1,...,t_i in function f_i, i=1,...,n

a_{ij} : the slope of the cost function for interval j $\forall$ j=1,...,t_i-1 in function f_i, i=1,...,n

b_{ij} : cost intercept of the function for interval j $\forall$ j=1,...,t_i-1 in function f_i, i=1,...,n

t_i : number of break points of component function $f_i(x_i)$, i=1,...,n

It is assumed that $x_{ij} \le x_{ij+1}$ $\forall$ j=1,...,t_i-1, i=1,...,n and the domain is bounded, i.e. $x_{t_i} < \infty$. It is also further assumed that component piecewise linear functions are continuous fulfilling the conditions in Eq. (A.1).

Then we can find the minimum of $f(x)$ with the following LP formulation:

$$\begin{aligned} min \quad & f(x) \\ st \quad & f(x) = \sum_{i=1}^{n} y_i \\ & y_i = a_{i1}x_{i1} + b_{i1} + \sum_{j=1}^{t_i-1} a_{ij}y_{ij} \ \ \forall \ \ i = 1, ..., n \\ & x_i = x_{i1} + \sum_{j=1}^{t_i-1} y_{ij} \ \ \forall \ \ i = 1, ..., n \\ & 0 \le y_{ij} \le x_{ij+1} - x_{ij} \quad \forall \ \ j = 1, ..., t_i - 1 \ \ , \forall \ \ i = 1, ..., n \\ & y_{ij+1} \le \frac{(x_{ij+2} - x_{ij+1})y_{ij}}{x_{ij+1} - x_{ij}} \quad \forall \ \ j = 1, ..., t_i - 2 \ \ , \forall \ \ i = 1, ..., n \end{aligned} \tag{3.31}$$

We can apply the same formulation to find the maximization of $f(x)$ by replacing objective of the formulation from minimization to maximization.

Illustrative Example - Separable Piecewise Continuous Linear Functions

Let f(x) be a separable piecewise continuous linear function defined by Eq. 3.32

$$f(x) = f_1(x_1) + f_2(x_2) \tag{3.32}$$

where $f_1(x_1)$ and $f_2(x_2)$ defined by Eq. 3.33 and Eq. 3.34 respectively.

$$f_1(x_1) = \begin{cases} 2x_1 + 1 & \text{if } 0 \leq x_1 \leq 4 \\ -x_1 + 1 & \text{if } 4 \leq x_1 \leq 7 \end{cases} \tag{3.33}$$

$$f_2(x_2) = \begin{cases} -2x_2 & \text{if } -1 \leq x_2 \leq 2 \\ x_2 - 6 & \text{if } 2 \leq x_2 \leq 5 \\ 3x_2 - 16 & \text{if } 5 \leq x_2 \leq 10 \end{cases} \tag{3.34}$$

The first component function $f_1(x_1)$ has 2 intervals, with break points $x_{11} = 0$, $x_{12} = 4$, $x_{13} = 7$, slopes $a_{11} = 2$, $a_{12} = -1$ and cost intercepts $b_{11} = 1$, $b_{12} = 1$. The second component function $f_2(x_2)$ has 3 intervals with break points $x_{21} = -1$, $x_{22} = 2$, $x_{23} = 5$, $x_{24} = 10$, slopes $a_{21} = -2$, $a_{22} = 1$, $a_{23} = 3$ and cost intercepts $b_{21} = 0$, $b_{22} = -6$ and $b_{23} = -16$.

Then the LP formulation in Eq. 3.35 finds the minimum of $f(x)$ exactly.

$$\begin{aligned} min \quad & f(x) \\ st \quad & f(x) = y_1 + y_2 \\ & y_1 = 1 + 2y_{11} - y_{12} \\ & y_2 = -2y_{21} + y_{22} + 3y_{23} \end{aligned}$$

$$
\begin{aligned}
x_1 &= y_{11} + y_{12} \\
x_2 &= -1 + y_{21} + y_{22} + y_{23} \\
0 &\leq y_{11} \leq x_{12} - x_{11} \\
0 &\leq y_{12} \leq x_{13} - x_{12} \\
0 &\leq y_{21} \leq x_{22} - x_{21} \\
0 &\leq y_{22} \leq x_{23} - x_{22} \\
0 &\leq y_{23} \leq x_{24} - x_{23} \\
y_{12} &\leq \frac{(x_{13} - x_{12})y_{11}}{x_{12} - x_{11}} \\
y_{22} &\leq \frac{(x_{23} - x_{22})y_{21}}{x_{22} - x_{21}} \\
y_{23} &\leq \frac{(x_{24} - x_{23})y_{22}}{x_{23} - x_{22}}
\end{aligned}
\tag{3.35}
$$

Chapter 4

NUMERICAL STUDY AND RESULTS

In this chapter, multiple choice model, convex combination model, incremental cost model and its LP relaxation are compared with our formulation. The comparisons base on the cpu time spend by each model to find the optimal solution of minimization/maximization problems of different structured PWL functions. The reason why we chose those modeling methods is that they are the most well known modeling methods in literature. We added LP relaxation of incremental cost model as well, since it can find the extrema of a PWL function in the existence of no other constraints (locally ideal property [42]) which is the same problem we are considering.

Our numerical study consists of two parts. In first part we consider benchmarks of small sized piecewise linear functions, i.e. number of break points up to 10. In second part, we consider larger instances of piecewise linear functions, i.e. break points more than 50.

During benchmarks, GAMS version 23.4 is used on a Intel Core 2 Duo CPU T 8100 @2.10 GHz 2 GB of RAM computer. A sample GAMS code from benchmark studies can be found in appendix.

4.1 Small Sized PWL Functions

All PWL functions stated in this section belong to the continuous PWL functions category. Within this category 4 different types of problems are taken into account and those are:

- Convex Piecewise Linear Functions
- Concave Piecewise Linear Functions
- Non-Convex Piecewise Linear Functions

- Separable Piecewise Continuous Linear Functions

We created 5 PWL functions from each category and solved the problem of finding their extrema both with our model and other modeling methods. We compared cpu times spend to solve both minimization and maximization problem of these functions with different types of modeling techniques. Details of each PWL function with corresponding performances of modeling methods can be found in appendix.

In table 4.1 we put all the run times together and in table 4.2 we indicated best performing models on each PWL function studied. In table 4.2, the ***"x"*** on a row indicates that the modeling method under which ***"x"*** placed has found the extrema of corresponding function in shortest time. For example minimum of fourth problem in convex PWL functions was solved fastest by our formulation and this is indicated with a ***"x"*** under *Our Formulation* column in row 7. More than one ***"x"*** on a row indicates that more than one modeling method found the optimal solution with shortest cpu time.

20 different functions were used for benchmark studies with small instances and their minimization and maximization result in 40 different optimization problems. In 30 of these problems our formulation has been one of the modeling methods that reached the optimal solution fastest and in 23 of these problems our formulation has been the only fastest method. Additionally, our formulation has always been one of the 2 fastest formulations in all benchmark problems. There were 5 PWL functions and 10 optimization problems in convex PWL functions category. In 5 of these problems our formulation has been one of the fastest modeling methods and in 4 of them our formulation has been the only fastest method. There were 5 PWL functions and 10 optimization problems in concave PWL functions category. In 7 of these problems our formulation has been one of the fastest modeling methods and again in 7 of them our formulation has been the only fastest method. There were 5 PWL functions and 10 optimization problems in non-convex PWL functions category. In 8 of these problems our formulation has been one of the fastest modeling methods and in 6 of them our formulation has been the only fastest method. There were 5 separable PWL functions and 10 optimization problems derived from them. In all of these problems our formulation has been one of the fastest modeling methods and in 6 of them

CPU Times Matrix				MODELING METHODS				
				Our Formulation	Convex Combination	Multiple Choice	Incremental Cost	Inc Cost (LP relaxation)
PROBLEM TYPES	Convex Continuous PWL Functions	P1	Min	0,027	0,125	0,028	0,031	0,026
			Max	0,012	0,092	0,030	0,110	0,011
		P2	Min	0,027	0,161	0,011	0,036	0,031
			Max	0,010	0,098	0,013	0,126	0,033
		P3	Min	0,029	0,118	0,030	0,035	0,028
			Max	0,011	0,040	0,033	0,038	0,034
		P4	Min	0,010	0,146	0,032	0,029	0,032
			Max	0,010	0,097	0,034	0,112	0,032
		P5	Min	0,010	0,101	0,012	0,017	0,010
			Max	0,020	0,116	0,024	0,030	0,011
	Concave Continuous PWL Functions	P1	Min	0,010	0,113	0,026	0,064	0,011
			Max	0,013	0,165	0,022	0,025	0,011
		P2	Min	0,029	0,120	0,031	0,046	0,027
			Max	0,010	0,141	0,032	0,030	0,028
		P3	Min	0,028	0,029	0,033	0,051	0,034
			Max	0,011	0,112	0,014	0,036	0,033
		P4	Min	0,030	0,101	0,031	0,040	0,027
			Max	0,010	0,092	0,028	0,034	0,032
		P5	Min	0,010	0,026	0,012	0,042	0,024
			Max	0,010	0,060	0,018	0,038	0,024
	Non-Convex Continuous PWL Functions	P1	Min	0,013	0,023	0,026	0,012	0,013
			Max	0,010	0,052	0,015	0,011	0,010
		P2	Min	0,024	0,045	0,028	0,024	0,033
			Max	0,015	0,059	0,030	0,024	0,024
		P3	Min	0,010	0,118	0,028	0,057	0,026
			Max	0,010	0,073	0,032	0,034	0,031
		P4	Min	0,028	0,106	0,029	0,035	0,010
			Max	0,010	0,098	0,034	0,033	0,012
		P5	Min	0,010	0,060	0,016	0,022	0,011
			Max	0,010	0,061	0,023	0,023	0,012
	Seperable Piecewise Continuous Linear Functions	P1	Min	0,010	0,033	0,011	0,011	0,010
			Max	0,011	0,047	0,027	0,056	0,020
		P2	Min	0,011	0,030	0,011	0,066	0,012
			Max	0,010	0,065	0,011	0,021	0,011
		P3	Min	0,011	0,093	0,011	0,040	0,011
			Max	0,012	0,140	0,012	0,023	0,015
		P4	Min	0,010	0,032	0,011	0,012	0,011
			Max	0,010	0,074	0,011	0,057	0,030
		P5	Min	0,010	0,026	0,012	0,011	0,012
			Max	0,010	0,037	0,011	0,021	0,020

Table 4.1: CPU Times of Modeling Methods in Benchmark studies with Small Instances

Best Performing Methods				MODELING METHODS				
				Our Formulation	Convex Combination	Multiple Choice	Incremental Cost	Inc Cost (LP relaxation)
PROBLEM TYPES	Convex Continuous PWL Functions	P1	Min					x
			Max					x
		P2	Min			x		
			Max	x				
		P3	Min					x
			Max	x				
		P4	Min	x				
			Max	x				
		P5	Min	x				x
			Max					x
	Concave Continuous PWL Functions	P1	Min	x				
			Max					x
		P2	Min					x
			Max	x				
		P3	Min	x				
			Max	x				
		P4	Min					x
			Max	x				
		P5	Min	x				
			Max	x				
	Non-Convex Continuous PWL Functions	P1	Min				x	
			Max	x				x
		P2	Min	x			x	
			Max	x				
		P3	Min	x				
			Max	x				
		P4	Min					x
			Max	x				
		P5	Min	x				
			Max	x				
	Seperable Piecewise Continuous Linear Functions	P1	Min	x				x
			Max	x				
		P2	Min	x		x		
			Max	x				
		P3	Min	x		x		x
			Max	x		x		
		P4	Min	x				
			Max	x				
		P5	Min	x				
			Max	x				

Table 4.2: Best Performances in Benchmark Studies with Small Instances

our formulation has been the only fastest method.

Most frequently (75% of time) our formulation has been the fastest method that found the optimal solution. It was followed by LP Relaxation of Incremental Cost model (30% of time), Multiple Choice Model (10% of time), Incremental Cost Model (5% of time) and Convex Combination model respectively (never). LP relaxation of Incremental Cost formulation has been one of the fastest modeling methods in 12 of 40 problems and in 8 of them it has been the only fastest modeling method. In 4 of all problems LP Relaxation of Incremental Cost formulation has been one of the fastest modeling methods and in 1 of them it has been the only fastest modeling method. In 2 of all problems Incremental Cost formulation has been one of the fastest modeling methods and in 1 one of them it has been the only fastest method. Convex Combination formulation has never been among the best performers. We summarized percentage of time being fastest model and being among fastest models in Table 4.3. When we take the average of cpu times for each modeling method, it is observed that our formulation finds the extrema fastest for general and all sub categories (blue cells in Table 4.4) of PWL functions. From table 4.4 we observe that our formulation has the least average CPU time as 0.0143 and its closest follower LP relaxation of Incremental Cost formulation's average CPU time is 0.0208. In other words, our formulation is 1.45 times faster than its closest follower.

% of Time a Model is Among Best Performers and % of Time a Model is the only Best Performer		MODELING METHODS				
		Our Formulation	Convex Combination	Multiple Choice	Incremental Cost	Inc Cost (LP Relaxation)
General View	Among Best Performers	75%	0%	10%	5%	30%
	Only Best Performer	57,5%	0%	2,5%	2,5%	20%
Convex Continuous PWL	Among Best Performers	50%	0%	10%	0%	50%
	Only Best Performer	40%	0%	10%	0%	40%
Concave Continuous PWL	Among Best Performers	70%	0%	0%	0%	30%
	Only Best Performer	70%	0%	0%	0%	30%
Non-Convex Continuous PWL	Among Best Performers	80%	0%	0%	20%	20%
	Only Best Performer	60%	0%	0%	10%	10%
Seperable Piecewise	Among Best Performers	100%	0%	30%	0%	20%
	Only Best Performer	60%	0%	0%	0%	0%

Table 4.3: Modeling Methods and Best Performance Distributions with Small Instances

Avg CPU Times Matrix		Our Formulation	Convex Combination	Multiple Choice	Incremental Cost	Inc Cost (LP relaxation)
PWL Function Types	Convex	0,0166	0,1094	0,0247	0,0564	0,0248
	Concave	0,0161	0,0959	0,0247	0,0406	0,0251
	Non-Convex	0,0140	0,0695	0,0261	0,0275	0,0182
	Seperable	0,0105	0,0577	0,0128	0,0318	0,0152
	All Functions	0,0143	0,0831	0,0221	0,0391	0,0208

Table 4.4: Average CPU Times of Modeling Methods in Benchmark studies with Small Instances

4.2 Larger Instances

In this section we consider large instances of piecewise linear functions that has fifty break points or more. For this purpose, we generated 3 separable piecewise continuous linear functions that have 5 component functions. All the component functions of first separable piecewise continuous linear functions have 50 break points. All the component functions of second separable piecewise continuous linear functions have 100 break points. All the component functions of third separable piecewise continuous linear functions have 200 break points.

Additionally, we approximated two functions from signal processing: Duffing map and henon map. Duffing map is defined by Eq. (4.1) and Henon map is defined by Eq.(4.2). We first initialized the first 2 iterations's values and using them we iterated mapping functions several times. For each iteration, we assigned a break point and initialized the first break point as 1 ($x_1 = 1$), while keeping the length of each interval as one. In this way if a mapping function is iterated 10 times, we obtained its PWL approximation with 10 break points and at each break point the value of the PWL function is equal to mapping functions value at corresponding iteration. The duffing map approximation consisted of 100 break points (100 iterations of duffing map) and henon map approximation consisted of 2000 break points (2000 iterations of henon map).

$$f(t) = -0.2f(t-1) + 2.75f(t-2) - f(t-1)^3 \tag{4.1}$$

$$f(t) = 1 - 1.4f(t-1)^2 + 0.3f(t-2) \tag{4.2}$$

We compared cpu times spend to solve both minimization and maximization problem of large PWL functions with different types of modeling techniques.

In table 4.5 we put all the CPU times for large instances and in table 4.6 we indicated best performing models on each large PWL function studied. Our formulation has been the most frequently best performing formulation followed by incremental cost formulation and its LP relaxation. In 9 out of 10 problems our formulation has found the optimal solution with the least CPU time and in 8 of them our formulation has been the unique solution that has found the optimal solution with the least CPU time. One of the observations that we gained from large instances is the following: As the number of break points increase, our formulations performance has dominated other formulations more. Our formulation has become more frequently best performing method when the sample PWL functions are large. Another observation is that: The gap between our formulation's CPU usage and its closest followers' CPU usage has increased. From table 4.7 we observe that our formulation has the least average CPU time as 0.0114 and its closest follower Incremental Cost formulation's average CPU time is 0.0287. In other words, our formulation is 2.5 times faster than its closest follower when the number of break points are high. This ratio was 1.45 for small instances.

Table 4.8 represents the distribution of time a modeling method has become one of the best performing methods among all benchmark studies (both small and large instances). Our formulation has become 78% of time one of the best performing methods in all benchmark studies. Moreover, 88% of time our formulation is one of the best performing methods, whereas this ratio decreases to 68% for minimization problems. In table 4.9 we put average CPU times of modeling methods in all benchmark studies. Our formulation has the least average CPU time as 0.014 and its closest follower LP relaxation of incremental cost formulation's average CPU time is 0.023. In other words, our formulation is 1.6 times faster than its closest follower in all benchmark studies.

CPU Times Matrix		Our Formulation	Convex Combination	Multiple Choice	Incremental Cost	Inc Cost (LP Relaxation)
Separable PWL Function 1	min	0,008	0,027	0,013	0,008	0,011
	max	0,008	0,018	0,011	0,01	0,012
Separable PWL Function 2	min	0,009	0,046	0,047	0,012	0,008
	max	0,01	0,022	0,022	0,048	0,018
Separable PWL Function 3	min	0,008	0,046	0,013	0,01	0,012
	max	0,008	0,012	0,033	0,042	0,049
Henon Map	min	0,008	0,01	0,025	0,024	0,022
	max	0,008	0,01	0,021	0,02	0,027
Duffing Map	min	0,031	0,062	0,116	0,038	0,046
	max	0,016	0,043	0,055	0,075	0,093

Table 4.5: CPU Times of Modeling Methods in Benchmark Studies with Large Instances

Best Performing Methods		Our Formulation	Convex Combination	Multiple Choice	Incremental Cost	Inc Cost (LP Relaxation)
Separable PWL Function 1	min	x			x	
	max	x				
Separable PWL Function 2	min					x
	max	x				
Separable PWL Function 3	min	x				
	max	x				
Henon Map	min	x				
	max	x				
Duffing Map	min	x				
	max	x				

Table 4.6: Best Performances in Benchmark Studies with Large Instances

Formulations	Average CPU Times		
	min	max	All
Our Formulation	0,011	0,008	0,011
Convex Combination	0,032	0,018	0,030
Multiple Choice	0,036	0,024	0,036
Incremental Cost	0,015	0,033	0,029
Inc Cost LP Relaxation	0,017	0,033	0,030

Table 4.7: Average CPU Times of Modeling Methods in Benchmark studies with Large Instances

Best Performances	All Instances		
	Min	Max	Overall
Our Formulation	68%	88%	78%
Convex Combination	0%	0%	0%
Multiple Choice	12%	4%	8%
Incremental Cost	12%	0%	6%
Inc Cost LP Relaxation	36%	16%	26%

Table 4.8: Modeling Methods and Best Performance Distributions in All Instances

Avg CPU Times in All Benchmarks	
Our Formulation	0,014
Convex Combination	0,072
Multiple Choice	0,025
Incremental Cost	0,037
Inc Cost LP Relaxation	0,023

Table 4.9: Average CPU Times of Modeling Methods in All Benchmark Studies

As we illustrated in Chapter 3 our formulation has lower complexity than other formulations and we validate this result in our benchmark studies as well. Our novel formulation has dominated other formulations both in general and in all sub categories of PWL functions studied in terms of CPU time. It has the highest frequency of being fastest modeling method and there is a huge gap between its nearest follower (1.45 times faster for small sized problems, 2.5 times faster for large sized problems, 1.6 faster for all benchmarks). We also observed that as the number of break points increase, the performance of our formulation dominates other formulations more significantly.

Chapter 5

CONCLUSIONS AND FUTURE WORK

In this thesis we proposed a novel linear programming formulation for finding the extrema of continuous piecewise linear functions. Although, modeling of piecewise linear functions is extensively studied in literature, this novel formulation differs from others in terms of complexity, run time and simplicity. We can group previous studies on modeling of piecewise linear functions into three categories. The first category of work includes the use of mixed integer programming (MIP) models. Several MIP formulations are proposed to model piecewise linear functions; incremental cost, convex combination and multiple choice models are the most well known of these formulations. They make use of binary variables and hence are hard to solve. On the other hand, second category of work concentrated on developing models that do not require the use of binary variables or decrease the number of binary variables in the models. They make use of the branch and bound, branch and cut algorithms combined with special ordered sets. The third group of studies focus on modeling piecewise linear functions with special structures useful for their formulation. For example, convex piecewise linear functions fall into this category. Their minimization can be modelled as LP without binary variables or a modified version of simplex algorithm exists to solve the same problem.

All of these attempts provide useful insights; however, these formulations are either complex or target a very specific class of PWL functions. In this thesis we presented a novel and effective linear programming formulation for finding the extrema of continuous PWL functions. We proved that our formulation finds the extrema of any continuous PWL function (convex, concave or non-convex) exactly. While developing this formulation we made use of two facts: First, simplex algorithm moves along the extreme points of the feasible region while searching for the optimal solution. Second, extrema of any continuous PWL function

lies at one of its break points. We developed a linear programming formulation with a special feasible region such that the extreme points of this region overlap with the break points of the corresponding PWL function. This property enables the simplex algorithm to find the extrema of PWL function exactly. Being free from binary variables, our formulation can be implemented and solved in a general purpose LP solver. Furthermore, our formulation has lower number variables and constraints than the existing formulations in the literature. As a result, all of these properties decrease the complexity of our formulation and the CPU time to find the optimal solution. We then extended our formulation to find the extrema of separable piecewise continuous linear functions.

We supported our findings computationally by benchmarking our formulation with the most common formulations in the literature. We derived PWL functions with different structural properties and found their extrema using both our formulation and 4 different formulations from literature. The comparison based on the CPU times spend to find the optimal solution using each formulation method. Benchmark studies showed that our formulation is performing far better than other formulation types. 78% of time our formulation was one of the formulations that found the optimal solution with shortest cpu time and 64% of time our formulation was the fastest model alone. Additionally, our method was 100% of time among the fastest 2 methods. The nearest follower of our model was the LP relaxation of incremental cost formulation. It was 26% of time among best performers and 18% of time the fastest model alone.

In conclusion, our contribution to the literature with this thesis is threefold. First, we developed a general formulation that can be used to model all classes of continuous PWL functions. Previous LP formulations (without binary variables) were targeting specific types of problems, such as minimizing convex PWL functions or maximizing concave PWL functions. Second, our formulation does not include binary variables, hence it has a simple structure and can be solved with a general purpose LP solver. Other formulations covering all classes of PWL functions were harder to implement. They either include binary variables or reach the solution step by step using branching algorithms and special ordered sets. Third, the complexity of our formulation is less than the existing methods and it reaches

the optimal solution in shorter time.

We believe there are two directions for future research. First direction is extending our formulation to cover the discontinuous PWL functions and the second direction is extending our formulation to find the extrema of PWL functions when there are other constraints.

BIBLIOGRAPHY

[1] E.H. Aghezzaf and L.A. Wolsey. Modeling piecewise linear concave costs in tree partitioning problem. *Discrete Appl. Math.*, 50:101–109, 1994.

[2] A.A. Andronov, A.A. Vitt, and S.E. Kaikin. *Theory of Oscillators*. Newyork: Pergamon Press, 1966.

[3] A. Balakrishnan and S. Graves. A composite algorithm for a concave-cost network flow problem. *Networks*, 19:175–202, 1989.

[4] E.L.M. Beale and J.A. Tomlin. Special facilities in a general mathematical programming system for nonconvex problems using ordered sets of variables. *Operational Research*, 69, 1970.

[5] D. Beinstock and O. Günlük. Capacitated network design - polyhedral structure and computation. *INFORMS J. Computing*, 8:243–259, 1996.

[6] J. Black. *The Oxford Dictionary of Economics*. Oxford University Press, 2002.

[7] A. Charnes and C.E. Lemke. Minimization of nonlinear separable convex functionals. *Naval Research Logistics Quarterly*.

[8] L.O. Chua. *Introduction to Nonlinear Network Theory*. Newyork: McGraw-Hill, 1969.

[9] L.O. Chua and S.M Kang. Section-wise piecewise-linear functions: Canonical representation, properties,and applications. *Proceedings of the IEEE*, 65:915–929, 1977.

[10] K.K. Clarke and D.T. Hess. *Communication Circuits: Analysis and Design*. Reading, MA: Addison - Weasley, 1971.

[11] K.S. Cole. *Membranes, Ions and Impulses*. Berkeley C.A. : University of California Press, 1972.

[12] K. L. Croxton, B. Gendron, and T. L. Magnanti. A comparison of mixed-integer programming models for nonconvex piecewise linear cost minimization problems. *Management Science*, 49:1268–1273, 2003.

[13] K. L. Croxton, B. Gendron, and T. L Magnanti. Variable disaggregation in network flow problems with piecewise linear costs. *Operations Research*, 55:146–157, 2007.

[14] K.L. Croxton. *Modeling and Solving Network Flow Problems with Piecewise Linear Costs with Applications in Supply Chain Management*. Phd. thesis, Operations Research Center, MIT, Cambridge, 1999.

[15] K.L. Croxton, B. Gendron, and T.L. Magnanti. Models and methods for merge in transit operations. *Transportation Sci.*, 37:1–22, 2003.

[16] G.B. Dantzig. Recent advances in linear programming. *Management Science*, 2:131–144, 1956.

[17] G.B. Dantzig. On the significance of solving linear programming problems with some integer variables. *Econometrica*, 1960.

[18] G.B. Dantzig. *Linear Programming and Extensions*. Princeton University Press, 1963.

[19] G.B. Dantzig, S. Johnson, and W. White. A linear programming approach to the chemical equilibrium problem. *Management Science*, 5:38–43, 1958.

[20] G.B. Dantzig and M.N. Thapa. *Linear Programming 1: Introduction*. Springer - Verlag, 1997.

[21] I.R. de Farias, M. Zhao, and H. Zhao. A special ordered sets approach to discontinuous piecewise linear optimization. submitted.

[22] J.B. Dennis. *Mathematical Programming and Electrical Networks.* Newyork: Wiley, 1959.

[23] I. Flügge-Lotz. *Discontinuous and Optimal Control.* Newyork: McGraw-Hill, 1948.

[24] R. Fourer. A simplex algorithm for piecewise linear programming i: Derivation and proof. 1985.

[25] R. Fourer. A simplex algorithm for piecewise linear programming ii: Finiteness, feasibility and degeneracy. 1988.

[26] R. Fourer. A simplex algorithm for piecewise linear programming iii: Computational analysis and applications. 1992.

[27] V. Gabrel, A. Knippel, and M. Minoux.

[28] O. Günlük. A branch and bound algorithm for capacitated netwrok design problems. *Math. Programming*, 2:82–104, 1999.

[29] D. Hartog. *Mechanics.* Newyork: Dover, 1948.

[30] J. Ho. Relationships among linear formulations of separable convex piecewise linear programs. *Mathematical Programming Study*, 24:126–140, 1985.

[31] K. Holmberg. Solving the staircase cost facility location problem with decomposition and piecewise linearization. *Eur J. of Oper. Res.*, 75:41–61, 1994.

[32] K. Holmberg and J. Ling. A lagrangean heuristic for the facility location problem with staircase costs. *Eur. J. of Oper. Res.*, 97:63–74, 1997.

[33] J.F.P. Hudson. *Piecewise Linear Topology.* Newyork: B.A. Benjamin, 1969.

[34] M. Iri. *Network Flow, Transporation and Scheduling.* Newyork: Academic Press, 1969.

[35] A.B. Keha, I.R. de Farias, and G.L. Nemhauser. Models for representing piecewise linear cost functions. *Operations Research Letters*, 31:44–48, 2004.

[36] A.B. Keha, I.R. de Farias, and G.L. Nemhauser. A branch-and-cut algorithm without binary variables for nonconvex piecewise linear optimization. *Operations Research*, 54:847–858, 2006.

[37] H.L. Li, H.C. Lu, C.H. Huang, and N.Z. Hu. A superior representation method for piecewise linear functions. *NFORMS Journal on Computing*, 21:314–321, 2009.

[38] T.L. Magnanti, P. Mirchandani, and R. Vachani. Modeling and solwing the two-facility capacitated network loading problem. *Operations Research*, 43:142–157, 1995.

[39] O.L. Mangasarian, J.B. Rosen, and M.E. Thompson. Global minimization via piecewise-linear underestimation. *Journal of Global Optimization*, pages 1–9, 2004.

[40] H.M. Markowitz and A.S. Manne. On the solution of discrete programming problems. *Econometrica*, 25:84–110, 1957.

[41] S. Osowski. Optimisation approach to the analysis of piecewise-linear convex circuits. *IEE Proceedings*, 139:295–300, 1992.

[42] M. Padberg. Approximating separable nonlinear functions via mixed zero-one programs. *Operations Research Letters*, 27:1–5, 2000.

[43] M. Padberg and M. Rijal. *Location, Scheduling, Design and Integer Programming.* Kluwer Academic Publishers, Boston, 1996.

[44] G. Rich. *Hydraulic Transients.* Newyork: Dover, 1962.

[45] E. Schmidt. *Thermodynamics.* Newyork: Dover, 1966.

[46] H.D. Sherali. On mixed-integer zero-one representations for separable lower-semicontinuous piecewise-linear functions. *Operations Research Letters*, pages 155–160, 2001.

[47] R.D. Snyder. Linear programming with special ordered sets. *Journal of Operational Research Society*, 35:69–74, 1984.

[48] T.E. Stern. *Theory of Nonlinear Networks and Systems, An Introduction.* Reading, MA: Addison - Weasley, 1965.

[49] S. Wang and X. Sun. *Advances in Neural Networks*, chapter A Special Kind of Neural Networks: Continuous Piecewise Linear Functions, pages 375–379. Springer Berlin, 2005.

[50] O.C. Zienkiewcz. *The Finite Element Method in Engineering Science.* Newyork: McGraw - Hill, 1971.

[51] H.J. Zimmerman and S.J. Mason. *Electronic Circuit Thoery.* Newyork: Wiley, 1959.

Appendix A

TYPES OF PIECEWISE LINEAR FUNCTIONS

Continuous Piecewise Linear Function A piecewise linear function is classified as *continuous* if the following is satisfied [20]:

$$\begin{aligned} a_1x_2 + b_1 &= a_2x_2 + b_2 \\ a_2x_3 + b_2 &= a_3x_3 + b_3 \\ ... &= ... \\ ... &= ... \\ a_{t-2}x_{t-1} + b_{t-2} &= a_{t-1}x_{t-1} + b_{t-1} \end{aligned} \tag{A.1}$$

This set of equations provides that the values of local functions on both sides of each break point are exactly the same [49]. For example, Figure A.1 is the graph of a continuous piecewise linear function described by Eq. (A.2):

$$f(x) = \begin{cases} 4x + 3 & \text{if } 0 \leq x \leq 2 \\ -2x + 15 & \text{if } 2 \leq x \leq 3 \\ x + 6 & \text{if } 3 \leq x \leq 7 \end{cases} \tag{A.2}$$

Here, the function is defined on the domain [0,7] with the break points $x_1 = 0$, $x_2 = 2$, $x_3 = 3$ and $x_=7$. Local functions on both sides of the break points has the same value. For example at second break point $x_2 = 2$, the value of the function does not change. When we calculate $f(2)$ using equations in the first interval $f(x) = 4x + 3$ or in the second interval $f(x) = -2x + 15$, the same value is obtained $f(2) = 11$. The same is true at break point $x_3 = 3$ too. When we calculate $f(3)$ using equations in the

second interval $f(x) = -2x + 15$ or in the third interval $f(x) = x + 6$, the same value is obtained $f(3) = 9$.

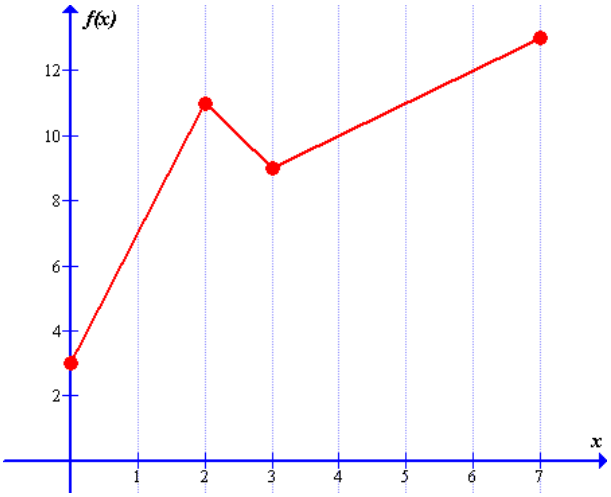

Figure A.1: A Continuous Piecewise Linear Function

On the other hand, figure A.2 is the graph of a discontinuous piecewise linear function described by Eq. (A.3):

$$f(x) = \begin{cases} 3x + 7 & \text{if } 0 \leq x \leq 3 \\ -2x + 10 & \text{if } 3 \leq x \leq 5 \\ x & \text{if } 5 \leq x \leq 6 \end{cases} \tag{A.3}$$

In this example, the function is defined on the domain [0,6] with the break points $x_1 = 0$, $x_2 = 3$, $x_3 = 5$ and $x_=6$. Local functions on both sides of at least one break point does not have the same value. For example at second break point $x_2 = 3$, the value of the function changes. When we calculate $f(3)$ using the equation in first interval $f(x) = 3x + 7$, the result is $f(3) = 16$; whereas, when we calculate $f(3)$ using equation in second interval $f(x) = -2x + 10$, the result is $f(3) = 4$. $16 \neq 4$ and continuity conditions in Eq. (A.1)are not satisfied; hence, the function is classified as

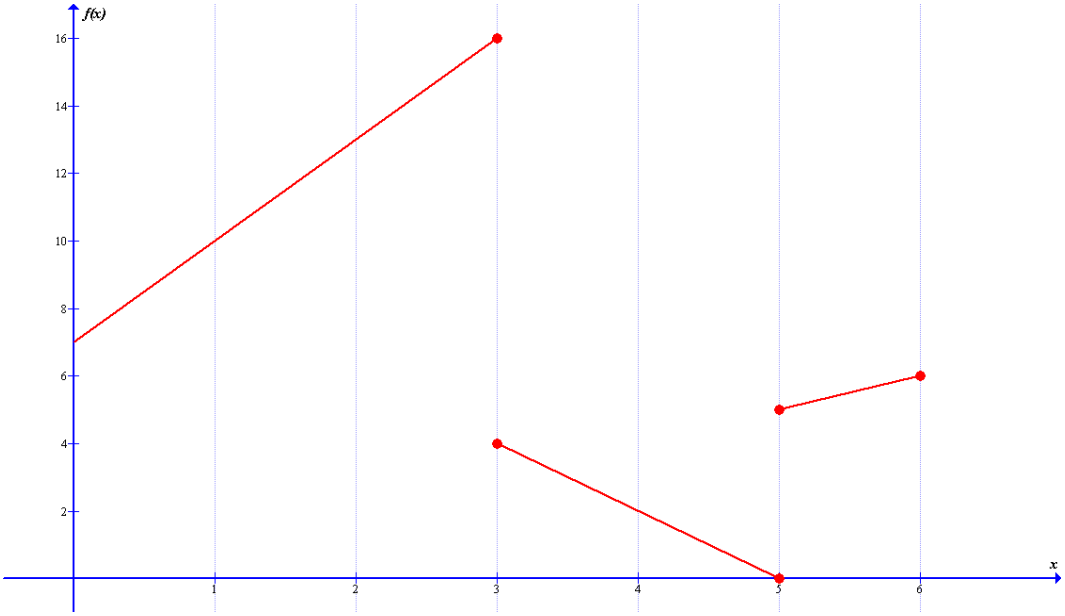

Figure A.2: A Discontinuous Piecewise Linear Function

discontinuous.

Convex Piecewise Linear Function A piecewise linear function is classified as *convex* if the following is satisfied [20]:

$$a_j \leq a_{j+1} \ \forall \ j \in 1, ..., t-1 \tag{A.4}$$

For example, figure A.3 is the graph of a convex piecewise linear function described by Eq. (A.5):

$$f(x) = \begin{cases} -4x + 2 & \text{if } -2 \leq x \leq 0 \\ -3x + 2 & \text{if } 0 \leq x \leq 2 \\ x - 6 & \text{if } 2 \leq x \leq 5 \\ 2x - 11 & \text{if } 5 \leq x \leq 6 \\ 3x - 17 & \text{if } 6 \leq x \leq 9 \end{cases} \tag{A.5}$$

The slopes of the function in each segment satisfy the condition in Eq. (A.4), i.e. $a_1 = -4 \leq a_2 = -3 \leq a_3 = 1 \leq a_4 = 2 \leq a_5 = 3$.

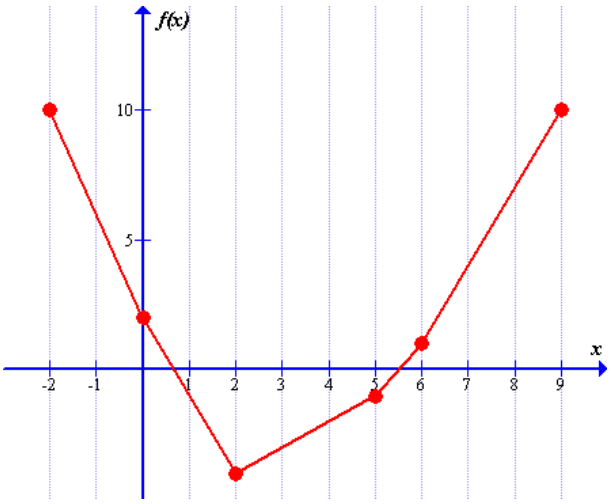

Figure A.3: A Convex Piecewise Linear Function

Concave Piecewise Linear Function A piecewise linear function is classified as *concave* if the following is satisfied [20]:

$$a_j \geq a_{j+1} \ \forall \ j \in 1, ..., t-1 \tag{A.6}$$

Figure A.4 is graph a concave piecewise linear function described by Eq. (A.7) :

$$f(x) = \begin{cases} 10x + 20 & \text{if } 5 \leq x \leq 15 \\ 6x + 80 & \text{if } 15 \leq x \leq 20 \\ 200 & \text{if } 20 \leq x \leq 30 \\ -2x + 260 & \text{if } 30 \leq x \leq 40 \\ -10x + 580 & \text{if } 40 \leq x \leq 45 \end{cases} \tag{A.7}$$

The slopes of the function in each segment satisfy the condition in Eq. (A.6), i.e.

$$a_1 = 10 \geq a_2 = 6 \geq a_3 = 0 \geq a_4 = -2 \geq a_5 = -10.$$

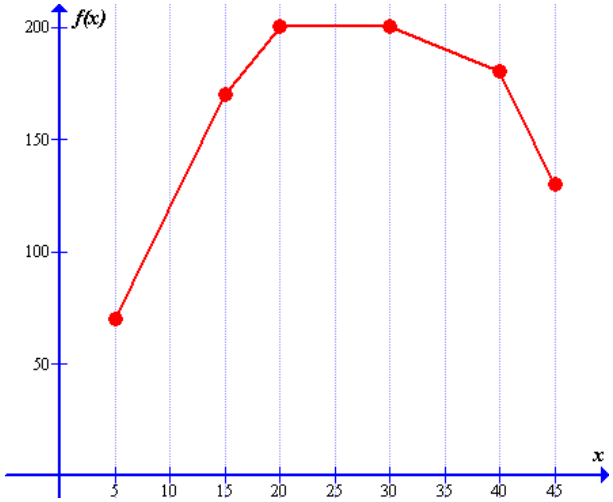

Figure A.4: A Concave Piecewise Linear Function

From this definitions the following is observed: Convexity (concavity) is related with the slope of the cost function in each segment. If the slope of the function in each segment is non-decreasing (non-increasing), then the piecewise linear function is convex (concave).

Non-convex Piecewise Linear Function If a continuous piecewise linear function is neither convex nor concave, then it is classified as *non-convex*. Figures A.1 and A.2 are examples of piecewise linear functions that are neither convex nor concave, hence defined as non-convex piecewise linear functions.

Appendix B

DETAILS OF PROBLEMS IN BENCHMARK STUDIES

B.1 Benchmark of Convex PWL Functions

Function 1: The first convex PWL function used for benchmark is the function used to illustrate a convex PWL function. It was defined by equation A.5 and represented in Fig. A.3. Its minimization and maximization with different models yield the results in Table B.1. When the objective is both minimization and maximization, the LP relaxation of incremental cost formulation finds the optimal solution with the least cpu time.

Table B.1: CPU times for Convex PWL Function Example-1

	Minimize	Maximize
Our Formulation	0.027	0.012
Convex Combination	0.125	0.092
Multiple Choice	0.028	0.030
Incremental Cost	0.031	0.110
Incremental Cost (LP Relaxation)	0.026	0.011

Function 2: The second convex PWL function is defined by equation B.1 and represented in Fig. B.1.

$$f(x) = \begin{cases} -10x - 20 & \text{if } 5 \leq x \leq 15 \\ -6x - 80 & \text{if } 15 \leq x \leq 20 \\ -200 & \text{if } 20 \leq x \leq 30 \\ 2x - 260 & \text{if } 30 \leq x \leq 40 \\ 10x - 580 & \text{if } 40 \leq x \leq 45 \end{cases} \tag{B.1}$$

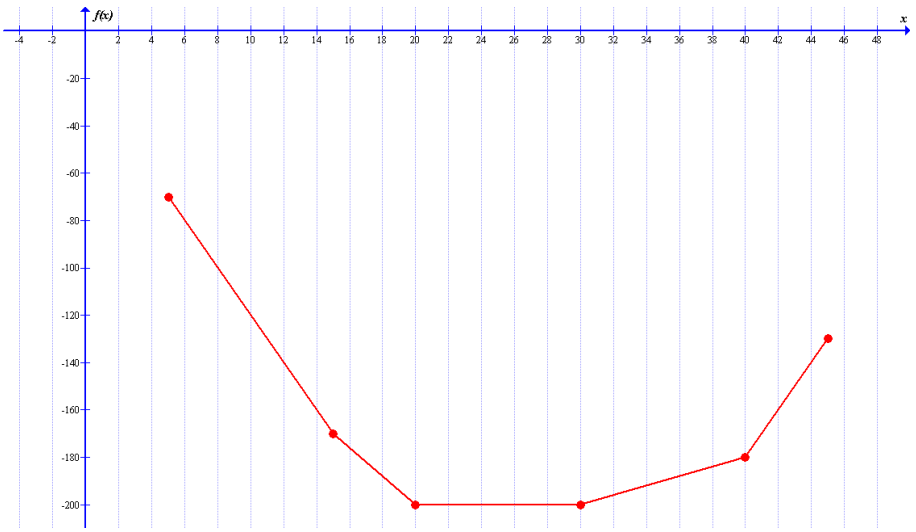

Figure B.1: Convex PWL Function - 2

Its minimization and maximization with different models yield the results in Table B.2. When the objective is minimization, multiple choice formulation finds the optimal solution with the least cpu time. On the other hand, when the objective is maximization our formulation finds the optimal solution with the least cpu time.

Table B.2: CPU times for Convex PWL Function Example-2

	Minimize	Maximize
Our Formulation	0.027	0.010
Convex Combination	0.161	0.098
Multiple Choice	0.011	0.013
Incremental Cost	0.036	0.126
Incremental Cost (LP Relaxation)	0.031	0.033

Function 3: The third convex PWL function is defined by equation B.2 and represented in Fig. B.2.

$$f(x) = \begin{cases} -6x + 10 & \text{if } -10 \leq x \leq -2 \\ -5x + 12 & \text{if } -2 \leq x \leq 1 \\ -3x + 10 & \text{if } 1 \leq x \leq 5 \\ -x & \text{if } 5 \leq x \leq 13 \\ -13 & \text{if } 13 \leq x \leq 20 \\ 0.5x - 23 & \text{if } 20 \leq x \leq 25 \end{cases} \tag{B.2}$$

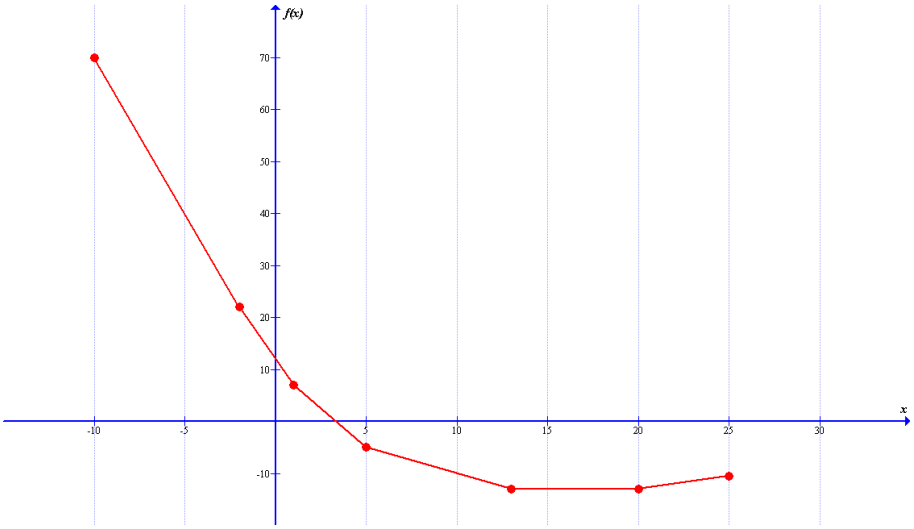

Figure B.2: Convex PWL Function - 3

Its minimization and maximization with different models yield the results in Table B.3. When the objective is minimization the LP relaxation of incremental cost formulation finds the optimal solution with least cpu time. On the other hand, when the objective if maximization our formulation finds the optimal solution with the least cpu time.

Function 4: The fourth convex PWL function is defined by equation B.3 and represented in Fig. B.3.

Table B.3: CPU times for Convex PWL Function Example-3

	Minimize	Maximize
Our Formulation	0.029	0.011
Convex Combination	0.118	0.040
Multiple Choice	0.030	0.033
Incremental Cost	0.035	0.038
Incremental Cost (LP Relaxation)	0.028	0.034

$$f(x) = \begin{cases} -0.5x + 8 & \text{if } -10 \leq x \leq -2 \\ 9 & \text{if } -2 \leq x \leq 1 \\ x + 8 & \text{if } 1 \leq x \leq 5 \\ 3x - 2 & \text{if } 5 \leq x \leq 13 \\ 5x - 28 & \text{if } 13 \leq x \leq 20 \\ 6x - 48 & \text{if } 20 \leq x \leq 25 \end{cases} \tag{B.3}$$

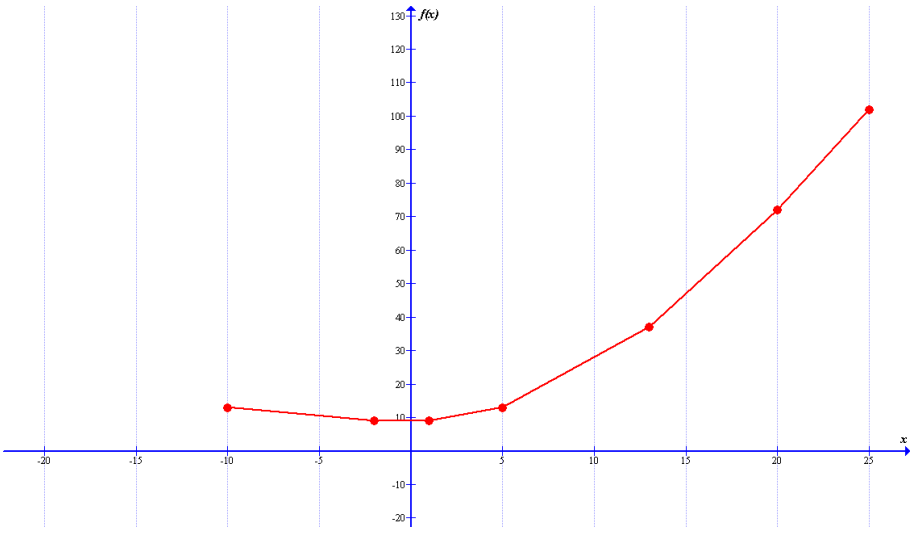

Figure B.3: Convex PWL Function - 4

Its minimization and maximization with different models yield the results in Table B.4.

When the objective is both minimization and maximization, our formulation finds the optimal solution with the least cpu time.

Table B.4: CPU times for Convex PWL Function Example-4

	Minimize	Maximize
Our Formulation	0.010	0.010
Convex Combination	0.146	0.097
Multiple Choice	0.032	0.034
Incremental Cost	0.029	0.112
Incremental Cost (LP Relaxation)	0.032	0.032

Function 5: The fifth convex PWL function is defined by equation B.4 and represented in Fig. B.4.

$$f(x) = \begin{cases} -x + 5 & \text{if } -80 \leq x \leq -70 \\ -0.75x + 22.5 & \text{if } -70 \leq x \leq -65 \\ -0.5x + 38.75 & \text{if } -65 \leq x \leq -40 \\ -0.25x + 48.75 & \text{if } -40 \leq x \leq -5 \\ 50 & \text{if } -5 \leq x \leq 0 \\ 0.25x + 50 & \text{if } 0 \leq x \leq 10 \\ 0.5x + 47.5 & \text{if } 10 \leq x \leq 60 \\ 0.75x + 32.5 & \text{if } 60 \leq x \leq 70 \\ x + 15 & \text{if } 70 \leq x \leq 80 \end{cases} \tag{B.4}$$

Its minimization and maximization with different models yield the results in Table B.5. When the objective is minimization, both the LP relaxation of incremental cost formulation and our formulation find the optimal solution with least cpu time. On the other hand, when the objective is maximization, the LP relaxation of incremental cost formulation finds the optimal solution with the least cpu time.

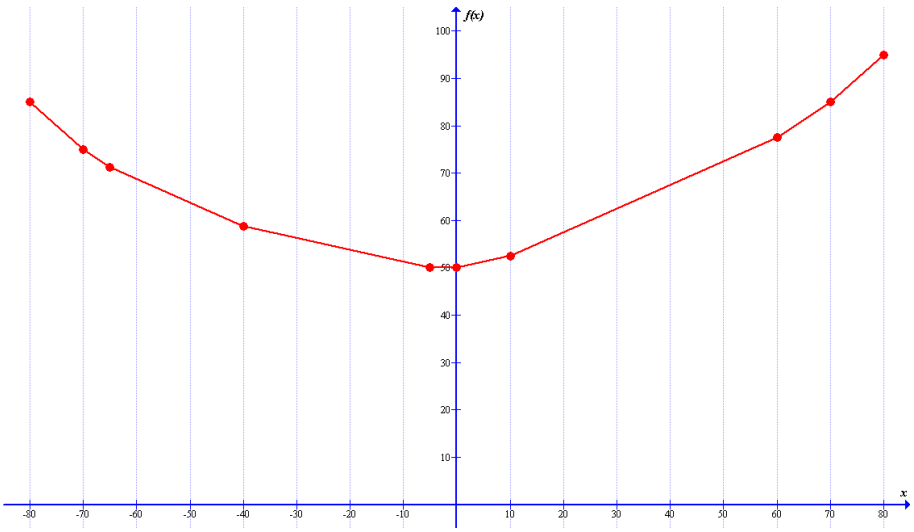

Figure B.4: Convex PWL Function - 5

Table B.5: CPU times for Convex PWL Function Example-5

	Minimize	Maximize
Our Formulation	0.010	0.020
Convex Combination	0.101	0.116
Multiple Choice	0.012	0.024
Incremental Cost	0.017	0.030
Incremental Cost (LP Relaxation)	0.010	0.011

B.2 Benchmark of Concave PWL Functions

Function 1: The first concave PWL function used for benchmark is the function used to illustrate a concave PWL function. It was defined by equation A.7 and represented in Fig. A.4. Its minimization and maximization with different models yield the results in Table B.6. When the objective is minimization, our formulation finds the optimal solution with the least cpu time. On the other hand, when the objective is maximization the LP relaxation of incremental cost formulation finds the optimal solution with the least cpu time.

Function 2: The second concave PWL function is defined by equation B.5 and represented in Fig. B.5.

Table B.6: CPU times for Concave PWL Function Example-1

	Minimize	Maximize
Our Formulation	0.010	0.013
Convex Combination	0.113	0.165
Multiple Choice	0.026	0.022
Incremental Cost	0.064	0.025
Incremental Cost (LP Relaxation)	0.011	0.011

$$f(x) = \begin{cases} 4x - 2 & \text{if } -2 \leq x \leq 0 \\ 3x - 2 & \text{if } 0 \leq x \leq 2 \\ -x + 6 & \text{if } 2 \leq x \leq 5 \\ 2x + 11 & \text{if } 5 \leq x \leq 6 \\ -3x + 17 & \text{if } 6 \leq x \leq 9 \end{cases} \tag{B.5}$$

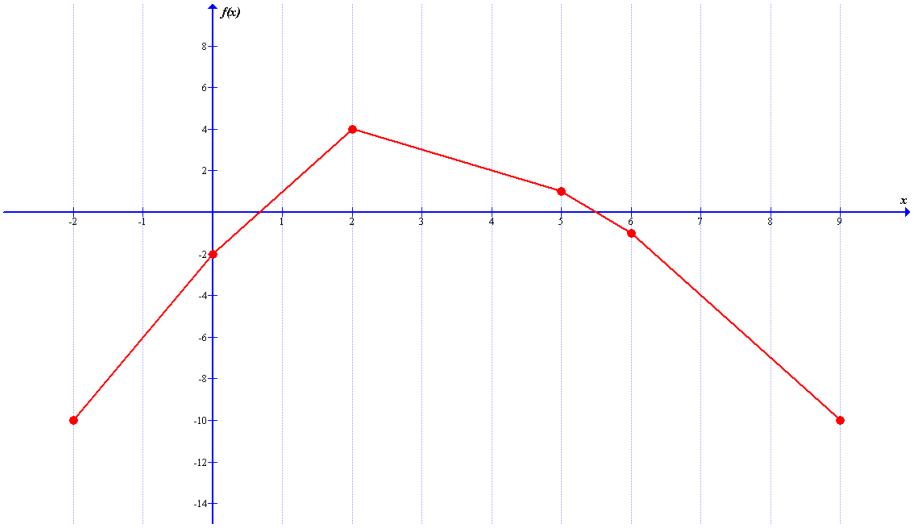

Figure B.5: Concave PWL Function - 2

Its minimization and maximization with different models yield the results in Table B.7. When the objective is minimization, the LP relaxation of incremental cost formulation

finds the optimal solution with least cpu time. On the other hand, when the objective is maximization, our formulation finds the optimal solution with the least cpu time.

Table B.7: CPU times for Concave PWL Function Example-2

	Minimize	Maximize
Our Formulation	0.029	0.010
Convex Combination	0.120	0.141
Multiple Choice	0.031	0.032
Incremental Cost	0.046	0.030
Incremental Cost (LP Relaxation)	0.027	0.028

Function 3: The third concave PWL function is defined by equation B.6 and represented in Fig. B.6.

$$f(x) = \begin{cases} 6x - 10 & \text{if } -10 \leq x \leq -2 \\ 5x - 12 & \text{if } -2 \leq x \leq 1 \\ 3x - 10 & \text{if } 1 \leq x \leq 5 \\ x & \text{if } 5 \leq x \leq 13 \\ 13 & \text{if } 13 \leq x \leq 20 \\ -0.5x + 23 & \text{if } 20 \leq x \leq 25 \end{cases} \tag{B.6}$$

Its minimization and maximization with different models yield the results in Table B.8. When the objective is both minimization and maximization our formulation finds the optimal solution with the least cpu time.

Function 4: The fourth concave PWL function is defined by equation B.7 and represented in Fig. B.7.

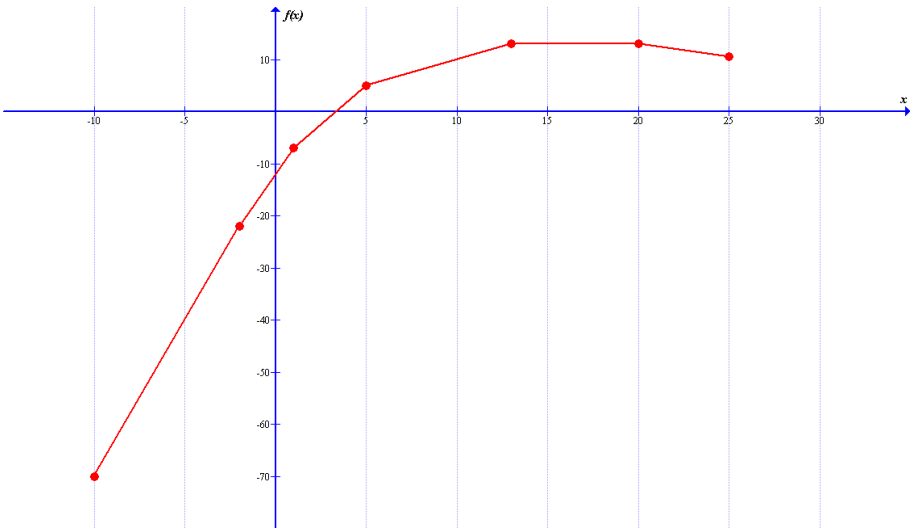

Figure B.6: Concave PWL Function - 3

Table B.8: CPU times for Concave PWL Function Example-3

	Minimize	Maximize
Our Formulation	0.028	0.011
Convex Combination	0.029	0.112
Multiple Choice	0.033	0.014
Incremental Cost	0.051	0.036
Incremental Cost (LP Relaxation)	0.034	0.033

$$f(x) = \begin{cases} 0.5x - 8 & \text{if } -10 \leq x \leq -2 \\ -9 & \text{if } -2 \leq x \leq 1 \\ -x - 8 & \text{if } 1 \leq x \leq 5 \\ -3x + 2 & \text{if } 5 \leq x \leq 13 \\ -5x + 28 & \text{if } 13 \leq x \leq 20 \\ -6x + 48 & \text{if } 20 \leq x \leq 25 \end{cases} \tag{B.7}$$

Its minimization and maximization with different models yield the results in Table B.9.

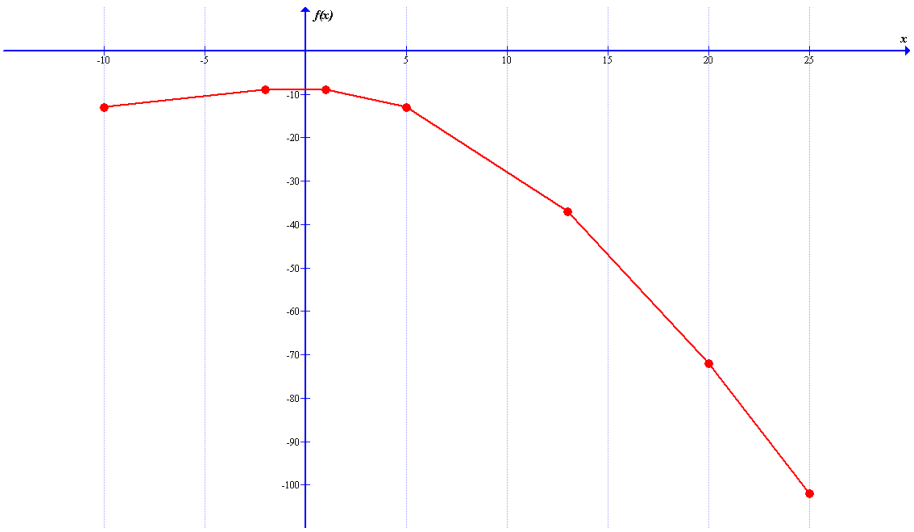

Figure B.7: Concave PWL Function - 4

When the objective is minimization, the LP relaxation of incremental cost formulation finds the optimal solution with least the cpu time. On the other hand when the objective is maximization our formulation finds the optimal solution with the least cpu time.

Table B.9: CPU times for Concave PWL Function Example-4

	Minimize	Maximize
Our Formulation	0.030	0.010
Convex Combination	0.101	0.092
Multiple Choice	0.031	0.028
Incremental Cost	0.040	0.034
Incremental Cost (LP Relaxation)	0.027	0.032

Function 5: The fifth concave PWL function is defined by equation B.8 and represented in Fig. B.8.

$$f(x) = \begin{cases} x - 5 & \text{if } -80 \leq x \leq -70 \\ 0.75x - 22.5 & \text{if } -70 \leq x \leq -65 \\ 0.5x - 38.75 & \text{if } -65 \leq x \leq -40 \\ 0.25x - 48.75 & \text{if } -40 \leq x \leq -5 \\ -50 & \text{if } -5 \leq x \leq 0 \\ -0.25x - 50 & \text{if } 0 \leq x \leq 10 \\ -0.5x - 47.5 & \text{if } 10 \leq x \leq 60 \\ -0.75x - 32.5 & \text{if } 60 \leq x \leq 70 \\ -x - 15 & \text{if } 70 \leq x \leq 80 \end{cases} \tag{B.8}$$

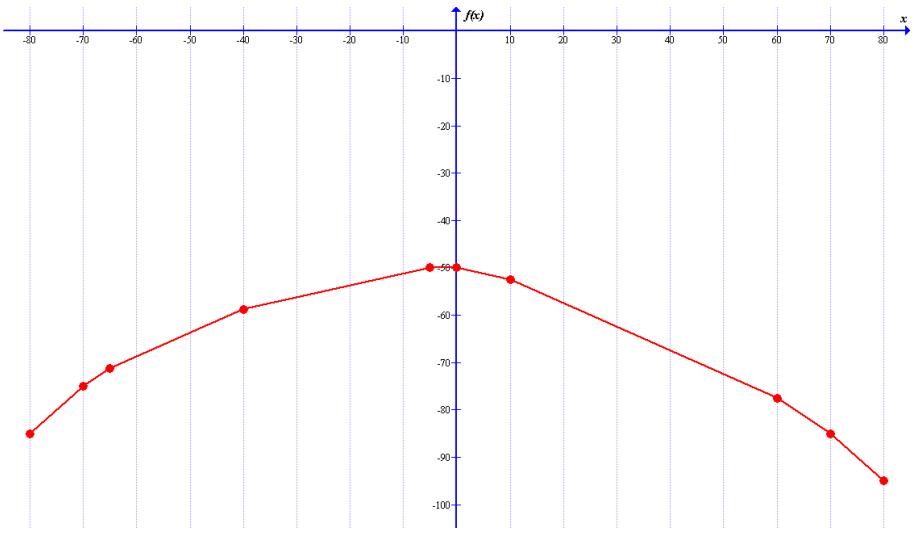

Figure B.8: Concave PWL Function - 5

Its minimization and maximization with different models yield the results in Table B.10. When the objective is both minimization and maximization our formulation finds the optimal solution with the least cpu time.

Table B.10: CPU times for Concave PWL Function Example-5

	Minimize	Maximize
Our Formulation	0.010	0.010
Convex Combination	0.026	0.060
Multiple Choice	0.012	0.018
Incremental Cost	0.042	0.038
Incremental Cost (LP Relaxation)	0.024	0.024

B.3 Benchmark of Non-Convex PWL Functions

Function 1: The first non-convex PWL function used for benchmark is the function used to illustrate a continuous PWL function. It was defined by equation A.2 and represented in Fig. A.1. Its minimization and maximization with different models yield the results in Table B.11. When the objective is minimization, incremental cost formulation finds the optimal solution with least cpu time. On the other hand, when the objective is maximization, both the LP relaxation of incremental cost formulation and our formulation find the optimal solution with the least cpu time.

Table B.11: CPU times for Non-Convex PWL Function Example-1

	Minimize	Maximize
Our Formulation	0.013	0.010
Convex Combination	0.023	0.052
Multiple Choice	0.026	0.015
Incremental Cost	0.012	0.011
Incremental Cost (LP Relaxation)	0.013	0.010

Function 2: The second non-convex PWL function is defined by equation B.9 and represented in Fig. B.9.

$$f(x) = \begin{cases} 0.2x + 5 & \text{if } 0 \leq x \leq 10 \\ 0.3x + 4 & \text{if } 10 \leq x \leq 20 \\ 0.5x & \text{if } 20 \leq x \leq 30 \\ -0.4x + 27 & \text{if } 30 \leq x \leq 40 \\ 0.6x - 13 & \text{if } 40 \leq x \leq 50 \end{cases} \tag{B.9}$$

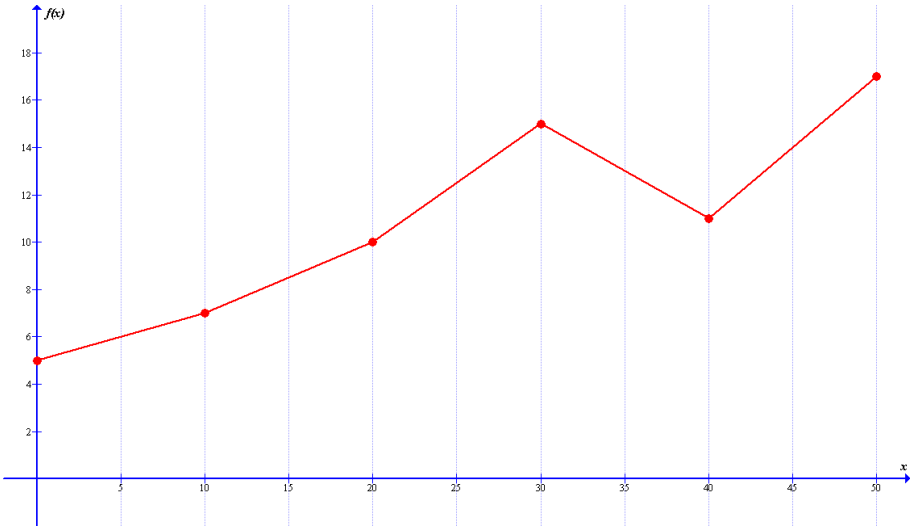

Figure B.9: Non-Convex PWL Function - 2

Its minimization and maximization with different models yield the results in Table B.12. When the objective is minimization our formulation and incremental cost formulation find the optimal solution with the least cpu time. On the other hand, when the objective is maximization, our formulation finds the optimal solution with least cpu time.

Function 3: The third non-convex PWL function is defined by equation B.10 and represented in Fig. B.10.

Table B.12: CPU times for Non-Convex PWL Function Example-2

	Minimize	Maximize
Our Formulation	0.024	0.015
Convex Combination	0.045	0.059
Multiple Choice	0.028	0.030
Incremental Cost	0.024	0.024
Incremental Cost (LP Relaxation)	0.033	0.024

$$f(x) = \begin{cases} 1.667x & \text{if } 0 \leq x \leq 30 \\ x + 20 & \text{if } 30 \leq x \leq 60 \\ -2x + 200 & \text{if } 60 \leq x \leq 90 \\ 2.667x - 220 & \text{if } 90 \leq x \leq 120 \end{cases} \tag{B.10}$$

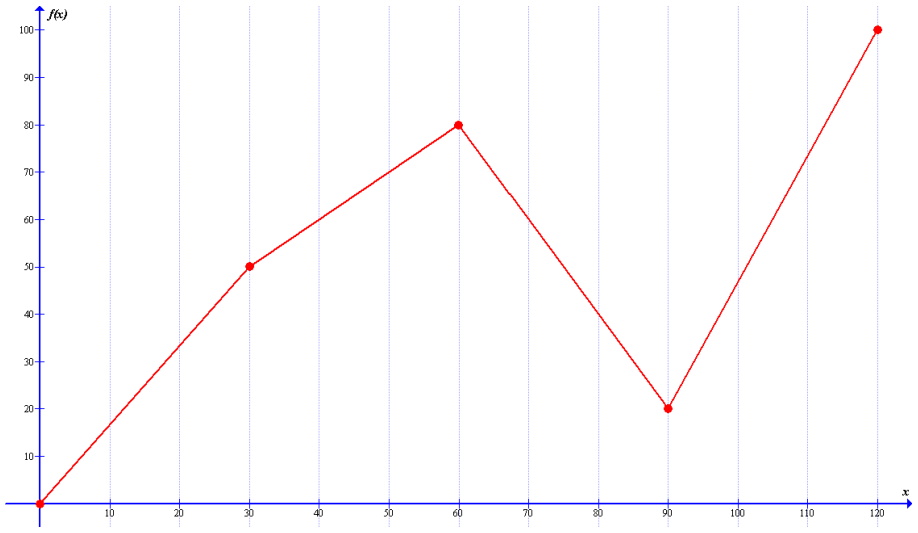

Figure B.10: Non-Convex PWL Function - 3

Its minimization and maximization with different models yield the results in Table B.13. When the objective is both minimization and maximization our formulation finds the optimal solution with least cpu time.

Table B.13: CPU times for Non-Convex PWL Function Example-3

	Minimize	Maximize
Our Formulation	0.010	0.010
Convex Combination	0.118	0.073
Multiple Choice	0.028	0.032
Incremental Cost	0.057	0.034
Incremental Cost (LP Relaxation)	0.026	0.031

Function 4: The fourth non-convex PWL function is defined by equation B.11 and represented in Fig. B.11.

$$f(x) = \begin{cases} 3x + 5 & \text{if } -20 \leq x \leq -14 \\ 5x + 33 & \text{if } -14 \leq x \leq -5 \\ -0.5x + 5.5 & \text{if } -5 \leq x \leq 15 \\ -11x + 163 & \text{if } 15 \leq x \leq 20 \\ 9.5x - 247 & \text{if } 20 \leq x \leq 26 \\ x - 16 & \text{if } 26 \leq x \leq 30 \end{cases} \tag{B.11}$$

Its minimization and maximization with different models yield the results in Table B.14. When the objective is minimization, the LP relaxation of incremental cost formulation finds the optimal solution with the least cpu time. On the other hand, when the objective is maximization our formulation finds the optimal solution with least cpu time.

Table B.14: CPU times for Non-Convex PWL Function Example-4

	Minimize	Maximize
Our Formulation	0.028	0.010
Convex Combination	0.106	0.098
Multiple Choice	0.029	0.034
Incremental Cost	0.035	0.033
Incremental Cost (LP Relaxation)	0.010	0.012

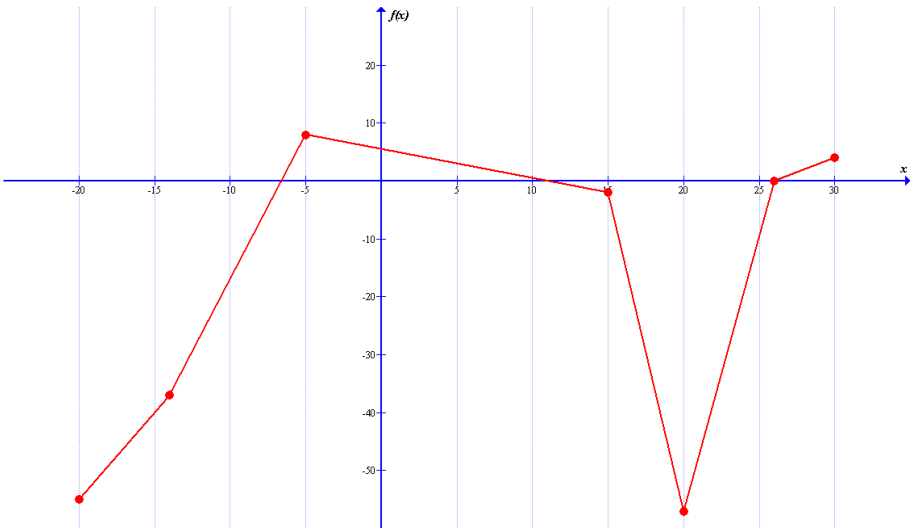

Figure B.11: Non-Convex PWL Function - 4

Function 5: The fifth non-convex PWL function is defined by equation B.12 and represented in Fig. B.12.

$$f(x) = \begin{cases} 3 & \text{if } -3 \leq x \leq -2 \\ -2x + 7 & \text{if } -2 \leq x \leq 10 \\ -13 & \text{if } 10 \leq x \leq 12 \\ x - 25 & \text{if } 12 \leq x \leq 15 \\ 2x - 40 & \text{if } 15 \leq x \leq 20 \\ 2.5x - 50 & \text{if } 20 \leq x \leq 28 \\ -x + 48 & \text{if } 28 \leq x \leq 30 \\ 18 & \text{if } 30 \leq x \leq 35 \end{cases} \tag{B.12}$$

Its minimization and maximization with different models yield the results in Table B.15. When the objective is both minimization and maximization our formulation finds the optimal solution with the least cpu time.

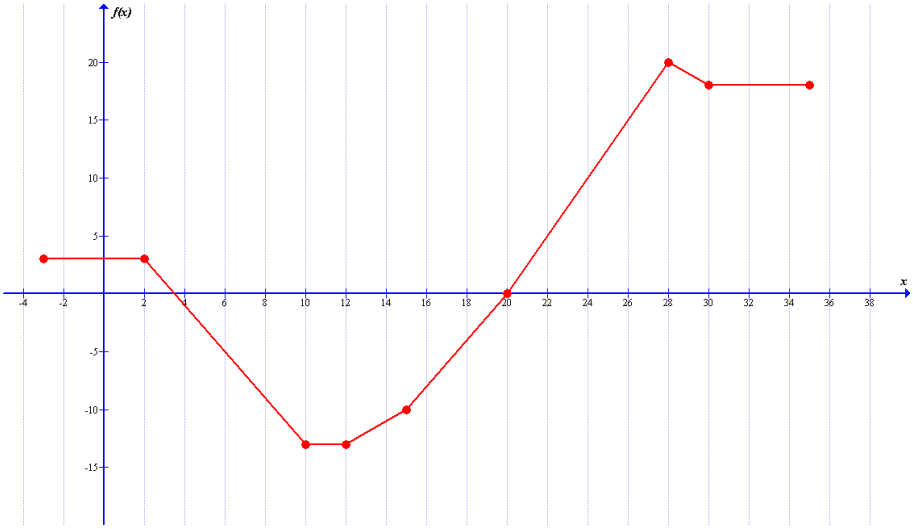

Figure B.12: Non-Convex PWL Function - 5

Table B.15: CPU times for Non-Convex PWL Function Example-5

	Minimize	Maximize
Our Formulation	0.010	0.010
Convex Combination	0.060	0.061
Multiple Choice	0.016	0.023
Incremental Cost	0.022	0.023
Incremental Cost (LP Relaxation)	0.011	0.012

B.4 Benchmark of Separable PWL Functions

Function 1: The first separable PWL function has five component functions (Eq. B.13) that are defined by equations B.14 - B.18. Its minimization and maximization with different models yield the results in Table B.16. When the objective is minimization, both the LP relaxation of incremental cost formulation and our formulation find the optimal solution with the least cpu time. On the other hand, when the objective is maximization our formulation finds the optimal solution with least cpu time.

$$f(x) = f_1(x_1) + f_2(x_2) + f_3(x_3) + f_4(x_4) + f_5(x_5) \tag{B.13}$$

$$f_1(x_1) = \begin{cases} -4x_1 + 2 & \text{if } -2 \le x_1 \le 0 \\ -3x_1 + 2 & \text{if } 0 \le x_1 \le 2 \\ x_1 - 6 & \text{if } 2 \le x_1 \le 5 \\ 2x_1 - 11 & \text{if } 5 \le x_1 \le 6 \\ 3x_1 - 17 & \text{if } 6 \le x_1 \le 9 \end{cases} \tag{B.14}$$

$$f_2(x_2) = \begin{cases} -10x_2 - 20 & \text{if } 5 \le x_2 \le 15 \\ -6x_2 - 80 & \text{if } 15 \le x_2 \le 20 \\ -200 & \text{if } 20 \le x_2 \le 30 \\ 2x_2 - 260 & \text{if } 30 \le x_2 \le 40 \\ 10x_2 - 580 & \text{if } 40 \le x_2 \le 45 \end{cases} \tag{B.15}$$

$$f_3(x_3) = \begin{cases} -6x_3 + 10 & \text{if } -10 \le x_3 \le -2 \\ -5x_3 + 12 & \text{if } -2 \le x_3 \le 1 \\ -3x_3 + 10 & \text{if } 1 \le x_3 \le 5 \\ -x_3 & \text{if } 5 \le x_3 \le 13 \\ -13 & \text{if } 13 \le x_3 \le 20 \\ 0.5x_3 - 23 & \text{if } 20 \le x_3 \le 25 \end{cases} \tag{B.16}$$

$$f_4(x_4) = \begin{cases} -0.5x_4 + 8 & \text{if } -10 \le x_4 \le -2 \\ 9 & \text{if } -2 \le x_4 \le 1 \\ x_4 + 8 & \text{if } 1 \le x_4 \le 5 \\ 3x_4 - 2 & \text{if } 5 \le x_4 \le 13 \\ 5x_4 - 28 & \text{if } 13 \le x_4 \le 20 \\ 6x_4 - 48 & \text{if } 20 \le x_4 \le 25 \end{cases} \tag{B.17}$$

$$f_5(x_5) = \begin{cases} -x_5 + 5 & \text{if } -80 \leq x_5 \leq -70 \\ -0.75x_5 + 22.5 & \text{if } -70 \leq x_5 \leq -65 \\ -0.5x_5 + 38.75 & \text{if } -65 \leq x_5 \leq -40 \\ -0.25x_5 + 48.75 & \text{if } -40 \leq x_5 \leq -5 \\ 50 & \text{if } -5 \leq x_5 \leq 0 \\ 0.25x_5 + 50 & \text{if } 0 \leq x_5 \leq 10 \\ 0.5x_5 + 47.5 & \text{if } 10 \leq x_5 \leq 60 \\ 0.75x_5 + 32.5 & \text{if } 60 \leq x_5 \leq 70 \\ x_5 + 15 & \text{if } 70 \leq x_5 \leq 80 \end{cases} \tag{B.18}$$

Table B.16: CPU times for Separable PWL Function Example-1

	Minimize	Maximize
Our Formulation	0.010	0.011
Convex Combination	0.033	0.047
Multiple Choice	0.011	0.027
Incremental Cost	0.011	0.056
Incremental Cost (LP Relaxation)	0.010	0.020

Function 2: The second separable PWL function has five component functions (Eq. B.19) that are defined by equations B.20 - B.24. Its minimization and maximization with different models yield the results in Table B.17. When the objective is minimization both our formulation and multiple choice formulation find the optimal solution with least cpu time. On the other hand, when the objective is maximization our formulation finds the optimal solution with least cpu time.

$$f(x) = f_1(x_1) + f_2(x_2) + f_3(x_3) + f_4(x_4) + f_5(x_5) \tag{B.19}$$

$$f_1(x_1) = \begin{cases} 10x_1 + 20 & \text{if } 5 \le x_1 \le 15 \\ 6x_1 + 80 & \text{if } 15 \le x_1 \le 20 \\ 200 & \text{if } 20 \le x_1 \le 30 \\ -2x_1 + 260 & \text{if } 30 \le x_1 \le 40 \\ -10x_1 + 580 & \text{if } 40 \le x_1 \le 45 \end{cases} \tag{B.20}$$

$$f_2(x_2) = \begin{cases} 4x_2 - 2 & \text{if } -2 \le x_2 \le 0 \\ 3x_2 - 2 & \text{if } 0 \le x_2 \le 2 \\ -x_2 + 6 & \text{if } 2 \le x_2 \le 5 \\ 2x_2 + 11 & \text{if } 5 \le x_2 \le 6 \\ -3x_2 + 17 & \text{if } 6 \le x_2 \le 9 \end{cases} \tag{B.21}$$

$$f_3(x_3) = \begin{cases} 6x_3 - 10 & \text{if } -10 \le x_3 \le -2 \\ 5x_3 - 12 & \text{if } -2 \le x_3 \le 1 \\ 3x_3 - 10 & \text{if } 1 \le x_3 \le 5 \\ x_3 & \text{if } 5 \le x_3 \le 13 \\ 13 & \text{if } 13 \le x_3 \le 20 \\ -0.5x_3 + 23 & \text{if } 20 \le x_3 \le 25 \end{cases} \tag{B.22}$$

$$f_4(x_4) = \begin{cases} 0.5x_4 - 8 & \text{if } -10 \le x_4 \le -2 \\ -9 & \text{if } -2 \le x_4 \le 1 \\ -x_4 - 8 & \text{if } 1 \le x_4 \le 5 \\ -3x_4 + 2 & \text{if } 5 \le x_4 \le 13 \\ -5x_4 + 28 & \text{if } 13 \le x_4 \le 20 \\ -6x_4 + 48 & \text{if } 20 \le x_4 \le 25 \end{cases} \tag{B.23}$$

$$f_5(x_5) = \begin{cases} x_5 - 5 & \text{if } -80 \le x_5 \le -70 \\ 0.75x_5 - 22.5 & \text{if } -70 \le x_5 \le -65 \\ 0.5x_5 - 38.75 & \text{if } -65 \le x_5 \le -40 \\ 0.25x_5 - 48.75 & \text{if } -40 \le x_5 \le -5 \\ -50 & \text{if } -5 \le x_5 \le 0 \\ -0.25x_5 - 50 & \text{if } 0 \le x_5 \le 10 \\ -0.5x_5 - 47.5 & \text{if } 10 \le x_5 \le 60 \\ -0.75x_5 - 32.5 & \text{if } 60 \le x_5 \le 70 \\ -x_5 - 15 & \text{if } 70 \le x_5 \le 80 \end{cases} \tag{B.24}$$

Table B.17: CPU times for Separable PWL Function Example-2

	Minimize	Maximize
Our Formulation	0.011	0.010
Convex Combination	0.030	0.065
Multiple Choice	0.011	0.011
Incremental Cost	0.066	0.021
Incremental Cost (LP Relaxation)	0.012	0.011

Function 3: The third separable PWL function has five component functions (Eq. B.25) that are defined by equations B.26 - B.30. Its minimization and maximization with different models yield the results in Table B.18. When the objective is minimization; the LP relaxation of incremental cost formulation, multiple choice formulation and our formulation find the optimal solution with the least cpu time. On the other hand, when the objective is maximization, both our formulation and multiple choice formulation find the optimal solution with least cpu time.

$$f(x) = f_1(x_1) + f_2(x_2) + f_3(x_3) + f_4(x_4) + f_5(x_5) \tag{B.25}$$

$$f_1(x_1) = \begin{cases} 4x_1 + 3 & \text{if } 0 \le x_1 \le 2 \\ -2x_1 + 15 & \text{if } 2 \le x_1 \le 3 \\ x_1 + 6 & \text{if } 3 \le x_1 \le 7 \end{cases} \tag{B.26}$$

$$f_2(x_2) = \begin{cases} 0.2x_2 + 5 & \text{if } 0 \le x_2 \le 10 \\ 0.3x_2 + 4 & \text{if } 10 \le x_2 \le 20 \\ 0.5x_2 & \text{if } 20 \le x_2 \le 30 \\ -0.4x_2 + 27 & \text{if } 30 \le x_2 \le 40 \\ 0.6x_2 - 13 & \text{if } 40 \le x_2 \le 50 \end{cases} \tag{B.27}$$

$$f_3(x_3) = \begin{cases} 1.667x_3 & \text{if } 0 \le x_3 \le 30 \\ x_3 + 20 & \text{if } 30 \le x_3 \le 60 \\ -2x_3 + 200 & \text{if } 60 \le x_3 \le 90 \\ 2.667x_3 - 220 & \text{if } 90 \le x_3 \le 120 \end{cases} \tag{B.28}$$

$$f_4(x_4) = \begin{cases} 3x_4 + 5 & \text{if } -20 \le x_4 \le -14 \\ 5x_4 + 33 & \text{if } -14 \le x_4 \le -5 \\ -0.5x_4 + 5.5 & \text{if } -5 \le x_4 \le 15 \\ -11x_4 + 163 & \text{if } 15 \le x_4 \le 20 \\ 9.5x_4 - 247 & \text{if } 20 \le x_4 \le 26 \\ x_4 - 16 & \text{if } 26 \le x_4 \le 30 \end{cases} \tag{B.29}$$

$$f_5(x_5) = \begin{cases} 3 & \text{if } -3 \le x_5 \le -2 \\ -2x_5 + 7 & \text{if } -2 \le x_5 \le 10 \\ -13 & \text{if } 10 \le x_5 \le 12 \\ x_5 - 25 & \text{if } 12 \le x_5 \le 15 \\ 2x_5 - 40 & \text{if } 15 \le x_5 \le 20 \\ 2.5x_5 - 50 & \text{if } 20 \le x_5 \le 28 \\ -x_5 + 48 & \text{if } 28 \le x_5 \le 30 \\ 18 & \text{if } 30 \le x_5 \le 35 \end{cases} \tag{B.30}$$

Table B.18: CPU times for Separable PWL Function Example-3

	Minimize	Maximize
Our Formulation	0.011	0.012
Convex Combination	0.093	0.140
Multiple Choice	0.011	0.012
Incremental Cost	0.040	0.023
Incremental Cost (LP Relaxation)	0.011	0.015

Function 4: The fourth separable PWL function has three component functions (Eq. B.31) that are defined by equations B.32 - B.34. Its minimization and maximization with different models yield the results in Table B.19. When the objective is both minimization and maximization our formulation finds the optimal solution with least cpu time.

$$f(x) = f_1(x_1) + f_2(x_2) + f_3(x_3) \tag{B.31}$$

$$f_1(x_1) = \begin{cases} -x_1 + 5 & \text{if } -80 \le x_1 \le -70 \\ -0.75x_1 + 22.5 & \text{if } -70 \le x_1 \le -65 \\ -0.5x_1 + 38.75 & \text{if } -65 \le x_1 \le -40 \\ -0.25x_1 + 48.75 & \text{if } -40 \le x_1 \le -5 \\ 50 & \text{if } -5 \le x_1 \le 0 \\ 0.25x_1 + 50 & \text{if } 0 \le x_1 \le 10 \\ 0.5x_1 + 47.5 & \text{if } 10 \le x_1 \le 60 \\ 0.75x_1 + 32.5 & \text{if } 60 \le x_1 \le 70 \\ x_1 + 15 & \text{if } 70 \le x_1 \le 80 \end{cases} \tag{B.32}$$

$$f_2(x_2) = \begin{cases} 0.5x_2 - 8 & \text{if } -10 \le x_2 \le -2 \\ -9 & \text{if } -2 \le x_2 \le 1 \\ -x_2 - 8 & \text{if } 1 \le x_2 \le 5 \\ -3x_2 + 2 & \text{if } 5 \le x_2 \le 13 \\ -5x_2 + 28 & \text{if } 13 \le x_2 \le 20 \\ -6x_2 + 48 & \text{if } 20 \le x_2 \le 25 \end{cases} \tag{B.33}$$

$$f_3(x_3) = \begin{cases} 3 & \text{if } -3 \le x_3 \le -2 \\ -2x_3 + 7 & \text{if } -2 \le x_3 \le 10 \\ -13 & \text{if } 10 \le x_3 \le 12 \\ x_3 - 25 & \text{if } 12 \le x_3 \le 15 \\ 2x_3 - 40 & \text{if } 15 \le x_3 \le 20 \\ 2.5x_3 - 50 & \text{if } 20 \le x_3 \le 28 \\ -x_3 + 48 & \text{if } 28 \le x_3 \le 30 \\ 18 & \text{if } 30 \le x_3 \le 35 \end{cases} \tag{B.34}$$

Function 5: The fifth separable PWL function has five component functions (Eq.

Table B.19: CPU times for Separable PWL Function Example-4

	Minimize	Maximize
Our Formulation	0.010	0.010
Convex Combination	0.032	0.074
Multiple Choice	0.011	0.011
Incremental Cost	0.012	0.057
Incremental Cost (LP Relaxation)	0.011	0.030

B.35) that are defined by equations B.36 - B.40. Its minimization and maximization with different models yield the results in Table B.20. When the objective is both minimization and maximization our formulation finds the optimal solution with least cpu time.

$$f(x) = f_1(x_1) + f_2(x_2) + f_3(x_3) + f_4(x_4) + f_5(x_5) \tag{B.35}$$

$$f_1(x_1) = \begin{cases} -x_1 + 5 & \text{if } -80 \leq x_1 \leq -70 \\ -0.75x_1 + 22.5 & \text{if } -70 \leq x_1 \leq -65 \\ -0.5x_1 + 38.75 & \text{if } -65 \leq x_1 \leq -40 \\ -0.25x_1 + 48.75 & \text{if } -40 \leq x_1 \leq -5 \\ 50 & \text{if } -5 \leq x_1 \leq 0 \\ 0.25x_1 + 50 & \text{if } 0 \leq x_1 \leq 10 \\ 0.5x_1 + 47.5 & \text{if } 10 \leq x_1 \leq 60 \\ 0.75x_1 + 32.5 & \text{if } 60 \leq x_1 \leq 70 \\ x_1 + 15 & \text{if } 70 \leq x_1 \leq 80 \end{cases} \tag{B.36}$$

$$f_2(x_2) = \begin{cases} 0.5x_2 - 8 & \text{if } -10 \le x_2 \le -2 \\ -9 & \text{if } -2 \le x_2 \le 1 \\ -x_2 - 8 & \text{if } 1 \le x_2 \le 5 \\ -3x_2 + 2 & \text{if } 5 \le x_2 \le 13 \\ -5x_2 + 28 & \text{if } 13 \le x_2 \le 20 \\ -6x_2 + 48 & \text{if } 20 \le x_2 \le 25 \end{cases} \tag{B.37}$$

$$f_3(x_3) = \begin{cases} 3 & \text{if } -3 \le x_3 \le -2 \\ -2x_3 + 7 & \text{if } -2 \le x_3 \le 10 \\ -13 & \text{if } 10 \le x_3 \le 12 \\ x_3 - 25 & \text{if } 12 \le x_3 \le 15 \\ 2x_3 - 40 & \text{if } 15 \le x_3 \le 20 \\ 2.5x_3 - 50 & \text{if } 20 \le x_3 \le 28 \\ -x_3 + 48 & \text{if } 28 \le x_3 \le 30 \\ 18 & \text{if } 30 \le x_3 \le 35 \end{cases} \tag{B.38}$$

$$f_4(x_4) = \begin{cases} 1.667x_4 & \text{if } 0 \le x_4 \le 30 \\ x_4 + 20 & \text{if } 30 \le x_4 \le 60 \\ -2x_4 + 200 & \text{if } 60 \le x_4 \le 90 \\ 2.667x_4 - 220 & \text{if } 90 \le x_4 \le 120 \end{cases} \tag{B.39}$$

$$f_5(x_5) = \begin{cases} 3x_5 + 5 & \text{if } -20 \le x_5 \le -14 \\ 5x_5 + 33 & \text{if } -14 \le x_5 \le -5 \\ -0.5x_5 + 5.5 & \text{if } -5 \le x_5 \le 15 \\ -11x_5 + 163 & \text{if } 15 \le x_5 \le 20 \\ 9.5x_5 - 247 & \text{if } 20 \le x_5 \le 26 \\ x_5 - 16 & \text{if } 26 \le x_5 \le 30 \end{cases} \tag{B.40}$$

Table B.20: CPU times for Separable PWL Function Example-5

	Minimize	Maximize
Our Formulation	0.010	0.010
Convex Combination	0.026	0.037
Multiple Choice	0.012	0.011
Incremental Cost	0.011	0.021
Incremental Cost (LP Relaxation)	0.012	0.020

Appendix C

GAMS MODEL FOR BENCHMARK PROBLEMS

```
$Non-Convex PWL Function-1 Benchmark Code


Sets j /1,2,3,4/;


Parameters


x(j)
/ 1 0
  2 2
  3 3
  4 7/


a(j)
/1 4
 2 -2
 3 1/


b(j)
/1 3
 2 15
 3 16/


f(j)
```

```
/1 3
 2 11
 3 9
 4 13
/;

Variables

y1(j)
fx1
xx1
xx2
lamda2(j)
fx2
z3(j)
fx3
xx3
fx4
xx4
q4(j)
fx5
xx5
q5(j)
y5(j);

Positive variables

y1(j)
lamda2(j)
```

```
q4(j)
q5(j)
y5(j);

Binary variables

sigma2(j)
y3(j)
y4(j);

Equations

bound1(j)
domain1
key1(j)
functionvalue1
boundlamda2(j)
domainxx2
sumoflamdas2
lamda12
lamdat2(j)
lamda22(j)
sumofsigma2
functionvalue2
functionfx3
domainx3
boundlowerz3(j)
boundupperz3(j)
sumofy3
```

```
functionfx4
domainx4
yjgreaterthan4(j)
yjlessthan4(j)
functionfx5
domainx5
yjgreaterthan5(j)
yjlessthan5(j)
yjlessthan50(j);

bound1(j)$(ord(j) lt card(j)).. y1(j)=l=(x(j+1)-x(j));

domain1.. sum(j$(ord(j) lt card(j)),y1(j))+x('1')=e=xx1;

key1(j)$(ord(j) lt card(j)-1)..
y1(j+1)=l=(x(j+2)-x(j+1))*y1(j)/(x(j+1)-x(j));

functionvalue1.. fx1=e=a('1')*x('1')+b('1')+sum(j$(ord(j) lt
card(j)), a(j)*y1(j));

boundlamda2(j).. lamda2(j)=l=1;

domainxx2.. xx2=e=sum(j,lamda2(j)*x(j));

sumoflamdas2.. sum(j,lamda2(j))=e=1;

lamda12.. lamda2('1')=l=sigma2('1');

lamdat2(j)$(ord(j) eq card(j)).. lamda2(j)=l=sigma2(j-1);
```

```
lamda22(j)$(ord(j) gt 1 and ord(j) lt card(j))..
lamda2(j)=L=sigma2(j-1)+sigma2(j);

sumofsigma2.. sum(j$(ord(j) lt card(j)),sigma2(j))=E=1;

functionvalue2.. fx2=E=sum(j,f(j)*lamda2(j));

functionfx3.. fx3=E=sum(j$(ord(j) lt
card(j)),a(j)*z3(j)+b(j)*y3(j));

domainx3.. xx3=E=sum(j$(ord(j) lt card(j)),z3(j));

boundlowerz3(j)$(ord(j) lt card(j)).. x(j)*y3(j)=L=z3(j);

boundupperz3(j)$(ord(j) lt card(j)).. z3(j)=L=x(j+1)*y3(j);

sumofy3.. sum(j$(ord(j) lt card(j)),y3(j))=E=1;

functionfx4.. fx4=E=b('1')+a('1')*x('1')+sum(j$(ord(j) lt
card(j)),a(j)*q4(j));

domainx4.. xx4=E=x('1')+sum(j$(ord(j) lt card(j)),q4(j));

yjgreaterthan4(j)$(ord(j) lt card(j))..
y4(j)=g=q4(j)/(x(j+1)-x(j));

yjlessthan4(j)$(ord(j) lt card(j)-1)..
y4(j+1)=L=q4(j)/(x(j+1)-x(j));
```

```
functionfx5.. fx5=E=b('1')+a('1')*x('1')+sum(j$(ord(j) lt
card(j)),a(j)*q5(j));

domainx5.. xx5=E=x('1')+sum(j$(ord(j) lt card(j)),q5(j));

yjgreaterthan5(j)$(ord(j) lt card(j))..
y5(j)=g=q5(j)/(x(j+1)-x(j));

yjlessthan5(j)$(ord(j) lt card(j)-1)..
y5(j+1)=L=q5(j)/(x(j+1)-x(j));

yjlessthan50(j)$(ord(j) lt card(j)).. y5(j)=L=1;

Model ourmodelmax /bound1, domain1, key1, functionvalue1/ ;

Solve ourmodelmax using lp maximizing fx1 ;

Model ourmodelmin /bound1, domain1, key1, functionvalue1/ ;

Solve ourmodelmin using lp minimizing fx1 ;

Model convexcombmax / boundlamda2, domainxx2 , sumoflamdas2,
lamda12, lamdat2, lamda22, sumofsigma2, functionvalue2/ ;

Solve convexcombmax using mip maximizing fx2;

Model convexcombmin / boundlamda2, domainxx2 , sumoflamdas2,
lamda12, lamdat2, lamda22, sumofsigma2, functionvalue2/ ;
```

```
Solve convexcombmin using mip minimizing fx2;

Model multiplechoicemax
/functionfx3,domainx3,boundlowerz3,boundupperz3,sumofy3/ ;

Solve multiplechoicemax using mip maximizing fx3;

Model multiplechoicemin
/functionfx3,domainx3,boundlowerz3,boundupperz3,sumofy3/ ;

Solve multiplechoicemin using mip minimizing fx3;

Model incrcostmax
/functionfx4,domainx4,yjgreaterthan4,yjlessthan4/ ;

Solve incrcostmax using mip maximizing fx4;

Model incrcostmin
/functionfx4,domainx4,yjgreaterthan4,yjlessthan4/ ;

Solve incrcostmin using mip minimizing fx4;

Model LPincrcostmax /
functionfx5,domainx5,yjgreaterthan5,yjlessthan5,yjlessthan50/ ;

Solve LPincrcostmax using lp maximizing fx5;

Model LPincrcostmin /
```

```
functionfx5,domainx5,yjgreaterthan5,yjlessthan5,yjlessthan50/ ;

Solve LPincrcostmin using lp minimizing fx5;

display ourmodelmin.resusd;

display ourmodelmax.resusd;

display convexcombmin.resusd;

display convexcombmax.resusd;

display multiplechoicemin.resusd;

display multiplechoicemax.resusd;

display incrcostmin.resusd;

display incrcostmax.resusd;

display LPincrcostmin.resusd;

display LPincrcostmax.resusd;
```

VITA

ÖZGE ARSLAN was born on April 3, 1986. She was graduated from Cagaloglu Anatolian High School, Istanbul in 2003 and received her B.Sc. degree in Industrial Engineering from Bilkent University, Ankara, in 2008. Following that, she joined the M.S. Program in Industrial Engineering at Koç University, Istanbul. From September 2008 to June 2010, she has worked as a teaching and research assistant. Currently, she is working as strategic analyst.