INVENTORY PINCH ALGORITHMS FOR GASOLINE BLEND PLANNING

INVENTORY PINCH ALGORITHMS FOR GASOLINE BLEND PLANNING

By PEDRO ALEJANDRO CASTILLO CASTILLO, B. CHEM. ENG.

A Thesis Submitted to the School of Graduate Studies in Partial Fulfilment of the Requirements for the Degree Master of Applied Sciences in Chemical Engineering

MASTER OF APPLIED SCIENCE (2013) McMaster University

(Chemical Engineering Department) Hamilton, Ontario

TITLE: Inventory Pinch Algorithms for Gasoline Blend Planning

AUTHOR: Pedro Alejandro Castillo Castillo, B. Chem. Eng. (Universidad Autonoma de San Luis Potosi, San Luis Potosi, Mexico)

SUPERVISOR: Dr. Vladimir Mahalec

PAGES: xi, 81

Abstract

Current gasoline blend planning practice is to optimize blend plans via discrete-time multi-period NLP or MINLP models and schedule blends via interactive simulation. Solutions of multi-period models using discrete-time representation typically have different blend recipes for each time period. In this work, the concept of an inventory pinch point is introduced and used it to construct a new decomposition of the multi-period MINLP problems: at the top level nonlinear blending problems for periods delimited by the inventory pinch points are solved; at the lower level a fine grid multi-period MILP model that uses optimal recipes from the top level is solved in order to determine how much to blend of each product in each fine grid period, subject to minimum threshold blend size.

Two algorithms at the top level are examined: a) multi-period nonlinear model (MPIP) and b) single-period non-linear model (SPIP). Both algorithms optimize multi-grade blend recipes for each period delimited by the inventory pinch points and then use a fine-grid multi-period fixed-recipe MILP to compute blend volumes profile. If MILP is infeasible, corresponding period between the pinch points is subdivided and recipes are re-optimized. Case studies show that the MPIP algorithm produces solutions that have the same optimal value of the objective function as fine-grid calendar based MINLP. The SPIP algorithm computes solutions that are most often within 0.01% of the solutions by multi-period MINLP. Both algorithms require substantially less computational effort (typically an order of magnitude or even less) than the fine grid calendar based MINLP. Reduced number of blend recipes makes it easier for blend scheduler to create a schedule by interactive simulation.

It is expected that the reduced number of diverse blend recipes will enable rapid solutions of the scheduling within the constraints computed by MPIP or SPIP algorithms.

Acknowledgements

I would like to express my sincerely gratitude to my Supervisor, Dr. Vladimir Mahalec, whose guidance and dedication helped me to complete this dissertation.

I would like to thank the faculty members and staff of the Chemical Engineering Department, as well as my colleagues, who helped me to install and use the computer software required to develop this work.

Finally, I would like to dedicate this work to my parents and all my family, who have given me their confidence, love, and strength to be able to fulfill one of my personal and professional goals.

Table of Contents

Chapter 1. Introduction 1
1.1 Brief Review of the Oil Refining Process 1
1.2 Gasoline Blending Process 2
1.3 Prior Work on Gasoline Blend Planning 4
1.4 Thesis Work 7
Chapter 2. Problem Statement 8
Chapter 3. Inventory Pinch Concept 10
Chapter 4. Inventory Pinch Algorithm 15
4.1 Mathematical Model for the MINLP formulation 16
4.2 Mathematical Model for the Top Level of the Inventory Pinch Algorithm 19
4.2.1 Multi-Period Formulation 19
4.2.2 Single-Period Formulation 21
4.2.3 Initial product inventories off-spec: Tank heel quality correction 22
4.3 Mathematical Model for the Lower Level of the Inventory Pinch Algorithm 24
4.4 Multi-Period Inventory Pinch Algorithm 26
4.5 Single-Period Inventory Pinch Algorithm 28
Chapter 5. Numerical Results 31
5.1 Example #1: Case study 27 (RVP blends nonlinearly) 39
5.1.1 Multi-Period Inventory Pinch Algorithm: Case Study 27 39
5.1.2 Single-Period Inventory Pinch Algorithm: Case Study 27 44
5.2 Example #2: Case study 30 (2 blenders) 51
5.3 Discussion 54
Chapter 6. Conclusions and Future Work 61
Bibliography 63
APPENDIX A: GAMS models 65

List of Figures

Figure 1.1: Graphic scheme of a standard refinery system. Re-adapted from (Mendez et al., 2006) 2
Figure 1.2: Representative gasoline blending system 4
Figure 3.1: Problem with one pinch point 11
Figure 3.2: Graphic representation of the inventory pinch algorithm 12
Figure 3.3: Time grids for the two levels of the algorithm 12
Figure 3.4: One "local" inventory pinch and one true inventory pinch 13
Figure 3.5: Example with two true inventory pinch points 13
Figure 3.6: Inventory infeasibilities elimination by inventory pinch algorithm. a) Infeasible blend recipes, b) feasible blend recipes 14
Figure 4.1: Proposed decomposition of gasoline blending 16
Figure 4.2: Inventory Pinch Multi-Period Algorithm 28
Figure 4.3: Inventory Pinch Single-Period Algorithm 30
Figure 5.1: Cumulative demand profiles of some case studies. The vertical dashed lines indicate the pinch points. 38
Figure 5.2: Demand profile - Case study 27 39
Figure 5.3: Inventory pinch points - Case study 27 40
Figure 5.4: Product inventory profile – Case study 27, multi-period algorithm, *it*=1 41
Figure 5.5: Product inventory profile – Case study 27, multi-period algorithm, *it*=2 42
Figure 5.6: Products inventory profile – Case study 27, single-period algorithm, *it*=1 ... 45
Figure 5.7: Products inventory profile – Case study 27, single-period algorithm, *it*=2 ... 46
Figure 5.8: Products inventory profile – Case study 27, single-period algorithm, *it*=3 ... 47
Figure 5.9: Products inventory profile – Case study 27, single-period algorithm, *it*=5 ... 49
Figure 5.10: Demand profile - Case study 30 51
Figure 5.11: Inventory pinch points - Case study 30 52
Figure 5.12: Products inventory profile – Case study 30, single-period algorithm, *it*=1 . 53

List of Tables

Table 5.1: Components data (properties, cost, supply rates and inventory limits) 32
Table 5.2: Supply rate of components along planning horizon, cases 10 – 18, 27, and 28 33
Table 5.3: Minimum and maximum quality specifications of the products 33
Table 5.4: Products' initial quality, initial inventory, and inventory bounds 34
Table 5.5: Demand profiles (x10^3BBL), case studies 1 – 12 35
Table 5.6: Demand profiles (x10^3BBL), case studies 13 – 24 36
Table 5.7: Demand profiles (x10^3BBL), case studies 25 – 30 37
Table 5.8: Cost coefficients profile for the product inventory slack variables, lower level MILP model 37
Table 5.9: Volumes to blend (10^3BBL), case study 27, multi-period algorithm, *it* = 1 ... 40
Table 5.10: Volumes to blend (10^3BBL), case study 27, multi-period algorithm, *it* = 2.. 42
Table 5.11: Blend plan for case study 27, multi-period algorithm (volume in 10^3BBL) . 43
Table 5.12: Blend recipes computed for case study 27, multi-period algorithm 43
Table 5.13: Volumes to blend (10^3BBL), case study 27, single-period algorithm, *it* = 3 46
Table 5.14: Volumes to blend (10^3BBL), case study 27, single-period algorithm, *it* = 4 47
Table 5.15: Volumes to blend (10^3BBL), case study 27, single-period algorithm, *it* = 5 48
Table 5.16: Blend plan for case study 27, single-period algorithm (volume in 10^3BBL) 49
Table 5.17: Blend recipes for case study 27, single-period algorithm (volume in 10^3BBL) 50
Table 5.18: Blend plan for case study 30, single-period algorithm (volume in 10^3BBL) 53
Table 5.19: Blend recipes for case study 30, single-period algorithm (volume in 10^3BBL) 54
Table 5.20: Objective function values and solution times for case studies 1 to 18 56
Table 5.21: Objective function values and solution times for case studies 19 to 30 57
Table 5.22: Objective function values and solution times for case studies 22, 26, 27, and 28..... 58
Table 5.23: Number of blend recipes (case studies 1 to 18 have 14 time *l*-periods, 1 blender, RVP blends nonlinearly, and the initial product inventory is on-spec) 59
Table 5.24: Number of blend recipes (case studies 19 to 30 have 14 time *l*-periods and 1 blender) 60
Table 5.25: Model size comparison (RVP blends nonlinearly) 61

List of all Abbreviations and Symbols

Abbreviations

ALK	Alkylate
ARO	Aromatics content
BBL	Barrel (unit of volume)
BEN	Benzene content
BIs	Blend indices
BUT	Butane
CATP	Cumulative average total production
CTD	Cumulative total demand
CTP	Cumulative total production
EPA	Environmental Protection Agency
GAMS	General Algebraic Modeling System
HCL	Hydrochloric acid
HCN	Hydrocarbons
LCN	Light cracked naphtha
LNP	Light naphtha
LP	Linear programming
MILP	Mixed-integer linear programming
MINLP	Mixed-integer nonlinear programming
MPIP	Multi-period inventory pinch
MON	Motor octane number
NLP	Nonlinear programming
OLF	Olefins content
RFT	Reformate
RON	Research octane number
RVP	Reid vapor pressure
SLP	Successive linear programming
SPG	Specific gravity
SPIP	Single-period inventory pinch
SQP	Sequential quadratic programming
SUL	Sulfur content

Sets

bl	Blender {A, B}
g	Gasoline grades {U87, U91, U93}
i	Blend components {ALK, BUT, HCL, HCN, LCN, LNP, RFT}
k	t-periods at the top level $\{1,\ldots, K_T\}$
n	*l*-periods at the lower level $\{1,\ldots, N\}$

Subscripts

0	Initial product quality correction period (tank heel correction period)
1	Aggregate period of the planning horizon not including the period to correct the tank heel
B	Blender
C	Blend component
K	Top level
P	Product

Parameters

$Cost(i)$	Cost of blend component i
$D_P(g,n)$	Demand of product g in l-period n
$D_{P,K}(g)$	Demand of product g in t-period K
$F_B^{max}(bl)$	Maximum blending capacity of blender bl
H	Length of the planning horizon
$np(bl)$	Number of products that can be produced in a l-period in blender bl
$Penalty_C(i)$	Penalty for the inventory slack variables of blend components i
$Penalty_{P,K}(g)$	Penalty for the inventory slack variables of product g
$Penalty_P(g,n)$	Penalty for the inventory slack variables of product g in l-period n
$Q_C(i,s)$	Quality s of blend component i
$Q_P^{max}(g,s)$	Maximum requirement of quality s in grade g
$Q_P^{min}(g,s)$	Minimum requirement of quality s in grade g
$t(n)$	Duration of l-period n
t_K	Duration of t-period K
$V_B^{lost}(bl)$	Volume lost due to switching from one grade to another in blender bl
$V_B^{max}(g)$	Maximum volume allowed to blend of grade g during any blender run
$V_B^{min}(g)$	Minimum volume allowed to blend of grade g during any blender run
$V_{C,K}^{in}(i)$	Volume supply of blend component i in t-period K
$V_C^{in}(i,n)$	Volume supply of blend component i in l-period n
$V_C^{max}(i)$	Maximum volume of tank with blend component i
$V_C^{min}(i)$	Minimum volume of tank with blend component i
$V_P^{max}(g)$	Maximum volume of tank with product g
$V_P^{min}(g)$	Minimum volume of tank with product g

Variables

$A(g,n,bl)$	Binary variable to determine if product g is blended in l-period n in blender bl
$Q_P(g,s,n)$	Quality s of product g at l-period n
$Q_P^{close}(g,s,n)$	Quality s of product g at the end of l-period n
$Q_P^{open}(g,s,n)$	Quality s of product g at the beginning of l-period n
$Q_{P,0}^{close}(g,s)$	Quality s of product g at the end of t-period K_0
$Q_{P,0}^{open}(g,s)$	Quality s of product g at the beginning of t-period K_0
$S_C^+(i,n)$	Positive inventory slack variable of blend component i at l-period n
$S_C^-(i,n)$	Negative inventory slack variable of blend component i at l-period n

$S_{C,K}^{+}(i)$	Positive inventory slack variable of blend component *i* at *t*-period *K*
$S_{C,K}^{-}(i)$	Negative inventory slack variable of blend component *i* at *t*-period *K*
$S_{P}^{+}(g,n)$	Positive inventory slack variable of product *g* at *l*-period *n*
$S_{P}^{-}(g,n)$	Negative inventory slack variable of product *g* at *l*-period *n*
$S_{P,0}^{+}(g)$	Positive inventory slack variable of product *g* at *t*-period K_0
$S_{P,0}^{-}(g)$	Negative inventory slack variable of product *g* at *t*-period K_0
$S_{P,K}^{+}(g)$	Positive inventory slack variable of product *g* at *t*-period *K*
$S_{P,K}^{-}(g)$	Negative inventory slack variable of product *g* at *t*-period *K*
V(0)	Sum of the initial inventories of all products
$V_B(g,n,bl)$	Volume of product *g* blended in *l*-period *n* in blender *bl*
$V_{B,0}(g)$	Volume of product *g* blended in *t*-period K_0
$V_{B,K}(g)$	Volume of product *g* blended in *t*-period *K*
$V_C(i,g,n,bl)$	Volume of blend component i into product *g* in *l*-period *n* in blender *bl*
$V_C^{close}(i,n)$	Closing volume of tank with blend component *i* in *l*-period *n*
$V_C^{open}(i,n)$	Opening volume of tank with blend component *i* in *l*-period *n*
$V_{C,0}(i,g)$	Volume of blend component i into product *g* in *t*-period K_0
$V_{C,K}(i,g)$	Volume of blend component i into product *g* in *t*-period *K*
$V_{C,K}^{close}(i)$	Closing volume of tank with blend component *i* in *t*-period *K*
$V_{C,K}^{open}(i)$	Opening volume of tank with blend component *i* in *t*-period *K*
$V_P^{close}(g,n)$	Closing volume of tank with product *g* in *l*-period *n*
$V_P^{open}(g,n)$	Opening volume of tank with product *g* in *l*-period *n*
$V_{P,0}^{close}(g)$	Closing volume of tank with product *g* in *t*-period K_0
$V_{P,0}^{open}(g)$	Opening volume of tank with product *g* in *t*-period K_0
$V_{P,K}^{close}(g)$	Closing volume of tank with product *g* in *t*-period *K*
$V_{P,K}^{open}(g)$	Opening volume of tank with product *g* in *t*-period *K*
$x(i,g,n,bl)$	Volume fraction of blend component *i* in product *g* at *l*-period *n* in blender *bl*
$x_K(i,g)$	Volume fraction of blend component *i* in product *g* at *t*-period *K*
XC	Extra cost for volume of components used to correct the heel off-spec

Declaration of Academic Achievement

I, Pedro Alejandro Castillo Castillo, declare that my contributions to this research work are the following: i) I provided ideas to develop the algorithms introduced in this work, ii) I developed and wrote the mathematical models used to solve the optimization problems on a computer modelling language (GAMS), and iii) I solved the case studies presented in this work.

In addition, I declare that Dr. Vladimir Mahalec provided the main idea of the inventory pinch concept and the algorithms based on it, and he provided ideas and guidance to develop such algorithms and their respective mathematical models.

Sincerely,

Pedro Alejandro Castillo Castillo

Chapter 1. Introduction

Gasoline blending is a very important part of refinery operations. Due to a large volume of the product, it is very important to blend gasoline at the lowest possible cost, while satisfying quality constraints. If gasoline blends are not made at the smallest possible deviations from the constraints, the refinery profit is impacted very significantly. Research presented in this thesis focuses on new algorithms for gasoline blend planning, with the intent to find optimal blend recipes that change much less often than it is the current practice. In addition, the goal is to simplify the structure of the models being solved, thereby leading to a large reduction in execution times. Successful reduction of computational effort for a given model size then enables use of more complex blend models (e.g. reformulated gasoline model).

This thesis will discuss in the first chapter the gasoline blending system and the prior work on the planning optimization approaches and techniques used for this process. Then, in the second chapter, the problem formulation is stated. In chapter 3, the inventory pinch point concept is introduced and the mathematical models and algorithms developed are presented in chapter 4. Chapter 5 contains the numerical results obtained from which several points are discussed. Finally, chapter 6 summarizes the conclusions drawn from the results. The GAMS code used to solve the case studies is shown in Appendix A.

1.1 Brief Review of the Oil Refining Process

Since its beginning during the 1850's, the oil refining industry has played a fundamental part in the economy of a country that has oil resources, as well as a major role in the chemical and energy production areas. The oil refining process consists in transforming the crude oil into more usable products. Figure 1.1 shows a standard refinery system divided in its three major sections: crude oil unloading and blending, production units, and product blending and shipping. In the first section, crude oils arriving to the refinery are unloaded to storage tanks and then blended accordingly to yield the distillation targets. The second section is constituted by the production units such as the distillation columns, reformer, catalytic cracker, hydrocracker, etc. The third section consists of the blending units to produce the final liquid products and shipping operations.

Crude oil refineries produce various liquid fuels by blending intermediate product streams in a manner which minimizes use of more valuable components, while meeting product specifications. Product specification are either "greater than or equal" or "less than or equal" constraints for various product properties (also referred as qualities, e.g. octane number, olefins content, sulphur content, specific gravity, etc.). Usually, there is one blender for each type of liquid products, i.e. one blender for gasoline, one blender for

diesel, etc. Refineries operate to meet the contracted product demand. Hence, an optimal refinery operation is the one that meets contracted products liftings while minimizing operating costs and inventory carrying costs. Total refinery production is planned on a long term basis (e.g. month-to-month) while the refinery operation over short time horizons (e.g. two or four weeks) is usually separated in two major tasks: (i) scheduling of process units, and (ii) planning and scheduling of liquid products blending. Refinery production planning linear programming (LP) models are being replaced by the nonlinear programming (NLP) models due to new regulations, more expensive raw materials, higher utility costs, and other factors (Kelly, 2004). Most of the commercial software use successive linear programming (SLP) techniques to solve NLP models (Mendez, Grossman, Harjunkoski, & Kabore, 2006), while AspenTech's nonlinear PIMS uses sequential quadratic programming (SQP) methods. The refinery planning models are formulated as multi-period models (Neiro & Pinto, 2005) where the planning horizon is divided into several time periods of specified duration (i.e. discrete time representation) or several time slots with variable duration that will be computed (i.e. continuous time representation). Constraints are satisfied at the time period boundaries and serve as the basis for scheduling operations of the process units. Gasoline comprises the largest volume of liquid products and minimization of the gasoline blending costs has a major impact on refinery profitability as it represents 60 to 70% of the refinery revenues (DeWitt, Lasdon, Waren, Brenner, & Melhem, 1989).

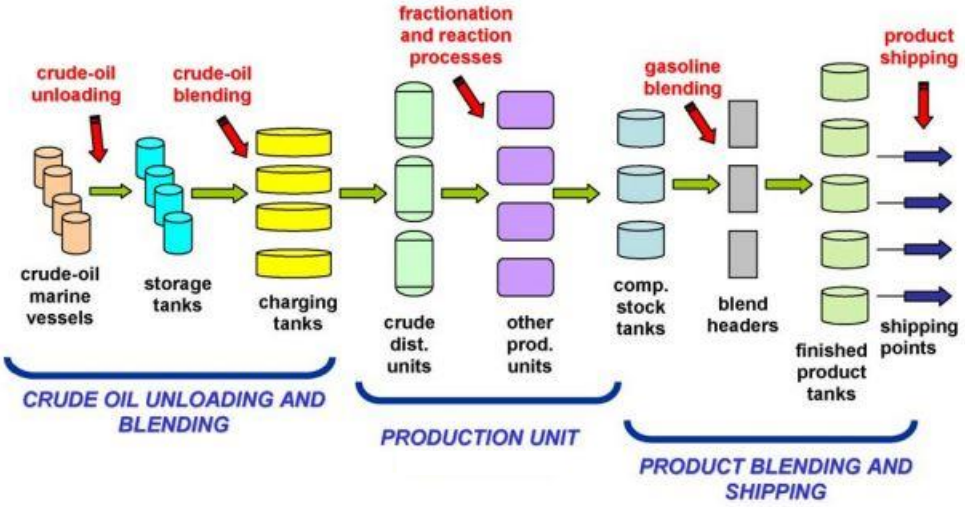


Figure 1.1: Graphic scheme of a standard refinery system. Re-adapted from (Mendez et al., 2006)

1.2 Gasoline Blending Process

Gasoline blending process is within the third section of the oil refinery process (i.e. product blending and shipping). Figure 1.2 shows a representative gasoline blending

system. The blend components arrive from upstream process units and are stored in their respective tanks. In the blend headers ("blenders") the blend components are mixed in a predetermined ratio (i.e. the "blend recipe") to meet the specifications for the gasoline grade (e.g. regular, medium, premium, etc.) that is being blended. There can be one or more blenders operating in parallel. A gasoline blender switches from blending one grade to another grade. During the switching the analytical instruments (e.g. octane engine) need to be recalibrated to ensure accurate online measurement of the blend properties. Preparation time results in a lost blend capacity.

Properties of the blended gasoline are nonlinear functions of the properties of the components that are blended to make a specific grade. Nonlinearities in the blend planning model arise from (i) nonlinear blending properties which introduce non-convex terms (Misener & Floudas, 2009), (ii) including unknown future quality and future volume (tank heel) for each grade of gasoline in the multi-period model, and (iii) other attributes of the pooling problem (Main, 1993). During the 60's and 70's the practitioners developed transformations of individual properties into so called blending indices (BIs), which blend linearly component BIs into gasoline BIs. If blending index transformations are applied also to product constraints, then the equations governing the blend properties become linear, if the properties of every blend component are known. However, recent changes in gasoline blend specification (Environmental Protection Agency (EPA), Title 40 Code of Federal Regulations Part 80.45: Complex Emissions Model [40CFR80.45, 2007]) introduce highly nonlinear constraints with respect to product properties. Misener, Gounaris, and Floudas (2010) presented a single-period MINLP model that includes binary variables to represent the quality breakpoints defined by the EPA model. To solve this highly nonlinear mixed-integer model to global optimality, they introduced a specialized algorithm. One of their case studies consisted of 1104 continuous variables, 150 binary variables, and 640 nonlinear terms; it was solved in 5274 CPU seconds with an optimality gap of 0.5%.

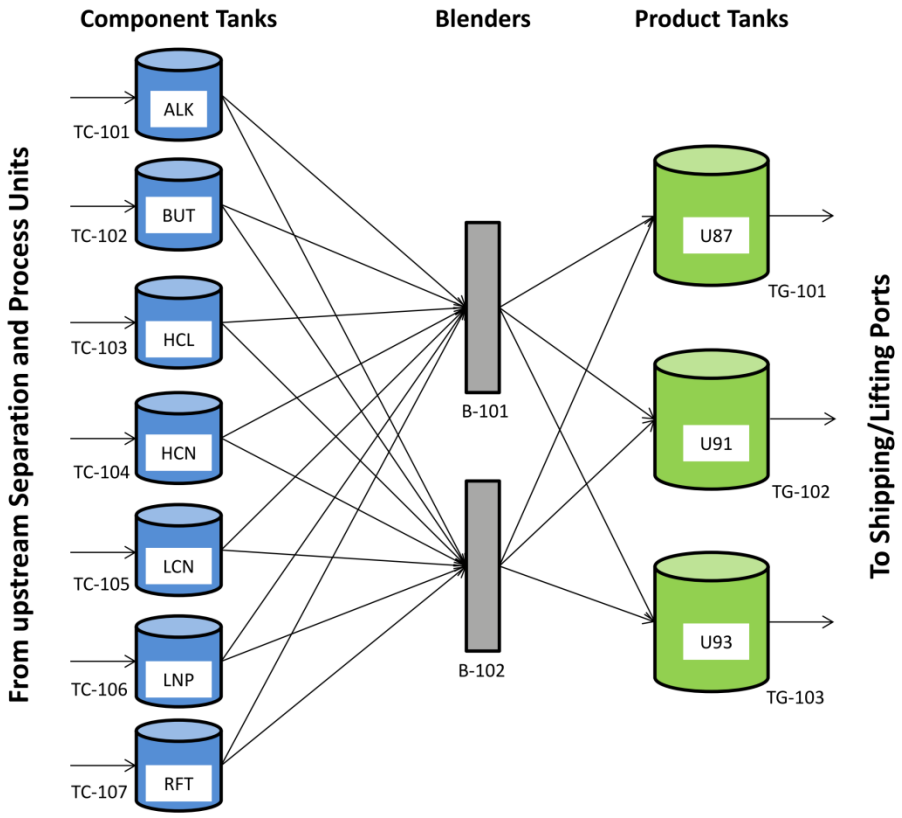


Figure 1.2: Representative gasoline blending system

1.3 Prior Work on Gasoline Blend Planning

Earliest approach to gasoline blending decomposes the problem in two steps: blend planning (also known as off-line blending problem or recipe optimization problem) and blend scheduling. This is still the prevailing practice, as witnessed by the commercial success of blend planning tools (e.g. Honeywell BLEND, Aspen PIMS-MBO) and scheduling tools (e.g. Production Scheduler from Honeywell, ORION from AspenTech). In the planning step, a discrete-time multi-period MINLP model is used to optimize the total cost by computing the volume of each blend and its corresponding blend recipe (e.g. Honeywell BLEND). The term "blend recipe" refers to the volume fractions of the components used to blend one unit of a specific grade of gasoline during a time period. Blender operation is then scheduled via interactive simulation (e.g. ORION) within the constraints imposed by the solution of the multi-period model. The planning model includes constraints w.r.t. minimum size of the blend, maximum blend rate, and component and product inventories. Cost of switching from blending one grade to another is included in the total cost, while the duration of the switch can be included as a reduction of the available blender capacity.

An advantage of decomposing the problem into planning and scheduling is a relative simplicity of solving the planning model. Disadvantages of the approach are:

(i) Blend plan is guaranteed to be feasible only at the period boundaries; however, the operation may be infeasible at some intermediate point within a period.

(ii) Blend recipes vary from one period to another. In other words, if the blend planning horizon is 14 periods, typically there will be 14 different blend recipes for each grade of gasoline.

Variations in blend recipes require more decisions by the scheduler, making it more difficult to schedule operation with minimum number of switches. Hence, the current practice is to keep the blend recipes relatively constant, since the schedulers know that the optimal blending of the same gasoline grade by using the same recipe should be possible for extended lengths of time. In order to avoid meandering of the blend recipe, one can minimize deviations of blend recipes in every period from some preferred or average blend recipe. This strategy has been implemented by adding penalty terms to the objective function (e.g. Mendez et al., 2006). However, such formulation means that the optimum value of the objective function is a combination of the costs and the penalties for deviations from the average blend recipes, i.e. the solution might not end up being the same real economic optimum of the unconstrained recipe problem. Another option to reduce recipe variation is to use preferred blend recipes (e.g. Jia & Ierapetritou, 2003); in this case, the possibility to obtain the best solution depends on the selection of this set of preferred fixed recipes.

Since blend plan computed from a multi-period model is feasible at the period boundaries and possibly infeasible within some time periods, Thakral and Mahalec (2012) introduced a composite algorithm which computes blend recipes via multi-period MINLP and then uses a genetic algorithm to minimize the number of switches followed by agent-based simulation to detect infeasibilities, if they exist. If an infeasibility is encountered, the corresponding period is subdivided, the blend recipes are recomputed via MINLP, and the process is repeated until there are no infeasibilities.

Glismann and Gruhn (2001) used a discrete-time multi-period NLP to compute blend recipes and a discrete-time multi-period MILP to solve the short term scheduling problem. The planning periods are defined by product demands and other specific planning priorities and the scheduling periods were defined to be 2 hours long. If a feasible solution cannot be found for the scheduling problem or if the deviations from the goals cannot be accepted, the planning step is solved again but this time including the information from the MILP solution through the addition of constraints regarding the blend components consumption. The new blend recipes computed are added to the scheduling problem and can be chosen as alternatives to the old recipes. In order to avoid many recipe changeovers, constraints are added to enforce a minimum running time for particular recipes on a blender.

Joly and Pinto (2003) presented a discrete-time MILP model to optimize the scheduling of fuel oil and asphalt production. They assumed linear blending properties and solved three real-world examples using a schedule horizon from 1 to 6 days, divided in uniform time periods of 2 hours each. In one of their examples, feasible solutions with relative optimality gaps varying from 1.3 to 8.2% were found in less than 4 CPU hours.

They reported that smaller relative gaps required more time and sometimes algorithm failure was observed.

An alternative to the discrete time representation is a continuous-time model. Jia and Ierapetritou (2003) solved simultaneously gasoline blend scheduling and distribution. They presented a continuous-time event-based MILP model for the scheduling problem of the third subsystem. The model includes multi-purpose product tanks (tank switching), delivery of the same order from multiple product tanks, and one product tank delivering multiple orders. They solved realistic large-scale problems to global optimality in less than 5 CPU hours. According to Li and Karimi (2011), Jia and Ierapetritou's model may lead to infeasible solutions which have one tank holding more than one product at a time.

Mendez et al. (2006) introduced both discrete-time and continuous-time models to optimize blend recipes and schedule operations using an iterative algorithm. The model includes non-linear quality constraints and employs their linearization to formulate a MILP. After the blend recipes are computed, correction factors for the product properties are calculated. Algorithm stops when the correction factors converge and the products properties meet the specifications. Minimum blend run constraints and multipurpose tanks are not included in the models. Mendez et al. (2006) and Kelly and Mann (2003a, 2003b) have pointed out that solving logistics and quality aspects for large-scale problems required significant amount of time with then-current MINLP codes and global optimization techniques. Even though recent advances in MINLP and MILP codes have made it possible to solve much larger problems, there is a continuous research effort for improved formulations that lead to shorter solution times and/or more accurate results.

Li, Karimi, and Srinivasan (2010) presented a continuous-time slot-based model that uses process slots to simultaneously consider computation of recipes, blender operation, inventory constraints and order scheduling. Quality constraints are handled using blend indices (i.e. they are linear constraints). Their MILP model includes parallel non-identical blenders, multipurpose tanks and other attributes that describe systems encountered in practice. Their objective function does not include penalties to minimize deviations from a preferred or average blend recipe since their model computes a blend recipe for each specific blend run. In order to ensure a constant blend rate, they included a schedule adjustment step in their algorithm. While their model poses many of the characteristics of the industrial systems, computational times for large examples are still not reasonable; nevertheless, their solutions were better than those provided by DICOPT and BARON trying to solve the corresponding MINLP model. Li and Karimi (2011) replaced process slots with unit slots and expanded the model by Li et al. (2010) to include blender setup times, limited inventory of components and simultaneous receipt/delivery by the product tanks. Compared to work of Li et al. (2010), they accomplish a significant reduction of computational times for smaller size problems. Their results show that some realistic-scale problems (2 to 3 blenders, 4 to 6 products, 9 components, 9 properties, 11 product tanks, 10 to 45 orders, and a planning horizon of 8 days) can be solved, but others were unsolvable to proven optimality within the allocated CPU times (10800 to 108000 seconds, depending of the problem).

1.4 Thesis Work

This work introduces algorithms designed to improve discrete-time approach to blend planning. The goals are:

(i) Maintain the blend recipe for each grade of gasoline as constant as possible across the planning horizon, i.e. minimize the number of different blend recipes,

(ii) Determine a blend plan that is feasible at the boundaries of time periods, thereby making it easier to produce a feasible schedule, and

(iii) Use nonlinear single-period blending models to arrive at solutions that have objective function value equal to (or very close to) the solutions obtained by the multi-period approach, thereby reducing the computational requirements of the problem.

Basis for the algorithm is a notion of an inventory pinch point. An inventory pinch point is defined as the point where the cumulative average total production curve, CATP curve, (which is based on optimal blend recipes that will meet the demands for various grades of gasoline) is tangent to the cumulative total demand curve, CTD curve, and if CATP curve is extrapolated from this point onwards it will not cross the CTD curve (see Figure 3.4). The top level of the algorithm optimizes blend recipes, while the lower level of the algorithm computes blend profile (when and how much of each grade to blend) based on the recipes computed at the top level. The algorithm uses different period lengths at each level. At the top level, periods are delimited by the inventory pinch points, while the lower level has fine-grid fixed-length periods. In order to distinguish between the periods at top and the lower level, they will be called *t*-period and *l*-period, respectively. If inventory infeasibilities (i.e. insufficient gasoline inventory) are detected in some *l*-period in the solution of the MILP, the corresponding *t*-period at the top level is subdivided; then the blend recipes are re-optimized at the top level and the lower level MILP is resolved. The objective at the top level is to minimize the total cost of blending. Since the refinery operations are based on the results of the longer term production plan, process units continuously produce gasoline blend components as determined by that plan. These components are stored in an intermediate storage and then used to blend gasoline. The refinery carries the inventories either as blend components or as finished gasoline, as determined by the refinery production plan. Hence, the cost of carrying the product inventory over the gasoline blend planning and scheduling horizon (typically 1 to 2 weeks) does not need to be included in the gasoline planning objective function, since the refinery will carry either the blend components or the finished gasoline. The lower level MILP computes how much of each grade of gasoline to blend in each *l*-period. If the recipes computed at the top level lead to a feasible production plan, then the objective function at the lower level is equal to zero. 30 case studies are presented and results of the algorithms are compared with those computed via multi-period MINLP solved by DICOPT.

Chapter 2. Problem Statement

As mentioned in section 1.3, the current practice in gasoline blending relies on discrete-time decomposition of the blending problem in two levels. At the top level, a multi-period MILP or MINLP computes blend recipes, while the lower level computes (heuristically or algorithmically) the corresponding schedule. This work pursues improvements in the discrete-time formulation, with the intent to enhance the current practice.

The sample system shown in Figure 1.2 will be used to describe attributes of a gasoline blend planning model. Blend planning problem is described by:

Component supply and product demand data:

1. Planning horizon [0, H] divided into fixed-duration time periods 1,2,…N.
2. Set of blend components, their properties, initial inventories, costs, and flow rates along the planning horizon (i.e. supply profile).
3. Set of products (i.e. gasoline grades) with prescribed minimum and maximum quality specifications, their initial inventories and corresponding initial quality.
4. Set of delivery orders for each product along the planning horizon (i.e. demand profile). If there are multiple orders for the same grade of gasoline in a given time period, these orders are lumped into a total demand for that grade of gasoline in that period.

Blending system structure and operating characteristics considered:

1. Minimum and maximum inventories (for every time period) for each blend component and for each grade of gasoline. If there are multiple tanks available for storage of the same material, they are treated as one aggregate inventory capacity for that material.
2. There can be one or more blenders operating in parallel. Maximum blending capacity of each blender is given.
3. Blender capacity loss due to switching the blender from one grade to another.
4. If a specific grade of gasoline is to be blended, then the amount blended must be greater than or equal to some threshold amount.

The goal is to compute:

1. Which gasoline grades to produce in each time period and their respective volumes.
2. How much of each blend component to use for each gasoline blend (blend recipes) for each grade and for each period.
3. Inventory profiles of the components and of finished gasoline.

Assumptions are:

1. Refinery production plan has determined the crude feed rate as required to meet the demand for various products. Hence, gasoline blend components are available in required quantities to meet the total gasoline demand during any time period of the production plan.
2. Component qualities are constant for the whole planning horizon. Since blend components are produced by upstream units which are operated to meet target

qualities of the blend components, this assumption is valid so long as the process units produce blend components to their target qualities.

3. Quality constraints are either linear or nonlinear. For simplicity, only one nonlinear constraint (Reid vapor pressure) is included in some case studies.
4. Perfect mixing occurs in the blender.
5. Changeover times between product runs on the blender are product and sequence independent.
6. Each order involves only one product.
7. Each order is completed during the planning horizon.
8. Product liftings (demands) need to be met.
9. If the initial inventory (heel) of some product tank is off-spec, it will be used as blend component to produce on-spec gasoline.
10. Blend planning horizon is typically 1 to 2 weeks long with periods being one day or half a day in duration.

Note that the above multi-period model implies:

1. Within each time period, the products required during that period are first produced in the required quantities (if they are not available in the storage tanks), and then the products are shipped ("lifted") in the required quantities.
2. Product inventory tank may receive new blend while gasoline is being withdrawn from the tank.
3. Component inventory tank may receive additional amounts of the component while the material is being withdrawn from the tank and sent to the blender.

Chapter 3. Inventory Pinch Concept

For simplicity's sake, consider a single product gasoline blender, i.e. imagine the system shown in Figure 1.2 but having only one blender and one product. In order to simplify exposition and the graphs, data corresponding to fixed time periods will be used, where a "fixed time period" is specified by its start and end time. This does not impact the method or the conclusions. Cumulative demand curve (CTD) vs. time will be plotted. Note that this curve can be constructed from actual demand data (e.g. a shipment can start 3 days and 7hr or at some other time from the start of the planning horizon), i.e. CTD does not have to use aggregate data per day. Cumulative total production (CTP) consists of the available initial product inventory, V(0), and the amount produced up to a specific point in time. If it is assumed that the blending will be carried out at the same average rate throughout the horizon, then the cumulative average total production (CATP) is a straight line connecting the initial available inventory to the final total demand. Clearly, in the example shown in Figure 3.1, blending at average rate for the entire horizon is not feasible (cumulative demand curve CTD becomes greater than the cumulative average total production CATP starting in period 6). In order to have a feasible operation, cumulative average total production (CATP) must always be greater than or equal to the cumulative demand (CTD). Hence, a feasible blending operation requires that the cumulative average total production curve CATP traverse from the initial available inventory V(0) to the point where it touches the cumulative total demand curve CTD and then to the final point on CTD curve (see Figure 3.2). Inventory pinch points are the points where cumulative average production curve CATP becomes tangent to the cumulative total demand curve CTD. Shown in Figure 3.1 is also the maximum possible product volume, which corresponds to the maximum available blender capacity.

One lower bound on the optimum solution is obtained by solving the total aggregate blending problem (Eq. (1) to (9)). This solution is a single blend recipe (per product) to be applied along the entire horizon; hence the entire horizon is considered as a "fixed recipe interval". If the blending occurs at the same blend rate throughout the horizon, then the blender operates at constant rate equal to the average rate across the horizon. Corresponding to the average blend rate is the average available product volume, represented by CATP line. Since this is infeasible, the blend rate has to be increased until the cumulative product volume line becomes tangent to the cumulative total demand curve (segment CATP1 in Figure 3.2), i.e. it is required to produce sufficient amount of gasoline to meet the cumulative demand at the pinch point. In the example shown in Figure 3.2, product blending in the interval from the pinch point to the end of the horizon proceeds along CATP2 segment. If blend components are available in sufficient quantities at the appropriate points along the horizon, and if no inventory constraints are encountered along the horizon, then a single blend recipe can be used to blend the volumes corresponding to the cumulative product curve (CATP1 and CATP2 segments in Figure 3.2). Whether or not such operation is feasible can be determined by solving a multi-period MILP which uses the aggregate blend recipe (which is constant for all periods) to determine how much to blend in each period.

If blending with fixed recipe across the entire horizon is not feasible, the infeasibility is caused by insufficient amount of blend components required for blending based on optimal recipe while meeting the demand at the pinch points. Therefore, the aggregate blend recipe from the start of the horizon to the first inventory pinch point (period 1) is computed. This recipe will make the best possible use of the components available in period 1. Similarly, another aggregate blend recipe needs to be computed from the pinch point to the end of the horizon (or to the next pinch point, if there are more than one), as shown in Figure 3.2.

If the amount of available blend components and their qualities in period 1 are such that the cumulative demand at the pinch point cannot be met, then the problem is infeasible. If the product unit cost corresponding to the blend recipe computed for interval is higher than the aggregate blend recipe for the entire horizon, this is due to the fact that the mix of components available during this period does not allow to blend at the lowest possible cost.

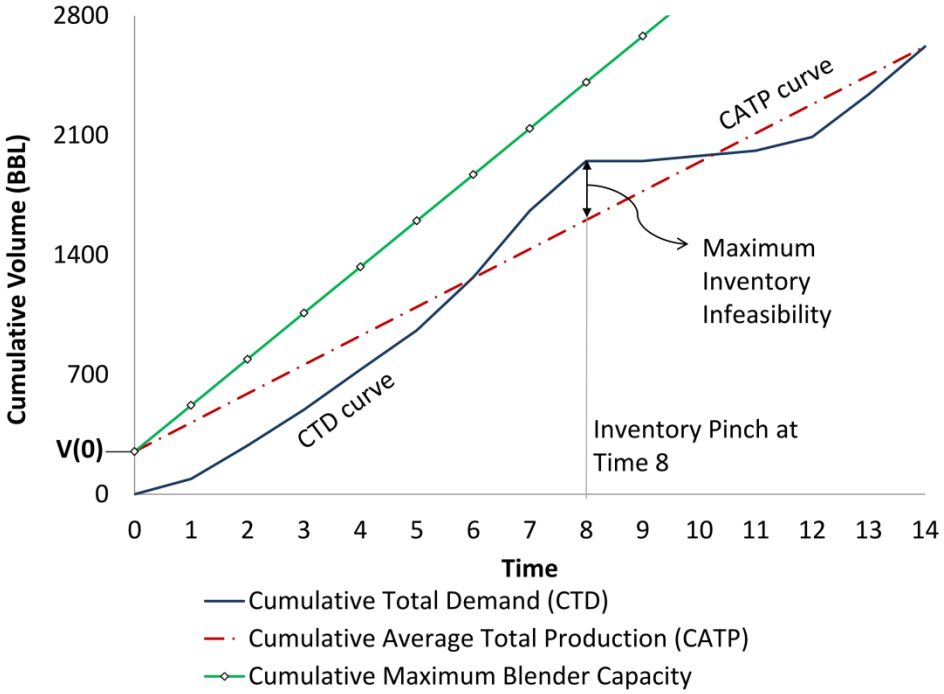


Figure 3.1: Problem with one pinch point

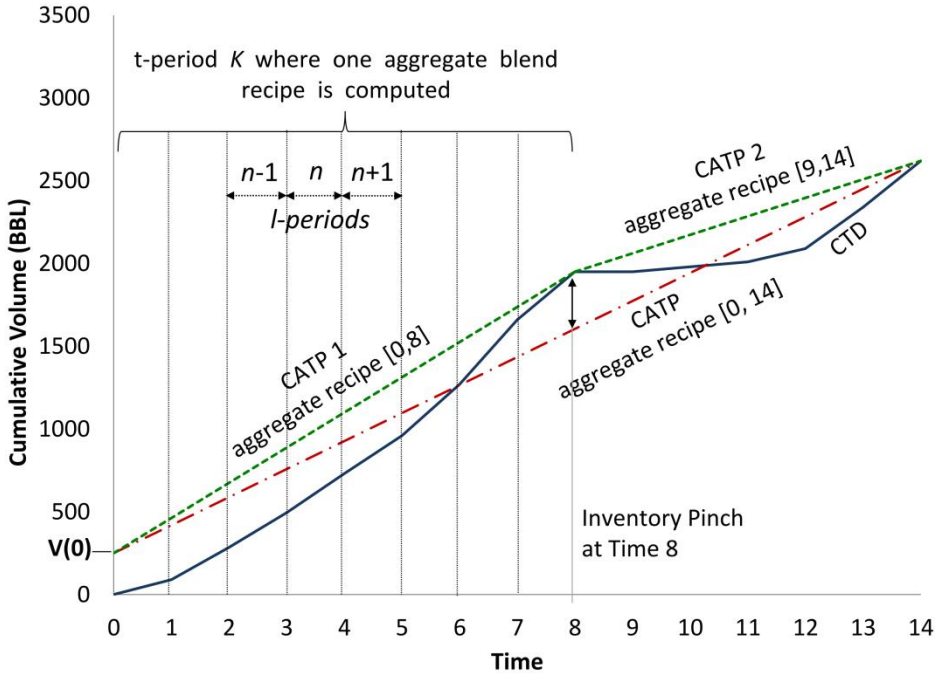


Figure 3.2: Graphic representation of the inventory pinch algorithm

From here on, the term *t*-period (i.e. top level period) will be used as the length of time where one aggregate recipe will be computed (i.e. aggregate interval). A *t*-period contains at least one *l*-period from the lower level (see Figure 3.2 and Figure 3.3).

It should be pointed out that cumulative average total production curve CATP should be constructed as shown by segment CATP2 in Figure 3.4. End point of a segment of the CATP curve must be a tangent to the cumulative demand curve CTD and if CATP segment is extended beyond the tangent point, the extension must not intersect the cumulative demand curve. If the aggregate blending problem corresponding to CATP2 is solved, the solution includes the "local" inventory pinch in t=3. Hence, solving for an aggregate blend recipe for CATP1 is not necessary. Figure 3.5 shows an example with two true inventory pinch points.

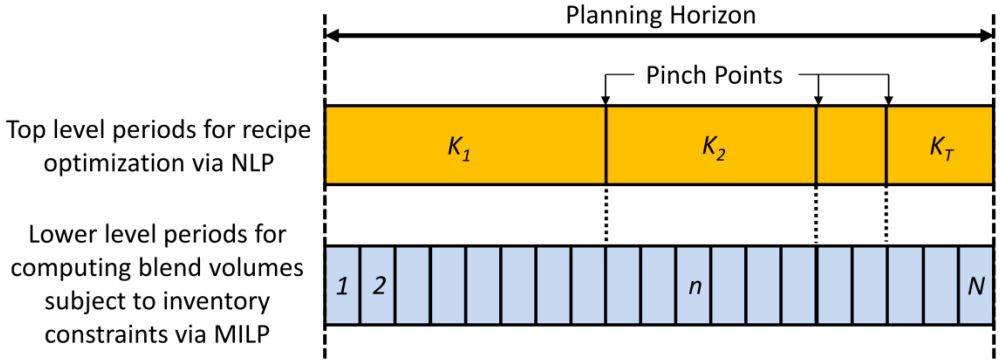


Figure 3.3: Time grids for the two levels of the algorithm

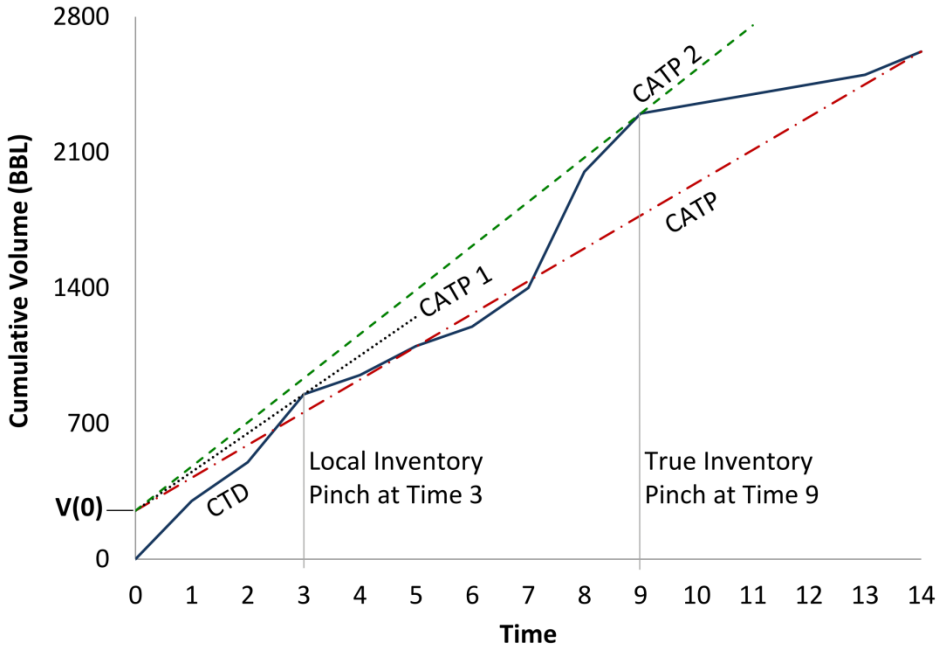


Figure 3.4: One "local" inventory pinch and one true inventory pinch

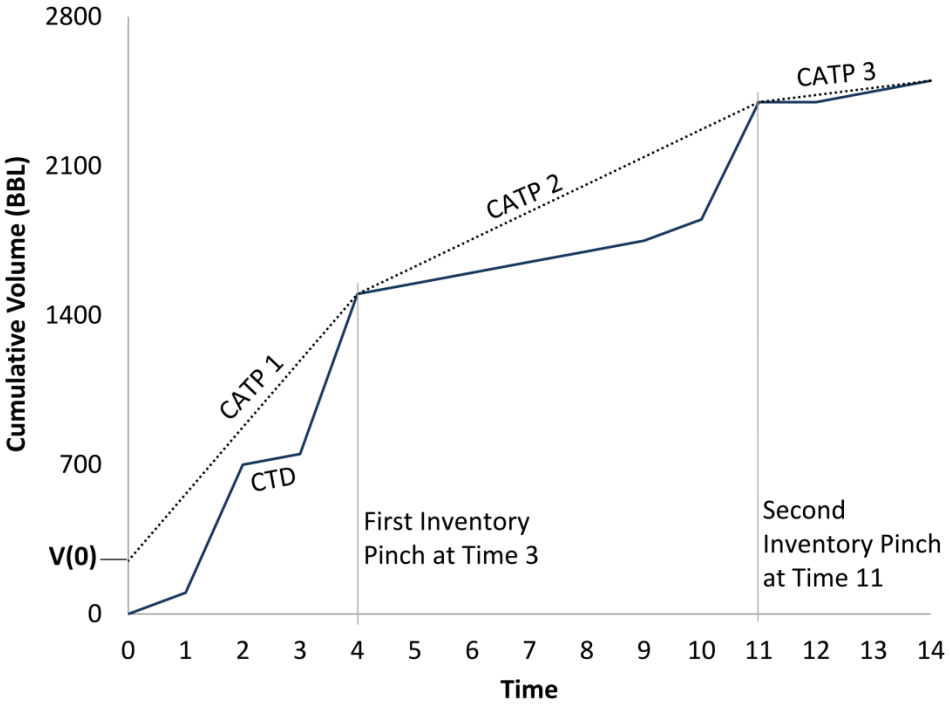


Figure 3.5: Example with two true inventory pinch points

When introducing inventory pinch, it was assumed that there is only one product being blended. That assumption does not alter the methodology, if (i) cumulative demand curve is defined to be the cumulative total demand (CTD) of all products, and (ii) cumulative product volume (CATP) is defined to be the sum of all product volumes.

Having computed aggregate blend recipe for each *t*-period delimited by the pinch points, it is still needed to determine how much of each gasoline grade to produce at what point in time. For this purpose, a multi-period (*l*-periods) fixed-recipe volume-only MILP model is formulated. Purpose of the optimization is to find the blend volumes for each *l*-period in such a manner that possible infeasibilities in the product inventories are pushed as far into the future as possible. Constraints are fixed blend recipe equations, inventory constraints on the product and component tanks, product demands, and blend component availability. Volumetric constraints on the inventory tanks include non-negative positive and negative slack variables. Cost coefficients (i.e. penalties) for the product inventory slacks are decreasing along the time horizon, thereby "pushing" any possible product inventory infeasibility as far forward as possible. In addition, one may include minimum threshold blend amount for each grade and capacity lost due to the set-up time when switching between the grades.

If blend components are not available as required by the blend recipe for a given *t*-period, then the solution of the multi-period volume-only MILP model will have infeasible product inventories in some *l*-periods which are contained within that *t*-period. This can be seen directly on the product inventory profiles and also in the cumulative curves (Figure 3.6). The algorithm presented in this work eliminates these infeasibilities via an iterative procedure.

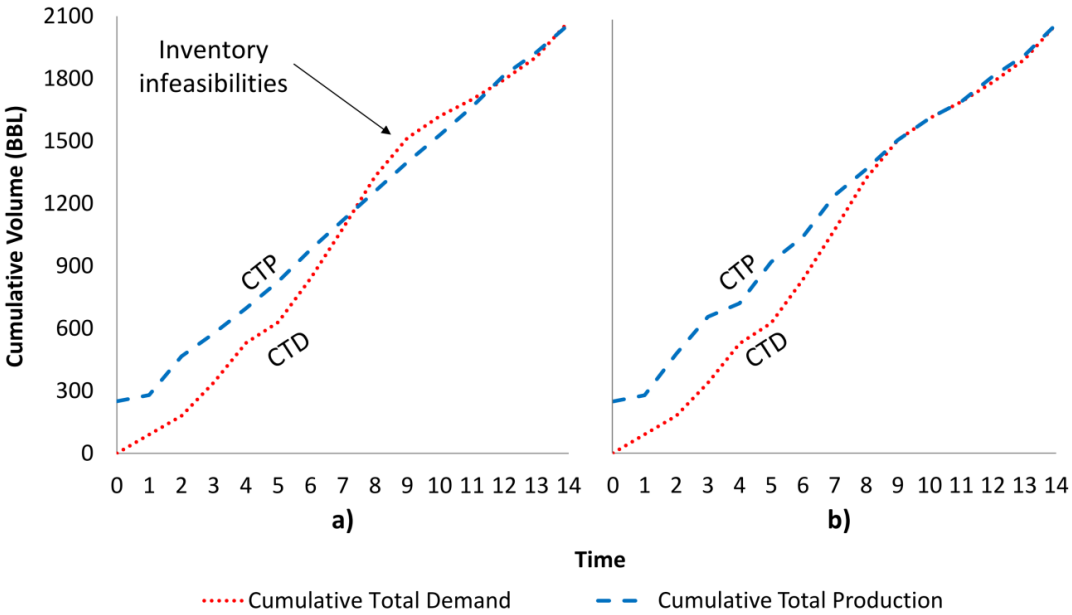


Figure 3.6: Inventory infeasibilities elimination by inventory pinch algorithm. a) Infeasible blend recipes, b) feasible blend recipes

Chapter 4. Inventory Pinch Algorithm

The inventory pinch algorithm decomposes the gasoline blend planning in two levels (Figure 4.1), each one based on a different view of the blending process:

Top level: Computation of aggregate blend recipes (quality constraints fulfilment) that leads to the lowest possible cost per unit of the volume blended, subject to availability of the components in a given *t*-period.
Lower level: Computation of volumes to be blended in each *l*-period while meeting the inventory, demand, and blend threshold constraints. The blend recipe computed for a *t*-period is fixed for all the *l*-periods that it contains.

At the top level, the planning horizon is divided in K_T *t*-periods delineated by the inventory pinch points. Blend recipes between inventory pinch points are often constant. The model at the top level does not consider variations of component inventories over the horizon only the overall component availability. As mentioned in section 1.4, since gasoline blend planning is carried out within the constraints computed by the longer range production plan, the inventory carried by the refinery is determined by that plan. Hence, the refinery caries either the inventory of blending components or the inventory of blended gasoline. Over gasoline blend planning horizon, the carrying costs of these inventories are fixed and need not to be included in the objective function. Once blend recipes are known, how much of each grade to blend across the entire blend planning horizon is computed via fixed-recipe volume-only MILP model. Length of *l*-periods (total of *N*) in the MILP model is selected in a manner that makes it simple to make decisions at the scheduling level (e.g. one day, half a day, or shorter). If the blend recipes from the top level lead to a state where at some point along the horizon there are insufficient amounts of the blend components, then infeasible product inventories (denoted as inventory infeasibilities) will appear in the lower level. Formulation of the MILP is such that any potential infeasibilities are "pushed" forward, as far as possible, in the planning horizon. If product infeasibilities are found in the lower level for a specific *l*-period, it means that either the blender capacity is too small or that the amount of components available in the *l*-period is insufficient. In either case, it means that an additional amount of the product needs to be blended prior to the infeasibility; that additional amount is equal to the infeasible volume for that product. If there are any infeasibilities, then the *t*-period at the top level is subdivided and new recipes are computed (i.e. K_T increments in each iteration). Case studies showed that under constant component supply the algorithms lead to the optimum solution computed via multi-period MINLP. If there are significant variations in component supply, then the algorithms produce solutions that are very close to the optimum.

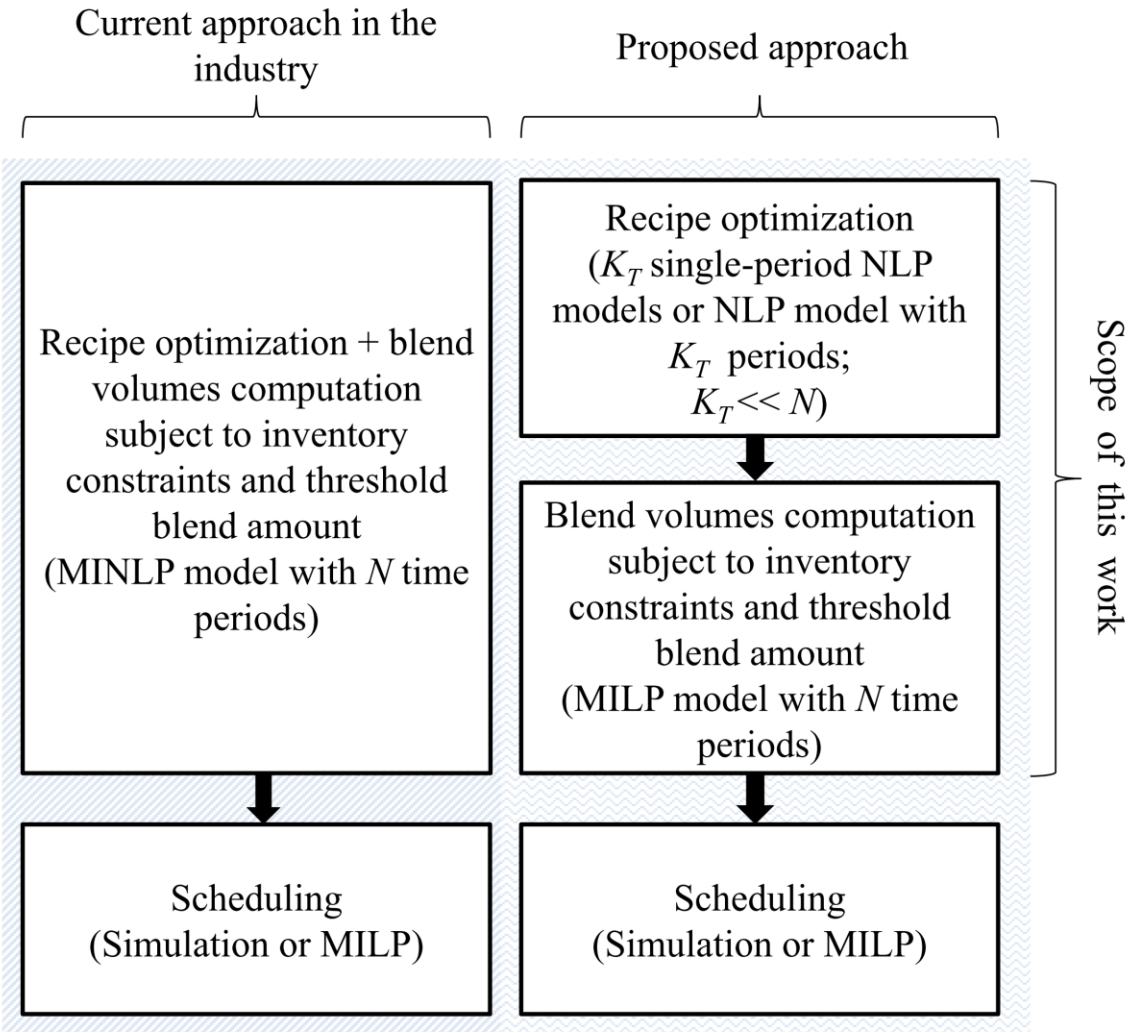


Figure 4.1: Proposed decomposition of gasoline blending

4.1 Mathematical Model for the MINLP formulation

In this section, a mixed-integer nonlinear program will be presented for the gasoline blend planning model. From this MINLP model, the mathematical models for the top and lower levels of the algorithms will be derived. The following MINLP model computes simultaneously the blend recipes and how much volume of each product is required to be blended at each period along the planning horizon.

The objective function minimizes the blend cost and is given by Eq. (4.1). Slack variables are included to ensure that the solver always returns a numerical solution. If any of the slack variables are non-zero, the solution is physically unrealizable since the

inventory constraints are violated. The penalty coefficients for the slacks variables are much greater than the cost of the blending components

$$\min\left\{\sum_{n}\left(\begin{array}{l}\sum_{bl}\sum_{g}\sum_{i}V_C(i,g,n,bl)\times Cost(i)+\sum_{i}\left(S_C^{+}(i,n)+S_C^{-}(i,n)\right)\times Penalty_C(i)\\+\sum_{g}\left(S_P^{+}(g,n)+S_P^{-}(g,n)\right)\times Penalty_P(g)\end{array}\right)\right\} \quad (4.1)$$

The volume balance equations and inventory constraints for the component and product tanks for every period *n* are given by (4.2) to (4.7). Here the slack variables S_C^+, S_C^-, S_P^+, and S_P^- are introduced to detect infeasibilities in the blending component and product inventory levels.

$$V_C^{close}(i,n) = V_C^{open}(i,n) + V_C^{in}(i,n) - \sum_{bl}\sum_{g} V_C(i,g,n,bl) + S_C^{+}(i,n) - S_C^{-}(i,n) \quad \forall\, i,n \quad (4.2)$$

$$V_C^{min}(i) \le V_C^{close}(i,n) \le V_C^{max}(i) \quad \forall\, i,n \quad (4.3) \text{ and } (4.4)$$

$$V_P^{close}(g,n) = V_P^{open}(g,n) + \sum_{bl} V_B(g,n,bl) - D_P(g,n) + S_P^{+}(g,n) - S_P^{-}(g,n) \quad \forall\, g,n \quad (4.5)$$

$$V_P^{min}(g) \le V_P^{close}(g,n) \le V_P^{max}(g) \quad \forall\, g,\, n \quad (4.6) \text{ and } (4.7)$$

Equations (4.8) and (4.9) set the closing inventories of period *n* as the opening inventories of period *n+1*.

$$V_C^{open}(i,n+1) - V_C^{close}(i,n) = 0 \quad \forall\, i,n=1,\ldots,N\text{-}1 \quad (4.8)$$

$$V_P^{open}(g,n+1) - V_P^{close}(g,n) = 0 \quad \forall\, g,n=1,\ldots,N\text{-}1 \quad (4.9)$$

Eq (4.10) and Eq (4.11) together represent the volume balance around a blender, in this case by using the x(*i,g,n,bl*) variable the blend recipe is directly computed. Eq (4.10) is nonlinear since x(*i,g,n,bl*) and V_B(*g,n,bl*) are not known. This formulation is used due to the quality constraint requiring the x(*i,g,n,bl*) variable (Eq (4.18)). Eq. (4.15) indicates that the number of grades that can be blended in one period *n* in blender *bl* must be less than or equal to the parameter np(*bl*). If the binary variable A(*g,n,bl*) is 1, the volume to blend of grade *g* in blender *bl* will be between the minimum and maximum specified limits (Eq. (4.12) and (4.13)), and less than or equal to the maximum blending rate. Since a discrete-time formulation is used, t(*n*) is known in advance. If A(*g,n,bl*) is 0, then product *g* will not be blended in period *n* in blender *bl*. The lost volume term due to setup times for switching from blending one grade to another is included in Eq. (4.14) and it is equal to the maximum blender capacity times the setup time.

$$V_C(i,g,n,bl) - x(i,g,n,bl) \cdot V_B(g,n,bl) = 0 \qquad \forall\ i,g,n,bl \qquad (4.10)$$

$$\sum_i x(i,g,n,bl) = 1 \qquad \forall\ g,n,bl \qquad (4.11)$$

$$V_B^{min}(g) \cdot A(g,n,bl) - V_B(g,n,bl) \le 0 \qquad \forall\ g,n,bl \qquad (4.12)$$

$$V_B^{max}(g) \cdot A(g,n,bl) - V_B(g,n,bl) \ge 0 \qquad \forall\ g,n,bl \qquad (4.13)$$

$$\sum_g V_B(g,n,bl) + V_B^{lost}(bl) \cdot \sum_g A(g,n,bl) - F_B^{max}(bl) \cdot t(n) \le 0 \qquad \forall\ n,bl \qquad (4.14)$$

$$\sum_g A(g,n,bl) \le np(bl) \qquad \forall\ n,bl \qquad (4.15)$$

Eq (4.16) and Eq (4.17) are the quality constraints for those properties considered to blend linearly.

$$Q_P^{min}(g,s) \cdot V_B(g,n,bl) \le \sum_i Q_C(i,s) \cdot V_C(i,g,n,bl) \le Q_P^{max}(g,s) \cdot V_B(g,n,bl) \quad (4.16) \text{ and } (4.17)$$

$$\forall\ g,n,bl,s = \{\textit{linear properties}\}$$

Eq (4.18) is the blending index approach to model the Reid vapour pressure (RVP) used by Singh, Forbes, Vermeer, and Woo (2000). Eq (4.19) and Eq (4.20) are the quality constraints for nonlinear properties.

$$Q_P(g,s,n,bl) = \left[\sum_{i=1}^{I} x(i,g,n,bl) \cdot \left(Q_C(i,s)\right)^{1.25} \right]^{0.8} \qquad \forall\ g,n,bl,s = RVP \qquad (4.18)$$

$$Q_P^{min}(g,s) \le Q_P(g,s,n,bl) \le Q_P^{max}(g,s) \qquad \forall\ g,n,bl,s = RVP \quad (4.19) \text{ and } (4.20)$$

Equations (4.1) to (4.20) comprise the MINLP model (RVP blends nonlinearly) or only Eq. (4.1) to (4.17) if RVP is assumed to blend linearly.

If the initial product inventories are off-spec and all properties are assumed to blend linearly, Eq. (4.21) and (4.22) replace Eq. (4.16) to (4.17) only in the first time period (n=1).

$$\sum_{bl}\sum_i Q_C(i,s) \cdot V_C(i,g,n,bl) + Q_P^{open}(g,s) \cdot V_P^{open}(g,n) \ge Q_P^{min}(g,s) \cdot \left(\sum_{bl} V_B(g,n,bl) + V_P^{open}(g,n) \right)$$

$$\forall\ g,bl,s = \{\textit{linear properties}\},\ n=1 \qquad (4.21)$$

$$\sum_{bl}\sum_{i} \mathrm{Q_C}(i,s)\cdot \mathrm{V_C}(i,g,n,bl) + \mathrm{Q_P^{open}}(g,s)\cdot \mathrm{V_P^{open}}(g,n) \le \mathrm{Q_P^{max}}(g,s)\cdot\left(\sum_{\mathrm{bl}} \mathrm{V_B}(g,n,bl) + \mathrm{V_P^{open}}(g,n)\right)$$
$$\forall\, \mathrm{g, bl, s} = \{\textit{linear properties}\},\ n{=}1 \qquad (4.22)$$

4.2 Mathematical Model for the Top Level of the Inventory Pinch Algorithm

The objective of the top level is to determine the volume ratios (i.e. blend recipes) to mix different blending components available at the refinery in such a way that the final products meet the demand and quality specifications and the blend cost is the minimum possible. The following subsections will explain the multi-period and single-period models used in the inventory pinch algorithms, as well as the model to bring the initial product inventories back to specification if required.

4.2.1 Multi-Period Formulation

This model is similar to the MINLP formulation, but instead of *N* periods, it has K_T *t*-periods ($K_T \le N$); and it does not have binary variables (it is a NLP model). The boundaries of these K_T *t*-periods are initially defined by the inventory pinch points. One *t*-period contains one or more *l*-periods; therefore, the products' demand and blend components' supply on a *t*-period are the aggregated values of the corresponding *l*-periods. This applies as well for the Single-Period Formulation (section 4.2.2). The objective function defined by Eq. (4.23) minimizes the blend cost. The penalty coefficients for the product slacks variables are much greater than the penalty coefficients for the component slacks variables (i.e. $\mathrm{Penalty_{P,K}(g)} >> \mathrm{Penalty_C}(i)$).

$$\min\left\{\sum_{k}\left(\begin{array}{l}\displaystyle\sum_{g}\sum_{i} V_{C,K}(i,g,k)\times Cost(i) + \sum_{i}\left(S^{+}_{C,K}(i,k)+S^{-}_{C,K}(i,k)\right)\times Penalty_{C}(i)\\ \displaystyle+\sum_{g}\left(S^{+}_{P,K}(g,k)+S^{-}_{P,K}(g,k)\right)\times Penalty_{P,K}(g)\end{array}\right)\right\} \qquad (4.23)$$

The volume balance equations, inventory constraints and inventory connections for component and product tanks for every *t*-period *k* are given by Eq. (4.24) to (4.31).

$$\mathrm{V}_{C,K}{}^{close}(i,k) = \mathrm{V}_{C,K}{}^{open}(i,k) + \mathrm{V_{C,K}^{in}}(i,k) - \sum_{g}\mathrm{V_{C,K}}(i,g) + S^{+}_{C,K}(i,k) - S^{-}_{C,K}(i,k) \quad \forall\, i,k \qquad (4.24)$$

$$\mathrm{V_C^{min}}(i) \le \mathrm{V_{C,K}^{close}}(i,k) \le \mathrm{V_C^{max}}(i) \qquad \forall\, i,k \qquad (4.25)\text{ and }(4.26)$$

$$\mathrm{V_{C,K}^{open}}(i,k+1) - \mathrm{V_{C,K}^{close}}(i,k) = 0 \qquad \forall\, i,k=1,\ldots,K_T\text{-}1 \qquad (4.27)$$

$$V_{P,K}^{close}(g,k) = V_{P,K}^{open}(g,k) + V_{B,K}(g,k) - D_{P,K}(g,k) + S_{P,K}^{+}(g,k) - S_{P,K}^{-}(g,k) \quad \forall\, g,k \quad (4.28)$$

$$V_{P}^{min}(g) \le V_{P,K}^{close}(g,k) \le V_{P}^{max}(g) \qquad \forall\, g,k \qquad (4.29) \text{ and } (4.30)$$

$$V_{P,K}^{open}(g,k+1) - V_{P,K}^{close}(g,k) = 0 \qquad \forall\, g,k=1,\ldots,K_T\text{-}1 \qquad (4.31)$$

Eq. (4.10) and (4.11) are rewritten as Eq. (4.32) and (4.33), respectively. Although Eq. (4.12), (4.13), (4.14), and (4.15) can be rewritten for this model, they are omitted to avoid a MINLP formulation; the algorithm will deal with the maximum blending capacity and the minimum threshold required to blend of each product by setting appropriate constraints at the lower level in the MILP model.

$$V_{C,K}(i,g,k) - x_K(i,g,k) \cdot V_{B,K}(g,k) = 0 \qquad \forall\, i,g,k \qquad (4.32)$$

$$\sum_i x_K(i,g,k) = 1 \qquad \forall\, g,k \qquad (4.33)$$

Eq. (4.16) to (4.20) are rewritten as Eq. (4.34) to (4.38).

$$Q_P^{min}(g,s) \cdot V_{B,K}(g,k) \le \sum_i Q_C(i,s) \cdot V_{C,K}(i,g,k) \le Q_P^{max}(g,s) \cdot V_{B,K}(g,k) \qquad (4.34) \text{ and } (4.35)$$

$\forall\, g, k, s = \{\text{linear properties}\}$

$$Q_P(g,s,k) = \left[\sum_{i=1}^{I} x_K(i,g,k) \cdot \left(Q_C(i,s)\right)^{1.25} \right]^{0.8} \qquad \forall\, g, k, s = RVP$$

(4.36)

$$Q_P^{min}(g,s) \le Q_P(g,s,k) \le Q_P^{max}(g,s) \qquad \forall\, g, k, s = RVP \qquad (4.37) \text{ and } (4.38)$$

Equations (4.23) to (4.38) complete the NLP model for the top level of the multi-period algorithm (RVP blends nonlinearly) or only Eq. (4.23) to (4.35) if RVP is assumed to blend linearly.

In the first iteration of the algorithm, the volumes to be blended in each *t*-period are set in order to have the product closing inventories at the minimum allowed and in the following iterations they are set by Eq. (4.39), which links the variables from the lower level to the top level.

$$\mathrm{V}_{\mathrm{B,K}}(g,k) = \sum_{n\in k}\sum_{\mathrm{bl}}\left[\mathrm{V}_B(g,n,bl) + S_P^+(g,n) - S_P^-(g,n)\right] \qquad \forall\ g,k \qquad (4.39)$$

4.2.2 Single-Period Formulation

In this formulation, the blend recipes are computed by solving an aggregate NLP model for each *t*-period. The aggregate models are solved in sequence (solution from one *t*-period define the starting point of the next *t*-period). If the aggregate model is solved for the whole planning horizon, this is referred as the *total aggregate model* and its solution is a lower bound of the problem. If the total aggregate model is feasible, then proceed to compute the blend recipes for the periods delimited by the pinch points.

Eq. (4.40) is the objective function to minimize and it comprises of the components cost and the penalties for the inventory slack variables ($\mathrm{Penalty_P(g)} >> \mathrm{Penalty_C}(i)$).

$$\min\left\{\begin{array}{l}\displaystyle\sum_g\sum_i V_{C,K}(i,g)\times Cost(i) + \sum_i\left(S_{C,K}^+(i) + S_{C,K}^-(i)\right)\times Penalty_C(i)\\ \displaystyle+\sum_g\left(S_{P,K}^+(g) + S_{P,K}^-(g)\right)\times Penalty_P(g)\end{array}\right\} \qquad (4.40)$$

Eq. (4.2) to (4.7) are rewritten as Eq. (4.41) to (4.44). Since this is a single-period model, there is no *k* index.

$$\mathrm{V}_{C,K}{}^{close}(i) = \mathrm{V}_{C,K}{}^{open}(i) + \mathrm{V}_{\mathrm{C,K}}^{\mathrm{in}}(i) - \sum_g \mathrm{V}_{\mathrm{C,K}}(i,g) + S_{C,K}^+(i) - S_{C,K}^-(i) \qquad \forall\ i \qquad (4.41)$$

$$\mathrm{V}_{\mathrm{C}}^{\min}(i) \le \mathrm{V}_{\mathrm{C,K}}^{\mathrm{close}}(i) \le \mathrm{V}_{\mathrm{C}}^{\max}(i) \qquad \forall\ i \qquad (4.42)$$

$$\mathrm{V}_{\mathrm{P,K}}^{\mathrm{close}}(g) = \mathrm{V}_{\mathrm{P,K}}^{\mathrm{open}}(g) + \mathrm{V}_{\mathrm{B,K}}(g) - D_{\mathrm{P,K}}(g) + \mathrm{S}_{\mathrm{P,K}}^+(g) - \mathrm{S}_{\mathrm{P,K}}^-(g) \qquad \forall\ g \qquad (4.43)$$

$$\mathrm{V}_{\mathrm{P}}^{\min}(g) \le \mathrm{V}_{\mathrm{P,K}}^{\mathrm{close}}(g) \le \mathrm{V}_{\mathrm{P}}^{\max}(g) \qquad \forall\ g \qquad (4.44)$$

The volume balance around the blender is given by (4.45) and (4.46). No constraints are required since the algorithm will set appropriate targets for the product closing inventories.

$$\mathrm{V}_{\mathrm{C,K}}(i,g) - x_K(i,g)\cdot \mathrm{V}_{B,K}(g) = 0 \qquad \forall\ g,\ i \qquad (4.45)$$

$$\sum_i \mathrm{x_K}(i,g) = 1 \qquad \forall\, g \qquad (4.46)$$

Eq. (4.16) to (4.20) are rewritten as Eq. (4.47) to (4.51).

$$\mathrm{Q_P^{min}}(g,s)\cdot \mathrm{V_{B,K}}(g) \le \sum_{\mathrm{i}} \mathrm{Q_C}(i,s)\cdot \mathrm{V_{C,K}}(i,g) \le \mathrm{Q_P^{max}}(g,s)\cdot \mathrm{V_{B,K}}(g) \qquad (4.47) \text{ and } (4.48)$$

$\forall\, g$, s={linear properties}

$$Q_{P,K}(g, s=RVP) = \left[\sum_{i=1}^{I} x_K(i,g)\cdot\left(Q_C(i, s=RVP)\right)^{1.25}\right]^{0.8} \qquad \forall\, g, \text{ s=RVP} \qquad (4.49)$$

$$\mathrm{Q_P^{min}}(g,s) \le \mathrm{Q_{P,K}}(g,s) \le \mathrm{Q_P^{max}}(g,s) \qquad \forall\, g, s\text{=RVP} \qquad (4.50) \text{ and } (4.51)$$

Eq. (4.40) to (4.51) define the aggregate model for the case when the initial inventory is on-spec and RVP blends nonlinearly, and Eq. (4.40) to (4.48) if RVP blends linearly.

Equations (4.52) to (4.53) are required to provide parameters to the aggregate model; therefore, they need to be solved before the current *t*-period model. Equations (4.52) and (4.53) are required to link the inventories of adjacent *t*-periods. As in the multi-period formulation, in the first iteration of the algorithm the volumes to be blended in each *t*-period are set in order to have the product closing inventories at the minimum allowed and in the following iterations they are set by Eq. (4.54), which links the variables from the lower level to the top level.

$$\mathrm{V_{C,K}^{open}}(i) - \mathrm{V_{C,K\text{-}1}^{close}}(i) = 0 \qquad \forall\, i,\ K=K_2,\ldots,K_T \qquad (4.52)$$

$$\mathrm{V_{P,K}^{open}}(g) - \mathrm{V_{P,K\text{-}1}^{close}}(g) = 0 \qquad \forall\, g,\ K=K_2,\ldots,K_T \qquad (4.53)$$

$$\mathrm{V_{B,K}}(g) = \sum_{n\in K}\left(\mathrm{V_B}(g,n) + \mathrm{S_P^{+}}(g,n) - \mathrm{S_P^{-}}(g,n)\right) \qquad \forall\, g \qquad (4.54)$$

4.2.3 Initial product inventories off-spec: Tank heel quality correction

If the initial product inventory is off-spec, the inventory needs to be brought back to specification before it can be shipped/lifted. In order to do this, the off-spec heel needs to be considered as a blend component by introducing equations (4.55) and (4.56). The subscript "0" specifies the variables during this quality correction *t*-period K_0. The length of this *t*-period K_0 must be enough to bring back the product inventories into specification.

$$\sum_{\mathrm{i}} \mathrm{Q_C}(i,s)\cdot \mathrm{V_{C,0}}(\mathrm{i},g) + \mathrm{Q_{P,0}^{open}}(g,s)\cdot \mathrm{V_{P,0}^{open}}(g) \geq \mathrm{Q_P^{min}}(g,s)\cdot\left(\mathrm{V_{B,0}}(g) + \mathrm{V_{P,0}^{open}}(g)\right) \quad (4.55)$$

$\forall$ g, *s*

$$\sum_{\mathrm{i}} \mathrm{Q_C}(i,s)\cdot \mathrm{V_{C,0}}(\mathrm{i},g) + \mathrm{Q_{P,0}^{open}}(g,s)\cdot \mathrm{V_{P,0}^{open}}(g) \leq \mathrm{Q_P^{max}}(g,s)\cdot\left(\mathrm{V_{B,0}}(g) + \mathrm{V_{P,0}^{open}}(g)\right) \quad (4.56)$$

$\forall$ g, *s*

For the rest of the planning horizon (subscript "1") the heel is not required to be considered a blend component since it is already on-spec. Therefore, the quality balance is written as:

$$\mathrm{Q_P^{min}}(g,s)\cdot \mathrm{V_{B,1}}(g) \leq \sum_{\mathrm{i}} \mathrm{Q_C}(i,s)\cdot \mathrm{V_{C,1}}(\mathrm{i},g) \leq \mathrm{Q_P^{max}}(g,s)\cdot \mathrm{V_{B,1}}(g) \quad (4.57) \text{ and } (4.58)$$

$\forall$ *g*, s

The volume balance around the blender is given by Eq. (4.59) and (4.60) for the first *t*-period when the heel is corrected and for the rest of the planning horizon, respectively. Eq. (4.61) to (4.64) set the closing product inventories to be within the limits.

$$\mathrm{V_{P,0}^{close}}(g) = \mathrm{V_{P,0}^{open}}(g) + \sum_{\mathrm{i}} \mathrm{V}_{C,0}(i,g) - D_{P,0}(g) + \mathrm{S_{P,0}^{+}}(g) - \mathrm{S_{P,0}^{-}}(g) \qquad \forall \text{ g} \qquad (4.59)$$

$$\mathrm{V_{P,1}^{close}}(g) = \mathrm{V_{P,0}^{close}}(g) + \sum_{\mathrm{i}} \mathrm{V}_{C,1}(i,g) - D_{P,1}(g) + \mathrm{S_{P,1}^{+}}(g) - \mathrm{S_{P,1}^{-}}(g) \qquad \forall \text{ g} \qquad (4.60)$$

$$\mathrm{V_P^{min}}(\mathrm{g}) \leq \mathrm{V_{P,0}^{close}}(g) \leq \mathrm{V_P^{max}}(g) \qquad \forall \text{ g} \qquad (4.61) \text{ and } (4.62)$$

$$\mathrm{V_P^{min}}(\mathrm{g}) \leq \mathrm{V_{P,1}^{close}}(g) \leq \mathrm{V_P^{max}}(g) \qquad \forall \text{ g} \qquad (4.63) \text{ and } (4.64)$$

Eq. (4.65) and (4.66) identify the volume used during the heel correction ($\mathrm{V_{C,0}}$) and the volume used afterwards ($\mathrm{V_{C,1}}$). Eq. (4.67) to (4.70) are the inventory constraints.

$$\mathrm{V_{C,0}^{close}}(i) = \mathrm{V_{C,0}^{open}}(i) + \mathrm{V_{C,0}^{in}}(i) - \sum_{g} \mathrm{V}_{C,0}(i,g) + S_{C,0}^{+}(i) - S_{C,0}^{-}(i) \qquad \forall\, i \qquad (4.65)$$

$$\mathrm{V_{C,1}^{close}}(i) = \mathrm{V_{C,0}^{close}}(i) + \mathrm{V_{C,1}^{in}}(i) - \sum_{g} \mathrm{V}_{C,1}(i,g) + S_{C,1}^{+}(i) - S_{C,1}^{-}(i) \qquad \forall\, i \qquad (4.66)$$

$$\mathrm{V_C^{min}}(\mathrm{i}) \leq \mathrm{V_{C,0}^{close}}(i) \leq \mathrm{V_C^{max}}(i) \qquad \forall\, i \qquad (4.67) \text{ and } (4.68)$$

$$\mathrm{V_C^{min}}(i) \leq \mathrm{V_{C,1}^{close}}(i) \leq \mathrm{V_C^{max}}(i) \qquad \forall\, i \qquad (4.69) \text{ and } (4.70)$$

The objective function is given by Eq. (4.71). *XC* is a coefficient set to a value greater than 1 and much less than the penalties for the slack variables (in this work, *XC* = 10). It forces the off-spec heel to be corrected with the smallest possible blend volume and therefore to be ready to be lifted/shipped as soon as possible.

$$\min\left\{\begin{array}{l} XC\cdot\left(\sum_{g}\sum_{i}V_{C,0}(i,g)\cdot Cost(i)\right)+\sum_{g}\sum_{i}V_{C,1}(i,g)\cdot Cost(i) \\ +\sum_{i}\left(S^{+}_{C,0}(i)+S^{-}_{C,0}(i)+S^{+}_{C,1}(i)+S^{-}_{C,1}(i)\right)\cdot Penalty_{C}(i) \\ +\sum_{g}\left(S^{+}_{P,0}(g)+S^{-}_{P,0}(g)+S^{+}_{P,1}(g)+S^{-}_{P,1}(g)\right)\times Penalty_{P}(g) \end{array}\right\} \quad (4.71)$$

The aggregate model given by Eq. (4.55) to (4.71) is used to bring the initial product inventories back to spec, assuming that all properties blend linearly.

4.3 Mathematical Model for the Lower Level of the Inventory Pinch Algorithm

After the blend recipes are defined at the top level, it is required to calculate the volumes to be blended in each period of the lower level. This step of the algorithm is modeled as a MILP (Eq. (4.72) to (4.87)) in order to set constraints on the minimum and maximum amount of product to be blended in each *l*-period (threshold constraints). Objective function, Eq. (4.72), is constructed in such a manner that the solution will delay occurrence of potential inventory infeasibilities as far into the future as possible. The objective function contains only the penalties for the inventory slack variables. If the blend recipes computed at the top level correspond to a feasible operation, the inventory slack variables will be zero at the solution of MILP. If the blend recipes are infeasible, then the MILP solution will show which specific products and in which *l*-periods cannot be produced in the amounts required to meet the demands. The penalty for the components' inventory slack variables is much greater than the penalty for the products' inventory slacks ($Penalty_C(i)$ >>$Penalty_P(g,n)$ $\forall$ i,g,n) which forces the inventory infeasibilities to be on the products' side. Such formulation allows knowing how much additional volume for the product is required to meet the inventory constraints (if there is an infeasibility) and the aggregate model can compute the proper blend recipe. In addition, the penalty for a product inventory slack decrease with time ($Penalty_P(g,n)$> $Penalty_P(g,n+1)$ $\forall$ g, n) in order to move the infeasibilities as late as possible in the planning horizon. The blend plan cost is computed but not included in the objective function since fixed blend recipes and product target inventories are defined by the solution of the aggregate model. Since cost of switching is not included in the MILP model, the optimum solution of this model is 0.0 (i.e. slack variables are zero for a

feasible solution; therefore, their use in this model does not affect the solution obtained as in the case where they are used to minimize deviations between blend recipes). If the solution of the MILP has inventory infeasibilities, it signifies that either component supply or product demand are such that the recipes computed at the top level are not feasible within a *t*-period. The algorithm will subdivide such *t*-periods and re-optimize the blend recipes.

$$\min\left\{\sum_{n}\left(\begin{array}{l}\sum_{i}\left(S_C^+(i,n)+S_C^-(i,n)\right)\times Penalty_C(i)\\ +\sum_{g}\left(S_P^+(g,n)+S_P^-(g,n)\right)\times Penalty_P(g,n)\end{array}\right)\right\} \tag{4.72}$$

The volume balance equations and inventory constraints for the component and product tanks for every *l*-period *n* are given by Eq. (4.2) to (4.7). Equations (4.8) and (4.9) set the closing inventories as the opening inventories of the next *l*-period *n*. They are presented again in this section as Eq. (4.73) to (4.80).

$$\mathrm{V}_C^{\,close}(i,n) = \mathrm{V}_C^{\,open}(i,n) + \mathrm{V}_C^{\,in}(i,n) - \sum_{bl}\sum_{g}\mathrm{V}_C(i,g,n,bl) + S_C^+(i,n) - S_C^-(i,n) \quad \forall\, i,n \tag{4.73}$$

$$\mathrm{V}_C^{\min}(i) \le \mathrm{V}_C^{close}(i,n) \le \mathrm{V}_C^{\max}(i) \qquad \forall\, i,n \qquad \text{(4.74) and (4.75)}$$

$$\mathrm{V}_P^{close}(g,n) = \mathrm{V}_P^{open}(g,n) + \sum_{bl}\mathrm{V}_B(g,n,bl) - D_P(g,n) + \mathrm{S}_P^+(g,n) - \mathrm{S}_P^-(g,n) \quad \forall\, g,n \tag{4.76}$$

$$\mathrm{V}_P^{\min}(g) \le \mathrm{V}_P^{close}(g,n) \le \mathrm{V}_P^{\max}(g) \qquad \forall\, g,\, n \qquad \text{(4.77) and (4.78)}$$

$$\mathrm{V}_C^{open}(i,n+1) - \mathrm{V}_C^{close}(i,n) = 0 \qquad \forall\, i,n=1,\ldots,N\text{-}1 \tag{4.79}$$

$$\mathrm{V}_P^{open}(g,n+1) - \mathrm{V}_P^{close}(g,n) = 0 \qquad \forall\, g,n=1,\ldots,N\text{-}1 \tag{4.80}$$

The balance around the blender is given by equations (4.81) to (4.86). The blend recipes are fixed within the *l*-period boundaries $\pm 1\times10^{-7}$ in order to avoid finding very small infeasibilities due to numerical discrepancy between the two levels of the algorithm using Eq. (4.81) and (4.82). Eq. (4.12) to (4.15) are included in this model and presented in this section as Eq. (4.83) to (4.86).

$$\mathrm{V}_C(g,i,n,bl) - \left(x(g,i,n) + 1\times10^{-7}\right)\cdot \mathrm{V}_B(g,n,bl) \le 0 \qquad \forall\, g,i,n,bl \tag{4.81}$$

$$\mathrm{V}_C(g,i,n,bl) - \left(x(g,i,n) - 1\times10^{-7}\right)\cdot \mathrm{V}_B(g,n,bl) \ge 0 \qquad \forall\, g,i,n,bl \tag{4.82}$$

$$\mathrm{V}_B^{\min}(g)\cdot A(g,n,bl) - \mathrm{V}_B(g,n,bl) \le 0 \qquad \forall\, g,n,bl \tag{4.83}$$

$$V_B^{\max}(g)\cdot A(g,n,bl) - V_B(g,n,bl) \geq 0 \qquad \forall\ g,n,bl \qquad (4.84)$$

$$\sum_g V_B(g,k,bl) + V_B^{lost}(bl)\cdot \sum_g A(g,n,bl) - F_B^{\max}(bl)\cdot t(n) \leq 0 \qquad \forall\ n,bl \qquad (4.85)$$

$$\sum_g A(g,n,bl) \leq np(bl) \qquad \forall\ n,bl \qquad (4.86)$$

Eq. (4.87a) and (4.87b), link the lower level with the top level by setting the target closing inventories of the *l*-periods that are at the boundary end of the *t*-periods, when using the multi-period formulation and the single-period formulation, respectively.

$$V_P^{close}(g,n) = V_{P,K}^{close}(g,k) \qquad \forall\ g,n \text{ at the end of k} \qquad (4.87a)$$

$$V_P^{close}(g,n) = V_{P,K}^{close}(g) \qquad \forall\ g,n \text{ at the end of K} \qquad (4.87b)$$

Eq. (4.88a) and (4.88b) link the lower level with the top level by providing as parameters the fixed blend recipes to be used in the *l*-periods within the corresponding *t*-period, when using the multi-period formulation and the single-period formulation, respectively.

$$x(i,g,n) = x_K(i,g,k) \qquad \forall\ g,i,n \in k \qquad (4.88a)$$

$$x(i,g,n) = x_K(i,g) \qquad \forall\ g,i,n \in K \qquad (4.88b)$$

The economic cost is minimized at the top level by computing the appropriate blend recipes. Note that the multi-period MILP model does not deal with the blend recipe optimization and does not include the nonlinear terms. Thus, all the nonlinearities intrinsic to the blending problem are solved at the top level.

4.4 Multi-Period Inventory Pinch Algorithm

- Step 1:
  - If the initial product inventory is off-spec, solve the aggregate model given by Eq. (4.55) to (4.71). The length of the quality correction *t*-period K_0 must be enough to bring the product inventories back to spec. If a feasible solution is found, store the blend recipes and volumes to be blended in K_0 and do not consider this part of the planning horizon for the rest of the algorithm. If the problem is infeasible, the blend components supply rates must be modified or investigate if demurrage costs can be accepted.

    - If the initial product inventory is on-spec, go directly to Step 2.
- Step 2: Construct the Cumulative Total Demand (CTD) and the Cumulative Average Total Production (CATP) curves. Determine the pinch point(s) location.
- Step 3: Divide the planning horizon (at the top level) in the number of *t*-periods indicated by the pinch points (i.e. $K_T^{(1)}$). Set iteration counter $it = 1$.
- Step 4: Solve the multi-period model given by Eq. (4.23) to (4.38).
    - In the first iteration ($it = 1$), the volumes to produce of each product in each *t*-period *k* are the amounts that will make the final product inventories to be at the minimum limits.
    - In following iterations ($it > 1$), the volumes to produce are defined according to the solution of the volume allocation problem (Step 5). If necessary, the volumes to blend are adjusted as follows:
        - If the volume to be blended in *t*-period *k* is greater than the maximum blender capacity, the blender at *t*-period *k* needs to work at full capacity and the remaining volume must be blended in the previous *t*-period.
        - If the volume to be blended in *t*-period *k* is less than the minimum allowed, the difference to reach the minimum will be taken from the next *t*-period. The exception is when *k* is the last *t*-period; in this case, the volume must be blended in the previous *t*-period.
- Step 5: Solve the volume allocation problem [equations (4.72) to (4.87)] for the entire horizon as one MILP, which uses fixed recipes computed in Step 4 in the corresponding *l*-periods.
- Step 6: If the inventory slack variables from Step 5 are zero, a feasible set of blend volumes, based on optimal recipes computed at the top level, has been found. Stop, since the optimal production plan has been computed. Otherwise, continue to Step 7.
- Step 7: The planning horizon (at the top level) is divided as follows:
    - If a *t*-period *k* does not contain inventory infeasibilities in the solution from Step 5, it is unchanged.
    - If a *t*-period *k* contains infeasibilities in the solution from Step 5, it is divided into two new time periods; the first new *t*-period ends after the *l*-period *n* where the first infeasibility was detected.
- Step 8: $K_T^{(it+1)} = K_T^{(it)} + 1$. $it = it + 1$. Go back to Step 4.

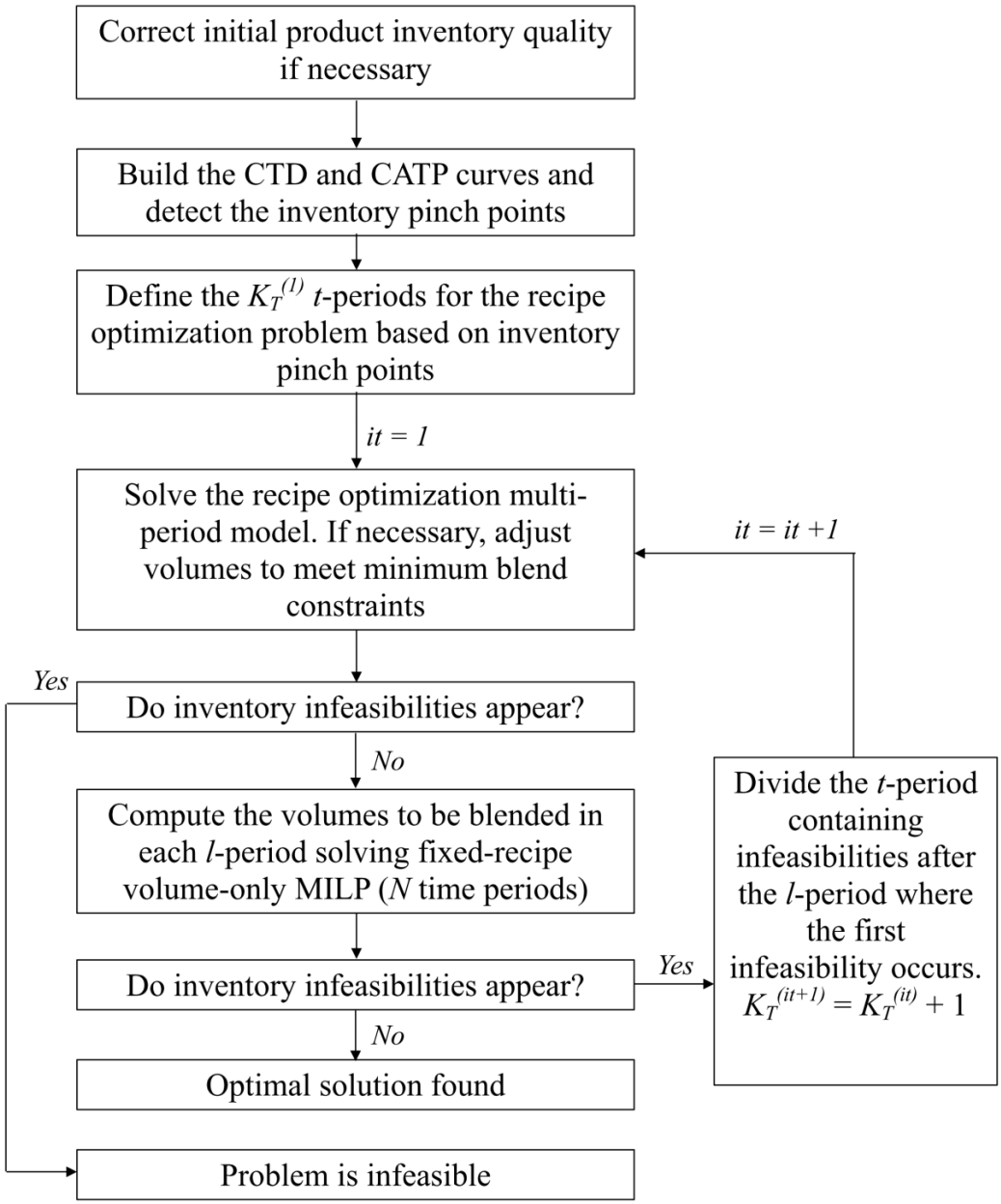


Figure 4.2: Inventory Pinch Multi-Period Algorithm

4.5 Single-Period Inventory Pinch Algorithm

- Step 1:
  - If the initial product inventory is on-spec, solve the aggregate model given by Eq. (4.40) to (4.51) for the whole planning horizon. If a feasible solution is found, continue with Step 2. If the problem is infeasible, the

blend components supply rates must be modified in order to fulfill the demand, i.e. refinery production plan has created infeasible constraints.

- If the initial product inventory is off-spec, solve the aggregate model given by Eq. (4.55) to (4.71). The length of the quality correction *t*-period K_0 must be enough to bring the product inventories back to spec. If the solution is feasible, store the blend recipes and volumes to be blended in K_0 and do not consider this part of the planning horizon for the rest of the algorithm. If the solution is infeasible, the blend components supply rates must be modified or investigate if demurrage costs can be accepted.

- Step 2: Construct the Cumulative Total Demand (CTD) and the Cumulative Average Total Production (CATP) curves. Determine the pinch point(s) location.
- Step 3: Divide the planning horizon (at the top level) in the number of *t*-periods indicated by the pinch points (i.e. $K_T^{(1)}$). Set iteration counter $it = 1$.
- Step 4: Solve each one of these *t*-periods separately as single period problems using equations (4.40) to (4.51), and Eq. (4.52) to (4.54) to link them.
  - In the first iteration ($it = 1$), the volumes to produce of each product in each *t*-period *K* are the amounts that will make the final product inventories to be at the minimum limits.
  - In following iterations ($it > 1$), the volumes to produce are defined according to the solution of the volume allocation problem (Step 5). If necessary, the volumes to blend are adjusted as follows:
    - If the volume to be blended in *t*-period *K* is greater than the maximum blender capacity, the blender at *t*-period *K* needs to work at full capacity and the remaining volume must be blended in the previous *t*-period.
    - If the volume to be blended in *t*-period *K* is less than the minimum allowed, the difference to reach the minimum will be taken from the next *t*-period. The exception is when *K* is the last *t*-period; in this case, the volume must be blended in the previous *t*-period.
- Step 5: Solve the volume allocation problem [equations (4.72) to (4.87)] for the entire horizon as one MILP, which uses fixed recipes computed in Step 4 in the corresponding *l*-periods.
- Step 6: If the inventory slack variables from Step 5 are zero, a feasible set of blend volumes, based on optimal recipes computed at the top level, has been found. Stop, since the optimal production plan has been found. Otherwise, continue to Step 7.
- Step 7: The planning horizon (at the top level) is divided as follows:
  - If a *t*-period *K* does not contain inventory infeasibilities in the solution from Step 5, it is unchanged.
  - If a *t*-period *K* contains infeasibilities in the solution from Step 5, it is divided into two new time periods; the first new *t*-period ends after the *l*-period *n* where the first infeasibility was detected.
- Step 8: $K_T^{(it+1)} = K_T^{(it)} + 1$. $it = it + 1$. Go back to Step 4.

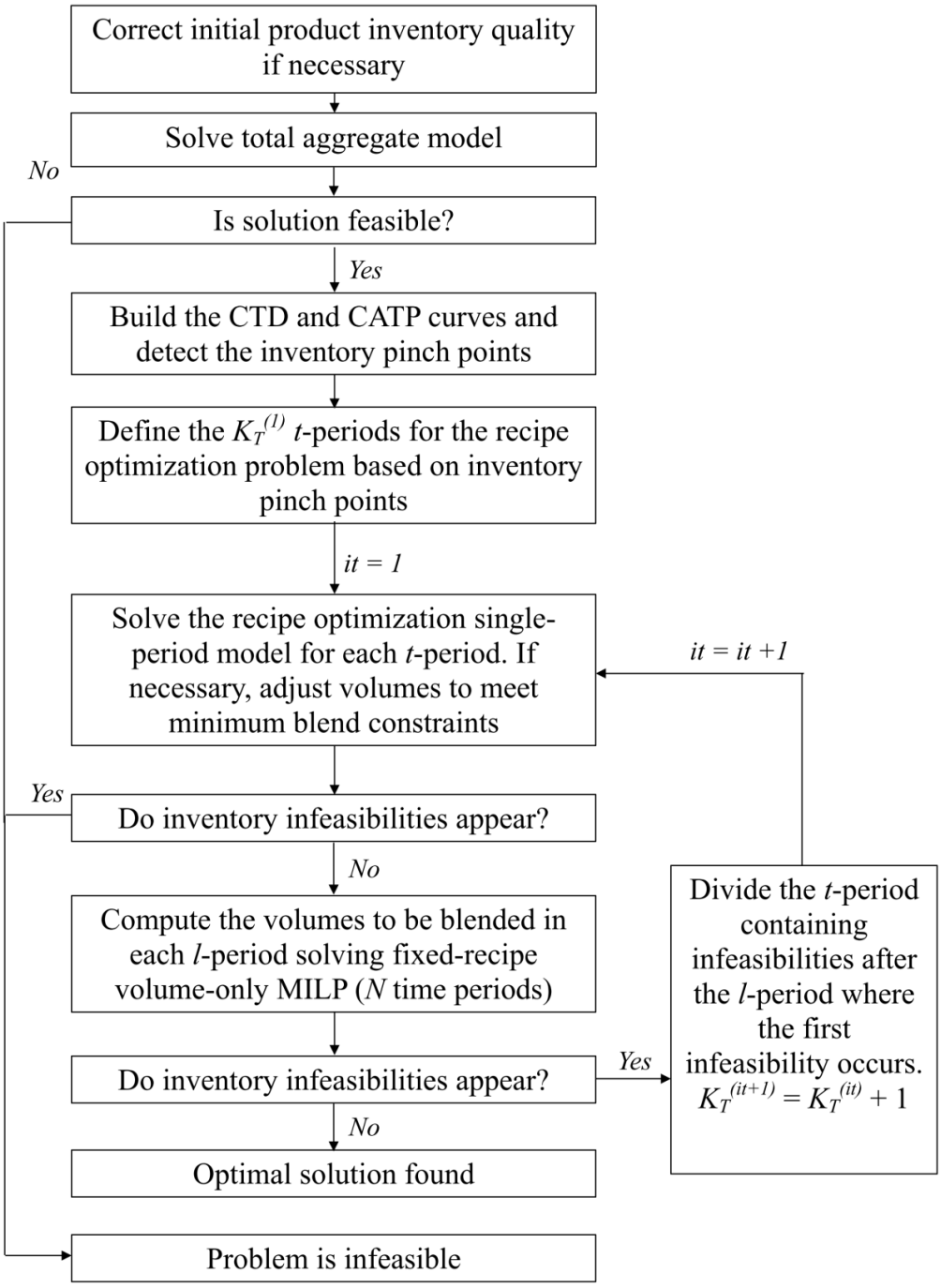


Figure 4.3: Inventory Pinch Single-Period Algorithm

Chapter 5. Numerical Results

The gasoline blending system shown in Figure 1.2, with either one or two blenders, has been studied. Each blender can produce all products (i.e. $np(bl)=3$). In the cases with only one blender, the maximum blend capacity is 175×10^3 BBL/day (case studies 1 to 18) and 200×10^3 BBL/day (case studies 19 to 30); for the two-blender cases, blender A and blender B have a maximum blend capacity of 120×10^3BBL/day and 80×10^3BBL/day, respectively. The minimum blend allowed for one product in each blender (i.e. V_B^{min}) is 30×10^3 BBL/day. For both blenders, the volume lost per switch (i.e. V_B^{lost}) is 8×10^3 BBL. In all cases, the system uses seven blend components (ALK, BUT, HCL, HCN, LCN, LNP, and RFT) and produces three products (grades of gasoline U87, U91, and U93). Each component and each product have their particular storage tank. A planning horizon of 14 one-day *l*-periods was considered for all case studies (H = 14 days, $t(n)$ = 1 day for all n). The demand orders involve a single product and their time windows are assumed to be one day. Eight blend properties are considered: aromatic content (% by volume, ARO), benzene content (% by volume, BEN), olefin content (% by volume, OLF), motor octane number (MON), research octane number (RON), Reid vapour pressure (psi, RVP), specific gravity (SPG), and sulfur content (% by volume, SUL). Table 5.1 contains the blend components data, Table 5.2 shows the supply profile of blend components for case studies 10 to 18, 27, and 28 (uneven supply), and Table 5.3 presents the minimum and maximum specifications for the product properties. Table 5.4 shows the initial quality, initial inventory, and inventory limits of the products. Table 5.5, Table 5.6, and Table 5.7 contain the demand profiles of each case and the cumulative curves for some case studies are shown in Figure 5.1. For the MILP volume allocation problem, the cost coefficients for component inventory slack variables are set to $1x10^{12}$, and the cost coefficients for product inventory slack variables are shown in Table 5.8. Case Study 7 and 30 are explained for a better understanding of the algorithm. All data is presented in English system of units since it prevails in the refining industry.

In order to compute the number of different blend recipes in the MINLP solution, recipes of each blend component were grouped in composition intervals of 1% and 5%. Recipes in the same group were considered to be the same. A blend recipe is considered to be different if at least two components fall in different composition intervals. Since 3 grades of gasoline are being blended, the total number of different blend recipes has been computed by adding the number of different blend recipes for each grade and then dividing it by 3.

All case studies have been computed on a HP Pavilion dv6 Notebook PC, AMD A8-3510MX APU processor, 1.80GHz, Windows 7 OS and 8 GB RAM. GAMS IDE 23.7.3 was used to solve each one of the case studies. The aggregate LP/NLP models were solved using IPOPT, and the MILP model was solved using CPLEX 12.3. To compare the results obtained with the inventory pinch algorithm, the corresponding MINLP model was solved using DICOPT.

Table 5.1: Components data (properties, cost, supply rates and inventory limits)

Components	**ALK**	**BUT**	**HCL**	**HCN**	**LCN**	**LNP**	**RFT**
ARO (%vol aromatics)	0	0	0	25	18	2.974	74.9
BEN (%vol benzene)	0	0	0	0.5	1	0.595	7.5
MON	93.7	90	79.8	75.8	81.6	66	90.8
OLF (%vol olefin)	0	0	0	14	27	0	0
RON	95	93.8	82.3	86.7	93.2	67.8	103
RVP (psi)	5.15	138	22.335	2.378	13.876	19.904	3.622
SPG	0.703	0.584	0.695	0.791	0.744	0.677	0.818
SUL (%vol sulfur)	0	0	0	0.485	0.078	0.013	0
Cost ($/BBL)	29.2	11.5	20	22	25	19.7	24.5
Minimum Inventory ($x10^3$ BBL)	5	5	5	5	5	5	5
Maximum Inventory ($x10^3$ BBL)	150	75	50	50	150	100	150
Initial Inventory ($x10^3$ BBL) Cases 1 – 18	30	20	20	10	30	20	50
Initial Inventory ($x10^3$ BBL) Cases 19 -30	20	20	20	10	50	30	30
Supply Rate ($x10^3$ BBL/day) Cases 1 – 10	18	5	3	5	25	20	44
Supply Rate ($x10^3$ BBL/day) Cases 19 – 26, 29, 30	25	5	3	5	25	20	50

Table 5.2: Supply rate of components along planning horizon, cases 10 – 18, 27, and 28

Case Study	10 - 18							27							28						
Component	ALK	BUT	HCL	HCN	LCN	LNP	RFT	ALK	BUT	HCL	HCN	LCN	LNP	RFT	ALK	BUT	HCL	HCN	LCN	LNP	RFT
l-period	$x10^3$ BBL/day																				
1	25	7	0	3	27	24	45	25	5	0	5	20	30	50	25	5	3	5	25	20	50
2	25	7	0	3	27	24	45	25	5	0	5	25	30	50	25	5	3	5	25	20	50
3	25	7	0	3	27	24	45	30	5	0	5	25	30	25	25	5	3	5	25	20	50
4	20	5	3	5	25	20	40	30	4	3	6	25	20	25	20	4	0	3	30	25	60
5	15	3	7	9	20	25	35	30	4	3	6	30	20	50	20	4	0	3	30	25	60
6	15	3	7	9	20	25	35	25	4	3	6	30	20	50	20	4	0	3	30	25	60
7	15	3	7	9	20	25	35	25	3	3	7	30	20	60	25	5	3	5	25	20	50
8	20	5	3	5	25	20	40	20	5	3	6	30	20	60	25	5	3	5	25	20	50
9	20	5	3	5	25	20	40	20	5	3	5	30	10	60	30	6	6	7	20	15	40
10	25	7	0	3	27	24	45	20	6	4	4	30	10	60	30	6	6	7	20	15	40
11	25	7	0	3	27	24	45	25	6	5	4	25	10	50	30	6	6	7	20	15	40
12	25	7	0	3	27	24	45	25	6	5	4	25	20	50	25	5	3	5	25	20	50
13	20	5	3	5	25	20	40	25	6	5	4	25	20	50	25	5	3	5	25	20	50
14	20	5	3	5	25	20	40	25	6	5	3	0	20	60	25	5	3	5	25	20	50

Table 5.3: Minimum and maximum quality specifications of the products

Specification	Minimum			Maximum		
Product	U87	U91	U93	U87	U91	U93
ARO (%vol aromatics)	0	0	0	60	60	60
BEN (%vol benzene)	0	0	0	5.9	5.9	5.9
MON	81.5	85.7	87.5	-	-	-
OLF (%vol olefin)	0	0	0	24.2	24.2	24.2
RON	91.4	94.5	97.5	-	-	-
RVP (psi)	0	0	0	15.6	15.6	15.6
SPG	0.73	0.73	0.73	0.81	0.81	0.81
SUL (%vol sulfur)	0	0	0	0.1	0.1	0.1

Table 5.4: Products' initial quality, initial inventory, and inventory bounds

Initial Product Quality						
Product	**U87**	**U91**	**U93**	**U87**	**U91**	**U93**
Cases	1 – 28			29 and 30		
ARO (%vol aromatics)	5	10	20	20	40	63
BEN (%vol benzene)	4	4	5	4	5	7
MON	83.2	87	88.2	80	85	86
OLF (%vol olefin)	15	12	18	25	24	26
RON	91.4	95	98.2	90	93	96
RVP (psi)	15	8	12	16	15	17
SPG	0.75	0.76	0.75	0.73	0.82	0.83
SUL (%vol sulfur)	0.05	0.03	0.04	0.11	0.08	0.04
Product	**U87**		**U91**		**U93**	
Initial Product Inventory ($x10^3$ BBL)						
Cases 1 – 18	70		140		30	
Cases 19 – 28	80		180		20	
Case 29	100		100		20	
Case 30	50		50		40	
Minimum Product Inventory ($x10^3$ BBL)						
All cases	10		10		10	
Maximum Product Inventory ($x10^3$ BBL)						
Cases 1 - 25, 26 – 30	300		300		100	
Case 25	800		500		200	

Table 5.5: Demand profiles (x10^3BBL), case studies 1 – 12

Case Study	Product	*l*-periods													
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
1	**U87**	60	50	50	80	50	60	60	50	75	50	50	50	80	100
	U91	50	80	70	30	50	0	40	30	30	50	40	40	30	50
	U93	30	30	0	0	40	40	0	35	30	0	0	40	30	40
2	**U87**	120	80	80	120	0	30	40	50	50	50	50	30	80	100
	U91	50	80	70	50	30	0	30	30	30	50	40	40	30	50
	U93	30	30	45	30	30	0	0	35	30	0	30	40	30	0
3	**U87**	80	80	60	80	80	100	90	0	0	50	50	30	60	100
	U91	50	50	50	30	30	50	50	30	30	50	0	50	60	50
	U93	30	30	35	30	35	0	30	35	30	0	30	40	30	0
4	**U87**	70	70	50	70	70	60	60	60	50	70	120	0	50	70
	U91	50	50	50	30	30	50	50	30	30	50	50	30	30	50
	U93	30	30	45	30	40	0	0	35	30	0	30	35	0	30
5	**U87**	40	50	50	80	80	30	30	50	75	100	120	110	100	30
	U91	50	80	70	30	50	0	30	30	0	50	0	40	50	0
	U93	30	30	45	0	30	40	0	35	30	0	45	0	30	40
6	**U87**	70	50	50	120	100	30	30	50	75	110	50	50	50	90
	U91	50	80	70	30	50	0	0	30	50	50	0	40	30	0
	U93	30	30	45	0	40	40	0	35	30	30	30	0	30	30
7	**U87**	60	50	50	70	90	80	130	50	0	30	50	50	50	80
	U91	50	80	70	50	50	30	30	30	30	30	0	40	30	0
	U93	30	30	45	0	30	40	30	30	30	30	30	0	30	40
8	**U87**	70	50	50	70	50	60	70	50	30	70	90	115	40	30
	U91	50	80	70	50	50	30	30	30	30	30	30	35	30	0
	U93	30	30	30	0	40	0	30	30	30	30	40	30	30	30
9	**U87**	100	70	80	100	40	30	40	110	0	50	70	100	0	50
	U91	50	80	70	50	30	30	30	50	30	30	30	35	30	30
	U93	30	30	45	30	0	30	30	30	30	0	0	30	30	30
10	**U87**	40	50	50	80	50	30	60	50	75	50	50	50	50	70
	U91	50	80	70	30	50	0	40	30	30	50	40	40	30	50
	U93	30	30	0	0	40	40	0	35	30	0	0	40	30	40
11	**U87**	120	80	80	120	0	30	40	50	50	50	50	30	80	100
	U91	50	80	70	50	30	0	30	30	30	50	40	40	30	50
	U93	30	30	45	30	30	0	0	35	30	0	30	40	30	0
12	**U87**	80	80	60	80	80	100	90	0	0	50	50	30	60	100
	U91	50	50	50	30	30	50	50	30	30	50	0	50	60	50
	U93	30	30	35	30	35	0	30	35	30	0	30	40	30	0

Table 5.6: Demand profiles (x10^3BBL), case studies 13 – 24

Case Study	Product	*l*-periods													
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
13	**U87**	70	70	50	70	70	60	60	60	50	70	120	0	50	70
	U91	50	50	50	30	30	50	50	30	30	50	50	30	30	50
	U93	30	30	45	30	40	0	0	35	30	0	30	35	0	30
14	**U87**	40	50	50	80	80	30	30	50	75	100	120	110	100	30
	U91	50	80	70	30	50	0	30	30	0	50	0	40	50	0
	U93	30	30	45	0	30	40	0	35	30	0	45	0	30	40
15	**U87**	70	50	50	120	100	30	30	50	75	110	50	50	50	90
	U91	50	80	70	30	50	0	0	30	50	50	0	40	30	0
	U93	30	30	45	0	40	40	0	35	30	30	30	0	30	30
16	**U87**	60	50	50	70	90	80	130	50	0	30	50	50	50	80
	U91	50	80	70	50	50	30	30	30	30	30	0	40	30	0
	U93	30	30	45	0	30	40	30	30	30	30	30	0	30	40
17	**U87**	70	50	50	70	50	60	70	50	30	70	90	115	40	30
	U91	50	80	70	50	50	30	30	30	30	30	30	35	30	0
	U93	30	30	30	0	40	0	30	30	30	30	40	30	30	30
18	**U87**	100	70	80	100	40	30	40	110	0	50	70	100	0	50
	U91	50	80	70	50	30	30	30	50	30	30	30	35	30	30
	U93	30	30	45	30	0	30	30	30	30	0	0	30	30	30
19	**U87**	30	30	50	50	50	75	100	120	200	220	150	75	50	30
	U91	60	120	80	70	50	0	0	30	0	0	50	0	0	60
	U93	0	0	30	45	0	40	40	0	35	30	30	45	0	20
20	**U87**	50	80	120	150	200	60	80	70	70	50	50	0	30	50
	U91	40	40	40	40	70	50	40	50	40	60	60	0	0	30
	U93	20	20	20	20	40	30	20	20	30	30	30	30	30	30
21	**U87**	50	80	120	150	200	120	50	30	20	20	50	140	150	50
	U91	40	40	40	40	40	40	40	40	40	40	40	40	40	40
	U93	20	20	20	20	20	25	20	20	20	20	20	20	20	20
22	**U87**	50	120	130	200	30	30	50	50	50	60	60	60	80	100
	U91	40	40	50	70	80	0	40	40	40	30	30	30	50	30
	U93	0	50	50	50	40	0	20	20	30	30	0	30	30	30
23	**U87**	50	50	50	150	200	120	50	30	20	50	50	160	200	50
	U91	40	40	40	40	40	40	40	40	40	40	40	40	40	40
	U93	20	20	20	20	20	25	20	20	20	20	20	20	20	20
24	**U87**	30	30	50	75	100	120	200	220	150	75	50	50	50	30
	U91	60	60	80	70	0	50	0	30	0	0	30	0	60	120
	U93	0	0	30	45	0	40	40	0	35	30	0	45	0	20

Table 5.7: Demand profiles (x10^3BBL), case studies 25 – 30

Case Study	Product	*l*-periods													
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
25	**U87**	0	0	0	0	0	720	0	0	0	0	0	0	0	510
	U91	0	0	0	0	0	280	0	0	0	0	0	0	0	280
	U93	0	0	0	0	0	145	0	0	0	0	0	0	0	140
26	**U87**	50	80	120	150	200	120	50	30	20	20	140	170	50	30
	U91	40	40	40	40	40	40	40	40	40	40	40	68	32	20
	U93	20	20	20	20	20	25	20	20	20	20	20	0	40	20
27	**U87**	50	50	50	150	200	120	50	30	20	50	50	160	200	50
	U91	40	40	40	40	40	40	40	40	40	40	40	40	40	40
	U93	20	20	20	20	20	25	20	20	20	20	20	20	20	20
28	**U87**	30	30	50	75	100	120	200	220	150	75	50	50	50	30
	U91	60	60	80	70	0	50	0	30	0	0	30	0	60	120
	U93	0	0	30	45	0	40	40	0	35	30	0	45	0	20
29	**U87**	30	30	50	50	50	75	100	120	200	220	150	75	50	30
	U91	60	120	80	70	50	0	0	30	0	0	50	0	0	60
	U93	0	0	30	45	0	40	40	0	35	30	30	45	0	20
30	**U87**	60	100	120	120	50	80	30	50	60	80	50	60	50	30
	U91	50	40	60	40	40	40	0	0	60	50	50	50	30	20
	U93	30	30	30	30	0	50	50	0	20	30	30	40	40	0

Table 5.8: Cost coefficients profile for the product inventory slack variables, lower level MILP model

l-period	Cost coefficients		
	Case 1 to 26, 29, and 30	**Case 27**	**Case 28**
1	9 x10^8	9 x10^8	9 x10^9
2	8 x10^8	8 x10^8	8.5 x10^9
3	7 x10^8	7 x10^8	8 x10^9
4	6 x10^8	6 x10^8	7.5 x10^8
5	5 x10^8	5 x10^8	7 x10^8
6	1 x10^8	4 x10^8	6.5 x10^8
7	8 x10^7	9 x10^5	6 x10^8
8	7 x10^7	8 x10^3	5.5 x10^8
9	5 x10^6	7 x10^3	5 x10^5
10	1 x10^6	6 x10^3	4.5 x10^5
11	5 x10^4	5 x10^3	4 x10^5
12	1 x10^3	4 x10^3	3.5 x10^5
13	5 x10^1	3 x10^3	3 x10^2
14	1	1	1

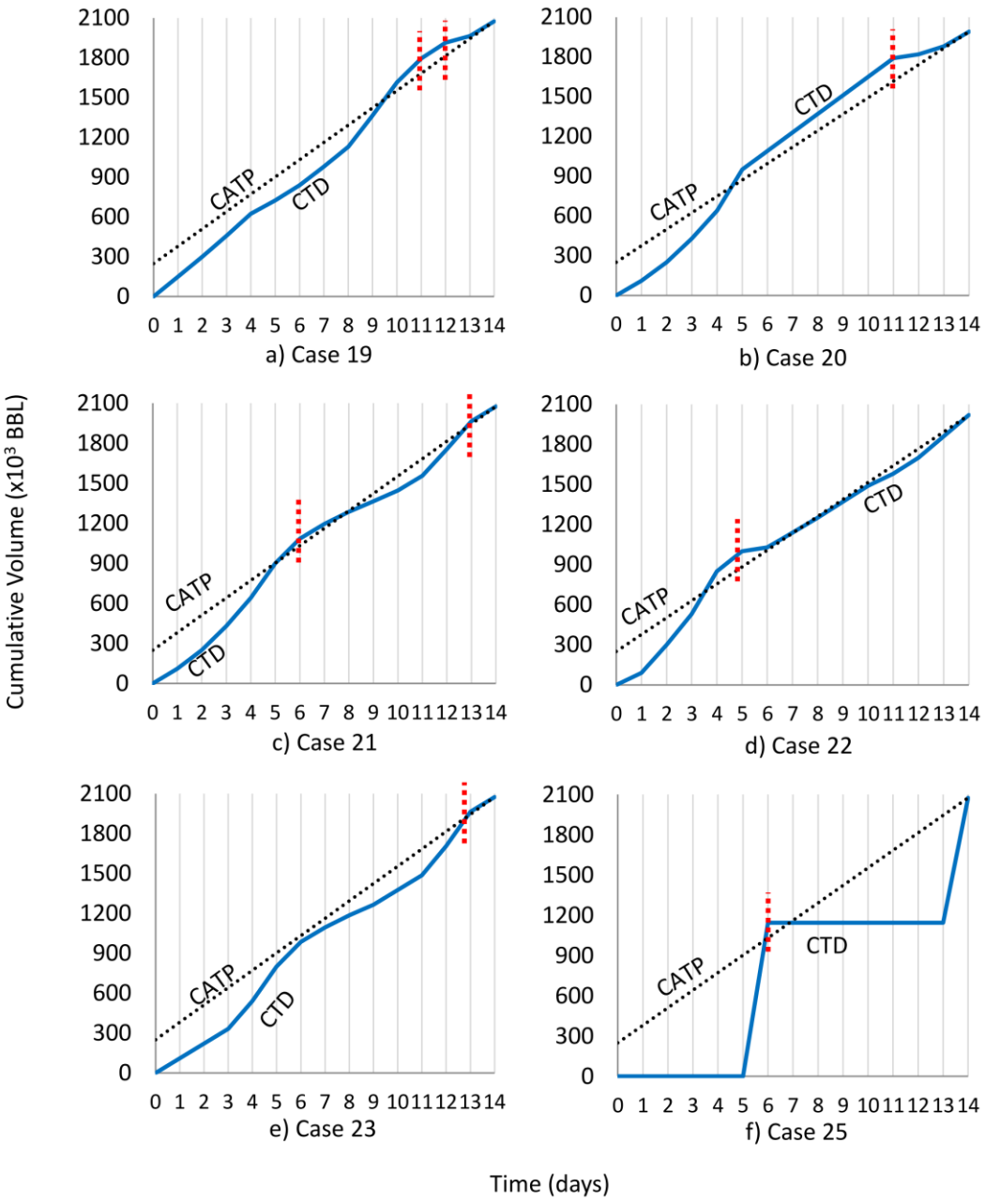


Figure 5.1: Cumulative demand profiles of some case studies. The vertical dashed lines indicate the pinch points.

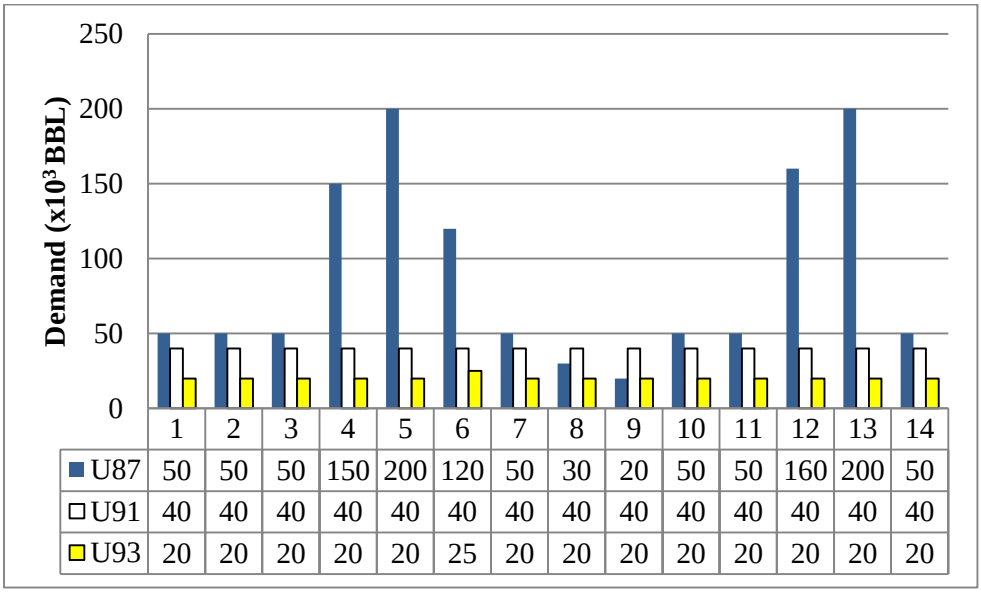
5.1 Example #1: Case study 27 (RVP blends nonlinearly)

The gasoline blending system of case study 27 has only 1 blender, the blend component supply flow rates are irregular along the planning horizon, property RVP blends nonlinearly, and the initial product inventories are on-spec. The inventory pinch algorithm using the multi-period formulation will be shown first and then the algorithm using the single-period formulation.

5.1.1 Multi-Period Inventory Pinch Algorithm: Case Study 27

- **Step 1 and 2:**

Since the initial product inventories are on-spec, the first thing to do is to determine the inventory pinch point(s) occurrence by plotting the Cumulative Total Demand (CTD) and the Cumulative Average Total Production (CATP) curves. Figure 5.2 shows the demand profile of each of the gasoline grades. The CTD and CATP curves are shown in Figure 5.3 and they are constructed using the cumulative data of all gasoline grades.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14
U87	50	50	50	150	200	120	50	30	20	50	50	160	200	50
U91	40	40	40	40	40	40	40	40	40	40	40	40	40	40
U93	20	20	20	20	20	25	20	20	20	20	20	20	20	20

Figure 5.2: Demand profile - Case study 27

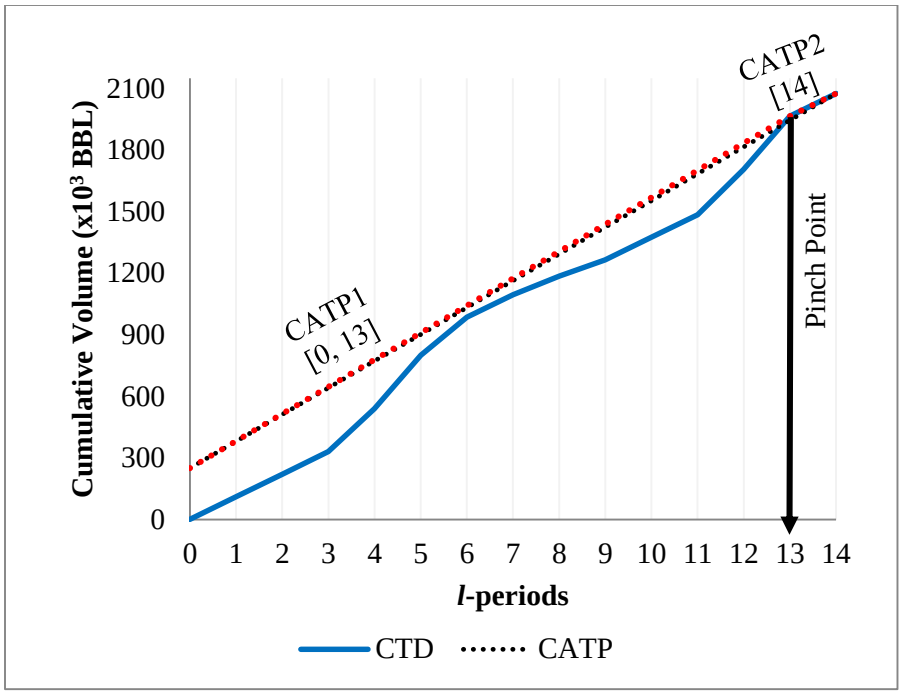


Figure 5.3: Inventory pinch points - Case study 27

- **Step 3:**

Figure 5.3 shows one inventory pinch point at the end of *l*-period 13. In the third step of the algorithm, the planning horizon is divided according to this pinch point: *t*-period $k^{(1)}$=1 goes from *l*-period *n*=1 to *n*=13, *t*-period $k^{(1)}$=2 contains only *l*-period *n*=14 (the superscript "*(1)*" indicates that these *t*-periods correspond to the first iteration of the algorithm). Therefore, $K_T^{(1)} = 2$.

- **Step 4 (*it*=1):**

In Step 4, the multi-period model given by Eq. (4.23) to (4.38) is solved to find the optimal blend recipes. Because this is the first iteration, the volumes to blend at each *t*-period are set in order to be at the minimum product closing inventories (i.e. 10×10^3BBL) at the boundaries (see Table 5.9). The solution found has a cost equal to $\$43{,}627.51\times10^3$ and the blend recipes appear in Table 5.11.

Table 5.9: Volumes to blend (10^3BBL), case study 27, multi-period algorithm, *it* = 1

***t*-period**		***k* = 1**	***k* = 2**
Product	**U87**	1110	50
	U91	350	40
	U93	275	0
Total		1735	90

- **Step 5 (*it*=1):**

In Step 5, the volume allocation problem (i.e. Eq. (4.72) to (4.87)) is solved. In this model, the blend recipes computed previously in Step 4 are fixed in the corresponding *l*-periods.

- **Step 6 (*it*=1):**

The solution contains infeasibilities (see Figure 5.4); therefore, it is necessary to divide the first *t*-period.

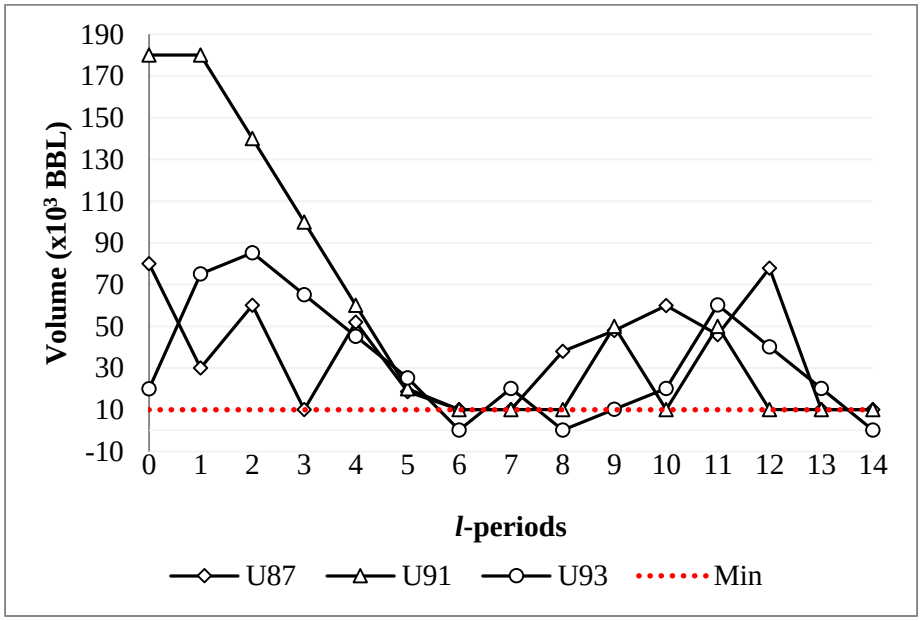


Figure 5.4: Product inventory profile – Case study 27, multi-period algorithm, *it*=1

- **Step 7 (*it*=1):**

The first infeasibility (equal to 9.82×10^3BBL) appears at *l*-period n=6 on the U93 inventory; thus, *t*-period $k^{(1)}$=1 is divided into two new periods: new *t*-period $k^{(2)}$=1 goes from *l*-period n=1 to n=6, and new *t*-period $k^{(2)}$=2 goes from *l*-period n=7 to n=13. *t*-period $k^{(2)}$=3 contains *l*-period n=14. No volume adjustments are required since there is enough capacity at $k^{(2)}$=1 and $k^{(2)}$=2 to blend the volumes required to overcome the infeasibility (see Table 5.10).

Table 5.10: Volumes to blend (10^3BBL), case study 27, multi-period algorithm, *it* = 2

***t*-period**		***k* = 1**	***k* = 2**	***k* = 3**
Product	**U87**	550	560	50
	U91	70	280	40
	U93	115	160	0
Total		735	1000	90

- **Step 8 (*it*=1):**

  $K_T^{(2)}$ = 2+1=3. *it*=1+1=2. Go back to Step 4.

- **Step 4 (*it*=2):**

The blend recipes computed for the *t*-periods of the 2nd iteration are shown in Table 5.12. The blend cost is \$43,627.49×10^3.

- **Step 5 (*it*=2):**

The volume allocation problem is solved with the new blend recipes and this time no inventory infeasibilities are present. Figure 5.5 show the product inventory profiles.

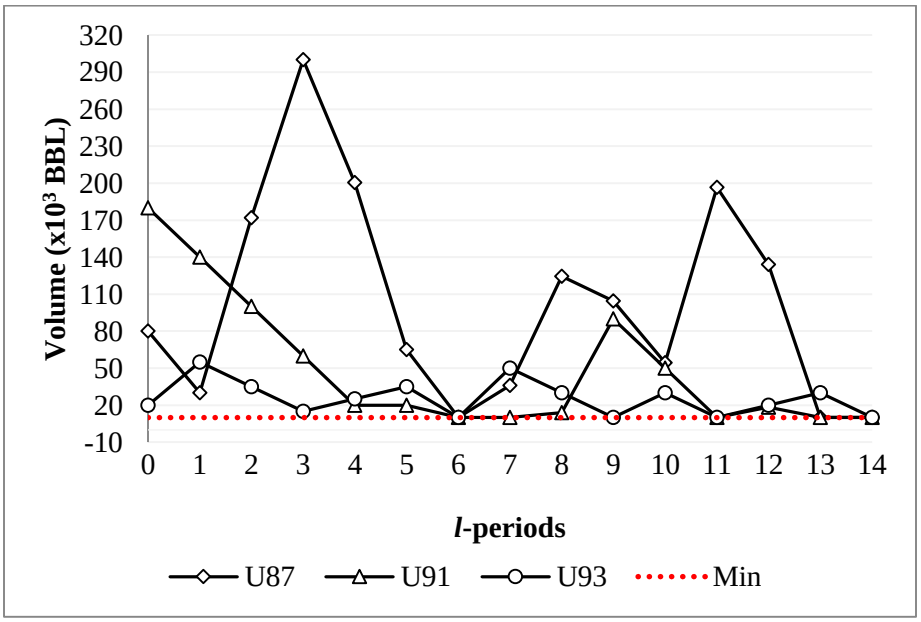


Figure 5.5: Product inventory profile – Case study 27, multi-period algorithm, *it*=2

Blend component inventory profiles are within the limits and they are not shown here.

- **Step 6 (*it*=2):**

Since there are no inventory infeasibilities, this is an optimal solution and the algorithm stops. The blend cost is equal to $\$43{,}627.56\times10^3$; which is only 0.0007% higher than the solution from the corresponding MINLP model. The blend plan is shown in Table 5.11.

Table 5.11: Blend plan for case study 27, multi-period algorithm (volume in 10^3BBL)

Blender	Product	*l*-periods													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	U87		192	178	51	65	65	76	118			192	98	76	50
	U91					40	30	40	44	116			48	32	40
	U93	55			30	30		60			40		30	30	
	Total	55	192	178	81	135	95	176	162	116	40	192	176	138	90

Table 5.12: Blend recipes computed for case study 27, multi-period algorithm

Iteration	it = 1								
t-period	k = 1						k = 2		
Product	U87		U91		U93		U87	U91	U93
ALK	0.1095		0.2294		0.1743		0.0000	0.0000	0.1512
BUT	0.0250		0.0360		0.0436		0.0326	0.0321	0.0324
HCL	0.0240		0.0475		0.0316		0.0000	0.1250	0.0341
HCN	0.0305		0.0673		0.0532		0.0600	0.0000	0.0678
LCN	0.2640		0.1851		0.1350		0.0000	0.0000	0.1072
LNP	0.1964		0.0693		0.0244		0.2932	0.1492	0.0191
RFT	0.3506		0.3654		0.5378		0.6141	0.6937	0.5882
Iteration	**it = 2**								
***t*-period**	**k = 1**			**k = 2**			**k = 3**		
Product	**U87**	**U91**	**U93**	**U87**	**U91**	**U93**	**U87**	**U91**	**U93**
ALK	0.1708	0.2436	0.1874	0.0665	0.1951	0.1582	0.0000	0.0000	0.1512
BUT	0.0259	0.0371	0.0439	0.0244	0.0351	0.0433	0.0326	0.0321	0.0324
HCL	0.0148	0.0518	0.0276	0.0318	0.0491	0.0346	0.0000	0.1250	0.0341
HCN	0.0240	0.0852	0.0519	0.0383	0.0592	0.0552	0.0600	0.0000	0.0678
LCN	0.2649	0.1693	0.1361	0.2628	0.1888	0.1358	0.0000	0.0000	0.1072
LNP	0.1904	0.0594	0.0241	0.1985	0.0791	0.0253	0.2932	0.1492	0.0191
RFT	0.3092	0.3536	0.5291	0.3777	0.3936	0.5476	0.6141	0.6937	0.5882

5.1.2 Single-Period Inventory Pinch Algorithm: Case Study 27

- **Step 1, 2, and 3:**

These are the same as in the multi-period example. In this case, instead of using *k*, the notation for the *t*-periods changes to *K* since they are aggregate models; therefore, *t*-period $K^{(1)}$=1 goes from *l*-period *n*=1 to *n*=13, *t*-period $K^{(1)}$=2 contains only *l*-period *n*=14

- **Step 4 (*it*=1):**

In Step 4, the single-period model given by Eq. (4.40) to (4.51) is solved for each t-period. The models are solved in sequence since the results from a *t*-period (blend component closing inventories) are required as parameters for the next *t*-period (blend component opening inventories). Because this is the first iteration, the volumes to blend at each *t*-period are set in order to be at the minimum product closing inventories (i.e. 10×10^3BBL) at the boundaries (see Table 5.9). The solution found has a cost equal to $\$43{,}627.51\times10^3$ and the blend recipes are shown in Table 5.17.

- **Step 5 (*it*=1):**

In Step 5, the volume allocation problem (i.e. Eq. (4.72) to (4.87)) is solved. As with the multi-period algorithm, the blend recipes computed previously in Step 4 are fixed in the corresponding *l*-periods.

- **Step 6 (*it*=1):**

The solution contains infeasibilities (see Figure 5.6); therefore, it is necessary to divide a *t*-period.

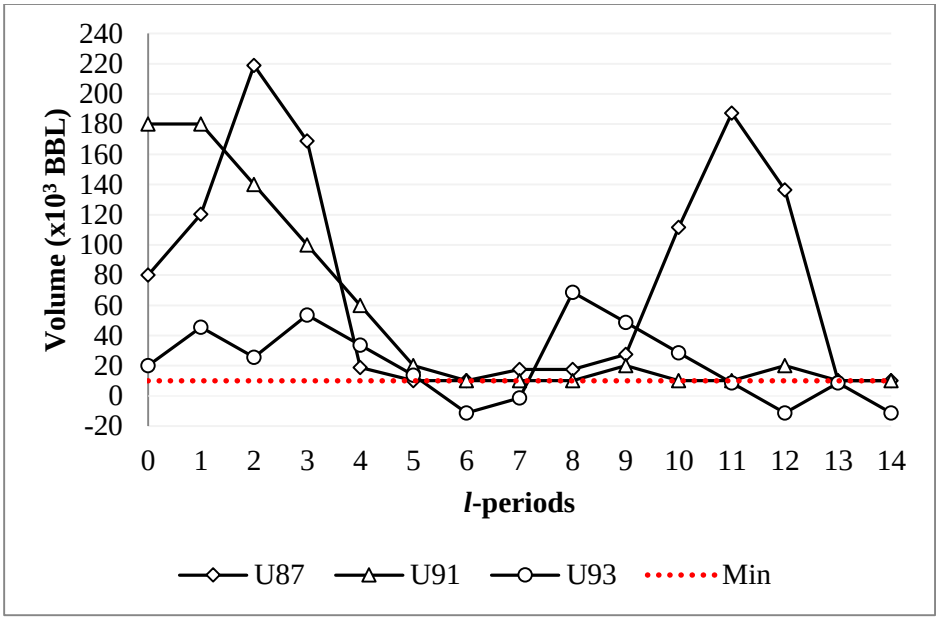


Figure 5.6: Products inventory profile – Case study 27, single-period algorithm, *it*=1

- **Step 7 (*it*=1):**

The first infeasibility (equal to 21.36×10^3BBL) appears at *l*-period *n*=6 on the U93 inventory; thus, *t*-period $K^{(1)}$=1 is divided into two new periods: new *t*-period $K^{(2)}$=1 goes from *l*-period *n*=1 to *n*=6, and new *t*-period $K^{(2)}$=2 goes from *l*-period *n*=7 to *n*=13. *t*-period $K^{(2)}$=3 contains *l*-period *n*=14. No volume adjustments are required since there is enough capacity at $K^{(2)}$=1 and $K^{(2)}$=2 to blend the volumes required to overcome the infeasibility (see Table 5.10).

- **Step 8 (*it*=1):**

$K_T^{(2)}$ = 2+1=3. *it*=1+1=2. Go back to Step 4.

- **Step 4 (*it*=2):**

The blend recipes computed for the *t*-periods of the 2nd iteration are shown in Table 5.17. The blend cost is $\$43,627.49\times10^3$.

- **Step 5 and 6 (*it*=2):**

The volume allocation problem (i.e. MILP model) is solved with the new blend recipes. Inventory infeasibilities appear again (see Figure 5.7).

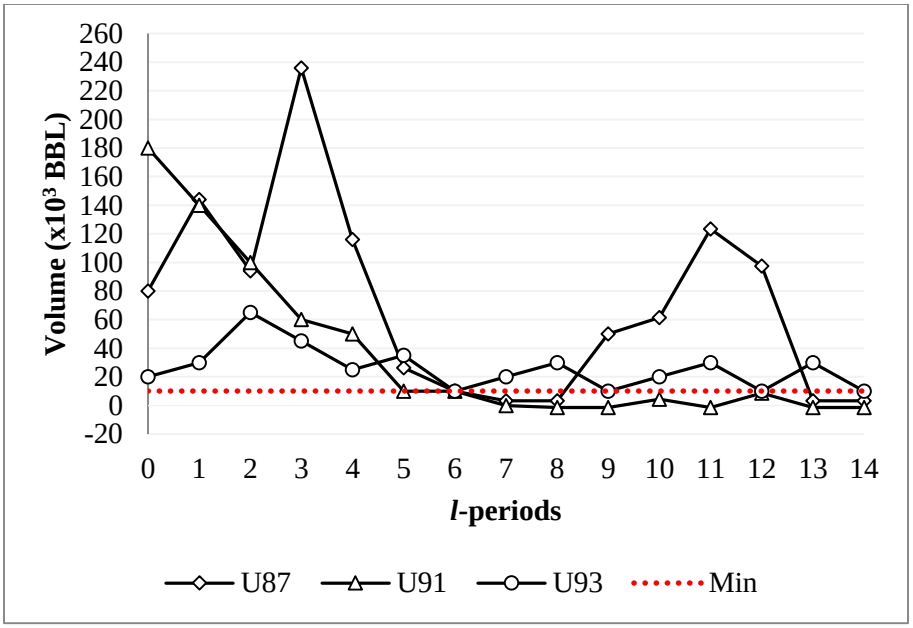


Figure 5.7: Products inventory profile – Case study 27, single-period algorithm, *it*=2

- **Step 7 (*it*=2):**

The first infeasibilities appear at n=7, they are equal to 6.64×10^{3}BBL on the U87 inventory and 10×10^{3}BBL on the U91 inventory. Hence, t-period $K^{(2)}$=2 is divided into two new periods: new t-period $K^{(3)}$=2 contains l-period n=7, and new t-period $K^{(3)}$=3 goes from l-period n=8 to n=13. t-period $K^{(3)}$=4 contains l-period n=14. t-period $K^{(3)}$=1 is the same as $K^{(2)}$=1 and there is no need to re-compute the blend recipe for that interval. No volume adjustments are required since there is enough capacity at $K^{(2)}$=1 and $K^{(2)}$=2 to blend the volumes required to overcome the infeasibility (see Table 5.13).

Table 5.13: Volumes to blend (10^{3}BBL), case study 27, single-period algorithm, *it* = 3

***t*-period**		***K* = 1**	***K* = 2**	***K* = 3**	***K* = 4**
Product	**U87**	550	50	510	50
	U91	70	40	240	40
	U93	115	30	130	0
Total		735	120	880	90

- **Step 8 (*it*=2):**

$K_T^{(3)}$ = 3+1=4. *it*=2+1=3. Go back to Step 4.

- **Step 4 (*it*=3):**

The blend recipes computed for the new *t*-periods of the 3rd iteration provide a blend cost of $\$43{,}627.47\times10^3$.

- **Step 5 and 6 (*it*=3):**

After solving the volume allocation problem, inventory infeasibilities still appear (see Figure 5.8).

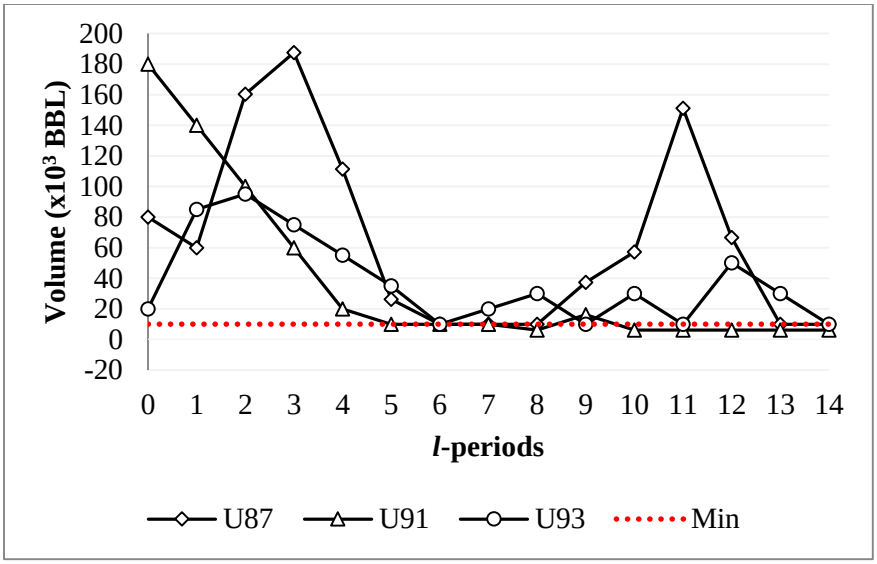


Figure 5.8: Products inventory profile – Case study 27, single-period algorithm, *it*=3

- **Step 7 (*it*=3):**

The first infeasibility appears at *n*=8 (3.79×10^3BBL on the U91 inventory). Thus, *t*-period $K^{(3)}$=3 is divided into two new periods: new *t*-period $K^{(4)}$=3 contains *l*-period *n*=8, and new *t*-period $K^{(4)}$=4 goes from *l*-period *n*=9 to *n*=13. *t*-period $K^{(4)}$=5 contains *l*-period *n*=14. *t*-period $K^{(4)}$=2 is the same as $K^{(3)}$=2 and there is no need to re-compute the blend recipe for that interval. No volume adjustments are required since there is enough capacity at $K^{(4)}$=3 and $K^{(4)}$=4 (see Table 5.14).

Table 5.14: Volumes to blend (10^3BBL), case study 27, single-period algorithm, *it* = 4

***t*-period**		***K* = 1**	***K* = 2**	***K* = 3**	***K* = 4**	***K* = 5**
Product	**U87**	550	50	30	480	50
	U91	70	40	40	200	40
	U93	115	30	30	100	0
Total		735	120	100	780	90

- **Step 8 (*it*=3):**

  $K_T^{(4)}$ = 4+1=5. *it*=3+1=4. Go back to Step 4.

- **Step 4, 5, 6, and 7 (*it*=4):**

The blend recipes for the new *t*-periods of the 4th iteration are computed and the blend cost is equal to $\$43{,}627.45\times10^3$. The MILP model is solved and one inventory infeasibility appear at *l*-period *n*=13 (3.34×10^3BBL on the U93 inventory). Therefore, *t*-period $K^{(4)}$=4 is divided into two new periods: new *t*-period $K^{(5)}$=4 goes from *l*-period *n*=9 to *n*=12, and new *t*-period $K^{(5)}$=5 contains *l*-period *n*=13. *t*-period $K^{(5)}$=6 contains *l*-period *n*=14. *t*-period $K^{(5)}$=3 is the same as $K^{(4)}$=3 and there is no need to re-optimize the blend recipe for that interval. A volume adjustment is required since there is no more volume of U93 being blended after *l*-period *n*=13, and since the volume required to blend is less than the minimum threshold (30×10^3BBL), it means that the inventory infeasibility should be blended in the previous *t*-period (see Table 5.15).

Table 5.15: Volumes to blend (10^3BBL), case study 27, single-period algorithm, *it* = 5

***t*-period**		***K* = 1**	***K* = 2**	***K* = 3**	***K* = 4**	***K* = 5**	***K* = 6**
Product	**U87**	550	50	30	336	144	50
	U91	70	40	40	160	40	40
	U93	115	30	30	100	0	0
Total		735	120	100	596	184	90

- **Step 8 (*it*=4):**

  $K_T^{(5)}$ = 5+1=6. *it*=4+1=5. Go back to Step 4.

- **Step 4, 5, 6, and 7 (*it*=5):**

The blend recipes computed for the new *t*-periods of the 5th iteration are shown in Table 5.17. The blend cost is $\$43{,}627.48\times10^3$. Inventory infeasibilities do not appear anymore in the solution of the MILP model (Figure 5.9); therefore, this is an optimal solution and the algorithm stops. The blend cost is equal to $\$43{,}627.45\times10^3$; which is only 0.0004% higher than the solution from the corresponding MINLP model. The blend plan is shown in Table 5.16.

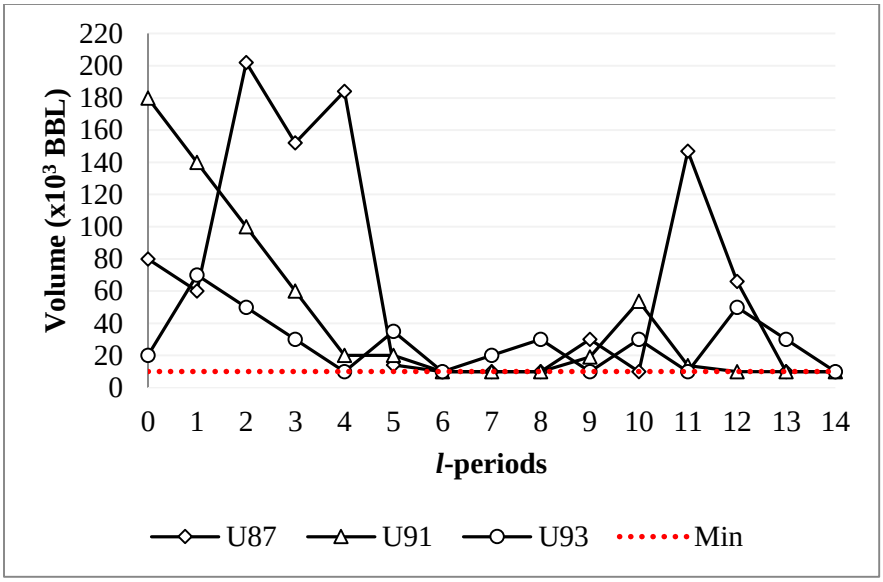


Figure 5.9: Products inventory profile – Case study 27, single-period algorithm, *it*=5

Table 5.16: Blend plan for case study 27, single-period algorithm (volume in 10^3BBL)

Blender	Product	*l*-periods													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	U87	30	192		182	30	116	50	30	40	30	187	79	144	50
	U91					40	30	40	40	49	75		36	40	40
	U93	70				45		30	30		40		60		
	Total	100	192	0	182	115	146	120	100	89	145	187	175	184	90

Table 5.17: Blend recipes for case study 27, single-period algorithm (volume in 10^3BBL)

Iteration	**it = 1**																					
***t*-period**	**K = 1**												**K = 2**									
Product	**U87**			**U91**			**U93**			**U87**			**U91**			**U93**					
ALK	0.0729			0.3173			0.2099			0.0000			0.0000			0.1690					
BUT	0.0243			0.0375			0.0445			0.0326			0.0321			0.0281					
HCL	0.0259			0.0435			0.0291			0.0000			0.1250			0.0247					
HCN	0.0320			0.0632			0.0522			0.0600			0.0000			0.0649					
LCN	0.2725			0.1682			0.1222			0.0000			0.0000			0.1217					
LNP	0.2007			0.0579			0.0216			0.2932			0.1492			0.0134					
RFT	0.3716			0.3125			0.5205			0.6141			0.6937			0.5781					
Iteration	**it = 2**																					
***t*-period**	**K = 1**			**K = 2**									**K = 3**									
Product	**U87**	**U91**	**U93**	**U87**			**U91**			**U93**			**U87**			**U91**			**U93**		
ALK	0.0657	0.2193	0.1609	0.1478			0.2412			0.1839			0.0000			0.0000			0.1690		
BUT	0.0235	0.0368	0.0437	0.0265			0.0356			0.0435			0.0326			0.0321			0.0281		
HCL	0.0280	0.0665	0.0342	0.0232			0.0363			0.0301			0.0000			0.1250			0.0247		
HCN	0.0446	0.0929	0.0604	0.0250			0.0445			0.0470			0.0600			0.0000			0.0649		
LCN	0.3163	0.1565	0.1309	0.2137			0.1902			0.1380			0.0000			0.0000			0.1217		
LNP	0.1832	0.0563	0.0237	0.2043			0.0825			0.0252			0.2932			0.1492			0.0134		
RFT	0.3385	0.3717	0.5461	0.3594			0.3698			0.5324			0.6141			0.6937			0.5781		
Iteration	**it = 5**																					
***t*-period**	**K = 1**			**K = 2**			**K = 3**			**K = 4**			**K = 5**			**K = 6**					
Product	**U87**	**U91**	**U93**	**U87**	**U91**	**U93**	**U87**	**U91**	**U93**	**U87**	**U91**	**U93**	**U87**	**U91**	**U93**	**U87**	**U91**	**U93**				
ALK	0.0657	0.2193	0.1609	0.0036	0.0829	0.0522	0.0053	0.0818	0.0501	0.1453	0.2353	0.1822	0.3510	0.3624	0.1690	0.0000	0.0000	0.1690				
BUT	0.0235	0.0368	0.0437	0.0234	0.0367	0.0432	0.0317	0.0367	0.0433	0.0248	0.0347	0.0431	0.0309	0.0372	0.0281	0.0326	0.0321	0.0281				
HCL	0.0280	0.0665	0.0342	0.0075	0.0378	0.0371	0.0117	0.0380	0.0376	0.0233	0.0383	0.0306	0.0224	0.0443	0.0247	0.0000	0.1250	0.0247				
HCN	0.0446	0.0929	0.0604	0.0988	0.0246	0.0359	0.1373	0.0223	0.0330	0.0187	0.0400	0.0433	0.0158	0.0431	0.0649	0.0600	0.0000	0.0649				
LCN	0.3163	0.1565	0.1309	0.4228	0.0332	0.0534	0.1446	0.0257	0.0405	0.2717	0.2100	0.1447	0.1271	0.1673	0.1217	0.0000	0.0000	0.1217				
LNP	0.1832	0.0563	0.0237	0.1548	0.1702	0.0798	0.2093	0.1735	0.0849	0.1921	0.0794	0.0253	0.1858	0.0568	0.0134	0.2932	0.1492	0.0134				
RFT	0.3385	0.3717	0.5461	0.2892	0.6147	0.6984	0.4600	0.6220	0.7106	0.3242	0.3623	0.5309	0.2670	0.2889	0.5781	0.6141	0.6937	0.5781				

5.2 Example #2: Case study 30 (2 blenders)

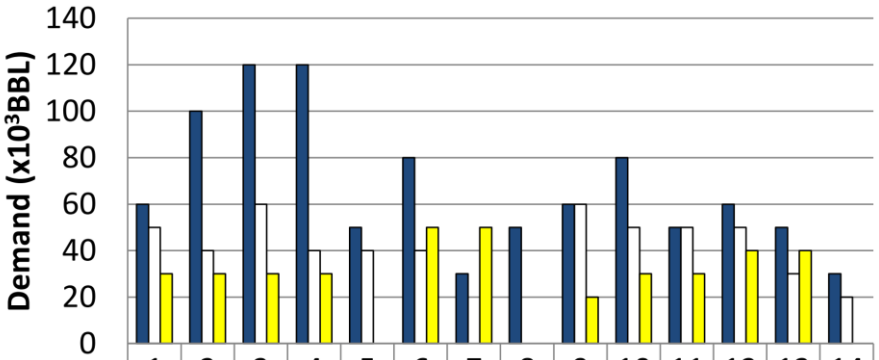
The gasoline blending system of case study 30 has 2 blenders, the blend component supply flow rates are constant along the planning horizon, property RVP is assumed to blend linearly, and the initial product inventories are off-spec. The inventory pinch algorithm using the single-period formulation will be shown.

- **Step 1:**

Since the initial product inventories are off-spec and quality properties are assumed to blend linearly, the model given by Eq. (4.55) to (4.71) is used to bring the inventories back to spec. Initially, *t*-period K_0 will be defined as *l*-period *n*=1. If this time period is not enough to correct the tank heel, it will be increased and a demurrage cost will be included in the solution. In this case however, the model provides a feasible solution (*l*-period *n*=1 is enough to bring the inventories back to spec) with objective function equal to $\$41{,}211.25 \times 10^3$. The blend recipe for the heel correction is shown in Table 5.19.

- **Step 2:**

Figure 5.10 shows the demand profile for this example. The inventory pinch points are identified to be at the end of *l*-periods *n*=4, *n*=6, and *n*=13, as shown in Figure 5.11. Since the tank heel is corrected in *n*=1, this *l*-period is no longer considered for the rest of the algorithm.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14
U87	60	100	120	120	50	80	30	50	60	80	50	60	50	30
U91	50	40	60	40	40	40	0	0	60	50	50	50	30	20
U93	30	30	30	30	0	50	50	0	20	30	30	40	40	0

Figure 5.10: Demand profile - Case study 30

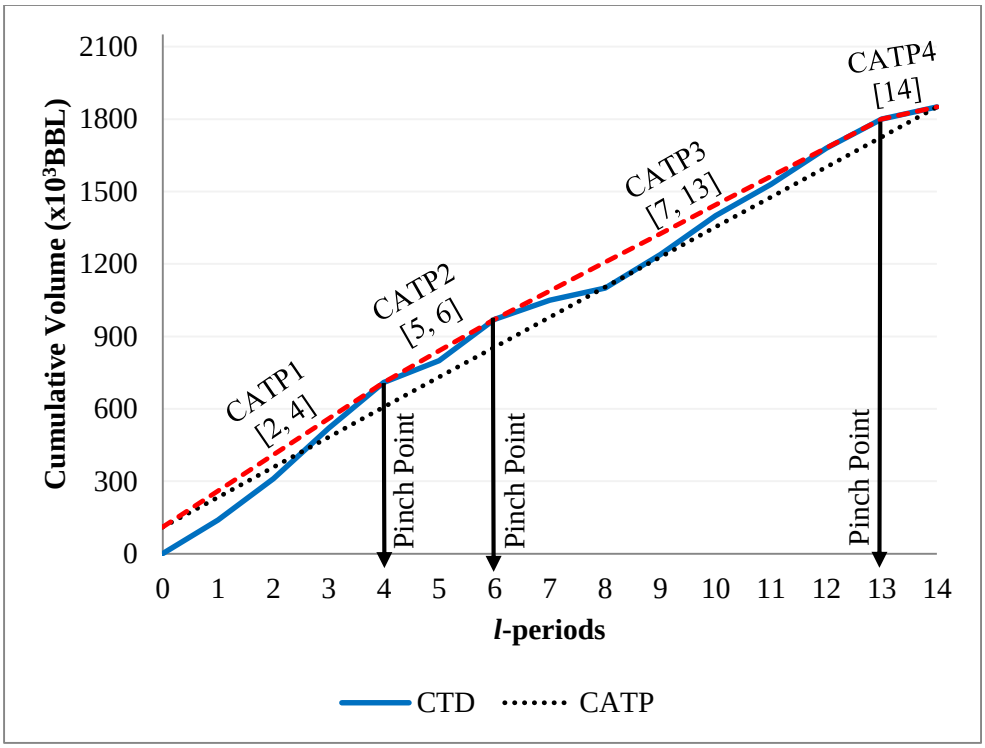


Figure 5.11: Inventory pinch points - Case study 30

- **Step 3:**

The planning horizon is divided according to the pinch points: *t*-period $K^{(1)}$=1 goes from *l*-period *n*=2 to *n*=4, *t*-period $K^{(1)}$=2 goes from *l*-period *n*=5 to *n*=6, *t*-period $K^{(1)}$=3 goes from *l*-period *n*=7 to *n*=13, and *t*-period $K^{(1)}$=4 consists of *l*-period *n*=14. Therefore, $K_T^{(1)} = 4$.

- **Step 4 (*it*=1):**

The model given by Eq. (4.40) to (4.48) is solved for each *t*-period. Because this is the first iteration, the volumes to be blended are chosen in such a way that the product closing inventories will be at the minimum allowed (i.e. 10×10^3BBL), except for *t*-period $K^{(1)}$=3 where the closing inventory of U91 at *n*=13 is set to 30×10^3BBL in order to avoid blending 30×10^3BBL (i.e. the minimum allowed) at *l*-period *n*=14, where the demand is only 20×10^3BBL, thus avoiding blending more than required. The blend recipes computed are shown in Table 5.19.

- **Step 5 and 6 (*it*=1):**

The MILP model (i.e. Eq. (4.72) to (4.87)) is solved. The solution does not contain infeasibilities; therefore, this is an optimal solution and the algorithm stops. Figure 5.12 shows the inventory profiles of the products. The blend cost is equal to

$41,470.74×10^3, which is 0.0007% higher than the solution from the corresponding MINLP model. The blend plan is shown in Table 5.18.

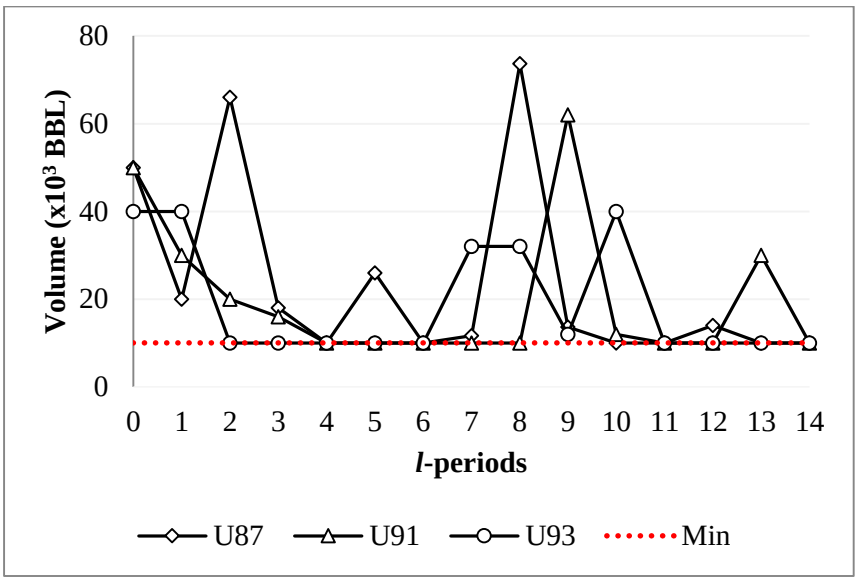


Figure 5.12: Products inventory profile – Case study 30, single-period algorithm, *it*=1

Table 5.18: Blend plan for case study 30, single-period algorithm (volume in 10^3BBL)

Blender	Product	*l*-periods													
		1	2	3	4	5	6	7	8	9	10	11	12	13	14
A	U87	30.0	74.0		112.0	66.0	64.0	31.7	112.0		76.3		64.0	46.0	30.0
	U91	30.0	30.0	56.0			40.0			112.0		48.0		50.0	
	U93	30.0		30.0									40.0		
B	U87		72.0	72.0								50.0			
	U91				34.0	40.0							50.0		
	U93				30.0		50.0	72.0			58.0			40.0	

Table 5.19: Blend recipes for case study 30, single-period algorithm (volume in 10^3BBL)

***t*-periods**	**[K_0]**[a]	**[$K_1^{(1)}$]**	**[$K_2^{(1)}$]**	**[$K_3^{(1)}$]**	**[$K_4^{(1)}$]**
***l*-periods**	**[n=1]**	**[n=2 to 4]**	**[n=5 o 6]**	**[n=7 to 13]**	**[n=14]**
Product	**U87**				
ALK	0.0000	0.1061	0.1509	0.1011	0.0000
BUT	0.0445	0.0377	0.0445	0.0342	0.0550
HCL	0.2959	0.0224	0.0191	0.0220	0.1000
HCN	0.1676	0.0193	0.0311	0.0364	0.1667
LCN	0.0000	0.3323	0.2172	0.2437	0.0000
LNP	0.0000	0.1810	0.1975	0.2000	0.1792
RFT	0.4920	0.3013	0.3396	0.3626	0.4991
Product	**U91**				
ALK	0.0000	0.2147	0.2281	0.2086	0.0000
BUT	0.0809	0.0562	0.0585	0.0465	0.0000
HCL	0.1234	0.0427	0.0278	0.0288	0.0000
HCN	0.1657	0.0472	0.0456	0.0469	0.0000
LCN	0.0000	0.2142	0.1870	0.1967	0.0000
LNP	0.0000	0.0713	0.0848	0.0885	0.0000
RFT	0.6300	0.3535	0.3682	0.3841	0.0000
Product	**U93**				
ALK	0.2964	0.1577	0.1684	0.1644	0.0000
BUT	0.0680	0.0703	0.0706	0.0558	0.0000
HCL	0.0000	0.0319	0.0259	0.0244	0.0000
HCN	0.0000	0.0494	0.0462	0.0427	0.0000
LCN	0.0514	0.1349	0.1359	0.1488	0.0000
LNP	0.0000	0.0228	0.0250	0.0287	0.0000
RFT	0.5843	0.5329	0.5279	0.5350	0.0000

[a] Blend recipe to correct the heel

5.3 Discussion

The costs of the blend plans computed by inventory pinch algorithms are very close to the optimal solutions computed from the corresponding fine-grid multi-period MINLP model (see Table 5.20 and 5.21). All solutions from multi-period inventory pinch (MPIP) algorithm are less than 0.001% higher than those provided by the corresponding MINLP model. The difference between the single-period inventory pinch (SPIP) algorithm and MINLP results is less than 0.01% when the supply flow rates of blend components are constant along the planning horizon (case studies 6 and 7 are the exception). When blend component supply rates vary along the horizon usually more inventory infeasibilities appear at the lower level, especially when using the single-period

algorithm, more iterations are required, and the costs computed via single-period inventory pinch algorithm can be higher than those computed via multi-period MINLP by a fractional percentage. For SPIP algorithm, the maximum difference observed in our case studies is 1% (case study 16). The higher the number of periods at the top level, the more likely is that solution of SPIP algorithm will be suboptimal since it does not consider simultaneously all periods; for instance, the algorithm tries to minimize use of expensive materials at each *t*-period, but the optimal solution for a time interval is not necessarily part of the global optimal solution. Suboptimal solutions are found as well when the SPIP algorithm accumulates too much of a blend component in previous periods: when the inventory level approaches or is at the maximum limit, the algorithm will necessarily have to use this component although it is not the best blend to achieve a global optimal solution. MPIP algorithm requires less number of iterations than single-period algorithm as can be seen from Table 5.20 and 5.22 (case study 7 being the exception).

Decomposition of blend planning in two levels, as presented in this work, significantly simplifies the model solved at each level in comparison with the fine-grid MINLP model. Table 5.25 shows that the number of equations, continuous variables, and non-zero elements in the NLP model (i.e. blend recipe computation) and MILP model (i.e. volume allocation problem) is smaller than those in the MINLP model.

The solution times of the inventory pinch algorithms (using IPOPT for the NLP model and CPLEX for the MILP model) are much smaller than those required by DICOPT to solve the corresponding MINLP model. If (quality*volume) constraints are nonlinear, then the computational effort is reduced by at least an order of magnitude. This is expected since the constraints at each level of the inventory pinch algorithm are simpler to solve and also the algorithm is a heuristic algorithm that does not prove that the optimality conditions are met.

The inventory pinch algorithms lead to a smaller number of different blend recipes per product compared to the solution of the corresponding multi-period MINLP model (see Table 5.23 and 5.24). Since blend recipes computed by MINLP model are not exactly the same, for comparison purposes, composition ranges of 1% and 5% were used to count the number of unique recipes. When using a 5% composition range sometimes the number given by the inventory pinch algorithm is greater. Significant reduction is seen in the cases with only one pinch point and constant supply rates of blend components. In addition, repeated recipes from the MINLP solution may or may not be used in adjacent periods; thus, the number of recipe switching is equal or greater than the number of different recipes. Usually, the number of different blend recipes and iterations is less for the multi-period algorithm than those of the single-period algorithm.

For both inventory pinch algorithms, solver selection for the recipe optimization level is very important. As observed with our case studies, blend recipes that disregard the use of several blend components (i.e. volume fraction of blend component is zero or close to zero) are not likely to generate feasible blend plans at the lower level. This can be due to having few degrees of freedom when using such blend recipes at the lower level. Blend recipes computed by IPOPT and COUENNE led to feasible solutions (in this work only the results using IPOPT are included) while recipes computed by

BARON, MINOS, CONOPT and GUROBI often led to MILP solutions containing inventory infeasibilities (i.e. non-zero slack variables) at the lower level.

Table 5.20: Objective function values and solution times for case studies 1 to 18

Case Study	Supply rate	DICOPT Solution (MINLP model)		Multi-Period Inventory Pinch Algorithm Solution (IPOPT, CPLEX)			Single-Period Inventory Pinch Algorithm Solution (IPOPT, CPLEX)		
		Objective Function ($\times 10^3$\$)	Total CPU time (s)	Objective Function ($\times 10^3$\$)	Total CPU time (s)	Iterations	Objective Function ($\times 10^3$\$)	Total CPU time (s)	Iterations
1	Constant	37,542.5	13.9	37,542.5	2.2	1	37,542.5	2.2	1
2	Constant	38,121.2	99.4	38,121.2	1.9	1	38,121.2	1.5	1
3	Constant	38,309.9	11.6	38,309.9	1.0	1	38,309.9	1.1	1
4	Constant	37,991.1	12.5	37,991.1	1.1	1	37,991.1	1.1	1
5	Constant	37,864.2	7.0	37,864.2	1.0	1	37,864.2	1.1	1
6	Constant	37,680.6	17.2	37,680.6	1.0	1	37,702.5	2.7	2
7	Constant	37,324.5	17.2	37,324.5	7.8	4	37,412.1	3.8	3
8	Constant	37,761.7	7.7	37,761.8	1.5	1	37,761.7	1.6	1
9	Constant	37,377.5	14.5	37,377.5	1.2	1	37,377.5	2.2	1
10	Irregular	37,943.4	14.4	37,943.4	0.9	1	37,943.4	0.9	1
11	Irregular	38,518.4	17.7	38,518.5	1.0	1	38,518.4	4.7	4
12	Irregular	38,754.2	14.6	38,754.2	2.5	2	38,849.8	3.9	4
13	Irregular	38,405.3	22.3	38,405.3	1.1	1	38,437.9	1.2	1
14	Irregular	38,196.0	14.2	38,196.0	1.3	1	38,196.0	1.4	1
15	Irregular	38,073.4	18.7	38,073.4	1.2	1	38,291.5	3.4	2
16	Irregular	37,784.5	21.9	37,784.5	2.5	2	38,184.8	7.5	5
17	Irregular	38,192.6	15.6	38,192.6	1.3	1	38,316.0	3.2	2
18	Irregular	37,796.5	14.4	37,796.5	1.5	1	38,076.3	5.2	3

Table 5.21: Objective function values and solution times for case studies 19 to 30

Case Study	Number of blenders	Initial Product Inventory	RVP Blend Property	DICOPT Solution (MINLP model)		Single-Period Inventory Pinch Algorithm Solution (IPOPT, CPLEX)		
				Objective Function ($\times 10^3$\$)	Total CPU time (s)	Objective Function ($\times 10^3$\$)	Total CPU time (s)	Iterations
19	1	On-Spec	Linear	43,421.7	12.3	43,421.7	4.2	2
20	1	On-Spec	Linear	41,350.0	29.1	41,350.0	1.2	1
21	1	On-Spec	Linear	43,161.3	37.7	43,161.3	1.4	1
21	2	On-Spec	Linear	43,161.3	63.5	43,161.3	1.5	1
22	1	On-Spec	Linear	41,874.4	20.5	41,874.4	1.4	1
22	2	On-Spec	Linear	41,874.5	115.2	41,874.5	2.8	1
23	1	On-Spec	Nonlinear	43,658.0	13.8	43,658.0	1.0	1
24	1	On-Spec	Nonlinear	43,612.0	12.8	43,612.0	2.2	2
25	1	On-Spec	Nonlinear	43,612.0	33.2	43,612.0	1.1	1
25	2	On-Spec	Nonlinear	43,612.0	34.6	43,612.0	1.2	1
26	1	On-Spec	Nonlinear	43,935.0	18.5	43,935.0	1.7	1
26	2	On-Spec	Nonlinear	43,935.0	82.3	43,935.0	2.0	1
27	1	On-Spec	Nonlinear	43,627.5	372.6	43,627.5	8.8	5
27	1	On-Spec	Linear	43,142.3	425.0	43,142.3	7.8	5
28	1	On-Spec	Nonlinear	43,612.0	641.8	43,667.7	5.0	3
28	1	On-Spec	Linear	43,101.3	235.6	43,126.3	4.3	3
29	1	Off-Spec	Linear	43,425.2	15.2	43,425.2	2.9	1
30	1	Off-Spec	Linear	41,470.7	7.7	41,470.7	3.1	1
30	2	Off-Spec	Linear	41,470.7	56.9	41,470.7	3.6	1

Table 5.22: Objective function values and solution times for case studies 22, 26, 27, and 28

Case Study	Number of blenders	Initial Product Inventory	RVP Blend Property	DICOPT Solution (MINLP model)		Single-Period Inventory Pinch Algorithm Solution (IPOPT, CPLEX)			Multi-Period Inventory Pinch Algorithm Solution (IPOPT, CPLEX)		
				Objective Function ($\times 10^3$\$)	Total CPU time (s)	Objective Function ($\times 10^3$\$)	Total CPU time (s)	Iterations	Objective Function ($\times 10^3$\$)	Total CPU time (s)	Iterations
22	1	On-Spec	Linear	41,874.4	20.5	41,874.4	1.4	1	41,874.5	0.9	1
22	2	On-Spec	Linear	41,874.5	115.2	41,874.5	2.8	1	41,874.5	1.2	1
26	1	On-Spec	Nonlinear	43,935.0	18.5	43,935.0	1.7	1	43,935.0	1.8	1
26	2	On-Spec	Nonlinear	43,935.0	82.3	43,935.0	2.0	1	43,935.0	2.1	1
27	1	On-Spec	Nonlinear	43,627.5	372.6	43,627.5	8.8	5	43,627.6	2.1	2
27	1	On-Spec	Linear	43,142.3	425.0	43,142.3	7.8	5	43,142.4	2.2	2
28	1	On-Spec	Nonlinear	43,612.0	641.8	43,667.7	5.0	3	43,612.0	0.9	1
28	1	On-Spec	Linear	43,101.4	235.6	43,126.3	4.3	3	43,101.4	0.9	1

Table 5.23: Number of blend recipes (case studies 1 to 18 have 14 time *l*-periods, 1 blender, RVP blends nonlinearly, and the initial product inventory is on-spec)

Case Study	Supply profile	Number of Pinch Points	DICOPT Solution (MINLP model)		Multi-Period Inventory Pinch Algorithm Solution (IPOPT, CPLEX)	Single-Period Inventory Pinch Algorithm Solution (IPOPT, CPLEX)
			Number of different recipes (1%)[a]	Number of different recipes (5%)[b]	Number of different recipes (0%)[c]	Number of different recipes (0%)[c]
1	Constant	0	6	3	1	1
2	Constant	1	6	4	2	2
3	Constant	1	8	3	2	2
4	Constant	1	8	5	2	2
5	Constant	1	7	3	2	2
6	Constant	2	7	4	3	4
7	Constant	2	8	5	6	5
8	Constant	2	7	5	3	3
9	Constant	3	8	4	4	4
10	Irregular	0	6	3	1	1
11	Irregular	1	8	3	2	4
12	Irregular	1	6	3	3	5
13	Irregular	1	7	3	2	2
14	Irregular	1	6	4	2	2
15	Irregular	2	7	4	3	4
16	Irregular	2	8	6	4	6
17	Irregular	2	7	4	3	3
18	Irregular	3	8	4	4	5

[a] Blend recipes were counted using composition intervals of 1%
[b] Blend recipes were counted using composition intervals of 5%
[c] Repeated blend recipes are exactly the same

Table 5.24: Number of blend recipes (case studies 19 to 30 have 14 time *l*-periods and 1 blender)

Case Study	Supply profile	Initial Product Inventory	RVP Blend Property	Number of Pinch Points	DICOPT Solution (MINLP model)		Multi-Period Inventory Pinch Algorithm Solution (IPOPT, CPLEX)	Single-Period Inventory Pinch Algorithm Solution (IPOPT, CPLEX)
					Number of different recipes (1%)[a]	Number of different recipes (5%)[b]	Number of different recipes (0%)[c]	Number of different recipes (0%)[c]
19	Constant	On-Spec	Linear	2	8	6	4	4
20	Constant	On-Spec	Linear	1	7	4	2	2
21	Constant	On-Spec	Linear	2	6	3	3	3
22	Constant	On-Spec	Linear	1	7	3	2	2
23	Constant	On-Spec	Nonlinear	1	7	3	2	2
24	Constant	On-Spec	Nonlinear	1	9	5	4	3
25	Constant	On-Spec	Nonlinear	1	7	4	2	2
26	Constant	On-Spec	Nonlinear	3	9	5	4	4
27	Irregular	On-Spec	Nonlinear	1	7	4	3	6
27	Irregular	On-Spec	Linear	1	7	5	3	6
28	Irregular	On-Spec	Nonlinear	1	6	3	2	4
28	Irregular	On-Spec	Linear	1	8	4	2	4
29	Constant	Off-Spec	Linear	2	7	3	4	4
30	Constant	Off-Spec	Linear	3	7	5	5	5

[a] Blend recipes were counted using composition intervals of 1%
[b] Blend recipes were counted using composition intervals of 5%
[c] Repeated blend recipes are exactly the same

Table 5.25: Model size comparison (RVP blends nonlinearly)

Model	# Equations	# Continuous Variables	# Discrete Variables	# Non-zeros
MINLP model (1 blender, 14 time periods)	1,723	1,275	42	7,187
MINLP model (2 blenders, 14 time periods)	2,885	1,989	84	12,983
NLP model (MPIP algorithm, 2 time periods)	231	171	0	869
NLP model (SPIP algorithm)	106	76	0	410
MILP model (1 blender, 14 time periods, 2 fixed recipes)	1,268	940	42	3,498
MILP model (2 blenders, 14 time periods, 2 fixed recipes)	1,968	1,318	84	5,598

Chapter 6. Conclusions and Future Work

Two novel algorithms based on the concept of inventory pinch points and the associated two-level decomposition of the gasoline blend planning problem are introduced in this work. An inventory pinch point is the point in time where the cumulative average total production curve is tangent to the cumulative total demand curve, and if extrapolated from this point onwards, it does not cross the CTD curve. At the top level of the algorithms, blend recipes are optimized based on the total demand between the pinch points using a NLP model. At the lower level, blend volumes are computed from a fine-grid MILP model. Both algorithms address minimization of the number of different blend recipes.

Experiments with a number of examples show that the solutions computed by the single period inventory pinch algorithm are most of the time less than 0.01% away from the optimum solutions computed by the corresponding discrete-time multi-period calendar based MINLP model when the supply of blend components is constant. In all cases, solution times are smaller than those required by DICOPT MINLP solver; as a rule, in cases with nonlinear blend constraints, execution times are an order of magnitude

lower. The number of blend recipe switches along the blend planning horizon is much smaller than computed by the MINLP model; this feature makes it simpler for schedulers that use interactive simulation software to create blend schedules.

Multi-period inventory pinch planning algorithm computes the same optimal value of the objective function, but with a lot smaller number of distinct recipes and a lot smaller number of recipe switches.

Single-period algorithm was developed in order to use computationally intensive nonlinear models for blend planning since oil refineries still use rigorous single-period models for production planning. Improvements are required for the single-period algorithm since it produces suboptimal solutions especially when the blend component supply flow rates are irregular and when the inventories of blend components are close to the maximum inventory limits.

Decomposition of the MINLP non-linear planning model introduced in this work is expected to be valid for general planning models. Intuitive explanation of MPIP algorithm capability to compute optimal solutions is that any optimal production will have to pass through the inventory pinch points. Proposed decomposition is likely to work for general production planning problems with convex constraints; this is the topic that will be investigated in the future.

Solver selection for the recipe optimization level is important and has a great effect on the algorithm performance; IPOPT and COUENNE provided the best results among the solvers tested.

Goals of future work are to solve the complete blend scheduling and oil refinery production planning optimization problems using these algorithms. The mathematical models will be upgraded in order to include more aspects of the gasoline blending system such as minimization of number of product switches, inclusion of swing and multi-purpose tanks, and use of demurrage costs. In addition, research will be carried out to explore use of different time representations at the different algorithm levels (e.g. top level can use discrete-time models and the lower level a continuous-time formulation). Since the top level solution is a lower bound of the global solution, and the lower level result is an upper bound, work will be done in order to guarantee optimality conditions of the final result provided by these algorithms.

Bibliography

DeWitt, C.W., Lasdon, L.S., Waren, A.D., Brenner, D.A., & Melhem, S.A. (1989). Omega: An improved gasoline blending system for Texaco. Interfaces, 85-101.

Glismann, K., & Gruhn, G. (2001). Short-term scheduling and recipe optimization of blending processes. Computers and Chemical Engineering, 25, 627-634.

Jia, Z., & Ierapetritou, M. (2003). Mixed-integer linear programming model for gasoline blending and distribution scheduling. Industrial and Engineering Chemistry Research, 42, 825-835.

Joly, M., & Pinto, J.M. (2003). Mixed-integer programming techniques for the scheduling of fuel oil and asphalt production. Institution of Chemical Engineers *A*., 81, 427 – 447.

Kelly, J.D., & Mann, J.L. (2003a). Crude-oil blend scheduling optimization: An application with multi-million dollar benefits: Part I. Hydrocarbon Processing, 47-53.

Kelly, J.D., & Mann, J.L. (2003b). Crude-oil blend scheduling optimization: An application with multi-million dollar benefits: Part II. Hydrocarbon Processing, 72-79.

Kelly, J.D. (2004). Formulating production planning models. Chemical Engineering Progress, 100, 43-50.

Li, J., Karimi, I.A., & Srinivasan, R. (2010). Recipe determination and scheduling of gasoline blending operations. A.I.Ch.E. Journal, 56, 441-465.

Li, J., & Karimi, I.A. (2011). Scheduling gasoline blending operations from recipe determination to shipping using unit slots. Industrial and Engineering Chemistry Research, 50, 9156-9174.

Main, R.A. (1993). Large recursion models: practical aspects of recursion techniques. In: Ciriani TA, Leachman RC, eds. Optimization in Industry. New York, NY: John Wiley & Sons Ltd.

Mendez, C.A., Grossman, I.E., Harjunkoski, I., & Kabore, P. (2006). A simultaneous optimization approach for off-line blending and scheduling of oil refinery operations. Computers and Chemical Engineering, 30, 614-634.

Misener, R., & Floudas, C. (2009). Advances for the pooling problem: Modeling, global optimization, and computational studies – Survey. Applied and Computational Mathematics, 8, 3-22.

Misener, R., Gounaris, C.E., & Floudas, C. (2010). Mathematical modeling and global optimization of large-scale extended pooling problems with EPA complex emissions constraints. Computers and Chemical Engineering, 34, 1432-1456.

Neiro, S.M.S., & Pinto, J. (2005). Multiperiod optimization for production planning of petroleum refineries. Chemical Engineering Communications, 192, 62-88.

Sing, A., Forbes, J.F., Vermeer, P.J., & Woo, S.S. (2000). Model-based real-time optimization of automotive gasoline blending operations. Journal of Process Control, 10, 43-58.

Thakral, A., & Mahalec, V. (2012). Composite planning and scheduling algorithm addressing intra-period infeasibilities of gasoline blend planning models. Canadian Journal of Chemical Engineering, 9999, 1-12.

U.S. Government Printing Office. Code of Federal Regulations: Complex emissions model. Year: 2007. Title: 40. Section: CFR80.45. http://www.gpoaccess.gov/cfr/retrieve.html. Accessed August 13, 2012.

APPENDIX A: GAMS models

The General Algebraic Modeling System (GAMS) is a high-level modeling system for mathematical programming and optimization. It consists of a language compiler and a stable of integrated high-performance solvers. It is specifically designed for modeling linear, nonlinear and mixed integer optimization problems

- **MINLP model (1 blender, RVP blends nonlinearly, constant blend components supply flow rates)**

```
SETS
I         blend components /ALK, BUT, HCL, HCN, LCN, LNP, RFT/
N         time periods /1*14/
TC(N)     time periods for connection equations /1*13/
K         quality properties /aro, ben, mon, olf, ron, rvi, spg, sul/
L(K)      linear quality properties /aro, ben, mon, olf, ron, spg, sul/
NL(K)     nonlinear quality properties /rvi/
J         products /U87, U91, U93/
Bl        blenders /A/;

PARAMETERS

C(I)      blend components cost ($ per BBL)
/ALK        30
BUT         12
HCL         20
HCN         22
LCN         25
LNP         20
RFT         25/

F(I)   blend components flowrates (BBL per day)
/ALK        18
BUT         5
HCL         3
HCN         5
LCN         25
LNP         20
RFT         44/

Vmax(I)    blend components maximum inventory constraints (BBL)
/ALK         150
BUT         75
HCL         50
HCN         50
LCN         150
LNP         100
RFT         150/
```

```
Vmin(I)    blend components minimum inventory constraints (BBL)
/ALK 5
BUT 5
HCL 5
HCN 5
LCN 5
LNP 5
RFT 5/

VPmax(J)    maximum product inventory constraints (BBL)
/U87 300
U91 300
U93 100/

VPmin(J)    minimum product inventory constraints (BBL)
/U87 10
U91 10
U93 10/

SpI(J)    initial product inventory (BBL)
/U87 70
U91 140
U93 30/

SoI(I)    blend component starting opening inventory (BBL)
/ALK       30
BUT        20
HCL        20
HCN        10
LCN        30
LNP        20
RFT        50/

BlendRatemax(Bl) Maximum blend capacity (BBL per day)
/A       175/

Blendmin(Bl) Minimum volume to blend of each product if it is going to be
blended (BBL per day)
/A       30/

Blendmax(Bl) BBL per day
/A       175/

setupvol(Bl) BBL lost per switch
/A       8/

np(Bl) number of products that can be blended in one time period
/A       3/

TABLE D(J, N)    product demand rate (BBL)
           1         2         3         4         5         6        7
8          9        10        11        12        13        14
U87        60        50        50        80        50        60       60
50         75        50        50        50        80       100
U91        50        80        70        30        50         0       40
30         30        50        40        40        30        50
U93        30        30         0         0        40        40        0
35         30         0         0        40        30        40
```

```
TABLE Q(K, I)    blend components quality properties
           ALK      BUT      HCL      HCN       LCN       LNP       RFT
aro        0        0        0        25        18        2.974     74.9
ben        0        0        0        0.5       1         0.595     7.5
mon        93.7     90       79.8     75.8      81.6      66        90.8
olf        0        0        0        14        27        0         0
ron        95       93.8     82.3     86.7      93.2      67.8      103
rvi        5.15     138      22.335   2.378     13.876    19.904    3.622
spg        0.703    0.584    0.695    0.791     0.744     0.677     0.818
sul        0        0        0        0.485     0.078     0.013     0;

TABLE IQP(K,J)   initial quality of each product
           U87        U91        U93
aro        5          10         20
ben        4          4          5
mon        83.2       87         88.2
olf        15         12         18
ron        91.4       95         98.2
rvi        15         8          12
spg        0.75       0.76       0.75
sul        0.05       0.03       0.04;

TABLE Qmin(K,J)      minimum qaulity specifications for the products
           U87        U91        U93
aro        0          0          0
ben        0          0          0
mon        81.5       85.7       87.5
olf        0          0          0
ron        91.4       94.5       97.5
rvi        0          0          0
spg        0.73       0.73       0.73
sul        0          0          0;

TABLE Qmax(K,J)      maximum quality specifications for the products
           U87        U91        U93
aro        60         60         60
ben        5.9        5.9        5.9
mon        200        200        200
olf        24.2       24.2       24.2
ron        200        200        200
rvi        15.6       15.6       15.6
spg        0.81       0.81       0.81
sul        0.1        0.1        0.1;

SCALAR t time periods length (day) /1/
SCALAR PenaltyBC penalty for the inventory infeasibilities on the Blend
Components' side /1000/
SCALAR PenaltyP penalty for the inventory infeasibilities on the Products' side
/1000/

FREE VARIABLES
Z                    total cost

POSITIVE VARIABLES
x(I,J,N,Bl)          blend recipes
Vopen(I,N)           raw materials opening inventories
Vclose(I,N)          raw materials closing inventories
VopenP(J,N)          product opening inventories
VcloseP(J,N)         product closing inventories
SposSlack(I,N)       raw materials inventory positive slack variable
```

```
SnegSlack(I,N)      raw materials inventory negative slack variable
SpposSlack(J,N)     products inventory positive slack variable
SpnegSlack(J,N)     products inventory negative slack variable
Vb(I,J,N,Bl)        raw material amount to be processed per product at N
Vblend(J,N,Bl)      volume of J to be blended at N
qnew(K,J,N,Bl)      quality K of product J at N

BINARY VARIABLES
det(J,N,Bl)   binary variable defining if product J is going to be blended at N

EQUATIONS

COST                defines objective function
MASSBALANCE         observe mass balance in blend components storage tanks
PRODCOMPCONST       observe mass balance in product storage tanks
XCONST              sum of volume fractions equal to 1
PRODUCTBALANCE      observe mass balance in product storage tanks
MaxBlend            maximum blending rate
MinOneBlend         minimum blending rate per product (1)
MaxOneBlend         minimum blending rate per product (2)
Discret             binary variable constraint
QUALITY1            observe maximum quality specifications of linear blending
properties
QUALITY2            observe minimum quality specifications of linear blending
properties
QUALITYrvp          RVP quality calculation
QUALITYNmax         observe maximum quality specifications of nonlinear
blending properties
QUALITYNmin         observe minimum quality specifications of nonlinear
blending properties
SS                  raw materials starting inventories
SSp                 product starting inventory
SocR                raw materials inventories correspondence in N
SocP                product inventory correspondence in N
InvCon1             raw materials minimum inventory constraints
InvCon2             raw materials maximum inventory constraints
InvCon3             products minimum inventory constraints
InvCon4             products maximum inventory constraints
;


MASSBALANCE(I,N) ..           F(I)*t+Vopen(I,N)-Vclose(I,N)-
SUM((J,Bl),Vb(I,J,N,Bl))+SposSlack(I,N)-SnegSlack(I,N) =E= 0;
PRODCOMPCONST(I,J,N,Bl)..     Vb(I,J,N,Bl) =E= x(I,J,N,Bl)*Vblend(J,N,Bl);
XCONST(J,N,Bl)..              SUM(I,x(I,J,N,Bl)) =E= 1;
PRODUCTBALANCE(J,N)..         SUM(Bl,Vblend(J,N,Bl))+VopenP(J,N)-VcloseP(J,N)-
D(J,N)+SpposSlack(J,N)-SpnegSlack(J,N)=E= 0;
Discret(N,Bl)..               SUM(J,det(J,N,Bl)) =L= np(Bl);
MaxBlend(N,Bl)..
SUM(J,Vblend(J,N,Bl))+(SUM(J,det(J,N,Bl)))*setupvol(Bl) =L= BlendRatemax(Bl);
MinOneBlend(J,N,Bl)..         Vblend(J,N,Bl) =G= Blendmin(Bl)*det(J,N,Bl);
MaxOneBlend(J,N,Bl)..         Vblend(J,N,Bl) =L= Blendmax(Bl)*det(J,N,Bl);
QUALITY1(K,J,N,Bl)$L(K)..     SUM(I,Vb(I,J,N,Bl)*Q(K,I))=L=
Qmax(K,J)*Vblend(J,N,Bl);
QUALITY2(K,J,N,Bl)$L(K)..     SUM(I,Vb(I,J,N,Bl)*Q(K,I))=G=
Qmin(K,J)*Vblend(J,N,Bl);
QUALITYrvp(J,N,Bl)..
(SUM(I,x(I,J,N,Bl)*(Q("rvi",I)**(1.25))))**(0.8)=E= qnew("rvi",J,N,Bl);
QUALITYNmax(K,J,N,Bl)$NL(K).. qnew(K,J,N,Bl) =L= Qmax(K,J);
QUALITYNmin(K,J,N,Bl)$NL(K).. qnew(K,J,N,Bl) =G= Qmin(K,J);
```

```
SS(I) ..                        Vopen(I,"1") =E= SoI(I);
SSp(J) ..                       VopenP(J,"1") =E= SpI(J);
SocR(I,N)$TC(N)..               Vopen(I,N+1)-Vclose(I,N) =E= 0;
SocP(J,N)$TC(N)..               VopenP(J,N+1)-VcloseP(J,N) =E= 0;
InvCon1(I,N)..                  Vclose(I,N) =G= Vmin(I);
InvCon2(I,N)..                  Vclose(I,N) =L= Vmax(I);
InvCon3(J,N)..                  VcloseP(J,N) =G= VPmin(J);
InvCon4(J,N)..                  VcloseP(J,N) =L= VPmax(J);
COST ..                         Z =E= SUM(N,SUM((I,J,Bl),C(I)*Vb(I,J,N,Bl))
                              +SUM(I,PenaltyBC*(SposSlack(I,N)+SnegSlack(I,N)))
                              +SUM(J,PenaltyP*(SpposSlack(J,N)+SpnegSlack(J,N))));

MODEL BLENDING /ALL/;
BLENDING.DOMLIM=100;
SOLVE BLENDING USING MINLP MINIMIZING Z;
OPTION decimals=8;
DISPLAY Z.L, x.L, Vb.L, Vblend.L, Vopen.L, Vclose.L, VopenP.L, VcloseP.L,
qnew.L, det.L,
SposSlack.L,SnegSlack.L,SpposSlack.L,SpnegSlack.L;
```

- **Multi-period NLP model (RVP blends nonlinearly, 4 time periods, constant blend components supply flow rates)**

```
SETS
I        blend components /ALK, BUT, HCL, HCN, LCN, LNP, RFT/
N        t-periods /1,2,3,4/
TC(N)    time periods for connection equations /1,2,3/
K        quality properties /aro, ben, mon, olf, ron, rvi, spg, sul/
L(K)     linear quality properties /aro, ben, mon, olf, ron, spg, sul/
NL(K)    nonlinear quality properties /rvi/
J        products /U87, U91, U93/

PARAMETERS

t(N) time periods length (day)
/1       4
2        4
3        4
4        2/

C(I)     blend components cost ($ per BBL)
/ALK       30
BUT        12
HCL        20
HCN        22
LCN        25
LNP        20
RFT        25/

F(I)   blend components flowrates (BBL per day)
/ALK       18
BUT        5
HCL        3
HCN        5
LCN        25
LNP        20
RFT        44/
```

```
Vmax(I)    blend components maximum inventory constraints (BBL)
/ALK        150
BUT        75
HCL        50
HCN        50
LCN        150
LNP        100
RFT        150/

Vmin(I)    blend components minimum inventory constraints (BBL)
/ALK 5
BUT 5
HCL 5
HCN 5
LCN 5
LNP 5
RFT 5/

VPmax(J)    maximum product inventory constraints (BBL)
/U87 300
U91 300
U93 100/

VPmin(J)    minimum product inventory constraints (BBL)
/U87 10
U91 10
U93 10/

SpI(J)    initial product inventory (BBL)
/U87 70
U91 140
U93 30/

SoI(I)    blend component starting opening inventory (BBL)
/ALK       30
BUT        20
HCL        20
HCN        10
LCN        30
LNP        20
RFT        50/

TABLE D(J, N)    product demand rate (BBL)
           1          2          3       4
U87        350        220        220     50
U91        250        140        125     60
U93        135        90         60      60

TABLE Vblend(J, N)    product demand rate (BBL)
           1          2          3       4
U87        290        220        220     50
U91        120        140        125     60
U93        115        90         60      60
```

```
TABLE Q(K, I)    blend components quality properties
           ALK      BUT      HCL      HCN       LCN       LNP       RFT
aro        0        0        0        25        18        2.974     74.9
ben        0        0        0        0.5       1         0.595     7.5
mon        93.7     90       79.8     75.8      81.6      66        90.8
olf        0        0        0        14        27        0         0
ron        95       93.8     82.3     86.7      93.2      67.8      103
rvi        5.15     138      22.335   2.378     13.876    19.904    3.622
spg        0.703    0.584    0.695    0.791     0.744     0.677     0.818
sul        0        0        0        0.485     0.078     0.013     0;

TABLE IQP(K,J)   initial quality of each product
           U87        U91        U93
aro        5          10         20
ben        4          4          5
mon        83.2       87         88.2
olf        15         12         18
ron        91.4       95         98.2
rvi        15         8          12
spg        0.75       0.76       0.75
sul        0.05       0.03       0.04;

TABLE Qmin(K,J)      minimum qaulity specifications for the products
           U87        U91        U93
aro        0          0          0
ben        0          0          0
mon        81.5       85.7       87.5
olf        0          0          0
ron        91.4       94.5       97.5
rvi        0          0          0
spg        0.73       0.73       0.73
sul        0          0          0;

TABLE Qmax(K,J)      maximum quality specifications for the products
           U87        U91        U93
aro        60         60         60
ben        5.9        5.9        5.9
mon        200        200        200
olf        24.2       24.2       24.2
ron        200        200        200
rvi        15.6       15.6       15.6
spg        0.81       0.81       0.81
sul        0.1        0.1        0.1;

SCALAR PenaltyBC penalty for the inventory infeasibilities on the Blend
Components' side /500/
SCALAR PenaltyP penalty for the inventory infeasibilities on the Products' side
/1000/

FREE VARIABLES
Z                total cost

POSITIVE VARIABLES
x(I,N,J)         blend recipes
Vopen(I,N)       raw materials opening inventories
Vclose(I,N)      raw materials closing inventories
VopenP(J,N)      product opening inventories
VcloseP(J,N)     product closing inventories
SposSlack(I,N)   raw materials inventory positive slack variable
SnegSlack(I,N)   raw materials inventory negative slack variable
```

```
SpposSlack(J,N)  products inventory positive slack variable
SpnegSlack(J,N)  products inventory negative slack variable
Vb(I,J,N)        raw material amount to be processed per product at N
qnew(K,J,N)      quality K of product J at N

BINARY VARIABLES
det(J,N)   binary variable defining if product J is going to be blended at N

EQUATIONS

COST                defines objective function
MASSBALANCE         observe mass balance in blend components storage tanks
PRODCOMPCONST       observe mass balance in product storage tanks
XCONST              sum of volume fractions equal to 1
PRODUCTBALANCE      observe mass balance in product storage tanks
QUALITY1            observe maximum quality specifications of linear blending
properties
QUALITY2            observe minimum quality specifications of linear blending
properties
QUALITYrvp          RVP quality calculation
QUALITYNmax         observe maximum quality specifications of nonlinear
blending properties
QUALITYNmin         observe minimum quality specifications of nonlinear
blending properties
SS                  raw materials starting inventories
SSp                 product starting inventory
SocR               raw materials inventories correspondence in N
SocP               product inventory correspondence in N
InvCon1             raw materials minimum inventory constraints
InvCon2             raw materials maximum inventory constraints
InvCon3             products minimum inventory constraints
InvCon4             products maximum inventory constraints
;


MASSBALANCE(I,N) ..        F(I)*t(N)+Vopen(I,N)-Vclose(I,N)-
SUM(J,Vb(I,J,N))+SposSlack(I,N)-SnegSlack(I,N) =E= 0;
PRODCOMPCONST(I,J,N)..     Vb(I,J,N) =E= x(I,N,J)*Vblend(J,N);
XCONST(J,N)..              SUM(I,x(I,N,J)) =E= 1;
PRODUCTBALANCE(J,N)..      Vblend(J,N)+VopenP(J,N)-VcloseP(J,N)-
D(J,N)+SpposSlack(J,N)-SpnegSlack(J,N)=E= 0;
QUALITY1(K,J,N)$L(K)..     SUM(I,Vb(I,J,N)*Q(K,I))=L= Qmax(K,J)*Vblend(J,N);
QUALITY2(K,J,N)$L(K)..     SUM(I,Vb(I,J,N)*Q(K,I))=G= Qmin(K,J)*Vblend(J,N);
QUALITYrvp(J,N)..          (SUM(I,x(I,N,J)*(Q("rvi",I)**(1.25))))**(0.8)=E=
qnew("rvi",J,N);
QUALITYNmax(K,J,N)$NL(K).. qnew(K,J,N) =L= Qmax(K,J);
QUALITYNmin(K,J,N)$NL(K).. qnew(K,J,N) =G= Qmin(K,J);
SS(I) ..                   Vopen(I,"1") =E= SoI(I);
SSp(J) ..                  VopenP(J,"1") =E= SpI(J);
SocR(I,N)$TC(N)..         Vopen(I,N+1)-Vclose(I,N) =E= 0;
SocP(J,N)$TC(N)..         VopenP(J,N+1)-VcloseP(J,N) =E= 0;
InvCon1(I,N)..             Vclose(I,N) =G= Vmin(I);
InvCon2(I,N)..             Vclose(I,N) =L= Vmax(I);
InvCon3(J,N)..             VcloseP(J,N) =G= VPmin(J);
InvCon4(J,N)..             VcloseP(J,N) =L= VPmax(J);
COST ..                    Z =E= SUM(N,SUM((I,J),C(I)*Vb(I,J,N))
                           +SUM(I,PenaltyBC*(SposSlack(I,N)+SnegSlack(I,N)))
                           +SUM(J,PenaltyP*(SpposSlack(J,N)+SpnegSlack(J,N))));
```

```
MODEL BLENDING /ALL/;
BLENDING.DOMLIM=100;
SOLVE BLENDING USING NLP MINIMIZING Z;
OPTION decimals=8;
DISPLAY Z.L, x.L, Vb.L, Vopen.L, Vclose.L, VopenP.L, VcloseP.L, qnew.L,
SposSlack.L,SnegSlack.L,SpposSlack.L,SpnegSlack.L;
```

- **Single-period NLP model (RVP blends nonlinearly, constant blend components supply flow rates)**

```
SETS
I        blend components /ALK, BUT, HCL, HCN, LCN, LNP, RFT/
K        quality properties /aro, ben, mon, olf, ron, rvi, spg, sul/
L(K)     linear quality properties /aro, ben, mon, olf, ron, spg, sul/
NL(K)    nonlinear quality properties /rvi/
J        products /U87, U91, U93/

PARAMETERS

C(I)     blend components cost ($ per BBL)
/ALK       30
BUT        12
HCL        20
HCN        22
LCN        25
LNP        20
RFT        25/

F(I)   blend components flowrates (BBL per day)
/ALK       18
BUT        5
HCL        3
HCN        5
LCN        25
LNP        20
RFT        44/

Vmax(I)    blend components maximum inventory constraints (BBL)
/ALK        150
BUT        75
HCL        50
HCN        50
LCN        150
LNP        100
RFT        150/
```

```
Vmin(I)    blend components minimum inventory constraints (BBL)

/ALK 5
BUT 5
HCL 5
HCN 5
LCN 5
LNP 5
RFT 5/

VPmax(J)    maximum product inventory constraints (BBL)
/U87 300
U91 300
U93 100/

VPmin(J)    minimum product inventory constraints (BBL)
/U87 10
U91 10
U93 10/

VopenP(J)    initial product inventory (BBL)
/U87 70
U91 140
U93 30/

Vopen(I)    blend component starting opening inventory (BBL)
/ALK       30
BUT        20
HCL        20
HCN        10
LCN        30
LNP        20
RFT        50/

D(J)    product demand rate (BBL)
/U87        865
U91        590
U93        315/

Vblend(J)    product demand rate (BBL)
/U87        805
U91        460
U93        295/
```

```
TABLE Q(K, I)    blend components quality properties
           ALK      BUT      HCL      HCN      LCN      LNP      RFT
aro        0        0        0        25       18       2.974    74.9
ben        0        0        0        0.5      1        0.595    7.5
mon        93.7     90       79.8     75.8     81.6     66       90.8
olf        0        0        0        14       27       0        0
ron        95       93.8     82.3     86.7     93.2     67.8     103
rvi        5.15     138      22.335   2.378    13.876   19.904   3.622
spg        0.703    0.584    0.695    0.791    0.744    0.677    0.818
sul        0        0        0        0.485    0.078    0.013    0;

TABLE IQP(K,J)   initial quality of each product
           U87        U91        U93
aro        5          10         20
ben        4          4          5
mon        83.2       87         88.2
olf        15         12         18
ron        91.4       95         98.2
rvi        15         8          12
spg        0.75       0.76       0.75
sul        0.05       0.03       0.04;

TABLE Qmin(K,J)      minimum qaulity specifications for the products
           U87        U91        U93
aro        0          0          0
ben        0          0          0
mon        81.5       85.7       87.5
olf        0          0          0
ron        91.4       94.5       97.5
rvi        0          0          0
spg        0.73       0.73       0.73
sul        0          0          0;

TABLE Qmax(K,J)      maximum quality specifications for the products
           U87        U91        U93
aro        60         60         60
ben        5.9        5.9        5.9
mon        200        200        200
olf        24.2       24.2       24.2
ron        200        200        200
rvi        15.6       15.6       15.6
spg        0.81       0.81       0.81
sul        0.1        0.1        0.1;

SCALAR t time periods length (day) /14/
SCALAR PenaltyBC penalty for the inventory infeasibilities on the Blend
Components' side /500/
SCALAR PenaltyP penalty for the inventory infeasibilities on the Products' side
/10000/

FREE VARIABLES
Z                  total cost
```

```
POSITIVE VARIABLES
x(I,J)              blend recipes
Vclose(I)           raw materials closing inventories
VcloseP(J)          product closing inventories
SposSlack(I)        raw materials inventory positive slack variable
SnegSlack(I)        raw materials inventory negative slack variable
SpposSlack(J)       products inventory positive slack variable
SpnegSlack(J)       products inventory negative slack variable
Vb(I,J)             raw material amount to be processed per product at N
qnew(K,J)           quality K of product J at N

BINARY VARIABLES
det(J)   binary variable defining if product J is going to be blended at N

EQUATIONS

COST                defines objective function
MASSBALANCE         observe mass balance in blend components storage tanks
PRODCOMPCONST       observe mass balance in product storage tanks
XCONST              sum of volume fractions equal to 1
PRODUCTBALANCE      observe mass balance in product storage tanks
QUALITY1            observe maximum quality specifications of linear blending
properties
QUALITY2            observe minimum quality specifications of linear blending
properties
QUALITYrvp          RVP quality calculation
QUALITYNmax         observe maximum quality specifications of nonlinear
blending properties
QUALITYNmin         observe minimum quality specifications of nonlinear
blending properties
InvCon1             raw materials minimum inventory constraints
InvCon2             raw materials maximum inventory constraints
InvCon3             products minimum inventory constraints
InvCon4             products maximum inventory constraints
;


MASSBALANCE(I) ..        F(I)*t+Vopen(I)-Vclose(I)-SUM(J,Vb(I,J))+SposSlack(I)-
SnegSlack(I) =E= 0;
PRODCOMPCONST(I,J)..     Vb(I,J) =E= x(I,J)*Vblend(J);
XCONST(J)..              SUM(I,x(I,J)) =E= 1;
PRODUCTBALANCE(J)..      Vblend(J)+VopenP(J)-VcloseP(J)-D(J)+SpposSlack(J)-
SpnegSlack(J)=E= 0;
QUALITY1(K,J)$L(K)..     SUM(I,Vb(I,J)*Q(K,I))=L= Qmax(K,J)*Vblend(J);
QUALITY2(K,J)$L(K)..     SUM(I,Vb(I,J)*Q(K,I))=G= Qmin(K,J)*Vblend(J);
QUALITYrvp(J)..          (SUM(I,x(I,J)*(Q("rvi",I)**(1.25))))**(0.8)=E=
qnew("rvi",J);
QUALITYNmax(K,J)$NL(K).. qnew(K,J) =L= Qmax(K,J);
QUALITYNmin(K,J)$NL(K).. qnew(K,J) =G= Qmin(K,J);
InvCon1(I)..             Vclose(I) =G= Vmin(I);
InvCon2(I)..             Vclose(I) =L= Vmax(I);
```

```
InvCon3(J)..              VcloseP(J) =G= VPmin(J);
InvCon4(J)..              VcloseP(J) =L= VPmax(J);
COST ..                   Z =E= SUM((I,J),C(I)*Vb(I,J))
                                +SUM(I,PenaltyBC*(SposSlack(I)+SnegSlack(I)))
                                +SUM(J,PenaltyP*(SpposSlack(J)+SpnegSlack(J)));

MODEL BLENDING /ALL/;
BLENDING.DOMLIM=100;
SOLVE BLENDING USING NLP MINIMIZING Z;
OPTION decimals=8;
DISPLAY Z.L, x.L, Vb.L, Vclose.L, VcloseP.L, qnew.L,
SposSlack.L,SnegSlack.L,SpposSlack.L,SpnegSlack.L;
```

- **MILP model (2 blenders, 2 fixed recipes, constant blend components supply flow rates)**

```
SETS
I         blend components /ALK, BUT, HCL, HCN, LCN, LNP, RFT/
J         products /U87, U91, U93/
Bl        blenders /A/
N         time periods /1*14/
TC(N)     time periods for connection equations /1*13/
TN(N)     time periods for target inventories /7,14/
K         t-periods /1,2/
NK(N,K)   /(1*7).(1),(8*14).(2)/

PARAMETERS

C(I)      blend components cost ($ per BBL)
/ALK        30
BUT         12
HCL         20
HCN         22
LCN         25
LNP         20
RFT         25/

F(I)    blend components flowrates (BBL per day)
/ALK        18
BUT         5
HCL         3
HCN         5
LCN         25
LNP         20
RFT         44/

Vmax(I)     blend components maximum inventory constraints (BBL)
/ALK         150
BUT         75
HCL         50
HCN         50
LCN         150
LNP         100
RFT         150/
```

```
Vmin(I)    blend components minimum inventory constraints (BBL)
/ALK 5
BUT 5
HCL 5
HCN 5
LCN 5
LNP 5
RFT 5/

VPmax(J)    maximum product inventory constraints (BBL)
/U87 300
U91 300
U93 100/

VPmin(J)    minimum product inventory constraints (BBL)
/U87 10
U91 10
U93 10/

SpI(J)    initial product inventory (BBL)
/U87 70
U91 140
U93 30/

SoI(I)    blend component starting opening inventory (BBL)
/ALK       30
BUT        20
HCL        20
HCN        10
LCN        30
LNP        20
RFT        50/

BlendRatemax(Bl) Maximum blend capacity (BBL per day)
/A       175/

Blendmin(Bl) Minimum volume to blend of each product if it is going to be
blended (BBL per day)
/A       30/

Blendmax(Bl) BBL per day
/A       175/

setupvol(Bl) BBL lost per switch
/A       8/

np(Bl) number of products that can be blended in one time period
/A       3/
```

```
PenaltyP(N) penalty for the inventory infeasibilities on the Blend Components'
side
/1        9E+8
2        8E+8
3        7E+8
4        6E+8
5        5E+8
6        1E+8
7        8E+7
8        7E+7
9        5E+6
10        1E+6
11        5E+4
12        1E+3
13        5E+1
14        1/

TABLE D(J, N)    product demand rate (BBL)
           1         2         3         4         5         6        7
8         9         10        11        12        13        14
U87        80        80        60        80        80        100      90
0         0         50        50        30        60        100
U91        50        50        50        30        30        50       50
30        30        50        0         50        60        50
U93        30        30        35        30        35        0        30
35        30        0         30        40        30        0

TABLE x(I,K,J) blend recipes
              U87         U91         U93
ALK.1  0.07785223  0.22527888  0.17553849
ALK.2  0.02027854  0.12753703  0.10031876
BUT.1  0.02445865  0.03595991  0.04341723
BUT.2  0.02145566  0.03433762  0.04184494
HCL.1  0.02202055  0.04440414  0.03127493
HCL.2  0.03288581  0.05775305  0.04442318
HCN.1  0.03175707  0.05908895  0.04486521
HCN.2  0.06554035  0.04892784  0.05044591
LCN.1  0.27544802  0.17630886  0.13017793
LCN.2  0.40294803  0.15057279  0.14027931
LNP.1  0.20111665  0.07831806  0.02959704
LNP.2  0.15745582  0.10497649  0.03397475
RFT.1  0.36734682  0.38064122  0.54512916
RFT.2  0.29943578  0.47589518  0.58871315

TABLE TarV(N,J) target product inventories
             U87               U91               U93
7            10                10                10
14           10                10                10

SCALAR t time periods length (day) /1/
SCALAR PenaltyBC penalty for the inventory infeasibilities on the Blend
Components' side /1E+12/

FREE VARIABLES
Z                   total cost

POSITIVE VARIABLES
BlendCost           blend cost
Vopen(I,N)          raw materials opening inventories
Vclose(I,N)         raw materials closing inventories
```

```
VopenP(J,N)          product opening inventories
VcloseP(J,N)         product closing inventories
SposSlack(I,N)       raw materials inventory positive slack variable
SnegSlack(I,N)       raw materials inventory negative slack variable
SpposSlack(J,N)      products inventory positive slack variable
SpnegSlack(J,N)      products inventory negative slack variable
Vb(I,J,N,Bl)         raw material amount to be processed per product at N
Vblend(J,N,Bl)       volume of J to be blended at N

BINARY VARIABLES
det(J,N,Bl)   binary variable defining if product J is going to be blended at N

EQUATIONS

COST                 defines objective function
BCOST                computes blend cost
MASSBALANCE          observe mass balance in blend components storage tanks
FIXRECIPE1           fixed recipe max limit
FIXRECIPE2           fixed recipe min limit
PRODUCTBALANCE       observe mass balance in product storage tanks
MaxBlend             maximum blending rate
MinOneBlend          minimum blending rate per product (1)
MaxOneBlend          minimum blending rate per product (2)
Discret              binary variable constraint
SS                   raw materials starting inventories
SSp                  product starting inventory
SocR                 raw materials inventories correspondence in N
SocP                 product inventory correspondence in N
InvCon1              raw materials minimum inventory constraints
InvCon2              raw materials maximum inventory constraints
InvCon3              products minimum inventory constraints
InvCon4              products maximum inventory constraints
Target               sets target inventories
;


MASSBALANCE(I,N) ..           F(I)*t+Vopen(I,N)-Vclose(I,N)-
SUM((J,Bl),Vb(I,J,N,Bl))+SposSlack(I,N)-SnegSlack(I,N) =E= 0;
FIXRECIPE1(I,J,N,K,Bl)$NK(N,K)..     Vb(I,J,N,Bl) =L=
(x(I,K,J)+0.0000001)*Vblend(J,N,Bl);
FIXRECIPE2(I,J,N,K,Bl)$NK(N,K)..     Vb(I,J,N,Bl) =G= (x(I,K,J)-
0.0000001)*Vblend(J,N,Bl);
PRODUCTBALANCE(J,N)..         SUM(Bl,Vblend(J,N,Bl))+VopenP(J,N)-VcloseP(J,N)-
D(J,N)+SpposSlack(J,N)-SpnegSlack(J,N)=E= 0;
Discret(N,Bl)..               SUM(J,det(J,N,Bl)) =L= np(Bl);
MaxBlend(N,Bl)..
SUM(J,Vblend(J,N,Bl))+(SUM(J,det(J,N,Bl)))*setupvol(Bl) =L= BlendRatemax(Bl);
MinOneBlend(J,N,Bl)..         Vblend(J,N,Bl) =G= Blendmin(Bl)*det(J,N,Bl);
MaxOneBlend(J,N,Bl)..         Vblend(J,N,Bl) =L= Blendmax(Bl)*det(J,N,Bl);
SS(I) ..                      Vopen(I,"1") =E= SoI(I);
SSp(J) ..                     VopenP(J,"1") =E= SpI(J);
SocR(I,N)$TC(N)..             Vopen(I,N+1)-Vclose(I,N) =E= 0;
SocP(J,N)$TC(N)..             VopenP(J,N+1)-VcloseP(J,N) =E= 0;
InvCon1(I,N)..                Vclose(I,N) =G= Vmin(I);
InvCon2(I,N)..                Vclose(I,N) =L= Vmax(I);
InvCon3(J,N)..                VcloseP(J,N) =G= VPmin(J);
InvCon4(J,N)..                VcloseP(J,N) =L= VPmax(J);
Target(J,N)$TN(N)..           VcloseP(J,N) =E= TarV(N,J);
BCOST..                       BlendCost =E=
SUM(N,SUM((I,J,Bl),C(I)*Vb(I,J,N,Bl)));
```

```
COST ..                          Z =E=
                     SUM(N,SUM(I,PenaltyBC*(SposSlack(I,N)+SnegSlack(I,N)))
                     +SUM(J,PenaltyP(N)*(SpposSlack(J,N)+SpnegSlack(J,N))));

MODEL BLENDING /ALL/;
BLENDING.optcr=0.001;
SOLVE BLENDING USING MIP MINIMIZING Z;
OPTION decimals=8;
DISPLAY Z.L, BlendCost.L, Vb.L, Vblend.L, Vopen.L, Vclose.L, VopenP.L,
VcloseP.L, det.L, SposSlack.L,SnegSlack.L,SpposSlack.L,SpnegSlack.L;
```