SIMULATION-BASED CUTTING PLANE METHODS FOR OPTIMIZATION OF SERVICE SYSTEMS

by

Júlíus Atlason

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Operations Engineering)
in The University of Michigan
2004

Doctoral Committee:

Assistant Professor Marina A. Epelman, Co-Chair
Associate Professor Shane G. Henderson, Co-Chair
Professor Stephen M. Pollock
Professor Thomas J. Schriber

ACKNOWLEDGMENTS

Thank you Marina and thank you Shane, for not only being great advisors, but also for being my good friends. I also thank you, Professors Stephen Pollock and Tom Schriber for your helpful comments and valuable discussion on my dissertation topic, and for serving on my dissertation committee.

Thank you Ryan Anthony, Ernest Fung, Paul Rosenbaum, Michael Beyer, Maria Ng, Shang Yen See and Anton Seidel, undergraduate students at Cornell University, for your excellent contribution to the numerical experiments of this thesis. Thank you Shane and Ally, for being the most gracious hosts one can imagine on my two research/pleasure trips to Ithaca.

I also thank all of the extraordinary people in the IOE department at the University of the Michigan. I would especially like to mention our chair Larry Seiford and Professor Thomas Armstrong for the generous funding from the department, Nancy Murray, Celia Eidex, Pam Linderman, Gwen Brown, Tina Blay and Mary Winter for their support, Chris Konrad, Rod Kapps and Mint for the excellent computing support, and Professors Robert Smith and Romesh Saigal for their instruction and support. I made a number of good friends among my fellow graduate students. David Kaufman, my long time office mate with whom I had endless discussions about nothing and everything, and Darby Grande, who started and finished with me, were especially helpful in the final stages of the writing.

I also thank Michael Fu, Andrew Ross, Ármann Ingólfsson and Pierre L'Ecuyer for inviting us to present this work in their conference sessions, with special thanks to Ármann for exchange of research papers and insightful discussions on my work.

I am grateful for the rich soccer community in Ann Arbor, and for being a part of the All Nations Football/Beer/Social/Family Club. Thank you Chris Grieshaber, for saving my mental and physical health by introducing me to the Bacardi Club, and for the nonillion lunch conversations on my favorite sport.

My greatest appreciation goes out to my family: my parents, my sister, my wife and my kids. Lilja Björk, thank you for your endless support, and for sharing with me all the joys and disappointments of this journey. Sóley Birna, thank you for sleeping through the night, and Björgvin Atli, thank you for sleeping through some nights, and thank you both for being the most wonderful kids.

I express my gratitude to my advisors and to the IOE department for their generous financial support throughout my Ph.D. studies. This research was supported by a number of financial sources. I wish to acknowledge National Science Foundation grants DMI 0230528 and DMI 0400287, a Horace H. Rackham School of Graduate Studies Faculty Grant, and a fellowship from the Department of Industrial and Operations Engineering.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF APPENDICES

CHAPTER I

INTRODUCTION

Computer simulation is a powerful tool for analyzing a complex system. Information about the system can be obtained through a computer model without having to make costly experiments on the real system. When a decision needs to be made about the operating policies and settings of the system, some form of optimization is required. If there are only a few possible alternatives then it may be feasible to simulate the system at each different setting in order to determine which is the best one. In other cases, the computational time needed for such total enumeration is prohibitive. Then a more sophisticated optimization technique is required.

Linear integer programming problems, along with many other mathematical programming models, are well studied and many algorithms have been developed for solving problems in this form, but a simplification of the system is often required for modeling. In general, these methods cannot be applied when the performance of the model is evaluated by simulation. Therefore, optimization with simulation when there is a large number of possible alternatives is difficult. The existing methods for optimization with simulation include such heuristics as simulated annealing and tabu search. They require little or no structure of the underlying problem, but do not fully utilize such structure when it exists.

In this dissertation we develop methods for exploiting concavity properties, or more generally, pseudoconcavity properties, of the underlying problem in order to find a good solution more efficiently. By exploiting the structure of the underlying problem

we are also able to make stronger statements about the quality of the solutions we get than the aforementioned heuristics can.

The general framework of the methods is as follows: first, we solve a mathematical program to get an initial candidate solution. A simulation is run with the solution as an input to get information about the system performance. The information from the simulation at the candidate solution is used to build additional constraints into the mathematical program. We re-solve the mathematical program to get a new candidate solution and repeat the process until some stopping criteria are satisfied.

These methods are designed to solve problems where simulation may be the only viable option for estimating system performance and a decision is chosen from a large number of possible alternative system configurations. Additionally, the performance measure should at least approximately satisfy the pseudoconcavity property.

The method of combining simulation and optimization in this way has potential applications in various service systems, such as call center staffing and emergency vehicle dispatching. In fact, the method could potentially, with proper modifications, be utilized in many other areas where simulation is an appropriate modeling tool. In this thesis we focus our attention on the inbound call center staffing problem of scheduling workers at a minimum cost, while satisfying some minimum service level requirements.

The call center staffing problem has received a great deal of attention in the literature, and so one can reasonably ask why there is a need for a computational tool of this sort. To answer that question we first need to describe the overall staffing process. There are variations on the following theme (e.g., Castillo et al., 2003), but the essential structure is sequential in nature and is as follows (Mason et al., 1998).

1. (Forecasting) Obtain forecasts of customer load over the planning horizon, which is typically one or two weeks long. The horizon is usually broken into short periods that are typically between 15 minutes and 1 hour long.

2. (Work requirements) Determine the minimum number of agents needed during each period to ensure satisfactory customer service. Service is typically measured in terms of customer waiting times and/or abandonment rates in the queue.

3. (Shift construction) Select staff shifts that cover the requirements. This problem is usually solved through a set-covering integer program; see Mason et al. (1998) for more details.

4. (Rostering) Allocate employees to the shifts.

The focus in this thesis is on Steps 2 and 3. Steps 1 and 4 are not considered further.

Step 2 is usually accomplished through the use of analytical results for simple queuing models. Green et al. (2001) coined the term SIPP (stationary, independent, period by period) to describe the general approach. In the SIPP approach, each period of the day is considered independently of other periods, the arrival process is considered to be stationary in that period, and one approximates performance in the period by a steady-state performance measure that is usually easily obtained from analytical results for particular queuing models. Heuristics are used to select input parameters for each period that yield a close match between the predictions and reality. For a wide variety of queuing models, this procedure results in some form of the "square-root rule," which is a rule of thumb that provides surprisingly successful staffing level suggestions. See, e.g., Borst et al. (2004); Kolesar and Green (1998); Jennings et al. (1996) for more on the square-root rule.

The SIPP approach is appealing from the standpoint of computational tractability and due to the insights it provides. However, there are cases where the SIPP approach does not do as well as one might hope (Green et al., 2001, 2003). This can occur, for example, when the use of steady-state measures to represent performance over a short period is inappropriate. See Whitt (1991) for more on this point. Moreover,

call centers can be somewhat complex in structure, and this complexity can make it difficult to identify a queuing model of the center that is both mathematically tractable and a reasonable match to the true system. In such cases, simulation is a viable alternative. Indeed, simulation is now increasingly used in Step 2 (see Section VIII of Mandelbaum, 2003, for many examples) and commercial simulation packages, specially designed for call centers (e.g., Rockwell Software's Arena Contact Center Edition), are available.

Further motivation for the use of simulation involves the linkage between staffing decisions in adjacent periods. Boosting staffing levels in one period can often help in reducing workload in subsequent periods, so that there can be linkage in performance between different periods. Such linkage can imply that there are multiple solutions to the work requirements problem that can offer valuable flexibility in Step 3. Traditional queuing approaches are not satisfactory in the presence of such linkage between periods, and in such cases one turns to simulation or other numerical methods. Indeed, Green et al. (2001, 2003) solve a system of ordinary differential equations through numerical integration to get the "exact" results for their models in order to compare the performance of various heuristics.

Assuming that one uses simulation or some other numerical method to predict performance in the call center, one then needs to devise a method to guide the selection of potential staffing levels to be evaluated through simulation. There have been several suggestions in the literature, all of which explicitly capture the linkage between periods in an attempt to realize cost savings. Like Green et al. (2001, 2003), Ingolfsson et al. (2002) use numerical integration to compute service level performance for a proposed set of staffing levels, and a genetic algorithm to guide the search. Ingolfsson et al. (2003) again use a numerical method to compute service level performance, and integer programming to guide the search. Castillo et al. (2003) devise a method for randomly generating sets of staff shifts that can be expected to perform well,

then use simulation to evaluate the service level performance of each set of generated staff shifts, and finally, plot the cost versus service level of the potential solutions to identify an efficient frontier. Henderson and Mason (1998) proposed a method that uses simulation to evaluate service level performance of a proposed set of shifts, and uses integer programming in conjunction with Kelley's (Kelley, Jr., 1960) cutting plane method to guide the search.

1.1 Dissertation Outline

In Chapter II we formulate the problem of minimizing staffing costs in a call center while maintaining an acceptable level of service. The service level in the call center is estimated via simulation, so the call center staffing problem is then a simulation optimization problem. We adopt the "sample-average approximation" (SAA) approach (see Shapiro, 2003; Kleywegt et al., 2001) for solving simulation-optimization problems. This approach, specialized to our setting, is as follows. One first generates the simulation input data for n independent replications of the operations of the call center over the planning horizon. This data includes call arrival times, service times and so forth. The simulation data is fixed, and one then solves a deterministic optimization problem that chooses staffing levels so as to minimize staffing cost, while ensuring that average service *computed only over the generated realizations* is satisfactory.

Solving the SAA problem is non-trivial, since each function evaluation requires a simulation. In Chapter III we combine the cutting plane method of Henderson and Mason (1998) with the sample average approximation approach in the simulation-based Kelley's cutting plane method (SKCPM). We establish its convergence properties and give an implementation. This implementation identified two of the key issues, to be discussed next, of the cutting plane method and motivated the research in Chapters IV and V.

First, the method requires some form of gradient information to generate the cutting planes. The service level functions are discrete and, therefore, a gradient does not exist. Furthermore, the functions are estimated via simulation, so an analytical expression of the functions are not available. In Chapter IV we discuss the three most prominent gradient estimation techniques in the simulation literature: the method of finite differences, the likelihood ratio method and infinitesimal perturbation analysis. We show how each method can be applied in our setting and compare the three methods through a numerical example. The finite difference method gives the best gradient estimates, but is at the same time the most computationally expensive.

Second, the approach in Chapter III relies on an assumption that service in a period is concave and componentwise-increasing as a function of the staffing level vector. To understand this assumption, consider a single period problem. Increasing the staffing level should lead to improved performance. Furthermore, one might expect "diminishing returns" as the staffing level increases, so that performance would be concave in staffing level. Empirical results suggest that this intuition is correct, at least for sufficiently high staffing levels. But for low staffing levels, the empirical results suggest that performance is increasing and *convex* in the staffing level. So performance appears to follow an "S-shaped" curve (see Ingolfsson et al., 2003, and Chapter III) in one dimension. This non-concavity can cause the SKCPM to cut off feasible solutions, and the problem can be so severe as to lead to the algorithm suggesting impractical staffing plans. Nevertheless, the ability of the SKCPM to efficiently sift through the combinatorially huge number of potential staffing plans is appealing.

One might ask whether there is a similar optimization approach that can efficiently search through alternative staffing plans while satisfactorily dealing with S-shaped curves and their multidimensional extensions. In Chapter V we combine a relaxation on the assumption of concavity, i.e., pseudoconcavity, and additional techniques to

handle both the S-shaped curves alluded to above, as well as multidimensional behavior seen in the numerical experiments (see, e.g., Figure 5.2). We developed the simulation-based analytic center cutting plane method for solving the SAA of the call center staffing problem. We prove under which conditions the method converges and include extensive numerical experiments, which show that the method is a robust one and often outperforms the traditional heuristics based on analytical queuing methods.

Finally, in Chapter VI we provide concluding remarks and some directions for future research related to this thesis.

1.2 Contribution

We view the primary contributions of the dissertation as follows.

1. We demonstrate the potential of bringing simulation and traditional optimization methods together. Our methods are developed for the call center staffing problem, but our ideas and results will hopefully push research along these lines in other application areas.

2. We apply the method to the call center staffing problem, and our research lends valuable insight into the problem of scheduling employees at a low cost while maintaining an acceptable level of service, by identifying, and tackling, many of the key difficulties in the problem. Our computational results further indicate that the analytic center cutting plane method in Chapter V can be useful when traditional staffing methods fail.

3. The implementation of the methods is a challenging task. There is no standard software package that combines simulation and optimization in this way. Our successful implementation should be encouraging for other researchers as well as for makers of simulation and optimization software.

CHAPTER II

PROBLEM FORMULATION AND SAMPLE AVERAGE APPROXIMATION

2.1 Introduction

In this chapter we formulate the call center staffing problem that serves as a motivating example for our work. We consider the problem of minimizing staffing costs while maintaining an acceptable level of service, where the service level can only be estimated via simulation. The random nature of the problem and the absence of an algebraic form for the service level function makes the optimization challenging. We use sampling to get an estimate of the service level function, and optimize over constraints on the sample average approximation (SAA). An important question is whether the solution to the sample average approximation problem converges to a solution to the original problem as the sample size increases, and if so, how fast.

We apply the strong law of large numbers to prove conditions for almost sure convergence and apply a result due to Dai et al. (2000) to prove an exponential rate of convergence of the optimal solutions as the sample size increases. Vogel (1994) proved almost sure convergence in a similar setting, but we include proofs for reasons listed in Section 2.3.1. Kleywegt et al. (2001) established conditions for an exponential rate of convergence of the probability that the solution to the SAA problem is exactly the solution to the original discrete optimization problem when the expected value is in the objective. Vogel (1988) proved a polynomial rate of convergence in a similar setting, but under weaker conditions than we require. Homem-de-Mello

(2003) established convergence results for a variable sample method, which is similar to the SAA approach, but uses a different sample for each function evaluation. The optimization of SAA problems has also been studied in the simulation context (Chen and Schmeiser, 2001; Healy and Schruben, 1991; Robinson, 1996; Rubenstein and Shapiro, 1993).

The chapter is organized as follows. We formulate the call center staffing problem and its sample average approximation in Section 2.2. The convergence and the rate of convergence of the solutions of the SAA problem to solutions of the original problem are proved in Section 2.3.

2.2 Formulation of the Call Center Staffing Problem

The problem of determining optimal staffing levels in a call center (see, e.g., Thompson, 1997) is a motivating example for our work. We consider a somewhat simplified version of the real world problem for the purpose of developing the theory. We assume, for instance, that there is an unlimited number of trunk lines, one customer class, one type of agent and no abandonments. The decision maker faces the task of creating a collection of tours (work schedules) of low cost that together ensure a satisfactory service level.

A tour is comprised of several shifts and has to observe several restrictions related to labor contracts, management policies, etc. We divide the planning horizon (typically a day or a week) into small periods (15-60 minutes). The set of permissible tours can be conveniently set up in a matrix (see Dantzig, 1954). More specifically we have

$$A_{ij} = \begin{cases} 1 & \text{if period } i \text{ is included in tour } j \\ 0 & \text{otherwise.} \end{cases}$$

From the above we see that a column in A represents a feasible tour and a row in

A represents a specific period. We let p be the total number of periods and m be the number of feasible tours. If we let $x \in \mathbb{Z}_+^m$ be a vector where the j^{th} component represents the number of employees that work tour j, then $Ax = y \in \mathbb{Z}_+^p$ is a vector where the i^{th} component of y corresponds to the number of employees that work in period i. We make the following natural assumption that every period is covered by at least one tour.

Assumption 2.1. For every period i there is at least one tour j such that $A_{ij} = 1$.

The cost function is usually relatively straightforward to calculate. We can calculate the cost of each tour (salary costs, appeal to employees, etc.), and multiply by the number of employees working each tour to get the overall cost. Let c be the cost vector, where c_j is the cost per employee working tour j. Define the cost function

$$\begin{aligned} f(y) = \quad & \min \quad c^T x \\ & \text{s.t.} \quad Ax \quad \geq y \\ & \qquad\quad x \quad \geq 0 \text{ and integer.} \end{aligned} \tag{2.1}$$

It follows by Assumption 2.1 that (2.1) is feasible for any y. The value $f(y)$ gives the minimum cost set of shifts that can cover the desired work requirements vector y. We make the following assumption on the cost vector.

Assumption 2.2. The cost vector c is positive and integer valued.

Assumption 2.2 implies that $f(y)$ is integer valued and, moreover, since c is positive and the entries in A are either 0 or 1, the z-level set of f,

$$\{y \geq 0 \text{ and integer} : \exists x \geq 0 \text{ and integer}, Ax \geq y, c^T x \leq z\},$$

is finite for any $z \in \mathbb{R}$.

The management of a call center needs some criteria to follow when they decide on a set of staffing levels. It is not unusual in practice to determine the staffing levels from a service level perspective. In an emergency call center, for example, it might

be required that 90% of received calls should be answered within 10 seconds. We let $l \in [0,1]^p$ be the vector whose i^{th} component is the minimum acceptable service level in period i, e.g. 90%. Since, for example, the arrival and service times of customers are not known but are random, the service level in each period will be a random variable. Let Z, a random vector, denote all the random quantities in the problem and let $z^1, \ldots, z^n$ denote independent realizations of Z. Let $N_i(Z)$ be the number of calls received in period i and let $S_i(y, Z)$ be the number of those calls answered within a pre-specified time limit, for example 10 seconds, based on the staffing level y. The fraction of customers receiving adequate service in period i in the long run is then

$$\lim_{n\to\infty} \frac{\sum_{d=1}^{n} S_i(y, z^d)}{\sum_{d=1}^{n} N_i(z^d)} = \frac{\lim_{n\to\infty} n^{-1} \sum_{d=1}^{n} S_i(y, z^d)}{\lim_{n\to\infty} n^{-1} \sum_{d=1}^{n} N_i(z^d)}.$$

If $E[N_i(Z)] < \infty$ then the strong law of large numbers can be applied separately to both the numerator and denominator of this expression, and then the desired long-run ratio is $E[S_i(y, Z)]/E[N_i(Z)]$. Thus, $E[S_i(y, Z)]/E[N_i(Z)] \geq l_i$ is a natural representation of the service level constraint (excluding the pathological case $E[N_i(Z)] = 0$) in period i. If we define $G_i(y, Z) := S_i(y, Z) - l_i N_i(Z)$ then we can conveniently write the service level constraint as $E[G_i(y, Z)] \geq 0$. Define

$$g_i(y) := E[G_i(y, Z)] \tag{2.2}$$

as the expected service level in period i as a function of the server allocation vector y and let $g : \mathbb{R}^p \to \mathbb{R}^p$ be a function whose i^{th} component is g_i. Finally, since an agent does not hang up on a call that he or she has already started working on we assume that if a server is still in service at the end of a period it finishes that service before becoming unavailable.

We are now ready to formulate the problem of minimizing staffing costs subject

to satisfying a minimum service level in each period. It is

$$\begin{array}{lll} \min & f(y) & \\ \text{subject to} & g(y) \geq 0 & \\ & y \geq 0 \text{ and integer,} & \end{array} \tag{2.3}$$

and note that problem (2.3) is equivalent to

$$\begin{array}{lll} \min & c^T x & \\ \text{subject to} & Ax \geq y & \\ & g(y) \geq 0 & \\ & x, y \geq 0 & \text{and integer.} \end{array}$$

The functions $g_i(y)$ are expected values, and the underlying model is typically so complex that an algebraic expression for $g(y)$ can not be easily obtained. Therefore, simulation could be the only viable method for estimating $g(y)$. In the next section we formulate an optimization problem which is an approximation of (2.3) obtained by replacing the expected values by sample averages and prove statements about the solutions of the approximation problem as solutions of the original problem (2.3).

2.3 Sample Average Approximation of the Call Center Staffing Problem

In this thesis we assume that the algebraic form of the service level function $g(y)$ is not available, and that its value is estimated using simulation. Suppose we run a simulation with sample size n, where we independently generate the realizations $\{z^d\}_{d=1}^n$ from the distribution of Z, to get an estimate of the expected value of $g(y)$. Let

$$\bar{g}(y;n) = \frac{1}{n}\sum_{d=1}^{n} G(y, z^d)$$

be the resulting estimates and let $\bar{g}_i(y;n)$ denote the i^{th} component of $\bar{g}(y;n)$. We use this notation to formulate the SAA problem

$$\begin{array}{lll} \min & f(y) & \\ \text{subject to} & \bar{g}(y;n) & \geq 0 \\ & y & \geq 0 \text{ and integer.} \end{array} \tag{2.4}$$

In Section 2.3.1 we show, by using the strong law of large numbers (SLLN), that the set of optimal solutions of the SAA problem (2.4) is a subset of the set of optimal solutions for the original problem (2.3) with probability 1 (w.p.1) as the sample size gets large. Furthermore, we show in Section 2.3.2 that the probability of this event approaches 1 exponentially fast when we increase the sample size. These results require the existence of at least one optimal solution for the original problem to satisfy the expected service level constraints with strict inequality, but this regularity condition can be easily justified for practical purposes as will be discussed later.

2.3.1 Almost Sure Convergence of Optimal Solutions of the Sample Average Approximation Problem

The results in this section may be established by specializing the results in Vogel (1994). We choose to provide direct proofs in this section for 3 main reasons:

1. The additional structure in our setting allows a clearer statement and proof of the results.

2. The proofs add important insight into why solving the SAA problem is a sensible approach.

3. The proofs serve as an excellent foundation to develop an understanding of the "rate of convergence" results that follow in Section 2.3.2.

We are interested in the properties of the optimal solutions of (2.4) as the sample size n gets large. It turns out, by an application of the SLLN, that any optimal solution of (2.3) that satisfies $g(y) > 0$, i.e., $g_i(y) > 0$ for all i, is an optimal solution

of (2.4) with probability 1 (w.p.1) as n goes to infinity. We introduce additional notation before we prove this. Let

$$\begin{aligned} \bar{g}(y;\infty) &:= \lim_{n\to\infty} \bar{g}(y;n), \\ F^* &:= \text{the optimal value of (2.3)} \end{aligned}$$

and define the sets

$$\begin{aligned} Y^* &:= \text{the set of optimal solutions to (2.3)}, \\ Y_0^* &:= \{y \in Y^* : g(y) > 0\}, \\ Y_1 &:= \{y \in \mathbb{Z}_+^p : f(y) \leq F^*, g(y) \ngeq 0\}, \\ Y_n^* &:= \text{the set of optimal solutions to (2.4)}. \end{aligned}$$

Note that Y_1 is the set of solutions to (2.3) that have the same or lower cost than an optimal solution, and satisfy all constraints except the service level constraints. We are concerned with solutions in this set since they could be feasible (optimal) for the SAA problem (2.4) if the difference between the sample average, $\bar{g}(\cdot;n)$, and g is sufficiently large. We show that when Y_0^* is not empty, $Y_0^* \subseteq Y_n^* \subseteq Y^*$ for all n large enough w.p.1.[1] The sets Y^*, Y_n^* and Y_0^* are finite by Assumption 2.2. (The sets Y^*, Y_n^* and Y_0^* can be empty). Furthermore, if Y^* is nonempty then $F^* < \infty$ and then, again by Assumption 2.2, the set Y_1 is finite.

We start with two lemmas. The first one establishes properties of $\bar{g}(y;\infty)$ by repeatedly applying the SLLN. The second shows that solutions to (2.3) satisfying $g(y) > 0$, and infeasible solutions, will be feasible and infeasible, respectively, w.p.1 for problem (2.4) when n gets large. The only condition $g(y)$ has to satisfy is that it has to be finite for all $y \in \mathbb{Z}_+^p$. That assumption is easily justified by noting that the absolute value of each component of $g(y)$ is bounded by the expected number of

[1] We say that property $E(n)$ holds for all n large enough w.p.1 if and only if $P[\exists N < \infty : E(n)$ holds $\forall\, n \geq N] = 1$. (Here N should be viewed as a random variable.) Sometimes such statements are communicated by saying that $E(n)$ holds *eventually*.

arrivals in that period, which would invariably be finite in practice.

Define

$$\|g\| = \max_{y \in \mathbb{Z}_+^p} \|g(y)\|_\infty = \max_{y \in \mathbb{Z}_+^p} \max_{i=1,\ldots,p} |g_i(y)|.$$

Lemma 2.1.

1. *Suppose that* $\|g(y)\|_\infty < \infty$ *for some fixed* $y \in \mathbb{Z}_+^p$. *Then* $\bar{g}(y;\infty) = g(y)$ *w.p.1.*
2. *Suppose that* $\|g\| < \infty$ *and* $\Gamma \subset \mathbb{Z}_+^p$ *is finite. Then* $\bar{g}(y;\infty) = g(y)\ \forall\ y \in \Gamma$ *w.p.1.*

Proof:

1. The SLLN (Theorem B.1) gives $\bar{g}_i(y;\infty) = g_i(y)$ w.p.1. So $P[\bar{g}(y;\infty) = g(y)] \geq 1 - \sum_{i=1}^p P[\bar{g}_i(y;\infty) \neq g_i(y)] = 1$ by Boole's inequality; see Equation (B.1).
2. Note that $P[\bar{g}(y;\infty) = g(y)\ \forall\ y \in \Gamma] \geq 1 - \sum_{y \in \Gamma} P[\bar{g}(y;\infty) \neq g(y)] = 1$ since Γ is finite. □

Lemma 2.2. *Suppose that* $\|g\| < \infty$ *and that Assumption 2.2 holds. Then*

1. $\bar{g}(y;n) \geq 0\ \forall\ y \in Y_0^*$ *for all* n *large enough w.p.1.*
2. *All* $y \in Y_1$ *are infeasible for the SAA problem* (2.4) *for all* n *large enough w.p.1 if* $F^* < \infty$.

Proof:

1. The result is trivial if Y_0^* is empty, so suppose it is not. Let

$$\epsilon = \min_{y \in Y_0^*} \min_{i \in \{1,\ldots,p\}} \{g_i(y)\}.$$

Then $\epsilon > 0$ by the definition of Y_0^*. Let

$$N_0 = \inf\{n_0 : \max_{y \in Y_0^*} \|\bar{g}(y;n) - g(y)\|_\infty < \epsilon\, \forall\, n \geq n_0\}.$$

Then $\bar{g}(y;n) \geq 0\ \forall\ y \in Y_0^*\ \forall\ n \geq N_0$. The set Y_0^* is finite, so $\lim_{n\to\infty} \bar{g}(y;n) = g(y)\ \forall\ y \in Y_0^*$ w.p.1 by part 2 of Lemma 2.1. Therefore, $N_0 < \infty$ w.p.1.

2. The result is trivial if Y_1 is empty, so suppose it is not. Let

$$\epsilon = \min_{y \in Y_1} \max_{i \in \{1,\ldots,p\}} \{-g_i(y)\}.$$

Then $\epsilon > 0$, since $g_i(y) < 0$, for at least one $i \in \{1, \ldots, p\}$ $\forall$ $y \in Y_1$. Let

$$N_1 = \inf\{n_1 : \max_{y \in Y_1} \|g(y) - \bar{g}(y; n)\|_\infty < \epsilon \ \forall \ n \geq n_1\}$$

and then all $y \in Y_1$ are infeasible for (2.4) for all $n \geq N_1$. The set Y_1 is a finite by Assumption 2.2 and since $F^* < \infty$, so

$$\lim_{n \to \infty} \bar{g}(y; n) = g(y) \ \forall \ y \in Y_1 \text{ w.p.1}$$

by part 2 of Lemma 2.1. Therefore, $N_1 < \infty$ w.p.1. □

Lemma 2.2 shows that all the "interior" optimal solutions for the original problem are eventually feasible for the SAA problem and remain so as the sample size increases. Furthermore, all solutions that satisfy the constraints that are common for both problems, but not the service level constraints, and have at most the same cost as an optimal solution, eventually become infeasible for the SAA problem. Hence, we have the important result that for a large enough sample size an optimal solution for the SAA problem is indeed optimal for the original problem.

Theorem 2.3. *Suppose that $\|g\| < \infty$ and that Assumption 2.2 holds. Then $Y_0^* \subseteq Y_n^*$ for all n large enough w.p.1. Furthermore, if Y_0^* is nonempty then $Y_0^* \subseteq Y_n^* \subseteq Y^*$ for all n large enough w.p.1.*

Proof: The first inclusion holds trivially if Y_0^* is empty, so assume that Y_0^* is not empty, and hence $F^* < \infty$. On each sample path let $N = \sup\{N_0, N_1\}$, where N_0 and N_1 are the same as in Lemma 2.2. When $n \geq N$ we know that all $y \in Y_0^*$ are feasible for (2.4) and that all $y \in Y_1$ are infeasible for (2.4). Hence, all $y \in Y_0^*$ are optimal for (2.4) and no $y \notin Y^*$ is optimal for (2.4) whenever $n \geq N$. Thus, $Y_0^* \subseteq Y_n^* \subseteq Y^*$ for all $n \geq N$. Finally, $P[N < \infty] = P[N_0 < \infty, N_1 < \infty] \geq P[N_0 < \infty] + P[N_1 <$

$\infty] - 1 = 1$. □

Corollary 2.4. *Suppose that* $\|g\| < \infty$*, Assumption 2.2 holds and that* (2.3) *has a unique optimal solution,* y^**, such that* $g(y^*) > 0$*. Then* y^* *is the unique optimal solution for* (2.4) *for all* n *large enough w.p.1.*

Proof: In this case $Y_0^* = Y^* = \{y^*\}$ and the result follows from Theorem 2.3. □

The conclusion of Theorem 2.3 relies on existence of an "interior" optimal solution for the original problem. A simple example illustrates how the conclusion can fail if this requirement is not satisfied. Let Z be a uniform random variable on $[-0.5, 0.5]$ and consider the following problem:

$$\begin{array}{lrcl} \min & y & & \\ \text{subject to} & y & \geq & |E[Z]| \\ & y & \geq & 0 \quad \text{and integer.} \end{array}$$

Then $y^* = 0$ for this problem since $E[Z] = 0$. We form the SAA problem by replacing $E[Z]$ with $\bar{Z}(n)$, the sample average of n independent realizations of Z. Then $0.5 > |\bar{Z}(n)| > 0$ w.p.1 for all $n > 0$ and thus we get that $y_n^* = 1$ w.p.1.

Remark 2.1. The existence of an "interior" optimal solution is merely a regularity condition. In reality it is basically impossible to satisfy the service level constraints in any period exactly, since the feasible region is discrete. Even if this occurred, we could subtract an arbitrarily small positive number, say ε, from the right hand side of each service level constraint and solve the resulting ε-perturbed problem. Then all solutions with $g_i(y) = 0$ for some i satisfy $g_i(y) > -\varepsilon$ and it is sufficient for the problem to have an optimal solution (not necessarily satisfying $g(y) > 0$) for Theorem 2.3 to hold.

Remark 2.2. It may possible to replace the constraint $\bar{g}(y; n) \geq 0$ with $\bar{g}(y; n) \geq -\varepsilon(n)$, and prove a similar result as in Theorem 2.3 under stronger conditions on $g(y)$, by letting $\varepsilon(n) \to 0$ at a slow enough rate as $n \to \infty$.

Remark 2.1 also applies in the next subsection where we prove an exponential rate of convergence as the sample size increases.

2.3.2 Exponential Rate of Convergence of Optimal Solutions of the Sampled Problems

In the previous subsection we showed that we can expect to get an optimal solution for the original problem (2.3) by solving the SAA problem (2.4) if we choose a large sample size. In this section we show that the probability of getting an optimal solution this way approaches 1 exponentially fast as we increase the sample size. We use large deviations theory and a result due to Dai et al. (2000) to prove our statement. Vogel (1988) shows, under weaker conditions, that the feasible region of a sample average approximation of a chance constraint problem approaches the true feasible region at a polynomial rate and conjectures, without giving a proof, that an exponential rate of convergence is attainable under similar conditions to those we impose.

The following theorem is an intermediate result from Theorem 3.1 in Dai et al. (2000):

Theorem 2.5. *Let $H : \mathbb{R}^p \times \Omega \to \mathbb{R}$ and assume that there exist $\gamma > 0$, $\theta_0 > 0$ and $\eta : \Omega \to \mathbb{R}$ such that*

$$|H(y, Z)| \leq \gamma\eta(Z), \qquad E[e^{\theta\eta(Z)}] < \infty,$$

for all $y \in \mathbb{R}^p$ and for all $0 \leq \theta \leq \theta_0$, where Z is a random element taking values in the space Ω. Then for any $\delta > 0$, there exist $a > 0$, $b > 0$, such that for any $y \in \mathbb{R}^p$

$$P[|h(y) - \bar{h}(y; n)| \geq \delta] \leq ae^{-bn},$$

for all $n > 0$, where $h(y) = E[H(y, Z)]$, and $\bar{h}(y; n)$ is a sample mean of n independent and identically distributed realizations of $H(y, Z)$.

In our setting take $H(y, Z) = G_i(y, Z)$ and note that $|G_i(y, Z)| \leq N_i(Z)$, where N_i is the number of calls received in period i. If the arrival process is, for example, a (nonhomogeneous or homogeneous) Poisson process, which is commonly used to model incoming calls at a call center, then N_i satisfies the condition of Theorem 2.5 since it is a Poisson random variable, which has a finite moment generating function (Ross, 1996).

Before we prove the exponential rate of convergence we prove a lemma that shows

that for any n, $Y_0^* \subseteq Y_n^* \subseteq Y^*$ precisely when all the solutions in Y_0^* are feasible for the SAA problem and all infeasible solutions for (2.3) that are equally good or better, i.e., are in the set Y_1, are also infeasible for (2.4).

Lemma 2.6. *Let $n > 0$ be an arbitrary integer and let Y_0^* be nonempty. The properties*

1. *$\bar{g}(y; n) \geq 0 \; \forall \; y \in Y_0^*$, and*
2. *$\bar{g}(y; n) \ngeq 0 \; \forall \; y \in Y_1$*

hold if and only if $Y_0^* \subseteq Y_n^* \subseteq Y^*$.

Proof: Suppose properties 1 and 2 hold. Then by property 1 all $y \in Y_0^*$ are feasible for (2.4) and the optimal value of (2.4) is at most F^* since Y_0^* is nonempty. By property *2* there are no solutions with a lower objective that are feasible for (2.4), so $Y_0^* \subseteq Y_n^*$. By property 2, no solutions outside Y^* with objective value equal to F^* are feasible for (2.4). Hence, $Y_0^* \subseteq Y_n^* \subseteq Y^*$.

Suppose $Y_0^* \subseteq Y_n^* \subseteq Y^*$. Then F^* is the optimal value for (2.4). Now, since all $y \in Y_0^*$ are optimal for (2.4) they are also feasible for (2.4) and property 1 holds. All $y \in Y_1$ are infeasible for (2.4) since $Y_n^* \subseteq Y^*$ and therefore property 2 holds. □

Theorem 2.7. *Suppose $G_i(y, Z)$ satisfies the assumptions of Theorem 2.5 for all $i \in \{1, \ldots, p\}$, Assumption 2.2 holds and that Y_0^* is nonempty. Then there exist $\alpha > 0, \beta > 0$ such that*

$$P[Y_0^* \subseteq Y_n^* \subseteq Y^*] \geq 1 - \alpha e^{-\beta n}.$$

Proof:

Define

$$\delta_1 := \min_{y \in Y_0^*} \min_{i \in \{1,\ldots,p\}} \{g_i(y)\},$$

$$i(y) := \arg \max_{i \in \{1,\ldots,p\}} \{-g_i(y)\},$$

$$\delta_2 := \min_{y \in Y_1} \{-g_{i(y)}(y)\}, \text{ and}$$

$$\delta := \min\{\delta_1, \delta_2\}.$$

Here $\delta_1 > 0$ is the minimal amount of slack in the constraints "$g(y) \geq 0$" for any solution $y \in Y_0^*$. Similarly $\delta_2 > 0$ is the minimal violation in the constraints "$g(y) \geq 0$" induced by any solution $y \in Y_1$. Thus,

$$
\begin{aligned}
&P[Y_0^* \subseteq Y_n^* \subseteq Y^*] \\
&= P[\bar{g}(y;n) \geq 0 \; \forall \; y \in Y_0^*, \bar{g}(y;n) \ngeq 0 \; \forall \; y \in Y_1] && (2.5) \\
&= 1 - P[\bar{g}(y;n) \ngeq 0 \text{ for some } y \in Y_0^* \text{ or } \bar{g}(y;n) \geq 0 \text{ for some } y \in Y_1] \\
&\geq 1 - \sum_{y \in Y_0^*} \sum_{i=1}^{p} P[\bar{g}_i(y;n) < 0] - \sum_{y \in Y_1} P[\bar{g}(y;n) \geq 0] && (2.6) \\
&\geq 1 - \sum_{y \in Y_0^*} \sum_{i=1}^{p} P[|\bar{g}_i(y;n) - g_i(y)| \geq \delta] - \sum_{y \in Y_1} P[|\bar{g}_{i(y)}(y;n) - g_{i(y)}(y)| \geq \delta] && (2.7) \\
&\geq 1 - \sum_{y \in Y_0^*} \sum_{i=1}^{p} a_i e^{-b_i n} - \sum_{y \in Y_1} a_{i(y)} e^{-b_{i(y)} n} && (2.8) \\
&\geq 1 - \alpha e^{-\beta n}.
\end{aligned}
$$

Here $\alpha = |Y_0^*| \sum_{i=1}^{p} a_i + \sum_{y \in Y_1} a_{i(y)}$ and $\beta = \min_{i \in \{1,\ldots,p\}} b_i$, where $|Y_0^*|$ is the cardinality of the set Y_0^*. The sets Y_0^* and Y_1 are finite by Assumption 2.2, so $\alpha < \infty$. Equation (2.5) follows by Lemma 2.6. Equation (2.6) is Boole's inequality; see (B.1). Equation (2.7) follows since $P[\bar{g}(y;n) \geq 0] \leq P[\bar{g}_{i(y)}(y;n) \geq 0]$ and $g_i(y) \geq \delta_1 \geq \delta$ for $y \in Y_0^*$ and $g_{i(y)}(y) \geq \delta_2 \geq \delta$ for $y \in Y_1$. Equation (2.8) follows from Theorem 2.5. □

The case where Y_0^* is empty but Y^* is not would almost certainly never arise in practice. But in such a case one can solve an ε-perturbation of (2.4) as described in Remark 2.1, and the results of Theorem 2.7 hold for $0 \leq \varepsilon < \delta$.

Justified by the results of this chapter, in the remainder of the thesis we discuss how to solve the SAA problem (2.4).

CHAPTER III

THE SIMULATION-BASED KELLEY'S CUTTING PLANE METHOD

3.1 Introduction

In Chapter II we formulated the call center staffing problem (2.3) and its sample average approximation (SAA) problem (2.4). In this chapter we present a conceptual cutting plane method for solving the SAA problem. This method was proposed by Henderson and Mason (1998) and combines simulation and integer programming in an iterative cutting plane algorithm. The algorithm relies on the concavity of the problem constraints, but in our algorithm we have a built-in subroutine for detecting nonconcavity, so the method is robust.

The method is based on the one developed by Kelley, Jr. (1960) and solves a linear (integer) program to obtain the staffing levels, and the solution is used as an input for a simulation to calculate the service level. If the service level is unsatisfactory, a constraint is added to the linear program and the process is repeated. If the service level is satisfactory the algorithm terminates with an optimal solution of the SAA problem.

Kelley's cutting plane method applies to minimization problems where both the objective function and feasible region (of the continuous relaxation of the integer problem) need to be convex. The costs in the call center problem are linear and we assume that the service level function is concave, so that (see Equation (2.3)), the feasible region, relaxing the integer restriction, is convex. Since the service level

function is unknown beforehand, we need to incorporate a mechanism into the method to verify that the concavity assumption holds. In Section 3.4 we present a numerical method for checking concavity of a function, when the function values and, possibly, gradients are only known at a finite number of points.

Cutting plane methods have been successfully used to solve two stage stochastic linear programs. In many applications the sample space becomes so large that one must revert to sampling to get a solution (Birge and Louveaux, 1997; Infanger, 1994). The general cutting plane algorithm for two stage stochastic programming is known as the L-Shaped method (van Slyke and Wets, 1969) and is based on Benders decomposition (Benders, 1962). Stochastic Decomposition (Higle and Sen, 1991) for solving the two stage stochastic linear program starts with a small sample size, which is increased as the algorithm progresses and gets closer to a good solution. Stochastic Decomposition could also be applied in our setting, but that is not within the scope of this thesis.

Morito et al. (1999) use simulation in a cutting-plane algorithm to solve a logistic system design problem at the Japanese Postal Service. Their problem is to decide where to sort mail provided that some post offices have automatic sorting machines but an increase in transportation cost and handling is expected when the sorting is more centralized. The algorithm proved to be effective for this particular problem and found an optimal solution in only 3 iterations where the number of possible patterns (where to sort mail for each office) was 2^{30}. Their discussion of the algorithm is ad hoc, and they do not discuss its convergence properties.

Koole and van der Sluis (2003) developed a local search algorithm for a call center staffing problem with a global service level constraint. When the service level constraint satisfies a property called multimodularity their algorithm is guaranteed to terminate with a global optimal solution. There are, however, examples where the service level constraint, as defined in Chapter II, is not multimodular even for

nondecreasing and concave service level functions.

The chapter is organized as follows. In Section 3.2 we discuss the concavity assumption in more detail. We present the cutting plane algorithm and its convergence properties in Section 3.3. The numerical method for checking concavity is described in Section 3.4 and an implementation of the overall method is described in Section 3.5.

3.2 Concave Service Levels and Subgradients

Intuitively, we would expect that the service level increases if we increase the number of employees in any given period. We also conjecture that the marginal increase in service level decreases as we add more employees. If these speculations are true then $g_i(y)$ is increasing and concave in each component of y for all i. In this chapter, we will make the stronger assumption that $g_i(y)$ and $\bar{g}_i(y;n)$ are increasing componentwise and jointly concave in y, for all i. Our initial computational results suggest that this is a reasonable assumption, at least within a region containing practical values of y (see Section 3.5). Others have also studied the convexity of performance measures of queuing systems. Akşin and Harker (2001) show that the throughput of a call center is stochastically increasing and directional concave in the sample path sense as a function of the allocation vector y in a similar setting. Analysis of the steady state waiting time of customers in an $M/M/s$ queue shows that its expected value is a convex and decreasing function of the number of servers s (Dyer and Proll, 1977), its expected value is convex and increasing as a function of the arrival rate (Chen and Henderson, 2001) and its distribution function evaluated at any fixed value of the waiting time, is concave and decreasing as a function of the arrival rate (Chen and Henderson, 2001). See other references in Chen and Henderson (2001) for further studies in this direction.

If the concavity assumption holds (we will discuss the validity of this assumption later), then we can approximate the service level function with piecewise linear con-

cave functions, which can be generated as described below. The following definition is useful:

Definition 3.1. (In Rockafellar, 1970, p. 308) Let $\hat{y} \in \mathbb{R}^p$ be given and $h : \mathbb{R}^p \to \mathbb{R}$ be concave. The vector $q(\hat{y}) \in \mathbb{R}^p$ is called a *subgradient* of h at $\hat{y}$ if

$$h(y) \leq h(\hat{y}) + q(\hat{y})^T(y - \hat{y}) \; \forall \, y \in \mathbb{R}^p. \tag{3.1}$$

The term "supergradient" might be more appropriate since the hyperplane

$$\{(y, p(y)) : p(y) = h(\hat{y}) + q(\hat{y})^T(y - \hat{y}) \; \forall y \in \mathbb{R}\}$$

lies "above" the function h, but we use "subgradient" to conform with the literature. A concave function has at least one subgradient at every point in the interior of its convex domain (see Theorem 3.2.5 in Bazaraa et al., 1993). The notion of concavity and subgradients is defined for functions of continuous variables, but we are dealing with functions of integer variables here. Therefore, we need to define concavity of discrete functions; see the illustration in Figure 3.1.

Definition 3.2. The function $h : \mathbb{Z}^p \to \mathbb{R}$ is *discrete concave* if no points $(x, h(x)) \in \mathbb{Z}^p \times \mathbb{R}$ lie in the interior of the set $\text{conv}\{(y, t) : y \in \mathbb{Z}^p, t \in \mathbb{R}, t \leq h(y)\}$.

Here, $\text{conv}(S)$ is the convex hull of the set S, which is the smallest convex set containing S. Definition 3.2 is similar to the definition of a convex extensible function in Murota (2003, p. 93) when the function h is finite. We define the subgradient of a discrete concave function as follows.

Definition 3.3. Let $\hat{y} \in \mathbb{Z}^p$ be given and $h : \mathbb{Z}^p \to \mathbb{R}$ be discrete concave. The vector $q(\hat{y}) \in \mathbb{Z}^p$ is called a *subgradient of the discrete concave* function h at $\hat{y}$ if

$$h(y) \leq h(\hat{y}) + q(\hat{y})^T(y - \hat{y}) \; \forall \, y \in \mathbb{Z}^p. \tag{3.2}$$

A discrete concave function has a subgradient at every point by Theorem 2.4.7 in Bazaraa et al. (1993), since the set $\text{conv}\{(y, t) : y \in \mathbb{Z}^p, t \in \mathbb{R}, t \leq h(y)\}$ is closed.

Let $q_i(\hat{y})$ and $\bar{q}_i(\hat{y}; n)$ be subgradients at $\hat{y}$ of $g_i(y)$ and $\bar{g}_i(y; n)$, respectively. There are many potential methods one might consider to obtain the subgradients. Finite

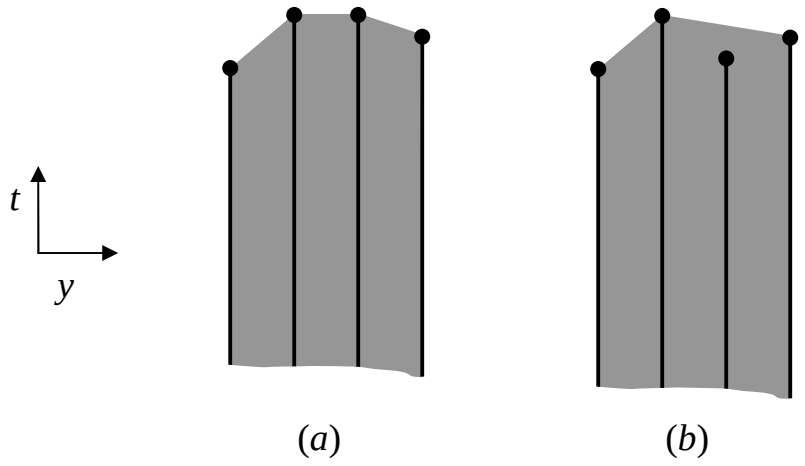


Figure 3.1: Illustration of a discrete concave function. (a) discrete concave. (b) not discrete concave. The dots are points of the form $(y, h(y))$. The lines represent the points in the set $\{(y,t) : y \in \mathbb{Z}^p, t \in \mathbb{R}, t \leq h(y)\}$ and the shaded area is the set $\text{conv}\{(y,t) : y \in \mathbb{Z}^p, t \in \mathbb{R}, t \leq h(y)\}$.

differences using differences of length 1 appear reasonable since we are working with integer variables. There are, however, examples where that fails to produce a subgradient, even for a concave nondecreasing function. Still, we used finite differences in our numerical study and converged to an optimal solution of the SAA problem. Gradients might also be obtained using infinitesimal perturbation analysis (IPA) (see, e.g., Glasserman, 1991). Before using IPA we would have to extend the service level function to a differentiable function defined over a continuous domain, since IPA is applied in settings where the underlying function is differentiable. See Chapter IV for a detailed treatment.

The subgradients are used to approximate the sample average of the service level constraints. Let $\hat{y}$ be a given server allocation vector, and suppose that $\bar{g}_i(\hat{y}; n)$ and $\bar{q}_i(\hat{y}; n)$ are as above. If our assumptions about concavity of $\bar{g}$ hold, then by Definition 3.3 we must have $\bar{g}_i(y; n) \leq \bar{g}_i(\hat{y}; n) + \bar{q}_i(\hat{y}; n)^T(y - \hat{y})$ for all allocation vectors y, and all i. We want y to satisfy $\bar{g}(y; n) \geq 0$ and therefore it is necessary that

$$0 \leq \bar{g}_i(\hat{y}; n) + \bar{q}_i(\hat{y}; n)^T(y - \hat{y}), \tag{3.3}$$

for all i.

3.3 The Simulation-Based Kelley's Cutting Plane Method

In this section we present a cutting plane algorithm for solving the SAA problem (2.4).

To make the presentation of our algorithm clearer and to establish the convergence results we make the following assumption on the tour vector x.

Assumption 3.1. The tour vector x is restricted to a compact set X.

The above assumption can be easily justified in practice. It is, for example, impossible to hire an infinite number of employees, and there are usually budget constraints which impose an upper bound on x since c is positive by Assumption 2.2. Under Assumption 3.1 the original problem (2.3) becomes

$$\begin{array}{lllll} \min & c^T x & & & \\ \text{subject to} & Ax & \geq & y & \\ & g(y) & \geq & 0 & \\ & x & \in & X & \\ & x, y & \geq & 0 & \text{and integer,} \end{array} \tag{3.4}$$

and the sample average approximation of (3.4) is

$$\begin{array}{lllll} \min & c^T x & & & \\ \text{subject to} & Ax & \geq & y & \\ & \bar{g}(y;n) & \geq & 0 & \\ & x & \in & X & \\ & x, y & \geq & 0 & \text{and integer.} \end{array} \tag{3.5}$$

The results of Section 2.3 for the optimal solutions of (2.3) and (2.4) still hold for problems (3.4) and (3.5) if Y^* and Y_n^* are defined as the sets of optimal solutions of (3.4) and (3.5), respectively.

We also define, for future reference,

$$Y := \{y \geq 0 \text{ and integer} : \exists\, 0 \leq x \in X \text{ and integer with } Ax \geq y\}. \tag{3.6}$$

This is the set of staffing levels y that satisfy all the constraints in (3.4) except the expected service level constraints. Note that Y is a finite set since X is compact and the entries in A are either 0 or 1.

Our algorithm fits the framework of Kelley's cutting plane method (Kelley, Jr., 1960). We relax the nonlinear service level constraints of (3.5) to convert the call center staffing problem into a linear integer problem. We then solve the linear integer problem and run a simulation with the staffing levels obtained from the solution. If the service levels meet the service level constraints as approximated by the sample average then we stop with an optimal solution to (3.5). If a service level constraint is violated then we add a linear constraint to the relaxed problem that eliminates the current solution but does not eliminate any feasible solutions to the SAA problem. The procedure is then repeated.

Our algorithm differs from the traditional description of the algorithm only in that we use a simulation to generate the cuts and evaluate the function values instead of having an algebraic form for the function and using analytically determined gradients to generate the cuts. Nevertheless, we include a proof of convergence of our cutting plane method, since its statement is specific to our algorithm and it makes the results clearer.

The relaxed problem for (3.5) that we solve in each iteration is

$$\begin{array}{lllll} \min & c^T x & & \\ \text{subject to} & Ax & \geq & y & \\ & D^k y & \geq & d^k & \\ & x & \in & X & \\ & x, y & \geq & 0 \text{ and integer.} & \end{array} \tag{3.7}$$

The constraints $\bar{g}(y;n) \geq 0$ have been replaced with linear constraints $D^k y \geq d^k$. The superscript k indicates the iteration number in the cutting plane algorithm. The constraint set $D^k y \geq d^k$ is initially empty, but we add more constraints to it as the algorithm progresses.

We select a fixed sample size, n, at the beginning of the algorithm and use the same sample (common random numbers) in each iteration. It is usually not necessary to store the n independent realizations. Instead, we only need to store a few numbers, namely seeds, and reset the random number generators (streams) in the simulation with the stored seeds at the beginning of each iteration. See Law and Kelton (2000) for more details on this approach to using common random numbers. Using common random numbers minimizes the effect of sampling in that we only work with one function $\bar{g}(\cdot;n)$ instead of getting a new $\bar{g}(\cdot;n)$ function in each iteration, which could, for example, invalidate the concavity assumption.

At iteration k we solve an instance of (3.7) to obtain the solution pair (x^k, y^k). For the server allocation vector y^k we run a simulation to calculate $\bar{g}(y^k;n)$. If we find that the service level is unacceptable, i.e., if $\bar{g}_i(y^k;n) < 0$ for some i, then we add the constraint (3.3) to the set $D^k y \geq d^k$, i.e., we add the component $-\bar{g}_i(y^k;n) + \bar{q}_i(y^k;n)^T y^k$ to d^k and the row vector $\bar{q}_i(y^k;n)^T$ to D^k. We add a constraint for all periods i where the service level is unacceptable. Otherwise, if the service level is acceptable in all periods, then we terminate the algorithm with an optimal solution to the SAA problem (3.5). We summarize the simulation-based Kelley's plane method (SKCPM) for the call center staffing problem in Figure 3.2

To speed up the algorithm it is possible to start with D^1 and d^1 nonempty. Ingolfsson et al. (2003) developed, for example, lower bounds on y. They point out that if there is an infinite number of servers in all periods except period i and if $\tilde{y}_i$ is the minimum number of employees required in period i in this setting so that the service level in period i is acceptable, then $y_i \geq \tilde{y}_i$ for all y satisfying $g(y) \geq 0$. We could

Initialization Generate n independent realizations from the distribution of Z. Let $k := 1$, D^1 and d^1 be empty.

Step 1 Solve (3.7) and let (x^k, y^k) be an optimal solution.

Step 1a Stop with an error if (3.7) was infeasible.

Step 2 Run a simulation to obtain $\bar{g}(y^k; n)$.

Step 2a If $\bar{g}(y^k; n) \geq 0$ then stop. Return (x^k, y^k) as a solution for (3.5).

Step 3 Compute, by simulation, $\bar{q}_i(y^k; n)$ for all i for which $\bar{g}_i(y^k; n) < 0$, and add the cuts (3.3) to D^k and d^k.

Step 4 Let $d^{k+1} := d^k$ and $D^{k+1} := D^k$. Let $k := k + 1$. Go to Step 1.

Figure 3.2: The simulation-based Kelley's cutting plane method (SKCPM).

select D^1 and d^1 to reflect such lower bounds.

If the SKCPM terminates in Step 1a then the SAA problem is infeasible. That could be due to either a sampling error, i.e., the SAA problem does not have any feasible points even though the original problem is feasible, or that the original problem is infeasible. As a remedy, either the sample size should be increased, or the original problem should be reformulated, for example, by lowering the acceptable service level, or X should be expanded, e.g., by allocating more employees.

In the SKCPM an integer linear program is solved and constraints are added to it in each iteration until the algorithm terminates. The integer linear problem always has a larger feasible region than the SAA problem (3.5), so $c^T x^k \leq c^T x^{k+1} \leq c^T x_n^*$, where (x_n^*, y_n^*) is an optimal solution for (3.5). An important question is whether $\lim_{k\to\infty} c^T x^k = c^T x_n^*$. The following theorem answers this question in the positive.

Theorem 3.1.

1. *The algorithm terminates in a finite number of iterations.*
2. *Suppose that each component of $\bar{g}(y; n)$ is concave in y. Then the algorithm terminates with an optimal solution to* (3.5) *if and only if* (3.5) *has a feasible solution.*

Proof:

1. The set Y defined in (3.6) is the set of all staffing levels y that satisfy all the constraints in (3.4) except the expected service level constraints, so $y^k \in Y$. The set Y is finite and therefore it is sufficient to show that no point in Y is visited more than once. Suppose that the algorithm did not terminate after visiting point y^t. That means that $\bar{g}(y^t; n) \ngeq 0$ and we added one or more cuts of the form

$$0 \leq \bar{g}_i(y^t; n) + \bar{q}_i(y^t; n)^T(y - y^t), \quad i \in \{1, \ldots, p\}$$

to (3.7). Suppose that $y^k = y^t$ for some $k > t$. Since y^k is the solution for (3.7) at step k it must satisfy the cuts added at iteration t, i.e., $0 \leq \bar{g}_i(y^t; n) + \bar{q}_i(y^t; n)^T(y^k - y^t) = \bar{g}_i(y^t; n)$, which is a contradiction because this constraint was added since $\bar{g}_i(y^t; n) < 0$. Hence, we visit a new point in the set Y in each iteration and thus the algorithm terminates in a finite number of iterations.

2. Suppose first that (3.5) does not have a feasible solution. Then no $y \in Y$ satisfies $\bar{g}(y; n) \geq 0$. The algorithm only visits points in Y, so the optimality condition in Step 2a is never satisfied. Since the algorithm terminates in a finite number of iterations it must terminate with the relaxed problem being infeasible.

   Suppose now that (3.5) is feasible. The problem (3.7) solved in Step 1 is a relaxed version of (3.5) since $\bar{g}(\cdot; n)$ is concave, so (3.7) is feasible in every iteration. Therefore, the algorithm terminates in Step 2a with (x^k, y^k) as the solution. But $\bar{g}(y^k; n) \geq 0$ by the termination criteria, so it is an optimal solution to (3.5). □

3.4 Numerically Checking Concavity

The success of the cutting plane algorithm relies on concavity of each component of the service level function $\bar{g}(\cdot; n)$. If a component of $\bar{g}(\cdot; n)$ is not concave, then the algorithm may "cut off" a portion of the feasible set and terminate with a suboptimal solution. In each iteration of the algorithm we obtain new information about $\bar{g}(\cdot; n)$. To improve the robustness of the algorithm, we would like to ensure that the information we receive is consistent with the notion that each component of $\bar{g}(\cdot; n)$ is concave.

There are two cases to consider. The first is where the vectors $\bar{q}_i(y; n)$ as returned by the simulation are guaranteed to be subgradients of $\bar{g}_i(\cdot; n)$ *if* $\bar{g}_i(\cdot; n)$ is concave. For example, this would occur if the vectors were exact gradients of the function $\bar{g}_i(\cdot; n)$ at y (assuming that it had a differentiable extension to $\mathbb{R}^p$ from $\mathbb{Z}^p$). In this case there is an easy test for nonconcavity, as we will see. The second case, that appears more likely to occur in practice, is where the vectors $\bar{q}_i(y; n)$ are obtained using some heuristic, and are therefore not guaranteed to be subgradients, even if $\bar{g}_i(\cdot; n)$ is indeed concave. In this case, we may decide to disregard some of the potentially-unreliable "subgradient" information and focus only on the function values themselves. (This setting may also be useful if one does not have "subgradient" information at all points, as arises using the finite-differencing heuristic mentioned earlier. When evaluating the "subgradient" at y, we also compute the function value, *but not gradient information*, at points of the form $y + e_i$ where e_i is the usual ith basis vector.) If the function values themselves are not consistent with the notion that the function is concave, then we may view our heuristically-derived "subgradients" with some suspicion, and even drop some of them from the optimization. An alternative would be to attempt to restrict the feasible region to a region where the functions are concave. In Chapter V we modify the method so that it applies to pseudoconcave functions, which is a weaker property than concavity. If the function values alone suggest nonconcavity, then the

algorithm results should be viewed with some caution. Indeed, values reported as optimal by the algorithm could, in this case, be suboptimal. The ability to detect when the key assumption of the cutting plane algorithm may not apply is, we believe, a strength of our approach.

Of course, one may either implement a check for nonconcavity either inline in each iteration of the cutting plane algorithm, or after the algorithm halts, or not at all. The choice depends on how conservative one wishes to be, and is therefore application dependent, and so we do not enter into a discussion of which approach to take here.

To simplify the presentation, let us consider the concavity of a real-valued function $f : \mathbb{R}^p \to \mathbb{R}$ instead of $\bar{g}_i(\cdot; n)$. Hopefully no confusion will arise since the previously-defined function f plays no role in this section. We assume that we are given a set of points y^1, y^2, ..., $y^k \in \mathbb{R}^p$ and their corresponding function values $f(y^1)$, $f(y^2)$, ..., $f(y^k)$. The tests below allow one to conclude that either f is nonconcave, or that there exists a concave function that matches the given function values. Of course, the tests cannot conclude that f is concave unless they examine all points in its domain, so that the conclusions that these tests reach are the best possible in that sense.

3.4.1 Concavity Check with Function Values and "Subgradients"

Suppose that we know the vectors $q(y^1)$, $q(y^2)$, ..., $q(y^k)$ in addition to the set of points and their function values. Here $q(y^v)$ should have the property that *if* f is concave, then $q(y^v)$ is a subgradient at y^v ($v = 1, \ldots, k$). If they are in fact subgradients then they need to satisfy (3.1), i.e., all k points must lie below the k hyperplanes defined by the $q(y^v)$s and the corresponding function values. This means that for each point y^v, $v \in \{1, \ldots, k\}$, we must check that

$$f(y^w) \leq f(y^v) + q(y^v)^T (y^w - y^v) \; \forall \, w \in \{1, \ldots, k\}. \tag{3.8}$$

If this inequality is violated by some v and w, then we conclude that f is not concave in y. Otherwise, the known values of f do not contradict the concavity assumption and $h(y) := \inf_{v\in\{1,\ldots,k\}} f(y)+q(y^v)^T(y-y^v)$ is a concave function (see Theorem 5.5 in Rockafellar, 1970), such that $h(y^w) = f(y^w)\ \forall\, w \in \{1,\ldots,k\}$. In other words if (3.8) holds $\forall\, v \in \{1,\ldots,k\}$ then a concave function exists that agrees with the observed function values $f(y^v)$ and "subgradients" $q(y^v)$, $v = 1,\ldots,k$.

When this test is implemented in the framework of the SKCPM, where in each iteration k we obtain y^{k+1}, $\bar{g}_i(y^{k+1}; n)$ and $\bar{q}_i(y^{k+1}; n)$, we need only check that (for each period i) the new point lies below all the previously defined hyperplanes and that all previous points lie below the hyperplane defined by the new "subgradient."

3.4.2 Concavity Check with Function Values Only

Now consider the case when only f is known at a finite number of points. We want to know whether or not there is a concave function, say h, which passes through f at all the given points. If such a function does not exist then we conclude that f is not concave. This problem appeared in Murty (1988, p. 539). A related problem is to numerically check, given an algebraic expression whether the function is concave in some region. In Chinnick (2001) a software package called MProbe is described, which "works with mathematical models specified in the AMPL language" and is used, among other things, for detecting concavity. The software uses sampling of line segments, i.e., a pair of points are sampled and then a number of points in between those two points are sampled to study whether the function values lie above (concave) or below (convex) the line defined by the pair of points. This is done for a number of such pairs. This method would not work well in the call center setting, since evaluating each function value is computationally expensive. Furthermore, the service level functions are discrete, so typically there are no integer points on the line segment between two points in the domain of the functions.

We present a method where we solve a linear program (LP) and draw our conclusions based on the results of the LP. The idea behind this method is that if a one-dimensional function is concave then it is possible to set a ruler above each point and rotate it until the function lies completely below the ruler. This can also be done when dealing with functions of higher dimensions, and then the ruler takes the form of a plane ($p = 2$) or a hyperplane ($p > 2$).

The LP changes the given function values so that a supporting hyperplane for the convex hull of the points can be fitted through each point. The objective of this LP is to minimize the change in the function values that needs to be made to accomplish this goal. If the changes are measured in the L_1- or L_∞-norm then the objective function is linear. The LP also gives an idea of how far, in some sense, the function is from being concave if a concave function cannot be fitted through the given points. If a concave function can be fitted then the LP will return such a function, namely the pointwise minimum of the hyperplanes computed by the LP.

It is most straightforward to use the L_1-norm to measure the changes in the function values. Then the LP can be formulated as follows:

$$\begin{array}{lll} \text{min} & \sum_{v=1}^{k} |b^v| & \\ \text{subject to} & a_0^v + (a^v)^T y^v = f(y^v) + b^v & \forall\, v \in \{1,\dots,k\} \\ & a_0^v + (a^v)^T y^w \geq f(y^w) + b^w & \forall\, v \in \{1,\dots,k\}, \\ & & \forall\, w \in \{1,\dots,k\},\ w \neq v \end{array} \tag{3.9}$$

To linearize the objective function we adopt the standard trick of writing $b^v = (b^v)^+ - (b^v)^-$ and replace $|b^v|$ with $(b^v)^+ + (b^v)^-$, where $(b^v)^+$ and $(b^v)^-$ are nonnegative. The decision variables are

$$\begin{array}{llll} a_0^v & \in \mathbb{R} \quad v \in \{1,\dots,k\} & : & \text{intercepts of the hyperplanes,} \\ a^v & \in \mathbb{R}^p \quad v \in \{1,\dots,k\} & : & \text{slopes of the hyperplanes and} \\ (b^v)^+, (b^v)^- & \in \mathbb{R} \quad v \in \{1,\dots,k\} & : & \text{change in the function values.} \end{array}$$

The number of variables in this LP is $k(p+1)+2k = k(p+3)$ and the number of constraints is $k + k(k-1) = k^2$. We could split the LP up into k separate linear programs if that would speed up the computations, as might occur if we could run them on multiple processors in parallel, or if the LP solver was unable to detect the separable structure in this problem and exploit it. Here, the vth separate linear program tries to fit a hyperplane through the point $(y^v, f(y^v))$ that lies above all other points.

The LP is always feasible, since a feasible solution is given by $a^v = 0$, $a_0^v = 0$ and $b^v = -f(y^v)$ for all $v \in \{1, \ldots, k\}$. It is also bounded below by 0, since the objective function is a sum of absolute values. Therefore, this problem has a finite minimum. If the minimum value is 0, then the function defined by $h(y) := \inf_{v=1,\ldots,k} a_0^v + (a^v)^T y$ is concave and $f(y^v) = h(y^v)$ for all $v \in \{1, \ldots, k\}$. On the other hand, if f is indeed concave, then there exists a subgradient at every point of f (see Theorem 3.2.5 in Bazaraa et al., 1993) and hence the constraints of the LP can be satisfied with $b^v = 0$ for all $v \in \{1, \ldots, k\}$. We have proved the following result.

Theorem 3.2. *Consider the LP (3.9).*

1. *If the optimal objective value of the LP is* 0 *then there exists a concave function* $h(y)$ *such that* $h(y^v) = f(y^v)$ *for all* $v \in \{1, \ldots, k\}$.
2. *If* f *is concave then the optimal objective value of the LP is* 0.

So we see that a necessary condition for f to be concave is that the optimal objective value of the LP (3.9) is zero. Thus we have the following corollary.

Corollary 3.3. *If the optimal objective value of the LP (3.9) is positive, then* f *is not concave.*

Note that the hyperplanes obtained from the LP are generally not subgradients of f, so we cannot use them in the SKCPM as such. Hence, we have to solve this LP after Step 2 in each iteration, or as a check after the algorithm terminates. Given the computational demands of the cutting plane algorithm, repeatedly solving this LP in each iteration does not represent a significant increase in computational effort.

3.5 Computational Study

In this section we present a small numerical example that showcases the properties of our method. The example is far from being a realistic representation of a call center, but captures many issues in setting call center staffing levels. We will study 3 aspects of the problem in the context of the example:

1. Convergence of the cutting plane algorithm and the quality of the resulting solution.

2. Dependence of the service level in one period on staffing levels in other periods. This is of particular practical interest since traditional methods assume independence between periods.

3. Concavity of $\bar{g}(y; n)$.

Our implementation creates the integer programs (3.7) in AMPL and uses the CPLEX solver to solve them in Step 1 of the algorithm, and a simulation model built in ProModel to perform Steps 2 and 3. We used Microsoft Excel to pass data between the simulation and optimization components and to run the iterations of the algorithm. The implementation was exactly as described in the SKCPM except for the initialization, where we started with y^1 at the lower bounds described in Section 3.3 instead of starting with D^1 and d^1 empty.

3.5.1 Example

We consider an $M(t)/M/s(t)$ queue with $p = 5$ periods of equal length of 30 minutes. We let the service rate be $\mu = 4$ customers/hour. The arrival process is a nonhomogeneous Poisson process with the arrival rate a function of the time t in minutes equal to $\lambda(t) = \lambda(1 - |t/150 - .65|)$, i.e., the arrival rate is relatively low at the beginning of the first period, then increases linearly at rate λ until it peaks partway through

the fourth period and decreases at rate λ after that. We set $\lambda = 120$ customers/hour, which makes the average arrival rate over the 5 periods equal 87.3 customers/hour.

The goal is to answer 80% of received calls in each period in less than 90 seconds. The customers form a single queue and are served on a first come first served basis. If a server is still in service at the end of a period it finishes that service before becoming unavailable. For example, if there are 8 busy servers at the end of Period 3 and Period 4 only has 6 servers then the 8 servers will continue to serve the customers already in service, but the next customer in the queue will not enter service until 3 customers have finished service.

There are 6 permissible tours, including 4 tours that cover 2 adjacent periods, i.e., Periods 1 and 2, 2 and 3, 3 and 4, and finally 4 and 5. The remaining 2 tours cover only one period, namely the first and the last. The cost of the tours covering 2 periods is \$2 and the single period tours cost \$1.50.

3.5.2 Results

We selected a sample size of $n = 100$ for running the algorithm. Table 3.1 shows the iterates of the algorithm and Table 3.2 shows the corresponding service levels. The lower bounds on y are depicted in the row $k = 1$ in Table 3.1. Note that the staffing levels at the lower bounds result in an unacceptable level of service and thus a method which would treat the periods independently, would give an infeasible solution, since the service level is as low as 73.8% in period 4. The algorithm terminates after only 3 iterations with an optimal solution to the SAA problem. To verify that this is indeed an optimal solution we ran a simulation for all staffing levels that have lower costs than the optimal solution and satisfy the initial lower bounds. None of these staffing levels satisfied $\bar{g}_{100}(y) \geq 0$, so the solution returned by the algorithm is the optimal solution for the SAA problem. By including the 95% confidence interval we get information about the quality of the solution as a solution of the original problem. In the example,

the confidence intervals in periods 1, 3 and 5 cover zero, which is a concern since we cannot say with conviction that our service level is acceptable in those periods. To get a better idea of whether the current solution is feasible for the original problem we calculated $\bar{g}_{999}(y^3) = (0.5 \pm 0.3, 3.0 \pm 0.5, 2.3 \pm 0.7, 5.1 \pm 0.7, 0.0 \pm 0.8)^T$, so we are more confident that the service levels in Periods 1 and 3 are acceptable. The service level in Period 5 is close to being on the boundary, hence our difficulty in determining whether the solution is feasible or not. From a practical standpoint, if we are infeasible, then we are so close to being feasible that it probably is of little consequence.

We already noted that there is dependence between periods. To investigate the dependence further we calculated $\bar{r}^3(y; n)$, the percentage of calls received in Period 3 answered in less than 90 seconds, i.e., $\bar{r}^3(y; n) := \sum_{d=1}^{n} S^3(y, z^d) / \sum_{d=1}^{n} N^3(y, z^d)$. We chose Period 3 to demonstrate how the service level depends on staffing level in both the period before and after. Figure 3.3 illustrates this point. The graphs show the service level in Period 3 as a function of the staffing level in Period 3 (3.3.a), Period 2 (3.3.b) and Period 4 (3.3.c) when the staffing levels in other periods are fixed. The service level depends more on the staffing level in the period before than the period after as could be expected. That is because a low staffing level in an earlier period results in a queue buildup, which increases waiting in the next period. The reason why the staffing level in a later period affects the service level in an earlier period is that customers that called in the earlier period may still be waiting at the beginning of the next period and thus receive service earlier if there are more servers in that period. We noted dependence between periods as far apart as from the first period to the last. Figure 3.3 also supports the concavity assumption of the service level function when y is within a region of reasonable values, i.e., at least satisfies some lower bounds. It is, however, clear that the service level function looks like an S-shaped function over a broader range of y's, as pointed out by Ingolfsson et al.

(2003).

We also performed a separate concavity check based on the method in Section 3.4.2. In an effort to demonstrate these ideas as clearly as possible we performed the concavity check *outside* the scope of the cutting plane algorithm itself, using a selection of points that appear reasonable from a performance standpoint. We used a sample size of 300 and calculated $\bar{g}(y; 300)$ at 3 different staffing levels (labeled low, medium and high in Table 3.3) for each period, i.e., at $3^5 = 243$ points. We solved the linear program (3.9) for each $\bar{g}_i(y; 300)$, $i \in \{1, \ldots, 5\}$ and obtained the results in Table 3.3. We see that the service level functions in periods 1 and 2 do not violate the concavity assumption at the observed points. The other functions violate the concavity condition. The values of the b^vs, i.e., the changes needed to satisfy the concavity assumption are all small, as can be seen by the objective value. We examined the points at which nonconcavity was detected, and noted that they occurred when a change in staffing level in a different period was made. (It is a strength of the LP-based concavity check that we were able to discover a region where the nonconcavity was exhibited.) The service level in Period 3 increased, for example, more when the staffing level in Period 1 was increased form 12 to 14 than when it was increased from 10 to 12 at staffing levels 22 and 30 in Periods 2 and 3, respectively. The reason for this violation of the concavity assumption is not obvious.

One possible explanation is that our measure of service quality is binary for each customer, so that "rounding" may contribute to the nonconcavity. To elaborate, in the above example it is possible that unusually many customers exceed the waiting time limit of 90 seconds by very little when there are 12 servers in period 1, so that the effect of adding servers at this staffing level is more than when servers are added at a lower level. We would expect such a "rounding" effect to be averaged out in a longer simulation. In fact, we increased the sample size to 999 (the maximum number of replications in ProModel 4.2) and calculated the service level at the problematic

points. We discovered that the nonconcavity vanished. Therefore, we make the following conjecture:

Conjecture 3.4. *For $M(t)/M/s(t)$ queues of the form considered here there exists a finite $y_0 \geq 0$ such that the service level function g is nondecreasing and concave in y in the region $y \geq y_0$. Furthermore, $\bar{g}(\cdot; n)$ is nondecreasing and concave in y in the region $y_0 \leq y \leq y_1$ for all n large enough w.p.1., for any fixed $y_1 \geq y_0$.*

3.5.3 More Periods

In the example we considered a problem with only 5 periods. As part of this research See and Seidel (2003) performed a computational study of a similar call center. They studied a problem with 24 time periods and a problem with 72 time periods. They used both the finite difference method to compute subgradients and the infinitesimal perturbation analysis (IPA) gradient estimator (4.46) that will be described in detail in Chapter IV.

In summary, they did get a good solution of the 24 period problem using both the finite difference method and the IPA estimator to compute the subgradients. The solutions were different but had the same objective value. The method converged in only 2 iterations when the IPA estimator was used and in 5 iterations when the finite difference estimator was used to compute subgradients. On the other hand, when they applied the method on the 72 period problem the solution they got was clearly far from being an optimal solution of the sample average problem. This was because the initial lower bounds on the staffing levels were not sufficient to start the algorithm in a concave region of the sample average of the service level functions. When the cuts were added in the beginning, a significant portion of the feasible set (and the optimal set) of staffing levels was therefore cut off.

There are at least two possible reasons why this can occur. One is that the precomputed lower bounds are not tight enough to start the algorithm in a concave region of the service level functions. Another is that the optimal solution does not lie in a concave region of the service level functions (if such a region actually exisits). As

a remedy, one can either compute lower bounds that are better than the ones obtained by the "infinite number of servers in all other periods" model, or compute cuts that do not require the functions to be concave. In this dissertation we do not look at alternative methods for computing better bounds on the staffing levels, although some attempts are described in See and Seidel (2003). In Chapter V, however, we consider an analytic cutting plane method for pseudoconcave functions, relaxing the concavity assumption somewhat. There we include a numerical experiment where we solve the 72 period problem *without any precomputed lower bounds* other than the natural nonnegativity bounds.

k	y^k					$f(y^k)$
1	11	19	27	30	29	125.0
2	11	21	27	33	29	127.5
3	11	21	27	34	29	128.0

Table 3.1: The iterates of the SKCPM. $f(y^k)$ is the objective value at y^k.

k	$\bar{g}_{100}(y^k) \pm 95\%$ CI half width (% of calls received that are answered in less than 90 sec.)				
1	0.4 ± 1.0 (81.5%)	-1.1 ± 2.2 (77.2%)	-1.8 ± 3.1 (76.6%)	-3.5 ± 3.2 (73.8%)	-2.1 ± 2.3 (75.5%)
2	0.5 ± 0.9 (81.9%)	3.4 ± 1.5 (88.5%)	0.2 ± 2.6 (80.5%)	4.1 ± 2.2 (87.4%)	-0.3 ± 2.7 (79.4%)
3	0.5 ± 0.9 (81.9%)	3.4 ± 1.5 (88.5%)	0.4 ± 2.6 (80.7%)	5.8 ± 1.8 (90.4%)	0.0 ± 2.6 (80.0%)

Table 3.2: The resulting service level function values of the iterates displayed in Table 3.1 and their 95% confidence intervals (CI).

Period	1	2	3	4	5
Low	10	18	26	29	29
Medium	12	20	28	31	31
High	14	22	30	33	33
Optimal value	0.0	0.0	$4.0 \cdot 10^{-3}$	$1.5 \cdot 10^{-2}$	$5.7 \cdot 10^{-4}$

Table 3.3: Concavity study. Low, medium and high staffing levels in each period and the optimal values of (3.9).

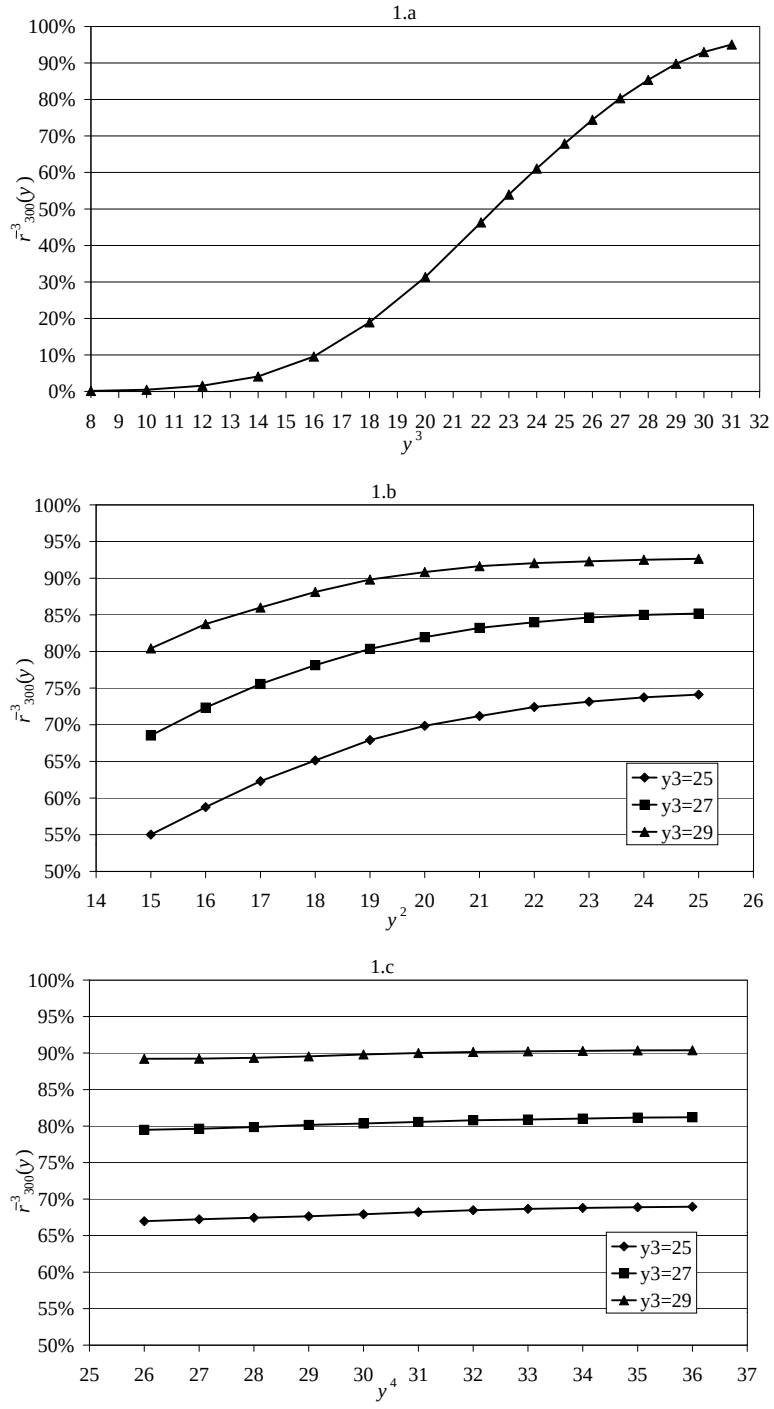


Figure 3.3: Dependence of staffing levels on the service level in Period 3 of the example in Section 3.5.1.

CHAPTER IV

USING SIMULATION TO APPROXIMATE SUBGRADIENTS OF CONVEX PERFORMANCE MEASURES IN SERVICE SYSTEMS

4.1 Introduction

In Chapter III we described a simulation based cutting plane algorithm for solving a call center staffing problem. The algorithm is guaranteed to converge to an optimal solution if one exists and if the service level constraints are concave. In this chapter we explore what is perhaps the most challenging part of the algorithm: computation of the (sub)gradients of the sample average approximation of the service level function. What makes this particularly challenging is that the service level function is a discrete function of the number of agents, so a gradient, which could otherwise have been used as a subgradient, does not exist. In addition, we do not have a closed form expression of the function.

In the call center staffing problem in Chapter II we defined the service level function in a particular time period i, $i \in \{1, \ldots, p\}$, as the fraction of calls received in that period answered within a certain amount of time τ. In fact, we re-formulated the service level function such that the true interpretation is the expected number of calls answered within time τ in excess of a fraction of the expected number of calls in the period or

$$g_i(y) = E[S_i(y, Z)] - l_i E[N_i(Z)],$$

where we defined $y \in \mathbb{Z}_+^p$ as the vector of staffing levels, Z represents all the random-

ness in the problem, N_i is the number of calls received in period i, S_i is the number of those calls answered within time τ and, finally, l_i is the fraction of calls required to be answered on time.

We approximated the service level function g_i with a sample average

$$\bar{g}(y;n) = \frac{1}{n}\sum_{d=1}^{n} S_i(y, z^d) - l_i \frac{1}{n}\sum_{d=1}^{n} N_i(z^d),$$

where $z^1, \ldots, z^n$ are independent realizations of Z. We assume that $\bar{g}(y;n)$ is concave and non-decreasing in y (at least in some range of staffing levels y). In the SKCPM in Chapter III we required a subgradient (cf. Definition 3.3) of $\bar{g}_i(y;n)$ for one or more $i \in \{1, \ldots, p\}$ at some point $\hat{y}$, in addition to the value of $\bar{g}_i(\hat{y};n)$ for all i.

Note that a subgradient of $\bar{g}(y;n)$ is also a subgradient of

$$\bar{s}(y;n) \equiv \frac{1}{n}\sum_{d=1}^{n} S(y, z^d)$$

since $l_i n^{-1} \sum_{d=1}^{n} N_i(z^d)$ is independent of y, i.e., if $\bar{q}_i(\hat{y};n)$ is a subgradient of $\bar{g}_i(y;n)$ at $\hat{y}$ then

$$\begin{gathered}
\bar{g}_i(y;n) \le \bar{g}_i(\hat{y};n) + \bar{q}_i(\hat{y};n)^T(y-\hat{y}) \\
\Updownarrow \\
\bar{g}_i(y;n) + l_i\frac{1}{n}\sum_{d=1}^{n} N_i(z^d) \le \bar{g}_i(\hat{y};n) + \bar{q}_i(\hat{y};n)^T(y-\hat{y}) + l_i\frac{1}{n}\sum_{d=1}^{n} N_i(z^d) \\
\Updownarrow \\
\bar{s}_i(y;n) \le \bar{s}_i(\hat{y};n) + \bar{q}_i(\hat{y};n)^T(y-\hat{y})
\end{gathered}$$

Similarly, it is clear that the concavity of $\bar{s}_i(y;n)$ follows from the concavity of $\bar{g}_i(y;n)$. *The problem is then to compute the vector $\bar{q}_i(\hat{y})$ for a given $\hat{y}$.*

For the numerical experiment in Section 3.5 we used a finite difference method to compute the subgradients. An obvious disadvantage of the FD method is that we need to simulate at $p+1$ different staffing levels, where p is the number of periods

and could be large.

We consider two other methods for computing the subgradients, infinitesimal perturbation analysis (IPA) and the likelihood ratio method (LR). These two are, along with the finite difference method, the most prominent methods for estimating gradients of a function evaluated by simulation. IPA and LR actually estimate gradients of a continuous function so we discuss how to approximate the discrete service level function with such a function. We study each of the FD method, LR method and IPA in an attempt to obtain a subgradient of the SAA of the service level function. We explore the advantages and disadvantages of each method for this particular problem.

The remainder of the chapter is organized as follows. The FD method is discussed in Section 4.2, in Section 4.3 we talk about how to approximate the service level function using a function of continuous variables which is necessary to apply the LR method, which we discuss in Section 4.4, and IPA, which we discuss in Section 4.5. We include a computational study to further enhance the comparison in Section 4.6.

4.2 Finite Differences

The simplest and perhaps the most intuitive method for estimating a gradient (or a subgradient) when an expression for the function is unknown is by the method of finite differences (see, e.g., Andradóttir, 1998). The FD method can easily be extended to discrete functions. There is a price to pay, however, for this ease of implementation. The number of function evaluations (each one requiring a new simulation) to get one gradient estimate is rather large and this method can fail to produce a subgradient even under rather stringent conditions on the service level function.

To estimate the partial derivative with respect to a continuous variable the function is evaluated at two different points where all other variables are held constant. Then an estimate of the derivative at a value at or between these two values can be estimated by linear interpolation. When the variable is integer, as in the staffing

problem, the smallest difference between the two points is one. We use the finite forward difference estimator as an estimate for the subgradient, i.e., we let

$$(\bar{q}_i)_j(\hat{y}; n) = \bar{s}_i(\hat{y} + e_j; n) - \bar{s}_i(\hat{y}; n) \qquad \forall\, j \in \{1, \ldots, p\}, \tag{4.1}$$

where e_j is the jth unit vector in $\mathbb{R}^p$. As we can see this estimator is easy to implement. To estimate the subgradient at the staffing level $\hat{y}$, given $\bar{s}(\hat{y}; n)$, simply run p simulations with the number of agents in period j increased by one in the jth simulation. This of course requires $p+1$ simulations to get a subgradient estimate at a single point. We note, however, that estimates of the pseudogradients of the service level functions in *all* p periods can be obtained from those same $p+1$ simulations.

What can be said about the properties of (4.1) as a subgradient of $\bar{s}_i(y; n)$? We know that the FD method can be used to compute a subgradient of a differentiable strictly concave function as summarized in the following proposition. The problem becomes more complicated, however, when the function is discrete as we will see.

Proposition 4.1. *Let $f : \mathbb{R}^p \to \mathbb{R}$ be a continuous, strictly concave function that is differentiable at the point $\hat{y}$. Let*

$$q_j(\hat{y}) = \frac{f(\hat{y} + \epsilon e_j) - f(\hat{y})}{\epsilon} \qquad \forall\, j \in \{1, \ldots, p\}.$$

Then, for any given $0 < \delta_1 \leq \delta_2 < \infty$ there exists an $\hat{\epsilon} > 0$ such that

$$f(x) \leq f(\hat{y}) + q(\hat{y})^T (x - \hat{y})$$

for all $x \in \{y \in \mathbb{R}^p :\ \delta_1 \leq |y - \hat{y}\| \leq \delta_2\}$ and for all $0 < \epsilon < \hat{\epsilon}$.

Proof: We need to show that for any $x \in \{y \in \mathbb{R}^p : \delta_1 \leq \|y - \hat{y}\| \leq \delta_2, \}$

$$f(x) \leq f(\hat{y}) + q(\hat{y})^T (x - \hat{y}).$$

By strict concavity and the differentiability of f at $\hat{y}$ we know that $f(y) < f(\hat{y}) + \nabla f(\hat{y})^T (y - \hat{y})\ \forall\, y \in \mathbb{R}^p \setminus \hat{y}$ (Bazaraa et al., 1993, Theorem 3.3.3).

Let $m_0 = \min_{y:\delta_1 \leq \|y - \hat{y}\| \leq \delta_2} (f(\hat{y}) + \nabla f(\hat{y})^T (y - \hat{y})) - f(y)$. By Weierstrass' Theo-

rem (Bazaraa et al., 1993, Theorem 2.3.1) the minimization problem has an optimal solution $\tilde{y} \in \{y : \delta_1 \leq \|y - \hat{y}\| \leq \delta_2\}$. Now, $f(\tilde{y}) < f(\hat{y}) + \nabla f(\hat{y})^T(\tilde{y} - \hat{y})$ since f is strictly concave and it follows that $m_0 > 0$.

First,

$$f(\hat{y}) + q(\hat{y})^T(x - \hat{y}) - f(x) \tag{4.2}$$

$$= f(\hat{y}) - f(x) + q(\hat{y})^T(x - \hat{y}) - \nabla f(\hat{y})^T(x - \hat{y}) + \nabla f(\hat{y})^T(x - \hat{y}) \tag{4.3}$$

$$\geq m_0 + (q(\hat{y}) - \nabla f(\hat{y}))^T(x - \hat{y}). \tag{4.4}$$

Next,

$$|(q(\hat{y}) - \nabla f(\hat{y}))^T(x - \hat{y})| \leq \|\nabla f(\hat{y}) - q(\hat{y})\| \cdot \|x - \hat{y}\| \leq \|\nabla f(\hat{y}) - q(\hat{y})\|\delta_2,$$

where the first inequality is the Cauchy-Schwarz inequality (Venit and Bishop, 1989, p. 215) and the second equality follows since $x \in \{y : \delta_1 \leq \|y - \hat{y}\| \leq \delta_2\}$. The ith component of the subgradient is $q_i(\hat{y}) = (f(\hat{y} + \epsilon e_i) - f(\hat{y}))/\epsilon$ and f is differentiable at f so $\lim_{\epsilon \to 0}(f(\hat{y} + \epsilon e_i) - f(\hat{y}))/\epsilon = \partial f(\hat{y})/\partial \hat{y}_i$. Thus there exists an $\hat{\epsilon} > 0$ such that $\|\nabla f(\hat{y}) - q(\hat{y})\| \leq m_0/\delta_2$ for all $\epsilon < \hat{\epsilon}$. Then,

$$m_0 + (\nabla f(\hat{y}) - q(\hat{y}))^T(x - \hat{y}) \geq m_0 - |(\nabla f(\hat{y}) - q(\hat{y}))^T(x - \hat{y})| \geq 0$$

and the result follows. □

One might think that this would also work at a point where the function is not differentiable. That is not true in general. We demonstrate this with a simple example. Consider the function

$$f(y_1, y_2) = \begin{cases} y_1, & 0 \leq y_1 \leq y_2, \\ y_2, & 0 \leq y_2 \leq y_1. \end{cases}$$

The function f is concave and is plotted in Figure 4.1. If we try to estimate a subgradient at any point $\hat{y}$ on the diagonal by the FD method we get $q(\hat{y}) = (0, 0)^T$

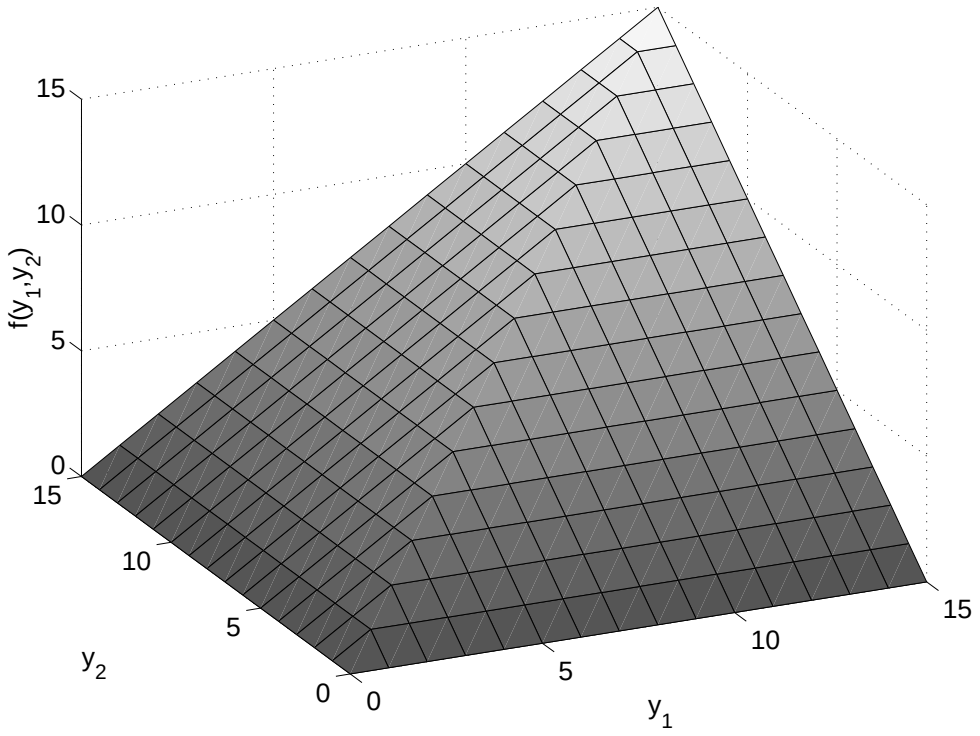


Figure 4.1: An example of a concave nondecreasing function $f(y_1, y_2)$ where the finite difference method can fail to produce a subgradient.

as our estimate. Let $h(y) = f(\hat{y}) + q(\hat{y})^T(y - \hat{y}) = f(\hat{y})$. It is clear that q is not a valid subgradient since $h(\hat{y}_1 + a, \hat{y}_1 + b) = f(\hat{y}) < f(\hat{y}_1 + a, \hat{y}_1 + b)$ for any positive numbers a and b.

What does this mean in the context of the staffing problem? If we look for the reason why the FD method failed for the function f above we see that the function f increases slower if we only change one variable at a time than when we change both variables. This would happen in the call center if there would be greater marginal benefit of adding one agent to each of two periods than the combined marginal benefit of adding one agent to the two periods separately. Then, the marginal return on the service level of adding an agent in any period decreases every time we add an agent *regardless of* what period the previous agent was added to, i.e., assume f is submodular (Topkis, 1998):

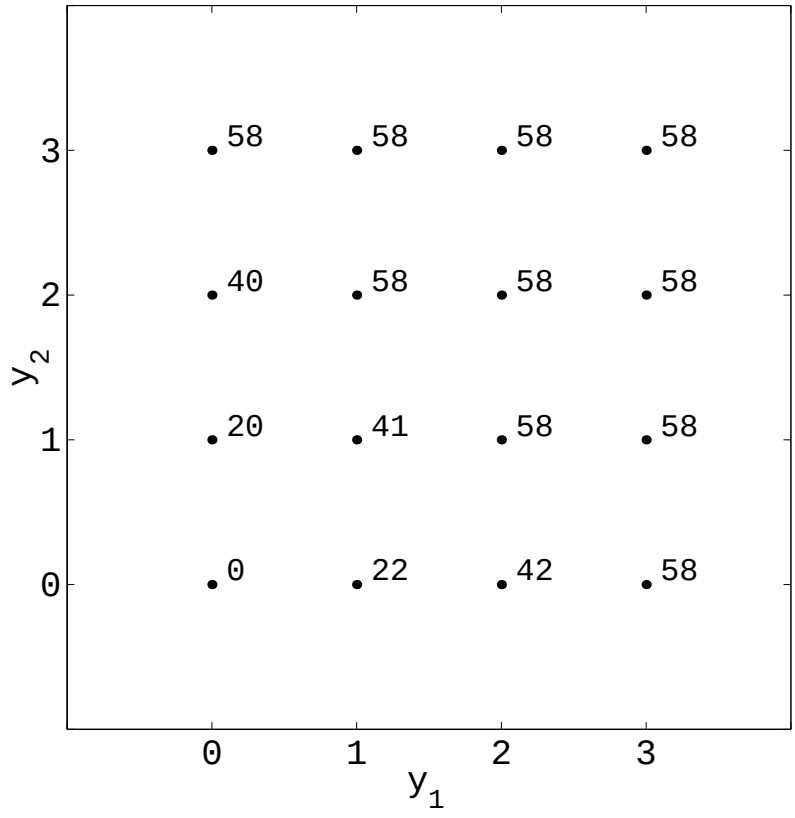


Figure 4.2: A submodular function. The value next to a point represents the function value at the point.

Definition 4.1. A function $f : \mathbb{Z}^p \to \mathbb{R}$ is submodular if and only if $\forall\, y^1, y^2 \in \mathbb{Z}$,

$$f(y^1 \vee y^2) + f(y^1 \wedge y^2) \leq f(y^1) + f(y^2),$$

where $\vee$ and $\wedge$ denote componentwise minimum and componentwise maximum, respectively.

Do we get a subgradient by the FD method if the service level function is submodular? Again, the short answer to that question is no. We also demonstrate this via a simple example. Consider the points and their corresponding function values (numbers by the dots) depicted in Figure 4.2. That function has a subgradient at every point and is submodular. Nevertheless, the FD estimator fails to produce a subgradient at the point (0,1). In the three-dimensional representation of the function, the point (1,0,22) lies above the proposed hyperplane.

We have shown by two examples that the FD method does not necessarily produce

a subgradient of the service level function. Still, there are many other examples where the FD method does produce a subgradient. Consider, for instance, any point other than (0,1) in Figure 4.2. For these points the FD method will indeed produce a subgradient. So it seems reasonable to conclude that, partly due to its ease of implementation, the FD method for obtaining a subgradient is a plausible approach when the number of periods p, i.e., input variables, is not too large.

The examples given above show that the FD method does not always yield a subgradient. What if we compute the function values at more points and use that information to compute a subgradient? For instance, for the function $f(y_1, y_2)$ above we can compute a subgradient at $\hat{y}$ if we know the function values at the tree points $(\hat{y}_1, \hat{y}_2), (\hat{y}_1 + 1, \hat{y}_2), (\hat{y}_1, \hat{y}_2 + 1), (\hat{y}_1 + 1, \hat{y}_2 + 1)$. Is there then a finite number of points (that can depend on the dimension p), such that if we know the function values at those points then we can compute a subgradient at $\hat{y}$ (if a subgradient exists at $\hat{y}$)?

The function $f(y_1, y_2)$ in Figure 4.3 is an example that shows that knowing the function values at all the points within distance 1 of $\hat{y}$ measured in the L_∞-norm (all points in a square centered at $\hat{y}$ with edge length 2) is not sufficient. There exists a continuous, concave and nondecreasing function that agrees with the function values at all the points. Such a function can be constructed by solving the linear program in Section 3.4.2 and restricting the slopes to be nonnegative. Suppose we want to compute a subgradient at $\hat{y} = (2, 2)$. The plane defined by the points $\{(2, 2), (2, 3), (3, 3)\}$ is given by $h(y_1, y_2) = 3 + 1.1(y_1 - 2) + 0.4(y_2 - 2)$. It is easy to verify that $h(y) \geq f(y)$ for all y in $\{y \in \mathbb{Z}^2 : 1 \leq y_i \leq 3 \; \forall \; i \in \{1, 2\}\}$, so it appears that $(1.1, 0.4)^T$ is a subgradient at $\hat{y}$ based on checking all points in the hypercube. At the point $(4, 3)$, however, we get $h(4, 3) = 3 + (1.1)(2) + (0.4)(1) = 5.6 < 5.55 = f(4, 3)$.

Therefore, adding all the points in the hypercube does not help. (The number of points in a hypercube of side length k in dimension p is given by $(k + 1)^p$). Our conclusion is that the number of points required to guarantee that the "subgradient" is

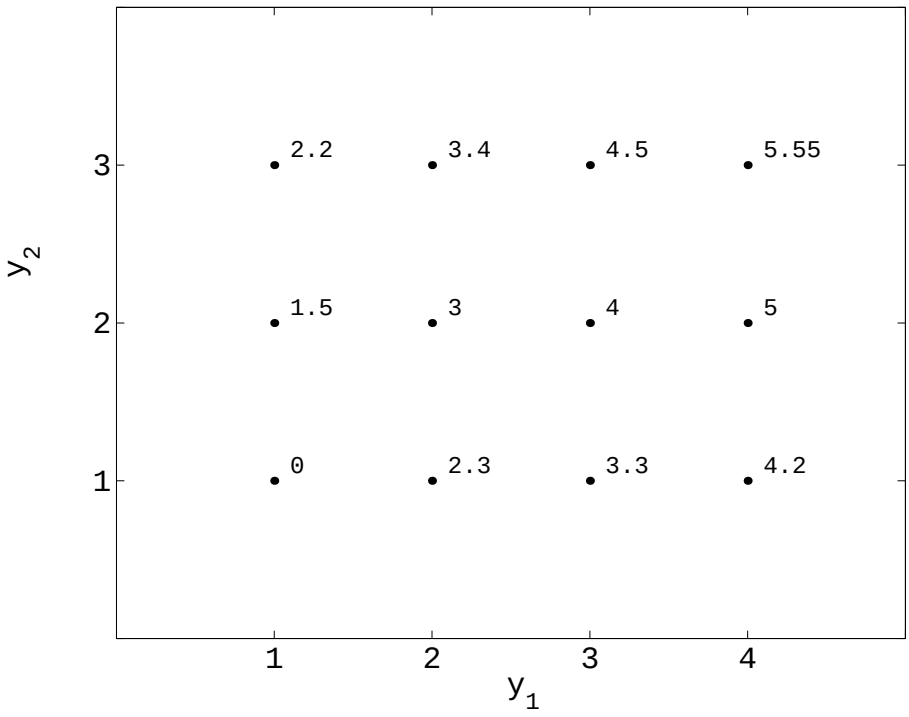


Figure 4.3: An example of a function to show that a subgradient cannot be computed using only function values in a neighborhood of a point that includes 3^p points. The values next to the points denote the function values $f(y_1, y_2)$.

indeed a subgradient is so large (if at all finite) that it would in most cases be impractical for implementation within the cutting plane algorithm. We therefore content ourselves with viewing FD "subgradients" as heuristically derived approximations for true subgradients.

In the context of the SKCPM, the implication of having an invalid subgradient is adding an invalid cut which can then cut off an optimal solution. Thus, we might terminate the algorithm with a suboptimal solution (or none at all). That is obviously a concern when the goal of the cutting plane algorithm is to find the best staffing levels. In many cases, however, the underlying problem might be so complicated that obtaining a "good" solution is a goal that can be reached even though the "subgradients" are invalid.

4.3 Using Continuous Variables to Approximate the Discrete Service Level Function

In real applications of the call center staffing problem, the number of periods p can be quite large. It would, therefore, greatly reduce the computational effort of computing a subgradient to use simulation gradient estimation techniques such as IPA or the LR method that can in some situations obtain unbiased gradient estimates using the results from a simulation at a *single* point.

For IPA and the LR method to apply, however, we must first approximate $\bar{s}(y; n)$ by a function of a continuous variable rather than the discrete variable y. For this problem, and queuing problems in general, a natural candidate is a function of the service rates in each period. When we use the service rates as the variables instead of the staffing levels we get a new function. We would like the function (or the gradients) to have similar characteristics as the original function.

4.3.1 A Simple Example: Changing the Number of Servers and Service Rates in an $M/M/s$ Queue

We begin with a simple example. An $M/M/s$ queuing system is a system to which customers arrive at a queue for service at one of s servers. The number of servers in this example should not be confused with the expected service level function, which is also denoted by s elsewhere in this chapter. The time between arrivals is modeled as an exponential random variable with rate λ, and the time it takes to serve one customer is also modeled as an exponential random variable with rate μ. There is a single period in this system that we assume never ends. A special case of the service level is the fraction of customers who are delayed, i.e., the case $\tau = 0$. For the $M/M/s$ queuing system, the long run (or steady state) fraction of customers who wait can be

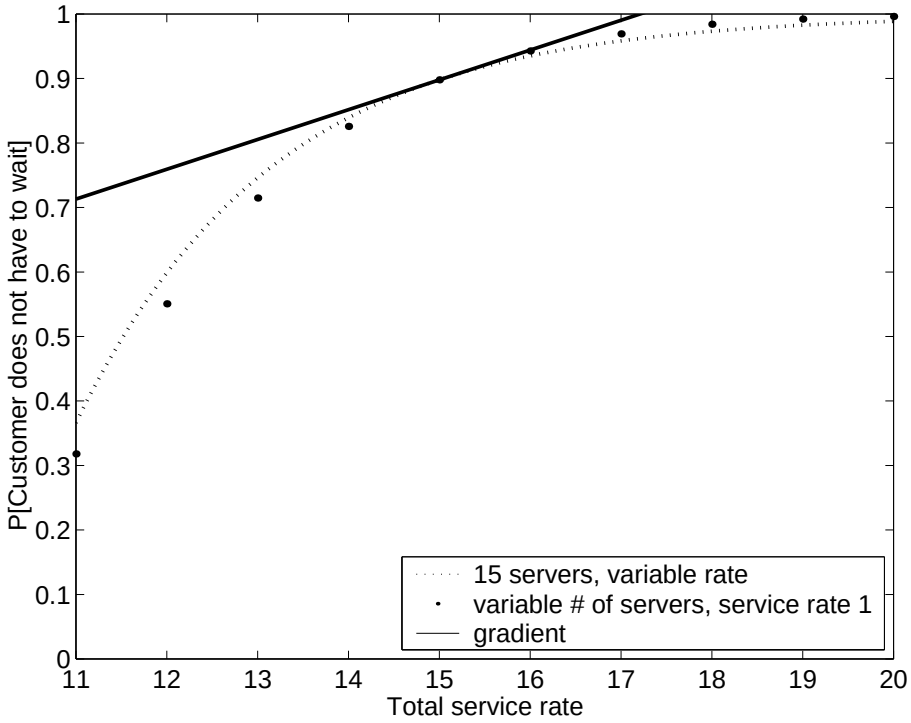


Figure 4.4: Performance as a function of service rate and as a function of the number of servers in an $M/M/s$ queue.

calculated via the Erlang C formula (Gross and Harris, 1998, p. 71)

$$P(\text{Customer is delayed}) = \frac{\sum_{n=0}^{s-1} \frac{R^n}{n!}}{\sum_{n=0}^{s-1} \frac{R^n}{n!} + \frac{R^s}{s!(1-\rho)}}, \tag{4.5}$$

where, $R = \lambda/\mu$ is the load. The equation only applies for server utilization $\rho = \lambda/s\mu < 1$. Otherwise, the probability that the customer is delayed is 1. Figure 4.4 shows the probability of delay as a function of the total service rate $s\mu$. The dotted line is obtained by fixing s at 15 servers and computing $1-P(\text{Customer is delayed})$ for $\mu \in [11/15, 20/15]$. The discrete points are obtained by fixing $\mu = 1$ and computing $1 - P(\text{Customer is delayed})$ for $s \in \{11, 12, \ldots, 20\}$. The arrival rate in this case is $\lambda = 10$. The solid line intersects the graph at total service rate 15, and its slope is given by the derivative of the Erlang C formula w.r.t. μ at total service rate 15 divided by 15. When μ changes by 1 the total service rate changes by 15 and this explains the need for the scaling by 1/15.

We see from the figure that the discrete function and the continuous function coincide at total service rate 15. That is natural since $\mu = 1$ and $s = 15$ in both cases. Furthermore, the tangent to the graph at 15 lies above the graph of the discrete function, and therefore, the derivative of the Erlang C formula w.r.t. the rate μ can be used as a subgradient w.r.t. the number of servers in this particular case.

We also notice that customers who arrive to the system with more servers, but same total service rate, have a higher probability of being served immediately. That is actually true in the more general $M/G/s$ setting (G stands for general service time distribution): for two systems with the same total service rate, the queue length in the system with more servers serving at a lower rate is stochastically smaller[1] than in the system that has fewer servers serving at the higher rate (Chao and Scott, 2000; Mori, 1975).

4.3.2 Approximating the Subgradients by Gradients Using Rates

We now apply the same logic as in the example above to the call center staffing problem. Let the service rate in period j be equal to μ_j and let $\mu = (\mu_1, \ldots, \mu_p)$. In the original problem $\mu_1 = \ldots = \mu_p$. Let $r_i(\mu; y)$ be the service level function as a function of the rates μ at staffing level y and let $s_i(y; \mu)$ be the original service level function with the service rates μ. In the example above $s_i(y; \mu)$ is the discrete function in Figure 4.4 with $\mu = 1$ and $y \in \{11, \ldots, 20\}$ and $r_i(\mu; y)$ is the continuous function depicted by the dotted line. For the r function in Figure 4.4, $y = 15$ and $\mu \in [11/15, 20/15]$.

The functions r_i and s_i are indeed the same function, i.e., when y and μ are the same for both functions they have the same value. We choose to represent them as two

---

[1] A random variable X is *stochastically smaller* than a random variable Y if $P(X > z) \leq P(Y > z)\ \forall\ z \in \mathbb{R}$ (Ross, 1996, p. 404).

functions to make the distinction clear that in the original problem we are interested in the service level as a function of the number of agents y for a fixed service rate vector μ, but to estimate a gradient we use the service level as a function of the service rate μ for a fixed staffing level y. When we estimate the gradient at different staffing levels we are in fact working with a new function $r_i(\mu; y)$ parameterized by the staffing level y. By definition $r_i(\mu; y) = s_i(y; \mu)$, but as an example, $r_i(2\mu; y) \neq s_i(2y; \mu)$ in general, so even if the functions agree for identical service rates and staffing levels, the effect of increasing the total service rate ($\mu_j y_j$) in each period by changing the service rates is not the same as when the same change is accomplished by changing the staffing levels.

Suppose that we add one agent in period j. Then the total service rate in period j increases by μ_j. If on the other hand we increase the service rate in period j by 1, the total service rate in that period increases by y_j. This suggests that

$$s_i(y + e_j; \mu) - s_i(y; \mu) \approx \frac{\mu_j}{y_j} \frac{r_i(\mu + he_j; y) - r_i(\mu; y)}{h} \rightarrow \frac{\mu_j}{y_j} \frac{\partial r_i(\mu; y)}{\partial \mu_j}, \tag{4.6}$$

as $h \rightarrow 0$ if r_i is indeed differentiable in μ_j.

In the SKCPM we require the subgradient of the *sample average* of the service level function. The FD method approximates this subgradient, but both IPA and the LR method are designed to estimate a gradient of the *expected* value of the performance measure. Most of the effort in showing that one can apply IPA and the LR method is to show that

$$\frac{\partial r_i(\mu; y)}{\partial \mu_j} = \frac{\partial E[R_i(\mu, Z; y)]}{\partial \mu_j} = E\left[\frac{\partial R_i(\mu, Z; y)}{\partial \mu_j}\right],$$

i.e., to establish conditions under which the second inequality holds. The function $R_i(\mu, Z; y)$ is the number of calls received in period i that are answered on time given the staffing level y as a function of the service rates μ and the randomness in Z.

In the LR method we multiply the service level function $R_i(\mu, Z; y)$ by a likelihood

ratio function (more on that later), and to apply IPA we use conditional probability and differentiate a function that has the same expected value as the service level (although the sample average does not have the same value as the sample average of the service level function). While this is perhaps not ideal, we hope that the sample average is close to the expected value. This is certainly true for a large sample size. For IPA, we could redefine the service level function to be the conditional expectation to get a more accurate subgradient estimate of the sample average function.

4.4 Likelihood Ratio Method

In general the likelihood ratio gradient estimation method (Glynn, 1990; Rubenstein and Shapiro, 1993; L'Ecuyer, 1990, 1995) is used to estimate the gradient of the expected value of a function, e.g., $\nabla_\mu r_i(\mu; y) = \nabla_\mu E[R_i(\mu, Z; y)]$, where the expected value is taken with respect to the random vector Z that has some distribution that we denote by P_μ. This distribution can depend on the variables μ, e.g., as a parameter of the distribution of the service times. Then we can write the expected service level as

$$r_i(\mu; y) = E[R_i(\mu, Z; y)] = \int_\Omega R_i(\mu, z; y) dP_\mu(z), \tag{4.7}$$

where the integral is taken over the set Ω of all possible realizations z of the random vector Z.

The random vector Z can be represented in several different ways. For instance, in a simulation setting Z can be thought of as all the random numbers that are used to generate one instance of Z. The transformation from the random numbers to the service times is represented in the R function, and then P_μ does not depend on μ, which is the representation we use in Section 4.5 to derive an IPA gradient estimator. In this section we focus on the other extreme, which is to let only Z depend on μ. Then the function $R_i(\mu, Z; y)$ does not depend on μ, except indirectly through Z. An example of that in the call center setting is to let Z include the actual service and

interarrival times in a single day.

The central idea of the LR method is to rewrite (4.7) such that the distribution that we integrate over does not depend on μ. To do that there must exist a distribution P^* such that P_μ is absolutely continuous[2] with respect to P^* for all μ. We drop μ as an argument of the R function to stress that the dependence on μ is only through P_μ and write

$$r(\mu; y) = \int_\Omega R(z; y) \frac{dP_\mu(z)}{dP^*(z)} dP^*(z). \tag{4.8}$$

The ratio $dP_\mu(z)/dP^*(z)$ is called the *likelihood ratio* and exists by the Radon-Nikodym Theorem (Theorem B.4). Under certain conditions on P_μ, P^* and R

$$\nabla_\mu r(\mu; y) = \int R(z; y) \nabla_\mu \left(\frac{dP_\mu(z)}{dP^*(z)} \right) dP^*(z). \tag{4.9}$$

To see this more intuitively, notice that we have multiplied the integrand in (4.7) by $dP^*(z)/dP^*(z)$. Since $P^*(O) = 0$ implies $P_\mu(O) = 0$ the set of points z where $dP^*(z) = 0$ has P_μ-measure zero and can therefore be excluded from the integral. To estimate the gradient at a single value of μ, say μ^*, P^* can usually be taken as P_{μ^*} (L'Ecuyer, 1990, 1995). Before we implement this for the call center staffing problem we give a simple example to demonstrate the approach.

4.4.1 A Simple Example of Derivative Estimation via the Likelihood Ratio Method

In this subsection we demonstrate the likelihood ratio approach with a (very) simple example that gives some insight into how the LR method can be applied. Suppose that a call center has only one customer. If the customer's call arrives in the first 30 minutes, the service time will be exponential with mean $1/\mu$. If the customer, on the other hand, calls after the 30 minutes are up, the service time for the call is

---

[2]The distribution (or, more generally, measure) P_μ is absolutely continuous w.r.t. P^* if for any measurable set $O \subseteq \Omega$, $P^*(O) = 0$ implies $P_\mu(O) = 0$ (Billingsley, 1995, p. 422).

exponential with mean 10 minutes. The time until the customer calls is exponential with mean 30 minutes. If the call is serviced in less than 5 minutes from when the customer calls in the customer is satisfied. We wish to determine the derivative of the probability that the call's service time is 5 minutes or less.

Solution: Let A be the time of the call, S be the service time of the call, $E_1(\mu)$ be an exponential random variable with mean $1/\mu$ and E_2 be an exponential random variable with mean 10 minutes. Define the indicator function

$$\mathbf{1}\{V\} = \begin{cases} 1 & \text{if condition } V \text{ is satisfied,} \\ 0 & \text{otherwise.} \end{cases} \tag{4.10}$$

Then

$$S = \mathbf{1}\{A \leq 30\}E_1(\mu) + \mathbf{1}\{A > 30\}E_2.$$

Let $r(\mu)$ be the probability that the service is completed in less than 5 minutes. Then

$$\begin{aligned} r(\mu) &= P(S \leq 5) \\ &= E[P(S \leq 5|A)] \\ &= P(E_1(\mu) \leq 5)P(A \leq 30) + P(E_2 \leq 5)P(A > 30) \\ &= (1 - e^{-5\mu})(1 - e^{-30/30}) + (1 - e^{-5/10})e^{-30/30} \\ &= \frac{e-1}{e}(1 - e^{-5\mu}) + \frac{1 - e^{-1/2}}{e}. \end{aligned}$$

Hence,

$$\frac{dr(\mu)}{d\mu} = \frac{e-1}{e}5e^{-5\mu}. \tag{4.11}$$

Alternatively, if we wish to use the likelihood ratio approach we could proceed as follows. Let $F(a) = P(A \leq a) = 1 - e^{-a/30}$ and $G_{\mu,a}(s) = P_\mu(S \leq s|A = a)$ and let

the density of $G_{\mu,a}$ be $g_{\mu,a} = dG_{\mu,a}(s)/ds$. Then,

$$\begin{aligned} r(\mu) &= \int_0^\infty \int_0^\infty \mathbf{1}\{s \le 5\} dG_{\mu,a}(s) dF(a) \\ &= \int_0^\infty \int_0^\infty \mathbf{1}\{s \le 5\} g_{\mu,a}(s) ds\, dF(a). \end{aligned}$$

Then, assuming that we can take the derivative inside the double integral (which can be fairly easily justified by means of the dominated convergence theorem B.2 and the generalized mean value theorem B.3),

$$\begin{aligned} \frac{dr(\mu)}{d\mu} &= \int_0^\infty \int_0^\infty \mathbf{1}\{s \le 5\} \frac{dg_{\mu,a}(s)}{d\mu} ds\, dF(a) \qquad (4.12) \\ &= \int_0^\infty \int_0^5 \frac{dg_{\mu,a}(s)}{d\mu} ds\, dF(a) \\ &= \int_0^{30} \int_0^5 \frac{d(\mu e^{-\mu s})}{d\mu} ds\, dF(a) \\ &= \int_0^{30} dF(a) \int_0^5 e^{-\mu s} - \mu s e^{-\mu s} ds \\ &= \frac{e-1}{e} \left(\int_0^5 e^{-\mu s} ds - \mu \int_0^5 s e^{-\mu s} ds \right) \\ &= \frac{e-1}{e} \left(\int_0^5 e^{-\mu s} ds - \mu \int_0^5 \frac{e^{-\mu s}}{\mu} ds - \mu \left[-\frac{s e^{-\mu s}}{\mu} \right]_{s=0}^{s=5} \right) \\ &= \frac{e-1}{e} 5 e^{-5\mu}. \qquad (4.13) \end{aligned}$$

We see that (4.13) agrees with (4.11), which further motivates the likelihood ratio approach. We can rewrite (4.12) as

$$\begin{aligned} &\int_0^\infty \int_0^\infty \mathbf{1}\{s \le 5\} \frac{dg_{\mu,a}(s)}{d\mu} ds\, dF(a) \\ &= \int_0^\infty \int_0^\infty \mathbf{1}\{s \le 5\} \frac{dg_{\mu,a}(s)}{d\mu} \frac{g_{\mu,a}(s)}{g_{\mu,a}(s)} ds\, dF(a) \\ &= \int_0^\infty \int_0^\infty \mathbf{1}\{s \le 5\} \frac{1}{g_{\mu,a}(s)} \frac{dg_{\mu,a}(s)}{d\mu} dG_{\mu,a}(s) dF(a) \\ &= \int_0^\infty \int_0^\infty \mathbf{1}\{s \le 5\} \mathbf{1}\{a \le 30\} \left(\frac{1}{\mu} - s \right) dG_{\mu,a}(s) dF(a) \\ &= E\left[\mathbf{1}\{S \le 5\} \mathbf{1}\{A \le 30\} \left(\frac{1}{\mu} - S \right) \right], \qquad (4.14) \end{aligned}$$

where the expected value is understood to be with respect to the joint arrival and service time distribution with service rate μ in the first 30 minutes. The expression (4.14) shows how the likelihood ratio derivative would be estimated via simulation.

4.4.2 Likelihood Ratio Gradient Estimation in the Call Center Staffing Problem

In this section we develop a model of the call center staffing problem in order to apply the likelihood ratio method to get a gradient estimator for the expected number of calls received in period i as a function of the service rates in all periods. The model is based on the formulation in Chapter II, but we use the approximations of the service level functions that we derived in Section 4.3.2. There we defined the function $r_i(\mu; y)$ as the expected number of calls received in period i that are answered on time.

There are two aspects of the call center problem that makes it especially challenging to derive a likelihood ratio gradient estimator. The number of calls in a planning horizon is random and the interarrival and service times are dependent. Recall that Z contains the interarrival and service times of all calls in the planning horizon. Suppose that there are C calls in the planning horizon and let $W = (W_1, \ldots, W_C)$ be the arrival times of calls $1, \ldots, C$. Also let $X = (X_1, \ldots, X_C)$ denote the service times of the C calls. In this model we assume that the service time of each call is determined by the service rate in the period in which the call begins service, i.e., once a call enters service it is served at the same rate until it is completed. Thus, the period in which a call enters service depends on its arrival time, and arrival and service times of all the previous calls.

This dependence potentially makes the distributions P_μ and P^* just as difficult to compute as computing the analytical expression for r. On the other hand, once $W_1, \ldots, W_k$ and $X_1, \ldots, X_{k-1}$ are known (as they would be in a simulation study) then it is relatively easy to determine X_k (and W_{k+1}, of course). Thus, we first generate the interarrival times and then generate the service times, depending on in

which period they occur. Let Q be the distribution of the arrival vector W and let $P_{\mu,w}$ be the distribution of the service time vector X, conditional on the arrival times given by the arrival times w, a realization of W. Using this notation we can rewrite (4.7) as

$$r_i(\mu; y) = \iint R_i(x, w; y) dP_{\mu,w}(x) dQ(w).$$

Let B_k be the time at which call k enters service, b_k be a realization of B_k and let $\pi(b_k)$ be the period containing b_k. Then $\mu_{\pi(B_k)}$ is the service rate of the kth call. Let $F(v; \mu, b_k) = P(X_k \leq v | B_k = b_k) \; \forall \; k \in \{1, \ldots, C\}$ denote the conditional distribution function of the service time of call k given that it enters service at time b_k, which is also a function of the service rates μ. Furthermore, assume that X_k has a conditional density $f(v; \mu, b_k) = dF(v; \mu, b_k)/dv$. Then $dP_{\mu,w}(x)$ can be written as

$$dP_{\mu,w}(x) = \prod_{k=1}^{C} f(x_k; \mu, b_k) dx_C \cdots dx_1, \tag{4.15}$$

where C is a constant since the arrival times w are known.

In (4.15), the time when a call enters service b_k is only known when all the previous service times are known, so the densities are all dependent. Verifying that the differentiation can be taken inside the integral in (4.9) is more difficult when this dependence exists. If, however, the service rate μ is a *scale parameter* of a family of distributions, i.e., there exist a random variable $\hat{X}$ such that $\hat{X}/\mu_{\pi(B_k)}$ has the same distribution as the service time of the kth call, then the problem simplifies somewhat. Thus, if we let X_k denote the random variable for the service time of the kth call then $X_k = \hat{X}_k/\mu_{\pi(B_k)}$ where $\hat{X}, \hat{X}_1, \hat{X}_2, \ldots$ are i.i.d. random variables that do not dependent on the period in which the corresponding call enters service. Let $\hat{F}$ be the distribution of $\hat{X}$. Then, $F(v; \mu, b_k) = P\{\hat{X}/\mu_{\pi(b_k)} \leq v\} = \hat{F}(\mu_{\pi(b_k)} v)$. Furthermore, we assume that $\hat{F}$ is differentiable with derivative $\hat{f}(v) = d\hat{F}(v)/dv$. It follows that

$$f(v; \mu, b_k) = \frac{dF(v; \mu, b_k)}{dv} = \frac{d\hat{F}(\mu_{\pi(b_k)} v)}{dv} = \mu_{\pi(b_k)} \hat{f}(\mu_{\pi(b_k)} v).$$

The gamma, Weibull and exponential distributions all have a scale parameter.

When the rates are scale parameters it is easy to see that if we fix the rates at μ^*, say, and require that $\hat{f}(x) > 0 \; \forall \; x > 0$ then the distribution $P_{\mu,w}(x)$ is absolutely continuous w.r.t. $P_{\mu^*,w}(x)$, since for any $v \in \mathbb{R}$,

$$f(v; \mu^*, b_k) = 0 \Rightarrow \hat{f}(\mu^*_{\pi(b)} v) = 0.$$

Therefore, we can use

$$\frac{dP_{\mu,w}(x)}{dP_{\mu^*,w}(x)} = \frac{\prod_{k=1}^{C} f(x_k; \mu, b_k)}{\prod_{k=1}^{C} f(x_k; \mu^*, b_k)} = \frac{\prod_{k=1}^{C} \mu_{\pi(b_k)} \hat{f}(\mu_{\pi(b_k)} x_k)}{\prod_{k=1}^{C} \mu^*_{\pi(b_k)} \hat{f}(\mu^*_{\pi(b_k)} x_k)} \tag{4.16}$$

as the likelihood ratio. Hence,

$$\begin{aligned} r_i(\mu; y) &= \iint R_i(x, w; y) dP_{\mu,w}(x) dQ(w) \\ &= \iint R_i(x, w; y) \frac{\prod_{k=1}^{C} \mu_{\pi(b_k)} \hat{f}(\mu_{\pi(b_k)} x_k)}{\prod_{k=1}^{C} \mu^*_{\pi(b_k)} \hat{f}(\mu^*_{\pi(b_k)} x_k)} dP_{\mu^*,w}(x) dQ(w). \end{aligned} \tag{4.17}$$

When this is implemented we run the simulation using the rates μ^* and generate the arrival times w. The epochs b_k, and hence the service times x_k, will be determined by $x_1, \ldots, x_{k-1}$ and w and μ^*.

For the model that we developed above there indeed exists an unbiased likelihood ratio estimator at $\mu = \mu^*$ as summarized in the following theorem.

Theorem 4.2. *Suppose that*

i. $\hat{f}(v) > 0$ *for all* $v > 0$, $\hat{f}(\cdot)$ *is continuous and piecewise differentiable.*

ii. $0 < \mu^*_j < \infty \; \forall \; j \in \{1, \ldots, p\}$.

iii. There exist functions $\beta(v)$ *and* $\gamma(v)$ *and a neighborhood of 1,* $\mathcal{N}(1)$*, such that*

$$\frac{\hat{f}(\alpha v)}{\hat{f}(v)} \leq \beta(v) \quad \textit{and} \quad \frac{|\hat{f}'(\alpha v)|}{\hat{f}(v)} \leq \gamma(v)$$

for all $\alpha \in \mathcal{N}(1)$*. Furthermore,* $E[\beta(\hat{X})] < \infty$ *and* $E[\hat{X}\gamma(\hat{X})] < \infty$.

iv. For any constant $M \geq 0$

$$\int M^C dQ(w) < \infty. \tag{4.18}$$

Then

$$\nabla_\mu r_i(\mu;y)|_{\mu=\mu^*} = \iint R_i(x,w;y)\frac{\nabla_\mu \left(\prod_{k=1}^C f(x_k;\mu_{\pi(b_k)})\right)\Big|_{\mu=\mu^*}}{\prod_{k=1}^C f(x_k;\mu^*_{\pi(b_k)})} dP_{\mu^*,w}(x)dQ(w). \tag{4.19}$$

Proof: First

$$\begin{aligned} r_i(\mu;y) &= \iint R_i(x,w;y)\frac{\prod_{k=1}^C f(x_k;\mu_{\pi(b_k)})}{\prod_{k=1}^C f(x_k;\mu^*_{\pi(b_k)})} dP_{\mu^*,w}(x)dQ(w) \\ &= \iint R_i(x,w;y)\frac{\prod_{k=1}^C \mu_{\pi(b_k)}\hat{f}(x_k\mu_{\pi(b_k)})}{\prod_{k=1}^C \mu^*_{\pi(b_k)}\hat{f}(x_k\mu^*_{\pi(b_k)})} dP_{\mu^*,w}(x)dQ(w) \\ &= \iint R_i(x,w;y)\frac{\left(\prod_{k=1}^C \mu_{\pi(b_k)}\right)\left(\prod_{k=1}^C \hat{f}(x_k\mu_{\pi(b_k)})\right)}{\prod_{k=1}^C \mu^*_{\pi(b_k)}\hat{f}(x_k\mu^*_{\pi(b_k)})} dP_{\mu^*,w}(x)dQ(w). \end{aligned}$$

Define

$$I(\mu,\mu^*,w,x) \equiv R_i(x,w;y)\frac{\left(\prod_{k=1}^C \mu_{\pi(b_k)}\right)\left(\prod_{k=1}^C \hat{f}(x_k\mu_{\pi(b_k)})\right)}{\prod_{k=1}^C \mu^*_{\pi(b_k)}\hat{f}(x_k\mu^*_{\pi(b_k)})}. \tag{4.20}$$

Then $I(\cdot,\mu^*,w,x)$ is continuous and piecewise differentiable by assumption i and

$$\left|\frac{I(\mu+he_j,\mu^*,w,x)-I(\mu,\mu^*,w,x)}{h}\right| \le \sup_{\underline{\mu}\le\mu\le\bar{\mu}}\left|\frac{\partial I(\mu,\mu^*,w,x)}{\partial\mu_j}\right| \tag{4.21}$$

by the generalized mean value theorem (Theorem B.3) for $\underline{\mu} \le \mu \le \bar{\mu}$ and $\underline{\mu} \le \mu + he_j \le \bar{\mu}$, where $0 < \underline{\mu}_k \le \bar{\mu}_k < \infty$ for all k. Below we bound the right hand side of (4.21) by an integrable function, and hence we can apply the dominated convergence theorem (Theorem B.2) to conclude that (4.19) holds. Now,

$$I' \equiv \frac{\partial I(\mu,\mu^*,w,x)}{\partial\mu_j} = I(\mu,\mu^*,w,x)\sum_{k=1}^C \mathbf{1}\{\pi(b_k)=j\}\left(x_k\frac{\hat{f}'(x_k\mu_{\pi(b_k)})}{\hat{f}(x_k\mu_{\pi(b_k)})}+\frac{1}{\mu_{\pi(b_k)}}\right),$$

so

$$
\begin{aligned}
|I'| &= R_i(x,w;y)\left(\prod_{k=1}^{C}\frac{\mu_{\pi(b_k)}}{\mu^*_{\pi(b_k)}}\frac{\hat f(x_k\mu_{\pi(b_k)})}{\hat f(x_k\mu^*_{\pi(b_k)})}\right)\\
&\quad\times\left|\sum_{k=1}^{C}\mathbf{1}\{\pi(b_k)=j\}\left(x_k\frac{\hat f'(x_k\mu_{\pi(b_k)})}{\hat f(x_k\mu_{\pi(b_k)})}+\frac{1}{\mu_{\pi(b_k)}}\right)\right|\\
&\le C\left(\frac{\mu_U}{\mu^*_L}\right)^C\left(\prod_{k=1}^{C}\frac{\hat f(x_k\mu_{\pi(b_k)})}{\hat f(x_k\mu^*_{\pi(b_k)})}\right)\left(\frac{C}{\mu_L}+\sum_{k=1}^{C}\mathbf{1}\{\pi(b_k)=j\}\frac{\mu^*_{\pi(b_k)}}{\mu^*_{\pi(b_k)}}x_k\frac{|\hat f'(x_k\mu_{\pi(b_k)})|}{\hat f(x_k\mu_{\pi(b_k)})}\right)\\
&\le C\left(\frac{\mu_U}{\mu^*_L}\right)^C\left(\prod_{k=1}^{C}\frac{\hat f(x_k\mu_{\pi(b_k)})}{\hat f(x_k\mu^*_{\pi(b_k)})}\right)\left(\frac{C}{\mu_L}+\frac{1}{\mu^*_L}\sum_{k=1}^{C}\mu^*_{\pi(b_k)}x_k\frac{|\hat f'(x_k\mu_{\pi(b_k)})|}{\hat f(x_k\mu_{\pi(b_k)})}\right), \qquad (4.22)
\end{aligned}
$$

where $\mu_L = \min_j \underline{\mu}_j$, $\mu_U = \max_j \bar\mu_j$ and $\mu^*_L = \min_j \mu^*_j$. By assumption *iii*, for μ sufficiently close to μ^*,

$$
\prod_{k=1}^{C}\frac{\hat f(x_k\mu_{\pi(b_k)})}{\hat f(x_k\mu^*_{\pi(b_k)})}\le\prod_{k=1}^{C}\beta(x_k\mu^*_{\pi(b_k)})
$$

and

$$
\begin{aligned}
&\left(\prod_{k=1}^{C}\frac{\hat f(x_k\mu_{\pi(b_k)})}{\hat f(x_k\mu^*_{\pi(b_k)})}\right)\sum_{k=1}^{C}\mu^*_{\pi(b_k)}x_k\frac{|\hat f'(x_k\mu_{\pi(b_k)})|}{\hat f(x_k\mu_{\pi(b_k)})}\\
&=\sum_{k=1}^{C}\left(\mu^*_{\pi(b_k)}x_k\frac{|\hat f'(x_k\mu_{\pi(b_k)})|}{\hat f(x_k\mu_{\pi(b_k)})}\prod_{m=1}^{C}\frac{\hat f(x_m\mu_{\pi(b_m)})}{\hat f(x_m\mu^*_{\pi(b_m)})}\right)\\
&=\sum_{k=1}^{C}\left(\mu^*_{\pi(b_k)}x_k\frac{|\hat f'(x_k\mu_{\pi(b_k)})|}{\hat f(x_k\mu^*_{\pi(b_k)})}\prod_{\substack{m=1\\m\neq k}}^{C}\frac{\hat f(x_m\mu_{\pi(b_m)})}{\hat f(x_m\mu^*_{\pi(b_m)})}\right)\\
&\le\sum_{k=1}^{C}\left(\mu^*_{\pi(b_k)}x_k\gamma(x_k\mu^*_{\pi(b_k)})\prod_{\substack{m=1\\m\neq k}}^{C}\beta(x_m\mu^*_{\pi(b_m)})\right).
\end{aligned}
$$

Next,

$$\int \sum_{k=1}^{C} \left(\mu^*_{\pi(b_k)} x_k \gamma(x_k \mu^*_{\pi(b_k)}) \prod_{\substack{m=1 \\ m \neq k}}^{C} \beta(x_m \mu^*_{\pi(b_m)}) \right) dP_{\mu^*,w}(x)$$

$$= \sum_{k=1}^{C} \int \mu^*_{\pi(b_k)} x_k \gamma(x_k \mu^*_{\pi(b_k)}) \left(\prod_{\substack{m=1 \\ m \neq k}}^{C} \beta(x_m \mu^*_{\pi(b_m)}) \right) dP_{\mu^*,w}(x)$$

$$= \sum_{k=1}^{C} \int \cdots \int \mu^*_{\pi(b_k)} x_k \gamma(x_k \mu^*_{\pi(b_k)}) \mu^*_{\pi(b_k)} \hat{f}(\mu^*_{\pi(b_k)} x_k)$$

$$\left(\prod_{\substack{m=1 \\ m \neq k}}^{C} \beta(x_m \mu^*_{\pi(b_m)}) \mu^*_{\pi(b_m)} \hat{f}(\mu^*_{\pi(b_m)} x_m) \right) dx_C \ldots dx_1$$

$$= \sum_{k=1}^{C} \int \cdots \int \hat{x}_k \gamma(\hat{x}_k) \mu^*_{\pi(b_k)} \hat{f}(\hat{x}_k) \left(\prod_{\substack{m=1 \\ m \neq k}}^{C} \beta(\hat{x}_m) \mu^*_{\pi(b_m)} \hat{f}(\hat{x}_m) \right) |J| d\hat{x}_C \ldots d\hat{x}_1 \quad (4.23)$$

where (4.23) is obtained by a change of variables, i.e., $\hat{x}_k = \mu^*_{\pi(b_k)} x_k$, and $|J|$ is the determinant of the Jacobian associated with the change of variables. To compute the Jacobian we note that b_k is a function of $x_1, \ldots, x_{k-1}$ so

$$\frac{\partial \hat{x}_k}{\partial x_l} = \begin{cases} x_k \frac{\partial \mu_{\pi(b_k)}}{\partial x_l} & \text{for } l < k, \\ \mu^*_{\pi(b_k)} & \text{for } l = k, \\ 0 & \text{for } l > k. \end{cases}$$

Define the matrix $\tilde{J}$ as $\tilde{J}_{kl} = \partial \hat{x}_k / \partial x_l$. Then the Jacobian is $\tilde{J}^{-1}$. The determinant of a triangular matrix is the product of its diagonal elements (Venit and Bishop, 1989, p. 120), and the determinant of the inverse of a matrix is the inverse of the determinant of the matrix. Hence,

$$|J| = \det(J) = \frac{1}{\det(\tilde{J})} = \frac{1}{\prod_{k=1}^{C} J_{kk}} = \prod_{k=1}^{C} \frac{1}{\mu^*_{\pi(b_k)}}.$$

Inserting this into (4.23), we get

$$\begin{aligned}
&\int \sum_{k=1}^{C} \left(\mu^*_{\pi(b_k)} x_k \gamma(x_k \mu^*_{\pi(b_k)}) \prod_{\substack{m=1 \\ m \neq k}}^{C} \beta(x_m \mu^*_{\pi(b_m)}) \right) dP_{\mu^*,w}(x) \\
&= \sum_{k=1}^{C} \int \cdots \int \hat{x}_k \gamma(\hat{x}_k) \hat{f}(\hat{x}_k) \left(\prod_{\substack{m=1 \\ m \neq k}}^{C} \beta(\hat{x}_m) \hat{f}(\hat{x}_m) \right) d\hat{x}_C \ldots d\hat{x}_1 \\
&= \sum_{k=1}^{C} \int \hat{x}_k \gamma(\hat{x}_k) \hat{f}(\hat{x}_k) d\hat{x}_k \prod_{\substack{m=1 \\ m \neq k}}^{C} \int \beta(\hat{x}_m) \hat{f}(\hat{x}_m) d\hat{x}_m \\
&= \sum_{k=1}^{C} E[\hat{X}\gamma(\hat{X})] \prod_{\substack{m=1 \\ m \neq k}}^{C} E[\beta(\hat{X})] \\
&= C E[\hat{X}\gamma(\hat{X})] (E[\beta(\hat{X})])^{C-1}. \qquad (4.24)
\end{aligned}$$

Similarly,

$$\int \prod_{k=1}^{C} \beta(x_k \mu^*_{\pi(b_k)}) dP_{\mu^*,w}(x) = (E[\beta(\hat{X})])^C. \qquad (4.25)$$

Thus if we integrate (4.22) with respect to $P_{\mu^*,w}(x)$, and since $E[\hat{X}\gamma(\hat{X})] < \infty$ and $E[\beta(\hat{X})] < \infty$, we get

$$C^2 \left(\frac{\mu_U}{\mu^*_L} \right)^C (E[\beta(\hat{X})])^{C-1} \left(\frac{E[\beta(\hat{X})]}{\mu_L} + \frac{E[\hat{X}\gamma(\hat{X})]}{\mu^*_L} \right) \leq a M^C$$

for some constants a and M. Finally, since $\int a M^C dQ(w) < \infty$, we can conclude that (4.19) holds. □

The final step is to translate (4.19) into an approximation of the subgradient of the service level function $s(y;\mu)$. We use (4.6) to get

$$(q_i)_j(\hat{y}) \approx \left(\frac{\mu_j}{\hat{y}_j} \frac{\partial r_i(\mu;\hat{y})}{\partial \mu_j} \right)_{\mu=\mu^*}, \qquad (4.26)$$

where the partial derivative is estimated by (4.19) and μ^* are the actual service rates.

Equation (4.26) is an approximation of the subgradient of the *expected* service level function. Our objective was to compute a subgradient that can be used to

create a valid cut for the SAA of the call center problem. For the cut to be valid we require a subgradient of the *sample average* of the service level function.

For a fixed $\hat{y}$ and at μ^* the sample average of $r(\mu; y)$ as defined by (4.8) agrees with the sample average of the service level function $s(y; \mu)$ since the likelihood ratio $dP_\mu(z)/dP^*(z)$ at $\mu = \mu^*$ equals 1 when $P^* = P_{\mu^*}$. Equation (4.19) gives a gradient of the sample average of r. Now, if we change either the staffing levels to y' or the service rates to μ' (i.e., let $P^* = P_{\mu'}$) then we get a new sample average function for r. Therefore, we cannot guarantee that a gradient estimated by (4.26) is a subgradient of the sample average of r.

The bound on the arrival process (4.18) is a rather strong condition although it holds, for example, for a Poisson arrival process. In the next section we investigate whether the gamma, exponential and Weibull distributions satisfy the conditions on the densities in Theorem 4.2.

4.4.3 Examples of when the Conditions on the Service Time Distributions Are Satisfied

In this section we study whether 3 distributions that are commonly used to model service times, namely the exponential, gamma and Weibull distributions, satisfy the conditions in Theorem 4.2. The exponential distribution is in fact a special case of the other two, but because of how widely the exponential distribution is used and because of how simple it is, we discuss it separately. Condition i is satisfied for these distributions and we will assume that condition ii holds. Condition iv is not related to the service time distribution, so our concern here is to verify condition iii. We will derive the functions β and γ. These functions will have constants that will determine the size of the neighborhood $\mathcal{N}(1)$ in which α must lie for the bounds in iii to hold.

The exponential distribution. The density of the exponential distribution with rate μ is

$$f(v;\mu) = \begin{cases} \mu e^{-\mu v} & \text{for } v > 0, \\ 0 & \text{for } v \leq 0. \end{cases}$$

Thus we can let

$$\hat{f}(v) = \begin{cases} e^{-v} & \text{for } v > 0, \\ 0 & \text{for } v \leq 0. \end{cases}$$

In this case for $v > 0$

$$\frac{\hat{f}(\alpha v)}{\hat{f}(v)} = \frac{|\hat{f}'(\alpha v)|}{\hat{f}(v)} = \frac{e^{-\alpha v}}{e^{-v}} = e^{(1-\alpha)v},$$

so we can let

$$\beta(v) = \gamma(v) = e^{\delta v}$$

for some constant $\delta > 0$ satisfying $\delta \geq 1 - \alpha$, i.e., we can fix δ and restrict $\alpha \geq 1 - \delta$. Next, $E[e^{\delta \hat{X}}]$ is the moment generating function of $\hat{X}$ and is defined for $\delta < 1$. Furthermore, $E[\hat{X}e^{\delta \hat{X}}]$ is the first derivative of the moment generating function of $\hat{X}$ and is also defined for $\delta < 1$. Therefore, we can choose $\delta \in (0, 1)$, and then $\mathcal{N}(1) = [1 - \delta, \infty)$.

The gamma distribution. The density of the gamma distribution with rate $\mu > 0$ and shape parameter $r > 0$ is (when $r = 1$ this is the exponential distribution)

$$f(v;\mu) = \begin{cases} \mu \frac{(\mu v)^{r-1} e^{-\mu v}}{\Gamma(r)} & \text{for } v > 0, \\ 0 & \text{for } v \leq 0. \end{cases}$$

Thus, we can let

$$\hat{f}(v) = \begin{cases} \frac{v^{r-1} e^{-v}}{\Gamma(r)} & \text{for } v > 0, \\ 0 & \text{for } v \leq 0, \end{cases}$$

where $\Gamma(r)$ is the gamma function and is a constant for a fixed r, so it does not play a role in the subsequent analysis. In this case, for $v > 0$

$$\frac{\hat{f}(\alpha v)}{\hat{f}(v)} = \frac{(\alpha v)^{r-1} e^{-\alpha v}}{v^{r-1} e^{-v}} = \alpha^{r-1} e^{(1-\alpha)v},$$

so we can let

$$\beta(v) = \delta_1 e^{\delta_2 v}$$

for some constants $\delta_1 > 0$ and δ_2 satisfying $\delta_1 \geq \alpha^{r-1}$ and $\alpha \geq 1 - \delta_2$, so we can choose any $\delta_2 > 0$. For $r > 1$, δ_1 is going to be an upper bound on α, i.e., $\alpha \leq (\delta_1)^{1/(r-1)}$, so $\delta_1 > 1$. For $r < 1$ we get $\alpha > \delta_1^{-1/(1-r)} = \delta_1^{1/(r-1)}$, so $\delta_1 < 1$. Then we see that δ_2 must satisfy the same requirements as δ for the exponential distribution. $E[e^{\delta_2 \hat{X}}]$ is the moment generating function for the gamma distribution and is defined for $\delta_2 < 1$.

Next,

$$\begin{aligned}
\frac{|\hat{f}'(\alpha v)|}{\hat{f}(v)} &= \frac{|r - 1 - \alpha v|(\alpha v)^{r-2} e^{-\alpha v}}{v^{r-1} e^{-v}} \\
&\leq \frac{|r-1|}{v} \alpha^{r-2} e^{(1-\alpha)v} + \alpha^{r-1} e^{(1-\alpha)v} \\
&\leq \left(\frac{|r-1|}{\alpha v} + 1 \right) \beta(v),
\end{aligned}$$

so we can let

$$\gamma(v) = \frac{\delta_3}{v} \beta(v) + \beta(v),$$

for some constant $\delta_3 > 0$ satisfying $\delta_3 \geq |r-1|/\alpha$ or $\alpha \geq |r-1|/\delta_3$. Hence, we need $\delta_3 < 1$. Finally,

$$E[\hat{X}\gamma(\hat{X})] = \delta_3 E[\hat{X}\beta(\hat{X})/\hat{X}] + E[\hat{X}\beta(\hat{X})].$$

The first expected value is finite since $E[\beta(\hat{X})] < \infty$. The second expected value is the derivative of the moment generating function and is also defined for $\delta_2 < 1$. Thus we can find a neighborhood around 1 so that condition *iii* of Theorem 4.2 is satisfied.

The Weibull distribution. The density of the Weibull distribution with rate $\mu > 0$ and shape parameter $r > 0$ is (when $r = 1$, this is the exponential distribution)

$$f(v;\mu) = \begin{cases} \mu r(\mu v)^{r-1} e^{-(\mu v)^r} & \text{for } v > 0 \\ 0 & \text{for } v \le 0. \end{cases}$$

Thus we can let

$$\hat{f}(v) = \begin{cases} r v^{r-1} e^{-v^r} & \text{for } v > 0 \\ 0 & \text{for } v \le 0. \end{cases}$$

In this case, for $v > 0$,

$$\frac{\hat{f}(\alpha v)}{\hat{f}(v)} = \frac{r(\alpha v)^{r-1} e^{-(\alpha v)^r}}{r v^{r-1} e^{-v^r}} = \alpha^{r-1} e^{(1-\alpha^r)v^r},$$

so we can let

$$\beta(v) = \delta_1 e^{\delta_2 v^r}.$$

We can choose δ_1 similarly as for the gamma distribution. To choose δ_2 we need

$$\int_0^\infty e^{\delta_2 v^r} r v^{r-1} e^{-v^r} dv = \int_0^\infty e^{(\delta_2 - 1)y} dy < \infty,$$

or $\delta_2 < 1$ and $\alpha \ge \sqrt[r]{1 - \delta_2}$.

Now,

$$\begin{aligned} \frac{|\hat{f}'(\alpha v)|}{\hat{f}(v)} &= \frac{|r - 1 - r(\alpha v)^r| r(\alpha v)^{r-2} e^{-(\alpha v)^r}}{r v^{r-1} e^{-v^r}} \\ &\le \frac{|r-1|}{v} \alpha^{r-2} e^{(1-\alpha^r)v^r} + r\alpha^{r-1} v^{r-1} e^{(1-\alpha^r)v} \\ &\le \frac{|r-1|}{\alpha v} \beta(v) + r\alpha^{r-1} v^{r-1} e^{(1-\alpha^r)v}, \end{aligned}$$

so we can let

$$\gamma(v) = \delta_3 \beta(v) + \delta_4 v^{r-1} e^{\delta_2 v^r},$$

where δ_3 can be chosen as δ_3 for the gamma distribution. We need $\delta_4 \ge r\alpha^{r-1}$, or $\alpha \le (\delta_4/r)^{1/(r-1)}$. Hence, if $r > 1$ we can choose $\delta_4 > r$ and if $r < 1$ we choose $\delta_4 < r$.

We see that $E[\hat{X}\gamma(\hat{X})] < \infty$ when $\delta_2 < 1$.

In conclusion, Theorem 4.2 holds for both the gamma and Weibull distributions for any value of the shape parameter r.

4.5 Infinitesimal Perturbation Analysis

So far we studied two different methods to approximate a subgradient of the service level function. In this section we study a third method, infinitesimal perturbation analysis (e.g. Glasserman, 1991).

An IPA gradient estimator is a gradient of the sample path function with respect to the variable of interest. Computing the gradient of the sample path function is a challenging task for several reasons. We now describe these challenges and how they can be overcome. More details follow in the subsequent subsections.

We obviously need a sample path function that is indeed differentiable. In this case the variables of interest are the number of servers in each period. These variables are integer so the sample path function is not differentiable. Therefore, to compute an IPA gradient estimator the service level function is first approximated with a function of the service rate in each period as described in Section 4.3.

Given that we are now dealing with a function of continuous variables, is the function also continuous? No, because the service level in any period as a function of the service rates on each sample path is a finite sum of indicator functions where the jth indicator function equals 1 if the jth call arrives in the period and begins service on time. The derivative of this function with respect to any of the service rates equals zero where it exists and is undefined at the points where there is a jump in any of the indicator functions. It is reasonable to assume, however, that the *expected* service level function is differentiable everywhere and that the gradient is nonzero in general.

We formulate a different representation of the service level function that may differ on each sample path from the original service level function but has the same

expected value. This procedure is called *smoothing* (Glasserman, 1991; Fu and Hu, 1997). Smoothing is done by conditioning on some of the random elements in the problem in order to obtain a function that is continuous on every (or almost every) sample path. At the same time, the conditioning argument ensures that this new function has the same expected value as the original function. In this case, for each call we condition on the value of the interarrival and service times of the previous calls so the information that we condition on increases with time, as in filtered Monte Carlo (Glasserman, 1993).

The next step is to show that the smoothed function is continuous and piecewise differentiable in μ. For a general y this is not the case. If, however, the number of servers in all periods is constant and if the service rate changes instantaneously, then we show that the smoothed function is indeed continuous and piecewise differentiable on each sample path for bounded and positive service rates. By an instantaneous change in service rates we mean that when a new period starts, then all calls in service are served at the service rate of the new period rather than at the service rate of the period when the call entered service. Thus a service time of a call will depend on what period it is being served in, as opposed to only the service rate that is in effect when it starts service as in the model in the previous section.

In summary the modeling steps taken are:

1. Use service rates as variables instead of number of servers.

2. Smooth the service level function on each sample path.

3. Fix the number of servers in all periods (and adjust the rates accordingly).

4. Use service rates as speeds, so that a server will work at a specific rate depending on the period.

4.5.1 A Model of the Call Center that Has a Fixed Number of Servers

In this section we describe the model that is used to derive an unbiased IPA gradient estimator with respect to the service rates in each period for the service level function in a call center that has a fixed number, ζ, of servers in all periods. Let U_k be the time between the arrivals of the $(k-1)$st and kth call. We assume that the distribution of U_k depends only on the arrival time of the $(k-1)$st call. Let $F(v;t)$ be the probability at time t that the time until the next call is less than v given that the previous call arrived at time t. We assume that $F(v;t)$ is continuous and differentiable in v for all $t \in [0,\infty)$ and $F(0;\cdot) = 0$. We further assume that the expected number of calls received in any finite amount of time is finite and that the calls are answered in the order they are received. Note that this model of the arrival process includes as a special case the frequently used nonhomogeneous Poisson process.

Each call requires a random amount of work, which we also assume can be modeled as a sequence of independent random variables, $\{X_k\}_{k=1}^{\infty}$, where X_k is the amount of work required to finish call k and has a distribution function G with $G(0) = 0$. Note that G can be either a discrete or continuous distribution function.

We assume that in period i each server performs work at a rate μ_i, $i \in \{1,\ldots,p\}$. For instance, if call k enters service in period i then, X_k/μ_i is the service time of call k *should it finish service before the end of period* i. Otherwise, if there are only t time units left in period i, then the residual load at the end of the period is given by $X_k - t\mu_i$. This recursion does not play a particular role in our development but hopefully lends some insight into how calls are served in this model.

Earlier we defined $R_i(\mu; y, Z)$ as the number of calls received in period i that are answered within τ time units. Here we let $Z = (U_1,\ldots,U_C,X_1,\ldots,X_C)$ where C is the number of calls received. The number of servers is fixed so we can omit the y

variable from R and redefine $R_i(\mu; y, Z)$ as $R_i(\mu; Z)$. Also define

$$L_k^i(\mu; Z) = \begin{cases} 1 & \text{if call } k \text{ arrives in period } i \text{ and} \\ & \text{has a waiting time less than or equal to } \tau, \\ 0 & \text{otherwise,} \end{cases}$$

so that $R_i(\mu; Z) = \sum_{k=1}^{\infty} L_k^i(\mu; Z)$. One problem with using IPA is that on a particular sample path, L_k^i is a step function of the rates in μ and hence the gradient is zero where it exists. Before we apply IPA we need to smooth the service level function so that it is differentiable on each sample path. We use conditioning in the next section to smooth the service level function.

The number of calls received in period i that have waiting time less than τ is on average

$$E[R_i(\mu; Z)] = E\left[\sum_{k=1}^{\infty} L_k^i(\mu; Z)\right] = \sum_{k=1}^{\infty} E[L_k^i(\mu; Z)], \tag{4.27}$$

where the second equality follows from Fubini's theorem (Billingsley, 1995), which holds since L_k^i is nonnegative. Furthermore, the expression in (4.27) is finite since

$$E\left[\sum_{k=1}^{\infty} L_k^i(\mu; Z)\right] \leq E[\text{total number of calls received in period } i] < \infty$$

by assumption.

The goal is to develop an unbiased gradient estimator of (4.27) w.r.t. the service rates μ, i.e., we want to estimate

$$\frac{\partial E[R_i(\mu; Z)]}{\partial \mu_j} \qquad \forall\, j \in \{1, \ldots, p\}. \tag{4.28}$$

We will focus on only one of these partial derivatives since they are all computed in the same way. For an arbitrary $j \in \{1, \ldots, p\}$ define

$$R_i(u; Z) = R_i(\mu_1, \ldots, \mu_{j-1}, u, \mu_{j-1}, \ldots, \mu_p; Z) \text{ and}$$
$$L_k^i(u; Z) = L_k^i(\mu_1, \ldots, \mu_{j-1}, u, \mu_{j-1}, \ldots, \mu_p; Z).$$

In essence, we assume that all elements except the jth element of the service rate vector are constant. This is a slight abuse of notation but makes the statements and derivation of the results that follow clearer, since now we are only dealing with a one dimensional function. Since j is arbitrary we get the partial derivatives in (4.28) by computing

$$\frac{dE[R_i(u;Z)]}{du} \equiv \frac{\partial E[R_i(\mu;Z)]}{\partial \mu_j}.$$

4.5.2 Smoothing the Service Level Function

We now use conditioning to smooth the service level function. For call k we condition on all the random quantities up to (but not including) the time of arrival of that call. We define $\mathcal{Z}_k = \sigma\{U_1, \ldots, U_k, X_1, \ldots, X_k\}$ (we condition on $\mathcal{Z}_{k-1}$). Here, $\sigma\{Z\}$ denotes the sigma-algebra generated by the random variables in Z (Billingsley, 1995).

Also let

$$\begin{aligned} W_k &= \sum_{i=1}^{k} U_i \text{ be the arrival time of call } k, \\ T_k(u) &= \text{ waiting time in queue of call } k \text{ not counting service}, \\ V_i &= \text{ end of period } i. \end{aligned}$$

Then period i starts at $V_{i-1}(V_0 \equiv 0)$ and ends at V_i, and

$$L_k^i(u) = \mathbf{1}\{T_k(u) \leq \tau, V_{i-1} < W_k \leq V_i\}.$$

The waiting time of call k is the difference between the time when call k starts service and the time when it arrived. If there is at least one empty server when call k arrives, then it starts service immediately and the waiting time is 0. Otherwise, call k starts service as soon as a server becomes available to serve call k. Define

$$\xi_k(u) \tag{4.29}$$

as the time when a server becomes available to serve call k as a function of the

service rate u. Note that $\xi_k(u) < W_k$ if a server is free when call k arrives. Using this notation, $T_k(u) = (\xi_k(u) - W_k)^+$, where $z^+ = \max\{0, z\}$. Then,

$$
\begin{aligned}
E[L_k^i(u)] &= E[\mathbf{1}\{(\xi_k(u) - W_k)^+ \le \tau, V_{i-1} < W_k \le V_i\}] \\
&= P(\xi_k(u) - W_k \le \tau, V_{i-1} < W_k \le V_i) && (4.30) \\
&= E[P(\xi_k(u) - W_k \le \tau, V_{i-1} < W_k \le V_i | \mathcal{Z}_{k-1})] && (4.31) \\
&= E[P(\xi_k(u) - \tau \le W_{k-1} + U_k, V_{i-1} < W_{k-1} + U_k \le V_i | \mathcal{Z}_{k-1})] && (4.32) \\
&= E[P(\max\{V_{i-1}, \xi_k(u) - \tau\} - W_{k-1} \le U_k \le V_i - W_{k-1} | \mathcal{Z}_{k-1})] \\
&= E[\mathbf{1}\{\max\{V_{i-1}, \xi_k(u) - \tau\} - W_{k-1} \le V_i - W_{k-1}\} \\
&\quad (F(V_i - W_{k-1}; W_{k-1}) - F(\max\{V_{i-1}, \xi_k(u) - \tau\} - W_{k-1}; W_{k-1}))] && (4.33) \\
&= E[\mathbf{1}\{\xi_k(u) - \tau \le V_i\}(F(V_i - W_{k-1}; W_{k-1}) \\
&\quad - F(\max\{V_{i-1}, \xi_k(u) - \tau\} - W_{k-1}; W_{k-1}))]. && (4.34)
\end{aligned}
$$

Relation (4.30) follows since $\{(\xi_k(u) - W_k)^+ \le \tau\}$ if and only if $\{\xi_k(u) - W_k \le \tau\}$ for $\tau \ge 0$, (4.31) follows by conditioning on $\mathcal{Z}_{k-1}$, (4.32) follows since $W_k = W_{k-1} + U_k$, (4.33) follows since U_k has c.d.f. F and $F(v; \cdot) = 0$ for $v \le 0$, and (4.34) follows since $V_{i-1} < V_i$.

Define for $v \in [0, \infty)$

$$
\begin{aligned}
J_i(v; a) &= \mathbf{1}\{v - \tau \le V_i\}(F(V_i - a; a) - F(\max\{V_{i-1}, v - \tau\} - a; a)) && (4.35) \\
&= \begin{cases} F(V_i - a; a) - F(V_{i-1} - a; a) & v \le V_{i-1} + \tau, \\ F(V_i - a; a) - F(v - \tau - a; a) & V_{i-1} + \tau < v \le V_i + \tau, \\ 0 & V_i + \tau < v, \end{cases}
\end{aligned}
$$

i.e., $J_i(v; a)$ is the probability, given the arrival time of the previous call is a, that call k arrives in period i and will be answered on time as a function of the time, v, when a server becomes available to answer call k when the arrival time of the previous call is a. Since $F(v; t)$ is continuous in v and $F(0; W_{k-1}) = 0$ it is clear that $J_i(v; a)$ is

continuous in v. Furthermore, $J_i(v;a)$ is differentiable everywhere except possibly at the break points $V_{i-1}+\tau$ and $V_i+\tau$.

If we assume that the derivative $dJ_i(\xi_k;W_{k-1})/du$ exists (we will show this in what follows) then it is easy to see that it is given by

$$\frac{dJ_i(\xi_k(u);W_{k-1})}{du} = \begin{cases} -\left.\frac{\partial F(v;W_{k-1})}{dv}\right|_{v=\xi_k(u)-\tau-W_{k-1}} \frac{d\xi_k(u)}{du}, & \text{if } V_{i-1}+\tau < \xi_k(u) \le V_i+\tau, \\ 0, & \text{otherwise.} \end{cases} \tag{4.36}$$

Since $\partial F(v;a)/\partial v \ge 0$ (F is a distribution function and thus nondecreasing in v) and $d\xi_k(u)/du \le 0$ (u is a service rate and $\xi_k(u)$ is nonincreasing in u), it follows that $dJ_i(\xi_k;W_{k-1})/du \ge 0$. Thus we can apply Fubini's Theorem to get

$$\sum_{k=1}^{\infty} E\left[\frac{dJ_i(\xi_k(u);W_{k-1})}{du}\right] = E\left[\sum_{k=1}^{\infty} \frac{dJ_i(\xi_k(u);W_{k-1})}{du}\right]. \tag{4.37}$$

Recall that

$$E[R_i(u;Z)] = \sum_{k=1}^{\infty} E[J_i(\xi_k(u);W_{k-1})].$$

We want to compute $dE[R_i(u;Z)]/du$ for all $j \in \{1,\dots,p\}$. In section 4.5.4 we show that

$$\frac{dE[R_i(u;Z)]}{du} = \frac{d}{du}\sum_{k=1}^{\infty} E[J_i(\xi_k(u);a)] = \sum_{k=1}^{\infty} \frac{dE[J_i(\xi_k(u);W_{k-1})]}{du}. \tag{4.38}$$

In Section 4.5.3 we focus on $dE[J_i(\xi_k(u);W_{k-1})]/du$. In particular, we give conditions under which

$$dE[J_i(\xi_k(u);W_{k-1})]/du = E[dJ_i(\xi_k(u))/du]. \tag{4.39}$$

To show that (4.39) holds we use the generalized mean value theorem to get a bound needed to apply the dominated convergence theorem. To apply the generalized mean value theorem we must show that $J_i(v;a)$ is continuous and piecewise differentiable with an appropriately bounded derivative. We already showed that $J_i(v;a)$ is

a continuous function and differentiable almost everywhere. Hence, if $\xi_k(u)$ is continuous, then so is $J_i(\xi_k(u); W_{k-1})$, and if $\xi_k(u)$ is differentiable then we can use the chain rule to compute $dJ_i(\xi_k(u))/du$ when $dJ_i(v)/dv$ (with $v = \xi_k(u)$) and $d\xi_k(u)/du$ both exist. It turns out that $\xi_k(u)$ satisfies these conditions as summarized in the following theorem. The proof of the theorem follows closely a derivation in Glasserman (1991) and is rather involved. Therefore, we also state the theorem in Appendix A and include the proof there.

Theorem 4.3. *Let ξ_k be as defined in* (4.29)*. Then,*

1. *$\xi_k(u)$ is continuous in u on $(0, \infty)$ w.p.1.*
2. *$\xi_k(u)$ is differentiable in u w.p.1 at each $u \in (0, \infty)$.*
3. *$\xi_k(u)$ is piecewise differentiable in u over $(0, \infty)$ w.p.1.*

Notice the distinction between parts 2 and 3 of the theorem. Part 2 says that for a *fixed* $u \in (0, \infty)$ the set where $\xi_k(u)$ is not differentiable in u has probability zero. Part 3 says that the set of non-differentiable points in $(0, \infty)$ is countable on every sample path w.p.1.

4.5.3 Unbiasedness

In this section we use the results of the previous section to show that Equation (4.39) holds under some additional conditions on the arrival process. We start by proving three lemmas (the proof of Lemma 4.6 is given in Appendix A). The first two lemmas show that $J_i(\xi_k(u); W_{k-1})$ is piecewise differentiable in u. Recall that $J_i(v; a)$ is piecewise continuous in v. A potential difficulty in showing that this translates to piecewise continuity (that is needed for piecewise differentiability) of $J_i(\xi_k(u); W_{k-1})$ is that the value $\xi_k(u)$ could take a single value v over a set of u that is not countable. We show on the other hand that such a set is connected and then $J_i(\xi_k(u); W_{k-1})$ is constant on such a set and is therefore differentiable on that set.

Lemma 4.4. *Suppose that $g : \mathbb{R} \to \mathbb{R}$ is monotone. Then, for any $v \in \mathbb{R}$, the set $\{u : g(u) = v\}$ is either empty or connected.*

Proof: Suppose g is increasing (the case when g is decreasing is identical). Define $\underline{u} = \inf\{u : g(u) = v\}$ and $\bar{u} = \sup\{u : g(u) = v\}$. Then $\underline{u} \leq \bar{u}$, and since g is increasing, $g(u) = v$ for $u \in (\underline{u}, \bar{u})$. □

Lemma 4.5. *Suppose that $F(v;t)$ is continuous and piecewise differentiable in v for each $t \in [0, \infty)$. Then $J_i(\xi_k(u); W_{k-1})$ is continuous and piecewise differentiable w.p.1.*

Proof: Let $j \in \{1, \ldots, p\}$ be arbitrary. Recall that $\xi_k(u)$ is continuous and piecewise differentiable w.p.1 by Theorem 4.3. Hence, $J_i(\xi_k(u); W_{k-1})$ is continuous in u w.p.1 since $J_i(v; a)$ is continuous in v.

Now, $J_i(\xi_k(u); W_{k-1})$ is differentiable for all u such that $(dJ_i/dv)|_{v=\xi_k(u)}$ and $d\xi_k(u)/du$ exist. Let $\mathcal{M}_\xi$ be the random set where ξ_k is not differentiable. Then $\mathcal{M}_\xi$ is countable w.p.1. Since u is the rate at which work is completed in period j, ξ_k is decreasing in u. Since $F(v;t)$ is piecewise differentiable in v, so is $J_i(v; a)$. Let $\mathcal{U}$ be the countable set where J_i is not differentiable. By Lemma 4.4 there exists for each $v_i \in \mathcal{U}$ a connected set $\mathcal{M}_i$ such that $\xi_k(u) = v_i \ \forall \ u \in \mathcal{M}_i$. The function $J_i(\xi_k(u); W_{k-1})$ is constant in u on the interior of $\mathcal{M}_i$ and thus differentiable. Therefore, $(dJ_k(v)/dv)|_{v=\xi_k(u)}$ exists everywhere except possibly at the endpoints of the sets $\mathcal{M}_i$. Let $\mathcal{M}$ be the set of endpoints of the sets $\mathcal{M}_i$, $i = 1, 2, \ldots$. The set $\mathcal{M}$ is countable w.p.1. Furthermore, $J_i(\xi_k(u); W_{k-1})$ is differentiable everywhere except possibly for $u \in \mathcal{M}_\xi \cup \mathcal{M}$. But $\mathcal{M}_\xi \cup \mathcal{M}$ is countable w.p.1, and hence $J_i(\xi_k(u); W_{k-1})$ is piecewise differentiable w.p.1. □

Before we prove the main result of this section we define

$$\begin{aligned} u^* < \infty \ &: \quad \text{a constant,} \\ u_* > 0 \ &: \quad \text{a constant,} \\ \rho \ &= \ u^*/u_*, \text{ and} \\ \Theta \ &= \ [u_*, u^*]. \end{aligned}$$

The set Θ contains the points u where the IPA gradient estimator at u is unbiased. We can choose u_* and u^* as the lowest and highest service rates that we will encounter in the problem. The magnitude of ρ depends on the choice of these rates. Recall that p is the number of periods in the problem. We use these quantities in the next lemma to bound the derivative of the epoch when a server becomes available. The proof of the lemma is given in Appendix A.

Lemma 4.6. *Let ξ_k be as defined in* (4.29)*. Then*

$$\left|\frac{d\xi_k(u)}{du}\right| \leq \frac{\rho^p \zeta}{u_*}\xi_k(u)$$

where the derivative exists.

Theorem 4.7. *Consider the problem described in Section 4.5.1. Suppose that $F(v;t)$ is piecewise differentiable for $v \in [0,\infty)$ for each $t \in [0,\infty)$. Let $\mathcal{D}_t$ be the set where $F(v;t)$ is differentiable and suppose that $\sup_{t\in[0,\infty)} \sup_{v\in\mathcal{D}_t} dF(v)/dv \leq K_F < \infty$. Then*

$$dE[J_i(\xi_k(u); W_{k-1})]/du = E[dJ_i(\xi_k(u; W_{k-1}))/du]$$

on Θ.

Remark 4.1. In case the arrival process is a non-homogeneous Poisson process,

$$\sup_{t\in[0,\infty)} \sup_{v\in\mathcal{D}_t} dF(v)/dv$$

is simply the maximum instantaneous rate.

Proof: The function $J_i(\xi_k(u); W_{k-1})$ is continuous and piecewise differentiable in u w.p.1 by Lemma 4.5. Also, $J_i(v;a)$ is differentiable for all v except possibly for $v \in \mathcal{U}_{W_{k-1}} = ([0,\infty) \setminus \mathcal{D}_{W_{k-1}}) \cup \{V_{i-1}+\tau, V_i+\tau\}$. Then, by the generalized mean

value theorem (Theorem B.3), we get that whenever u and $u+h$ are in Θ

$$\begin{aligned}
&\left|\frac{J_i(\xi_k(u+h);W_{k-1}) - J_i(\xi_k(u);W_{k-1})}{h}\right| \\
&\leq \sup_{u\in\Theta, u\notin\mathcal{M}\cup\mathcal{M}_\xi}\left|\frac{dJ_i(\xi_k(u))}{du}\right| \\
&\leq \sup_{u\in\Theta, u\notin\mathcal{M}\cup\mathcal{M}_\xi}\left|\left(\frac{dJ_i(v;a)}{dv}\bigg|_{v=\xi_k(u)}\right)\frac{d\xi_k(u)}{\partial u}\right| \\
&\leq \left(\sup_{u\in\Theta, u\notin\mathcal{M}\cup\mathcal{M}_\xi}\left|\frac{d\xi_k(u))}{\partial u}\right|\right)\left(\sup_{v\in[0,\infty), v\notin\mathcal{U}_{W_{k-1}}}\left|\frac{dJ_i(v;a)}{dv}\right|\right) \\
&\leq \left(\frac{\rho^p q}{u_*}\xi_k(u)\right)K_F, \qquad (4.40)
\end{aligned}$$

where $\mathcal{M}$ and $\mathcal{M}_\xi$ are defined as in Lemma 4.5. Notice that $J_i(\xi_k(u);W_{k-1}) = 0$ whenever $W_{k-1} > V_i$, or when $\xi_k(u) > V_i + \tau$. Thus,

$$J_i(\xi_k(u);W_{k-1}) = \mathbf{1}\{W_{k-1}\leq V_i\}\mathbf{1}\{\xi_k(u)\leq V_i+\tau\}J_i(\xi_k(u);W_{k-1}).$$

Now, for small h and $\epsilon > 0$,

$$\begin{aligned}
&|J_i(\xi_k(u+h);W_{k-1}) - J_i(\xi_k(u);W_{k-1})| \\
&= \mathbf{1}\{W_{k-1}\leq V_i\}\,|\mathbf{1}\{\xi_k(u+h)\leq V_i+\tau\}J_i(\xi_k(u+h);W_{k-1}) \\
&\quad - \mathbf{1}\{\xi_k(u)\leq V_i+\tau\}J_i(\xi_k(u);W_{k-1})| \\
&= \mathbf{1}\{W_{k-1}\leq V_i\}\mathbf{1}\{\xi_k(\min\{u,u+h\})\leq V_i+\tau\}\,|J_i(\xi_k(u+h);W_{k-1}) \\
&\quad - J_i(\xi_k(u);W_{k-1})| \qquad (4.41) \\
&= \mathbf{1}\{W_{k-1}\leq V_i\}\mathbf{1}\{\xi_k(u)\leq V_i+\tau+\epsilon\}|J_i(\xi_k(u+h);W_{k-1}) - J_i(\xi_k(u);W_{k-1})|, \\
&\qquad (4.42)
\end{aligned}$$

where all equalities hold w.p.1. Equation (4.41) follows since ξ_k is decreasing in u,

and (4.42) follows by the continuity of ξ_k. Combining this with (4.40) we get

$$\begin{aligned}
&\left|\frac{J_i(\xi_k(u+h);W_{k-1}) - J_i(\xi_k(u);W_{k-1})}{h}\right| \\
&\leq \mathbf{1}\{W_{k-1} \leq V_i\}\mathbf{1}\{\xi_k(u) \leq V_i + \tau + \epsilon\}\frac{\rho^p q K_F}{u_*}\xi_k(u) \\
&\leq \mathbf{1}\{W_{k-1} \leq V_i\}\frac{\rho^p q K_F}{u_*}(V_i + \tau + \epsilon) \qquad (4.43) \\
&< \infty.
\end{aligned}$$

Hence, we can apply the dominated convergence theorem (Theorem B.2) to conclude that

$$\begin{aligned}
\frac{dE[J_i(\xi_k(u);W_{k-1})]}{du} &= \lim_{h\to 0} E\left[\frac{J_i(\xi_k(u+h;W_{k-1})) - J_i(\xi_k(u;W_{k-1}))}{h}\right] \\
&= E\left[\lim_{h\to 0}\frac{J_i(\xi_k(u+h);W_{k-1}) - J_i(\xi_k(u);W_{k-1})}{h}\right] \\
&= E\left[\frac{dJ_i(\xi_k(u);W_{k-1})}{du}\right].
\end{aligned}$$

□

4.5.4 Interchanging Differentiation and Infinite Sum

At the end of section 4.5.2 we stated that (see Equation (4.38))

$$\begin{aligned}
&\frac{d}{du}\sum_{k=1}^{\infty} E[J_i(\xi_k(u);W_{k-1})] \\
&= \lim_{h\to 0}\sum_{k=1}^{\infty} E\left[\frac{J_i(\xi_k(u+h);W_{k-1}) - J_i(\xi_k(u);W_{k-1})}{h}\right] \\
&= \sum_{k=1}^{\infty}\lim_{h\to 0} E\left[\frac{J_i(\xi_k(u+h);W_{k-1}) - J_i(\xi_k(u);W_{k-1})}{h}\right] \\
&= \sum_{k=1}^{\infty}\frac{dE[J_i(\xi_k(u);W_{k-1})]}{du}.
\end{aligned}$$

The first and last equalities follow by definition and the following lemma justifies the second inequality of interchanging the limit and the sum operator.

Lemma 4.8. *Suppose that the conditions of Theorem 4.7 are in effect. Then*

$$\lim_{h\to 0}\sum_{k=1}^{\infty} E\left[\frac{J_i(\xi_k(u+h);W_{k-1})-J_i(\xi_k(u);W_{k-1})}{h}\right] = \sum_{k=1}^{\infty}\lim_{h\to 0} E\left[\frac{J_i(\xi_k(u+h);W_{k-1})-J_i(\xi_k(u);W_{k-1})}{h}\right].$$

Proof: At each u, $\xi_k(u)$ is differentiable w.p.1, and so is $J_i(\xi_k(u);W_{k-1})$. Therefore,

$$\lim_{h\to 0} E\left[\frac{J_i(\xi_k(u+h);W_{k-1})-J_i(\xi_k(u);W_{k-1})}{h}\right] = \frac{dE[J_i(\xi_k(u);W_{k-1})]}{du}.$$

Also,

$$\begin{aligned}
&\left|E\left[\frac{J_i(\xi_k(u+h);W_{k-1})-J_i(\xi_k(u);W_{k-1})}{h}\right]\right| \\
&\le E\left[\left|\frac{J_i(\xi_k(u+h);W_{k-1})-J_i(\xi_k(u);W_{k-1})}{h}\right|\right] \qquad (4.44)\\
&\le E\left[\mathbf{1}\{W_{k-1}\le V_i\}\frac{\rho^p q K_F}{u_*}(V_i+\tau+\epsilon)\right] \qquad (4.45)\\
&= \frac{\rho^p q K_F}{u_*}(V_i+\tau+\epsilon)E[\mathbf{1}\{W_{k-1}\le V_i\}].
\end{aligned}$$

Equation (4.44) follows by Jensen's inequality and (4.45) follows from (4.43). Since $\mathbf{1}\{W_{k-1}\le V_i\}$ is nonnegative, we can apply Fubini's theorem to get

$$\sum_{k=1}^{\infty} E[\mathbf{1}\{W_{k-1}\le V_i\}] = E\left[\sum_{k=1}^{\infty}\mathbf{1}\{W_{k-1}\le V_i\}\right] = E[C+1],$$

where C is the number of calls before time V_i. But $E[C]$ is finite by assumption. Thus, we can apply the dominated convergence theorem (Theorem B.2) to get the result. □

4.5.5 An Unbiased IPA Gradient Estimator

Here we summarize the results from the previous sections.

Theorem 4.9. *Suppose that the conditions of Theorem 4.7 are in effect. Then*

$$\frac{dE[R_i(u;Z)]}{du} = E\left[\sum_{k=1}^{C+1} \frac{dJ_i(\xi_k(u);W_{k-1})}{du}\right]. \tag{4.46}$$

Proof:

$$\begin{aligned}
\frac{dE[R_i(u;Z)]}{du} &= \frac{d}{du}E\left[\sum_{k=1}^{\infty} L_k^i(u;Z)\right] \\
&= \frac{d}{du}\sum_{k=1}^{\infty} E[L_k^i(u;Z)] && (4.47)\\
&= \frac{d}{du}\sum_{k=1}^{\infty} E[J_i(\xi_k(u);W_{k-1})] && (4.48)\\
&= \sum_{k=1}^{\infty} \frac{d}{du}E[J_i(\xi_k(u);W_{k-1})] && (4.49)\\
&= \sum_{k=1}^{\infty} E\left[\frac{dJ_i(\xi_k(u);W_{k-1})}{du}\right] && (4.50)\\
&= E\left[\sum_{k=1}^{\infty} \frac{dJ_i(\xi_k(u);W_{k-1})}{du}\right] && (4.51)\\
&= E\left[\sum_{k=1}^{C+1} \frac{dJ_i(\xi_k(u);W_{k-1})}{du}\right]. && (4.52)
\end{aligned}$$

Equation (4.47) follows from Fubini's Theorem since L_k^i is nonnegative. Equation (4.48) follows by the smoothing argument in Section 4.5.2. Equation (4.49) follows by Lemma 4.8. Equation (4.50) follows by Theorem 4.7. Equation (4.51) follows by (4.37). Finally, Equation (4.52) follows since the derivative of J_i is 0 for $k > C+1$ because W_{k-1} occurs after the end of period i when $k > C+1$. □

We can use simulation to estimate the right hand side of Equation (4.9). Suppose that we run n simulations and let C_d be the number of calls received in simulation d. Then an unbiased estimator for (4.46) is

$$\frac{1}{n}\sum_{d=1}^{n}\sum_{k=1}^{C_d+1} \frac{\partial J_i(\xi_k^d(\mu);W_{k-1}^d)}{\partial \mu_j},$$

where W_{k-1}^d and ξ_k^d are realizations of the random variables W_{k-1} and ξ_k. The partial derivative $\partial J_i(\xi_k^d(\mu); W_{k-1}^d)/\partial\mu_j$ is given by Equation (4.36). In Appendix A we give a pseudo-code for computing $\xi_k^d(\mu)$ and $\partial\xi_k^d(\mu)/\partial\mu_j$ via simulation.

4.5.6 An IPA Gradient Estimator for a Varying Number of Servers

We have derived an unbiased estimator for the gradient of the number of calls answered on time in a period with respect to the service rates in all periods. A major assumption we made was that the number of servers in all periods had to be the same. This assumption will not be satisfied in practice, so to approximate the real situation we adjust the service rates in all periods so that the total service capacity (service rate $\times$ the number of servers) in each period is the same in both situations.

Still, the modeling error that results from this can be significant. An alternative is to compute an IPA gradient estimator for the "correct" model, even though we cannot guarantee that such an estimator is unbiased. The estimator is similar to the IPA estimator that we derived previously.

We use the same notation as before, except that now we denote the number of servers in period i by y_i as opposed to ζ in all periods before. We again use $\xi_k(\mu)$ as the time epoch when a server becomes available to serve call k, but now it depends on the number of servers so we write $\xi_k(\mu; y)$. We can use the same arguments as in Section 4.5.2 to smooth the service level function $E[L_k(\mu)]$. The only difference is in Equation (4.33), because now $\xi_k(\mu; y)$ depends on which period W_k is in and thus on the value of U_k which is not measurable $\mathcal{Z}_{k-1}$. That is not a concern, however, since W_k occurs in period i under the condition $V_{i-1} < W_k \leq V_i$, and in that case we can determine $\xi_k(\mu; y)$ from $\mathcal{Z}_{k-1}$.

Unfortunately, $\xi_k(\mu; y)$ is not continuous in μ in general. It is possible that a change in μ could result in a call that previously ended in period l say, does not end until period $l+1$. If there are fewer servers in period $l+1$ and if all servers except

one, for example, were busy at the end of period l then a server would have been available to answer a call in period l but a server will now not be available until one or more calls have completed service in period $l+1$ or later.

When there is only an infinitesimal change (increase) in the service rate we can see that the above situation only occurs when the service completion occurs at the same moment as the end of period event. The probability of that event is zero, and therefore, we can argue that for a given μ, $\xi_k(\mu; y)$ and hence $J_i(\xi_k(\mu; y))$ are indeed differentiable on every sample path w.p.1. The derivative estimate will, however, generally be biased because of the failure to take into account the effect of call completions switching between periods.

We let the estimator be

$$\frac{\partial E[R_i(\mu; Z)]}{\partial \mu_j} \approx E\left[\sum_{k=1}^{C+1} \partial J_i(\xi_k(\mu; y); W_{k-1})\partial\mu\right]. \tag{4.53}$$

The gradient estimator (4.53) does not take into account the effect on the sampled service level function that occurs when a change in the service rate causes a service completion to move between periods. It may be possible to do some additional smoothing in an attempt to establish a function that is continuous in the service rates on every sample path w.p.1 and allow the number of servers to vary between time periods. That requires, however, significant additional computational work and we will therefore not derive such an estimator in detail; see Fu and Hu (1997, p. 135) for more on this approach.

4.6 Numerical Results

In the previous sections we developed three different methods and four different estimators for approximating subgradients of a discrete service level function via simulation. We mentioned some of the advantages and disadvantages of each method. In this section we present a small numerical example in order to cast light on the prac-

tical performance of each method. In particular we will discuss the computational effort of each method, the variance of the subgradient estimators and the validity of the estimators as a subgradient of the SAA of the service level function.

We consider an $M(t)/M/s(t)$ queue with $p = 2$ periods of equal length of 30 minutes. The service rate is $\mu = 4$ calls/hour. The arrival process is a nonhomogeneous Poisson process with the arrival rate a function of the time t equal to $\lambda(t) = \lambda(1 - |t/60\text{minutes} - .65|)$. We set $\lambda = 120$ calls/hour, which makes the average arrival rate over the 2 periods equal to 87.3 calls/hour. We say that a call begins service on time if it enters service less than 90 seconds after it arrives.

Our reasoning for presenting such a simple example, rather than a more realistic model of a call center, is that this example captures every complexity of the problem, it is easy to verify the properties of the subgradient approximation, and a complete visualization of the service level function is possible.

We computed the average number of calls received in each period answered in less than 90 seconds after they arrive. We did this for the number of servers in period 1 ranging from 10 to 30 and the number of servers in period 2 ranging from 22 to 40. Our sample size was 999. We also computed at each point an approximation of a subgradient from each of the FD, LR and IPA methods. The staffing level in period 2 does not have a great effect on the service level in period 1 so in the remainder of the discussion we focus on the service level in period 2 as a function of the staffing levels in periods 1 and 2.

Figure 4.5 shows the *number* of calls received in period 2 that are answered on time as a function of the staffing levels in periods 1 and 2. Figures 4.6-4.9 show a contour plot of the same function (curved lines going across) and the subgradient approximation at each point (arrows). The arrows originate at their corresponding point and show both the magnitude and the direction of the subgradient. Ideally, the arrows should be orthogonal to the contours, since the function values do not change

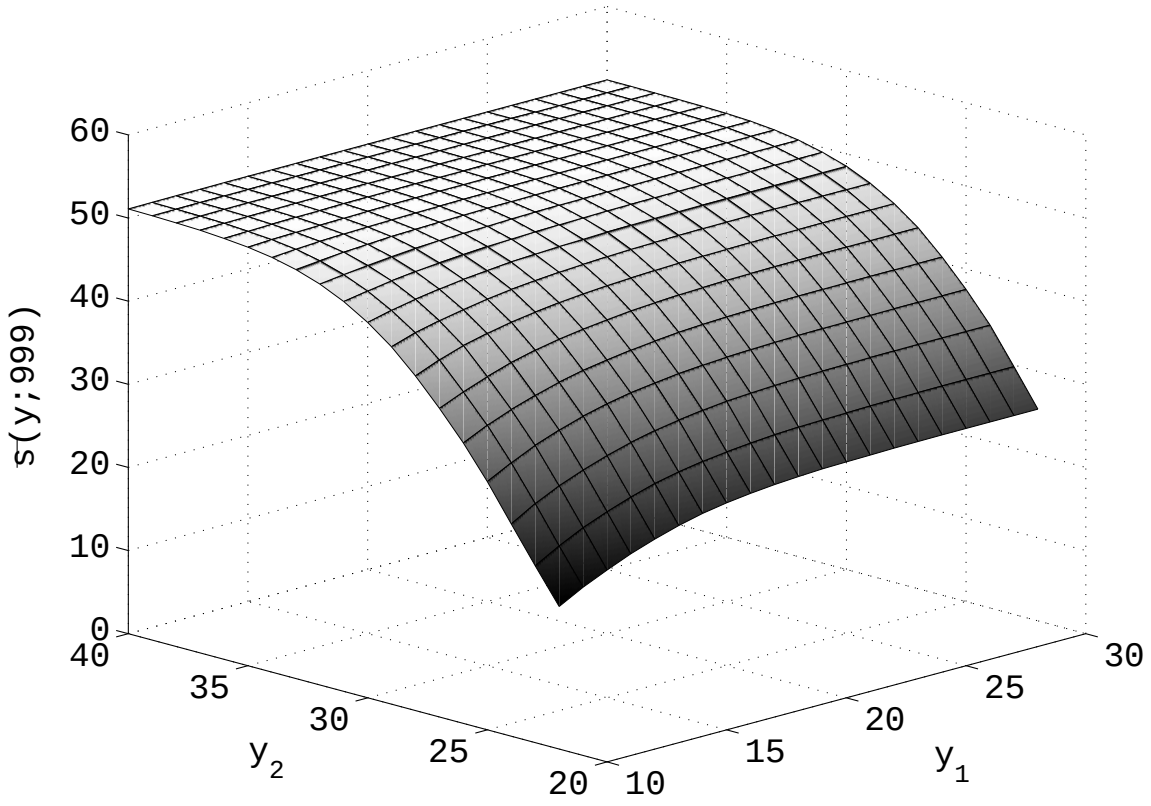


Figure 4.5: Sample average approximation (sample size 999) of the number of calls that are answered on time.

along the contours. The contours should also decrease in length as the distance between the contours decreases, since a longer arrow represents a larger magnitude subgradient, which in turn should translate into a significant change in function value in the direction of the subgradient. We also display 95% confidence regions as ellipses for selected points, except that we do not compute the variance for the IPA gradient estimator using a varying number of servers. The ellipses are centered at the endpoint of the corresponding arrow. The confidence regions for the FD and IPA methods are so small that the ellipses are barely visible in these plots.

From Figure 4.6 we see that the FD method appears to give a good approximation of the subgradients over the whole domain, so if the computational work of running $p+1$ simulations to get a subgradient at a single point is not overwhelming, then the

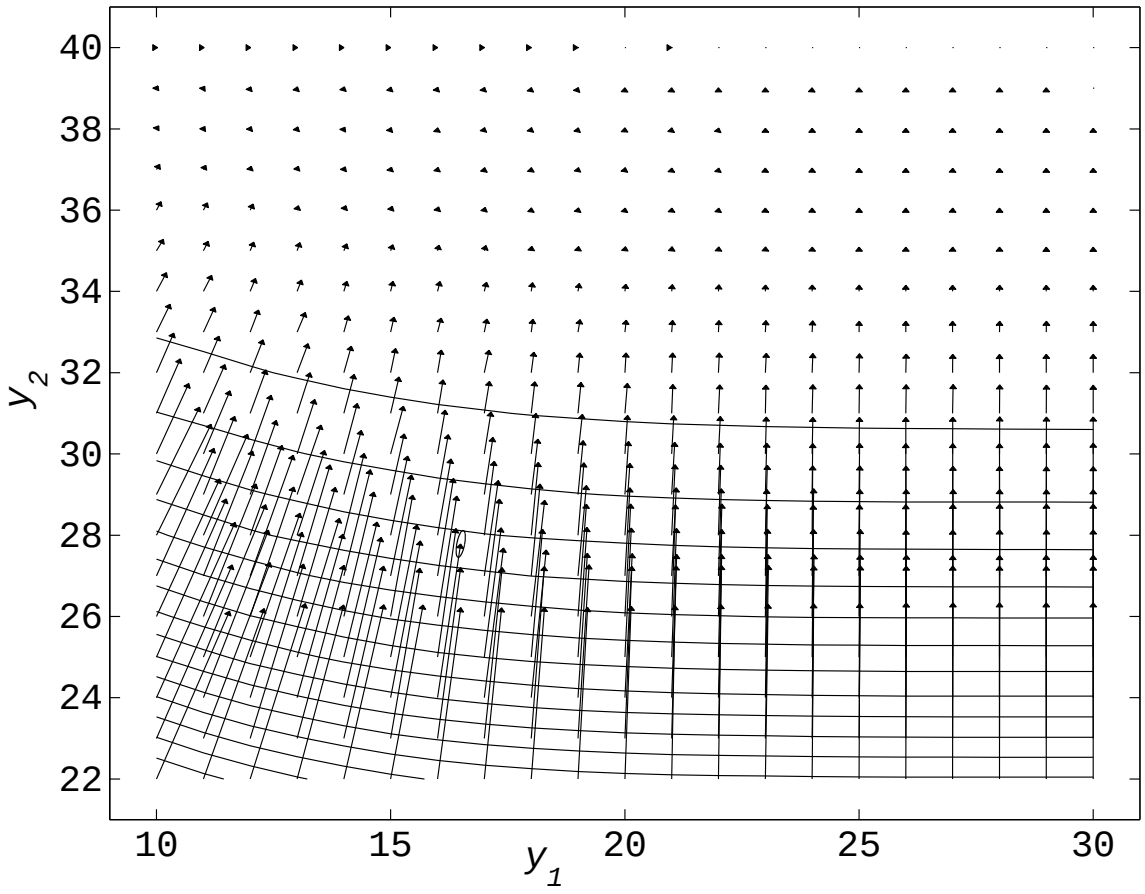


Figure 4.6: Subgradient estimates via the finite difference method.

FD method would certainly be a good candidate.

For exponential service times the likelihood ratio estimator (4.19) for the partial derivative of the number of calls answered on time in period i w.r.t. the service rate in period j simplifies to

$$\left(\frac{\partial r_i(\mu; y)}{\partial \mu_j}\right)_{\mu=\mu^*} = E\left[R_i(\mu^*; y) \sum_{k=1}^{C} \mathbf{1}\{\pi(B_k) = j\} \left(\frac{1}{\mu_j^*} - X_k\right)\right], \tag{4.54}$$

where X_k is the service time of the kth call. In this example $\mu_1^* = \mu_2^* = 15$ minutes. We see from (4.54) that the LR method requires only the additional work of summing up the terms in (4.54) and multiplying by the respective service level function. On the other hand, we see that the confidence regions are much larger for the LR method than the other two methods. High variance of the LR method has also been observed

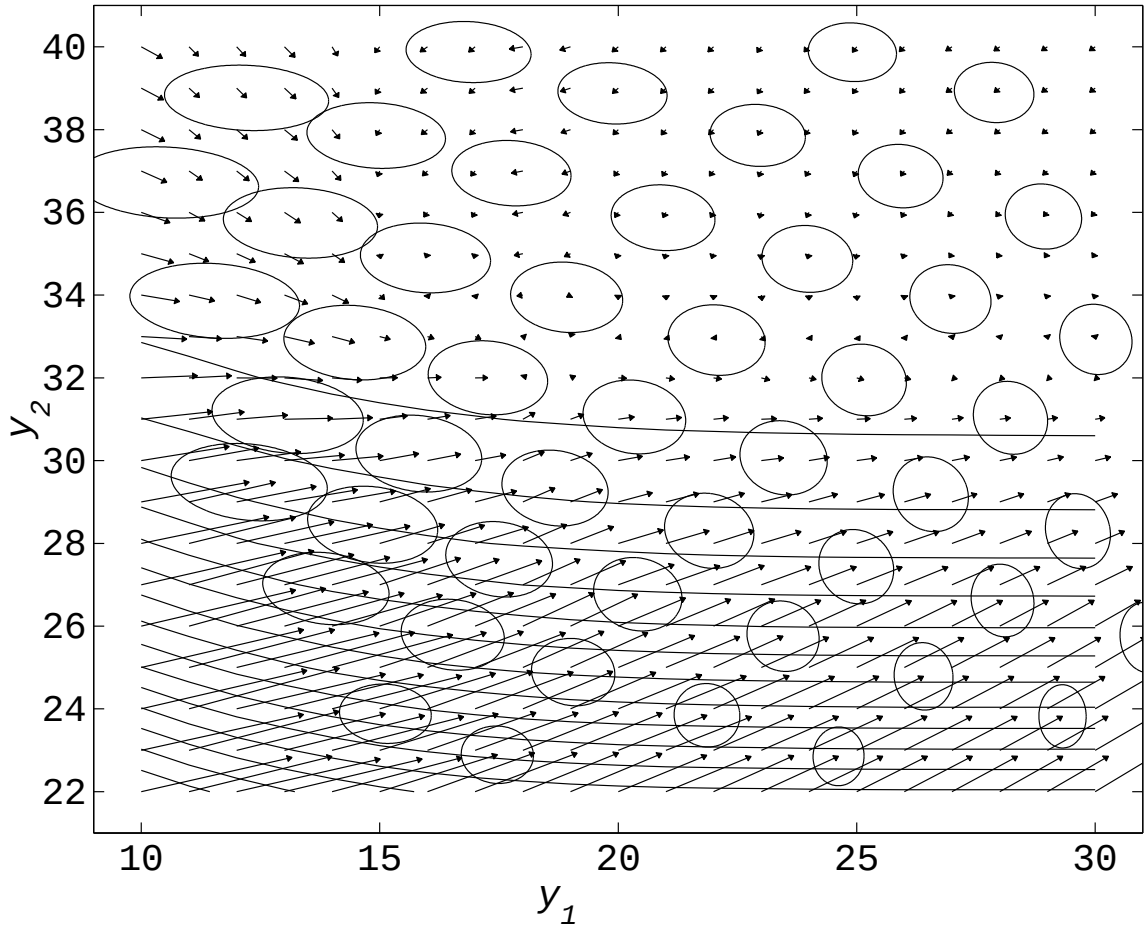


Figure 4.7: Subgradient estimates via the likelihood ratio method.

by several others in the literature (e.g. Fu, 2002).

A second observation from Figure 4.7 is that in the lower right corner, the LR gradients suggest that the service level in period 2 will improve significantly if servers are added in period 1 even when there are plenty of servers in period 1. To see why note that even if only a portion of the servers are busy at the end of period 1, increasing the service rate will reduce their residual service times. The service times depend only on when the call begins service in the LR estimator, so the state of the servers in period 2 will be greatly impacted by a change in service rate in period 1, since calls that begin service late in period 1 will still be in service during at least a portion of period 2. A possible remedy would be to modify the LR to take into

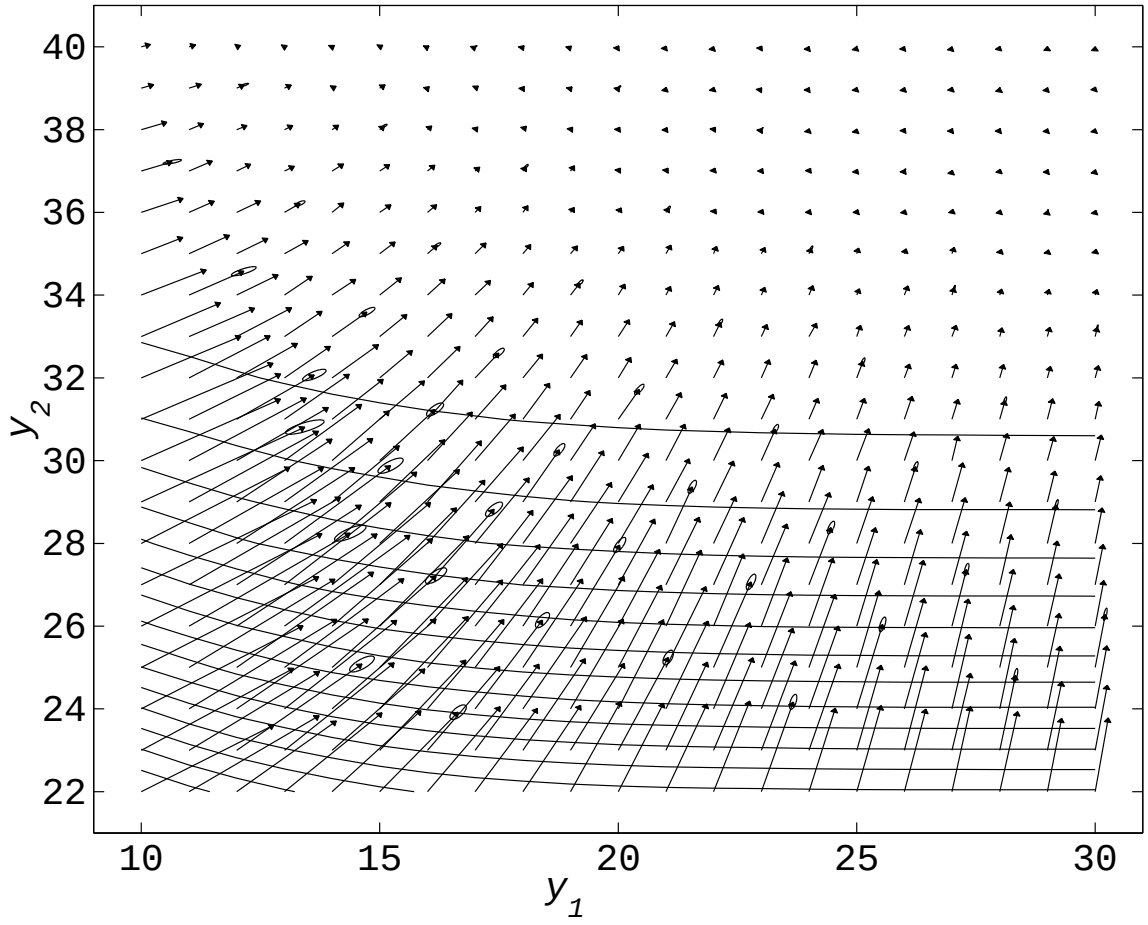


Figure 4.8: Subgradient estimates via IPA using a fixed number of servers.

account how much time a call actually spends in each period.

Our last comment on the LR method is that some of the subgradient estimates have a negative component, which contradicts a nondecreasing service level function. This is due to the large variance of the LR method, and we see that many of the confidence regions contain the origin of the corresponding arrow, meaning that the subgradient is not statistically different from zero.

The variance of the unbiased IPA gradient estimator using a fixed number of servers (Figure 4.8) is much lower than the variance of the LR estimator. The computational effort is only slightly greater for IPA than LR. It can be seen, however, that the estimates differ from the FD estimates especially for low staffing levels in

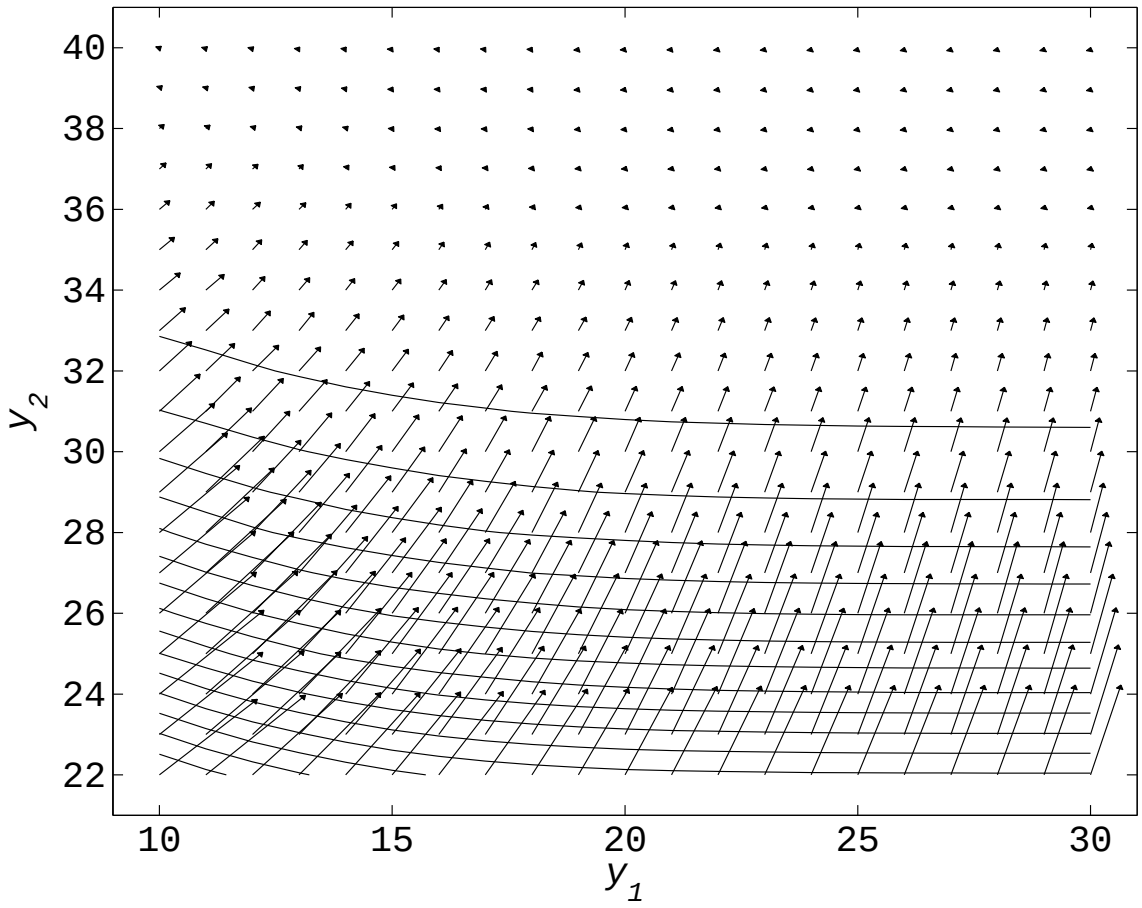


Figure 4.9: Subgradient estimates via IPA using a varying number of servers.

period 1. That is because to compute the IPA estimators we fix the number of servers at the maximum of the number of servers in period 1 and 2 and adjust the service rates in each period accordingly.

Finally, we compare the results of the two IPA gradient estimators in Figure 4.8 and Figure 4.9. It appears the estimator using a varying number of servers does better when there are more servers in period 2 than in period 1, and worse when the number of servers decreases between period 1 and period 2. That is consistent with the fact the estimator using varying number fails to account for the potential discontinuity on the sample path when the number of servers decreases between periods. There is no discontinuity on the sample paths when the number of servers in period 2 is higher than in period 1.

We have proposed three different approaches to a difficult problem. The FD approach seems to be the most reliable while at the same time being the most computationally expensive. The likelihood ratio method suffers from a high variance, which makes it the least attractive of the three methods. IPA does not give as good gradient estimates as the FD method, but the computational requirements of IPA are obviously much lower.

CHAPTER V

PSEUDOCONCAVE SERVICE LEVEL FUNCTIONS AND AN ANALYTIC CENTER CUTTING PLANE METHOD

5.1 Introduction

Recall from Chapter II that we are trying to find an optimal solution of the call center staffing problem

$$\begin{array}{lll} \min & f(y) & \\ \text{s.t.} & g_i(y) & \geq 0 \text{ for } i \in \{1, \ldots, p\} \\ & y & \geq 0 \text{ and integer,} \end{array} \tag{5.1}$$

by solving its sample average approximation

$$\begin{array}{lll} \min & f(y) & \\ \text{s.t.} & \bar{g}_i(y; n) & \geq 0 \text{ for } i \in \{1, \ldots, p\} \\ & y & \geq 0 \text{ and integer,} \end{array} \tag{5.2}$$

where $y \in \mathbb{R}^p$, $\bar{g}_i(y; n)$ is the sample average approximation of the service level function $g_i(y)$ for $i \in \{1, \ldots, p\}$, and

$$\begin{array}{lllll} f(y) = & \min & c^T x & & \\ & \text{s.t.} & Ax & \geq y & \\ & & x & \geq 0 \text{ and integer.} & \end{array} \tag{5.3}$$

In Section 2.3 we showed that optimal solutions of (5.2) are optimal for (5.1) for a large enough sample size n and that the probability of this event occurring goes to 1 at an exponential rate when the sample size increases. The problem is then to solve

the sample average approximation problem (5.2).

In Chapter III we presented the simulation-based Kelley's cutting plane method (SKCPM) to solve (5.2). The approach relies on the assumption that service in a period is concave and componentwise-increasing as a function of the staffing level vector. The empirical results in Section 3.5 suggest that this assumption is reasonable, at least for sufficiently high staffing levels. But for low staffing levels, the empirical results suggest that performance is increasing and *convex* in the staffing level. So performance appears to follow an "S-shaped" curve (see also Ingolfsson et al., 2003) in one dimension.

This non-concavity can cause the SKCPM to cut off feasible solutions, and the problem can be so severe as to lead to the algorithm suggesting impractical staffing plans. Nevertheless, the ability of the SKCPM to efficiently sift through the combinatorially huge number of potential staffing plans is appealing. One might ask whether there is a similar optimization approach that can efficiently search through alternative staffing plans while satisfactorily dealing with S-shaped curves and their multidimensional extensions. That is the subject of this chapter.

In this chapter we combine a relaxation on the assumption of concavity, i.e., pseudoconcavity, and additional techniques to handle multi-dimensional extensions of S-shaped curves alluded to above, seen in examples like that depicted in Figure 5.2; see Section 5.3.1 for more details.

Assuming that the service level functions are pseudoconcave, one can then develop an analytic center cutting plane algorithm that efficiently searches the combinatorially large space of potential staffing plans to solve (5.2). In essence the algorithm works as follows. One begins with a polyhedron that contains an optimal solution to the staffing problem. At each stage, the algorithm selects a staffing plan that lies close to the analytic center of the polyhedron and runs a simulation at that point to determine service level performance. Depending on the results of the simulation an "optimality

cut" or one or more feasibility cuts are added, thereby shrinking the polyhedron. The algorithm terminates when the polyhedron contains no integer points, or when the difference between upper and lower bounds on the optimal objective is sufficiently small.

The remainder of this chapter is organized as follows. In Section 5.2 we review cutting plane methods for continuous problems with pseudoconcave constraints. Section 5.3 describes modifications for discrete problems, such as the call center staffing problem, and proves that the algorithm converges. In Section 5.4 we give an overview of analytical queuing staffing heuristics that we use as benchmarks for our method. In Section 5.5 we consider two call center examples. The first example highlights the properties and implementation issues of our method. In the second example we compare our method to the analytical queuing heuristics. The results show that our simulation-based analytic center cutting plane method, when used in conjunction with the finite difference method for estimating gradients, does better in most cases than the analytical queuing methods.

5.2 The Analytic Center Cutting Plane Method

In this section we state a version of the traditional analytic center cutting plane method (ACCPM) to fix ideas and provide a departure point for a reader unfamiliar with cutting plane methods in continuous optimization.

There are many cutting plane methods for solving convex optimization problems, including what may be termed boundary methods, such as Kelley's algorithm (Kelley, Jr., 1960) (that was discussed in more detail in Chapter III) and its extensions (e.g., Westerlund and Pörn, 2002), and interior point methods, recently reviewed by Mitchell (2003). In this chapter, we will focus our attention on one of the latter, namely the ACCPM, which was first implemented in duMerle (1995), as pointed out in Elhedhli and Goffin (2003). The ACCPM has proven effective in terms of both the-

oretical complexity (Atkinson and Vaidya, 1995; Nesterov, 1995; Goffin et al., 1996; Mitchell, 2003) and practical performance on a variety of problems (Bahn et al., 1995; Mitchell, 2000, and other references in Mitchell, 2003). Software packages implementing the method are available (e.g., Peton and Vial, 2001).

Many versions of the ACCPM for convex feasibility and optimization problems have been explored in the literature. The description we chose below borrows from Nesterov (1995), duMerle et al. (1998), and Mitchell (2003), as just some of the possible references.

Consider an optimization problem in the following general form:

$$\begin{aligned} \min \quad & b^T y \\ \text{s.t.} \quad & y \in Y, \end{aligned} \tag{5.4}$$

where $Y \subset \mathbb{R}^n$ is a convex set, and $b \in \mathbb{R}^n$. (A problem of minimizing a general convex function over a convex set can be easily represented in this form.) To simplify the presentation, assume that the set Y is bounded and has a non-empty interior.

To apply the ACCPM (or any other cutting plane algorithm), the feasible region Y needs to be described by a separation oracle. Such an oracle will, given an input $\hat{y} \in \mathbb{R}^n$, either correctly assert that $\hat{y} \in Y$, or otherwise return a non-zero vector $q \in \mathbb{R}^n$ such that

$$q^T(y - \hat{y}) \geq 0 \ \forall y \in Y,$$

i.e., produce a *feasibility cut*. (Depending on how the set Y is described, the oracle might produce a deep or a shallow cut, which have the same form as the constraint above, but a nonzero right hand side.)

We now describe a typical iteration of the ACCPM. At the beginning of an iteration, we have available a finite upper bound z on the optimal value of (5.4), and a polyhedron $P = \{y \in \mathbb{R}^n : Dy \geq d\}$ that is known to contain the set Y. Here $D \in \mathbb{R}^{r \times n}$ and $d \in \mathbb{R}^r$ for some finite r. We first compute the (weighted) analytic

center of the set $P \cap \{y \in \mathbb{R}^n : b^T y < z\}$ (for ease of presentation we assume that the set $P \cap \{y \in \mathbb{R}^n : b^T y < z\}$ is bounded), defined as the solution of the convex optimization problem

$$\min_y \{-w \log(z - b^T y) - \sum_{j=1}^{r} \log(D_{j\cdot} y - d_j)\}, \tag{5.5}$$

where $D_{j\cdot}$ is the jth row of D and $w > 0$ is a weight constant that affects the convergence rate of the algorithm (see, for example, duMerle et al., 1998). The set $P \cap \{y \in \mathbb{R}^n : b^T y < z\}$ is often referred to as the *localization set*, since it contains all feasible solutions with objective function values lower than z.

Finding a solution to (5.5) with high degree of precision is a relatively simple task from a practical standpoint and can be done, e.g., via Newton's method. Let $\hat{y}$ be the analytic center found. Next, the oracle is called with $\hat{y}$ as the input. If $\hat{y} \in Y$, then, by construction, $b^T \hat{y} < z$, and the upper bound is lowered by taking $z := b^T \hat{y}$. Otherwise, if $\hat{y} \notin Y$, the oracle will produce a vector q providing a feasibility cut, which is then added to the description of the polyhedron P. The procedure is then repeated. A slightly more detailed description of the algorithm is presented in Figure 5.1.

Intuitively, the algorithm's efficiency stems from the fact that at each iteration the cut being added passes through the analytic center of the localization set, which is often located near a geometric center. Thus, the volume of the localization set reduces rapidly with each iteration.

Suppose the set Y is specified by $Y = \{y \in \mathbb{R}^n : g_i(y) \geq 0,\ i \in \{1, \ldots, p\}\}$, where the functions $g_i(y),\ i \in \{1, \ldots, p\}$ are pseudoconcave, as defined below:

Definition 5.1. (Cf. Definition 3.5.10 in Bazaraa et al., 1993) Let $g : S \to \mathbb{R}$ be differentiable on S, where S is a nonempty open set in $\mathbb{R}^n$. The function g is said to be *pseudoconcave* if for any $\hat{y}, y \in S$ with $\nabla g(\hat{y})^T (y - \hat{y}) \leq 0$ we have $g(y) \leq g(\hat{y})$. Equivalently, if $g(y) > g(\hat{y})$, then $\nabla g(\hat{y})^T (y - \hat{y}) > 0$.

With Y in the above form, the feasibility cut at point $\hat{y} \notin Y$ which violates the

Initialization Start with a polyhedron $P^0 := \{y \in \mathbb{R}^n : D^0 y \geq d^0\}$ such that $Y \subset P^0$, and an upper bound z^0. Let $w^0 > 0$, and set the iteration counter $k := 0$.

Step 1 If termination criterion is satisfied, then stop, and return y^* as a solution. Otherwise, solve problem (5.5) with $w = w^k$, $z = z^k$, and $P = P^k := \{y \in \mathbb{R}^n : D^k \geq d^k\}$; let y^k be the solution.

Step 2a If $y^k \in Y$ then let $z^{k+1} := b^T y^k$, $y^* := y^k$, $D^{k+1} := D^k$ and $d^{k+1} := d^k$.

Step 2b If $y^k \notin Y$ then generate one or more feasibility cuts at y^k. Update D^k and d^k to include the new constraints, and let D^{k+1} and d^{k+1} represent the new constraint set. Let $z^{k+1} := z^k$.

Step 3 Compute w^{k+1} and let $k := k + 1$. Go to Step 1.

Figure 5.1: The analytic center cutting plane method (ACCPM).

ith constraint can be specified as

$$\nabla g_i(\hat{y})^T (y - \hat{y}) \geq 0, \tag{5.6}$$

since any solution y that satisfies $g_i(y) \geq 0$ also satisfies $g_i(y) > g_i(\hat{y})$.

5.3 A Cutting Plane Method for Discrete Problems

In this section we describe how the ACCPM algorithm of Section 5.2 can be modified to solve the sample average approximation of the call center staffing problem (5.2).

The most significant modification to the ACCPM for continuous problems is needed to take into account the fact that the feasible region of (5.2) is no longer a convex, or even connected, set, due to the integrality restriction on the variables. Cutting plane algorithms for nonlinear mixed integer programs have been explored in the past (see, for example, Westerlund and Pörn, 2002). However, in this and other similar papers it is assumed that the constraint functions are in fact differen-

tiable functions of continuous variables; the integrality restrictions on the variables are, in a sense, exogenous. In such a setting the concept of a convex (continuous) nonlinear relaxation of the integer program is straightforward, and feasibility cuts are generated simply using the gradients of these continuous functions. In our setting, however, the service level functions and their sample average approximations are not defined at non-integer values of y, and devising their continuous extension, especially one that is easy to work with from the algorithmic point of view, is non-trivial at best. Therefore, we take a different approach in adapting the ACCPM to the discrete case.

As far as we know, the use of ACCPM as a solution method for nonlinear integer programs has not been reported, although the method has been successfully used to solve the linear relaxation subproblems in branching algorithms for integer programs (see, for example, Mitchell (2000) and Elhedhli and Goffin (2003), among many others).

In Section 5.3.1 we extend the notion of pseudoconcavity to functions of integer variables, and show how feasibility cuts can be generated assuming that the functions $\bar{g}_i(y; n),\ i \in \{1, \ldots, p\}$ are, in fact, discrete pseudoconcave. This leads to an ACCPM method for (mixed) integer programming problems satisfying the pseudoconcavity assumption; the algorithm is applicable to the types of problems considered in Westerlund and Pörn (2002), for example.

We also discuss whether the S-shaped form of the service level functions in the call center staffing problem is consistent with this assumption, and propose alternative feasibility cuts at points where it is violated.

The following Section 5.3.2 discusses other modifications of the original algorithm for the problem (5.2) and details of our implementation. Section 5.3.3 gives a proof of convergence.

5.3.1 Discrete Pseudoconcave Functions

We begin by defining the notions of a discrete convex set and a discrete pseudoconcave function. We denote the convex hull of the set C by $\text{conv}(C)$.

Definition 5.2. We say that the set $C \subseteq \mathbb{Z}^n$ is a *discrete convex set* if $\text{conv}(C) \cap \mathbb{Z}^n = C$, i.e, the set C equals the set of integer points in $\text{conv}(C)$.

Definition 5.3. Let $C \subseteq \mathbb{Z}^n$ be a discrete convex set and $g : C \to \mathbb{R}$. Then g is *discrete pseudoconcave* if for any $\hat{y} \in C$ there exists a vector $q(\hat{y}) \in \mathbb{R}^n$ such that for any $y \in C$,

$$q(\hat{y})^T(y - \hat{y}) \leq 0 \Rightarrow g(y) \leq g(\hat{y}).$$

Equivalently, if $g(y) > g(\hat{y})$, then $q(\hat{y})^T(y - \hat{y}) > 0$. We call the vector $q(\hat{y})$ a *pseudogradient* of g at $\hat{y}$.

In the continuous case, pseudoconcavity is a weaker property than concavity of a function; discrete pseudoconcavity can be viewed as a relaxation of the concave extensible function property defined in Murota (2003, p. 93).

If the functions $\bar{g}_i(y; n)$ in (5.2) are discrete pseudoconcave, then a feasibility cut at an integer point $\hat{y}$ that violates the ith constraint can be obtained in the form

$$\bar{q}_i(\hat{y}; n)^T(y - \hat{y}) \geq \epsilon,$$

where $\bar{q}_i(\hat{y}; n)$ is the pseudogradient of $\bar{g}_i(\cdot; n)$ at $\hat{y}$, and $\epsilon > 0$ is sufficiently small.

Are the service level functions indeed discrete pseudoconcave? To provide an illustrative example, we computed the sample average of a service level function in period 2 of a simple two period model of a call center. Figure 5.2 (a) shows the number of calls answered on time in period 2 as a function of the staffing levels in period 1 and period 2. (Notice that this is equivalent to $\bar{g}$ with $l = 0$). The number of servers ranges from 1 to 30 in period 1 and from 1 to 40 in period 2. We also include the contours of the function, which should at a minimum form convex sets; see Figure 5.2 (b). The function appears to follow a multi-dimensional extension of an S-shaped curve discussed in Section 5.1 (see also Ingolfsson et al., 2003).

Relying on intuition derived from analyzing such S-shaped functions of continuous

variables,[1] one can observe that the pseudoconcavity property is violated at very low staffing levels, although it appears to hold at staffing levels that have more than 10 servers in period 1 and more than 20 servers in period 2. The violation at low staffing levels is due to the fact that the performance function is essentially flat in this region. I.e., there are so few servers and calls are so backed up that no calls are answered on time, and adding an extra server would do little to alleviate the situation; cf. low values of y_1 and y_2 in Figure 5.2. It is certainly possible that we would encounter such a staffing level in the cutting plane algorithm for the sample average approximation of the service level function; an attempt to compute or estimate a pseudogradient at this point would produce a zero vector. Therefore, as a feasibility cut at such a point $\hat{y}$, we might impose a lower bound on the number of servers in the period i, i.e., add the constraint $y_i \geq \hat{y}_i + 1$. This is not necessarily a valid feasibility cut, since the reason for calls backing up might be under-staffing in previous periods. In our implementation, if such a cut is added during the algorithm, we verify at termination that the corresponding constraint is not tight at the optimal solution found. If it is tight, then one should lower this bound and do more iterations of the cutting plane method.

Although the sample averages of the service level functions are not discrete pseudoconcave, they appear to be at least approximately discrete pseudoconcave for practical staffing levels. Furthermore, we have a strategy for dealing with areas where the function is not pseudoconcave. It seems reasonable, however, to assume that the *expected* service level function is pseudoconcave, since in expected value it would be likely that the probability of answering calls would always increase, although possibly by a very small value, when more servers are added. This would also hold for the sample average of the service level function, for a sufficiently large sample size.

---

[1]Differentiable functions of this shape can be characterized as *quasiconcave*, see Definition 3.5.5 in Bazaraa et al. (1993).

Therefore, when we prove the convergence results in Section 5.3.3 we assume that the sample averages of the service level functions are discrete pseudoconcave.

5.3.2 A Simulation-Based Cutting Plane Method for the Call Center Staffing Problem with Pseudoconcave Service Level Functions

In this subsection we describe our modification of ACCPM for the problem (5.2). At the beginning of a typical iteration, we have available a feasible solution y^* of (5.2), and an upper bound $z = f(y^*)$ on the optimal value of the problem. The point y^* is the best feasible solution found by the algorithm so far. We also have available a polyhedron $P = \{y \in \mathbb{R}^p : y \geq 0,\ Dy \geq d\}$ that is known to contain the feasible region.

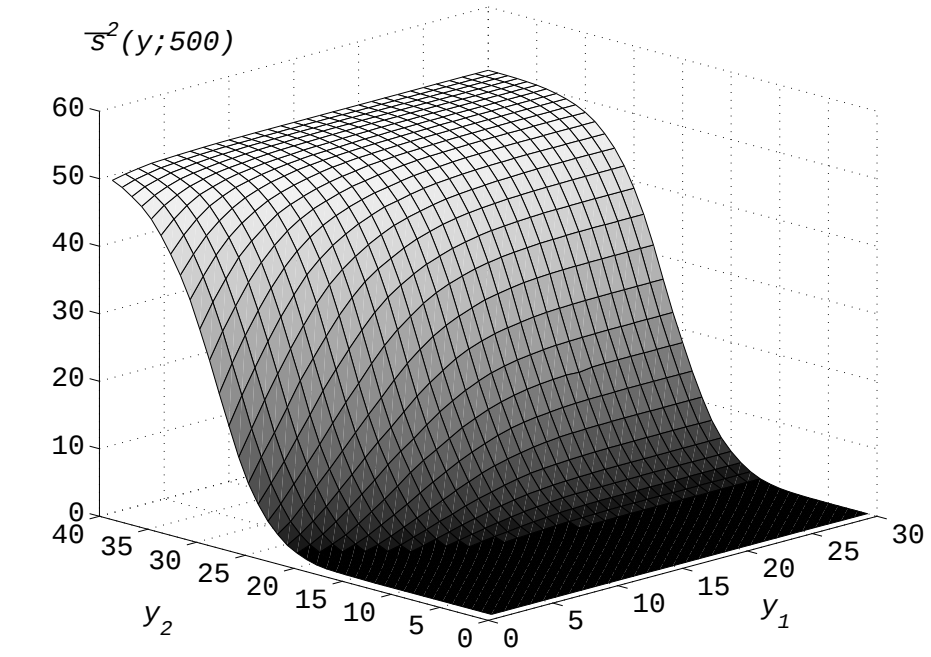
Suppose y^*, z and P are as above. If y^* is not an optimal solution, then, since $f(y)$ takes on integer values, the set

$$Q := \{y \in P : f(y) \leq z - 1,\ y \text{ integer}\},$$

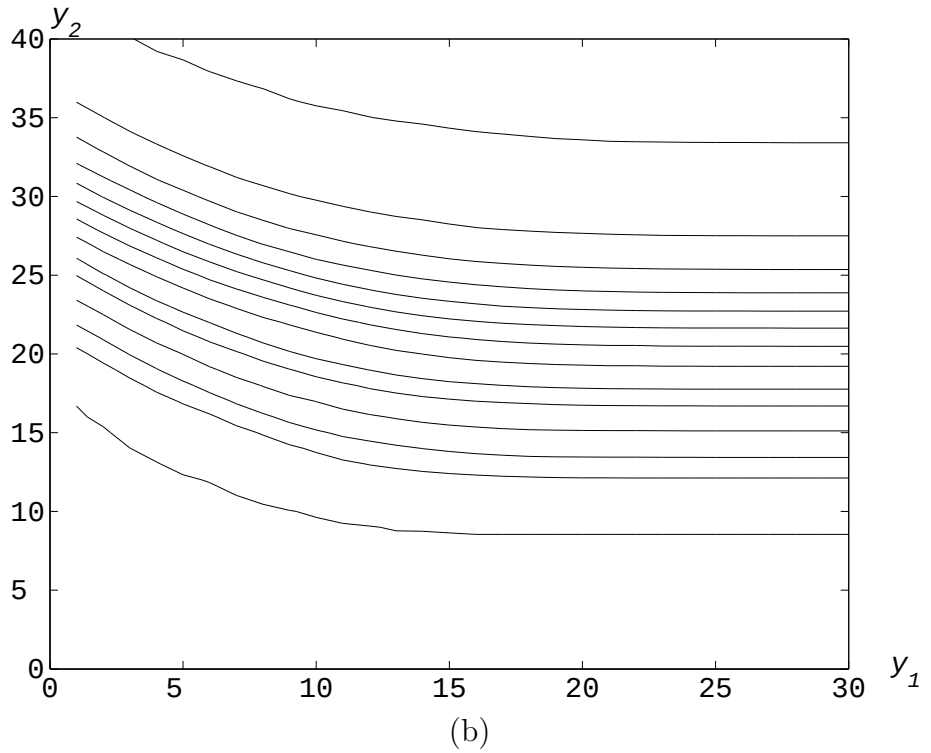
is nonempty and contains all feasible solutions with objective values better than z; hence we refer to it as the localization set. The localization set is empty precisely if y^* is an optimal solution. This observation provides grounds for the termination criteria we specify in our algorithm.

Computing the next iterate. First we find the analytic center of a polyhedral relaxation of the localization set Q. In particular, we solve the following optimization problem:

$$\begin{array}{ll} \min & -w \log(z - a - c^T x) - \sum_{i=1}^{p} \log y_i - \sum_{j=1}^{r} \log(D_{j\cdot} y - d_j) \\ \text{s.t.} & Ax \geq y \\ & x \geq 0, \end{array} \tag{5.7}$$



(a)



(b)

Figure 5.2: (a) The sample average (sample size $n = 500$) of the number of calls answered on time in period 2 as a function of the staffing levels in periods 1 and 2. (b) The contours of the service level function in (a).

where the constant $a \in (0,1)$ ensures that only points with an objective value better than z are considered. Due to Assumption 2.2 the effective feasible region of (5.7) is bounded; hence the problem has an optimal solution so long as it has a feasible point in the effective domain of the objective. If (5.7) has no solution, the algorithm terminates with y^* as an optimal solution; otherwise, let $(y^{\text{ac}}, x^{\text{ac}})$ be a solution of (5.7). Here, $w > 0$ is the weight constant; we discuss later how this constant is determined in the algorithm.

Note that the analytic center found in the previous step is almost certainly not integer, and rounding y^{ac} to the nearest integer can result in a point outside of the localization set. To capitalize on the centrality of the analytic center in the localization set, we instead find the closest integer point in the effective feasible region of (5.7), i.e., solve

$$\begin{array}{lll} \min & \|y - y^{\text{ac}}\| & \\ \text{s.t.} & Ax & \geq y \\ & c^T x & \leq z - 1 \\ & Dy & \geq d \\ & y & \geq 0 \text{ and integer} \\ & x & \geq 0 \text{ and integer.} \end{array} \tag{5.8}$$

If (5.8) is infeasible, the algorithm terminates with y^* as an optimal solution; otherwise, let $(\hat{y}, \hat{x})$ be the solution of (5.8) and choose $\hat{y}$ as the next iterate. Here, $\|y - y^{\text{ac}}\|$ is a measure of the distance between y and y^{ac}. If we choose the L_1-norm as the measure, i.e., $\|y - y^{\text{ac}}\| = \sum_{i=1}^{p} |y_i - y_i^{\text{ac}}|$, then (5.8) is a linear integer program. We discuss the computational requirements of solving this problem at each iteration when we talk about the overall computational effort in relation to the computational experiments.

Estimating the service levels. Next we compute $\bar{g}_i(\hat{y}; n)$ for all i via simulation.

Adding an optimality cut. If $\bar{g}_i(\hat{y}; n) \geq 0$ for all i, then $\hat{y}$ satisfies the service level requirements. Since $c^T\hat{x} \leq z - 1$, $\hat{y}$ is contained in the localization set, i.e., it is the best staffing level so far. Note that $c^T\hat{x}$ is not necessarily the cost associated with staffing level $\hat{y}$, since $c^T x$ is not being minimized in (5.8). To compute the cost associated with $\hat{y}$ we instead solve (5.3) to get $f(\hat{y})$ and update $z := f(\hat{y})$.

Adding a feasibility cut. If $\bar{g}_i(\hat{y}; n) < 0$ for some i, then we add a feasibility cut for each i such that $\bar{g}_i(\hat{y}; n) < 0$. In particular, we estimate a pseudogradient, $\bar{q}_i(\hat{y}; n)$, of $\bar{g}_i(\cdot; n)$ at $\hat{y}$. If $\bar{q}_i(\hat{y}; n) \neq 0$, we add a feasibility cut of the form

$$\bar{q}_i(\hat{y}; n)^T y \geq \bar{q}_i(\hat{y}; n)^T \hat{y} + \epsilon \tag{5.9}$$

for some small constant $\epsilon > 0$ (we discuss the role of ϵ in the discussion of the convergence of the method in Section 5.3.3). If $\bar{q}_i(\hat{y}; n) = 0$, the feasibility cut takes the form of a lower bound on the number of servers (see discussion in Section 5.3.1). We update D and d to reflect the cuts added.

The above procedure is then repeated. An illustration of the localization set and the feasible regions and solutions of problems (5.7) and (5.8) in each iteration is shown in Figure 5.3. We summarize the simulation-based analytic center cutting plane method (SACCPM) for the call center staffing problem in Figure 5.4. (To shorten the presentation, the description of the algorithm in Figure 5.4 is only specified for the case when the constraint functions $\bar{g}_i(\cdot; n)$ are in fact discrete pseudoconcave.)

The weight parameter w can be increased to "push" the weighted analytic center away from the optimality cuts. There are no theoretical results on how to choose the weights, but some computational experiments have been done to test different values of the weights, and in fact, weights on the feasibility cuts (Goffin et al., 1992; duMerle et al., 1998). The choice of the weights is problem and data dependent (see Section 5.5 for the particular choice used in our implementation).

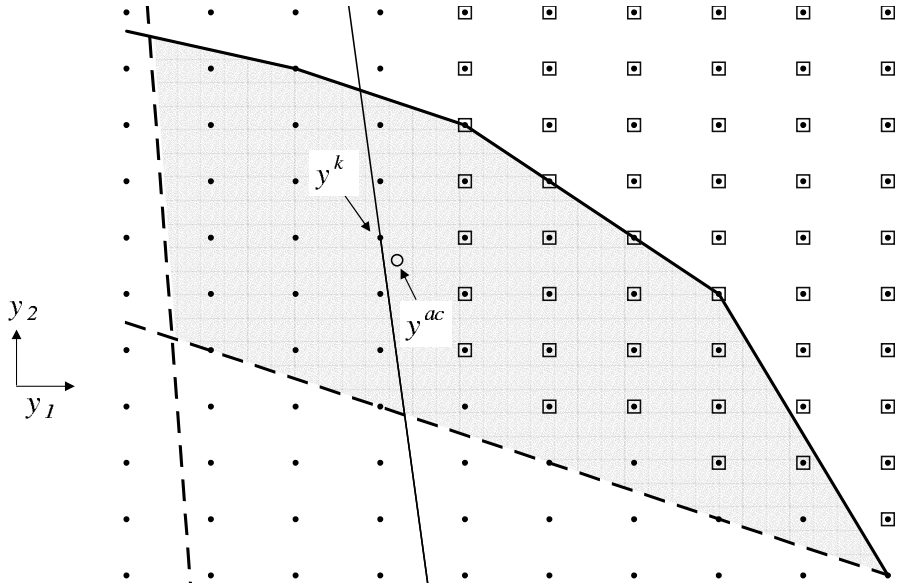


Figure 5.3: Illustration of the feasible region and an iterate in the SACCPM. The picture shows points in the space of the y variables in the case $p = 2$. The shaded area represents the localization set Q (ignoring integer constraints).The squares represent the points that are feasible for the sample average approximation problem (5.2). The thick solid line segments represent an optimality cut $f(y) \leq z - 1$ and the dotted lines represent the feasibility constraints $Dy \geq d$. The point $y^{\rm ac}$ is the solution of the analytic center problem and y^k is the closest integer in Q. The line through y^k represents a feasibility cut that would be added in this iteration.

Initialization Start with a feasible solution y^0 to (5.2). Let $z^0 := f(y^0)$, and $y^* := y^0$. Let $P^0 = \{y \geq 0 : D^0 y \geq d^0\}$, where D^0 and d^0 are empty. Choose an $\epsilon > 0$, $a \in (0, 1)$ and $w^0 > 0$. Let $k := 0$.

Step 1 Solve problem (5.7) with $w = w^k$, $z = z^k$, and $P = P^k$. If (5.7) does not have a (feasible) solution, then terminate with y^* as the optimal solution and z^k as the optimal value; otherwise let y^{ac} be the solution of (5.7).

Step 2 Solve problem (5.8) with $z = z^k$ and $P = P^k$. If (5.8) is infeasible, then terminate with y^* as the optimal solution and z^k as the optimal value; otherwise let y^k be the optimal solution of (5.8) found.

Step 3a If $\bar{g}_i(y^k; n) \geq 0 \ \forall \ i \in \{1, \ldots, p\}$, let $z^{k+1} := f(y^k)$, $y^* := y^k$, $D^{k+1} := D^k$ and $d^{k+1} := d^k$.

Step 3b If $\bar{g}_i(y^k; n) < 0$ for some $i \in \{1, \ldots, p\}$, then add the constraint $\bar{q}_i(y^k; n)^T y \geq \bar{q}_i(y^k; n)^T y^k + \epsilon$ for each i such that $\bar{g}_i(y^k; n) < 0$. Update D^k and d^k to include the added inequalities, and let $P^{k+1} = \{y \geq 0 : D^{k+1} y \geq d^{k+1}\}$ represent the new constraint set. Let $z^{k+1} := z^k$.

Step 4 Compute w^{k+1} and let $k := k + 1$. Go to Step 1.

Figure 5.4: The simulation-based analytic center cutting plane method (SACCPM).

In practice it is useful to maintain a lower bound on the optimal value of the problem (5.2) throughout the algorithm, in addition to the upper bound z. In particular, the algorithm can be terminated as soon as the gap between the upper and lower bounds is sufficiently small, indicating that the current "incumbent" y^* is a sufficiently good solution of the problem. A lower bound can be found as the optimal objective value of

$$\begin{array}{lll} \min & c^T x & \\ \text{s.t.} & Ax & \geq y \\ & c^T x & \leq z \\ & Dy & \geq d \\ & y & \geq 0 \text{ and integer} \\ & x & \geq 0 \text{ and integer,} \end{array} \tag{5.10}$$

or, for a weaker but easier to compute bound, of the linear programming relaxation of (5.10).

5.3.3 Convergence of the SACCPM

In this section we give conditions under which the SACCPM terminates with y^* as an optimal solution of (5.2). First, we argue that the algorithm does indeed terminate and then we show that y^* is an optimal solution of (5.2) at termination. To prove the results we make the following two assumptions.

Assumption 5.1. The functions $\bar{g}_i(y; n)$ are discrete pseudoconcave in y for all $i \in \{1, \ldots, p\}$.

Assumption 5.2. In the implementation of the SACCPM, $\bar{q}_i(\hat{y}; n)$ is a pseudogradient of $\bar{g}_i(\cdot; n)$ at $\hat{y}$.

We also define the sets

$$\begin{aligned}
\Gamma &:= \{y \geq 0 \text{ and integer} : f(y) \leq z^0\}, \\
\Psi &:= \Gamma \cap \{y : \bar{g}_i(y;n) \geq 0 \; \forall \; i \in \{1, \ldots p\}\}, \\
\Upsilon &:= \Gamma \setminus \Psi \text{ and} \\
I(y) &:= \{i : \bar{g}_i(y;n) < 0.\}
\end{aligned}$$

In words, Γ is the set of points that are potentially visited by the algorithm and contains the set of optimal solutions. The set Ψ is the set of points in Γ that are feasible for the sample average approximation problem (5.2). The set Υ is the set of points in Γ that are not feasible for the sample average approximation problem (5.2). The set $I(y)$ is the set of periods in which the sample average of the service level function is not acceptable given the staffing levels y. The following lemma says that all solutions in Ψ satisfy potential feasibility cuts (5.9) for some appropriately chosen ϵ.

Lemma 5.1. *Let $\bar{q}_i(\hat{y};n)$ be a pseudogradient of $\bar{g}_i(\cdot;n)$ at $\hat{y}$ and suppose Assumptions 2.2 and 5.1 hold. Then there exists an $\tilde{\epsilon} > 0$ such that $\bar{q}_i(\hat{y};n)^T(y-\hat{y}) \geq \tilde{\epsilon} \; \forall \; y \in \Psi, \hat{y} \in \Upsilon, \; i \in I(\hat{y})$.*

Proof: Let $\hat{y} \in \Upsilon$ be fixed. Suppose that $\bar{q}_i(\hat{y};n)^T(y-\hat{y}) \leq 0$ for some $y \in \Psi$ and $i \in I(\hat{y})$. Then $\bar{g}_i(y;n) \leq \bar{g}_i(\hat{y};n) < 0$, where the first inequality follows by Assumption 5.1 and the second inequality follows since $i \in I(\hat{y})$. This is a contradiction since $y \in \Psi$, and therefore $\bar{q}_i(\hat{y};n)^T(y-\hat{y}) > 0 \; \forall \; y \in \Psi, \; i \in I(\hat{y})$.

Let $\epsilon(\hat{y}) = \min_{i \in I(\hat{y})} \min_{y \in \Psi} \bar{q}_i(\hat{y};n)^T(y-\hat{y})$. The set Ψ in the inner minimum is finite by Assumption 2.2 and therefore the inner minimum is attained for some $y \in \Psi$. Since $I(\hat{y})$ is also finite, the outer minimum is also attained for some $i \in I(\hat{y})$. Therefore, $\epsilon(\hat{y}) > 0$.

Finally, let $\tilde{\epsilon} = \min_{\hat{y} \in \Upsilon} \epsilon(\hat{y})$. Then $\tilde{\epsilon} > 0$ since Υ is finite. □

Theorem 5.2. *Suppose* (5.2) *has an optimal solution and that Assumptions 2.2, 5.1 and 5.2 hold. Furthermore, let $0 < \epsilon \leq \tilde{\epsilon}$, where $\tilde{\epsilon}$ is as in Lemma 5.1. Then the*

SACCPM terminates in a finite number of iterations returning y^*, which is an optimal solution of (5.2).

Proof: The SACCPM only visits points that are feasible solutions of (5.8). The set of feasible solutions of (5.8) at every iteration is a subset of Γ and is therefore a finite set by Assumption 2.2. Suppose that the kth iterate y^k is the same as a previous iterate y^j for some $j < k$. If y^j is in Ψ then $f(y^k) \leq z - 1 \leq f(y^j) - 1$, where the second inequality follows since z is the cost of the best feasible solution visited before iteration k. This is a contradiction, so y^j is not in Ψ. On the other hand, if y^j is in Υ then $\bar{g}_i(y^j; n) < 0$ for some $i \in I(y^j)$. Since y^k is in P, $\bar{q}_i(y^k; n)^T(y^k - y^j) \geq \epsilon > 0$ which is also a contradiction and y^k can therefore not be equal to y^j for any $j < k$. Hence, the SACCPM does not visit any point in Γ more than once. Since Γ is a finite set, the algorithm is finite.

By Lemma 5.1 the feasibility cuts never cut off any of the feasible solutions of (5.2). When at termination, problems (5.7) or (5.8) do not have a feasible solution, it means that all the feasible solutions for (5.2) have an objective value greater than $z - 1$. But y^* is feasible for (5.2) and has an objective value of z and is, therefore, optimal, since $f(y)$ is integer by Assumption 2.2. □

The theorem asserts that there exists an $\tilde{\epsilon}$ such that the algorithm terminates with an optimal solution of (5.2) if $0 < \epsilon \leq \tilde{\epsilon}$. In practice, the value of $\tilde{\epsilon}$ is unknown, but in general a "small" value for ϵ should be chosen, as in Section 5.5.

5.4 Analytical Queuing Methods

In this section we give an overview of heuristics queuing heuristics for staffing call centers. Heuristics based on the Erlang C formula (5.11) are widely used to determine staffing levels in call centers (Cleveland and Mayben, 1997). In Green et al. (2001, 2003) several heuristics for improving the performance of the basic Erlang C model are evaluated. We believe that the methods in Green et al. (2001, 2003) are among the

best analytical methods (see Chapter I for more references) for determining required staffing levels to answer a minimum fraction of calls before a threshold time limit. Therefore, we use these methods as benchmarks for our method in the numerical study in Section 5.5.

Traditional queuing methods for staffing call centers assume that the process is in steady state, i.e., the arrival rate is fixed and the call center has been open for long enough that the initial state of the call center does not matter. Then assuming that the call center can be modeled as an $M/M/s$ queuing system, the probability that a customer has to wait more than τ time units, as a function of the number of servers s, is given by (Gross and Harris, 1998, p. 72)

$$P(s) = \left(\frac{\sum_{n=0}^{s-1} \frac{R^n}{n!}}{\sum_{n=0}^{s-1} \frac{R^n}{n!} + \frac{R^s}{s!(1-\rho)}} \right) e^{-(s\mu-\lambda)\tau} \tag{5.11}$$

where $R = \lambda/\mu$ is the load and $\rho = \lambda/s\mu$ is the server utilization. This equation is only defined for $\rho < 1$.

When the arrival rate changes between periods or within periods, Green et al. (2001) proposed to adjust the value of the arrival rate λ in (5.11). This is a straightforward method that can give good staffing levels, at least for a call center operation that is similar to the $M(t)/M/s(t)$ model. They consider 6 different adjustments of the arrival rate. The 6 different schemes are given in Table 5.1. In Green et al. (2003) only SIPPavg, LAGavg and LAGmax are considered. When we computed the arrival rate to use for the LAG methods in the first period we assumed that the arrival rate prior to time zero was equal to the arrival rate at the beginning of the first period.

The required staffing, y_i, in period i is computed by letting $\lambda = \Lambda_i$ in (5.11) and choosing

$$y_i = \min\{s > 0 \text{ and integer} : P(s) \geq 1 - l_i\}.$$

The cost of the resulting staffing level is $f(y)$. The actual number of agents available in period i can actually be greater than y_i because of slack in the shift constraint $Ax \geq y$

Method	Λ_i
SIPPavg	$\int_{t_{i-1}}^{t_i} \lambda(t)dt$
SIPPmax	$\max_{t_{i-1}<t\leq t_i} \lambda(t)$
SIPPmix	If $\lambda(t)$ is nondecreasing in period i use SIPPavg rate, otherwise use SIPPmax rate.
LAGavg	$\int_{t_{i-1}-1/\mu}^{t_i-1/\mu} \lambda(t)dt$
LAGmax	$\max_{t_{i-1}-1/\mu<t\leq t_i-1/\mu} \lambda(t)$
LAGmix	If $\lambda(t)$ is nondecreasing in $[t_{i-1}-1/\mu, t_{i-1}-1/\mu]$ use LAGavg rate, otherwise use LAGmax rate.

Table 5.1: Different methods for adjusting the arrival rate to use in Equation (5.11). Here, t_i is the time when period i ends ($t_0 \equiv 0$), $1/\mu$ is the mean service time and Λ_i is the rate to be used to determine the staffing in period i.

in (5.3). We included the additional staffing from the slack when we evaluated the performance of the staffing levels obtained by these analytical heuristics.

5.5 Numerical Results

In this section we consider two call center examples. In Section 5.5.1 we describe our first example, a 72 period call center staffing problem with time varying demand, and the discussion in Section 5.5.2 highlights the properties and implementation issues of our method. We study a similar example, described in Section 5.5.3, to compare the SACCPM to the analytical queuing heuristics. The results in Section 5.5.4 show that our simulation-based analytic center cutting plane method, when used in conjunction with the finite difference method for estimating gradients, does better in most cases than the analytical queuing methods. In Section 5.5.5 we comment on the computational requirements of the SACCPM.

5.5.1 Example 1: Staffing a Call Center over 72 Periods

The objective of this example is to see whether we can apply the SACCPM algorithm to solve a larger problem than the 5 period problem that was solved in Section 3.5. We pointed out in Section 3.5.3 that the SKCPM failed to give a good solution to the

problem described in this example, even if lower bounds on the staffing levels were computed in an attempt to start the algorithm in a region where the service level functions are concave. We will both study the rate of convergence and investigate the quality of the solutions given by the algorithm. As part of the study we compare the use of the IPA and FD techniques for computing pseudogradients and the effects of shift constraints on the solutions we get from the algorithm.

We tested the SACCPM on a call center that has the following characteristics (in queuing theory terms this is an $M(t)/M/s(t)$ system).

- The call center is open for 18 hours a day, e.g., from 6am-12am.
- There are 72 time periods, each of length 15 minutes.
- In each period 80% of calls should be answered in less than 90 seconds.
- The arrival process on any given day is a nonhomogeneous Poisson process with rate function $\lambda(t), t \in [6\text{am}, 12\text{am}]$ as shown in Figure 5.5. The rate function is piecewise linear with breakpoints possible only at the beginning of a period.
- Calls are answered in the order they are received.
- The service times for each call can be modeled as independent exponential random variables with rate of 4 calls per hour.
- When there is a reduction in the number of servers at the end of a period, every departing server completes a call that is already in service. A new call cannot enter service until there are fewer calls in service than there are servers for the new period.

We study the performance both in the presence of shift constraints and when there are no shift constraints.

- Each shift covers 6 hours, or 24 periods.

- The shifts can only start on the hour and no later than 6 hours before the end of the day. This results in 13 shifts; 6am-12pm, 7am-1pm, ..., 5pm-11pm, 6pm-12am.

- The cost for each shift is equal to 24 man-periods per shift.

- When there are no shifts the cost of the staffing levels is computed by adding the number of servers in all periods.

We tested the method using both IPA and the finite difference method of Chapter IV for estimating the pseudogradients.

- The IPA estimator used is the biased estimator with a varying number of servers; see Section 4.5.6. We used the biased IPA estimator rather than the unbiased estimator primarily because we can get the sample average of the service level and the gradient estimator in a single simulation. For the unbiased IPA estimator using a fixed number of servers in all periods we need to run a separate simulation to get better estimates of the service level function than those obtained by the fixed number of servers model. Furthermore, the biased estimator seemed to give slightly better gradient estimates in the small numerical study in Section 4.6. We did not consider the LR method because of its high variance.

- We used a C++ program to compute the sample average of the service level and the IPA gradient estimator when the IPA method was used.

- We built a simulation model using the ProModel simulation software to compute the sample average of the service level function when the FD method was used.

For the optimization, and to compute the cuts, we used the following applications.

- We used Visual Basic for Applications and Microsoft Excel to store the data and to compute the cuts.

- We used the AMPL modeling language to model and call a solver for the analytic center problem (5.7) and for the IP (5.8) of finding an integer point close to the analytic center.
- We used the MINOS solver to solve the analytic center problem (5.7).
- We used the CPLEX solver to solve the IP (5.8).
- We used the Excel solver to compute the cost $f(\hat{y})$ of a particular staffing level $\hat{y}$.

Finally, we describe the settings of the parameters that are specific to the SACCPM.

- In our initial experiments we let $w_k = 1$, but that resulted in very few feasibility cuts and slow convergence. To balance optimality and feasibility cuts we set the weights $w^k = r$ where r is the number of feasibility cuts that have been added in iterations 1 trough $k-1$ (we let $w^k = 1$ if no feasibility cuts have been added).
- We used $\epsilon \leq 10^{-5}$. The smaller ϵ is the less likely it is that we cut off a feasible solution, but too small values can result in that we do not cut off the current solution because of numerical precision.
- We chose $a = 1 - 10^{-5}$.
- Instead of a feasible starting point y^0 we started with an upper bound on the staffing levels, i.e., we added the constraints $y_i \leq 100 \; \forall \; i \in \{1, \ldots, p\}$ to the IP (5.8) and the term $-\sum_{i=1}^{p} \log(100 - y_i)$ to the objective of the analytic center problem. We chose 100 as the upper bound because it is unlikely, given the arrival and service rates, that the optimal staffing level would be greater than 100 in any period. Adding an upper bound instead of starting with a feasible solution speeds up the algorithm in the beginning, given the choice of w^k. If

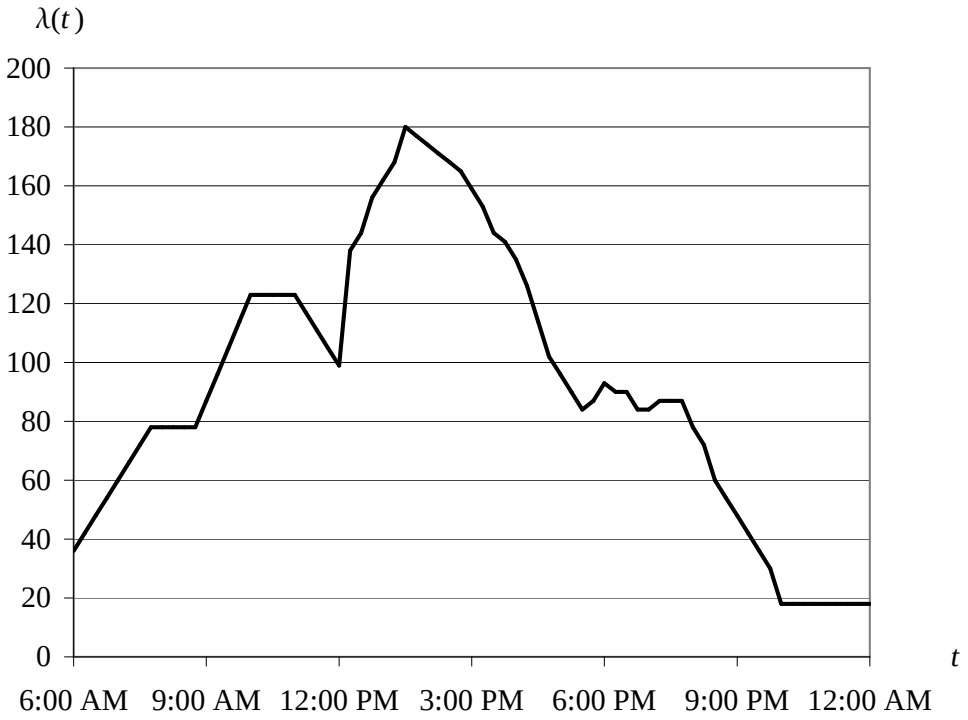


Figure 5.5: The arrival rate function of Example 1. The rate $\lambda(t)$ is in calls/hour.

we had started with a feasible solution and $w^1 = 1$ then, because there are 72 $(= p)$ lower bound constraint then the optimality cut has a small weight in the beginning. The upper bounds help "push" down the staffing levels, at least in the initial iterations.

5.5.2 Results of Example 1

Experiment 1. We first tested the algorithm on the SAA problem (5.2) with a simple size of $n = 500$, using the *finite difference method* for estimating the pseudogradients. We *did not include any shifts*, so the objective was simply to minimize the total number of servers over all periods. The solution returned by the algorithm was visited in iteration 104 and has an objective value of 1985. We terminated the algorithm in iteration 124 when the integer program failed to find a solution after searching more than 24 million nodes of the branch and bound tree. We could not

prove that the IP was infeasible, but the optimality gap in the SAA problem was only 14 man-periods, or less than 1%, at the incumbent solution at that iteration.

Figure 5.6 shows the staffing levels in each period in different iterations. We see that the shape of the final solution is very similar to the shape of the arrival rate function in Figure 5.5.

We also plot the upper and lower bounds, on the optimal objective value. We computed the lower bounds from the integer program (5.10) rather than its LP relaxation. After a number of feasibility cuts were added, the lower bound IP (5.10) became difficult to solve, so we terminated the IP early with a suboptimal solution. We see that the optimality gap decreases rather quickly in the beginning, and then decreases slowly as the iterates get close to an optimal solution. One could obviously use the bounds and terminate the algorithm earlier than iteration 124 if only a good solution is required.

From the optimality gap we see that the lower bound approaches the optimal value much sooner than the upper bound. This is partly due to the choice of the weight constants. In the initial stages, the weights of the optimality cuts were set equal to $w_k = 1$. After 77 iterations we changed the weights and let the weight in each iteration equal to r (the number of feasibility cuts in the preceding iterations) to speed up the convergence. The upper bound drops in iteration 83, which is the first feasible solution visited after we changed the weights. It is also notable that in iteration 78 there is a bit of a jump (though hard to see on the plot) in the lower bound. This is because the weight on the optimality cut pushed the solution to a point with fewer servers and, therefore, more feasibility cuts were added.

Experiment 2. In this experiment we used a sample size of $n = 100$. We used the *IPA* estimator to compute the pseudogradients. There are *no shift constraints*. We set the weights equal to r. We were able to use $\epsilon = 0$ here and still get convergence.

To see why, note that the analytic center is pushed away from the previous iterates, so even if the SACCPM does not cut off the current infeasible iterate (since $\epsilon = 0$), it may be cut off when a cut is added in a later iteration. The algorithm was terminated in iteration 33, again after the integer program failed to find a solution after searching more than 24 million nodes of the branch and bound tree. The objective value of the solution is 2112 and the optimality gap is only 9 man-periods, or less than 0.5%.

Figure 5.8 shows the iterates and Figure 5.9 shows the upper and lower bounds. The iterates look very jagged, and the final solution does not appear to be a true optimal solution. The cost is much higher than the cost of the solution in Experiment 1 for the same problem (although a different sample size and different random numbers are used). The reason is that the IPA gradient estimates do not seem to approximate the pseudogradients well, thus cutting off parts of the feasible and optimal regions. If we compare the optimality gap for Experiment 1 (Figure 5.6) with the one in this experiment (Figure 5.9) we see that the algorithm converges much faster in this experiment. This is primarily due to a better choice of weights rather than differences in the pseudogradients.

Experiment 3. In this experiment we used a sample size of $n = 100$. We used the *finite difference method* to compute pseudogradients, incorporated the *shifts*, and set w^k equal to r and set $\epsilon = 10^{-5}$. The integer program (5.8) was infeasible in iteration 20 and the solution returned by the algorithm was visited in iteration 17. The cost of the solution is 97 shifts, or 2328 man-periods. In Figure 5.11 we make a comparison between this experiment, the next experiment, which uses IPA gradients, and the LAGavg queuing method.

Experiment 4. In this experiment we used a sample size of $n = 100$. We used the *IPA* gradient estimation approach to compute pseudogradients, incorporated the *shifts* and set w^k equal to to r. The integer program (5.8) was infeasible in iteration

20 and the solution returned by the algorithm was visited in iteration 18. The cost of the solution is 97 shifts or 2328 man-periods, which is the same cost as that of the solution in Experiment 3. The IPA method appears to do better with shift constraints than without, because of the additional slack in staffing levels "smoothes" out the jagged solutions exhibited in Experiment 2.

Comparison with the LAGavg method. The LAGavg method described in Section 5.4 has been reported to do well for this type of problem (Green et al., 2001). Therefore, we computed the staffing levels using this method with and without the shifts.

Figure 5.10 shows the solutions from Experiments 1 and 2 versus the solution obtained by the LAGavg method. From the graph we see that the solution in Experiment 1 (finite differences) is very similar to the solution of the LAGavg method. The cost of the LAGavg method is 2008, which is about 1% higher than the 1985 in Experiment 1. The solution in Experiment 2 (IPA) does not compare favorably with the cost of 2112 and a solution that qualitatively does not look like a good solution.

Figure 5.11 shows the solutions from Experiments 3 and 4 versus the solution obtained by the LAGavg method. In the presence of shift constraints it is more difficult to compare the solutions by looking at the graph because there may be multiple solutions that have the same cost. The cost of the LAGavg method is 99 shifts, or 2376 man-periods, which is 2% higher than the cost in Experiment 3. The reason why the difference in the cost is greater now is that the SACCPM takes the shifts into account when computing the staffing levels, while in the LAGavg a lower bound on the staffing levels is pre-computed, and these lower bounds are then covered with the shifts at a minimum cost.

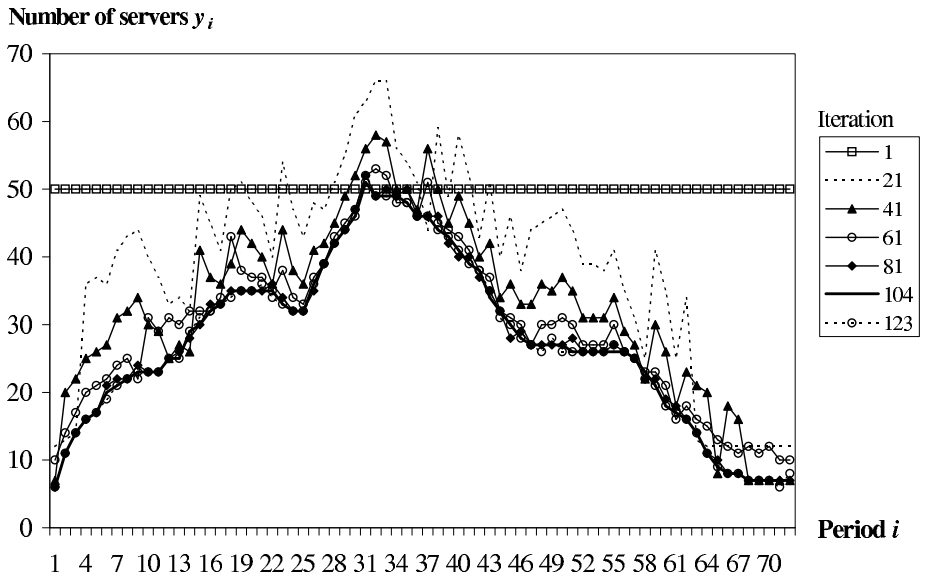


Figure 5.6: The iterates of Experiment 1. The algorithm terminated in iteration 124 and the solution returned by the algorithm was visited in iteration 104.

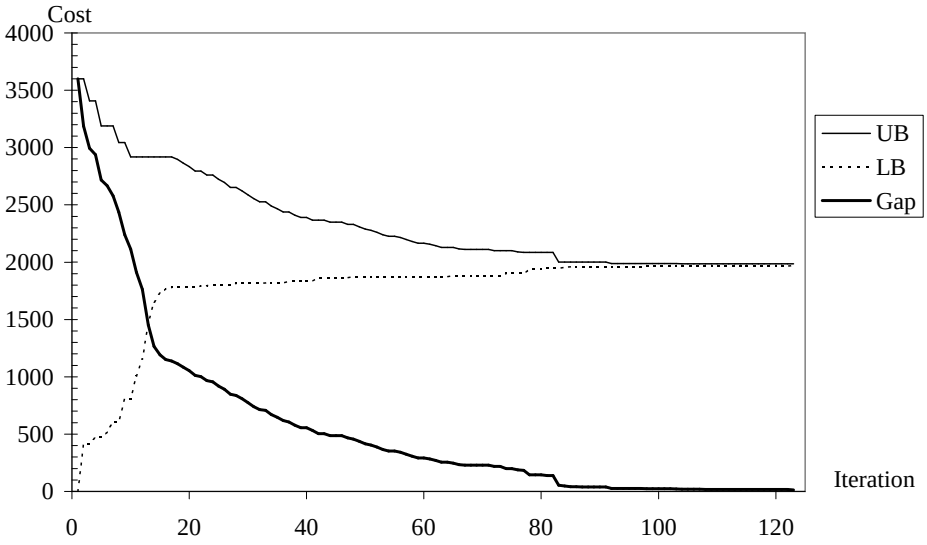


Figure 5.7: The optimality gap in Experiment 1.

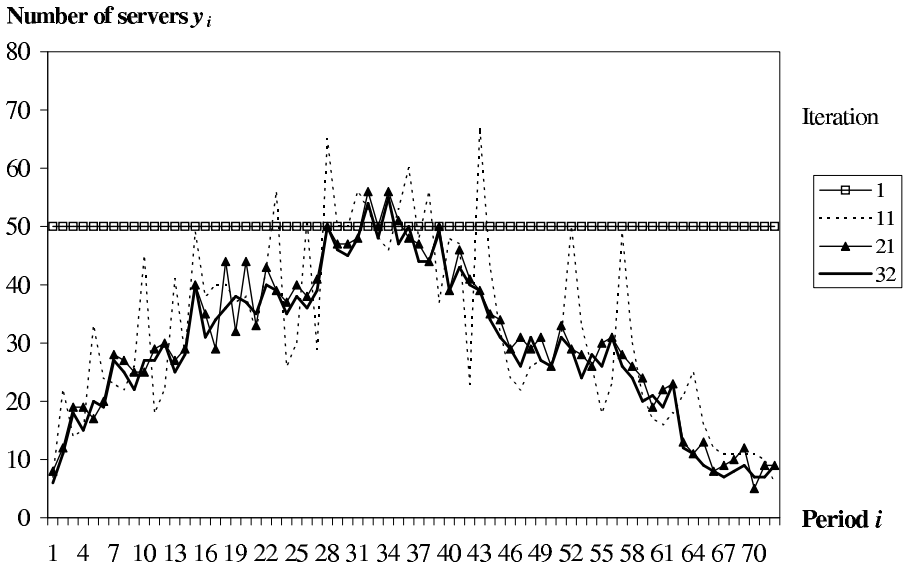


Figure 5.8: The iterates of Experiment 2. The algorithm was terminated in iteration 33 and the solution returned by the algorithm was visited in iteration 32.

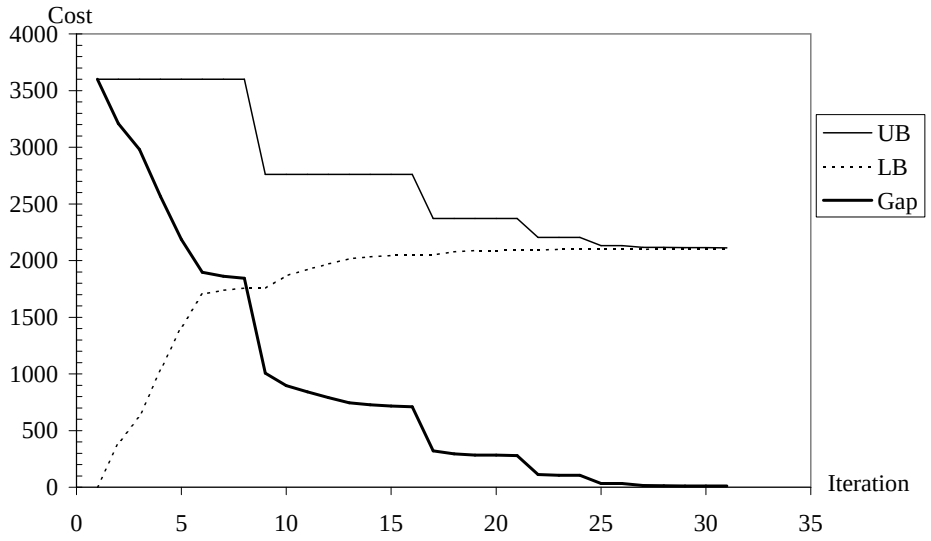


Figure 5.9: The optimality gap in Experiment 2.

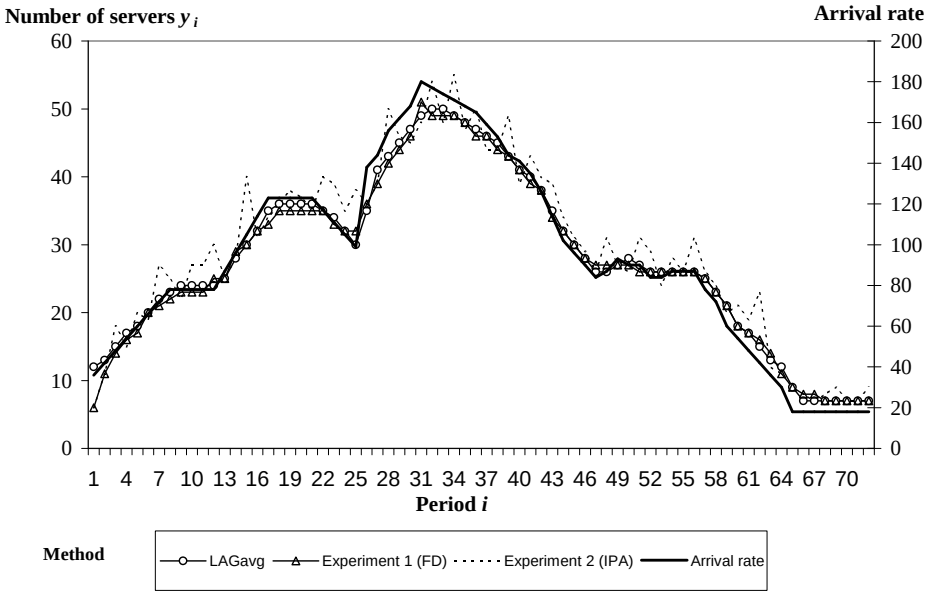


Figure 5.10: Example 1: No shifts. Comparison between the solutions from the SACCPM when there are no shift constraints using finite differences (Experiment 1) and IPA (Experiment 2) to compute pseudogradients. LAGavg is the solution obtained by the LAGavg queuing method. There are no shift constraints.

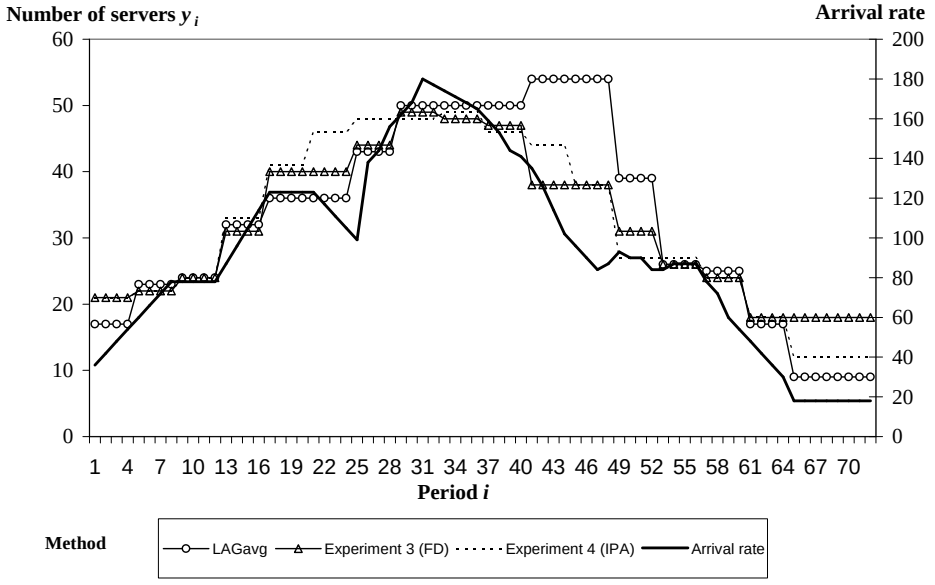


Figure 5.11: Example 1: Shifts. Comparison between the solutions from the SACCPM in the presence of shift constraints using finite differences (Experiment 3) and IPA (Experiment 4) to compute pseudogradients. LAGavg is the solution obtained by the LAGavg queuing method.

5.5.3 Example 2: Comparing the SACCPM to Analytical Queuing Methods

Our test model is similar to the models used in Green et al. (2001, 2003), which are call centers that can be modeled as $M(t)/M/s(t)$ queuing systems. In the Green et al. (2001) paper a call center operating 24 hours a day and 7 days a week is studied, while the subject of the Green et al. (2003) paper are call centers with limited hours of operation. We consider a call center with limited hours of operation that is open from 6am to 12am. The call center has the following additional characteristics.

- The planning horizon consists of a single day's operation. The 18 hour planning horizon is broken into 72 time periods, each of length 15 minutes.
- In each period 80% of calls should be answered immediately. This is equivalent to setting $\tau = 0$ and $l_i = 0.8$.
- The service times for each call can be modeled as independent exponential random variables at rate μ.
- The average load over the 18 hours is R. The load is the average arrival rate divided by the average service rate and is an indication of the size of the system.
- The arrival process on any given day is a nonhomogeneous Poisson process with arrival rate at the end of period i given by $\lambda_i = \lambda(1+\theta\sin(2\pi(t_i-6)/18)$, where t_i is the end time of period i and is measured in the hour of the day (e.g., $t_0 = 6$ and $t_{72} = 24$). The average daily arrival rate is $\lambda = R\mu$ and θ is the relative amplitude. The arrival rate at time t such that $t_{i-1} < t \leq t_{i-1}$ is given by linear interpolation of the rates λ_{i-1} and λ_i.
- Calls are answered in the order they are received.
- When there is a reduction in the number of servers at the end of a period, any

Parameter	Low	High
μ	4 calls/hour	16 calls/hour
R	8	32
θ	.25	.75
Shifts	Yes	No

Table 5.2: The parameter settings in Example 2.

server completes a call that is already in service. A new call cannot enter service until there are fewer calls in service than there are servers for the new period.

- The finite difference method and the biased IPA gradient estimator (4.53) were used to estimate the pseudogradients. In Section 5.5.4 a distinction is made between the two approaches and they are called SACCPM-FD and SACCPM-IPA, respectively.

- We study the performance both in the presence of shift constraints and when there are no shift constraints. The shifts were the same as in Example 1.

Note that we are yet to specify a value for μ, R and θ. In the experiments, we study how the SACCPM performs under two different settings (high and low) of each of these parameters. Along with the two settings for the shift constraints this results in a total of $2^4 = 16$ experiments. The high and low values for each parameter are given in Table 5.2. Green et al. (2001, 2003) additionally studied the effect of the target probability l. The target level did not appear to have as significant an effect on the reliability of each method as the other parameters did, so we do not include different settings of it in our study. Instead of shifts, they included a factor that they call the planning period which is similar to the shift factor in that the number of servers is constant in each planning period (which can be longer than what we, and they, call a period and measure the performance in).

5.5.4 Results of Example 2

We enumerated the 16 experiments as in Table 5.3. The cost of the staffing levels obtained by each method is shown in Table 5.4.

Determining feasibility. The solutions obtained by the 6 queuing methods and the SACCPM are not guaranteed to be feasible for the call center staffing problem with expected service level constraints (5.1). It is true that a solution obtained by the SACCPM is feasible for the respective sample average approximation problem. To determine the feasibility we simulated each solution using a sample size of $n = 999$ (the maximum number of iterations in Promodel). Since the simulation output of the 999 experiments is still random we decided to declare the service level requirements not met if the fraction of calls answered immediately is less than 75% in any period. This is similar to what Green et al. (2001) used to determine feasibility. They computed the service level using numerical integration of transient queuing formulae and said that the service level is infeasible in a period if the target probability is exceeded by at least 10%. We chose 75% as our threshold because the maximum 95% confidence interval half-width in our simulations using $n = 999$ was 4.4%.

The feasibility of the solutions is reported in Table 5.5. We first note that all the methods do well in most cases. The SIPPavg method struggles when there are no shifts present and the service rates are low. In this case there is a significant dependency between time periods which none of the SIPP methods take into account. We also note that the solutions from the SACCPM do not always meet the service level requirements by the 75% criterion and in many periods fail to meet the 80% requirements. This is because the sample size was only 100 when the solution was computed so there is a significant error in the estimates of the service level functions.

Ranking the methods. In Table 5.4 we put in boldface text the cost of the method that we declared a "winner." We selected the winner based on two criteria. The first criterion is that the service level must be greater than 75% in all periods based on the simulation using sample size 999. The second criterion is cost. We see that the SACCPM-FD is the winner in all but three experiments. In two of these, the solutions fail to meet the feasibility criterion. In the third case, the cost of the best solution was only 0.3% lower. In the case of ties it would make more sense to use one of the analytical queuing heuristics because they are much easier to implement. However, one would not know beforehand whether a tie would occur, and which heuristic to use.

The SACCPM-IPA does not do as well as the SACCPM-FD, since the pseudo-gradient estimates are not as reliable. The analytical queuing heuristics have been shown to do well on this particular type of queuing model. In the SACCPM, however, no assumptions are made on the arrival process or the service times, so it may apply in even more general settings.

In Section 5.3.1 we noted that the sample averages of the service level functions are not pseudoconcave for low staffing levels. In one or more iterations in some of the experiments we got zero pseudogradients, which cannot be used to generate feasibility cuts. Instead we imposed the lower bounds described in Section 5.3.1 and then checked upon termination of the algorithm whether these lower bounds were tight. The bounds were tight in Experiment 16, so we relaxed the bounds and ran the algorithm until it terminated again, this time with a solution where these kinds of bounds were not tight.

Experiment	μ	R	θ	Shifts
1	4	8	0.75	Y
2	16	8	0.75	Y
3	4	32	0.75	Y
4	16	32	0.75	Y
5	4	8	0.25	Y
6	16	8	0.25	Y
7	4	32	0.25	Y
8	16	32	0.25	Y
9	4	8	0.75	N
10	16	8	0.75	N
11	4	32	0.75	N
12	16	32	0.75	N
13	4	8	0.25	N
14	16	8	0.25	N
15	4	32	0.25	N
16	16	32	0.25	N

Table 5.3: The experiments of Example 2.

5.5.5 Computational Requirements

We can divide each iteration of the algorithm into 3 main parts in terms of computations:

1. *Solve the analytic center problem* (5.7). This usually took less than 1 second using the MINOS solver and never took more than 3 seconds.

2. *Solve the IP* (5.8) *to get an integer solution close to the analytic center.* In the beginning this takes on the order of seconds to solve. However, when there were no shifts, meaning that the solution space for y is larger, it could take millions of branch and bound nodes to find an optimal solution after a number of cuts had been added. In fact it is not necessary to find an optimal solution of the IP (5.8) to advance the algorithm, so we often terminated the IP with a suboptimal, but feasible solution to determine the next iterate. If the IP seemed infeasible, but the infeasibility was difficult to prove, and the optimality gap in

the SACCPM was less than 1%, the SACCPM was terminated with a possibly suboptimal solution.

3. *Simulate to estimate the expected service level and gradients.* When IPA was used this took from 13 seconds to 3 minutes, depending on the number of arrivals to the system, on a Pentium 4 3GHz computer. Simulating at each staffing level in ProModel took from 6 seconds to about 1 minute, but up to 73 staffing levels had to be valuated per iteration. It appeared that dependence between time periods was not a factor over more than 10 periods, so as a rule of thumb we only computed the differences for the 10 periods preceding period i and for period i if the service level constraint was not satisfied in period i. Staffing levels in subsequent periods do not have an effect on the service level in period i since the requirement is to answer 80% of the calls immediately.

Hence, estimating the pseudogradients and solving the IP (5.8) require the most computations; see Chapter VI for ideas on how the computational requirements can be reduced.

The number of iterations required is given in Table 5.4. In the initial stages, the SACCPM sometimes produced several optimality cuts before any feasibility cuts were generated. Because the weight on the optimality cuts is only 1 in the beginning, the optimality cuts will be fairly close to each other in such a case and convergence will be slow. This happens when the starting point has much higher staffing levels than what is really needed. When this occurred, we added deeper optimality cuts until the first feasibility cuts were generated by the algorithm, i.e., we lowered z. One could start the algorithm close to the solutions of the analytical queuing heuristics, or use expert guesses from call center managers, to hopefully speed up the convergence. We did not try to implement this approach.

#	SACCPM-FD	SACCPM-IPA	SIPPavg	SIPPmax	SIPPmix	LAGavg	LAGmax	LAGmix
1	**1008**	1128 111.9%	1056 104.8%	1056 104.8%	1056 104.8%	1056 104.8%	1056 104.8%	1056 104.8%
2	**1032**	1056 102.3%	1056 102.3%	1056 102.3%	1056 102.3%	**1032** 100.0%	1056 102.3%	**1032** 100.0%
3	**3456**	**3456** 100.0%	3552 102.8%	3624 104.9%	3576 103.5%	3456 100.0%	3552 102.8%	3552 102.8%
4	**3504**	**3504** 100.0%	3552 101.4%	3624 103.4%	3576 102.1%	3576 102.1%	3576 102.1%	3528 100.7%
5	**936**	1008 107.7%	**936** 100.0%	**936** 100.0%	**936** 100.0%	**936** 100.0%	**936** 100.0%	**936** 100.0%
6	**936**	960 102.6%	**936** 100.0%	**936** 100.0%	**936** 100.0%	**936** 100.0%	**936** 100.0%	**936** 100.0%
7	**3024**	3168 104.8%	3048 100.8%	3096 102.4%	3072 101.6%	3048 100.8%	3048 100.8%	3048 100.8%
8	**2976**	3096 104.0%	3048 102.4%	3096 104.0%	3072 103.2%	3024 101.6%	3072 103.2%	3048 102.4%
9	829	908 109.5%	848 102.3%	862 104.0%	855 103.1%	**848** 102.3%	862 104.0%	855 103.1%
10	**838**	849 101.3%	848 101.2%	858 102.4%	853 101.8%	847 101.1%	862 102.9%	853 101.8%
11	2746	3015 109.8%	2786 101.5%	2838 103.4%	2812 102.4%	2787 101.5%	2838 103.4%	**2813** 102.4%
12	2786	2830 101.6%	2786 100.0%	2838 101.9%	2812 100.9%	**2778** 99.7%	2830 101.6%	2804 100.6%
13	**846**	894 105.7%	856 101.2%	862 101.9%	859 101.5%	856 101.2%	862 101.9%	859 101.5%
14	**850**	854 100.5%	854 100.5%	860 101.2%	857 100.8%	856 100.7%	861 101.3%	859 101.1%
15	**2774**	2969 107.0%	2802 101.0%	2818 101.6%	2810 101.3%	2802 101.0%	2818 101.6%	2810 101.3%
16	**2790**	2820 101.1%	2802 100.4%	2818 101.0%	2810 100.7%	2797 100.3%	2815 100.9%	2806 100.6%

Table 5.4: Cost of the solutions in Example 2. The bold numbers in each row are the lowest cost solutions out of the solutions that satisfy the feasibility requirements; see Table 5.5. The percentages are the percentage of the SACCPM cost in each experiment. The symbol "#" denotes the experiment number.

#	SACCPM-FD	SACCPM-IPA	SIPPavg	SIPPmax	SIPPmix	LAGavg	LAGmax	LAGmix
1	2 79.0%	1 79.7%	1 79.0%	1 79.0%	2 79.0%	81.3%	84.4%	85.9%
2	82.1%	82.1%	83.1%	83.1%	83.1%	82.1%	83.1%	82.1%
3	1 76.7%	80.8%	4 (**1**) 74.4%	81.6%	81.5%	82.1%	85.9%	82.5%
4	82.1%	82.1%	82.0%	84.3%	83.1%	82.0%	86.2%	82.1%
5	82.4%	85.4%	83.1%	81.3%	82.5%	81.8%	83.3%	83.1%
6	81.8%	85.2%	81.8%	81.8%	81.8%	81.8%	81.8%	81.8%
7	80.1%	1 79.3%	80.4%	82.5%	80.8%	83.5%	83.8%	83.8%
8	3 76.6%	82.7%	82.0%	82.5%	83.0%	81.6%	82.3%	81.6%
9	15 (**3**) 74.2%	7 75.3%	14 (**3**) 73.2%	3 76.2%	3 76.2%	2 76.5%	81.2%	1 79.8%
10	12 77.8%	5 78.6%	3 78.4%	80.8%	80.8%	2 79.9%	81.1%	2 79.9%
11	21 (**3**) 74.7%	3 76.5%	33 (**23**) 61.2%	29 (**4**) 71.5%	29 (**4**) 71.5%	4 78.7%	80.8%	80.7%
12	5 77.9%	4 75.4%	11 76.0%	80.8%	80.8%	1 79.7%	81.9%	80.2%
13	7 76.1%	10 (**2**) 70.9%	1 79.9%	80.5%	80.5%	1 80.0%	1 80.0%	1 80.0%
14	6 75.9%	9 75.2%	3 79.0%	1 79.8%	2 79.6%	2 79.6%	1 79.8%	2 79.6%
15	17 75.1%	3 76.5%	19 76.7%	8 78.2%	8 78.2%	3 79.0%	80.3%	2 79.0%
16	10 76.6%	4 77.9%	2 79.1%	81.1%	81.1%	1 79.7%	81.1%	81.1%

Table 5.5: Feasibility of the solutions in Example 2. Each cell has up to 3 values. The first value is the number of periods in which the fraction of calls answered immediately is less than 80%. The boldface values in parentheses are the number of periods in which the fraction of calls answered immediately is less than 75%. The percentages show the lowest fraction of calls answered immediately in any period. We do not display the first two values when they are equal to 0. The symbol "#" denotes the experiment number.

Experiment	SACCPM-FD	SACCPM-IPA
1	17	13
2	20	53
3	10	13
4	15	18
5	39	21
6	39	22
7	12	17
8	11	20
9	47	79
10	24	69
11	38	36
12	26	31
13	51	70
14	27	39
15	32	118
16	53	74

Table 5.6: Number of iterations in SACCPM-FD and SACCPM-IPA in Example 2.

CHAPTER VI

CONCLUSIONS AND DIRECTIONS FOR FUTURE RESEARCH

In this thesis we have shown that combining simulation and cutting plane methods is a promising approach for solving optimization problems in which some of the constraints can only be assessed through simulation. We developed two simulation- and subgradient-based cutting plane methods, the SKCPM and the SACCPM, for solving a call center staffing problem of minimizing cost while maintaining an acceptable level of service. In Section 6.1 we provide concluding remarks, and we identify areas for related future research in Section 6.2.

6.1 Conclusions

Although the computational requirements of our methods are large we were able to solve, or at least approximately solve, a number of moderately sized hypothetical call center staffing problems. The results of the SACCPM were especially encouraging and they show that the method can potentially be applied in real world situations with good results. Furthermore, the SKCPM and the SACCPM can be automated, so that no user intervention is required while the algorithms are running.

In the examples considered in Chapter V, the SACCPM often outperformed traditional staffing heuristics that are based on analytical queuing methods. The SAACPM is robust in the sense that it works well on a range of different problems and requires almost no assumptions on the underlying model of the call center, which gives it fur-

ther credibility over the traditional methods. Compared to the traditional queuing methods, the SACCPM does best in the presence of shift constraints. To see why, recall that the SACCPM solves *both* the problem of determining minimum staffing levels *and* the problem of computing the best assignments to cover these staffing levels, while the queuing methods compute the required staffing levels first and the shift assignments afterwards, using the staffing levels as input.

The conceptual properties of the SKCPM helped us to explore the more successful SACCPM. It appears, however, that the SKCPM cannot be applied with much confidence as a solution method for the call center staffing problem, because the concavity assumption does not hold in general. The number of iterations in the SKCPM are likely considerably fewer than in the SACCPM, so it is a better option if the constraints are indeed concave. It is also quite possible that modifications of the SKCPM would apply in other service systems than call centers, as mentioned in Section 6.2.

A difficult part of the implementation of the methods is estimating sub- or pseudogradients. Of the three methods considered in Chaper IV, the finite difference method seems to be the only method that can be used to reliably compute valid cuts. The finite difference method is, at the same time, more computationally expensive than the likelihood ratio method and infinitesimal perturbation analysis.

6.2 Future Research

There are several interesting directions for future research as listed below.

- *Different methods for computing subgradients.* An obvious drawback of the methods is the large computational effort. From the numerical experience we identified that both the simulations and solving the integer programs can be computationally expensive. The finite difference approach seems to be the only method, out of the three considered, that gives reliable subgradients. Therefore, it would be beneficial to study more efficient methods that could mimic the

performance of the finite differences. There are other methods available, such as simultaneous perturbation analysis where the idea, as it would apply in this setting, is to randomly change the number of workers in all periods, but do it differently in different replications of the simulation (Spall, 1992). Another idea would be to use a smaller sample size to compute the finite difference estimator than the sample size used to estimate the service level functions.

- *Choosing the sample size.* We did not discuss the choice of a sample size. In our implementations we naively fixed the sample size in the beginning. A perhaps more intelligent method is to start with a small sample size and systematically increase the sample size as the iterates of the algorithm get closer to an optimal solution and more accuracy is needed. In this case it may be possible to update the previous cuts similarly to the stochastic decomposition method for the two stage stochastic linear program (Higle and Sen, 1991) or consider the variable sample method in Homem-de-Mello (2003).

- *Special techniques for solving the integer programs.* In relation to the integer programs one should investigate integer programming algorithms that can utilize the special structure of the relaxed problems solved in each iteration and consider allowing approximate solutions of the IPs, especially in early stages of the algorithms. One might consider Lagrangian relaxation techniques for solving these problems, still in the context of simulation and optimization, since we are approximating the constraints. Another technique to speed up the computation of the next iterate in the continuous case is to drop cuts that become redundant (see, e.g., Mitchell, 2003, for more on this approach), and it is quite possible that the same technique could reduce the time required to solve the IPs in the later stages of the SACCPM. One could also generate some initial cuts by running simulations at the staffing levels suggested by heuristics.

- *Other optimization techniques that can be used with simulation.* It is quite possible that other optimization methods could perform well in this setting. The extended cutting plane method in Westerlund and Pörn (2002) seems to fit the framework particularly well, although some details of the implementation are unclear.

- *Convexity study of the service level function.* A more detailed study of the properties of the service level function would also be useful. It may be difficult to prove discrete concavity or discrete pseudoconcavity of the service level functions theoretically, since, as we assumed, they cannot be expressed analytically. It may be possible, however, in some special cases, using perhaps sample path arguments. Moreover, the method in Chapter III can likely be applied, with minor modifications, to verify that the discrete pseudoconcavity property is satisfied for a finite set of points.

- *The call center staffing problem.* The problems solved in this thesis are fairly simple instances of a call center staffing problem, but since almost no assumptions are made on the arrival and service processes, and simulation is used to evaluate performance, it is quite possible that the method would also apply in more complicated settings. Call abandonments, skill-based routing and prioritizing multiple customer classes are problems that call center managers commonly face and it would be interesting to incorporate those into the model and the algorithm.

- *Other service systems.* The algorithms can potentially be applied in other settings such as determining the optimal base locations of emergency vehicles, police car patrolling and even systems other than service systems, such as the optimization of the throughput of a production line.

APPENDICES

APPENDIX A

PROOFS OF SOME OF THE IPA RESULTS AND TWO IPA ALGORITHMS

In this appendix we give proofs of some IPA results that were used in the derivation of the IPA estimator using a fixed number of servers. We do not include this derivation in the thesis because we introduce notation that is not used elsewhere in the thesis, and the derivation follows closely a derivation in Glasserman (1991).

We also give algorithms to compute the unbiased estimator (4.46) for the fixed number of servers model and the biased estimator (4.53) for the varying number of servers model.

A.1 The Call Center as a GSMP

To derive an unbiased derivative estimator for (4.38) we model the call center as a generalized semi Markov process (GSMP) with speeds. The critical elements of the GSMP are events, states and clock samples. The clocks run down at different speeds for different events, and the speeds are state dependent. We define the state, s, to be a combination of the servers that are busy, the number of calls in line and the current time period, i.e., $s = (m, m^w, i)$, where $m \in \{0,1\}^q$ is a vector in which $m_j = 1$ if server j is busy and 0 and 0 otherwise for all $j \in \{1, \ldots, \zeta\}$, $m^w \in \{0, 1, 2, \ldots\}$ denotes the number calls waiting to be answered and $i \in \{1, \ldots, p\}$ is the current period. Here, ζ is the number of servers and p is the number of periods.

The only time the state of the process changes is at the occurrence of an event.

The events of the GSMP are arrivals, service completions and a new period. We denote the set of all events by $A = \{\alpha_a, \alpha_\pi, \alpha_1, \ldots, \alpha_\zeta\}$ where

$$\alpha_a = \text{arrival of a call,}$$

$$\alpha_\pi = \text{new period event and}$$

$$\alpha_j = \text{service completion at server } j, j \in \{1, \ldots, \zeta\}.$$

For every state s, there is a set $\mathcal{E}(s)$, of possible events that can occur, and we assume that $\mathcal{E}(s)$ is never empty. An event can trigger the setting of one or more new events to occur in the future. We assume that if the occurrence of an event $\alpha \in \mathcal{E}(s)$ changes the state from s to s', then $\mathcal{E}(s) \setminus \{\alpha\} \subseteq \mathcal{E}(s')$, i.e., the occurrence of an event never interrupts the occurrence of other events already scheduled to occur.

When a server completes a call, the number of calls in the system decreases by one and if there are calls waiting then that server will still be busy, otherwise it will be idle. When a call arrives, we can arbitrarily assign it to the lowest numbered server that is idle, since the servers are identical, but if all servers are busy, then the number of calls in line is incremented by 1. The new period event simply increments the period i by 1. Let $\phi(s, \alpha)$ be a mapping that gives the next state visited, given that the event α occurs in state s, according to the rules just described.

The clock sample for a particular occurrence of event is a random variable with some probability distribution. For the arrivals and service completions the probability distribution are F and G, respectively. The new period event has a deterministic distribution.

The clocks for the arrival and new period events run down at a unit speed, but the clocks for the server completions (for all servers) run down at speed μ_i in period

i. Define $\lambda_\alpha(s)$ as the speed at which the clock for event α runs down in state s, i.e.,

$$\lambda_{\alpha_a}(m, m^w, i) = 1,$$

$$\lambda_{\alpha_\pi}(m, m^w, i) = 1 \text{ and}$$

$$\lambda_{\alpha_j}(m, m^w, i) = \mu_i \text{ for all } j \in \{1, \ldots, \zeta\}.$$

A.1.1 Recursive Construction of the GSMP and Propagation of Changes in Timing of One Event to the Timing of Other Events

We now use the definitions in the previous subsection to recursively construct the GSMP. We still need some more notation. Let

$$\begin{aligned}
\tau_n &:= \text{the epoch of the } n\text{th state transition,} \\
a_n &:= \text{the } n\text{th event,} \\
Y_n &:= \text{the } n\text{th state visited,} \\
c_n &:= \text{the vector of clock readings just after the } n\text{th transition,} \\
N(\alpha, n) &:= \text{number of instances of } \alpha \text{ among } a_1, \ldots, a_n, \text{ and} \\
T(\alpha, k) &:= \text{time of the } k\text{th occurrence of } \alpha.
\end{aligned}$$

All the terms above depend on μ, but we choose not to express the dependence explicitly. Also, let, for each event $\alpha \in A$, $\{X(\alpha, j), j = 1, 2, \ldots\}$ be the sequence of clock samples. When the jth occurrence of an event is scheduled at the occurrence of the nth transition, $c_n(\alpha)$ is set equal to $X(\alpha, j)$. Further let

$$t^*(Y_n, c_n) = \min_{\alpha \in \mathcal{E}(Y_n)} \{c_n(\alpha)/\lambda_\alpha(Y_n)\}, \tag{A.1}$$

i.e., $t^*(Y_n, c_n)$ is the time until the next event. If we assume some arbitrary ordering on A then

$$\alpha^*(Y_n, c_n) = \min\{\alpha \in \mathcal{E}(Y_n) : c_n(\alpha)/\lambda_\alpha(Y_n) = t^*(Y_n, c_n)\} \tag{A.2}$$

is the next event. Let the initial state be Y_0 and let $\tau_0 = 0$. The clock vector is initialized by generating and adding the clock samples for the possible events in Y_0. Next the quantities are updated by the following recursion.

$$\begin{aligned}
\tau_{n+1} =& \tau_n + t^*(Y_n, c_n), \\
a_{n+1} =& \alpha^*(Y_n, c_n), \\
N(\alpha, n+1) =& \begin{cases} N(\alpha, n) + 1 & \text{if } \alpha = a_{n+1}, \\ N(\alpha, n) & \text{otherwise,} \end{cases} \\
Y_{n+1} =& \phi(Y_n, a_{n+1}), \\
c_{n+1}(\alpha) =& \begin{cases} c_n(\alpha) - \lambda_\alpha(Y_n) t^*(Y_n, c_n) & \text{if } \alpha \in \mathcal{E}(Y_n) \setminus \{a_{n+1}\}, \\ X(\alpha, N(\alpha, n+1) + 1) & \text{otherwise.} \end{cases}
\end{aligned} \tag{A.3}$$

A.2 Continuity and Differentiability of $\xi_k(u)$

In this section we use the GSMP construction of Section A.1 to prove Theorem 4.3. The derivation is very similar to that of a similar result in Chapter 3 of Glasserman (1991).

Lemma A.1. *(Analogous to Lemma 3.1 in Glasserman, 1991, p. 45). Let $j \in \{1, \ldots, p\}$ be arbitrary.*

1. *Every τ_n and every finite $T(\alpha, k)$ is a.s. continuous in μ_j throughout $(0, \infty)$.*
2. *At a discontinuity of Y_n, $\tau_{n+1} = \tau_n$.*

Proof:

1. Suppose first that $a_1, \ldots, a_n$ are continuous. Then it immediately follows that $Y_1, \ldots, Y_n$ and $N(\alpha, 1), \ldots, N(\alpha, n)$ for all $\alpha \in A$ are continuous. From that it follows that $c_1, \ldots, c_n$ are continuous. Then $\tau_1, \ldots, \tau_n$ are continuous.

   Now suppose that at some value of μ_j, some a_l is discontinuous, and suppose that l is the smallest index such that a_l is discontinuous. Then, since

$a_1, \ldots, a_{l-1}$ are continuous, and hence, also $Y_1, \ldots, Y_{l-1}$ and $c_1, \ldots, c_{l-1}$, we see from relations (A.2) and (A.3) that there is more than one event in $\mathcal{E}(Y_{l-1})$ such that $c_{l-1}(\alpha)/\lambda_\alpha(Y_{l-1}) = t^*(Y_{l-1}, c_{l-1})$, i.e., their clocks run out simultaneously. If, say, clocks for α and β run out simultaneously then $c_{l-1}(\alpha)/\lambda_\alpha(Y_{l-1}) = c_{l-1}(\beta)/\lambda_\beta(Y_{l-1})$ and hence $c_{l-1}(a_l)/\lambda_{a_l}(Y_{l-1})$ is continuous even if a_l is not continuous. Note that $\tau_l = \tau_{l-1} + c_{l-1}(a_l)/\lambda_{a_l}(Y_{l-1})$ and is also continuous in μ_j.

Assume for simplicity that clocks for only two events, α and β run out simultaneously. The argument for more than two clocks is the same, only repeated. If α occurs first then

$$\begin{aligned} c_l(\beta) =& c_{l-1}(\beta) - \lambda_\beta(Y_{l-1})[c_{l-1}(\alpha)/\lambda_\alpha(Y_{l-1})] \\ =& c_{l-1}(\beta) - \lambda_\beta(Y_{l-1})[c_{l-1}(\beta)/\lambda_\beta(Y_{l-1})] \\ =& 0. \end{aligned}$$

Since all events have strictly positive clocks w.p.1, and $\lambda_{\alpha'}(s) < \infty$ for all $s \in S$ and all $\alpha' \in A$, then the next event must be β. Similarly if β occurs first, then the next event must be α. In either case $\tau_l = \tau_l + c_l(a_{l+1})/\lambda_{a_{l+1}}(Y_{l-1}) = \tau_l$, since $c_l(a_{l+1}) = 0$. The order in which α and β occur has no effect on Y_{l+1} for this model. Furthermore, since any event $\alpha' \in \mathcal{E}(Y_{l+1})$ was either in $\mathcal{E}(Y_{l-1})$ or was activated by α or β then $c_{l+1}(\alpha')$ is independent of the order of α and β. For every $\alpha'' \in A$, $N(\alpha'', l+1)$ is also independent of the order of α and β. But if Y_{l+1}, c_{l+1} and $N(\cdot, l+1)$ are all independent of the order of α and β then so is the rest of the sample path. In particular, every τ_n with $n > l$ is independent of the order of α and β at μ_j. Now assume that l' is the least integer greater than $l+1$ for which $a_{l'}$ is discontinuous at μ_j, then every τ_n, $n < l'$, is continuous at μ_j. At l' we may repeat the whole argument and proceed to the next discontinuous event. Thus, we conclude that every τ_n is continuous.

The same argument shows that if $T(\alpha, k)$ is finite, it is continuous. For suppose

that a_l is the kth occurrence of α so $T(\alpha, k) = \tau_l$. As argued above, in order for a_l to jump to say β, it is necessary that $c_{l-1}(\alpha)/\lambda_\alpha(Y_{l-1}) = c_{l-1}(\beta)/\lambda_\beta(Y_{l-1})$, in which case $c_l(\alpha) = 0$ just after a_l becomes β. This implies that α is the next event, a_{l+1}, to occur, and it occurs just after a_l. This makes a_{l+1} the kth occurrence of α and $T(\alpha, k) = \tau_{l+1} = \tau_l$. The same argument holds if a_{l-1} jumps to the kth occurrence of α. Thus, changing the order of α and β does not change $T(\alpha, k)$.

2. For Y_n to be discontinuous there must be at least two events α and β in $\mathcal{E}(Y_{n-1})$ with $c_{n-1}(\alpha)/\lambda_\alpha(Y_{n-1}) = c_{n-1}(\beta)/\lambda_\beta(Y_{n-1})$. As noted above, this implies that $\tau_{n+1} = \tau_n$. □

Define the propagation process $\psi^{(i)}_{\alpha\beta}(n)\ \forall\ \alpha, \beta \in A$ that is initialized as follows

$$\psi^{(i)}_{\alpha\beta}(n) = 0 \text{ for } n < i \text{ and}$$
$$\psi^{(i)}_{\alpha\beta}(i) = \mathbf{1}\{\alpha = \beta\}.$$

For $n > i$ the process is updated according to one of the following three equations, depending on when the event β was scheduled,

$$\psi^{(i)}_{\alpha\beta}(n) = \left(1 - \frac{\lambda_\beta(Y_{n-1})}{\lambda_\beta(Y_n)}\right)\psi^{(i)}_{\alpha a_n}(n-1) + \frac{\lambda_\beta(Y_{n-1})}{\lambda_\beta(Y_n)}\psi^{(i)}_{\alpha\beta}(n-1), \tag{A.4}$$

$$\psi^{(i)}_{\alpha\beta}(n) = \psi^{(i)}_{\alpha a_n}(n-1), \tag{A.5}$$

$$\psi^{(i)}_{\alpha\beta}(n) = \psi^{(i)}_{\alpha\beta}(n-1), \tag{A.6}$$

where (A.4) applies if β is scheduled to occur in state Y_n and was scheduled before the occurrence of a_n (β is an old event). Equation (A.5) applies if β is scheduled to occur in state Y_n and was scheduled upon the occurrence of a_n (β is a new event), and (A.6) applies if β is not scheduled to occur in Y_n.

Lemma A.2. *Let $j \in \{1, \ldots, p\}$ be arbitrary. At each μ_j, τ_n is differentiable w.p.1*

and the derivative is given by

$$\frac{\partial \tau_n}{\partial \mu_j} = \sum_{i=0}^{n-1} \mathbf{1}\{Y_i(3) = j\} \sum_{l=1}^{\zeta} \left(-\frac{\tau_{i+1} - \tau_i}{\mu_j} \right) \psi^{(i)}_{\alpha_l a_n}(n), \tag{A.7}$$

where $Y_i(3)$ denotes the period component of the state vector Y_i.

Proof: (This first paragraph is analogous to the proof of Lemma 2.2 in Glasserman, 1991, p.32) For a fixed μ_j the probability of two events occurring at the same time is zero since the distribution function of the interarrival clock is continuous, the service completions depend on the interarrival times and two end of period events can obviously not occur simultaneously. Therefore, we can assume that only one event occurs at any time. Then the inequalities implicit in t^* and α^* that determine $a_1, \ldots, a_n$ and $\tau_1, \ldots, \tau_n$ are strict. Since $c_n(\alpha)/\lambda_\alpha(Y_n)$ changes smoothly with μ_j then these inequalities retain their order and remain strict in a sufficiently small neighborhood of μ_j. Therefore, τ_n is continuous in that neighborhood and also differentiable since $c_n(\alpha)/\lambda_\alpha(Y_n)$ is differentiable w.r.t. μ_j.

For (A.7) we cite Lemma 3.2 in Glasserman (1991). The conditions that need to be satisfied are

1. (Non-interruption condition). An event that is scheduled will eventually occur, i.e., an event cannot be cancelled upon the occurrence of another event.

2. The distributions of the clock samples are continuous and $P(X(\alpha, k) = 0 \; \forall \; \alpha \in A)$.

3. $0 < \lambda_\alpha(s) < \infty \; \forall \; s, \alpha$.

Condition 1 is satisfied for this model of the call center. Condition 2 is required so that the probability of two events occurring at the same time is zero. It is only satisfied for the arrival event, but the result still holds by the same argument as for the continuity and differentiability of τ_n in the first paragraph of this proof. Finally,

it is natural to assume that the service rates are positive and finite, so condition 3 is also satisfied. So, by Lemma 3.2 in Glasserman (1991)

$$\frac{d\tau_n}{d\mu_j} = \sum_{i=0}^{n-1} \sum_{\alpha \in A} \left(-\frac{\tau_{i+1} - \tau_i}{\lambda_\alpha(Y_i)} \right) \frac{d\lambda_\alpha(Y_i)}{d\mu_j} \psi^{(i)}_{\alpha a_n}(n)$$

but

$$\frac{d\lambda_\alpha(Y_i)}{d\mu_j} = \begin{cases} 1 & \text{if } \alpha \text{ is a service completion and the period in } Y_i \text{ is equal to } j, \\ 0 & \text{otherwise.} \end{cases}$$

Finally, when the period is equal to j then the speed for the service completions is equal to μ_j and hence the result. □

Theorem A.3 (same as Theorem 4.3). *Consider the problem described in Section 4.5.1 and let $\xi_k(u)$ be as defined in* (4.29)*. Then,*

1. *$\xi_k(u)$ is continuous in u on $(0, \infty)$ w.p.1.*
2. *$\xi_k(u)$ is differentiable in u w.p.1 at each $u \in (0, \infty)$.*
3. *$\xi_k(u)$ is piecewise differentiable in u over $(0, \infty)$ w.p.1.*

Proof: By the definition of $\xi_k(u)$, the rate u is the same as the rate μ_j. Therefore, in this proof $\xi_k(\mu_j) = \xi_k(u)$.

1. By Lemma A.1 each τ_n and $T(\alpha, k)$ is continuous in μ_j w.p.1.

   Next, we note that $\xi_k(\mu_j)$ is one of these events, for suppose that at time 0 there are k_0 calls in the system. Since the service discipline is first come first served, call k can begin service after $k - \zeta + k_0$ service completions (ζ is the number of servers). the function $\chi : A \to [0, 1]$ as (see also Glasserman, 1991, p. 68)

   $$\chi(a) = \begin{cases} 1 & \text{if the event } a \text{ is a service completion,} \\ 0 & \text{otherwise.} \end{cases}$$

   Also let,

   $$n_k^* = \inf\{n \geq 0 : \sum_{j=1}^{C} \chi(a_j) = k - \zeta + k_0\}, \tag{A.8}$$

where C is the number of events in the simulation, be the index of the event when a server becomes available to answer call k. Then $\tau_{n_k^*} = \xi_k(\mu_j)$. Since the times of all events, $T(\alpha, j)$, are continuous in μ_j then so is $\tau_{n_k^*}$. It is possible that events change order so n_k^* may change but $\tau_{n_k^*}$ is unchanged at the point where the events change order.

2. We can use the same argument as in Lemma A.2 and conclude that for a sufficiently small change in μ_j the events do not change order. Therefore, n_k^* is fixed. By Lemma A.2, the epoch τ_n is differentiable at $\mu_j \; \forall \; n$ w.p.1 and therefore $\xi_k(\mu_j) = \tau_{n_k^*}$ is differentiable at μ_j.

3. The time when a server becomes available $\xi_k(\mu_j)$, is differentiable except possibly when some events change order at μ_j. If two events change order at μ_j then it is because of the change in the timing of a service completion event, say a_l (end of period and arrival events do not depend on μ_j). Since the clock runs down at speed μ_j then any change in μ_j to $\mu_j + \delta\mu_j, |\delta\mu_j| > 0$, will cause a positive change in the timing of event a_l. If $|\delta\mu_j|$ is sufficiently small then no other events will change order by this change in μ_j by the argument in the first part of Lemma A.1. Therefore, the order of events is piecewise constant in μ_j for $\mu_j \in (0, \infty)$ and the result follows. □

Lemma A.4 (same as Lemma 4.6). *Let ξ_k be as defined in* (4.29). *Then*

$$\left| \frac{d\xi_k(u)}{du} \right| \leq \frac{\rho^p \zeta}{u_*} \xi_k(u)$$

where the derivative exists.

Proof: We start by developing a bound on the propagation process $\psi_{\alpha\beta}^{(i)}(n)$. Note that by the update formulae (A.4)-(A.6),

$$|\psi_{\alpha\beta}^{(i)}(n)| \leq \max\{1, \lambda_\beta(Y_{n-1})/\lambda_\beta(Y_n)\} \times \max\{|\psi_{\alpha\beta}^{(i)}(n-1)|, |\psi_{\alpha a_n}^{(i)}(n-1)|\}.$$

Furthermore, $\lambda_\beta(Y_{n-1}) = \lambda_\beta(Y_n)$ except when Y_n and Y_{n-1} denote different periods, which can only occur p times, and then, $\lambda_\beta(Y_{n-1})/\lambda_\beta(Y_n) \leq \mu^*/\mu_* = \rho$. Thus $|\psi^{(i)}_{\alpha\beta}(n)| \leq \rho^p$. Therefore,

$$\begin{aligned}
\left|\frac{d\tau_n}{d\mu_j}\right| &= \left|\sum_{i=1}^{n-1} \mathbf{1}\{Y_i(3) = j\} \sum_{l=1}^{\zeta} \left(-\frac{\tau_{i+1} - \tau_i}{\mu_j}\right) \psi^{(i)}_{\alpha_l a_n}(n)\right| \\
&\leq \sum_{i=1}^{n-1} \sum_{l=1}^{\zeta} \left|\frac{\tau_{i+1} - \tau_i}{\mu_j}\right| |\psi^{(i)}_{\alpha_l a_n}(n)| \\
&\leq \sum_{i=1}^{n-1} \sum_{l=1}^{\zeta} \frac{\tau_{i+1} - \tau_i}{\mu_*} \rho^p \\
&= \frac{\zeta \rho^p}{\mu_*} \tau_n. \qquad \text{(A.9)}
\end{aligned}$$

The derivative $d\xi_k(u)/du$ only exists at a rate $u = \mu_j$ when events do not change order with a small change in u. Then, by the same argument as in part 1 of the proof of Theorem A.3, $\xi_k(u) = \tau_{n_k^*}$, where n_k^* is fixed. Therefore, the result follows from (A.9). □

A.3 Algorithm for IPA Derivative Estimation Using a Fixed Number of Servers

Here we describe an algorithm for computing $\xi_k^d(\mu)$ and $\partial \xi_k^d(\mu)/\partial \mu_j$. We run the algorithm for each replication d. Since d is fixed in each replication we omit it in the description of the algorithm. After running the algorithm $\xi_k(\mu)$ is given by the value of $T_{k-\zeta}$ ($\xi_k = 0$ for $k \leq \zeta$) and $\partial \xi_k(\mu)/\partial \mu_j$ is given by $\nabla_{k-\zeta}(j)$. The algorithm is based on Algorithm 3.1 in (Glasserman, 1991).

0. Initialization.
   - i. $t := 0$ is the time in the simulation.
   - ii. $k := 0$ is a counter for the number of incoming calls.
   - iii. $k_q := 0$ is a counter for the number of service completions.

iv. $k_s := 0$ is a counter for the number of calls that have started service.

v. $v := V_1$ is the time until the end of the current period.

vi. $\pi := 1$ is the current period.

vii. $m := 0$ is the number of calls in the system.

viii. Generate U_1 from $F(\cdot; 0)$. The first interarrival time.

ix. $a := U_1$ is the time until the next incoming call.

x. $B := \emptyset$ is the set of busy servers.

xi. $s(j) := 0$ is the remaining work at server j for $j = 1, \ldots, \zeta$.

xii. $\delta_i(j) := 0$ is the current accumulated infinitesimal delay at server j with respect to a change in μ_i for $i = 1, \ldots, p$ and $j = 1, \ldots, \zeta$.

1. Next transition.

i. $j^* := \arg\min\{s(j) : j \in B\}$ is the index of the server with the least amount of work remaining among all the busy servers. Let $j^* := 0$ if all servers are free.

ii. $t^* := \min\{s(j^*)/\mu_\pi, a, v\}$ is the time until the next event. Let $s(0) := \infty$ to account for the case when $j^* = 0$.

iii. $t := t + t^*$.

iv. $a := a - t^*$. Update the time until the next incoming call.

v. $v := v - t^*$. Update the time until the end of the current period.

vi. $s(j) := s(j) - \mu_\pi t^*$ for all $j \in B$. Update the work remaining at each busy server.

vii. $\delta_\pi(j) := \delta_\pi(j) - t^*/\mu_\pi$ for all $j \in B$. Update the delay. Only busy servers are affected and a change in the service rate of the current period is the only change that will affect the service times.

viii. The next event is chosen as the event that minimized t^* in ii. Go to 2.a if the next event is a service completion, 2.b if the next event is an incoming call and 2.c if the next event is the end of the current period.

2.a Service completion.

i. $k_q := k_q + 1$. This is the k_qth service completion.

ii. $T_{k_q} := t$. Record the time of the k_qth service completion.

iii. $\nabla_{k_q}(i) := \delta_i(j^*)$ for $i = 1, \ldots, p$. Record the partial derivatives of the time of the k_qth service completion.

iv. If $m > \zeta$ then there are calls waiting to be answered:

a. $k_s := k_s + 1$.

b. Generate X_{k_s} from G. Generate the work required for the next call in line.

c. $s(j^*) := X_{k_s}$.

v. If $m \leq \zeta$ then all the calls in the system are currently being answered:

a. $B := B \setminus \{j^*\}$. Free the server.

b. $\delta_i(j^*) := 0$ for $i = 1, \ldots, p$. Reset the accumulated delay.

vi. $m := m - 1$.

vii. Go to 1.

2.b Incoming call.

i. $k := k + 1$. This is the kth call.

ii. $W_k = t$. Record the time of the kth call.

iii. Generate U_{k+1} from $F(\cdot; t)$. The interarrival time of the next call.

iv. $a := t + U_{k+1}$. Update the time until the next incoming call.

v. If $m < \zeta$ then there is a server available to answer the call immediately:

a. $j^* := \min\{j : j \in \{1, \ldots, \zeta\} \setminus B\}$. j^* is the index of the server assigned to this call.

b. $k_s := k_s + 1$. Generate X_{k_s} from G. $s(j^*) := X_{k_s}$. Generate the work required for the call.

c. $B := B \cup j^*$. Mark the server busy.

vi. $m := m + 1$.

vii. Go to 1.

2.c End of period.

i. If $\pi < p$ then there are more periods:

a. $\pi := \pi + 1$.

b. $v := V_\pi$.

ii. If $\pi := p$ then we have reached the end of the last period:

a. $\pi := \pi + 1$.

b. $v := V_p + x$. We run the simulation to see if any calls waiting in the system will be answered on time.

c. $\mu_\pi := \mu_p$.

iii. If $\pi > p$ then we terminate the simulation:

a. Go to 3.

iv. Go to 1.

3. Termination.

i. Stop.

Now we have computed ξ_k^d and $\partial\xi_k^d/\partial\mu_j$ (as $\nabla_k(j)$) for $k = 1, \ldots, C_d + 1$, and W_k^d for $k = 1, \ldots, C_d$ for some d, where C_d is the number of calls received in replication d. Next, we repeat this for $d = 1, \ldots, n$, where n is the number of replications. Finally, the estimator in relation (4.46) is computed by

$$\frac{\partial E[R_i(\mu; Z)]}{\partial \mu_j} \approx -\frac{1}{n} \sum_{d=1}^{n} \sum_{k=1}^{C_d+1} \left[\nabla_k^d(j)\, \mathbf{1}\{V_{i-1} + x < \xi_k^d \leq V_i + x\} \right. \\ \left. \frac{\partial F(v; W_{k-1})}{\partial v}\bigg|_{v=\xi_k^d - x - W_{k-1}} \right]$$

for $i = 1, \ldots, p$ and $j = 1, \ldots, p$.

A.4 Algorithm for IPA Derivative Estimation for a Varying Number of Servers

This algorithm is for computing the IPA derivative estimator (4.53), where the number of servers in each period $y_i, i = 1, ..., p$ can vary but the service rate μ is fixed. The algorithm is used to compute $\xi_k^d(\mu; y)$ and $\partial\xi_k(\mu; y)/\partial\mu_j$. We run the algorithm for each replication d. Since d is fixed in each replication we omit it in the description of the algorithm. The algorithm yields $\xi_k^d(\mu; y)$ and $\partial\xi_k^d(\mu; y)/\partial\mu_j$ is given by $\nabla_k(j)$.

0. Initialization.
   - i. $t := 0$ is the time in the simulation.
   - ii. $k_a := 0$ is a counter for the number of incoming calls.
   - iii. $k_s := 0$ is a counter for the number of calls that have started service.
   - iv. $v := V_1$ is the time until the end of the current period.
   - v. $\pi := 1$ is the current period.
   - vi. $m := 0$ is the number of calls in the system.
   - vii. Generate U_1 from $F(\cdot; 0)$. The first interarrival time.
   - viii. $a := U_1$ is the time until the next incoming call.
   - ix. $B := \emptyset$ is the set of busy servers.
   - x. $s(j) := 0$ is the remaining work at server j for $j = 1, \ldots, y_\pi$.

xi. $\delta_i(j) := 0$ is the current accumulated infinitesimal delay at server j with respect to a change in μ_i for $i = 1, \ldots, p$ and $j = 1, \ldots, y_\pi$.

xii. $T(j) := 0$ is the time when server j became available for service.

1. Next transition.

   i. $j^* := \arg\min\{s(j) : j \in B\}$ is the index of the server with the least amount of work remaining among all the busy servers. Let $j^* := 0$ if all servers are free.

   ii. $t^* := \min\{s(j^*)/\mu, a, v\}$ is the time until the next event. Let $s(0) := \infty$ to account for the case when $j^* = 0$.

   iii. $t := t + t^*$.

   iv. $a := a - t^*$. Update the time until the next incoming call.

   v. $v := v - t^*$. Update the time until the end of the current period.

   vi. $s(j) := s(j) - \mu t^*$ for all $j \in B$. Update the work remaining at each busy server.

   vii. $\delta_\pi(j) := \delta_\pi(j) - t^*/\mu$ for all $j \in B$. Update the delay. Only busy servers are affected and a change in the service rate of the current period is the only change that will affect the service times.

   viii. The next event is chosen as the event that minimized t^* in ii. Go to 2.a if the next event is a service completion, 2.b if the next event is an incoming call and 2.c if the next event is the end of the current period.

2.a Service completion.

   i. $T(j^*) := t$. Record the time of this service completion.

   ii. If $m > y_\pi$ then there are calls waiting to be answered:

      a. $k_s := k_s + 1$.

      b. Generate X_{k_s} from G. Generate the work required for the next call in line.

      c. $s(j^*) := X_{k_s}$.

      d. $\xi_{k_s} = T(j^*)$. Record the time when a server became available to serve call k_s.

      e. $\nabla_{k_s}(i) := \delta_i(j^*)$ for $i = 1, \ldots, p$. Record the partial derivatives for the k_sth call.

   iii. If $m \leq y_\pi$ then all the calls in the system are currently being answered:

      a. $B := B \setminus \{j^*\}$. Free the server.

   iv. $m := m - 1$.

   v. Go to 1.

2.b Incoming call.

   i. $k_a := k_a + 1$. This is the k_ath call.

ii. $W_{k_a} := t$. Record the arrival time of the k_ath call.

iii. Generate U_{k_a+1} from $F(\cdot; t)$. The interarrival time of the next call.

iv. $a := t + U_{k_a+1}$. Update the time until the next incoming call.

v. If $m < y$ then there is a server available to answer the call immediately:

a. $j^* = \min\{T(j) : j \in \{1, \ldots, y_\pi\} \setminus B\}$. j^* is the index of the server assigned to this call.

b. $k_s := k_s + 1$.

c. Generate X_{k_s} from G.

d. $s(j^*) := X_{k_s}$.

e. $B := B \cup j^*$. Mark the server busy.

f. $\xi_{k_s} := T(j^*)$.

g. $\nabla_{k_s}(i) := \delta_i(j^*)$ for $i = 1, \ldots, p$. Record the partial derivatives for the k_sth call.

h. $\delta_i(j^*) := 0$ for $i = 1, \ldots, p$. Reset the accumulated delay since the server was free.

vi. $m := m + 1$.

vii. Go to 1.

2.c End of period.

i. If $\pi < p$ then there are more periods:

a. $\pi := \pi + 1$.

b. $v := V_\pi$.

c. if $y_\pi < y_{\pi-1}$ then we need to reduce the number of servers.

i. Let l be a vector such that $l(i)$ is the index of the server with the ith most remaining work.

ii. Reorder T,B,δ and s according to l.

iii. For $r = y_\pi + 1$ to $y_{\pi-1}$. Clean up remaining service.

If $s(r)/\mu > v$ then the call extends into the next period.

$y_\pi := y_\pi + 1$. Effectively one more server in the period.

If $s(r)/\mu < v$ then

$m := m - 1$

$B := B \setminus \{r\}$

d. if $y_\pi > y_{\pi-1}$ then we need to increase the number of servers.

i. For $r = y_{\pi-1} + 1$ to y_π.

$T(r) := t$.

$\delta_i(r) := 0$ for $i = 1, \ldots, p$.

If $m > r$ then a call is waiting to be served.

$k_s := k_s + 1$.

Generate X_{k_s} from G.

$s(r) := X_{k_s}$.

$\xi_{k_s} := T(r)$.

$\nabla_{k_s}(i) := \delta_i(r)$ for $i = 1, \ldots, p$.

$B := B \cup \{r\}$.

ii. If $\pi := p$ then we have reached the end of the last period:
   a. $\pi := \pi + 1$.
   b. $v := V_p + x$. We run the simulation to see if any calls waiting in the system will be answered on time.
   c. $y_\pi := y_p$.

iii. If $\pi > p$ then we terminate the simulation:
   a. Go to 3.

iii. Go to 1.

3. Termination.
   i. While $k_s <= k_a$. Add more server available epochs.
      a. $k_s := k_s + 1$.
      b. Let $B_I := \{1, \ldots, y_\pi\} \setminus B$ be the set of idle servers.
      c. If $B_I \neq \emptyset$ then there was a server available for an incoming call.
         i. $j^* := \min\{j : j \in B_I\}$. j^* is the index of the server that first became available.
         ii. $\xi_{k_s} := T(j^*)$.
         iii. $\nabla_{k_s}(i) := \delta_i(j^*)$ for $i = 1, \ldots, p$.
         iv. $B := B \cup \{j^*\}$.
      d. else no more servers become available before time $V_p + x$.
         i. $\xi_{k_s} := \infty$ (or just any value since the derivative is 0).
         ii. $\nabla_{k_s}(i) := 0$.
   ii. Stop.

Now we have computed $\xi_k^d(\mu; y)$ and $\partial\xi_k^d(\mu; y)/\partial\mu_j$ (as $\nabla_k(j)$) for $k = 1, \ldots, C_d+1$, and W_k^d for $k = 1, \ldots, C_d$ for some d, where C_d is the number of calls received in replication d. Next, we repeat this for $d = 1, \ldots, n$, where n is the number of replications. Finally, the estimator in relation (4.53) is computed by

$$\frac{\partial E[R_i(\mu; Z)]}{\partial \mu_j} \approx -\frac{1}{n} \sum_{d=1}^{n} \sum_{k=1}^{C_d+1} \left[\nabla_k^d(j)\, \mathbf{1}\{V_{i-1} + x < \xi_k^d \leq V_i + x\} \right. \\ \left. \frac{\partial F(v; W_{k-1})}{\partial v}\bigg|_{v=\xi_k^d - x - W_{k-1}} \right]$$

for $i = 1, \ldots, p$ and $j = 1, \ldots, p$.

APPENDIX B

SOME REFERENCED THEOREMS

In this appendix we list some theorems and relations that were used to prove some of the statements in this thesis.

Theorem B.1. *The Strong Law of Large Numbers. (Billingsley, 1995, Theorem 22.1) Let $X_1, X_2, \ldots$ be independent and identically distributed random variables and $\overline{X}(n) = n^{-1}\sum_{i=1}^{n} X_i$. If $E[X_1] < \infty$, then*

$$\lim_{n\to\infty} \overline{X}(n) = E[X_1] \quad w.p.1.$$

Theorem B.2. *Lebesgue Dominated Convergence Theorem. (Billingsley, 1995, Theorem 16.4) If $|f_n| \leq g$ almost everywhere, where g is integrable, and if $\lim_{n\to\infty} f_n = f$ w.p.1, then f and f_n are integrable and*

$$\lim_{n\to\infty} \int f_n d\mu = \int f d\mu.$$

Remark 2.1. The function g is *integrable* if $\int |g| d\mu < \infty$.

Boole's inequality. (Billingsley, 1995, p. 24).

$$P\left(\cup_{k=1}^{n} A_k\right) \leq \sum_{k=1}^{n} P(A_k). \tag{B.1}$$

Theorem B.3. *Generalized Mean Value Theorem. (Dieudonné, 1960, p. 154). Let $g : \mathbb{R} \to \mathbb{R}$ be a continuous function which is differentiable on a finite interval $[a, b]$ except possibly on a set D of countably many points. Then for all x and $x + h$ in (a, b)*

$$\left|\frac{g(x+h) - g(x)}{h}\right| \leq \sup_{a\leq y\leq b, y\notin D} |g'(y)|.$$

Theorem B.4. *Radon-Nikodym Theorem. (Billingsley, 1995, Theorem 32.2). If μ and ν are σ-finite measures such that ν is absolutely continuous w.r.t. μ then there*

exists a nonnegative f, a density, such that

$$\nu(A) = \int_A f d\mu$$

for all $A \in \mathcal{F}$, where $\mathcal{F}$ is an algebra of subsets of the sample space Ω.

BIBLIOGRAPHY

O. Z. Akşin and P. T. Harker. Modeling a phone center: Analysis of a multichannel, multiresource processor shared loss system. *Management Science*, 47(2):324–336, 2001.

S. Andradóttir. Simulation optimization. In J. Banks, editor, *Handbook of Simulation*, chapter 9, pages 307–333. John Wiley & Sons, New York, 1998.

D. S. Atkinson and P. M. Vaidya. A cutting plane algorithm for convex programming that uses analytic centers. *Math. Programming*, 69(1, Ser. B):1–43, 1995. ISSN 0025-5610. Nondifferentiable and large-scale optimization (Geneva, 1992).

J. Atlason, M. A. Epelman, and S. G. Henderson. Using simulation to approximate subgradients of convex performance measures in service systems. In S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, editors, *Proceedings of the 2003 Winter Simulation Conference*, pages 1824–1832, Piscataway, NJ, 2003. IEEE.

J. Atlason, M. A. Epelman, and S. G. Henderson. Call center staffing with simulation and cutting plane methods. *Annals of Operations Research*, 127:333–358, 2004.

O. Bahn, O. du Merle, J.-L. Goffin, and J.-P. Vial. A cutting plane method from analytic centers for stochastic programming. *Math. Programming*, 69(1, Ser. B):45–73, 1995. ISSN 0025-5610. Nondifferentiable and large-scale optimization (Geneva, 1992).

M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, New York, 1993.

J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

P. Billingsley. *Probability and Measure. Third Edition.* John Wiley & Sons, New York, 1995.

J. R. Birge and F. Louveaux. *Introduction to Stochastic Programming*. Springer Series in Operations Research. Springer Verlag, New York, NY, 1997.

S. Borst, A. Mandelbaum, and M. I. Reiman. Dimensioning large call centers. *Operations Research*, 52:17–34, 2004.

I. Castillo, T. Joro, and Y. Li. Workforce scheduling with multiple objectives. Submitted, 2003.

X. Chao and C. Scott. Several results on the design of queueing systems. *Operations Research*, 48(6):965–970, 2000.

B. P. K. Chen and S. G. Henderson. Two issues in setting call centre staffing levels. *Annals of Operations Research*, 108(1):175–192, 2001.

H. Chen and B. W. Schmeiser. Stochastic root finding via retrospective approximation. *IIE Transactions*, 33(3):259–275, 2001.

J. W. Chinnick. Analyzing mathematical programs using MProbe. *Annals of Operations Research*, 104:33–48, 2001.

B. Cleveland and J. Mayben. *Call Center Management on Fast Forward*. Call Center Press, Annapolis, MD, 1997.

L. Dai, C. H. Chen, and J. R. Birge. Convergence properties of two-stage stochastic programming. *Journal of Optimization Theory and Applications*, 106(3):489–509, 2000.

G. B. Dantzig. A comment on Edie's "Traffic delays at toll booths". *Operations Research*, 2(3):339–341, 1954.

J. A. Dieudonné. *Foundations of Modern Analysis*. Academic Press, New York, NY, 1960.

O. duMerle. *Interior points and cutting planes: Development and implementation of methods for convex optimization and large scale structured linear programming. (In French)*. PhD thesis, University of Geneva, Geneva, Switzerland, 1995.

O. duMerle, J.-L. Goffin, and J.-P. Vial. On improvements to the analytic center cutting plane method. *Computational Optimization and Applications*, 11:37–52, 1998.

M. E. Dyer and L. G. Proll. On the validity of marginal analysis for allocating servers in M/M/c queues. *Management Science*, 23(9):1019–1022, 1977.

S. Elhedhli and J.-L. Goffin. The integration of an interior-point cutting plane method within a branch-and-price algorithm. *Math. Programming*, 100, Ser. A:267–294, 2003.

M. C. Fu. Optimization for simulation: Theory vs. practice. *INFORMS Journal on Computing*, 14(3):192–215, 2002.

M. C. Fu and J. Q. Hu. *Conditional Monte Carlo: Gradient Estimation and Optimization Applications*. Kluwer, Norwell, MA, 1997.

P. Glasserman. *Gradient Estimation Via Perturbation Analysis*. Kluwer, Norwell, MA, 1991.

P. Glasserman. Filtered Monte Carlo. *Mathematics of Operations Research*, 18:610–634, 1993.

P. W. Glynn. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, 33:75–84, 1990.

J.-L. Goffin, A. Haurie, and J.-P. Vial. Decomposition and nondifferentiable optimization with the projective algorithm. *Management Science*, 38(2):284–302, 1992.

J.-L. Goffin, Z.-Q. Luo, and Y. Ye. Complexity analysis of an interior cutting plane method for convex feasibility problems. *SIAM J. Optim.*, 6(3):638–652, 1996. ISSN 1052-6234.

L. V. Green, P. J. Kolesar, and J. Soares. Improving the SIPP approach for staffing service systems that have cyclic demands. *Operations Research*, 49(4):549–564, 2001.

L. V. Green, P. J. Kolesar, and J. Soares. An improved heuristic for staffing telephone call centers with limited operating hours. *Production and Operations Management*, 12(1):46–61, 2003.

D. Gross and C. M. Harris. *Fundamentals of Queueing Theory*. John Wiley & Sons, New York, NY, 1998.

K. Healy and L. W. Schruben. Retrospective simulation response optimization. In B. L. Nelson, W. D. Kelton, and G. M. Clark, editors, *Proceedings of the 1991 Winter Simulation Conference*, pages 901–906, Piscataway, NJ, 1991. IEEE.

S. G. Henderson and A. J. Mason. Rostering by iterating integer programming and simulation. In D.J. Medeiros, E.F. Watson, J.S. Carson, and M.S. Manivannan, editors, *Proceedings of the 1998 Winter Simulation Conference*, pages 677–683, Piscataway, NJ, 1998. IEEE.

J.L. Higle and S. Sen. Stochastic decomposition: An algorithm for two-stage stochastic linear programs with recourse. *Mathematics of Operations Research*, 16:650–669, 1991.

T. Homem-de-Mello. Variable-sample methods for stochastic optimization. *ACM Transactions on Modeling and Computer Simulation*, 13(2):108–133, 2003.

G. Infanger. *Planning Under Uncertainty: Solving Large-Scale Stochastic Linear Programs*. Boyd and Fraser, Danvers, MA, 1994.

A. Ingolfsson, E. Cabral, and X. Wu. Combining integer programming and the randomization method to schedule employees. Submitted, 2003.

A. Ingolfsson, M. A. Haque, and A. Umnikov. Accounting for time-varying queueing effects in workforce scheduling. *European Journal of Operational Research*, 139: 585–597, 2002.

O. B. Jennings, A. Mandelbaum, W. A. Massey, and W. Whitt. Server staffing to meet time-varying demand. *Management Science*, 42(10):1383–1394, 1996.

J.E. Kelley, Jr. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

A. J. Kleywegt, A. Shapiro, and T. Homem-de-Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2001.

P. J. Kolesar and L. V. Green. Insights on service system design from a normal approximation to Erlang's delay formula. *Production and Operations Management*, 7:282–293, 1998.

G. Koole and E. van der Sluis. Optimal shift scheduling with a global service level constraint. *IIE Transactions*, 35(11):1049–1055, 2003.

A. M. Law and W. D. Kelton. *Simulation Modeling and Analysis. 3^{rd} ed.* McGraw-Hill, Boston, MA, 2000.

P. L'Ecuyer. A unified view on the IPA, SF, and LR gradient estimation techniques. *Management Science*, 36(11):1364–1383, 1990.

P. L'Ecuyer. Note: On the interchange of derivative and expectation for likelihood ratio derivative estimators. *Management Science*, 41(4):738–748, 1995.

A. Mandelbaum. Call centers (centres): Research bibliography with abstracts. Version 5. Accessible online via `<http://ie.technion.ac.il/serveng/References/ccbib.pdf>` [accessed June 7, 2004], 2003.

A. J. Mason, D. M. Ryan, and D. M. Panton. Integrated simulation, heuristic and optimisation approaches to staff scheduling. *Operations Research*, 46:161–175, 1998.

J. E. Mitchell. Computational experience with an interior point cutting plane algorithm. *SIAM J. Optim.*, 10(4):1212–1227 (electronic), 2000.

J. E. Mitchell. Polynomial interior point cutting plane methods. *Optim. Methods Softw.*, 18(5):507–534, 2003.

M. Mori. Some bounds for queues. *Journal of the Operations Research Society of Japan*, 18(3):152–181, 1975.

S. Morito, J. Koida, T. Iwama, M. Sato, and Y. Tamura. Simulation-based constraint generation with applications to optimization of logistic system design. In P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, editors, *Proceedings*

of the 1999 Winter Simulation Conference, pages 531–536, Piscataway, NJ, 1999. IEEE.

K. Murota. *Discrete Convex Analysis*. SIAM, Philadelphia, PA, 2003.

K. G. Murty. *Linear Complementarity, Linear and Nonlinear Programming*. Heldermann Verlag, Berlin, 1988.

Y. Nesterov. Complexity estimates of some cutting plane methods based on the analytic barrier. *Math. Programming*, 69(1, Ser. B):149–176, 1995. Nondifferentiable and large-scale optimization (Geneva, 1992).

O. Peton and J.-P. Vial. A tutorial on ACCPM: User's guide for version 2.01. Working Paper. University of Geneva, 2001.

S. M. Robinson. Analysis of sample-path optimization. *Mathematics of Operations Research*, 21(3):513–528, 1996.

R. T. Rockafellar. *Convex Analysis*. Princeton University Press, Princeton, New Jersey, 1970.

S. M. Ross. *Stochastic Processes. Second Edition.* John Wiley & Sons, New York, NY, 1996.

R. Y. Rubenstein and A. Shapiro. *Discrete Event Systems: Sensitivity Analysis and Stochastic Optimization by the Score Function Method*. John Wiley & Sons, Chichester, England, 1993.

S. Y. See and A. Seidel. Computational study of optimization methods for call center staffing using infinitesimal perturbation analysis. Research Report. School of ORIE, Cornell University, 2003.

A. Shapiro. Monte Carlo sampling methods. In A. Ruszczynski and A. Shapiro, editors, *Stochastic Programming*, Handbooks in Operations Research and Management Science. Elsevier, 2003. To appear.

J. C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992.

G. M. Thompson. Labor staffing and scheduling models for controlling service levels. *Naval Research Logistics*, 44(8):719–740, 1997.

D. M. Topkis. *Supermodularity and Complementarity*. Princeton University Press, Princeton, New Jersey, 1998.

R. M. van Slyke and R. Wets. L-shaped linear programs with applications to optimal control and stochastic programming. *SIAM Journal on Applied Mathematics*, 17 (4):638–663, 1969.

S. Venit and W. Bishop. *Elementary Linear Algebra*. PWS-Kent Publishing Company, Boston, MA, 1989.

S. Vogel. Stability results for stochastic programming problems. *Optimization*, 19(2): 269–288, 1988.

S. Vogel. A stochastic approach to stability in stochastic programming. *Journal of Computational and Applied Mathematics*, 56:65–96, 1994.

T. Westerlund and R. Pörn. Solving pseudo-convex mixed integer optimization problems by cutting plane techniques. *Optimization and Engineering*, 3:253–280, 2002.

W. Whitt. The pointwise stationary approximation for $M_t/M_t/s$ queues is asymptotically correct as the rates increase. *Management Science*, 37:307–314, 1991.

ABSTRACT

SIMULATION-BASED CUTTING PLANE METHODS FOR OPTIMIZATION OF SERVICE SYSTEMS

by

Júlíus Atlason

Co-Chairs: Marina A. Epelman and Shane G. Henderson

Simulation is a powerful tool for analyzing a complex system. When decisions need to be made about the operating policies and settings of a system, some form of optimization is required. In this dissertation we develop two iterative subgradient based cutting plane methods for solving resource allocation problems in service systems, when the objective function and or the constraints are evaluated via simulation.

This work is motivated by the call center staffing problem of minimizing cost while maintaining an acceptable level of service over multiple time periods. An analytical expression of the expected service level function in each period is typically not available. Instead, we formulate a sample average approximation (SAA) of the staffing problem. A proof of convergence is given for conditions under which the solutions of the SAA converge to the solutions of the original problem as the sample size increases. In addition, we prove that this occurs at an exponential rate with increasing sample size.

In some cases it is reasonable to assume that the expected service level functions are concave in the number of workers assigned in each period. In such cases, we show how Kelley's cutting plane method can be applied to solve the SAA. Empirical results suggest, however, that the expected service level function is approximately

pseudoconcave. In that case, we develop the simulation-based analytic center cutting plane method (SACCPM). Proofs of converge for both methods are included.

Our cutting plane methods use subgradient information to iteratively add constraints that are violated by non-optimal solutions. Computing the subgradients is a particularly challenging problem. We suggest and compare three existing techniques for computing gradients via simulation: the finite difference method, the likelihood ratio method, and infinitesimal perturbation analysis. We show how these techniques can be applied to approximate the subgradients, even when the variables, i.e., number of workers, are discrete.

Finally, we include numerical implementations of the methods and an extensive numerical study that suggests that the SACCPM usually does as well and often outperforms traditional queuing methods for staffing call centers in a variety of settings.