Efficient Controls for Finitely Convergent Sequential Algorithms and Their Applications

by

Wei Chen

A dissertation submitted to the Graduate Faculty in Computer Science
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy, The City University of New York

2010

This manuscript has been read and accepted for the
Graduate Faculty in Computer Science in satisfaction of the
dissertation requirements for the degree of Doctor of Philosophy.

Date — Dr. Gabor T. Herman, Chair of Examining Committee

Date — Dr. Theodore Brown, Executive Officer

Dr. Amotz Bar-Noy

Dr. Thomas R. Bortfeld

Dr. Robert M. Haralick
Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK

Abstract

Efficient Controls for Finitely Convergent Sequential Algorithms and Their Applications

by

Wei Chen

Advisor: Gabor T. Herman

Finding a feasible point that satisfies a set of constraints or a point that optimizes a function subject to a set of constraints is a common task in scientific computing: examples are the linear/convex feasibility/optimization problems. Projection methods have been used widely in solving many such problems of large size much more efficiently than other alternatives. Finitely convergent sequential algorithms are projection methods that sequentially iterate on individual constraints, one at a time, but overall find a feasible point in a finite number of iterations. ART3 is an example using cyclic control in the sense that it repeatedly cycles through the given constraints. The skipping unnecessary checks on constraints that are likely to be satisfied, lead to the new algorithm ART3+, a variant of ART3 whose control is no longer cyclic, but which is still finitely convergent. Experiments in fitting pixel images by blob images show that ART3+ is statistically significantly faster than ART3. Furthermore, a general methodology is proposed for automatic transformation of any finitely convergent sequential algorithm in such a way that (1) finite convergence is retained and (2) the speed of finite convergence is improved. The first property is proved by mathematical theorems, the second is illustrated by applying the algorithms to practical problems. This transformation is applicable, for

example, to the finitely convergent modified cyclic subgradient projection algorithm for solving convex feasibility problems.

One application is to intensity modulated radiation therapy (IMRT), whose goal is to deliver sufficient doses to tumors to kill them, but without causing irreparable damage to critical organs. The superior performance of the ART3+ for IMRT is demonstrated. An optimization algorithm based on ART3+ is proposed to handle linear constraints with either linear objectives or certain convex objectives. It is on average more than two orders of magnitude faster than the state of art industry-standard commercial algorithms (MOSEK's interior point optimizer and primal/dual simplex optimizer) and has almost no memory overhead. This allows for fast creation of multi-criteria treatment plan databases that span the global planning options available to the treatment planner.

To my parents.

Acknowledgments

I am indebted to many people for help toward the completion of my Ph.D. study. It is difficult to imagine that I could have finished this journey without their guidance and support. Instead, because of their generous help, I found this journey full of surprises and joys.

I own my greatest gratitude to my advisor, Professor Gabor Herman, for his patient guidance all through my Ph.D. study. This work benefited greatly from his insightful suggestions on computational methods. His down-to-earth research attitude and step-by-step approach impressed me deeply. His devotion to scientific research set an excellent paradigm for me to follow. Moreover, he is a great leader of our Discrete Imaging and Graphics (DIG) Group, a knowledgeable teacher and a tireless person. Particularly, he patiently read through every sentence of all the research papers I have written, gave me many valuable suggestions in paper writing, and helped me correct numerous English mistakes throughout my Ph.D. study. This dissertation would not have been possible without the help from him.

My gratitude also goes to many other professors for various reasons. I would like to thank Professor Yair Censor for his great suggestions and stimulating discussions on my research work. Professors Amotz Bar-Noy, Robert Haralick, János Pach have been in the committee of my thesis proposal and defense. I appreciate their valuable and constructive advice. I also thank Professor Ted Brown for his generous help on all the aspects of my Ph.D. study. Moreover, I am deeply indebted to Professor Peng Zhang back in China who initially guided me to the area of image reconstruction and mathematical modeling.

At the Massachusetts General Hospital, I would like to thank Professors Thomas Bortfeld, David Craft and Hanna Kooy for providing me the wonderful opportunity to work with challenging practical problems and clinical data and setting up the stage for my continuing research work.

I also wish to thank my wonderful team members of the DIG group for their friendship and help. They are Stuart Rowland, Ivan Kazantsev, Hstau Liao, Eilat Vardi-Gonen, Ran Davidi, Joanna Klukowska, Lucas Oliveira, etc. Thanks are due also to many of my friends at CUNY Graduate Center: Yi Feng, Yuqing Tang, Wenbo Cao, Jiang Wu. The warm discussions between us on countless research/non-research topics are part of my enjoyable memory.

My Ph.D. study is mainly supported by NIH Grant R01HL070472, NSF Grant DMS0306215, NCI Grant R01CA103904-01A1 and Science Fellowship from CUNY Graduate Center. I deeply appreciate the scholarship providers for giving me this precious opportunity of top quality education and research.

Finally I would like to express my loving thanks to my parents. It is their endless love, support and encouragement that stimulate me to walk through the twenty-two years of education and to advance further. I dedicate this dissertation to them.

Contents

1 Introduction **1**

1.1 Feasibility and Optimization Problems 1

1.2 Projection Methods 2

1.3 The Control of a Sequential Algorithm 4

1.4 Computerized Tomography 5

1.5 Intensity Modulated Radiation Therapy Planning 11

2 Methods **17**

2.1 Definitions and Problem Statement 17

2.2 Illustrative Examples of Finitely Convergent Sequential Algorithm . 24

2.2.1 ART3 and Its Generalization 24

2.2.2 MCSP and Its Generalization 27

2.3 A Basic Idea for Improving Efficiency 29

2.4 Improving the Efficiency of Finitely Convergent Algorithms 32

2.5 Methods for Constrained Optimization Problems 43

3 Experiments **46**

3.1 Experiments with Fitting a Pixel Image by a Blob Image 46

3.1.1 The Brain Phantom 49

3.1.2 Task-Oriented Comparisons of Algorithm Performance . . . 51

3.2 Experiments with IMRT Planning 53

3.2.1 Experiments where ART3+ is More Efficient than ART3 . . 54
3.2.2 Experiments where ART3+ is More Efficient than Linear Programming 61
3.2.3 Experiments where ART3+O is More Efficient than MOSEK's Interior Point Method and Simplex Methods 69

4 Conclusions **80**
4.1 Contributions 80
4.2 Future Works 82
4.2.1 Finitely Convergent Sequential Algorithms with Perturbations 82
4.2.2 CMCSP++ is Faster than CMCSP in IMRT Applications . . 83
4.2.3 Non-finitely Convergent Sequential Algorithms 83
4.2.4 A Novel Linear Programming Solver 84
4.2.5 Miscellaneous 84

Bibliography **86**

List of Tables

3.1 The average timings of the 30 runs. 53

3.2 Timing for finding solutions with the various algorithms. 61

3.3 Timing with decreasing upper bounds with the various algorithms. . 61

3.4 The prescription for the subvolumes in the head and neck phantom. . 66

3.5 Prescriptions for the subvolumes in the prostate phantom. DVCs are applied to the bladder in the upper table and to the rectum to the lower table. 68

3.6 Timings (in seconds) for finding solutions with the two algorithms. . 69

3.7 An example run of ART3+O with Task6 of the pancreas case. . . . 71

List of Figures

1.1 Image reconstruction by ART when $M \times N = 223{,}744 \times 51{,}152$. . . 7

1.2 Displays of a 243×243 digitized phantom (a) and of an ART reconstruction when $M \times N = 223{,}744 \times 51{,}152$ (b). 8

1.3 Image reconstruction by ART and the interior point method in MOSEK when $M \times N = 24{,}880 \times 5{,}711$. 10

1.4 Displays of an 81×81 digitized phantom (a) and of an ART reconstruction when $M \times N = 24{,}880 \times 5{,}711$ (b). 11

1.5 IMPT MCO planning system showing the esophagus case. The left panel displays the constraints and the objectives and shows the navigation sliders in positions for the best lung plan. The DVH displays and iso-dose lines are updated in real-time while the user navigates. Here we show the DVHs for the best lung plan and the DVHs for the best heart plan, which together display the range of planning options regarding these two critical structures. (Patient data courtesy of Dr. T. Hong.) 15

1.6 Two plans shown from the lung / chest-wall target case. The plans use three fields: anterior, posterior and posterior oblique. The left plan forces complete target volume coverage to be near to its maximum allowed value of 70 Gy (the purple DVH), while the right plan achieves maximal lung sparing (the green DVH). (Patient data courtesy of Dr. B. Eden.) 15

2.1 The iterative step of GART3 from x^k to x^{k+1}, i.e., to $p_g\left(x^k\right)$. 26

3.1 Row 1 contains a phantom and its windowed image. Rows 2, 3 and 4 contain the blob images and their corresponding windowed images provided for the phantom by CART3, CART3$^+$ and CART3^{++}, respectively. The differences between the blob images produced by the three algorithms are so small that they cannot be identified based on such displays. 50

3.2 The first row contains windowed images of blob images provided by CART3 and by CART3$^+$ for a phantom different from that of Figure 2. The second and third rows, respectively, show (dark) plots of the 131th columns of the blob images provided by CART3 and CART3$^+$ against that of the phantom (light). The differences between the blob images produced by CART3 and CART3$^+$ are so small that they cannot be identified based on such displays. 52

3.3 The indicated circle encloses the 2D cross-section of the patient's body. The grid defines a subdivision into voxels j, for $1 \leq j \leq J$, these voxels are indicated by the heavy edges. The gray bar indicates the extent of beamlet n, $1 \leq n \leq N$. The angle θ is between the direction of the beamlet and a fixed direction (in our case the positive horizontal axis). 55

3.4 The first experiment. 58
3.5 The second experiment. 59
3.6 The third experiment. 60
3.7 A typical curve of $\alpha[v]$ plotted against $\beta[v]$. 65
3.8 Searching for solution with $\alpha = 0.2, \beta = 0.4$. For $\beta[v] = 0.1$, ART3+ failed to converge. The search order for $\beta[v]$ is 0.4, 0.2, 0.1, 0.3, 0.15. 65
3.9 The head and neck phantom (d=1: bright moon-shaped region, a PTV; d=2: small disk on the left, a PTV; d=3: small disk on the right, an OAR; d=4: rest of the voxels within the body, normal tissue.) 67
3.10 The prostate phantom (d=1: center disk, a PTV; d=2: bright moon-shaped region, an OAR; d=3: bottom disk, an OAR; d=4: side disks, two OARs; d=5: rest of the voxels within the body, normal tissue.) 68
3.11 Timing of the four methods for the feasibility run and the eight optimization tasks. ART3+O is two to three orders of magnitude faster given the vertical time axes is with logarithmic scale. 72
3.12 Memory usage of the four algorithms for the feasibility run and the eight optimization tasks. ART3+O uses less than a tenth of the memory needed by other algorithms. 73
3.13 The vertical line segments are the initial ranges of r. The thickness of the horizontal segments is the gap between the optimal r found by ART3+O and the optimal r given by MOSEK. 74
3.14 The timing of ART3 and ART3+ in the last search of the feasibility task and all eight other tasks. 75

3.15 The times ART3+ needs to run for the feasibility problems in IMPT planning (without any objective function) when the lower bound of the PTV increases from 10 Gy to its maximum possible value. . . . 78

Nomenclature

$\mathbb{G}$	a problem set
$\mathbb{R}^N$	N-dimensional Euclidean space
$\mathbb{S}$	states set
$\mathbb{Z}_>$	positive integers
$\mathbb{Z}_\geq$	non-negative integers
$\sigma()$	choice function
A	system matrix or dose-influence matrix
$b_{a,\alpha,\delta}(r,\phi)$	blob basis function
$c(P)$	control sequence of path P
$d(P)$	state sequence of path P
K	maximum number of iterations executed in ART3+O
k	iteration number
l	lower bound vector
M	number of constraints, or projections
N	length of the decision vector x

P	an execution path
p_g	a projection function associated with function g
Q_g	the feasible set associated with function g
r_{max}	upper bound on the current search region
r_{min}	lower bound on the current search region
$s(P)$	solution sequence of path P
T	number of objectives in MCO
u	upper bound vector
x	decision vector, beamlet intensity vector or blob coefficient vector

Chapter 1

Introduction

1.1 Feasibility and Optimization Problems

Feasibility problems aim at finding a feasible point that satisfies a set of constraints. Convex feasibility problems and linear feasibility problems are two important subsets of feasibility problems whose constraints are convex and linear, respectively. Optimization problems that have an objective function along with (or without) a set of constraints are called constrained (or unconstrained) optimization problems. A general form of the convex feasibility problem is that of finding a point in the intersection of a family of closed convex sets in a Hilbert space. In such formulations, each set can be specified in various forms, e.g., as the fixed point set of a nonexpansive operator, the set of zeros of a maximal monotone operator, the set of solutions to a convex inequality, or the set of solutions to an equilibrium problem. Convex optimization problems are optimization problems with both the objective function and the constraints being convex. The convex feasibility problem and the convex optimization problem are at the core of the modeling of many problems in various areas of mathematics, physical sciences and computer science; see [21, 22] and references therein. Over the past four decades, they have been used to model sig-

nificant real-world problems in sensor networks [8], in radiation therapy treatment planning [12, 37], in resolution enhancement [15], in wavelet-based denoising [19], in antenna design [34], in computerized tomography [36], in materials science [41], in watermarking [43], in data compression [45], in demosaicking [47], in magnetic resonance imaging [57], in holography [58], in color imaging [59], in optics and neural networks [60], in graph matching [63] and in adaptive filtering [66], to name but a few. In these – and numerous other – problems, projection methods have been used to solve the underlying convex feasibility or optimization problems.

We focus on the important subclass of convex feasibility and optimization problems: linear feasibility problems and linearly constrained optimization problems, in which finitely many constraint sets are given and each of them is specified by a linear equality or inequality in the Euclidean space. For such problems, which arise in many important applications [21, 36, 37], alternatives to projection methods are available (see, e.g., [2, 33] and the references therein). Extension to general convex feasibility and optimization problems is discussed in the future works in Section 4.2.

1.2 Projection Methods

Projection methods were used to solve systems of linear equations in Euclidean spaces in the 1930s [20, 40] and were subsequently extended to systems of linear inequalities in [1, 49, 52]. The basic step in these early algorithms consists of a projection onto an affine subspace or a half-space. Modern projection methods are much more sophisticated [3, 4, 5, 7, 10, 14, 23, 24, 25, 30, 31, 42] and they can solve the general convex feasibility problem. Projection methods can have various algorithmic structures (some of which are particularly suitable for paral-

lel computing) and they also possess desirable convergence properties and good initial behavior patterns [4, 14, 22, 23, 24, 36, 56]. The main advantage of projection methods, which makes them successful in real-world applications, is computational. They commonly have the ability to handle huge-size problems of dimensions beyond which more sophisticated methods cease to be efficient. This is so because the building bricks of a projection algorithm are the projections onto the given individual sets, which are assumed to be easy to perform, and because the algorithmic structure is either sequential or simultaneous, or in-between, as in the block-iterative projection methods or in the more recently invented string-averaging projection methods. The number of sets used simultaneously in each iteration in block-iterative methods and the number and lengths of strings used in each iteration in string-averaging methods are variable, which provides great flexibility in matching the implementation of the algorithm with the parallel architecture at hand. Due to the structure of projection methods, only a very limited amount of memory besides the data from the problem is needed, while some other methods need working memory space whose size can be several times the data size. Therefore alternative methods may not be even applicable for for huge-size problems due to memory requirements. An example is given in Section 1.4.

Among all the projection methods that converge to a solution to the problem, some are finitely convergent, meaning that it is guaranteed to converge to a feasible point in a finite number of iterations, as opposed to in the limit. This dissertation focuses on finitely convergent sequential algorithms, but extension to general convergent sequential algorithms is discussed in the future works in Section 4.2.

Algebraic reconstruction techniques (ART) is a large family of projection methods for solving linear feasibility and optimization problems. The projection algorithm used in this dissertation, ART3+, is a finitely convergent sequential projection algorithm [16, 37]. ART3 [35] is a variant of ART for solving linear interval in-

equalities, and ART3+ [37] is a faster version of ART3, with the significant speed improvement stemming from a repetitive (non-cyclic) control for selecting the constraints. In each iteration of either ART3 or ART3+, a constraint is selected and checked. If the constraint is violated, then the current iterate is projected into or onto the constraint, making the point feasible for that constraint. The difference between the algorithms is in their behavior if the selected constraint is not violated: in ART3+ such a constraint is temporarily removed from the list of constraints, and thus computer time is saved by not selecting and checking constraints that are not likely to be violated; see [16, 37]. Both ART3 and ART3+ are finitely convergent provided that the feasible solution set is full-dimensional. Due to the speed of the ART3+ algorithm on some feasibility problem instances, it is efficacious to make use of it also for optimization. For example, to minimize the inner product $\langle v,x\rangle$ of x with a given vector v, we convert the objective into a constraint $\langle v,x\rangle \leq r$ and iteratively reduce r until convergence to a solution within a specified optimality tolerance. We call the resulting optimization algorithm ART3+O. The details of this new algorithm are given in Section 2.5 and experiments using it are reported in Subsection 3.2.3.

In Section 1.4 we demonstrate the superiority of a variant of ART over a variant of the interior point method, for both computational time and memory usage, by applying both methods to an image reconstruction problem in computerized tomography.

1.3 The Control of a Sequential Algorithm

For sequential projection algorithms that only treat one constraint in each iteration, the control sequence (or just control) that determines the order in which the constraints are chosen for the execution of the iterative algorithm can "have a sig-

nificant effect on the practical performance of the algorithm" [36]. Some important controls are the cyclic control, the almost cyclic control, the repetitive control, the remotest set control, the approximately remotest set control, the most violated constraint control, and the random control, see Section 2.1 and [14] for their definitions. Some of these controls are predetermined sequences of the indices of the constraints while some others depend on the problem instance, e.g., most violated constraint control chooses the constraint that is most violated by the current iterate.

In this dissertation we propose two automatic transformations of finitely convergent sequential algorithms, by which finitely convergent sequential algorithms using simple cyclic control can be adapted to their more efficient versions. Although the non-cyclic controls are determined on the fly by the iterative steps of the algorithm applied to a specific problem instance, they are proved to be almost cyclic or repetitive. The improved sequential algorithms can be proved to converge finitely. Their efficiency in computational time is demonstrated by experiments in image processing and radiation therapy planning.

1.4 Computerized Tomography

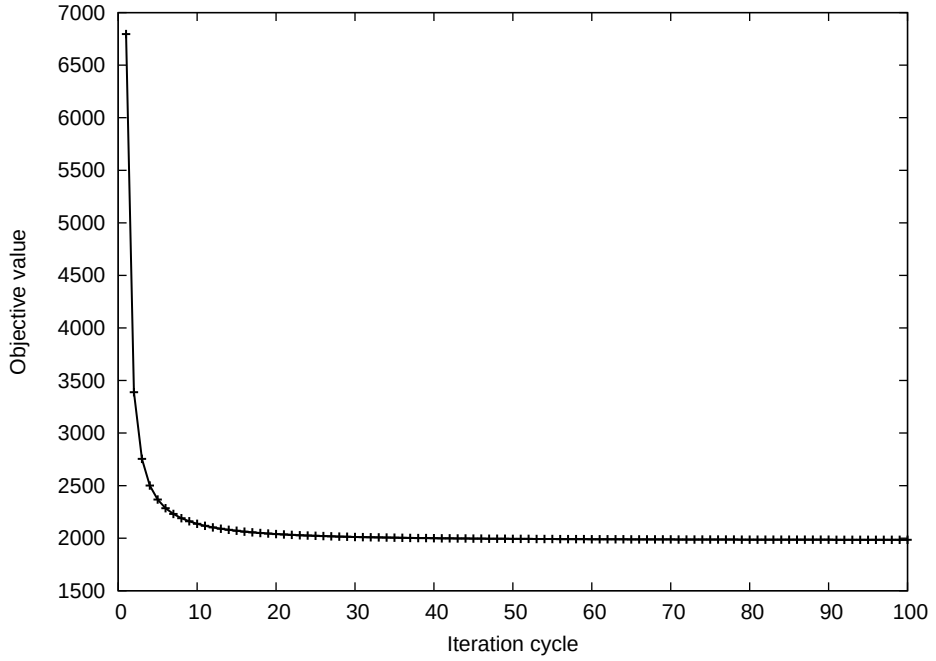
Computerized tomography (CT) is the problem of recovering an image from its measured (and hence not strictly accurate) integrals along a number of lines [36]. If we assume that the recovered image will be represented as a linear combination of a number of basis functions, then the task is to find the vector x whose components are the weights to be given to the basis functions. Due to the linearity of integration and based on the knowledge of the basis functions, we can produce a matrix A such that Ax is approximately the vector b of measurements. Since it is not likely that there is an x such that $Ax = b$, it is reasonable to aim instead, for example, at finding an x that minimizes a weighted sum of the squared norm of the residual $Ax - b$

and the squared norm of x. This unconstrained quadratic optimization problem can be transformed into a linearly constrained norm minimization problem that can be solve by ART, as well as by other quadratic optimization methods.

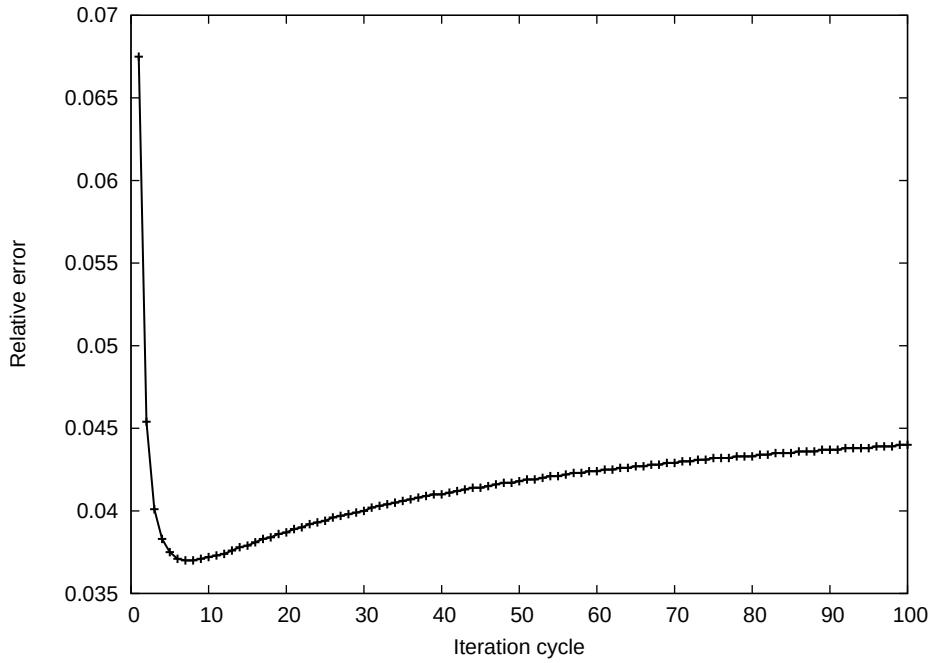
Recognizing that in one iterative step only one row of the matrix is needed and that in CT most entries of each row are zero, we see that an iterative step can be carried out very rapidly, provided that we have access to the locations and the values of the nonzero entries. If the memory of the computer is large enough, this can be accommodated by storing A in a row-by-row sparse representation, otherwise the locations and values of the nonzero entries can be generated within each iterative step by some rapid mechanism, such as the digital difference analyzer explained, e.g., in Section 4.6 of [36].

In Section 5.8 of [36] there is an exact specification of the so-called standard projection data that are used to evaluate various reconstruction algorithms in that book, the number of lines used in the standard projection data is $M = 223,744$. In the evaluations based on the standard projection data that are reported in [36] for reconstruction algorithms that use blob basis functions, the number of blobs used is $N = 51,152$.

In the first experiment we applied the ART algorithm to the standard projection data. In Figure 1.1(a) we show the behavior of the objective function as a function of iteration cycles (an iteration cycle is defined to be M iterations). It can be observed that the initial decrease in the objective function is very rapid. This desirable initial behavior is even more noticeable when we evaluate the algorithm not from the purely mathematical point of view of how well the objective function is reduced, but rather from the application point of view of how good are the reconstructed images. For this purpose, we report on the normalized mean absolute picture distance measure, as defined in [36]. In Figure 1.1(b) we plot the distance measure for this experiment. It is seen that its minimum is reached at the seventh iteration cycle, i.e.,

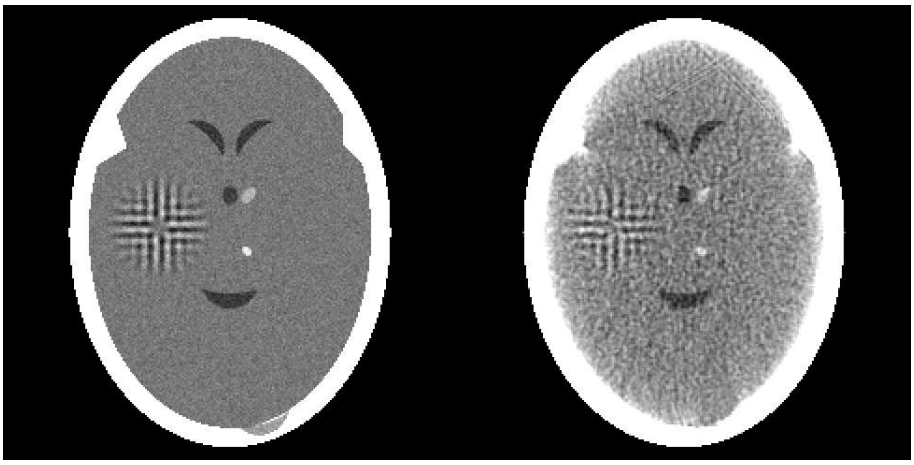


(a) Plot of the objective function.



(b) Plot of the distance measure.

Figure 1.1: Image reconstruction by ART when $M \times N = 223{,}744 \times 51{,}152$.



(a) (b)

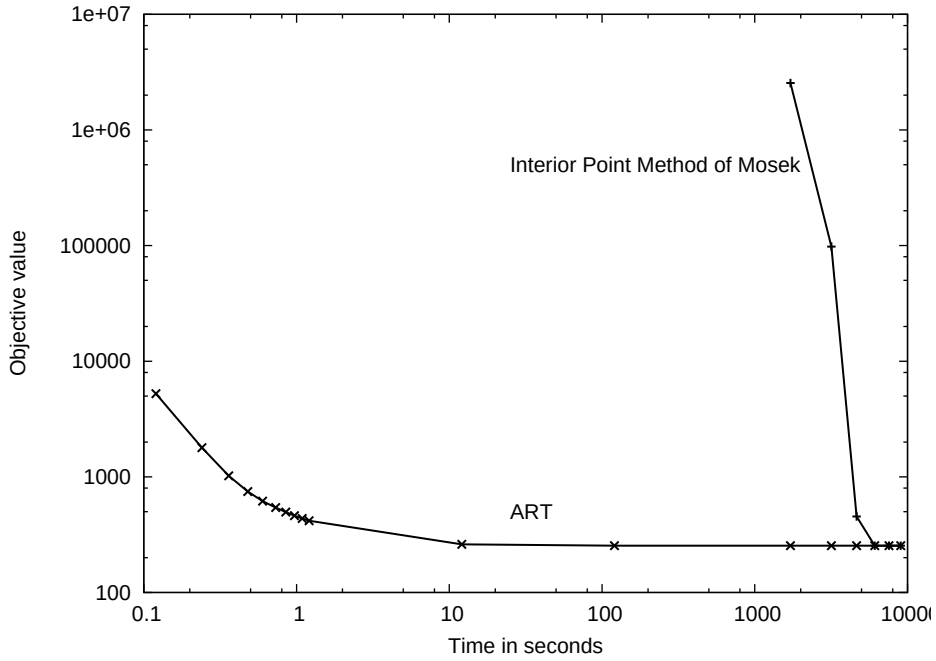
Figure 1.2: Displays of a 243×243 digitized phantom (a) and of an ART reconstruction when $M \times N = 223{,}744 \times 51{,}152$ (b).

when the iteration number $k = 7M$. This reflects the fact that the minimization objective does not (and, in fact, it cannot in real applications where the phantom is not known to us) fully describe the application objective. For this reason it is standard practice in tomography [36] to stop the iterative process after a few iteration cycles and use the result at that time as the reconstruction. A digitization for the phantom we used is shown in Figure 1.2(a). The digitization obtained from x^{7M} produced by this experiment is shown in Figure 1.2(b). The reconstruction is not perfect (as indeed it cannot possibly be since the measured data are only approximations of the line integrals assumed by the mathematics), but important features of the phantom are identifiable in the reconstruction. This ART reconstruction was carried out in 38.4 seconds on an Intel Core 1.6 GHz processor, 2 Gbyte memory, 32 bit laptop.

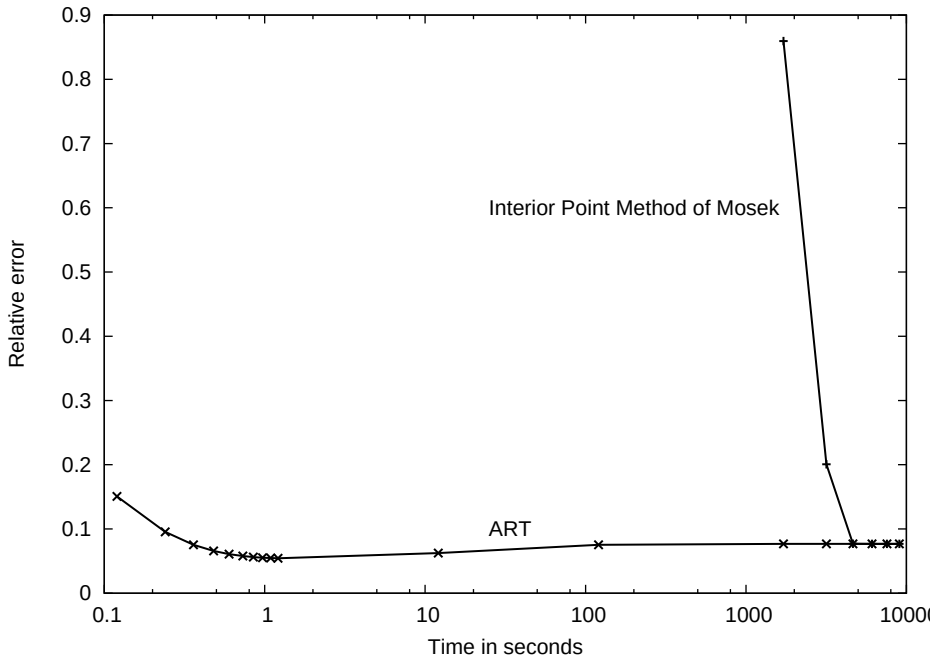
We wanted to compare the time needed by ART with the time needed to solve the system of consistent equations for the same data by the current implementation of the interior point method of MOSEK version 5 [2]. Unfortunately this could not be done, because the memory requirements of the MOSEK software were too large for our laptop. So we attempted to use a much more powerful Intel Xeon

2.66 GHz processor, 16 Gbyte memory, 64 bit workstation, but even the 16 Gbyte memory was too small to handle this problem using the MOSEK software. The importance of this memory requirement issue cannot be overemphasized: problems that routinely arise in real applications can be handled by projection methods using inexpensive laptops, while "more sophisticated alternatives" fail to produce any results even on much more powerful workstations due to their much greater demands on computer memory.

In order to be able to compare the efficiency of ART with that of the interior point method in MOSEK we had to reduce M and N to about a ninth of their previous sizes. Thus, in the second experiment on which we now report $M \times N = 24,880 \times 5,711$. For this smaller example we ran both ART and the interior point method in MOSEK (with its default parameters) on an Intel Xeon 2.66 GHz processor, 16 Gbyte memory, 64 bit workstation. In Figure 1.3 we plot both the objective function and the normalized mean absolute picture distance measure for both algorithms as a function of time. From the point of view of the objective function, MOSEK needed over 5000 seconds to reach a value as low as ART reached in 10 seconds. The advantage of ART is more pronounced when considering the picture distance measure: the optimal value is reached by ART at 1.7 seconds (when $k = 14M$) while the interior point method never reaches a distance value that is as low as that of ART and it needs approximately $5,000$ seconds to reach its lowest distance measure.

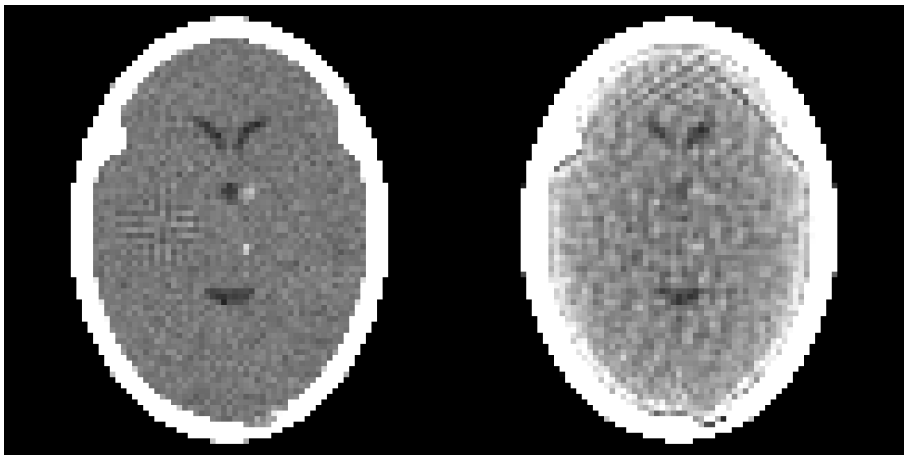


(a) Plot of the objective function.



(b) Plot of the distance measure.

Figure 1.3: Image reconstruction by ART and the interior point method in MOSEK when $M \times N = 24{,}880 \times 5{,}711$.



(a) (b)

Figure 1.4: Displays of an 81×81 digitized phantom (a) and of an ART reconstruction when $M \times N = 24{,}880 \times 5{,}711$ (b).

Since both M and N are about a ninth of their previous sizes, we report in Figure 1.4 on the 81×81 digitizations of the phantom and of the reconstruction x^{14M}. These are clearly inferior to the images in Figure 1.2, demonstrating the medical necessity for the larger system of equations.

1.5 Intensity Modulated Radiation Therapy Planning

Intensity modulated radiation therapy (IMRT) is a tool for treating cancer [54]. The goal of IMRT is to deliver a sufficiently high dose to the planning target volumes (PTVs), such as cancerous tumors, but to avoid depositing a harmfully high dose to organs at risk (OARs). IMRT uses a multi-leaf collimator (MLC) that allows the treatment planner to control the intensity of radiation within relatively thin beamlets. This requirement can be formulated as a linear feasibility problem $l \leq Ax \leq u$. The interpretation in this application is that each component of x is a to-be-determined strength of radiation to be delivered to the patient in N separate beamlets, the components of Ax are the resulting doses at M points in the patient's body, and l and u are provided by the radiation oncologist as the desired limits on

these doses. Projection algorithms have been successfully used in IMRT planning [12, 17, 18, 37, 64].

Intensity modulated proton therapy (IMPT) delivers proton beams to treat cancer instead of using photon beams in traditional IMRT, to get more conformal dose distribution on the targets and better sparing of the critical organs. Structurally, the mathematical problems of IMRT and IMPT planning are the same. Both require large-scale optimizers to choose the optimal set of beamlet intensities that map to voxel doses. An IMPT optimization problem has, however, on the order of 3-10 times more decision variables (beamlets) than the corresponding IMRT optimization problem, depending on the pencil-beam size and the number of energy layers delivered. In clinical applications, it is considered desirable to find multiple feasible points, each of which is optimal according to its own criterion. A typical optimization task is "find a feasible point that results in the smallest total dose delivered to the liver." The associated functional is a linear one: it is the sum of those components of Ax that are associated with points in the liver. The standard single criterion approach to treatment planning uses weights and dose volume histogram (DVH) control points to steer towards desired plans. The results of the underlying optimizations are, however, hard to control and often result in time-consuming iterations between the treatment planner, the treatment planning system and the physicians [65]. In contrast, multi-criteria optimization (MCO) uses a pre-calculated database of treatment plans that span the viable treatment planning options [61] and a user-interface that allows the treatment planner to navigate the approximated Pareto surface to select the desirable plan from a blend of database plans that suits the treatment goals for the patient [26, 50, 51]. Therefore, a Pareto surface based MCO requires the computation of multiple treatment plans to span the space of treatment possibilities, and thus requires a fast and reliable optimization algorithm. Such a system for IMPT is currently under development at the Massachusetts Gen-

eral Hospital.

We use a linearly constrained optimization problem formulation for the IMPT MCO planning implementation. It is often the case that algorithms for linear programming (LP) problems (or, more generally, for convex constrained optimization problems) are slower than gradient based penalty function methods applied to the related global optimization problems, because LP algorithms store and utilize the exact geometry of a polyhedral feasible set and solve to exact provable optimality. In certain cases, depending on the problem structure and on the nature of the constraints, linear formulations are solved rapidly with custom algorithms. This is the case for radiation therapy problems, where we demonstrate that ART3+O can solve linear instances fast and with little memory overhead.

The author of this dissertation and his adviser G.T. Herman have been working in this area with D. Craft, T.M. Madden, K. Zhang and H.M. Kooy of the Department of Radiation Oncology, Massachusetts General Hospital and Harvard Medical School. The experimental and clinical cases in the rest of this section and in Subsection 3.2.3 are the outcome of this collaboration.

In the clinical case that we use as an example we have $M \times N = 302,491 \times 13,734$. The number of nonzero elements in matrix A is $62,226,127$, which is less than 1.5% of the total number of entries of A, an important consideration for the efficacy of projection methods for solving the problem. There is an additional technical consideration: since it is impossible to deliver negative radiation, each component of x has to be nonnegative, which results in an additional $13,734$ inequality constraints.

Several software packages are available for solving such linearly constrained optimization problems, see, e.g., [2]. To compare the efficiency of our proposed method with currently popular standard approaches, we applied them to the problem for a patient with pancreatic cancer. We used all methods to find just a feasible

point and also for eight different tasks representing various optimization criteria. The three algorithms with which we compared ART3+O were the interior point optimizer, the primal simplex optimizer and the dual simplex optimizer in the commercial software package MOSEK version 5. Typically, for each task, ART3+O used about one to two minutes and the MOSEK algorithms needed one to several hours. It is also noteworthy that the memory requirements of the MOSEK algorithms were more than ten times the memory used by ART3+O.

We demonstrate the full system (database generation and navigation) on two clinical cases. The first case we present is an esophagus case. The esophagus is a difficult treatment site due to the proximity of several critical structures, namely the lungs, spinal cord, heart, stomach and liver. Target coverage is fixed by setting hard constraints of 45 Gy as the clinical target volume (CTV) minimum dose and 50.4 Gy as the gross tumor volume (GTV) minimum dose. (Since these are doses, we point out that throughout this dissertation we use Gy as the unit of dose.) All doses are kept below 112% of 50.4 Gy. The five objectives are minimizing the mean dose to the combined lungs, the spinal cord, the heart, the stomach and the liver. The trade-off between the heart dose and the combined lung dose has been found to be the most challenging compromise for many of the esophagus patients at Massachusetts General Hospital, so we display the range of possibilities for their DVHs in Figure 1.5.

The second case is a tumor along the right rib cage, and thus the right lung is a particular sensitive critical structure. Rather than specifying lower constraints for target coverage as in the esophagus case, here we are interested in assessing how target coverage affects the sparing of the lung, and so minimizing right lung mean dose and maximizing target minimum dose are put in as objectives. The other indicated objectives are minimizing the mean doses to the spinal cord and the heart. Figure 1.6 shows two solutions, one that forces the coverage of the target

Figure 1.5: IMPT MCO planning system showing the esophagus case. The left panel displays the constraints and the objectives and shows the navigation sliders in positions for the best lung plan. The DVH displays and iso-dose lines are updated in real-time while the user navigates. Here we show the DVHs for the best lung plan and the DVHs for the best heart plan, which together display the range of planning options regarding these two critical structures. (Patient data courtesy of Dr. T. Hong.)

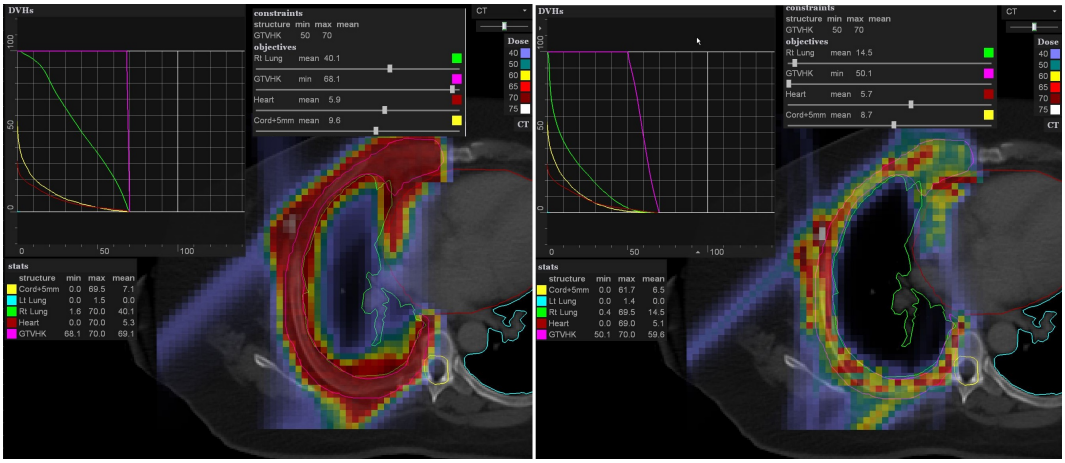


Figure 1.6: Two plans shown from the lung / chest-wall target case. The plans use three fields: anterior, posterior and posterior oblique. The left plan forces complete target volume coverage to be near to its maximum allowed value of 70 Gy (the purple DVH), while the right plan achieves maximal lung sparing (the green DVH). (Patient data courtesy of Dr. B. Eden.)

to be everywhere near to its upper constraint of 70 Gy and one that minimizes the exposure of the right lung. Note the complete sparing of the left lung as proton beams have no exit dose.

The dissertation is organized as follows: Two examples of finitely convergent sequential algorithms are first given in Chapter 2, then we design a general methodology for automatic transformation of finitely convergent sequential algorithms to their corresponding improved versions that retain finite convergence and intuitively converge faster; next we propose a convex optimization algorithm based on the fast linear feasibility algorithm ART3+. In Chapter 3, we first demonstrate by experiments on fitting pixel images by blob images that the improved algorithm ART3+ is statistically significantly faster than the original algorithm ART3; then we apply ART3+ to IMRT planning to show its superior performances over both ART3 and an LP solver; lastly our proposed linear optimization algorithm ART3+O is compared to the state of art implementation of a popular interior point method and two simplex methods in IMPT MCO and its superior efficiency in both computational time and memory usage in reported. The dissertation ends with a chapter in which its contributions, as well as planned future works, are discussed.

Chapter 2

Methods

2.1 Definitions and Problem Statement

We use $\mathbb{Z}_{\geq}$ and $\mathbb{Z}_{>}$ to denote the set of nonnegative and of positive integers, respectively. For any $N \in \mathbb{Z}_{>}$, we use $\mathbb{R}^N$ to denote the N-dimensional Euclidean space.

The subject matter of this dissertation is finitely convergent sequential algorithms. Those who work in the field "know" what these are, but this is more by consensus than by precise definition. In the next paragraphs we present our definition of this concept from a computer science point of view.

First we specify the class of problems that these algorithms are supposed to solve. We deal with constraints in $\mathbb{R}^N$ of the form $g(x) \leq 0$, where $g : \mathbb{R}^N \to \mathbb{R}$. (Equivalently such constraints could have been specified using subsets of $\mathbb{R}^N$ or predicates on $\mathbb{R}^N$.) Let $\mathbb{G}$ be a set such that if $G \in \mathbb{G}$, then G is a nonempty finite set for which there exists an N in $\mathbb{Z}_{>}$ such that if $g \in G$, then $g : \mathbb{R}^N \to \mathbb{R}$. (We will use N_G to denote this N and use M_G to denote the number of elements in G.) We refer to such a $\mathbb{G}$ as a problem set. For a problem $G \in \mathbb{G}$, if a point $x \in \mathbb{R}^{N_G}$ satisfies $g(x) \leq 0$, for all $g \in G$, then x is said to be feasible for G. The tasks that are the

subject matter of this dissertation are of the form: **Given** a $G \in \mathbb{G}$, **find** an $x \in \mathbb{R}^{N_G}$ that is feasible for G. Note that each problem set $\mathbb{G}$ requires its own algorithm.

A very simple example of such a problem set is the following.

$$\mathbb{G}_0 = \{\{\gamma_1, \gamma_2\} \mid \gamma_1, \gamma_2 : \mathbb{R} \to \mathbb{R}, \gamma_1(x) = -x, \gamma_2(x) = -x - 0.2\}. \tag{2.1}$$

This $\mathbb{G}_0$ has only one element and, for $G \in \mathbb{G}_0$, $M_G = 2$ and $N_G = 1$. The task associated with this example is to find a nonnegative real number. More complicated and useful examples are given in the next section. Those examples fall into the general class of convex feasibility problems; for a review see [4]. The methods discussed in that review are projection algorithms, as indeed are the algorithms of most of our examples in this dissertation. Such algorithms are widely used; for another example, see [6]. It should however be noted that the following definition of a sequential algorithm is general enough to include algorithms that would not be considered to be projection algorithms (an illustration is given below).

In what follows we often restrict our attention to closed problem sets, which have the following property: If $G \in \mathbb{G}$, $H \neq \emptyset$ and $H \subseteq G$, then $H \in \mathbb{G}$. (This is closure under taking nonempty subsets.) Note that $\mathbb{G}_0$ is not closed, since $\{\gamma_1\} \notin \mathbb{G}_0$.

In order to define our notion of a sequential algorithm on a problem set $\mathbb{G}$, we first introduce the more general notion of a sequential scheme on $\mathbb{G}$. It utilizes a set of states $\mathbb{S}$ that depends on the input $G \in \mathbb{G}$. In the description below $S^k \in \mathbb{S}$, for all $k \in \mathbb{Z}_{\geq}$.

Input:

$G \in \mathbb{G}$

Scheme:

1. $x^0 \leftarrow \sigma(Z_0)$
2. $S^0 \leftarrow \sigma(Z_1)$
3. $k \leftarrow 0$
4. Repeat
5. $\quad S^k \leftarrow S^0$
6. $\quad$ Repeat
7. $\quad\quad g^k \leftarrow \sigma\left(Z_2\left(S^k, k, x^k\right)\right)$
8. $\quad\quad$ If $g^k\left(x^k\right) \leq 0$
9. $\quad\quad\quad$ then
10. $\quad\quad\quad\quad x^{k+1} \leftarrow x^k$
11. $\quad\quad\quad\quad S^{k+1} \leftarrow \sigma\left(Z_3\left(S^k, k, x^k, g^k\right)\right)$
12. $\quad\quad\quad$ else
13. $\quad\quad\quad\quad x^{k+1} \leftarrow \sigma\left(Z_4\left(S^k, k, x^k, g^k\right)\right)$
14. $\quad\quad\quad\quad S^{k+1} \leftarrow \sigma\left(Z_5\left(S^k, k, x^k, g^k\right)\right)$
15. $\quad\quad k \leftarrow k+1$
16. $\quad$ Until $Condition\left(S^k\right)$ is $true$
17. Until $false$ is $true$

In this description $\sigma(Z)$ denotes an arbitrary element from the nonempty set Z. (σ is called the choice function in nondeterministic algorithms.)

In order to turn the sequential scheme on $\mathbb{G}$ into a sequential algorithm on $\mathbb{G}$, we first must specify the choice of $\mathbb{S}$ for any $G \in \mathbb{G}$. After this we need to provide algorithmic specifications of Z_0 that has to be a nonempty subset of $\mathbb{R}^{N_G}$, of Z_1 that has to be a nonempty subset of $\mathbb{S}$, and, for any $S \in \mathbb{S}, k \in \mathbb{Z}_{\geq}$ and $x \in \mathbb{R}^{N_G}$, of $Z_2(S,k,x)$ that has to be a nonempty subset of G and, for any $g \in G$, of $Z_3(S,k,x,g)$ and of $Z_5(S,k,x,g)$ that have to be nonempty subsets of $\mathbb{S}$ and of $Z_4(S,k,x,g)$ that has to be a nonempty subset of $\mathbb{R}^{N_G}$, and, for any $S \in \mathbb{S}$, of $Condition(S)$ that takes the value $true$ or $false$ depending on S. An algorithm containing such nondeterministic assignment statements will have many execution paths (usually we use just paths, for short), each path consisting of a sequence of actions that may be taken as we execute the statements (deterministic or nondeterministic) of the algorithm.

As an example, for the $\mathbb{G}_0$ defined earlier, we let $\mathbb{S} = \{\emptyset\}$, $Z_0 = \mathbb{R}$, $Z_1 = Z_3(\emptyset,k,x,g) = Z_5(\emptyset,k,x,g) = \{\emptyset\}$, $Z_2(\emptyset,k,x) = \{\gamma_1,\gamma_2\}$, for any $k \in \mathbb{Z}_{\geq}$, $x \in \mathbb{R}$, $Z_4(\emptyset,k,x,\gamma_1) = \{x-0.25\}$ and $Z_4(\emptyset,k,x,\gamma_2) = \{x+0.75\}$, and $Condition(S^k) = false$. In the paths of this algorithm in which the constraints selected by $\sigma(\{\gamma_1,\gamma_2\})$ form the sequence $(\gamma_1,\gamma_2,\gamma_1,\gamma_2,\ldots)$, the aim of the task will be achieved in a finite number of steps. On the other hand, in a path in which the constraints selected by $\sigma(\{\gamma_1,\gamma_2\})$ form the sequence $(\gamma_1,\gamma_1,\gamma_1,\gamma_2,\gamma_1,\gamma_1,\gamma_1,\gamma_2,\ldots)$, a negative x^0 does not lead to a nonnegative x^k and the aim of the task is never achieved.

A further example, in which $\mathbb{S} \neq \{\emptyset\}$, is also an illustration of how the choice function enters into the algorithmic operation of a device, such as a computer that is connected to a network from which it receives a stream of input. The device of our simple illustration is the proposed new check-out counter of Nevada Supermarkets that, instead of giving change to customers, rounds up the purchase price to the nearest dollar, but gives a prize to a customer whenever the total amount of extra profits due to rounding up has reached $(10+u)$ dollars, where u is an integer, $1 \leq u \leq U$. By choosing the u large enough and the prize cheap enough,

Nevada Supermarkets will not only speed up the check-out process, but will also make extra profit. To model this in our framework, we set $\mathbb{S} = \{P,R\}$, where P signifies that either the previous customer was given a prize or we are dealing with the first customer at this check-out counter, while R signifies the alternative. We use $\mathbb{G} = \{G_1, G_2, \ldots, G_U\}$ with, for $1 \leq u \leq U$, $G_u = G_{u,1} \cup G_{u,2}$ with, for $1 \leq v \leq 2$, $G_{u,v} = \{\gamma_{u,v,w} | 0 \leq w \leq 99\}$. The w signifies the amount (in cents) of the round up associated with a customer, it is to model the unpredictable variability in w that we have to resort to using a choice function in our algorithm. To complete the details, we define $\gamma_{u,1,w}(x) = w+1$, $\gamma_{u,2,w}(x) = 100(10+u) - w - x$, $Z_0 = \{0\}$, $Z_1 = \{P\}$, $Z_2(S,k,x)$ is $G_{u,1}$ if $S = P$ and is $G_{u,2}$ otherwise, $Z_3(S,k,x,g) = \{P\}$, $Z_4(S,k,x,g)$ is $\{w\}$ if $g = \gamma_{u,1,w}$ and is $\{w+x\}$ if $g = \gamma_{u,2,w}$, $Z_5 = \{R\}$, and $Condition(S)$ is *true* if, and only if, $S = P$. It is easy to see that in the running of this algorithm, k signifies the customer number (starting with 1) and that customer should be given a prize if $Condition(S^k)$ is *true*. It can also be seen that, for all k, $0 \leq x^k < 100(10+u)$ and, consequently, the constraint $\gamma_{u,2,0}(x^k) \leq 0$ is never satisfied. Thus this example is very different from the ones in the rest of this dissertation (which are the ones that motivated our research); in all those examples the aim is the satisfaction of all the constraints.

Let us call the execution paths of a sequential algorithm when provided with input G its G-paths. (Often, when G is understood, we just say path instead of G-path.) Each such G-path P produces, in particular, an infinite control sequence $c(P) = \left(g^0, g^1, g^2, \ldots\right)$ of elements of G (that we also denote by $\left(g^k\right)_{k \in \mathbb{Z}_{\geq}}$), an infinite solution sequence $s(P) = \left(x^0, x^1, x^2, \ldots\right)$ of elements of $\mathbb{R}^{N_G}$ (that we also denote by $\left(x^k\right)_{k \in \mathbb{Z}_{\geq}}$), and an infinite state sequence $d(P) = \left(S^0, S^1, S^2, \ldots\right)$ of elements of $\mathbb{S}$ (that we also denote by $\left(S^k\right)_{k \in \mathbb{Z}_{\geq}}$). We say that the control sequence is

1. cyclic if, for every $k \in \mathbb{Z}_{\geq}$, $\left\{g^k, g^{k+1}, \ldots, g^{k+M_G-1}\right\} = G$;

2. almost cyclic if there exists a $C \in \mathbb{Z}_>$ such that, for every $k \in \mathbb{Z}_\geq$, $\left\{g^k, g^{k+1}, \ldots, g^{k+C-1}\right\} = G$;

3. repetitive if, for every $k \in \mathbb{Z}_\geq$ and $g \in G$, there exists a $k' > k$, such that $g^{k'} = g$.

Note that a cyclic control sequence is almost cyclic with $C = M_G$, and an almost cyclic control sequence is repetitive. In our first example the control sequence $(\gamma_1, \gamma_2, \gamma_1, \gamma_2, \ldots)$ is cyclic and $(\gamma_1, \gamma_1, \gamma_1, \gamma_2, \gamma_1, \gamma_1, \gamma_1, \gamma_2, \ldots)$ is almost cyclic (and is also repetitive).

In the next three paragraphs, we provide definitions of the following concepts:

- finite convergence of a solution sequence,
- finite convergence of a sequential algorithm,
- finite convergence of a sequential algorithm under cyclic (respectively, almost cyclic or repetitive) control,
- the equivalence of two paths,
- the generalization of a sequential algorithm,
- the equivalence of two sequential algorithms.

We say that the solution sequence $\left(x^k\right)_{k \in \mathbb{Z}_\geq}$ is finitely convergent if, for some $K \in \mathbb{Z}_\geq$, $x^k = x^K$, for all $k \geq K$. Under these circumstances, we say that $\left(x^k\right)_{k \in \mathbb{Z}_\geq}$ converges finitely to x^K. In our first example, if the control sequence is $(\gamma_1, \gamma_2, \gamma_1, \gamma_2, \ldots)$, then the solution sequence is finitely convergent to a nonnegative real number; if the control sequence is $(\gamma_1, \gamma_1, \gamma_1, \gamma_2, \gamma_1, \gamma_1, \gamma_1, \gamma_2, \ldots)$, then the solution sequence is not finitely convergent, unless $x^0 \geq 0$. In our second example, the solution sequence is finitely convergent only if, after some point, the purchase of every customer that

comes to the check-out counter costs exactly some number of dollars (and hence there is no round up).

We say that a sequential algorithm on $\mathbb{G}$ is finitely convergent if, for every $G \in \mathbb{G}$, the solution sequence of every G-path is finitely convergent to a point feasible for G. We say that a sequential algorithm on $\mathbb{G}$ is finitely convergent under cyclic (respectively, almost cyclic or repetitive) control if, for every $G \in \mathbb{G}$ and G-path whose control sequence is cyclic (respectively, almost cyclic or repetitive) the solution sequence is finitely convergent to a point feasible for G. Note that the sequential algorithm we defined on $\mathbb{G}_0$ is finitely convergent under cyclic control, but it is not necessarily finitely convergent under almost cyclic control (for example, when the control sequence is $(\gamma_1, \gamma_1, \gamma_1, \gamma_2, \gamma_1, \gamma_1, \gamma_1, \gamma_2, \ldots)$).

Let A and A' be two sequential algorithms for the task associated with the problem set $\mathbb{G}$. For any $G \in \mathbb{G}$, we say that a G-path P of A and a G-path P' of A' are equivalent if they have the same control sequence and the same solution sequence. We say that A' is a generalization of A if, for every $G \in \mathbb{G}$ and every G-path of A, there exists a G-path of A' equivalent to it. We say that A and A' are equivalent if each of them is a generalization of the other.

The major concern of this dissertation is: Given a task of the kind specified at the beginning of this section for which there exists a finitely convergent sequential algorithm, how should we specify the set $\mathbb{S}$ of states, the sets Z_0, Z_1, Z_2, Z_3, Z_4, Z_5 and the *Condition* that determine the control sequence of the execution paths so that the resulting algorithm is more efficient than other such algorithms using alternative specifications. We do not attempt in this dissertation to give a precise definition of "the efficiency of an algorithm." We use the phrase in the intuitive sense of something that reflects the computational cost of the algorithm reaching a feasible point for a given problem.

2.2 Illustrative Examples of Finitely Convergent Sequential Algorithm

Many published algorithms can be reformulated so that they become sequential algorithms as defined in the previous section. We give two examples from the literature.

2.2.1 ART3 and Its Generalization

The first example that we use to illustrate the definitions of the previous section is a generalization of the algorithm ART3 [35] (we will still call it ART3 in the dissertation) designed to find a common point of given halfspaces and hyperslabs. ART3 is one of the large class of ART that has been the subject of very active research over the past decades; for a recent example, see [32]. To bring ART3 in line with the notation used in this dissertation, we define a particular problem set $\mathbb{G}_A$ that contains all G that satisfy:

1. All $g \in G$ are of the form

$$g(x) = \begin{cases} \langle a_g, x\rangle - u_g, & \text{if } \langle a_g, x\rangle > u_g, \\ l_g - \langle a_g, x\rangle, & \text{if } \langle a_g, x\rangle < l_g, \\ \max\left(l_g - \langle a_g, x\rangle, \langle a_g, x\rangle - u_g\right), & \text{otherwise,} \end{cases} \tag{2.2}$$

where $l_g \in \mathbb{R} \bigcup \{-\infty\}$, $u_g \in \mathbb{R} \bigcup \{+\infty\}$, but either $l_g \in \mathbb{R}$ or $u_g \in \mathbb{R}$, $l_g \leq u_g$, $a_g \in \mathbb{R}^{N_G}$ and $\|a_g\| \neq 0$. Note that each such function gives rise to a hyperslab

$$Q_g = \left\{x \in \mathbb{R}^{N_G} \,|\, l_g \leq \langle a_g, x\rangle \leq u_g\right\}, \tag{2.3}$$

or a halfspace when $l_g = -\infty$, $u_g \in \mathbb{R}$ or $u_g = +\infty$, $l_g \in \mathbb{R}$, such that $x \in Q_g$ if, and only if, $g(x) \leq 0$.

2. The set of all $x \in \mathbb{R}^{N_G}$ that are feasible for G (equivalently, the intersection of all Q_g for $g \in G$) is full-dimensional, in the sense that it is not a subset of any $(N_G - 1)$-dimensional hyperplane in $\mathbb{R}^{N_G}$.

This $\mathbb{G}_A$ is closed. All the problems in $\mathbb{G}_A$ are linear feasibility problems [4].

The choices that are made to turn the sequential scheme into the GART3 algorithm (which is a generalization of the ART3 algorithm of [35]) are the following.

- $\mathbb{S} = \{\emptyset\}$.
- $Z_0 = \mathbb{R}^{N_G}$.
- $Z_1 = \{\emptyset\}$.
- $Z_2(\emptyset, k, x) = G$.
- $Z_3(\emptyset, k, x, g) = \{\emptyset\}$.
- $Z_4(\emptyset, k, x, g) = \{p_g(x)\}$, where

$$p_g(x) = x - \begin{cases} \frac{\langle a_g, x\rangle - (u_g + l_g)/2}{\|a_g\|^2} a_g, & \text{if } \langle a_g, x\rangle < l_g - (u_g - l_g)/2, \\ 2\frac{\langle a_g, x\rangle - l_g}{\|a_g\|^2} a_g, & \text{if } l_g - (u_g - l_g)/2 \le \langle a_g, x\rangle < l_g, \\ 2\frac{\langle a_g, x\rangle - u_g}{\|a_g\|^2} a_g, & \text{if } u_g < \langle a_g, x\rangle \le u_g + (u_g - l_g)/2, \\ \frac{\langle a_g, x\rangle - (u_g + l_g)/2}{\|a_g\|^2} a_g, & \text{if } u_g + (u_g - l_g))/2 < \langle a_g, x\rangle. \end{cases} \tag{2.4}$$

- $Z_5(\emptyset, k, x, g) = \{\emptyset\}$.
- $Condition(\emptyset) = false$.

The function p_g of (2.4) is illustrated in Figure 2.1.

As a consequence of the definition of $Z_2(\emptyset, k, x)$, given any $G \in \mathbb{G}_A$, every element of $\left\{ \left(g^k\right)_{k \in \mathbb{Z}_\geq} \mid g^k \in G, \text{ for } k \in \mathbb{Z}_\geq \right\}$ is a control sequence of a G-path of

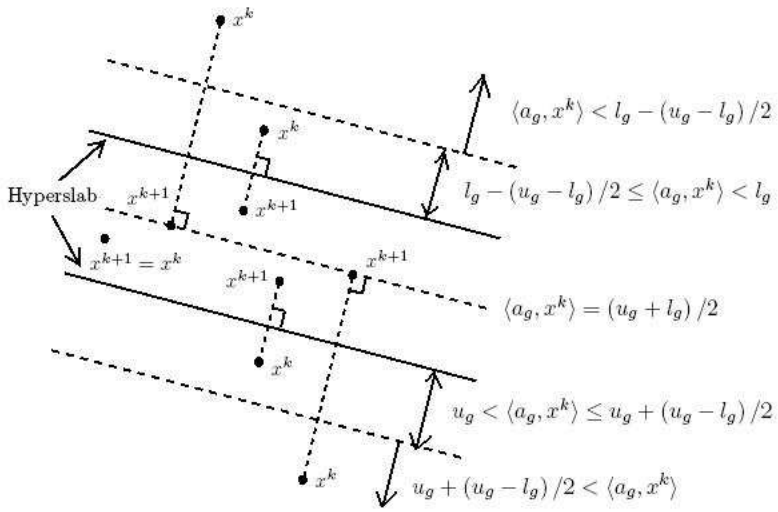


Figure 2.1: The iterative step of GART3 from x^k to x^{k+1}, i.e., to $p_g\left(x^k\right)$.

GART3. The way ART3 is specified in [35] restricts the G-paths to those whose control sequence is cyclic. A way of achieving this using our definition of a sequential algorithm is the following (this way of describing things has been chosen since it introduces a notation that makes easier in the next section the discussion of how an intuitively more efficient version can be automatically produced for a finitely convergent sequential algorithm.)

Let

$$\mathbb{F} = \{(h_1, h_2, \ldots, h_U) \mid U \in \mathbb{Z}_{\geq} \text{ and, for } 1 \leq u \leq U,\, h_u \in G\}\,. \tag{2.5}$$

$\mathbb{F}$ is the set of all finite sequences of elements of G including the empty sequence. Then we define ART3 as a sequential algorithm on $\mathbb{G}_A$ by making the choices (for $G \in \mathbb{G}_A$):

- $\mathbb{S} = \mathbb{F}$.
- $Z_0 = \mathbb{R}^{N_G}$.
- $Z_1 = \{(g_1, g_2, \ldots, g_{M_G}) \mid \{g_1, g_2, \ldots, g_{M_G}\} = G\}$.

- $Z_2\left(\left(h_1,h_2,\ldots,h_U\right),k,x\right)=\{h_1\}$.
- $Z_3\left(\left(h_1,h_2,\ldots,h_U\right),k,x,g\right)=\{(h_2,\ldots,h_U,h_1)\}$.
- $Z_4\left(\left(h_1,h_2,\ldots,h_U\right),k,x,g\right)=\left\{p_g\left(x\right)\right\}$.
- $Z_5\left(\left(h_1,h_2,\ldots,h_U\right),k,x,g\right)=\{(h_2,\ldots,h_U,h_1)\}$.
- $Condition\left(S\right)=false$, for all $S\in\mathbb{S}$.

The consequence is that, given the same input, for every path of ART3, there exists a path of GART3 (whose control sequence is cyclic) equivalent to it. Therefore GART3 is a generalization of ART3. ART3 with finite bounds was proved to be a finitely convergent sequential algorithm on $\mathbb{G}_A$ in [35], and the proof therein can be easily generalized to ART3 with infinite bounds as defined in this dissertation. Hence it is also the case that GART3 is a finitely convergent sequential algorithm on $\mathbb{G}_A$ under cyclic control. We provide a proof of the finite convergence of GART3 under repetitive control in Section 2.3.

2.2.2 MCSP and Its Generalization

The second illustrative example is the Modified Cyclic Subgradient Projection (MCSP) method of [29]. For this, $\mathbb{G}_B$ is the set of all G, in which every element $g:\mathbb{R}^{N_G}\rightarrow\mathbb{R}$ is a convex function (i.e., for all $x,y\in\mathbb{R}^{N_G}$ and $\alpha\in[0,1]$, $g\left(\alpha x+\left(1-\alpha\right)y\right)\leq\alpha g(x)+\left(1-\alpha\right)g(y)$) and there exists a positive number $\hat{\varepsilon}$ such that

$$g(x)+\hat{\varepsilon}\leq 0,\ \text{for all } g\in G, \tag{2.6}$$

has a solution $x\in\mathbb{R}^{N_G}$. Such a G is said to satisfy the Slater condition. Note that $\mathbb{G}_A\subset\mathbb{G}_B$ since any function $g\in G\in\mathbb{G}_A$, as defined by (2.2), is a convex function and full-dimensionality implies that a ball with some positive radius can be fitted

into the intersection of the hyperslabs determined by elements of G such that the center of the ball satisfies (2.6) for some positive $\hat{\varepsilon}$. The problem set $\mathbb{G}_B$ is also closed. The task of MCSP is to find, for a given $G \in \mathbb{G}_B$, an $x \in \mathbb{R}^{N_G}$ that is feasible for G. All the problems in $\mathbb{G}_B$ are convex feasibility problems [4].

A subgradient of a convex function $g : \mathbb{R}^N \to \mathbb{R}$ at the point $x \in \mathbb{R}^N$, is a vector $t \in \mathbb{R}^N$, that satisfies

$$\langle t, y - x \rangle \leq g(y) - g(x), \tag{2.7}$$

for all $y \in \mathbb{R}^N$. The set of all such subgradients is denoted by $\partial g(x)$. If g is differentiable at $x \in \mathbb{R}^N$, then its gradient $\bigtriangledown g(x)$ is the unique subgradient of g at x.

Let

$$\begin{aligned}
\mathbb{E} = \Big\{ \Big((\alpha_k)_{k \in \mathbb{Z}_\geq}, (\varepsilon_k)_{k \in \mathbb{Z}_\geq} \Big) \Big| \\
(\alpha_k)_{k \in \mathbb{Z}_\geq} \text{ is a sequence of } \alpha_k \in \mathbb{R} \text{ such that } 0 < \liminf \alpha_k \leq \limsup \alpha_k < 2, \\
(\varepsilon_k)_{k \in \mathbb{Z}_\geq} \text{ is a monotonically decreasing sequence of positive } \varepsilon_k \in \mathbb{R} \\
\text{such that } \varepsilon_k \to 0 \text{ as } k \to \infty \text{ and } \textstyle\sum_{k=0}^{\infty} \varepsilon_k = \infty \Big\}.
\end{aligned} \tag{2.8}$$

We turn the sequential scheme into the generalized MCSP algorithm (GMCSP) by making the following choices.

- $\mathbb{S} = \mathbb{E}$.
- $Z_0 = \mathbb{R}^{N_G}$.
- $Z_1 = \mathbb{E}$.
- $Z_2 \Big(\Big((\alpha_k)_{k \in \mathbb{Z}_\geq}, (\varepsilon_k)_{k \in \mathbb{Z}_\geq} \Big), k, x \Big) = G$.
- $Z_3 \Big(\Big((\alpha_k)_{k \in \mathbb{Z}_\geq}, (\varepsilon_k)_{k \in \mathbb{Z}_\geq} \Big), k, x, g \Big) = \Big\{ \Big((\alpha_k)_{k \in \mathbb{Z}_\geq}, (\varepsilon_k)_{k \in \mathbb{Z}_\geq} \Big) \Big\}$.
- $Z_4 \Big(\Big((\alpha_k)_{k \in \mathbb{Z}_\geq}, (\varepsilon_k)_{k \in \mathbb{Z}_\geq} \Big), k, x, g \Big) = \Big\{ x - \alpha_k \frac{g(x) + \varepsilon_k}{\|t\|^2} t \Big| t \in \partial g(x) \Big\}$.

- $Z_5\left(\left((\alpha_k)_{k\in\mathbb{Z}_\geq},(\varepsilon_k)_{k\in\mathbb{Z}_\geq}\right),k,x,g\right)=\left\{\left((\alpha_k)_{k\in\mathbb{Z}_\geq},(\varepsilon_k)_{k\in\mathbb{Z}_\geq}\right)\right\}$.
- $Condition\left(\left((\alpha_k)_{k\in\mathbb{Z}_\geq},(\varepsilon_k)_{k\in\mathbb{Z}_\geq}\right)\right)=false$.

It follows from the results in [29] that the GMCSP algorithm is a finitely convergent sequential algorithm on $\mathbb{G}_B$ under almost cyclic control.

2.3 A Basic Idea for Improving Efficiency

Consider the lines

8. If $g^k\left(x^k\right)\leq 0$
9. then
10. $x^{k+1}\leftarrow x^k$
11. $S^{k+1}\leftarrow\sigma\left(Z_3\left(S^k,k,x^k,g^k\right)\right)$

in the sequential scheme of Section 2.1. Since the vector x^k does not change as a result of these steps, they are essentially just wasting computer time. For example, in ART3 (and similarly for any other finitely convergent sequential algorithm in which every path P is forced to have a cyclic control sequence), as the k approaches K, it is the case most of the time that $g^k\left(x^k\right)\leq 0$. Nevertheless, the control sequence $c(P)$ keeps cycling through all the M_G elements of G just to find out that $l_{g^k}\leq\langle a_{g^k},x^k\rangle\leq u_{g^k}$ is satisfied. Since in practice both M_G and the dimension N_G of the space in which the inner product is taken are large, this can result in a considerable computational burden with very little to show for it. For this reason it appears to be a good idea to rewrite the algorithm in such a way that it is rarely the case that $g^k\left(x^k\right)\leq 0$.

Such considerations led us to introduce in [37] the algorithm ART3+, which is a more efficient special case of GART3 than ART3 is. In the following sections we report on this algorithm in the more general context of this dissertation. It is inherent in the discussion above that if ART3 is to be made more efficient in the manner

described, then the resulting control sequences will no longer be necessarily cyclic. The following theorem is proved based on an approach presented in our paper [37] and will be used in the next section.

Theorem 2.3.1 *The GART3 algorithm is finitely convergent for $\mathbb{G}_A$ under repetitive control.*

Proof Let $G \in \mathbb{G}_A$. In this proof we abbreviate M_G by M and N_G by N. We need to show that, for any G-path whose control sequence is repetitive, the solution sequence is finitely convergent to a point that is feasible for G.

Let $c(P) = \left(g^k\right)_{k\in\mathbb{Z}_\geq}$ be a repetitive control sequence for a G-path P, and let $s(P) = \left(x^k\right)_{k\in\mathbb{Z}_\geq}$. By the definition of $\mathbb{G}_A$, we know that $Q = \bigcap_{g\in G} Q_g$ is full-dimensional.

We first observe that, for all $k \in \mathbb{Z}_\geq$,

$$x^{k+1} \in Q_{g^k} \tag{2.9}$$

and, for any $z \in Q \subseteq Q_{g^k}$,

$$\left\|x^{k+1} - z\right\| \leq \left\|x^k - z\right\|. \tag{2.10}$$

This is trivially so if $g^k\left(x^k\right) \leq 0$, and it is checked otherwise based on simple geometrical considerations on the nature of hyperslabs and the definitions in (2.4), see Figure 2.1. Hence the solution sequence $\left(x^k\right)_{k\in\mathbb{Z}_\geq}$ is bounded and so it has accumulation points. But it cannot have two different accumulation points x and y, since in that case we would have from (2.10) that, for all $z \in Q$, $\|x-z\| = \|y-z\|$ and so Q would be a subset of a hyperplane separating x and y, contradicting that it is full-dimensional. It follows therefore that the solution sequence $\left(x^k\right)_{k\in\mathbb{Z}_\geq}$ is convergent to a point, let us call it x^*.

If $x^* \notin Q$, then $x^* \notin Q_g$ for at least one $g \in G$. Since Q_g is a closed convex set, there must exist a $K \in \mathbb{Z}_{\geq}$ such that $x^k \notin Q_g$, for all $k \geq K$. From the definition of repetitive control sequence, we see that we are bound to come across a $k \geq K$ such that $g^k = g$, and so (2.9) implies that $x^{k+1} \in Q_g$. This contradiction shows that $x^* \in Q$.

Now define the positive real numbers r_g, for all $g \in G$, by

$$r_g = \begin{cases} \min\left\{u_g - \left\langle a_g, x^*\right\rangle, \left\langle a_g, x^*\right\rangle - l_g\right\}, & \text{if } x^* \text{ is in the interior of } Q_g, \\ (u_g - l_g)/2, & \text{if } x^* \text{ is on the boundary of } Q_g \\ & \text{and } l_g \neq -\infty, u_g \neq +\infty, \\ 1, & \text{otherwise,} \end{cases} \tag{2.11}$$

and the ball B by

$$B = \left\{ x \in \mathbb{R}^N \mid \|x - x^*\| \leq \min_{g \in G} \frac{r_g}{\left\|a_g\right\|} \right\}. \tag{2.12}$$

By convergence, there is a $K \in \mathbb{Z}_{\geq}$ such that $x^k \in B$, for all $k \geq K$. We are now going to show that, for all $k \geq K$, $\left\|x^{k+1} - x^*\right\| = \left\|x^k - x^*\right\|$. This will complete our proof since it, combined with the convergence of $\left(x^k\right)_{k \in \mathbb{Z}_{\geq}}$ to x^*, implies that, for $k \geq K$, $x^k = x^* \in Q$.

So consider a $k \geq K$. We know that $x^k \in B$. If x^* is in the interior of Q_{g^k}, then (2.11) and (2.12) imply (consult Figure 2.1) that $x^k \in Q_{g^k}$, resulting in, $x^{k+1} = x^k$. The alternative is that x^* is not in the interior of Q_{g^k}, and so we must have that either $\left\langle a_{g^k}, x^*\right\rangle = l_{g^k}$ or $\left\langle a_{g^k}, x^*\right\rangle = u_{g^k}$. Consider the first case. By (2.11) and (2.12) we have either that $x^k \in Q_{g^k}$ and so $x^{k+1} = x^k$, or that $l_{g^k} - \left(u_{g^k} - l_{g^k}\right)/2 \leq \left\langle a_{g^k}, x^k\right\rangle < l_{g^k}$ and so $x^{k+1} = x^k - 2\frac{\left\langle a_{g^k}, x^k\right\rangle - l_{g^k}}{\left\|a_{g^k}\right\|^2} a_{g^k}$, which combined with $\left\langle a_{g^k}, x^*\right\rangle = l_{g^k}$ implies that $\left\|x^{k+1} - x^*\right\| = \left\|x^k - x^*\right\|$ (again, consult Figure 2.1). The same result can be similarly derived if $\left\langle a_{g^k}, x^*\right\rangle = u_{g^k}$. ■

2.4 Improving the Efficiency of Finitely Convergent Algorithms

As a generalization of the ART3 algorithm, we say that a sequential algorithm on a problem set $\mathbb{G}$ is cyclic if, given input $G \in \mathbb{G}$, the following are satisfied.

- $\mathbb{S} = \mathbb{F} \times \mathbb{H}$, where $\mathbb{F}$ is defined in (2.5) and $\mathbb{H}$ is some auxiliary set of states.
- $Z_0 = \mathbb{R}^{N_G}$.
- $Z_1 = \{(g_1, g_2, \ldots, g_{M_G}) \mid \{g_1, g_2, \ldots, g_{M_G}\} = G\} \times Z_{\mathbb{H}}$, $Z_{\mathbb{H}} \subseteq \mathbb{H}$, $Z_{\mathbb{H}} \neq \emptyset$.
- $Z_2\left(\left(\left(h_1, h_2, \ldots, h_U\right), H\right), k, x\right) = \{h_1\}$.
- $Z_3\left(\left(\left(h_1, h_2, \ldots, h_U\right), H\right), k, x, g\right) = \{\left(\left(h_2, \ldots, h_U, h_1\right), H\right)\}$.
- For any $H \in \mathbb{H}$,

$$Z_4\left(\left(F_2, H\right), k, x, g\right) = Z_4\left(\left(F_1, H\right), k.x, g\right), \tag{2.13}$$

  for all $F_1, F_2 \in \mathbb{F}$, $k \in \mathbb{Z}_{\geq}$, $x \in \mathbb{R}^{N_G}$, and $g \in G$.

- For any $H_1 \in \mathbb{H}$ and $l \in \mathbb{Z}_{\geq}$, there exists $H_2 \in \mathbb{H}$ such that

$$Z_4\left(\left(F_2, H_2\right), k, x, g\right) = Z_4\left(\left(F_1, H_1\right), l + k, x, g\right), \tag{2.14}$$

  for all $F_1, F_2 \in \mathbb{F}$, $k \in \mathbb{Z}_{\geq}$, $x \in \mathbb{R}^{N_G}$, and $g \in G$.

- $Z_5\left(\left(\left(h_1, h_2, \ldots, h_U\right), H\right), k, x, g\right) = \{\left(\left(h_2, \ldots, h_U, h_1\right), H\right)\}$.
- $Condition(S) = false$, for all $S \in \mathbb{S}$.

It is easy to check that the control sequence of every path of a cyclic sequential algorithm is cyclic. To demonstrate the appropriateness of our definition of a cyclic

sequential algorithm, we now give the specifications for two of them. The first we call CART3, it is essentially ART3 expressed in the form of the general definition. The other is CMCSP, it is a cyclic version of MCSP.

Given input $G \in \mathbb{G}_A$, the choices made for CART3 are:

- $\mathbb{S} = \mathbb{F} \times \{\emptyset\}$.
- $Z_0 = \mathbb{R}^{N_G}$.
- $Z_1 = \{((g_1, g_2, \ldots, g_{M_G}), \emptyset) \mid \{g_1, g_2, \ldots, g_{M_G}\} = G\}$.
- $Z_2(((h_1, h_2, \ldots, h_U), \emptyset), k, x) = \{h_1\}$.
- $Z_3(((h_1, h_2, \ldots, h_U), \emptyset), k, x, g) = \{((h_2, \ldots, h_U, h_1), \emptyset)\}$.
- $Z_4(((h_1, h_2, \ldots, h_U), \emptyset), k, x, g) = \{p_g(x)\}$.
- $Z_5(((h_1, h_2, \ldots, h_U), \emptyset), k, x, g) = \{((h_2, \ldots, h_U, h_1), \emptyset)\}$.
- $Condition(S) = false$, for all $S \in \mathbb{S}$.

If we plug in these specifications into the sequential scheme, we get that the cyclic algorithm CART3, for input $G \in \mathbb{G}_A$, is the following.

1. $x^0 \leftarrow \sigma\left(\mathbb{R}^{N_G}\right)$
2. $S^0 \leftarrow \sigma\left(\{((g_1, g_2, \ldots, g_{M_G}), \emptyset) \mid \{g_1, g_2, \ldots, g_{M_G}\} = G\}\right)$
3. $k \leftarrow 0$
4. Repeat
5. $\quad S^k \leftarrow S^0$
6. $\quad$ Repeat
7. $\quad\quad g^k \leftarrow h_1$, if $S^k = ((h_1, h_2, \ldots, h_{M_G}), \emptyset)$

8. If $g^k\left(x^k\right) \leq 0$

9. then

10. $x^{k+1} \leftarrow x^k$

11. $S^{k+1} \leftarrow ((h_2,\ldots,h_{M_G},h_1),\emptyset)$, if $S^k = ((h_1,h_2,\ldots,h_{M_G}),\emptyset)$

12. else

13. $x^{k+1} \leftarrow p_{g^k}\left(x^k\right)$

14. $S^{k+1} \leftarrow ((h_2,\ldots,h_{M_G},h_1),\emptyset)$, if $S^k = ((h_1,h_2,\ldots,h_{M_G}),\emptyset)$

15. $k \leftarrow k+1$

16. Until $false$ is $true$

It is easy to see that this algorithm is a cyclic sequential one with $\mathbb{H} = \{\emptyset\}$. In particular, (2.13) and (2.14) are satisfied, since Z_4 in CART3 depends only on x and g. Also, it is clearly the case that CART3 is equivalent to the ART3 in [35] and GART3 is a generalization of CART3.

The cyclic MCSP (CMCSP) uses $\mathbb{E}$ defined in (2.8) as the set $\mathbb{H}$ of auxiliary states. Given input $G \in \mathbb{G}_{\mathbb{B}}$, the choices made for CMCSP are:

- $\mathbb{S} = \mathbb{F}\times\mathbb{E}$.
- $Z_0 = \mathbb{R}^{N_G}$.
- $Z_1 = \{(g_1,g_2,\ldots,g_{M_G}) \mid \{g_1,g_2,\ldots,g_{M_G}\} = G\} \times \mathbb{E}$.
- $Z_2\left(\left((h_1,h_2,\ldots,h_U),\left((\alpha_k)_{k\in\mathbb{Z}_\geq},(\varepsilon_k)_{k\in\mathbb{Z}_\geq}\right)\right),k,x\right) = \{h_1\}$.

- $Z_3\left(\left((h_1,h_2,\ldots,h_U),\left((\alpha_k)_{k\in\mathbb{Z}_\geq},(\varepsilon_k)_{k\in\mathbb{Z}_\geq}\right)\right),k,x,g\right) =$
$$\left\{\left((h_2,\ldots,h_U,h_1),\left((\alpha_k)_{k\in\mathbb{Z}_\geq},(\varepsilon_k)_{k\in\mathbb{Z}_\geq}\right)\right)\right\}.$$
- $Z_4\left(\left((h_1,h_2,\ldots,h_U),\left((\alpha_k)_{k\in\mathbb{Z}_\geq},(\varepsilon_k)_{k\in\mathbb{Z}_\geq}\right)\right),k,x,g\right) =$
$$\left\{x-\alpha_k\tfrac{g(x)+\varepsilon_k}{\|t\|^2}t\,\middle|\,t\in\partial g(x)\right\}.$$
- $Z_5\left(\left((h_1,h_2,\ldots,h_U),\left((\alpha_k)_{k\in\mathbb{Z}_\geq},(\varepsilon_k)_{k\in\mathbb{Z}_\geq}\right)\right),k,x,g\right) =$
$$\left\{\left((h_2,\ldots,h_U,h_1),\left((\alpha_k)_{k\in\mathbb{Z}_\geq},(\varepsilon_k)_{k\in\mathbb{Z}_\geq}\right)\right)\right\}.$$
- $Condition\left(\left((h_1,h_2,\ldots,h_U),\left((\alpha_k)_{k\in\mathbb{Z}_\geq},(\varepsilon_k)_{k\in\mathbb{Z}_\geq}\right)\right)\right) = false.$

It is easy to see that this algorithm is a cyclic sequential one with $\mathbb{H}=\mathbb{E}$. In particular, (2.13) and (2.14) are satisfied, since Z_4 is independent of $(h_1,h_2,\ldots,h_U)$ and the tail of any element in $\mathbb{E}$ is also in $\mathbb{E}$. GMCSP is a generalization of CMCSP.

The method by which ART3+ is obtained from ART3 in [37] generalizes to obtaining a sequential algorithm A^+ from a cyclic sequential algorithm A. The specifications for A^+ are exactly the ones for A, with the exception of Z_3 and the *Condition*, which are replaced by

- $Z_3^+\left(((h_1,h_2,\ldots,h_U),H),k,x,g\right) = \{((h_2,\ldots,h_U),H)\}$.
- $Condition^+(S) = \begin{cases} true, & \text{if } S=((),H), \text{ for some } H\in\mathbb{H}, \\ false, & \text{otherwise.} \end{cases}$

To see the purpose of introducing such a change, consider the discussion at the beginning of Section 2.3. Suppose that at a time line 7 is executed in a path either of the cyclic algorithm A or of A^+ (recall the definitions of a sequential scheme and a sequential algorithm in Section 2.1) $S^k=((h_1,h_2,\ldots,h_U),H)$. Then, by the condition on Z_2 in a cyclic algorithm, execution of line 7 will result in $g^k=h_1$. Now observe what happens if the condition in line 8 is satisfied. In either algorithm we get $x^{k+1}\leftarrow x^k$, and then we get to line 11. Due to the fact that the Z_3^+ of A^+ is different from the Z_3 of A, the behavior of the two algorithms diverge from each

other: After executing line 11, we have $S^{k+1} = ((h_2, \ldots, h_U), H)$ in the path of A^+ and $S^{k+1} = ((h_2, \ldots, h_U, h_1), H)$ in the path of A. This means that as opposed to the cyclic nature of the control sequence of A, in the following repeated executions by A^+ of lines 7-15, we will not be assigning h_1 to g^k; avoiding the wasteful behavior of the cyclic algorithm pointed out in Section 2.3. However, this brings about the danger that even if A is finitely convergent to a feasible point for any input from $\mathbb{G}$, A^+ may not share this desirable property: just because we have that $h_1\left(x^k\right) \leq 0$, it does not mean that $h_1\left(x^{k'}\right) \leq 0$, for all $k' > k$. The aim of changing *Condition* to $Condition^+$ is to retain convergence to a feasible point. The next theorems specify the circumstances under which this aim is achieved.

Theorem 2.4.1 *Let $\mathbb{G}$ be a closed problem set. If a cyclic sequential algorithm A on $\mathbb{G}$ is finitely convergent, then the control sequence of every path of A^+ is repetitive.*

Proof Let $\mathbb{G}$ be a closed problem set and A be a cyclic sequential algorithm that is finitely convergent on $\mathbb{G}$. Assume that $G^+ \in \mathbb{G}$ is such that the control sequence $c(P^+)$ of a G^+-path P^+ of A^+ is not repetitive. We show that this assumption leads to a contradiction. We use the notations $c(P^+) = \left(f^l\right)_{l \in Z_{\geq}}$, $s(P^+) = \left(y^l\right)_{l \in Z_{\geq}}$, and $d(P^+) = \left(T^l\right)_{l \in Z_{\geq}}$. It follows from the assumption that after some point in P^+, $Condition^+(T^l)$ is never assigned the value *true*; for otherwise line 5 would set T^l to be $\left(\left(f_1, f_2, \ldots, f_{M_{G^+}}\right), H\right)$, with $\left\{f_1, f_2, \ldots, f_{M_{G^+}}\right\} = G^+$ and $H \in \mathbb{H}$, and the inner loop (lines 7-15) would be executed, which would result in all the constraints in G^+ occurring as the next M_{G^+} elements in the control sequence $c(P^+)$. Consequently, after some point in P^+, it cannot be the case that $T^l = ((), H)$, for some $H \in \mathbb{H}$, and the inner loop (lines 7-15) will be repeatedly executed without ever exiting from it. Since the definition of Z_3^+ implies that the execution of line 11 removes the first element of $(h_1, h_2, \ldots, h_U)$, this means that after some later point, line 11 is never executed and $f^l\left(y^l\right) \leq 0$ is never satisfied. Consider now the situation following an execution of line 7 after this point. Suppose that at that time

$l = l^+$ and $T^{l^+} = ((h_1, h_2, \ldots, h_{M^+}), H_1)$, with $M^+ \geq 1$ and $\{h_1, h_2, \ldots, h_{M^+}\} \subseteq G^+$. From the condition on Z_2 in a cyclic algorithm, we get that $f^{l^+} = h_1$. Also, if we define G to be the set $\{h_1, h_2, \ldots, h_{M^+}\}$, then $G \in \mathbb{G}$ (recall that $\mathbb{G}$ is closed). Note that $M_G = M^+$ and $N_G = N_{G^+}$.

We are going to show that there is a G-path P of the cyclic sequential algorithm A such that $c(P) = \left(g^k\right)_{k \in Z_\geq}$, $s(P) = \left(x^k\right)_{k \in \mathbb{Z}_\geq}$, $d(P) = \left(S^k\right)_{k \in \mathbb{Z}_\geq}$ and, for all $k \in Z_\geq$, $g^k = f^{l^+ + k}$, $x^k = y^{l^+ + k}$ and $F\left(S^k\right) = F\left(T^{l^+ + k}\right)$, where $F(S)$ is the first component of $S = (F, H)$. We prove this by inductively constructing a G-path P that satisfies the claim for all $k \in Z_\geq$.

The input to algorithm A is G. Since in algorithm A, $Z_0 = \mathbb{R}^{N_G}$, it is possible for line 1 to return $x^0 = y^{l^+}$. Also, since in the cyclic algorithm A, $Z_1 = \{(g_1, g_2, \ldots, g_{M_G}) \mid \{g_1, g_2, \ldots, g_{M_G}\} = G\} \times Z_{\mathbb{H}}$, it is possible for line 2 to return $S^0 = ((h_1, h_2, \ldots, h_{M_G}), H_2)$, where H_2 has the property that $Z_4((F_2, H_2), k, x, g) = Z_4((F_1, H_1), l^+ + k, x, g)$ for all $F_1, F_2 \in \mathbb{F}$, $k \in \mathbb{Z}_\geq$, $x \in \mathbb{R}^{N_G}$, and $g \in G$; see (2.14). Therefore at this moment, and even after the execution of line 5, $F\left(S^0\right) = F\left(T^{l^+}\right) = (h_1, h_2, \ldots, h_{M_G})$ and by the condition on Z_2, we have that, after the execution of line 7, $g^0 = f^{l^+} = h_1$. This shows that the base of the induction is true.

Now assume that the induction hypothesis (namely that after the execution of line 7, $g^k = f^{l^+ + k}$, $x^k = y^{l^+ + k}$ and $F\left(S^k\right) = F\left(T^{l^+ + k}\right)$) holds, for some $k \in Z_\geq$, and carry on constructing P so that the induction hypothesis holds for $k+1$ after the next execution of line 7. We make the observation that, since neither A nor A^+ ever changes the second component of the state, we have that $S^k = \left(F^k, H_2\right)$ and $T^{l^+ + k} = \left(F^k, H_2\right)$, where $F^k = F\left(S^k\right) = F\left(T^{l^+ + k}\right)$. Because $f^l\left(y^l\right) \leq 0$ is not satisfied if $l \geq l^+$ in the execution of A^+, and $g^k = f^{l^+ + k}$, $x^k = y^{l^+ + k}$, it must be the case that $g^k\left(x^k\right) > 0$. Hence, the execution of A goes to line 13. Since we have $Z_4\left(\left(F^k, H_2\right), k, x^k, g^k\right) = Z_4\left(\left(F^k, H_1\right), l^+ + k, y^{l^+ + k}, f^{l^+ + k}\right)$, and Z_4 is the same for the two algorithms, it is possible to make the choice in line 13 of A so that we get

$x^{k+1} = y^{l^+ + k+1}$. In line 14, by the common definition of Z_5, $F\left(S^k\right) = F\left(T^{l^+ + k}\right)$ results in $F\left(S^{k+1}\right) = F\left(T^{l^+ + k+1}\right)$. Since, in both A and A^+, $Condition()$ returns false, the execution returns back to line 7. In line 7, by the definition of Z_2 in the definition of a cyclic sequential algorithm, $F\left(S^{k+1}\right) = F\left(T^{l^+ + k+1}\right)$ results in $g^{k+1} = f^{l^+ + k+1}$. This completes our inductive construction and proof.

However, by the assumption that the cyclic sequential algorithm A on $\mathbb{G}$ is finitely convergent, we know that the solution sequence $\left(x^k\right)_{k \in Z_{\geq}}$ of every G-path P of A is finitely convergent to a feasible point. It follows that there exists a $K \in \mathbb{Z}_{\geq}$ such that, for all $k \geq K$, $g^k\left(x^k\right) \leq 0$ is satisfied for all $g \in G$. This contradicts the fact that $f^{l^+ + k}(y^{l^+ + k}) \leq 0$ is never satisfied for $k \geq 0$. ■

We define algorithm A^0 for $G = \{g_1, g_2, \ldots, g_{M_G}\} \in \mathbb{G}$ for a given cyclic algorithm A by the following specifications:

- $\mathbb{S}^0 = \mathbb{H}$.
- $Z_0^0 = \mathbb{R}^{N_G}$.
- $Z_1^0 = Z_{\mathbb{H}}$ (this is the same $Z_{\mathbb{H}}$ that is used in A).
- $Z_2^0(H,k,x) = \{g_1, g_2, \ldots, g_{M_G}\}$.
- $Z_3^0(H,k,x,g) = \{H\}$.
- $Z_4^0(H,k,x,g) = Z_4((F,H),k,x,g)$, for any $F \in \mathbb{F}$; see (2.13).
- $Z_5^0(H,k,x,g) = \{H\}$.
- $Condition^0(S) = false$.

As a consequence of the definition of $Z_2^0(H,k,x)$, given any $G \in \mathbb{G}$, every element of $\left\{\left(g^k\right)_{k \in \mathbb{Z}_{\geq}} \mid g^k \in G, \text{ for } k \in \mathbb{Z}_{\geq}\right\}$ is a control sequence of a G-path of A^0. Therefore, it is easy to see that A^0 is a generalization of A.

Lemma 2.4.2 *If A is a cyclic sequential algorithm on a problem set $\mathbb{G}$, then sequential algorithm A^0 is a generalization of the sequential algorithm A^+.*

Proof We need to prove that, for every $G \in \mathbb{G}$ and every G-path P^+ of A^+, there exists a G-path P^0 of A^0 equivalent to it. We use the notation $c(P^+) = \left(f^l\right)_{l \in Z_\geq}$, $s(P^+) = \left(y^l\right)_{l \in Z_\geq}$, and $d(P^+) = \left(T^l\right)_{l \in Z_\geq}$. We are going to show that there is a G-path P^0 of the sequential algorithm A^0 with $c\left(P^0\right) = \left(g^k\right)_{k \in Z_\geq}$, $s\left(P^0\right) = \left(x^k\right)_{k \in \mathbb{Z}_\geq}$, $d\left(P^0\right) = \left(S^k\right)_{k \in \mathbb{Z}_\geq}$ such that, for all $k \in Z_\geq$, $g^k = f^k$, $x^k = y^k$ and $S^k = H\left(T^k\right) = H\left(T^0\right)$, where $H(T)$ is the second component of $T = (F,H)$. We do this by inductively constructing a G-path P^0 that satisfies the claim for all $k \in Z_\geq$. First, it is possible for A^0 to pick $x^0 = y^0$ in line 1 (since $Z_0^0 = \mathbb{R}^{N_G}$), $S^0 = H\left(T^0\right)$ in line 2 (since $H\left(T^0\right) \in Z_\mathbb{H}$) and $g^0 = f^0$ in line 7 (since $Z_2^0(H,k,x) = \{g_1, g_2, \ldots, g_{M_G}\}$). Now we assume the induction hypothesis (namely that after the execution of line 7, $g^k = f^k$, $x^k = y^k$ and $S^k = H\left(T^k\right) = H\left(T^0\right)$) holds, for some $k \in Z_\geq$, and carry on constructing P^0 so that the induction hypothesis holds for $k+1$ after the next execution of line 7. Whether or not the condition in line 8 is satisfied, line 10 and line 13 can return $x^{k+1} = y^{k+1}$. Line 11 and line 14 will return $S^{k+1} = H\left(T^{k+1}\right) = H\left(T^0\right)$ because $S^{k+1} = S^k$ and $H\left(T^{k+1}\right) = H\left(T^k\right)$. After that, whether or not A^+ goes back to line 4 or line 6, $S^{k+1} = S^0 = H\left(T^0\right) = H\left(T^{k+1}\right)$. Also it is possible for A^0 to return $g^{k+1} = f^{k+1}$ in line 7. This completes our inductive construction and proof. ■

Theorem 2.4.3 *Let $\mathbb{G}$ be a closed problem set and A be a cyclic sequential algorithm on $\mathbb{G}$ such that the sequential algorithm A^0 on $\mathbb{G}$ is finitely convergent under repetitive control, then A^+ on $\mathbb{G}$ is finitely convergent.*

Proof The sequential algorithm A^0 on $\mathbb{G}$ is finitely convergent under cyclic control, which is a special case of repetitive control. Therefore the cyclic sequential algorithm A on $\mathbb{G}$ is finitely convergent, since the control sequence of any G-path of a

cyclic sequential algorithm is cyclic. By Theorem 2.4.1, the control sequence of any G-path of A^+ for $G \in \mathbb{G}$ is repetitive. By the assumption of this theorem that the sequential algorithm A^0 on $\mathbb{G}$ is finitely convergent under repetitive control and Lemma 2.4.2, the algorithm A^+ on $\mathbb{G}$ is finitely convergent. ■

Consider now the cyclic sequential algorithm CART3, which is equivalent to ART3 as defined in Subsection 2.2.1. The algorithm CART3^0 is exactly the algorithm GART3 and the algorithm CART3$^+$ is equivalent to the algorithm ART3+ of [37]. By Theorem 2.3.1, GART3 is finitely convergent on $\mathbb{G}$ under repetitive control. Hence, it follows from Theorem 2.4.3 that CART3$^+$ (and, equivalently, ART3+) is finitely convergent on $\mathbb{G}_A$. Thus the main mathematical result of [37] is an immediate consequence of the general theory that is put forth in this dissertation.

However, in those cases when the sequential algorithm A^0 on $\mathbb{G}$ is known to be finitely convergent only under almost cyclic control, we cannot conclude based on Theorem 2.4.3 the finite convergence of A^+ on $\mathbb{G}$. We now present an alternate procedure that produces, for any cyclic sequential algorithm A and any $i_0 \in \mathbb{Z}_>$, a sequential algorithm $A^{++}(i_0)$ for which we can prove a theorem analogous to Theorem 2.4.3, but with the less restrictive premise.

The procedure consists of replacing $\mathbb{S}, Z_1, Z_2, Z_3, Z_4, Z_5$ and $Condition(S)$ in algorithm A with:

- $\mathbb{S}^{++} = \mathbb{F} \times \mathbb{H} \times \mathbb{Z}_{\geq}$.
- $Z_1^{++} = \{(g_1, g_2, \ldots, g_{M_G}) \mid \{g_1, g_2, \ldots, g_{M_G}\} = G\} \times Z_{\mathbb{H}} \times \{0\}$.
- $Z_2^{++}(((h_1, h_2, \ldots, h_U), H, i), k, x) = \{h_1\}$.
- $Z_3^{++}(((h_1, h_2, \ldots, h_U), H, i), k, x, g) = \{((h_2, \ldots, h_U), H, i+1)\}$.
- $Z_4^{++}(((h_1, h_2, \ldots, h_U), H, i), k, x, g) = Z_4(((h_1, h_2, \ldots, h_U), H), k, x, g)$.
- $Z_5^{++}(((h_1, h_2, \ldots, h_U), H, i), k, x, g) = \{((h_2, \ldots, h_U, h_1), H, i+1)\}$.

- $Condition^{++}(S) = \begin{cases} true, & \text{if } S \text{ is of the form } (\mathbf{f}, \emptyset, i) \\ & \text{with either } \mathbf{f} = () \text{ or } i > i_0, \\ false, & \text{otherwise.} \end{cases}$

If we plug in these specifications into the algorithm CART3 that has been specified earlier, we get that the algorithm CART3$^{++}(i_0)$, for input $G \in \mathbb{G}_A$ is the following.

1. $x^0 \leftarrow \sigma\left(\mathbb{R}^{N_G}\right)$
2. $S^0 \leftarrow \sigma\left(\{((g_1, g_2, \ldots, g_{M_G}), \emptyset, 0) \,|\, \{g_1, g_2, \ldots, g_{M_G}\} = G\}\right)$
3. $k \leftarrow 0$
4. Repeat
5. $\quad S^k \leftarrow S^0$
6. $\quad$ Repeat
7. $\quad\quad g^k \leftarrow h_1$, if $S^k = ((h_1, h_2, \ldots, h_U), \emptyset, i)$
8. $\quad\quad$ If $g^k\left(x^k\right) \leq 0$
9. $\quad\quad\quad$ then
10. $\quad\quad\quad\quad x^{k+1} \leftarrow x^k$
11. $\quad\quad\quad\quad S^{k+1} \leftarrow ((h_2, \ldots, h_U), \emptyset, i+1)$, if $S^k = ((h_1, h_2, \ldots, h_U), \emptyset, i)$
12. $\quad\quad\quad$ else
13. $\quad\quad\quad\quad x^{k+1} \leftarrow p_{g^k}\left(x^k\right)$
14. $\quad\quad\quad\quad S^{k+1} \leftarrow ((h_2, \ldots, h_U, h_1), \emptyset, i+1)$, if $S^k = ((h_1, h_2, \ldots, h_U), \emptyset, i)$

15. $k \leftarrow k+1$

16. Until S^k is of the form $(\mathbf{f}, \emptyset, i)$ with either $\mathbf{f} = ()$ or $i > i_0$

In this algorithm, i keeps count of the number of executions of line 7 in the inner loop (irrespective of the truth of $g^k\left(x^k\right) \leq 0$, either line 11 or line 14 increases i by 1).

Theorem 2.4.4 *If $\mathbb{G}$ is a problem set, A is a cyclic sequential algorithm on $\mathbb{G}$, $G \in \mathbb{G}$, $i_0 \in \mathbb{Z}_>$ and $i_0 > M_G$, then the control sequence of every G-path of $A^{++}(i_0)$ is almost cyclic.*

Proof If line 5 is executed in any G-path of $A^{++}(i_0)$, then there can be at most i_0+1 consecutive executions of the inner loop (lines 7-15) before $Condition^{++}(S)$ is satisfied. This has to be followed by another execution of line 5, which sets S to $((g_1, g_2, \ldots, g_{M_G}), H, 0)$, with $\{g_1, g_2, \ldots, g_{M_G}\} = G$ and $H \in \mathbb{H}$. The following M_G executions of the inner loop result in all the constraints in G occurring as the next M_G elements in the control sequence (recall that $i_0 < M_G$). Therefore the control sequence $\left\{g^k\right\}_{k \in \mathbb{Z}_\geq}$ of the G-path of $A^{++}(i_0)$ is almost cyclic. ■

Lemma 2.4.5 *If A is a cyclic sequential algorithm on a problem set $\mathbb{G}$ and $i_0 \in \mathbb{Z}_>$, then A^0 is a generalization of the sequential algorithm $A^{++}(i_0)$.*

Proof The proof is similar to the proof of Lemma 2.4.2. ■

Theorem 2.4.6 *If $\mathbb{G}$ is a problem set, A is a cyclic sequential algorithm on $\mathbb{G}$ such that the sequential algorithm A^0 on $\mathbb{G}$ is finitely convergent under almost cyclic control, $G \in \mathbb{G}$, $i_0 \in \mathbb{Z}_>$ and $i_0 > M_G$, then the solution sequence of every G-path of $A^{++}(i_0)$ on $\mathbb{G}$ is finitely convergent.*

Proof Under the assumptions of the theorem, it follows from Theorem 2.4.4 that the control sequence of every G-path of $A^{++}(i_0)$ is almost cyclic. The result now

follows, since by Lemma 2.4.5 a solution sequence of a G-path of $A^{++}(i_0)$ is also a solution sequence of a G-path of A^0 on $\mathbb{G}$, which is assumed to be finitely convergent under almost cyclic control. ■

Our Theorem 2.3.1 regarding CART3^0 = GART3 and the theorem in [29] that implies that CMCSP^0 = GMCSP on $\mathbb{G}_B$ is finitely convergent under almost cyclic control can now be seen to provide the following.

Corollary 2.4.7 *If $G \in \mathbb{G}_A$ and $i_0 > M_G$, then the solution sequence of every G-path of CART3$^{++}(i_0)$ on $\mathbb{G}$ is finitely convergent. If $G \in \mathbb{G}_B$ and $i_0 > M_G$, then the solution sequence of every G-path of CMCSP$^{++}(i_0)$ on $\mathbb{G}$ is finitely convergent.*

2.5 Methods for Constrained Optimization Problems

While there are real-world applications that give rise to either linear or convex feasibility problems, many other applications are in the form of constrained optimization problems, see Chapter 5 of [55]. Mathematically, a constrained optimization problem set for a problem set $\mathbb{G}$ is determined by assigning to every $G \in \mathbb{G}$ an objective function $f_G : \mathbb{R}^{N_G} \to \mathbb{R}$. We say that $x \in \mathbb{R}^{N_G}$ is a solution of the constrained optimization problem (G, f_G) if x is feasible for G and, for every y that is feasible for G, $f_G(x) \leq f_G(y)$.

There are approaches to solving constrained optimization problems by solving a sequence of augmented feasibility problems, an example of which is the following. For a given (G, f_G) and $r \in \mathbb{R}$, define $h_{(G,f_G),r} : \mathbb{R}^{N_G} \to \mathbb{R}$ by $h_{(G,f_G),r}(x) = f_G(x) - r$ and $G_r = G \cup \{h_{(G,f_G),r}\}$. It is then easy to prove that x is a solution of the constrained optimization problem (G, f_G) if, and only if, there exists an $r^* \in \mathbb{R}$ such that x is feasible for G_{r^*} and, for all $s < r^*$, there is no $y \in \mathbb{R}^{N_G}$ that is feasible for G_s. As we illustrate below, such results can be turned into practical optimization algorithms, provided that we have a finitely convergent sequential algorithm on $\mathbb{G}$

that can be applied to the augmented versions of the problems in $\mathbb{G}$.

Our illustration is motivated by intensity modulated therapy planning. In that application the usual constraints result in having to deal with the problem set $\mathbb{G}_A$ of Subsection 2.2.1 and many objective functions that occur (such as minimization of the maximum dose delivered to an OAR) are of the form

$$f_G(x) = \max\{\langle b_1, x\rangle, \langle b_2, x\rangle, \ldots, \langle b_J, x\rangle\}, \tag{2.15}$$

where J is a positive integer and, for $1 \leq j \leq J$, $b_j \in \mathbb{R}^{N_G}$ and $||b_j|| \neq 0$. For any $r \in \mathbb{R}$, we define $G_r = G \cup \{h_{b_j,r} \mid 1 \leq j \leq J\}$, where $h_{b_j,r}(x) = \langle b_j, x\rangle - r$. Note that in this case the additional constraints in G_r are of the same type as the ones used in $\mathbb{G}_A$ and hence one can apply ART3+ to attempt finding a point in G_r. Again it is the case that x is a solution of the constrained optimization problem (G, f_G) if, and only if, there exists an $r^* \in \mathbb{R}$ such that x is feasible for G_{r^*} and, for all $s < r^*$, there is no $y \in \mathbb{R}^{N_G}$ that is feasible for G_s.

To turn these observations into an algorithm for solving the constrained optimization problem (G, f_G), we assume that there exists an r_{min} such that there is no feasible point for $G_{r_{min}}$, and that such an r_{min} is available to us. In Subsection 3.2.3 we explain why such an assumption is valid and how to find r_{min} in practice. A small positive real number ε is specified as the desired optimality tolerance. We also use the parameter K to denote the maximum number of iterations in any execution of ART3+.

The algorithm ART3+O is the following procedure:

Initialization. Apply ART3+ to G. If a solution is found in K or less iterations, we assign that solution to x^0, and let $r_{max} = f_G(x^0)$ and $i = 0$. Otherwise, we consider the optimization problem unsolvable.

ART3+. Let $r^i = (r_{min} + r_{max})/2$. Apply ART3+ to G_{r^i} starting with x^i.

1. If ART3+ finds a point that is feasible for G_{r^i} in K or less iterations, then assign that point to x^{i+1} and x^*, and let $r_{max} = f_G(x^{i+1})$.
2. Otherwise, let $r_{min} = r^i$ and x^{i+1} be the Kth iterate of ART3+.

Termination. If $r_{max} - r_{min} \leq \varepsilon$, then return x^* as the solution to the optimization problem, otherwise increase i by 1 and repeat ART3+.

The idea of the algorithm is to search for the r^* as defined below (2.15). We first find a range $[r_{min}, r_{max}]$ that is big enough to contain r^*. After this we apply a bisection search on the decreasing range $[r_{min}, r_{max}]$ until the length of this range is ε or less, which will happen for an $i \leq 2\lceil \log_2(r_{max} - r_{min})/\varepsilon \rceil$. ART3+ will return a solution x^* for which $f_G(x^*) - r^* \leq \varepsilon$, provided that, in each call to it, ART3+ stops in at most K iterations for the problem G_r that has a solution.

To achieve the same goal using interior point method, primal simplex method and dual simplex method in MOSEK, we have to take r as an auxiliary variable and solve the LP problem: minimize r, subject to the constraints in G_r.

We note by the way that, by using CMCSP$^{++}(i_0)$ instead of ART3+, the same strategy can be applied to solve convexly constrained optimization problems (G, f_G) with $G \in \mathbb{G}_B$ (see Subsection 2.2.2) and f_G a convex function.

Chapter 3

Experiments

There are many applications of the linear interval feasibility problem. In Section 3.1 we consider the application of conversion of pixel images to blob images, i.e., the finding of the blob coefficients to fit an image represented by pixel coefficients. We carry out a comprehensive study that assigns statistical significance to claims of superiority. Another important application is IMRT planning. In Section 3.2 we provide evidence for the claims that the efficiency of ART3+ is better than that of ART3 when applied to problems arising from IMRT and that the performance of ART3+O is much better than that of the interior point algorithm and the simplex algorithms as implemented in the state of art optimization software package MOSEK.

3.1 Experiments with Fitting a Pixel Image by a Blob Image

We first introduce the problem formulation, then explain the experimental setup, and finally report on the comparison results of applying CART3 (which is equivalent to ART3 in [35]), CART3$^+$ (which is equivalent to ART3+ in [37]) and

CART3$^{++}(i_0)$ to solving such problems. The experiments used the software package SNARK09 [28].

A $J \times J$ digitized image is one whose value in the interior of any pixel of a J^2-element grid is uniform. Sometimes alternative representations of an image are superior. For example, in image reconstruction from projections [36], we use blob basis functions in some series expansion methods to reduce artifacts in the reconstruction. Such a reduction is due to the fact that blob basis functions are smoother than pixel basis functions [44].

Assume that we have a set of M basis functions $\{q_1, q_2, \ldots, q_M\}$, whose linear combinations give us adequate approximations to images L. An example of such an approach is the $J \times J$ digitization. In that case $M = J^2$. We number the pixels from 1 to J^2, and define the J^2 basis functions by

$$q_j(r,\phi) = \begin{cases} 1, \text{ if } (r,\phi) \text{is inside the } j\text{th pixel,} \\ 0, \text{ otherwise,} \end{cases} \tag{3.1}$$

where (r,ϕ) are the polar coordinates. Then the $J \times J$ digitization of an image L is the image $\hat{L}$ defined by

$$\hat{L}(r,\phi) = \sum_{j=1}^{J^2} y_j q_j(r,\phi), \tag{3.2}$$

where y_j is the average value of L inside the jth pixel.

An alternative is to use blob basis functions [44]. Let

$$b_{a,\alpha,\delta}(r,\phi) = \begin{cases} C_{a,\alpha,\delta}\left(1-\left(\frac{r}{a}\right)^2\right) I_2\left(\alpha\sqrt{1-\left(\frac{r}{a}\right)^2}\right), \text{ if } 0 \le r \le a, \\ 0, \text{ otherwise,} \end{cases} \tag{3.3}$$

where I_m is the modified Bessel function of the first kind of order m, a is the non-

negative radius of the blob, α is a nonnegative real number that controls the blob's taper (shape) and $C_{a,\alpha,\delta}$ is the multiplying constant:

$$C_{a,\alpha,\delta} = \frac{\sqrt{3}\delta^2\alpha}{4\pi a^2 I_3(\alpha)}. \tag{3.4}$$

It is easy to see that the function defined above is circularly symmetric since its value does not depend on ϕ, and it is a smooth function since its first derivatives are continuous everywhere. For the purpose of this discussion, the parameters a, α and δ, and hence the function $b_{a,\alpha,\delta}$, are fixed. Consider a hexagonal grid (with spacing determined by δ), and let N be the number of grid points that cover the image region. Then, we set the number M of basis function to N and, for $1 \leq n \leq N$, the blob basis function b_n is obtained from $b_{a,\alpha,\delta}$ by shifting it in the plane so that its center is moved from the origin to the grid point n, and a blob image is

$$\sum_{n=1}^{N} x_n b_n(r,\phi), \tag{3.5}$$

where the x_n is called the nth blob coefficient.

For some applications (e.g., [36]), we sometimes need to convert a pixel image into a blob image. For this we need to solve J^2 linear interval inequalities:

$$y_j + \gamma_j \leq \sum_{n=1}^{N} x_n b_n(r_j,\phi_j) \leq y_j + \zeta_j, \text{ for } 1 \leq j \leq J^2, \tag{3.6}$$

where (r_j,ϕ_j) are the polar coordinates of the center of pixel j, y_j is the pixel value of pixel j, and $\gamma_j \leq 0$ and $\zeta_j \geq 0$ have some appropriately chosen values. For such a problem G, $M_G = J^2$, $N_G = N$, and $G \in \mathbb{G}_A$, provided that the solution space is full-dimensional. Therefore we can solve the problem G with CART3, CART3$^+$and CART3$^{++}(i_0)$, as long as $i_0 > M_G$.

3.1.1 The Brain Phantom

A phantom is put together by superimposing a number of elemental objects, placed at desired positions, at desired orientations and of desired size and density. The density of the image at any point is then defined as the sum of the densities associated with all the elemental objects within which the point lies.

The phantoms we used in the experiments are all simulations of a cross-section of a human head that was reconstructed by CT [36]. Based on this cross-section we specified a skull enclosing the brain with ventricles, two small tumors, and a hematoma (blood clot) using five ellipses, eight segments of circles, and two triangles. We also simulated the occurrence of a textured large tumor and of inhomogeneity in our phantom. This was done by adding to the list of elemental objects a much longer list of additional elemental objects, each coinciding exactly with one pixel. To simulate random inhomogeneity, the densities assigned to these objects were randomly selected from a zero-mean Gaussian distribution. Also, SNARK09 enables us to add to the list of elemental objects pairs of objects with a specified probability P of assigning density d_1 to the left object and density d_2 to the right object. Thus we can randomly generate infinitely many similar, but not identical, brain phantoms.

We used SNARK09 [28] to obtain the density in each of 243×243 pixels of size 0.0752 cm, for each randomly generated phantom. So $M_G = J^2 = 243 \times 243 = 59,049$. We used a hexagonal grid that resulted in blob images with $N_G = N = 51,152$.

In order to see clearly the features in the interior of the skull, we use zero (black) to represent the value 0.204 (or anything less) and 255 (white) to represent the value 0.21675 (or anything more). This way the small change in density by 0.001 corresponds to a change of twenty in display grayness, which is visible. The images we displayed below using this method are referred to as windowed images. Figure

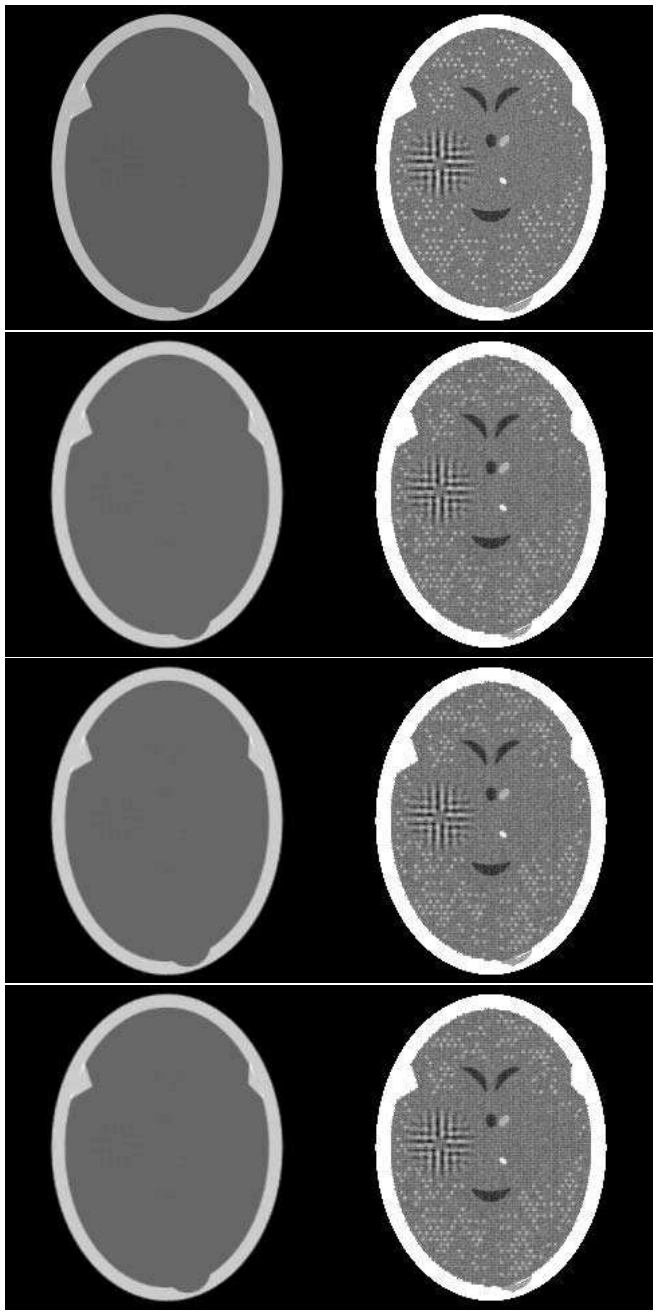


Figure 3.1: Row 1 contains a phantom and its windowed image. Rows 2, 3 and 4 contain the blob images and their corresponding windowed images provided for the phantom by CART3, CART3^{+} and CART3^{++}, respectively. The differences between the blob images produced by the three algorithms are so small that they cannot be identified based on such displays.

3.1 shows one of the phantoms together with its windowed image and the blob images and their corresponding windowed images as provided for that phantom by CART3, CART3$^+$ and CART3^{++}, respectively. Figure 3.2 shows the blob images as solved by CART3 and CART3$^+$ for another one of the phantoms, and the plots of the 131th columns of the blob images solved by CART3 and CART3$^+$, respectively, against that of that phantom.

3.1.2 Task-Oriented Comparisons of Algorithm Performance

The task was to convert brain phantoms into their blob representations. We did four experiments: two pair-wise comparisons between CART3 and CART3$^+$ and between CART3 and CART3^{++}. We ran each pair twice with different order to avoid bias due to caching. In each experiment, using current time as seed for the random number generator, 30 phantoms were generated and were solved by both algorithms that were being compared. The timings were averaged for the 30 runs. SNARK09 also reported on the significance (as measured by the P-value) for rejecting the null hypothesis that the two algorithms are equally fast in favor of the alternative hypothesis that the one with faster average performance is in fact faster.

All of the three algorithms CART3, CART3$^+$ and CART3^{++} initialize x^0 to be the vector all of whose components are zero (see line 1 of CART3 on page 33) and all of them initialize S^0 to use an efficient ordering (in the sense advocated in [39]) of the constraints associated with the pixel indices (see line 2 of CART). The variant of CART3^{++} that we used in our experiments is CART3$^{++}(i_0)$ with $i_0 = M_G + 70{,}000 = 59{,}049 + 70{,}000 = 129{,}049$, see Section 2.4. Due to the careful choice of the tolerances γ_j and ζ_j, for $1 \leq j \leq J^2 = M_G$, all the algorithm runs converged to a feasible solution.

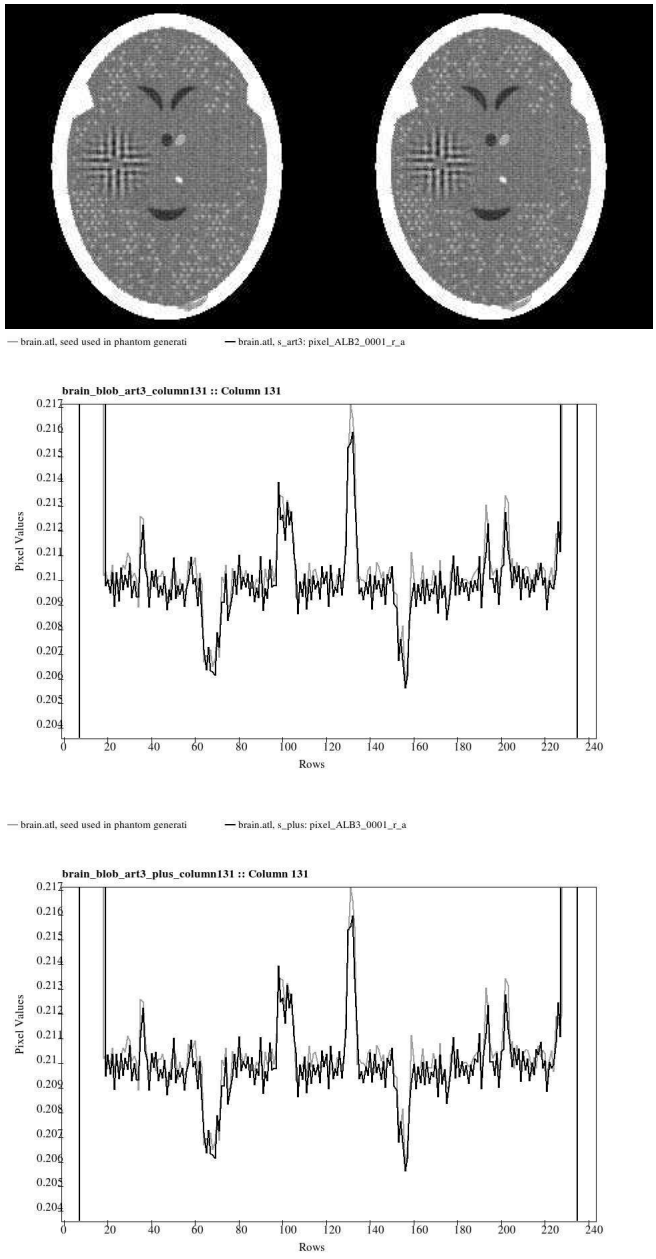


Figure 3.2: The first row contains windowed images of blob images provided by CART3 and by CART3$^+$ for a phantom different from that of Figure 2. The second and third rows, respectively, show (dark) plots of the 131th columns of the blob images provided by CART3 and CART3$^+$ against that of the phantom (light). The differences between the blob images produced by CART3 and CART3$^+$ are so small that they cannot be identified based on such displays.

	CART3 (s)	CART3$^+$ (s)	CART3^{++} (s)	Ratio	P-value
CART3 first	39.0	26.5		1.5	3.5×10^{-11}
CART3$^+$ first	37.7	24.9		1.5	2.3×10^{-13}
CART3 first	39.5		24.6	1.6	1.4×10^{-13}
CART3^{++} first	35.6		25.0	1.4	3.6×10^{-13}

Table 3.1: The average timings of the 30 runs.

The experiments were conducted on an Intel Xeon 1.7 GHz processor, 1 Gbyte memory, 32 bit workstation. Table 3.1 gives the average timings of 30 runs for both algorithms, the speedup ratios, and the P-values (which indicate that the null hypothesis of equal average performance can be rejected with extreme significance) for each experiment.

3.2 Experiments with IMRT Planning

In this section we report on experiments that we carried out for the following publications. In [37] we compared the performances of ART3+ and ART3 in the IMRT application, which involves finding a point in the intersection of a large number of hyperslabs; see (2.3). More recently, in [17], we compared the performances of ART3+ and LP in a more complicated IMRT application that also involved the so-called dose-volume constraints (DVCs). Lastly, we compared the performances of ART3+O with an interior point algorithm and with simplex algorithms on convex constrained optimization problems in IMPT planning.

We have given the formal definition of CART3 and CART3$^+$ in Chapter 2. Since CART3 is equivalent to ART3 of [35] (and also to ART3 as defined on page 26) and CART3$^+$ is equivalent to ART3+ of [37], we use in this section only the terms common in the open literature, namely ART3 and ART3+.

3.2.1 Experiments where ART3+ is More Efficient than ART3

We are going to show that the problem G that arises from the IMRT application satisfies the first of the two conditions for $\mathbb{G}_A$. Hence ART3 and ART3+ are applicable to such a G and, if G also satisfies the second condition for $\mathbb{G}_A$, then both ART3 and ART3+ converge finitely for G.

Consider Figure 3.3. The MLC allows us to control independently the intensity of radiation x_n in each beamlet n, for $1 \leq n \leq N$. Because of the 3D nature of the practical problems, we consider the region into which radiation is delivered as divided into voxels (volume elements) rather than pixels (picture elements). For each voxel j, $1 \leq j \leq J$, let a_j be the N-dimensional real-valued vector whose nth component is the dose contributed to voxel j if the intensity of radiation in beamlet n is unity, and the intensity of radiation in all other beamlets is zero. The total dose deposited into voxel j, is $\langle a_j, x \rangle = \sum_{n=1}^{N} a_{jn} x_n$.

Roughly speaking, the treatment planning proceeds as follows. The grid shown in Figure 3.3 is superimposed on an image (produced, for example, by CT) of a cross-section of the patient and the oncologist identifies the PTVs and the OARs. For voxels j in the PTVs the total deposited dose must be above a lower bound l_j (to insure destruction of the tumor) and, without loss of generality, we may also state that it must be below an upper bound u_j (an extreme amount of dose should not be delivered anywhere). For voxels j in the OARs, the total deposited dose must be below an upper bound u_j (so that the organ is not harmed beyond self-recovery) and must be above a lower bound l_j (zero, since dose is by necessity nonnegative). This way we get J hyperslabs in the form of the set (2.3): $Q_j = \left\{ x \in \mathbb{R}^N \,|\, l_j \leq \langle a_j, x \rangle \leq u_j \right\}$, for $1 \leq j \leq J$. In addition, we must have that, for $1 \leq n \leq N$, $x_n \geq 0$, since we cannot deliver a negative intensity by a beamlet. Again, without loss of generality, we may assume an upper bound c on the x_n, for $1 \leq n \leq N$, and so the nonnegative constraints on x can also be formulated as hyperslabs in

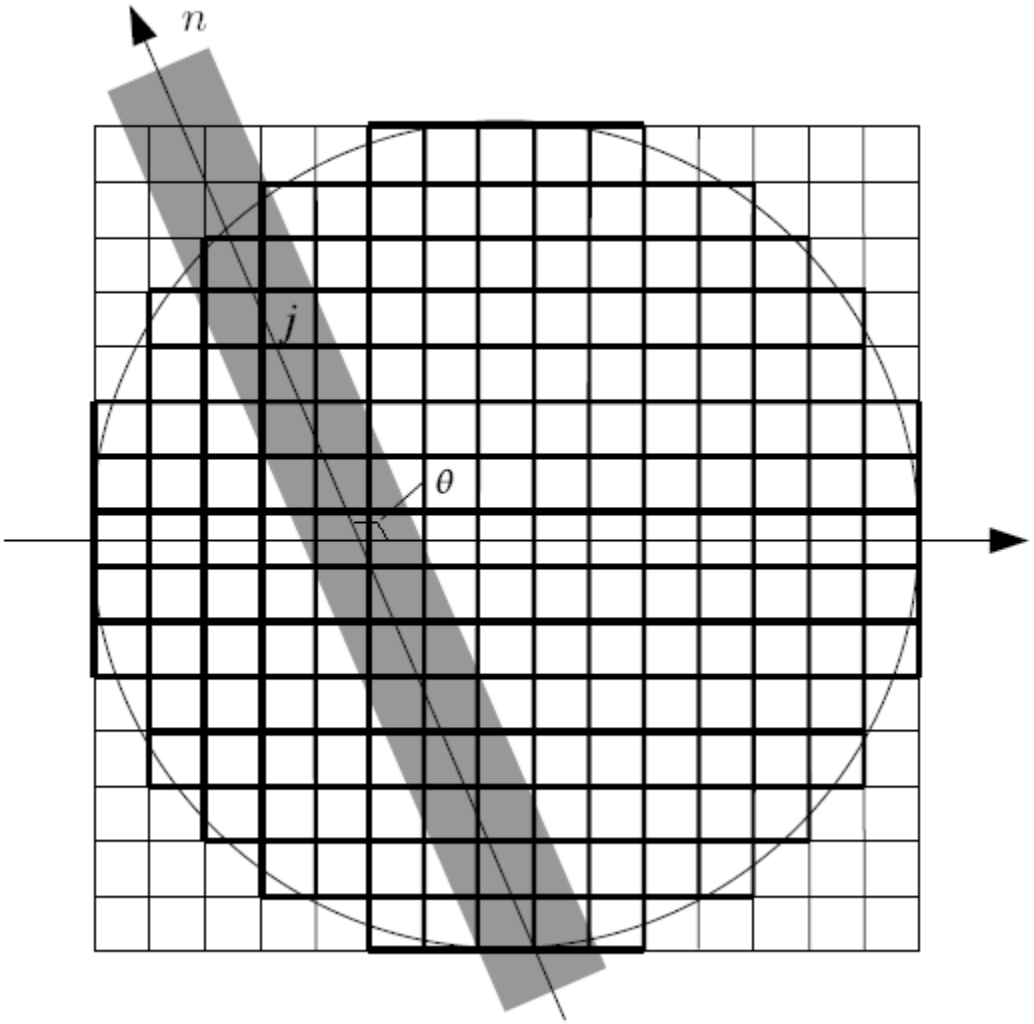


Figure 3.3: The indicated circle encloses the 2D cross-section of the patient's body. The grid defines a subdivision into voxels j, for $1 \leq j \leq J$, these voxels are indicated by the heavy edges. The gray bar indicates the extent of beamlet n, $1 \leq n \leq N$. The angle θ is between the direction of the beamlet and a fixed direction (in our case the positive horizontal axis).

$\mathbb{R}^N$. Let $N_G = N$, $M_G = J + N$ and we end up with a hyperslab intersection problem G as discussed in Subsection 2.2.1. To bring this discussion into alignment with the notation used in Subsection 2.2.1, we note that each hyperslab Q_m ($1 \leq m \leq M_G$) is of the form (2.3) and thus gives rise to a function $g_m : \mathbb{R}^{N_G} \to \mathbb{R}$ defined by (2.2) in such a way that $Q_{g_m} = Q_m$. When $Q = \bigcap_{m=1}^{M_G} Q_m$ is full-dimensional, then we have a problem $G = \{g_1, g_2, \ldots, g_{M_g}\} \in \mathbb{G}_A$ and the task of finding an $x \in Q$ can be solved by both ART3 and ART3+ in a finite number of steps.

For both algorithms we always choose x^0 to be the vector of all zeros and the ordered set $S^0 = (g_1, g_2, \ldots, g_{M_G})$ is determined by the J hyperslabs associated with the voxels in Figure 3.3 in the "natural ordering" (row-by-row and column-by-column within each row) followed by the N hyperslabs representing the nonnegativity constraints on the beamlets.

For our experiments, the square region in Figure 3.3, is subdivided into 405×405 voxels of size 1 mm^2. The circle identifies 128,153 voxels (i.e., $J = 128,153$). We always use the same five beam directions: 0°, 72°, 144°, 216°, 288°. For each direction, there are 103 beamlets of width 4 mm, with the center of the center beamlet going through the center of the square region. Therefore $N = 5 \times 103 = 515$ is the total number of beamlets and $M_G = 128,668$.

For $1 \leq n \leq N$ and $1 \leq m \leq J$, we set the nth component of a_m to 1 if the center of the mth voxel is within the nth beamlet, and to 0 otherwise. The same value of a_m is assigned for voxel m for $1 \leq m \leq J$ in the experiments in the next subsection. For $J + 1 \leq m \leq M_G$, the nth component of a_m is 1 if $m = J + n$ and is 0 otherwise. Also, for these values of m, $l_m = 0$ and $u_m = c = 10$.

The way our experiments differ from each other is in the choice of l_m and u_m for $1 \leq m \leq J$. The values of l_m and u_m for $1 \leq m \leq J$ can be represented as gray-value images of the same size as illustrated in Figure 3.3. This is what we do to report on our experiments: with each we associate four images, the top-left represents u_m,

the top-right l_m, the bottom-left the dose distribution produced by ART3, and the bottom-right the dose distribution produced by ART3+, see Figure 3.4-3.6. To be precise, for $1 \leq j \leq J$, the grayness in the jth voxel of one of these bottom images is determined by $\langle a_j, x \rangle$, where x is the output of the algorithm. In all cases, the treatment plan was defined by the u and l such that Q was full-dimensional, and so both ART3 and ART3+ are finitely convergent.

Experiment 1. The treatment plan for this experiment (see Figure 3.4) was suggested to us by James M. Galvin, D.Sc. from Department of Radiation Oncology, Thomas Jefferson University, Philadelphia. It has a small PTV on the left and a big OAR on the right. For the voxels j whose centers are in the PTV, we set $l_j = 9$ (this in practice would be 9 Gy), and for the voxels j whose centers are in the OAR, we set $u_j = 2.5$.

Experiment 2. The treatment plan for this experiment (see Figure 3.5) was adopted from the 2002 AAPM (American Association of Physicists in Medicine) poster of "Experience with an IMRT Head and Neck QA Phantom"[1], Andrea Nelson et al., Department of Radiation Physics, The University of Texas M.D. Anderson Cancer Center. It has a big PTV on the left and a small PTV on the right, and an even smaller OAR lies in the concave mouth of the left PTV, which makes the planning hard. For the voxels j whose centers are in the left PTV, we set $l_j = 6.6$, for the voxels j whose centers are in the right PTV, we set $u_j = 5.4$, and for the voxels j whose centers are in the OAR, we set $u_j = 4.5$.

Experiment 3. The treatment plan for this experiment (see Figure 3.6) was adopted from the talk of "IMRT Optimization Based on Physical Criteria"[2], Thomas

[1] See http://rpc.mdanderson.org/RPC/Publications/2002 AAPM Posters/Nelson AAPM phantom presentation.pdf.

[2] See http://www.aapm.org/meetings/03SS/Presentations/Bortfield.pdf.

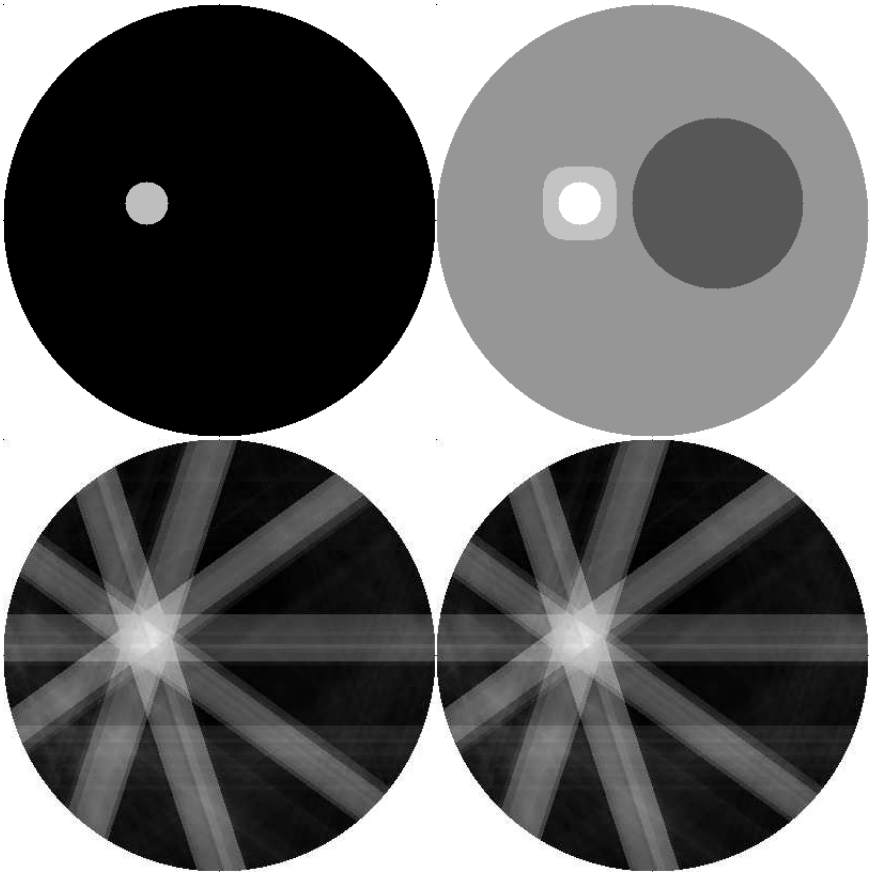


Figure 3.4: The first experiment.

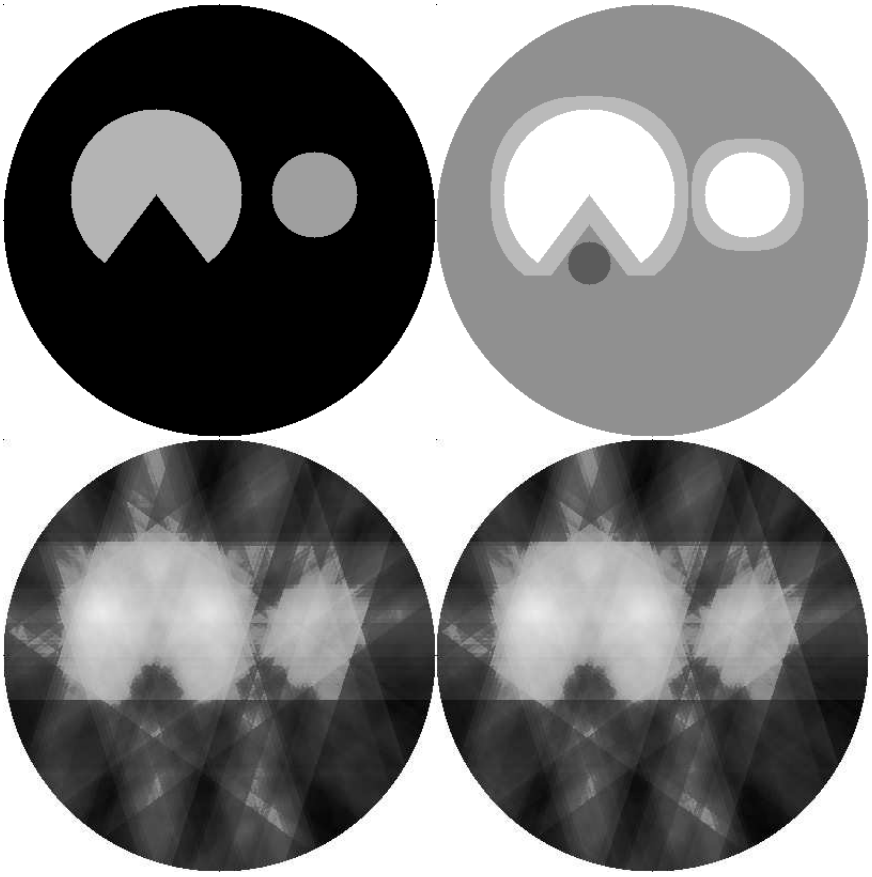


Figure 3.5: The second experiment.

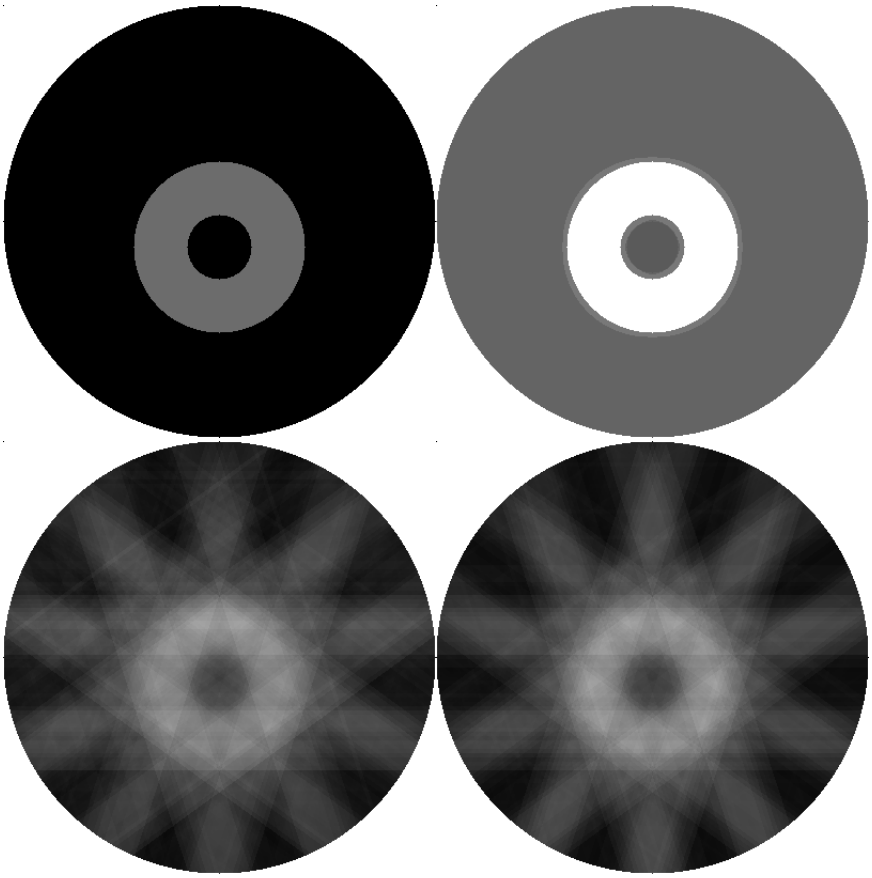


Figure 3.6: The third experiment.

Bortfeld et al., in the 2003 AAPM summer school on IMRT. It has a big ring shaped PTV and a small OAR right inside the PTV. For the voxels j whose centers are in the PTV, we set $l_j = 5.4$, and for the voxels j whose centers are in the OAR, we set $u_j = 4.5$.

The experiments were conducted using an Intel Xeon 1.7 GHz processor, 1 Gbyte memory, 32 bit workstation. (The same hardware was used for the experiments in the next subsection.) The total times needed by the two algorithms for each of the experiments are shown in Table 3.2. It can be seen that in these cases ART3 needed more than 1.45 times the time needed by ART3+ to get to a solution.

As the difficulty of the task increases (in the sense that Q is chosen to be smaller), so does the relative advantage of ART3+ over ART3 increase. This is

Algorithm	ART3 time (s)	ART3+ time (s)	ratio
Experiment 1	3.124	2.160	1.45
Experiment 2	64.996	33.270	1.95
Experiment 3	3.048	1.800	1.69

Table 3.2: Timing for finding solutions with the various algorithms.

u_j for OAR	ART3 time (s)	ART3+ time (s)	ratio
4.5	3.048	1.800	1.69
4.4	4.188	1.988	2.11
4.3	7.436	2.796	2.66
4.2	49.515	15.605	3.17

Table 3.3: Timing with decreasing upper bounds with the various algorithms.

illustrated in Table 3.3 for Experiment 3. As we decrease the upper bound for the voxels in the OAR from 4.5 to 4.2, the Q gets smaller and smaller and the ratio of the execution time of ART3 to that of ART3+ increases from 1.69 to 3.17.

The experiments have shown that ART3+ performs more efficiently than ART3.

3.2.2 Experiments where ART3+ is More Efficient than Linear Programming

In practice, the competing desires to destroy PTVs and not to harm OARs are likely to cause the intersection of the initially specified system of hyperslabs to be empty. A methodology in the presence of such infeasibility is provided by DVCs [9], which allows the lower bound (or upper bound) on a portion of the PTV (or OAR) to be lowered (or raised) by a certain amount. For example, the inequalities that characterize the hyperslabs for the OAR that consists of voxels whose indices are from the set O may be changed from

$$l_j \leq \langle a_j, x \rangle \leq u_j, \text{ for } j \in O, \tag{3.7}$$

to

$$l_j \leq \langle a_j, x \rangle \leq (1+\beta)\, u_j, \text{ for } j \in O, \tag{3.8}$$

and

$$\left| \left\{ j \in O \middle| \langle a_j, x \rangle > u_j \right\} \right| \leq \alpha \left| O \right|, \tag{3.9}$$

where $|O|$ stands for the cardinality (number of elements) of the set O. This amounts to allowing a specific portion (α) of the original inequalities to be violated, but only up to a specific fraction (β). If α and β are small enough, then delivering such doses to the voxels of this OAR should allow it to keep performing its function.

It is mathematically simpler to consider each PTV and each OAR simply as a subvolume. Suppose that there are D such subvolumes and O_d (for $d = 1, 2, \ldots, D$) is the index set of the voxels in subvolume d. We assume that, for some $1 \leq \overline{d} \leq D$, $O_{\overline{d}}$ is an OAR and that it is the only subvolume for which we have dose-volume constraints.

Under these assumptions the IMRT planning problem requires finding an $x \in \mathbb{R}^N$ such that

$$l_{\overline{d}} \leq \langle a_j, x \rangle \leq (1+\beta)\, u_{\overline{d}}, \text{ for } j \in O_{\overline{d}}, \tag{3.10}$$

$$\left| \left\{ j \in O_{\overline{d}} \middle| \langle a_j, x \rangle > u_{\overline{d}} \right\} \right| \leq \alpha |O_{\overline{d}}|, \tag{3.11}$$

$$l_d \leq \langle a_j, x \rangle \leq u_d, \text{ for all } j \in O_d, \text{ for } d \neq \overline{d}, 1 \leq d \leq D, \tag{3.12}$$

$$0 \leq x_n \leq c, \text{ for } n = 1, 2, \ldots, N. \tag{3.13}$$

Linear Programming. The LP method of [13] introduces, for each inequality j of the OAR $O_{\overline{d}}$, an auxiliary variable q_j that controls the amount by which the right-hand side of (3.10) goes above its initially prescribed upper bound $u_{\overline{d}}$. The LP task is formulated as follows:

$$\text{minimize } \sum_{j \in O_{\overline{d}}} q_j, \tag{3.14}$$

$$\text{subject to } l_{\overline{d}} \leq \langle a_j, x \rangle \leq q_j u_{\overline{d}}, \text{ for } j \in O_{\overline{d}}, \tag{3.15}$$

$$1 \leq q_j \leq 1+\beta, \text{ for } j \in O_{\overline{d}}, \tag{3.16}$$

$$\sum_{j \in O_{\overline{d}}} q_j \leq (1+\alpha\beta)\,|O_{\overline{d}}|, \tag{3.17}$$

$$l_d \leq \langle a_j, x \rangle \leq u_d, \text{ for all } j \in O_d, \text{ for } d \neq \overline{d}, 1 \leq d \leq D, \tag{3.18}$$

$$0 \leq x_n \leq c, \text{ for } n = 1, 2, \ldots, N. \tag{3.19}$$

The solution x of this LP problem will satisfy (3.10) (because it satisfies (3.15) and (3.16)), (3.12) and (3.13) (because these are the same as (3.18) and (3.19)). Given (3.15) and (3.16), (3.17) can be derived from (3.10) and (3.11) (by letting at most a fraction α of the q_j to be $1+\beta$ and the remaining q_j to be 1). The justification of the optimization goal (3.14) is that it pushes the solution set of the linear constraints (3.15)-(3.19) towards the solution set of (3.10)-(3.13).

In spite of this, there is still no guarantee that x satisfies (3.11); checking the solution x against (3.11) is thus required in the algorithm. However, the experiments show that, most of the time, solving the LP task gives us a solution that satisfies also (3.11). It turns out in our experiments that the optimization is not necessary most of the time and the LP problem can be solved much more efficiently without the optimization goal (3.14). In the experiments, our strategy is to solve the LP task first without the optimization goal and check if the solution satisfies (3.11); if not, then we solve the LP task again with the optimization goal and check again.

Our experiments use the COIN-OR Linear Program (CLP) Solver [46] to solve the LP task.

Algorithm Based on ART3+. If we ignore the constraint (3.9), the introduction of the DVCs does not change the way we build the problem G. Since ART3+ requires the solution set to be full-dimensional, we use it in the following manner to search a solution of (3.10)-(3.13). We generate a sequence of real numbers $\beta[v] \in (0, \beta]$, for $v = 0, 1, \ldots$. We then use ART3+ to attempt to solve

$$l_{\overline{d}} \leq \langle a_j, x \rangle \leq (1 + \beta[v]) u_{\overline{d}}, \text{ for } j \in O_{\overline{d}}, \tag{3.20}$$

$$l_d \leq \langle a_j, x \rangle \leq u_d, \text{ for all } j \in O_d, \text{ for } d \neq \overline{d}, 1 \leq d \leq D, \tag{3.21}$$

$$0 \leq x_n \leq c, \text{ for } n = 1, 2, \ldots, N. \tag{3.22}$$

If ART3+ does not converge within a prespecified number of steps, then we set $\alpha[v] = +\infty$. Otherwise we calculate and set

$$\alpha[v] = \left| \left\{ j \in O_{\overline{d}} \,|\, u_{\overline{d}} < \langle a_j, x \rangle \leq (1 + \beta[v]) u_{\overline{d}} \right\} \right| / \left| O_{\overline{d}} \right|. \tag{3.23}$$

If $\alpha[v] \leq \alpha$ then the solution of (3.20)-(3.22) is also a solution of (3.10)-(3.13) and we are done. Otherwise we repeat the process for $\beta[v+1]$.

This strategy does not guarantee that we find a solution to (3.10)-(3.13), even if there is one. However, the curve of $\alpha[v]$ plotted against $\beta[v]$ is typically "$\cup$-shaped" (see Figure 3.7) and hopefully the minimal $\alpha[v]$ we can find is smaller than the required α.

In the experiments, we use for ART3+ a simple recursive search strategy. The starting search range is set to be $[0, \beta]$. We then divide the search range into half, pick as that value of β that gives smallest $\alpha[v]$ among the middle point and the two endpoints, change the search range to the neighborhood of that $\beta[v]$ with half of the previous length. An example of the pairs $(\alpha[v], \beta[v])$ generated by this strategy is illustrated in Figure 3.8.

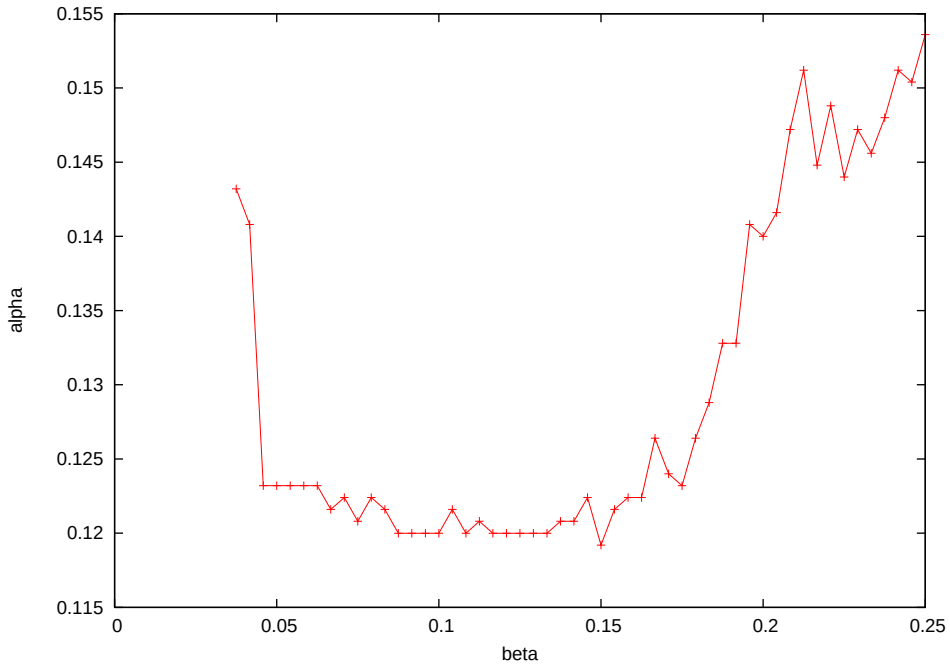


Figure 3.7: A typical curve of $\alpha[v]$ plotted against $\beta[v]$.

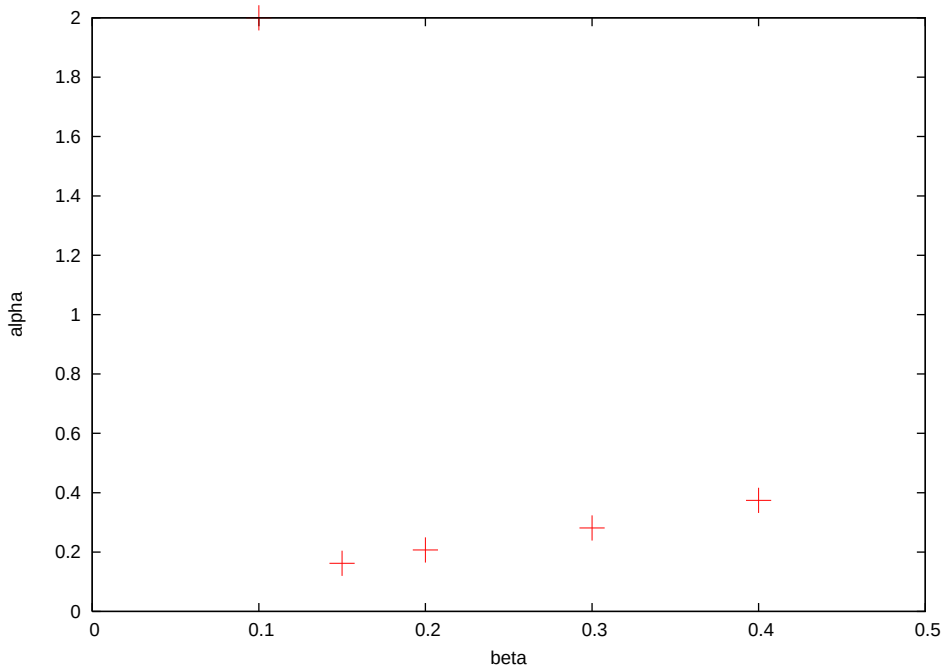


Figure 3.8: Searching for solution with $\alpha = 0.2, \beta = 0.4$. For $\beta[v] = 0.1$, ART3+ failed to converge. The search order for $\beta[v]$ is 0.4, 0.2, 0.1, 0.3, 0.15.

Another "trick" we use is that, for a given $\beta[v]$, once the algorithm have been running for a much longer time than a typical run, we save the current result and go to the next $\beta[v]$. We may go back to continue this saved run, if we do not get a good solution for the other $\beta[v]$ we tried.

Experimental Setting. In the experiments, we use two two-dimensional anthropomorphic phantoms provided by the Radiological Physics Center (RPC). One is the head and neck phantom (Figure 3.9), the other is the prostate phantom (Figure 3.10).

For our experiments, the square region into which the phantoms are embedded is subdivided into 101×101 voxels of size 16 mm^2, i.e., $J = 10,201$. We always use the same five beam directions: 0°, 72°, 144°, 216°, 288°. For each direction, there are 103 beamlets of width 4 mm, with the center of the center beamlet going through the center of the square region. Therefore the total number of beamlets is $N = 515$. The intensity of the radiation beamlets x_n $(1 \leq n \leq N)$ is nonnegative and has an upper bound 100, i.e., $c = 100$ in (3.13).

Experiment - Head and neck. In Figure 3.9, on the right the bright moon-shaped region is the primary PTV. The small disk under the primary PTV is the only OAR. The left disk is the secondary PTV. The rest of the voxels are considered to be in normal tissue that form the fourth subvolume. The l_d, u_d, α and β are presented in Table 3.4.

d	l_d	u_d	(α, β)
1	66	127.5	
2	54	127.5	
3=$\bar{s}$	0	20.0	(0.200, 0.400)
4	0	73.6	

Table 3.4: The prescription for the subvolumes in the head and neck phantom.

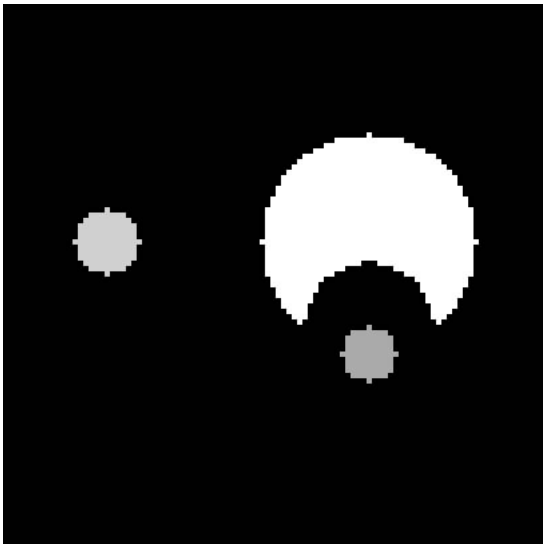


Figure 3.9: The head and neck phantom (d=1: bright moon-shaped region, a PTV; d=2: small disk on the left, a PTV; d=3: small disk on the right, an OAR; d=4: rest of the voxels within the body, normal tissue.)

Experiment - Prostate. In Figure 3.10, the bright moon-shaped region is an OAR (the bladder), under it is a PTV (the prostate), and under that is another OAR (the rectum). The two symmetric disks on the left and right are both OARs (the femoral heads). The l_d, u_d, α and β are presented in Table 3.5. There are eight tasks to be run.

The total computation times (the duration of the algorithm until it finds a solution) needed by the two algorithms for each of the experiments are shown in Table 3.6. In the LP time column, "3.452+194.032 (No solution)" means that the time LP runs without optimization is 3.452 seconds, but the solution it returns fails to satisfy (3.11) and so we run the LP again with optimization, and the time for this run is 194.032 seconds, but the solution it gives still fails to satisfy (3.11).

We can see from Table 3.6 that generally ART3+ works a little more efficiently than LP does in those cases where we do not need to run the LP again with the optimization goal. In the other cases, ART3+ is much more efficient. Also, there was a case for which ART3+ found a solution, but LP did not. However, more

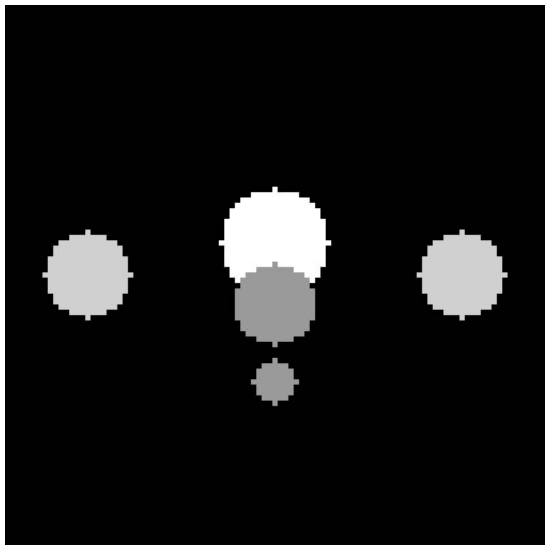


Figure 3.10: The prostate phantom (d=1: center disk, a PTV; d=2: bright moon-shaped region, an OAR; d=3: bottom disk, an OAR; d=4: side disks, two OARs; d=5: rest of the voxels within the body, normal tissue.)

d	l_d	u_d	(α, β)
1	60	127.5	
2=$\overline{d}$	0	49.0	(0.070, 0.347)
			(0.075, 0.276)
			(0.080, 0.225)
			(0.085, 0.164)
3	0	49.0	
4	0	49.0	
5	0	49.0	

d	l_d	u_d	(α, β)
1	60	127.5	
2	0	55.0	
3=$\overline{d}$	0	35.0	(0.200, 0.286)
			(0.250, 0.225)
			(0.300, 0.164)
			(0.350, 0.103)
4	0	49.0	
5	0	49.0	

Table 3.5: Prescriptions for the subvolumes in the prostate phantom. DVCs are applied to the bladder in the upper table and to the rectum to the lower table.

Algorithm	LP time (s)	ART3+ time (s)
Head and Neck	20.285+200.437	3.472
Prostate 1	3.452+194.032 (No solution)	9.324 (No solution)
Prostate 2	3.488+172.999 (No solution)	1.700
Prostate 3	3.360	2.776
Prostate 4	3.548	1.056
Prostate 5	2.296	6.532
Prostate 6	2.332	3.728
Prostate 7	2.372	2.336
Prostate 8	2.300	1.196

Table 3.6: Timings (in seconds) for finding solutions with the two algorithms.

experiments need to be done to arrive at a solid conclusion.

3.2.3 Experiments where ART3+O is More Efficient than MOSEK's Interior Point Method and Simplex Methods

The mathematical model of the IMPT planning for MCO database generation is a constrained optimization problem (G, f_G) as defined in Section 2.5. The linear feasibility problem G is constructed exactly the same as in Subsection 3.2.1, except here we allow the upper bound and lower bound to be infinite numbers, which is permitted by our definition of $\mathbb{G}_A$ in Subsection 2.2.1. The objective function f_G as defined in (2.15) can be the mean dose to a structure or a set of structures, the negative mean dose to a target or a set of targets (maximizing a function is the same as minimizing the negative of that function), the maximum dose to a structure or a set of structures, or the negative minimum dose to a target or a set of targets. We call a plan x feasible if it satisfies the constraints in G.

The r_{min} needed for the algorithm ART3+O of Section 2.5 can be found by considering the problem and the constraints at hand. For example, if f_G is the mean dose to an OAR, we choose $r_{min} = -0.01$. The upper bound of the number of calls to ART3+ is $2\lceil \log_2(r_{max} - r_{min})/\varepsilon \rceil$, which is a very modest number since the optimality tolerance ε required for radiation therapy planning is not very small. For

each call of ART3+ we use $K = 2 \times 10^7$.

For multi-criteria radiation therapy planning [26, 51], we first specify T objective functions f_G and, for each, we find an optimal plan x of the problem (G, f_G). We generate a database (to be used in the navigation step [51]) that contains these plans and some additional ones. The additional plans are determined by a variant, which we specify in the next paragraph, of the bounded objective function method [48]. Since each additional plan is obtained by solving a constrained optimization problem in the same form as (G, f_G), we again make use of ART3+O.

Let $\bar{x}$ denote the average of the T optimal plans. This averaged plan is used as the source of T additional constraints of the form $f_G(x) \leq f_G(\bar{x})$ (one for each of the original objective functions) for all the subsequent optimization tasks. This extended set of constraints is feasible, since $\bar{x}$ satisfies it. The additional plans in the database are obtained by solving, subject to the extended set of constraints, the following optimization problems:

1. minimize the sum of the mean doses of the structures that appeared in all of the original "minimize mean dose"-type objectives;

2. maximize the sum of the mean doses of the targets that appeared in all of the original "maximize mean dose"-type objectives;

3. repeat all of the original "minimize maximum dose" optimizations and "maximize minimum dose" optimizations.

The resulting database contains more than T but no more than $2T$ plans.

We use a pancreas case to demonstrate the advantages of ART3+O over the three general purpose LP algorithms implemented in MOSEK for the problem in the penultimate paragraph of Section 2.5. The patient volume (302,491 voxels) consists of liver, stomach, left kidney, right kidney, the PTV and skin (all remaining voxels). We select Rx (the prescription dose) to be 59.4 Gy (all Gy values reported

include a relative biological effectiveness, RBE, factor of 1.1), the maximum dose to all structures to be 1.12Rx and the minimum dose for the PTV to be 0.95Rx. We use three proton beams with 13,734 beamlets in total. The dose-influence matrix A whose jth row is a_j for $1 \leq j \leq J$ (see Subsection 3.2.1) has 62,226,127 nonzero elements. We perform the following optimizations for the numerical studies: [Task0] minimize mean skin dose; [Task1] maximize minimum PTV dose; [Task2] minimize mean liver dose; [Task3] minimize mean stomach dose; [Task4] minimize mean left kidney dose; [Task5] minimize mean right kidney dose; [Task6] set the minimum PTV dose to Rx and minimize the sum of the mean doses of the liver, stomach, left kidney and right kidney; and [Task7] set the minimum PTV dose to Rx and minimize the overall maximum dose. We use $\varepsilon = 0.1$ Gy for the optimality tolerance, which is well within radiation delivery precision. (For Tasks 6 and 7, we have tightened the constraint set by setting the PTV minimum dose to Rx. This was done for historical reasons and it does not matter for the comparison of the performance of algorithms reported in this subsection. For the clinical demonstrations in Section 1.5, we used the procedure exactly as described in Section 2.5.) We use [NoTask] to refer to the initial feasibility run.

All the experiments run on an Intel Xeon 2.66 GHz processor, 16 Gbyte memory, 64 bit workstation. To demonstrate the search process of ART3+O, we give as

t	$[r_{min}, r_{max}]$	r^t	Timing (s)	Find a solution in K iterations
			1.36	yes
0	[00.000, 43.809]	21.904	31.35	no
1	[21.904, 43.809]	32.857	4.08	yes
2	[21.904, 24.972]	23.438	11.55	yes
3	[21.904, 23.437]	22.671	31.55	no
4	[22.671, 23.437]	23.054	14.04	yes
5	[22.671, 23.051]	22.861	31.08	no
6	[22.861, 23.051]	22.956	2.72	yes
	[22.861, 22.927]			

Table 3.7: An example run of ART3+O with Task6 of the pancreas case.

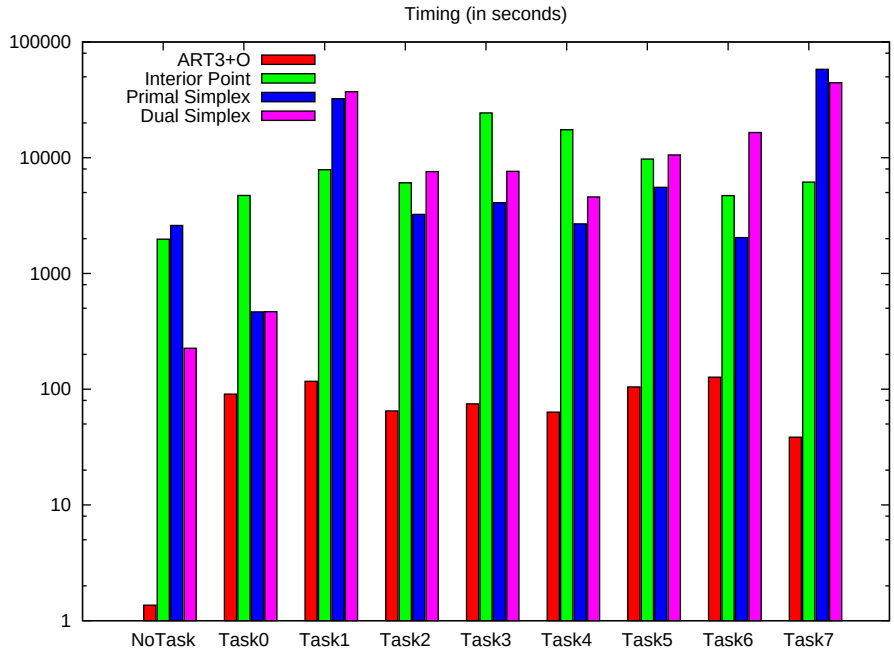


Figure 3.11: Timing of the four methods for the feasibility run and the eight optimization tasks. ART3+O is two to three orders of magnitude faster given the vertical time axes is with logarithmic scale.

an example the run of Task6; the results are shown in Table 3.7.

The timing results of ART3+O and the three standard LP algorithms, for the feasibility run and all eight tasks, are shown in Figure 3.11. ART3+O is far faster than the three algorithms in MOSEK. Typically, for each task, ART3+O uses about one to two minutes and MOSEK uses one to several hours. These results are not biased by the fact that ART3+O finds an ε-optimal value, while the algorithms in MOSEK find a better approximation to the true optimum for the following reasons: first, it is hard for MOSEK to stop earlier at an ε-optimal value and, at the same time, an exactly feasible solution like ART3+O does; second, even if we only consider the optimality, ART3+O still arrives at an ε-optimal value much faster than the algorithms in MOSEK (e.g., for Task0, ART3+O gets to 6.25 Gy in 90.75 s, the interior point method of MOSEK gets to 6.76 Gy in 2,099.99 s and it gets to 6.17 Gy, which it returns as the optimum, in 4,718.72 s). The total time to generate the

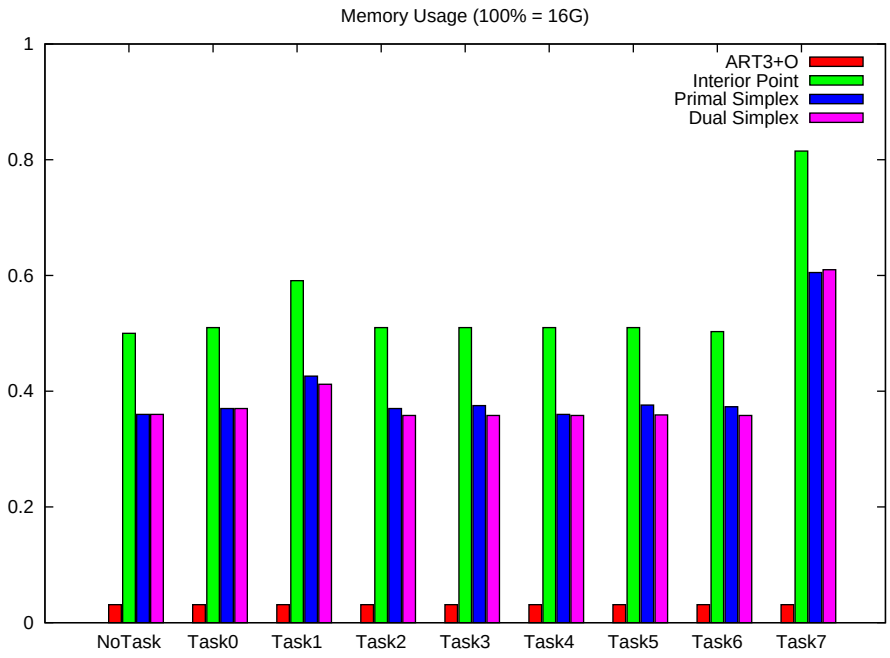


Figure 3.12: Memory usage of the four algorithms for the feasibility run and the eight optimization tasks. ART3+O uses less than a tenth of the memory needed by other algorithms.

plan database using ART3+O is on the order of 10 minutes.

Memory usage is an important consideration for radiation therapy planning algorithms because the advanced models that take into consideration uncertainty and organ motion require much larger optimization instances. The ART3+O algorithm has almost no memory overhead: its memory usage is just slightly over the memory required to hold the dose-influence matrix A and the (much smaller) dose bounds l and u whose jth components are l_j and u_j for $1 \leq j \leq J$, respectively. The general purpose algorithms come with a large memory overhead, as shown in Figure 3.12: ART3+O uses less than a tenth of the memory needed by other algorithms.

Given the fact that the treatment planned dose will not be delivered precisely (most institutions assume at least a 2% error in planned versus delivered dose on a per voxel basis), there is no need to run the optimizations to a high degree of optimality. Furthermore, running algorithms to suboptimality is typically much faster

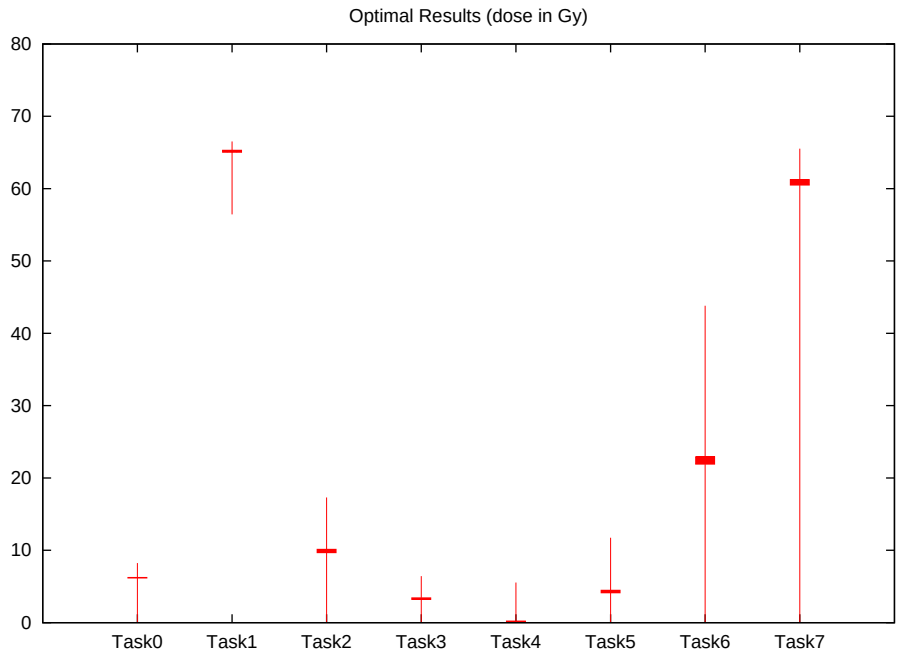


Figure 3.13: The vertical line segments are the initial ranges of r. The thickness of the horizontal segments is the gap between the optimal r found by ART3+O and the optimal r given by MOSEK.

than running to exact optimality. Using the results from the MOSEK algorithms as the "true" optimum, Figure 3.13 displays the ART3+O results in comparison. The results are well within the delivery precision for protons, and could be relaxed further for additional computation speed improvements.

To show how much faster ART3+ is than ART3, we ran the last search of the feasibility task and all eight tasks using both ART3 and ART3+. Figure 3.14 displays the timing of both algorithms. ART3+ is up to eight times faster than ART3 in our experiments. This is much greater than what is reported in Table 3.2 due to the larger size of our current problems.

The implementation of ART3+O is simple, fast and memory efficient. The underlying optimizations produce feasible plans (i.e., the resulting dose distributions satisfy all the specified constraints exactly) for which the objective function is within a distance from that of the truly optimal feasible plan that is controlled by the

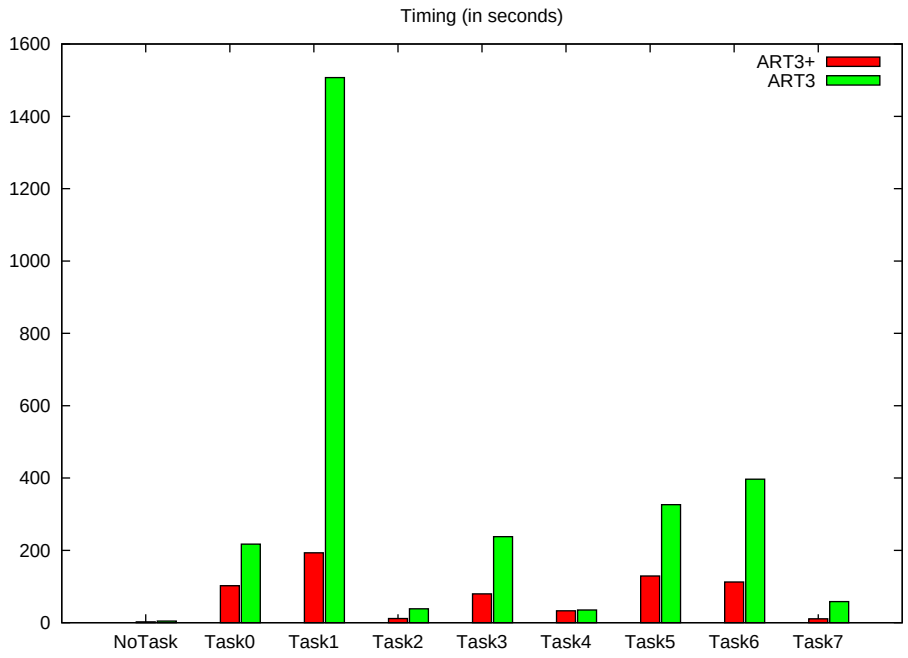


Figure 3.14: The timing of ART3 and ART3+ in the last search of the feasibility task and all eight other tasks.

optimality tolerance, for which we use $\varepsilon = 0.1$ Gy. Since all of our objectives and constraints are in units of Gy, this value of ε is directly interpretable, and 0.1 Gy is much smaller than the precision of radiation delivery. Additional speed gains for database generation are achievable by performing the optimization tasks in parallel or implementing ART3+O on a graphic processing unit (GPU).

ART3+ is a projection algorithm and such algorithms converge rapidly when the matrix used in the constraints, the dose-influence matrix A, is sparse. Sparsity of our A matrix implies that most of the time the normal vectors associated with a pair of constraints are mutually nearly orthogonal. Consecutive projections onto the two pairs of constraints with orthogonal normal vectors will produce a point that satisfies both. Therefore, by sequentially projecting onto the mostly mutually orthogonal constraints, one quickly finds a solution that satisfies all of the constraints. In the IMPT setting, the A matrix has on the order of 10% of its entries nonzero, and thus is fairly sparse. This is similar to IMRT, although the dose falls off more

slowly for photons and thus, depending on how you truncate small values, the IMRT matrices will generally be more full. Still, most of the entries will be close to zero, so regarding orthogonality of the rows, the IMRT matrix is still effectively sparse, and therefore ART3+O is also useful for IMRT beamlet optimizations.

Unlike most other projection algorithms, by using overrelaxation with different relaxation parameters depending on the distance of the current iterate to the nearest bounding hyperplane, ART3 converges finitely to a point in the feasible set, given that the feasible set is full-dimensional. Even though finite convergence does not imply rapid convergence, ART3 in practice stops at a feasible point in a short time, especially when the feasible set is large. Using ART3+ changes the underlying control from cyclic to repetitive and this saves time by not checking the easily-satisfied constraints frequently, but focusing on the constraints that are hard to satisfy. An extra benefit from doing this from the computer programming point of view is that the rapidly-accessible but limited-size CPU caches can be utilized at the late stage of the inner loop of ART3+, which further speeds up the computation.

In the implementation of ART3 or ART3+, an important parameter is the maximum number of iterations beyond which we judge the problem to be infeasible. Recall that the condition for both ART3 and ART3+ to be finitely convergent is that the feasible set is full-dimensional. However, unfortunately, we cannot determine before running ART3 or ART3+ whether or not the problem has a full-dimensional feasible set; this by itself is a problem as hard as finding a feasible point. In practice, the maximum number of iterations cannot be too small, because then ART3+ may stop too early for a problem with full-dimensional feasible set and return to the user that the problem is infeasible. On the other hand, the number of iterations cannot be too large, because for an infeasible problem instance (like some of the feasibility problem instances we solve by ART3+O), it will then take too long for ART3+ to return to the user that the problem is infeasible, thus making ART3+O inefficient.

Numerical experiments show that the times (roughly proportional to the number of iterations) ART3+ needs to run for feasible problems increases very mildly as the size of the interior of the feasible region decreases at the beginning, but increases dramatically when the size of the interior of the feasible region approaches to zero, which is the moment when we find the ε-optimal and stop the search, see Figure 3.15. This observation means that by the use of a relatively small maximum number of iterations an ε-optimal solution can be obtained rapidly by ART3+O. Nevertheless, because the optimality tolerance depends on the requirements of the application, a good choice of the maximum number of iterations is also application dependent and thus needs to be tuned. In practice, to determine what number K to use, we experiment on a typical problem with typical size, run various tasks with a quite large K and find a number that is larger than all the actual numbers of iterations needed for ART3+ to find a feasible solution to a feasible problem. However, it is impossible to find a K such that ART3+ is guaranteed to find a solution for a feasible problem in K iterations or ART3+ is guaranteed to find a solution that is ε-optimal.

A useful "trick" in the ART3+O algorithm of Section 2.5 is the warm-start: during the process of solving a series of augmented feasibility problems, it uses the solution returned by the previous ART3+ run as the initial point of the following ART3+ run, even when the solution in the previous run is not a feasible one. Experiments show that this "trick" reduces the overall search time significantly. The intuition behind it is that every projection iteration contributes to approaching the feasible set and the previous projection iterations should not be wasted.

The reasons given above to explain why ART3+ and ART3+O are efficient in our applications also provide us with scenarios where they may not be efficient. For some real-world applications the system matrix A is not sparse, or it is sparse but the zeros are not distributed evenly. In such cases, it is not true that most of

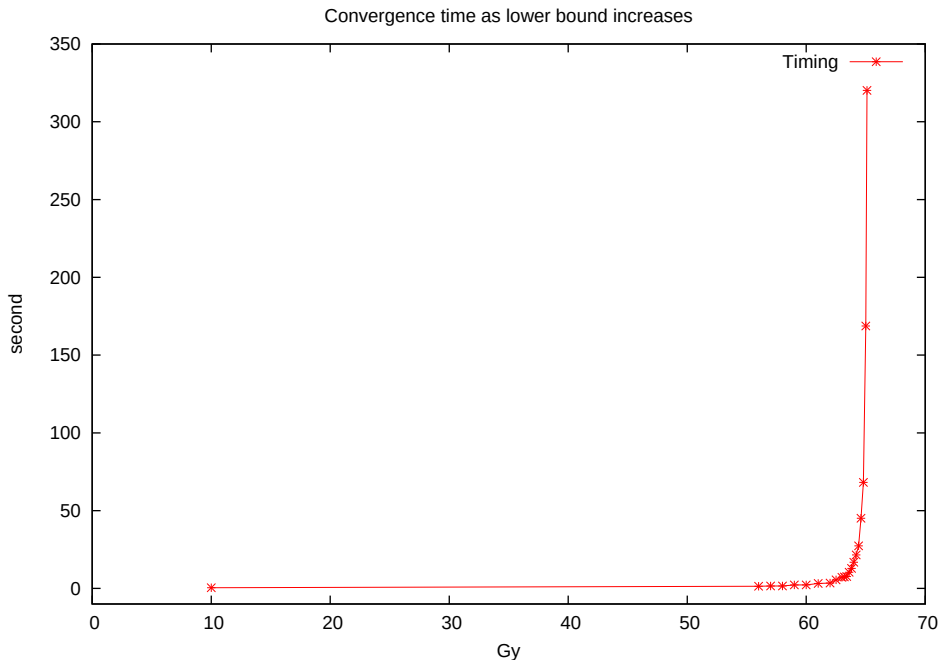


Figure 3.15: The times ART3+ needs to run for the feasibility problems in IMPT planning (without any objective function) when the lower bound of the PTV increases from 10 Gy to its maximum possible value.

the normal vectors of the constraints are mutually orthogonal. Therefore projection methods like ART3+ will possibly be slow, even for relatively small size problems. Also, as demonstrated in Table 3.3 and Figure 3.15, ART3+ will not be efficient for feasibility problems with a very small feasible set and ART3+O will not be efficient for optimization problems that require a very small optimality tolerance. ART3+ may not be efficient for solving interval inequalities such as $b-\varepsilon \leq Ax \leq b+\varepsilon$ with a small $\varepsilon > 0$, as may be specified when reconstructing from measurements b in CT (see Section 1.4).

An important consideration in the optimization of proton therapy plans is robustness. Small uncertainties in the location of the Bragg peaks of the proton beams can lead to large voxel dose uncertainties. Optimizers that do not account for this uncertainty might produce plans that are highly sensitive to errors in the proton beam range. Similarly, changes in patient geometry that occur during treatment,

especially in areas that are highly heterogeneous, such as nasal cavities, can lead to highly variable dose distributions. Accounting for range uncertainties can be done by using multiple instances of the A matrix [62]. Since the memory requirement of ART3+O is essentially only what is needed to store A, ART3+O should prove to be an ideal platform for robust IMPT optimization.

In IMRT and IMPT planning, MCO has been proved to be more efficient than single-criteria optimization [26, 51], since MCO provides a space of feasible plans from which to choose, while single-criteria optimization requires a trial and error process. ART3+O enables large scale MCO due to its efficiency in both time and memory as compared to interior point methods and simplex methods, and also the solution it produces satisfies the constraints exactly and its distance to the true optimal is controlled by a preassigned optimal tolerance ε, as compared to the fast gradient methods, which have no guarantee on either feasibility or optimality.

Chapter 4

Conclusions

4.1 Contributions

Most sequential or iterative algorithms in optimization use simple cyclic control even if they were proven to converge or finitely converge under more general controls. The motivation of this dissertation starts with the intuitive observation that sequential algorithms converge faster if constraints that are harder to be satisfied are picked more frequently. Consequently, the control sequence used is no longer cyclic, but almost cyclic or repetitive. The fundamental achievement of this dissertation is the introduction of two automatic procedures for converting a finitely convergent sequential algorithm into another sequential algorithm that (1) retains the property of finite convergence and (2) is likely to be more efficient. One of these procedures applies to sequential algorithms that converge finitely under repetitive control and the other applies to sequential algorithms that converge finitely under almost cyclic control. As examples, CART3^{+} and CART3$^{++}(i_0)$ were obtained from ART3 for solving linear inequalities and CMCSP$^{++}(i_0)$ was obtained from MCSP for solving convex feasibility problems. The superior performance of the new algorithm ART3+ has been shown by experiments in the application fields

of image reconstruction and IMRT planning.

From the performed experiments in the application of converting pixel images to blob images we conclude that both CART3^{+} and CART3$^{++}(i_0)$ are statistically significantly faster than CART3 (which is equivalent to ART3), with a speedup ratio better than 1.4. In the application of IMRT planning, we have treated the IMRT planning problem as a feasibility problem of finding a solution of a linear interval inequality system. The experiments have shown that ART3+ performs faster but less than twice as fast than ART3. However, as we decrease the upper bound for an OAR, the speedup ratio increases from 1.69 to 3.17. Even larger speedup ratios have been achieved in the experiments for IMPT planning. From the performed experiments in solving dose-volume constraints in IMRT planning we can see that generally ART3+ works a little bit faster than LP does when we do not need to run the LP with the optimization goal, but otherwise ART3+ is much faster. Also, there was even a case for which ART3+ found a solution, but LP did not.

Furthermore, by repeatedly applying ART3+ for solving linear feasibility problems, linearly constrained optimization problems have been solved within an optimality tolerance by the newly introduced algorithm ART3+O. This method first turns the objective function into a constraint bounded by an unknown variable representing the pursued optimal value and then search for the optimal value by solving a series of linear feasibility problems. The results of one feasibility task and eight optimization tasks for a pancreas case show that ART3+O is two to three orders of magnitude faster than the state of art industry standard commercial solvers and is much more memory efficient (assuming that only modest optimality tolerance is required, which is the case for the IMPT planning application).

These results support our claim that our proposed approach is appropriate and useful for automated speeding up of finitely convergent sequential algorithms.

4.2 Future Works

The research work in this dissertation can be extended in a number of aspects.

4.2.1 Finitely Convergent Sequential Algorithms with Perturbations

The motivation for studying the convergence behavior of modified finitely convergent sequential algorithms by adding summable perturbations to some or all the iterates lies in two aspects. Firstly, in some models that involve physical measurements, the existence of noise in the model is equivalent to a perturbation in one iteration using that measurement. If we can prove that a finitely convergent sequential algorithm is still finitely convergent when we add to it some kind of perturbations, we are confident to say that this algorithm is stable or resistant to such noise. Secondly, some recent works [11, 27, 38] show that a proper implementation of the perturbed algorithms steers the solution sequence towards a solution that is superior according to some criterion to the solution produced by the original algorithm. If we succeed in proving that adding perturbation conserves finite convergence, then we can implement the algorithm to steer to a superior solution while converging to a feasible solution in a finite number of steps.

We are going to first investigate on a perturbed version of ART3 (we name it PART3), by introducing two sequences $\{\beta_k\}_{k\in\mathbb{N}_\geq}$ and $\{v_k\}_{k\in\mathbb{N}_\geq}$ such that $\sum_{k=0}^{+\infty}\beta_k \leq +\infty$ and $\{v_k\}_{k\in\mathbb{N}_\geq}$ is a bounded sequence in $\mathbb{R}^{N_G}$, and adding in $Z_4\left(S^k,k,x^k,g^k\right)$ a perturbation term $\beta_k v_k$. We will try to prove the following conjecture:

Conjecture 4.2.1 *The PART3 algorithm is finitely convergent for $\mathbb{G}_A$ under repetitive control.*

If we succeed, then we will try to generalize the conjecture to our sequential algorithm scheme.

4.2.2 CMCSP++ is Faster than CMCSP in IMRT Applications

We have proved in Corollary 2.4.7 in Section 2.4 that CMCSP++(i_0) is finitely convergent for $\mathbb{G}_B$ which is the convex feasibility problem given that the feasible set satisfies the Slater condition. The IMRT planning problem has been formulated in Section 3.2 as a linear feasibility problem, which is a special case of convex feasibility problems. However, in practice, some constraints used in IMRT planning are not linear but convex, e.g., the equivalent uniform dose (EUD) constraints [53]. Mathematically EUD has been defined as the "generalized mean" dose of a specific organ:

$$EUD_t(d) = (\frac{1}{|O_t|}\sum_{j\in O_t} {d_j}^{\omega_t})^{1/\omega_t},$$

where O_t is the set of voxel indices j inside a specific organ indexed by t, $d_j = \langle a_j, x\rangle$, and ω_t is a structure depending parameter. The usual requirement for an OAR is:

$$EUD_t(d) \leq EUD_t^{max}.$$

We are going to solve convex feasibility problems arose from IMRT planning with CMCSP and CMCSP++ and compare the speed of the two algorithms.

MCSP++ can also be wrapped into a convex optimization algorithm MCSP++O, exactly as how we design ART3+O. The performance of MCSP++O for solving convex optimization problems relative to the performance of other existing convex optimization solver will be reported by experiments.

4.2.3 Non-finitely Convergent Sequential Algorithms

Since most iterative optimization algorithms are convergent but not finitely convergent, it is of interest to ask what we can do with the same speedup idea applied to

convergent sequential algorithms. Is there a similar automatic transformation from one convergent sequential algorithm to another? Is the new sequential algorithm still convergent? Does the new sequential algorithm converge faster in numerical experiments? Applications for such algorithms exist when exact constraint satisfaction is not required.

4.2.4 A Novel Linear Programming Solver

We realize that the constrained optimization problem (G, f_G) specified by (2.15) with $J = 1$ has a linear objective function and linear inequality constraints, and so it is a standard LP problem. Considering that LP is the most fundamental problem in optimization and a widely adopted model for practical problems, it is worth investigating whether our projection-based optimization algorithm can be applied efficiently to other real-world LP problems. Some qualitative or quantitative measures need to be developed to determine algorithm efficiency based on specific properties of problem instances. For example, one such measure is sparsity defined as the fraction of nonzero entries in the system matrix. One could also measure the extent to which most of the normal vectors in the system matrix are mutually orthogonal and the likelihood of the feasible region being full-dimensional (the solution set of an LP problem with equality constraints cannot be full-dimensional), etc. Such pre-processing can be integrated into a general purpose LP solver package, which will then automatically choose the most suitable algorithm.

4.2.5 Miscellaneous

Some popular optimization techniques, such as the interior point methods for solving constrained optimization problems, use a so-called "phase I" method to find a feasible point as the starting point for the optimality search afterward. The fast finitely convergent sequential algorithms ART3+ and MCSP++ are promising can-

didates for being such a phase I method.

Finitely convergent sequential algorithms, as we defined them, are not inherently parallelizable. However, lower level parallelization (such as parallel inner product evaluation) is possible for algorithm implementation. Inherently parallel iterative algorithms such as block ART3 exist and we plan to investigate their adaptation to our framework.

Bibliography

[1] S. Agmon. The relaxation method for linear inequalities. *Canadian Journal of Mathematics*, 6:382–392, 1954.

[2] E.D. Andersen and K.D. Andersen. The MOSEK interior point optimizer for linear programming: an implementation of the homogeneous algorithm. In H. Frenk, K. Roos, T. Terlaky, and S. Zhang, editors, *High Performance Optimization*, pages 197–232, Kluwer Academic Publishers, 2000.

[3] H.H. Bauschke. The approximation of fixed points of compositions of nonexpansive mappings in Hilbert space. *Journal of Mathematical Analysis and Applications*, 202:150–159, 1996.

[4] H.H. Bauschke and J.M. Borwein. On projection algorithms for solving convex feasibility problems. *SIAM Review*, 38:367–426, 1996.

[5] H.H. Bauschke and P.L. Combettes. A weak-to-strong convergence principle for Fejér-monotone methods in hilbert spaces. *Mathematics of Operations Research*, 26:248–264, 2001.

[6] H.H. Bauschke and P.L. Combettes. Iterating Bregman retractions. *SIAM Journal on Optimization*, 13:1159–1173, 2003.

[7] H.H. Bauschke, P.L. Combettes, and S.G. Kruk. Extrapolation algorithm for affine-convex feasibility problems. *Numerical Algorithms*, 41:239–274, 2006.

[8] D. Blatt and A.O. Hero, III. Energy based sensor network source localization via projection onto convex sets (POCS). *IEEE Transactions on Signal Processing*, 54:3614–3619, 2006.

[9] T. Bortfeld, J. Stein, and K. Preiser. Clinically relevant intensity modulation optimization using physical criteria. In D.D. Leavitt and G. Starkschall, editors, *The XIIth International Conference on the Use of Computers in Radiation Therapy (ICCR)*, pages 1–4, Salt Lake City, Utah, USA, May 27-30 1997.

[10] D. Butnariu, Y. Censor, and S. Reich, editors. *Inherently Parallel Algorithms in Feasibility and Optimization and Their Applications*. Elsevier, 2001.

[11] D. Butnariu, R. Davidi, G.T. Herman, and I.G. Kazantsev. Stable convergence behavior under summable perturbations of a class of projection methods for

convex feasibility and optimization problems. *IEEE Journal of Selected Topics in Signal Processing*, 1:540–547, 2007.

[12] Y. Censor, M.D. Altschuler, and W.D. Powlis. On the use of Cimmino's simultaneous projections method for computing a solution of the inverse problem in radiation therapy treatment planning. *Inverse Problems*, 4:607–623, 1988.

[13] Y. Censor, A. Ben-Israel, Y. Xiao, and J.M. Galvin. On linear infeasibility arising in intensity-modulated radiation therapy inverse planning. *Linear Algebra and Its Applications*, 428:1406–1420, 2008.

[14] Y. Censor and S.A. Zenios. *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, New York, 1997.

[15] A.E. Cetin, H. Ozaktas, and H.M. Ozaktas. Resolution enhancement of low resolution wavefields with POCS algorithm. *Electronics Letters*, 39:1808–1810, 2003.

[16] W. Chen and G.T. Herman. Efficient controls for finitely convergent sequential algorithms. *ACM Transactions on Mathematical Software*, 37:Article 14, 2010.

[17] W. Chen, G.T. Herman, and Y. Censor. Algorithms for satisfying dose-volume constraints in intensity-modulated radiation therapy. In Y. Censor, M. Jiang, and A.K. Louis, editors, *Mathematical Methods in Biomedical Imaging and Intensity-Modulated Radiation Therapy (IMRT)*, pages 97–106, Edizioni della Normale, Pisa, Italy, 2008.

[18] P.S. Cho, S. Lee, R.J. Marks, S. Oh, S.G. Sutlief, and M.H. Phillips. Optimization of intensity modulated beams with volume constraints using two methods: cost function minimization and projection onto convex sets. *Medical Physics*, 25:435–443, 1998.

[19] H. Choi and R.G. Baraniuk. Multiple wavelet basis image denoising using Besov ball projections. *IEEE Signal Processing Letters*, 11:717–720, 2004.

[20] G. Cimmino. Calcolo approssimato per le soluzioni dei sistemi di equazioni lineari. *La Ricerca Scientifica (Roma)*, 1:326–333, 1938.

[21] P.L. Combettes. The foundations of set theoretic estimation. *Proceedings of the IEEE*, 81:182–208, 1993.

[22] P.L. Combettes. The convex feasibility problem in image recovery. *Advances in Imaging and Electron Physics*, 95:155–270, 1996.

[23] P.L. Combettes. Convex set theoretic image recovery by extrapolated iterations of parallel subgradient projections. *IEEE Transactions on Image Processing*, 6:493–506, 1997.

[24] P.L. Combettes. Hilbertian convex feasibility problem: Convergence of projection methods. *Applied Mathematics and Optimization*, 35:311–330, 1997.

[25] P.L. Combettes and S.A. Hirstoaga. Equilibrium programming in Hilbert spaces. *Journal of Nonlinear and Convex Analysis*, 6:117–136, 2005.

[26] D. Craft, T. Halabi, H. Shih, and T. Bortfeld. Approximating convex Pareto surfaces in multi-objective radiotherapy planning. *Medical Physics*, 33:3399–3407, 2006.

[27] R. Davidi, G.T. Herman, and Y. Censor. Perturbation-resilient block-iterative projection methods with applications to image reconstruction from projections. *International Transactions in Operational Research*, 16:505–524, 2009.

[28] R. Davidi, G.T. Herman, and J. Klukowska. *SNARK09: A Programming System for the Reconstruction of 2D Images from 1D Projections*. The CUNY Institute for Software Design and Development, New York, 2009.

[29] A.R. De Pierro and A.N. Iusem. A finitely convergent row-action method for the convex feasibility problem. *Applied Mathematics and Optimization*, 17:225–235, 1988.

[30] F. Deutsch. *Best Approximation in Inner Product Spaces*. Springer-Verlag, 2001.

[31] J. Eckstein and B.F. Svaiter. General projective splitting methods for sums of maximal monotone operators. *SIAM Journal on Control and Optimization*, 48:787–811, 2009.

[32] D. Gordon and R. Gordon. CGMN revisited: Robust and efficient solution of stiff linear systems derived from elliptic partial differential equations. *ACM Transactions on Mathematical Software*, 35:1–27, 2008.

[33] N.I.M. Gould. How good are projection methods for convex feasibility problems? *Computational Optimization and Applications*, 40:1–12, 2008.

[34] J. Gu, H. Stark, and Y. Yang. Wide-band smart antenna design using vector space projection methods. *IEEE Transactions on Antennas and Propagation*, 52:3228–3236, 2004.

[35] G.T. Herman. A relaxation method for reconstructing objects from noisy X-rays. *Mathematical Programming*, 8:1–19, 1975.

[36] G.T. Herman. *Fundamentals of Computerized Tomography: Image Reconstruction from Projections. 2nd edition*. Springer, 2009.

[37] G.T. Herman and W. Chen. A fast algorithm for solving a linear feasibility problem with application to intensity-modulated radiation therapy. *Linear Algebra and Its Applications*, 428:1207–1217, 2008.

[38] G.T. Herman and R. Davidi. On image reconstruction from a small number of projections. *Inverse Problems*, 24:045011, 2008.

[39] G.T. Herman and L.B. Meyer. Algebraic reconstruction techniques can be made computationally efficient. *IEEE Transactions on Medical Imaging*, 12:600–609, 1993.

[40] S. Kaczmarz. Approximate solution of systems of linear equations (originally published as: Angenäherte Auflösung von Systemen linearer Gleichungen, Bulletin International de l'Academie Polonaise des Sciences, Letter A, 1937, 355-357). *International Journal of Control*, 57:1269–1271, 1993.

[41] I.G. Kazantsev, S. Schmidt, and H.F. Poulsen. A discrete spherical X-ray transform of orientation distribution functions using bounding cubes. *Inverse Problems*, 25:105009, 2009.

[42] K.C. Kiwiel and B. Łopuch. Surrogate projection methods for finding fixed points of firmly nonexpansive mappings. *SIAM Journal on Optimization*, 7:1084–1102, 1997.

[43] S.-H. Lee and K.-R. Kwon. Mesh watermarking based on projection onto two convex sets. *Multimedia Systems*, 13:323–330, 2008.

[44] R.M. Lewitt. Multidimensional digital image representation using generalized Kaiser-Bessel window functions. *Journal of the Optical Society of America A*, 7:1834–1846, 1990.

[45] A.W.-C. Liew, H. Yan, and N.-F. Law. POCS-based blocking artifacts suppression using a smoothness constraint set with explicit region modeling. *IEEE Transactions on Circuits and Systems for Video Technology*, 15:795–800, 2005.

[46] R. Lougee-Heimer. The common optimization interface for operations research. *IBM Journal of Research and Development*, 47:57–66, 2003.

[47] Y.M. Lu, M. Karzand, and M. Vetterli. Demosaicking by alternating projections: Theory and fast one-step implementation. *IEEE Transactions on Image Processing*, to appear.

[48] R.T. Marler and J.S. Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26:369–395, 2004.

[49] Y.I. Merzlyakov. On a relaxation method of solving systems of linear inequalities. *U.S.S.R. Computational Mathematics and Mathematical Physics*, 2:504–510, 1963.

[50] A. Messac, A. Ismail-Yahaya, and C.A. Mattson. The normalized normal constraint method for generating the Pareto frontier. *Structural and Multidisciplinary Optimization*, 25:86–98, 2003.

[51] M. Monz, K-H. Küfer, T. Bortfeld, and C. Thieke. Pareto navigation-algorithmic foundation of interactive multi-criteria IMRT planning. *Physics in Medicine and Biology*, 53:985–998, 2008.

[52] T.S. Motzkin and I.J. Schoenberg. The relaxation method for linear inequalities. *Canadian Journal of Mathematics*, 6:393–404, 1954.

[53] A. Niemierko. Reporting and analysing dose distributions: a concept of equivalent uniform dose. *Medical Physics*, 24:103–110, 1997.

[54] J.R. Palta and T.R. Mackie. *Intensity-Modulated Radiation Therapy: The State of the Art*. Medical Physics Publishing, Madison, Wisconsin, 2003.

[55] P.M. Pardalos and M.G.C. Resende, editors. *Handbook of Applied Optimization*. Oxford University Press, 2002.

[56] G. Pierra. Decomposition through formalization in a product space. *Mathematical Programming*, 28:96–115, 1984.

[57] A.A. Samsonov, E.G. Kholmovski, D.L. Parker, and C.R. Johnson. POCSENSE: POCS-based reconstruction for sensitivity encoded magnetic resonance imaging. *Magnetic Resonance in Medicine*, 52:1397–1406, 2004.

[58] N.T. Shaked and J. Rosen. Multiple-viewpoint projection holograms synthesized by spatially incoherent correlation with broadband functions. *Journal of the Optical Society of America A*, 25:2129–2138, 2008.

[59] G. Sharma. Set theoretic estimation for problems in subtractive color. *Color Research & Application*, 25:333–348, 2000.

[60] H. Stark and Y. Yang. *Vector Space Projections: A Numerical Approach to Signal and Image Processing, Neural Nets, and Optics*. Wiley-Interscience, 1998.

[61] C. Thieke, K-H. Küfer, M. Monz, A. Scherrer, F. Alonso, S. Nill, C. Thilmann, and T. Bortfeld. Beyond weight factors: new concepts for defining and analysing dose optimisation. *Radiotherapy and Oncology*, 73:S75, 2004.

[62] J. Unkelbach, T.C.Y. Chan, and T. Bortfeld. Accounting for range uncertainties in the optimization of intensity modulated proton therapy. *Physics in Medicine and Biology*, 52:2755–2773, 2007.

[63] B.J. van Wyk and M.A. van Wyk. A POCS-based graph matching algorithm. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 26:1526–1530, 2004.

[64] L. Xing, R.J. Hamilton, D. Spelbring, C.A. Pelizzari, G.T.Y. Chen, and A.L. Boyer. Fast iterative algorithms for three-dimensional inverse treatment planning. *Medical Physics*, 25:1845–1849, 1998.

[65] L. Xing, J. Li, S. Donaldson, Q. Le, and A. Boyer. Optimization of importance factors in inverse planning. *Physics in Medicine and Biology*, 44:2525–2536, 1999.

[66] M. Yukawa and I. Yamada. Pairwise optimal weight realization–Acceleration technique for set-theoretic adaptive parallel subgradient projection algorithm. *IEEE Transactions on Signal Processing*, 54:4557–4571, 2006.

Index

Algebraic Reconstruction Techniques, 3

almost cyclic, 22

ART, 3

ART3, 3

ART3+, 3

ART3+O, 4

basis function, 47

blob, 47

choice function, 19

closed problem set, 18

CLP, 63

computerized tomography, 5

constrained optimization problem, 1, 43

control, 4

control sequence, 21

convex feasibility problem, 1, 28

convex function, 27

convex optimization problem, 1

CT, 5

cyclic control, 21

cyclic sequential algorithm, 32

database, 70

dose-influence matrix, 71

dose-volume constraint, 53

dose-volume histogram, 12

DVC, 53

DVH, 12

equivalence of two paths, 23

equivalence of two sequential algorithms, 23

equivalent uniform dose, 83

EUD, 83

feasibility problem, 1

finitely convergent sequence, 22

finitely convergent sequential algorithm, 3, 17

full-dimensional, 4

generalization of a sequential algorithm, 23

gradient, 28

IMPT, 12

IMRT, 11

intensity modulated proton therapy, 12

intensity modulated radiation therapy, 11
iteration cycle, 6
linear feasibility problem, 1, 25
linear programming, 13
LP, 13
MCO, 12
MCSP, 27
MLC, 11
Modified Cyclic Subgradient Projection Method, 27
MOSEK, 8
multi-criteria optimization, 12
multi-leaf collimator, 11
normalized mean absolute picture distance measure, 6
OAR, 11
objective function, 43
optimality tolerance, 44
optimization problem, 1
organ at risk, 11
Pareto surface, 12
path, 20, 21
phantom, 49
pixel, 47
planning target volume, 11
problem set, 17
projection method, 2
PTV, 11
repetitive control, 22
Slater condition, 27
SNARK09, 47
solution sequence, 21
sparsity, 84
state, 18
state sequence, 21
subgradient, 28
unconstrained optimization problem, 1
warm-start, 77
windowed image, 49