LINEAR PROGRAMMING FORMULATIONS AND ALGORITHMS FOR RADIOTHERAPY TREATMENT PLANNING

ARINBJÖRN ÓLAFSSON* AND STEPHEN J. WRIGHT†

Abstract. Optimization has become an important tool in treatment planning for cancer radiation therapy. It may be used to determine beam weights, beam directions, and appropriate use of beam modifiers such as wedges and blocks, with the aim of delivering a required dose to the tumor while sparing nearby critical structures and normal tissue. Linear programming formulations are a core computation in many approaches to treatment planning, because of the abundance of highly developed linear programming software. However, these formulations of treatment planning often require a surprisingly large amount of time to solve—more than might be anticipated given the dimensions of the problems. Moreover, the choices of formulation, algorithm, and pivot rule that perform best from a computational viewpoint are sometimes not obvious, and the software's default choices are sometimes poor. This paper considers several linear programming formulations of treatment planning problem and tests several variants of simplex and interior-point methods for solving them. Conclusions are drawn about the most effective formulations and variants.

Key words. Linear Programming, Simplex Method, Interior-Point Method, Radiation Therapy.

1. Introduction. Radiation therapy is a widely used technique for treating many types of cancer. It works by depositing radiation into the body of the patient, so that prescribed amounts of radiation are delivered to the cancerous regions (tumors), while nearby non-cancerous tissues are spared to the extent possible. Radiation interferes with the DNA of cells, impeding their ability to reproduce. It tends to affect fast-multiplying cells (such as those found in tumors) preferentially, making them more likely to be eliminated.

In this paper, we consider external-beam radiotherapy, in which the radiation is delivered via beams fired into the patient's body from an external source. The linear accelerator that produces the beams is located in a gantry which can be moved around the patient, allowing the beams to be delivered from a number of different angles. Additionally, a collimator can be placed in front of the beam to change its shape, and wedges can be used to vary the intensity of the beam across the field. In the "step-and-shoot" mode of treatment, the beam is aimed from a number of different angles (typically between 4 and 20), a wedge orientation and collimator shape is chosen for each angle, and the radiation beam is exposed for a certain amount of time (known as the *beam weight*). Two major variants of this approach include *conformal therapy*, in which the shape of the collimator at each angle is chosen to match the shape of the tumor as viewed from that angle, and *intensity-modulated radiation therapy (IMRT)* in which the beam field is divided for planning purposes into a rectangular array of "beamlets," which are then assigned individual weights.

For purposes of modeling and planning, that part of the patient's body to which radiation is applied is divided using a regular grid with orthogonal principal axes. The space is therefore partitioned into small rectangular volumes called *voxels*. The treatment planning process starts by calculating the amount of radiation deposited by a unit weight from each beam into each voxel. These doses are assembled into a *dose matrix*. (Each entry A_{ij} in this matrix is the dose delivered to voxel i by a unit weight of beam j.) Once the dose matrix is known, *inverse treatment planning*

*Industrial Engineering Department, 1513 University Avenue, University of Wisconsin, Madison, WI 53706, U.S.A.

†Computer Sciences Department, 1210 W. Dayton Street, University of Wisconsin, Madison, WI 53706, U.S.A.

is applied to find a plan that optimizes a specified treatment objective while meeting certain constraints. The treatment plan consists of a specification of the weights for all beams.

Linear programming is at the core of many approaches to treatment planning. It is a natural way to model the problem, because the amount of radiation deposited by a particular beam in each voxel of the treatment space is directly proportional to the beam weight, and because the restrictions placed on doses to different parts of the treatment space often take the form of bounds on the doses to the voxels.

Not all constraints can, however, be modeled directly in a linear programming formulation. In many treatment situations, the closeness of the tumor to some critical structures (vital organs or glands, or the spinal cord) makes it inevitable that some part of these structures will receive high doses of radiation. Rather than limit the total dose received by the structure, treatment planners sometimes choose to "sacrifice" a certain fraction of the structure and curtail the dose to the remainder of the structure. A constraint of this type is known as a *dose-volume (DV) constraint*; it typically requires that "no more than a fraction f of the voxels in a critical region $\mathcal{C}$ shall receive dose higher than δ." This type of constraint cannot be expressed as a linear function of the beam weights. It *can* be formulated in a binary integer program, in which a binary variable associated with each voxel indicates whether or not the dose to that voxel exceeds the prescribed threshold, but such problems are generally quite expensive to solve; see for example Lee, Fox, and Crocker [14] and Preciado-Walters et al. [18]. In Section 3.2, we discuss alternative techniques for imposing dose-volume constraints, using heuristics that require the solution of a sequence of linear programs.

In solving the linear programs associated with treatment planning problems, we have observed that the computational time required can vary widely according to a number of factors, including:

- the type of constraints imposed;
- whether the primal or dual formulation of the linear program is used;
- the use of primal simplex, dual simplex, or interior-point algorithms;
- the choice of pivot rule in the simplex algorithm;
- whether the code's aggregator or presolver is used to reduce the size of the formulated problem or, alternatively, the formulation is reduced "by hand," prior to calling the solver.

We report in this paper on a computational study of several popular linear programming formulations of the treatment planning problem, for data sets arising from both conformal radiotherapy and IMRT. We aim to give some insight into the performance of the solvers on these various formulations, and as to which types of constraints cause significant increases in the runtime. We also give some general recommendations as to the best algorithms, pivot rules, and reduction techniques for each formulation.

The paper is a case study in the use of linear programming software on an important class of large problems. As we see in subsequent sections, considerable experience and experimentation is often needed to identify a strategy (that is, use of primal or dual formulation, choice of pivot rule, manual elimination of variables and constraints, and so on) that yields the solution with the least amount of computational effort. Moreover, even the best software cannot be relied on to make good default choices in this matter.

The remainder of the paper is structured as follows. Section 2 contains a description of the software tools that were used in our experiments. The four types of linear programming models of treatment planning that we tested, and their relevance

to treatment planning, are described in Section 3. The data sets used in experiments are described in Section 4; they include both data that is typical of conformal therapy and data that arises in IMRT planning. We interpret and discuss the computational results in Section 5. Section 6 contains our main conclusions.

2. Algorithms and Software Tools. In this section, we describe the software tools we use in the experiments of this paper. These are the GAMS modeling system and CPLEX 8.1.

GAMS [3] is a high-level modeling language that allows optimization problems to be specified in a convenient and intuitive fashion. It contains procedural features that allow solution of the model for various parameter values and analysis of the results of the optimization. It is linked to a wide variety of optimization codes, thereby facilitating convenient and productive use of high-quality software.

CPLEX is a leading family of codes for solving linear, quadratic, and mixed-integer linear programming problems. In this study, we make use of the CPLEX Simplex and Barrier codes for linear programming, and the CPLEX Barrier code for quadratic programming. The CPLEX Simplex code implements both primal simplex and dual simplex; the user can choose between the two algorithms, or can leave the code to make the choice automatically. CPLEX performs aggregation and preprocessing to reduce the number of constraints and variables in the model presented to it, eliminating redundant constraints and removing those primal and dual variables whose values can be determined explicitly or expressed easily in terms of other variables. CPLEX Simplex allows the user to choose between a number of pricing strategies for selecting the variable to enter the basis. For the primal simplex method, pricing options include reduced-cost (in which the entering variable is chosen to be the one with the smallest reduced cost, see Dantzig [4, Ch.12]) and variants of the steepest-edge strategy described by Forrest and Goldfarb [6]. All strategies are used in conjunction with partial pricing, which means that just a subset of the dual slacks are evaluated at each iteration, to save on the linear algebra costs associated with pricing. For dual simplex, the pricing options include reduced-cost, the devex rule (see Harris [8]), a strategy that combines the latter two rules, and two variants of steepest edge. The user can opt to let the code determine an appropriate pricing strategy automatically.

GAMS allows user-defined options to be passed to CPLEX by means of a text file. By setting an option `predual` in this file, the user can force GAMS to pass the dual formulation of the given problem to CPLEX, rather than the (primal) formulation specified in the model file. This transformation to dual formulation is carried out within the GAMS system, before calling CPLEX. Note that the choice of formulation (primal or dual) can be made independently of the choice between primal and dual simplex method. Application of the primal simplex algorithm to the dual formulation is *not* equivalent to applying dual simplex to the primal formulation. The effects of preprocessing sometimes are different for the two formulations, different methods for finding a starting point may be used, and the iterates evolve differently even when the same pivot rule is used. Different "menus" of pivot rules are available for the primal and dual simplex options.

The CPLEX Barrier code for linear programming also allows the user to choose between different ordering rules for the sparse Cholesky factorizations that are performed at each iteration. These options include minimum degree, minimum local-fill, and nested dissection. We found little performance difference between these options in our tests, so we report results only for the default choice.

In Section 3.4, we use the CPLEX Barrier code for quadratic programming, which implements a similar primal-dual interior-point algorithm to the one for linear programming, but performs different sparse linear algebra computations at each iteration due to the presence of a Hessian term. To call this quadratic programming code from GAMS, we use a QP wrapper utility that writes out a text file and then invokes CPLEX (see [7]).

3. Formulations of the Treatment Planning Problem. In this section we will describe four formulations of the radiation treatment planning problem that define goals and constraints in different ways. The first three are linear programming models and the fourth is a quadratic program. For each linear formulation, we describe the main features and present the most natural formulation. We then describe alternative formulations obtained by eliminating variables and applying duality theory.

For purposes of inverse treatment planning, the voxels in the treatment volume typically are partitioned into three classes. *Target voxels*, denoted by an index set $\mathcal{T}$, are those that lie in the tumor and to which we usually want to apply a substantial dose. *Critical voxels*, denoted by $\mathcal{C}$, are those that are part of a sensitive structure (such as the spinal cord or a vital organ) that we particularly wish to avoid irradiating. *Normal voxels*, denoted by $\mathcal{N}$, are those that fall into neither category. Ideally, normal voxels should receive little or no radiation, but it is less important to avoid dose to these voxels than to the critical voxels.

All three of these classes appear explicitly only in model II. For the remaining models, the critical voxels are lumped with the normal voxels in the formulation. This does not mean, however, that the models make no distinction between nontarget voxels. In model I, for instance, we can impose a smaller upper bound on the dose for the critical voxels than for the normal voxels, by modifying the "critical" components of the vector $x^U_{\mathcal{N}}$ appropriately. We can also impose a larger penalty for dose to a critical voxel than to a normal voxel by adjusting the components of the cost vector $c_{\mathcal{N}}$.

3.1. Model I: A Formulation with Explicit Bounds on Voxel Doses. In the first formulation we consider, the treatment area is partitioned into a target region $\mathcal{T}$ consisting of $n_{\mathcal{T}}$ voxels and a normal region $\mathcal{N}$ consisting of $n_{\mathcal{N}}$ voxels. The dose delivered to $\mathcal{T}$ is constrained to lie between a lower bound vector $x^L_{\mathcal{T}}$ and an upper bound vector $x^U_{\mathcal{T}}$. Dosage delivered to $\mathcal{N}$ is bounded above by $x^U_{\mathcal{N}}$. We wish to minimize a weighted sum of doses delivered to the normal voxels, where the weights are components of a cost vector $c_{\mathcal{N}}$. Defining the variables to be w (the vector of beam weights), $x_{\mathcal{T}}$ (the vector of doses to the target voxels), and $x_{\mathcal{N}}$ (the vector of doses to the normal voxels), we can express the model as follows.

$$\min_{w,x_{\mathcal{T}},x_{\mathcal{N}}} c^T_{\mathcal{N}} x_{\mathcal{N}} \text{ s.t.} \tag{3.1a}$$

$$x_{\mathcal{T}} = A_{\mathcal{T}} w, \tag{3.1b}$$

$$x_{\mathcal{N}} = A_{\mathcal{N}} w, \tag{3.1c}$$

$$x_{\mathcal{N}} \leq x^U_{\mathcal{N}}, \tag{3.1d}$$

$$x^L_{\mathcal{T}} \leq x_{\mathcal{T}} \leq x^U_{\mathcal{T}}, \tag{3.1e}$$

$$w \geq 0. \tag{3.1f}$$

The submatrices $A_{\mathcal{T}} \in \mathbb{R}^{n_{\mathcal{T}} \times p}$ and $A_{\mathcal{N}} \in \mathbb{R}^{n_{\mathcal{N}} \times p}$ are dose matrices for the target and normal regions, respectively.

Bahr et al. in [1, Fig. 8] was apparently the first to propose formulation (3.1). Since then it has appeared in similar format in various papers, sometimes with additional constraints. Hodes in [9] used this model without the upper bounds $x_{\mathcal{T}}^U$ on the tumor. Morrill et al. [17] used the same constraints but a slightly different objective. The formulation appears unchanged in Rosen et al. [19, p. 143]. By omitting the upper bound $x_{\mathcal{N}}^U$ on normal tissue voxels we obtain the model described by Shepard et al. [21, p. 731]. Sonderman and Abrahamson [22, p. 720] added restrictions on the number of beams that may be used by adding binary variables to the formulation, thereby obtaining a mixed integer program rather than a linear program.

The formulation (3.1) avoids hot and cold spots by applying explicit bounds to the dose on each target voxel. The advantage of this model is its simplicity and flexibility, in that choice of bounds can vary from voxel to voxel, as can choice of penalties in the objective. The disadvantages are that it may be infeasible and that it does not impose DV constraints. Hence the objective may not capture well the relative desirability of different treatment plans.

We can compress (3.1) by eliminating the $x_{\mathcal{N}}$ variable to obtain

$$\min_{w} (A_{\mathcal{N}}^T c_{\mathcal{N}})^T w \ \text{ s.t.} \tag{3.2a}$$

$$x_{\mathcal{T}} = A_{\mathcal{T}} w, \tag{3.2b}$$

$$A_{\mathcal{N}} w \le x_{\mathcal{N}}^U, \tag{3.2c}$$

$$x_{\mathcal{T}}^L \le x_{\mathcal{T}} \le x_{\mathcal{T}}^U, \tag{3.2d}$$

$$w \ge 0. \tag{3.2e}$$

We call this form the *reduced primal model*. The $x_{\mathcal{T}}$ variable could also be eliminated, leaving w as the only variable but yielding the following two general inequality constraints: $A_{\mathcal{T}} w \le x_{\mathcal{T}}^U$ and $A_{\mathcal{T}} w \ge x_{\mathcal{T}}^L$. Although it reduces the number of unknowns, this formulation replaces a two-sided bound with two general constraints, so its benefits are dubious. In any case, since the target region is typically much smaller than the normal region, the effect of this reduction on solution time is not great.

The dual of (3.1) can be written as follows:

$$\max_{\lambda, \mu_L, \mu_U} -(x_{\mathcal{N}}^U)^T \mu_B + (x_{\mathcal{T}}^L)^T \mu_L - (x_{\mathcal{T}}^U)^T \mu_U \ \text{ s.t.} \tag{3.3a}$$

$$\mu_T + \mu_L - \mu_U = 0, \tag{3.3b}$$

$$\mu_N - \mu_B = c_{\mathcal{N}}, \tag{3.3c}$$

$$-A_{\mathcal{T}}^T \mu_T \le A_{\mathcal{N}}^T \mu_N, \tag{3.3d}$$

$$\mu_L, \mu_U, \mu_B \ge 0. \tag{3.3e}$$

We can eliminate the equality constraints by substituting for μ_T and μ_N to obtain the *reduced dual form*:

$$\max_{\lambda, \mu_L, \mu_U} -(x_{\mathcal{N}}^U)^T \mu_B + (x_{\mathcal{T}}^L)^T \mu_L - (x_{\mathcal{T}}^U)^T \mu_U \ \text{ s.t.} \tag{3.4a}$$

$$A_{\mathcal{T}}^T (\mu_L - \mu_U) \le A_{\mathcal{N}}^T (c_{\mathcal{N}} + \mu_B), \tag{3.4b}$$

$$\mu_L, \mu_U, \mu_B \ge 0. \tag{3.4c}$$

In our computational experiments, we formulated each of the three models (3.1), (3.2), and (3.4) explicitly in GAMS and passed them to CPLEX. We also solved the full-size model (3.1) with the `predual` option set; this forces GAMS to formulate the

dual of this model internally before calling CPLEX. Note that we did not perform extensive experiments with the full-sized dual (3.3) since this model is obviously inefficient. (Indeed, our limited experiments showed that the majority of time in solving (3.3) was consumed in the presolve phase, in removing most of the rows and columns from the formulation.)

Our computational results are reported in detail in Section 5.1. We note here that when the bound (3.1d) on normal voxel dose is removed, the problems can be solved very rapidly. This observation motivates an practical approach in which we obtain a warm start for the actual problem by first solving the simplified problem with $x^U_{\mathcal{N}} = \infty$. We report on some experience with this approach in Section 5.1.

3.2. Model II: A Formulation with DV Constraints. We now consider a linear programming formulation that arises when DV constraints are present. As mentioned earlier, such constraints typically have the form that no more than a fraction f of the voxels in a critical region receives a dose higher than a specified threshold δ. This type of constraint was apparently first suggested by Langer and Leong in [12]. An exact formulation can be obtained by means of binary variables as follows. First, we denote the critical region by $\mathcal{C}$ (with n_c voxels) and dose matrix for this region by A_c. Introducing the binary vector χ_c (with n_c components, each of which must be either 0 or 1), we formulate the constraint as

$$x_c = A_c w, \qquad x_c \le \delta e_c + M\chi_c, \qquad e_c^T \chi_c \le f n_c, \qquad \chi_c \in \{0,1\}^{nc}, \tag{3.5}$$

where x_c is the dose vector for the critical region, M is a large constant and e_c is the vector of all 1s and dimension n_c. The components for which $\chi_i = 1$ are those that are allowed to exceed the threshold. A formulation of this type was first proposed by Langer et al. [11, p. 889] and has since appeared in Langer et al. [13, p. 959], Shepard et al. [21, p. 738] and Preciado-Walters et al. [18, Eq. 6].

Lee, Fox, and Crocker [14] use the model (3.5) and add a similar type of constraint to the target (requiring, for instance, that at least 95% of target voxels receive the prescribed dose). They devise a specialized branch-and-bound solver that uses column generation on the linear programming relaxations disjunctive cuts, and various heuristics. Their computation times indicate that the problems are quite difficult to solve. Preciado-Walters et al. [18] solve the mixed-integer program for the IMRT problem, using a column generation procedure in which each generated column is the dose distribution from a certain aperture consisting of a subset of beamlets, chosen using dual-variable information to be potentially useful for the problem at hand. Langer et al. [13] uses a heuristic based on solving a sequence of linear programs, using dual information to decide which voxels should have doses below the threshold. (They compare this approach to simulated annealing.)

Another approach (Shepard [20]) is to start by solving a problem like the one in (3.1) without the upper bound in (3.1d), applying a uniform penalty to all voxels in $\mathcal{C}$. If too many critical voxels have doses above the threshold, a new linear program is formulated (see below) in which the dose in excess of the threshold is penalized for some of the above-threshold voxels. In fact, we can form a sequence of similar linear programs, varying the penalties and the threshold values, until we obtain a solution that satisfies the original DV constraint.

A typical linear program arising in the course of the heuristic just described (and

possibly others) is as follows:

$$\min_{w,x_{\mathcal{T}},x_{\mathcal{N}},x_{\mathcal{C}},x_{\mathcal{E}}} c_{\mathcal{N}}^T x_{\mathcal{N}} + c_{\mathcal{E}}^T x_{\mathcal{E}} \text{ s.t.} \tag{3.6a}$$

$$x_{\mathcal{T}} = A_{\mathcal{T}} w, \tag{3.6b}$$

$$x_{\mathcal{N}} = A_{\mathcal{N}} w, \tag{3.6c}$$

$$x_{\mathcal{C}} = A_{\mathcal{C}} w, \tag{3.6d}$$

$$x_{\mathcal{T}}^L \leq x_{\mathcal{T}} \leq x_{\mathcal{T}}^U, \tag{3.6e}$$

$$x_{\mathcal{E}} \geq x_{\mathcal{C}} - b, \tag{3.6f}$$

$$w, x_{\mathcal{E}} \geq 0, \tag{3.6g}$$

where b is a vector of thresholds for the voxels in $\mathcal{C}$ (different thresholds may apply for different voxels in $\mathcal{C}$), $x_{\mathcal{E}}$ represents the dose to the critical voxels *in excess* of the doses specified in b. The cost vectors $c_{\mathcal{E}}$ and $c_{\mathcal{N}}$ are the penalties applied to excess doses in the $\mathcal{C}$ voxels and to any nonnegative dose in the $\mathcal{N}$ voxels. The threshold vector b and weight vector $c_{\mathcal{E}}$ are the quantities that are manipulated between iterations of the heuristic in an attempt to satisfy the given DV constraints.

The vectors $x_{\mathcal{N}}$ and $x_{\mathcal{C}}$ can be eliminated from (3.6) to obtain:

$$\min_{w,x_{\mathcal{T}},x_{\mathcal{E}}} c_{\mathcal{N}}^T A_{\mathcal{N}} w + c_{\mathcal{E}}^T x_{\mathcal{E}} \text{ s.t.} \tag{3.7a}$$

$$x_{\mathcal{T}} = A_{\mathcal{T}} w, \tag{3.7b}$$

$$x_{\mathcal{T}}^L \leq x_{\mathcal{T}} \leq x_{\mathcal{T}}^U, \tag{3.7c}$$

$$x_{\mathcal{E}} \geq A_{\mathcal{C}} w - b, \tag{3.7d}$$

$$w, x_{\mathcal{E}} \geq 0, \tag{3.7e}$$

which we refer to as the *reduced primal form*.

The dual of (3.6) is

$$\max_{\mu_L,\mu_U,\mu_N,\mu_T,\mu_C,\mu_E} (x_{\mathcal{T}}^L)^T \mu_L - (x_{\mathcal{T}}^U)^T \mu_U - b^T \mu_E \text{ s.t.} \tag{3.8a}$$

$$\mu_T + \mu_L - \mu_U = 0, \tag{3.8b}$$

$$\mu_N = c_{\mathcal{N}}, \tag{3.8c}$$

$$\mu_C - \mu_E = 0, \tag{3.8d}$$

$$\mu_E \leq c_{\mathcal{E}}, \tag{3.8e}$$

$$-A_{\mathcal{T}}^T \mu_T - A_{\mathcal{N}}^T \mu_N - A_{\mathcal{C}}^T \mu_C \leq 0, \tag{3.8f}$$

$$\mu_L, \mu_U, \mu_E \geq 0. \tag{3.8g}$$

By eliminating μ_C, μ_N, and μ_T, we obtain

$$\max_{\mu_L,\mu_U,\mu_N,\mu_T,\mu_C,\mu_E} (x_{\mathcal{T}}^L)^T \mu_L - (x_{\mathcal{T}}^U)^T \mu_U - b^T \mu_E \text{ s.t.} \tag{3.9a}$$

$$0 \leq \mu_E \leq c_{\mathcal{E}}, \tag{3.9b}$$

$$A_{\mathcal{T}}^T (\mu_L - \mu_U) - A_{\mathcal{C}}^T \mu_E \leq A_{\mathcal{N}}^T c_{\mathcal{N}}, \tag{3.9c}$$

$$\mu_L, \mu_U \geq 0. \tag{3.9d}$$

which we refer to as the *reduced dual form*. The target dose matrix $A_{\mathcal{T}}^T$ appears twice in constraint (3.9c), so it is reasonable to ask whether it might be better to avoid elimination of μ_T and leave the constraint (3.9c) in the form (3.8b) and (3.8f).

However, the target dose matrix $A_{\mathcal{T}}$ is often smaller than the critical dose matrix $A_{\mathcal{C}}$, and experiments with the two formulations did not show a significant difference in runtime for the best choices of algorithm and pivot rule.

Presolving may reduce the size of the reduced dual model (3.9) significantly. If, for instance, column j of both $A_{\mathcal{T}}$ and $A_{\mathcal{C}}$ is zero (which would occur if the beam corresponding to this column does not deposit significant dose into any voxels of the target or critical regions), the jth row of (3.9c) can be deleted from the formulation. Similarly, if column j of $A_{\mathcal{T}}$ is zero while column j of $A_{\mathcal{C}}$ contains a single nonzero, the jth row of (3.9c) reduces to a bound, which can be handled more efficiently than a general linear constraint by most linear programming software.

3.3. Model III: A Formulation with Range Constraints and Penalties. Our third formulation contains no DV constraints, but instead specifies a required range for dose to the target voxels, together with a desired dose inside this range. The differences with model I are the inclusion of a penalty term in the objective for any deviation from the desired dose, and the omission of an upper bound on dose to normal voxels.

We write the Model III formulation as follows:

$$\min_{s,t,w,x_{\mathcal{N}},x_{\mathcal{T}}} c_{\mathcal{T}}^T(s+t) + c_{\mathcal{N}}^T x_{\mathcal{N}} \text{ s.t.} \tag{3.10a}$$

$$x_{\mathcal{T}} = A_{\mathcal{T}} w, \tag{3.10b}$$

$$x_{\mathcal{N}} = A_{\mathcal{N}} w, \tag{3.10c}$$

$$x_{\mathcal{T}}^L \leq x_{\mathcal{T}} \leq x_{\mathcal{T}}^U, \tag{3.10d}$$

$$s - t = x_{\mathcal{T}} - d, \tag{3.10e}$$

$$w, s, t \geq 0, \tag{3.10f}$$

where d is the desired dose vector. The variable vector s represents the overdose (amount by which the actual dose exceeds the target dose), while t represents the underdose, so the term $c_{\mathcal{T}}^T(s+t)$ in the objective function penalizes the ℓ_1 norm of the deviation from the prescribed dose. We could modify this model easily to apply a more severe penalty for underdose than for overdose, but for simplicity of description we have chosen to use the same penalty vector $c_{\mathcal{T}}$ for both underdose and overdose.

A similar model to (3.10) is used by Wu [23], except that the objective is replaced by a sum-of-squares measure (see Section 3.4). It is also similar to the form in Shepard et al. [21, p. 733], which includes a bound on the total dose to the critical structures and constraints on the weights w.

Elimination of $x_{\mathcal{N}}$ from (3.10) yields the following *reduced primal form*:

$$\min_{s,t,w,x_{\mathcal{T}}} c_{\mathcal{T}}^T(s+t) + (A_{\mathcal{N}}^T c_{\mathcal{N}})^T w \text{ s.t.} \tag{3.11a}$$

$$x_{\mathcal{T}} = A_{\mathcal{T}} w, \tag{3.11b}$$

$$x_{\mathcal{T}}^L \leq x_{\mathcal{T}} \leq x_{\mathcal{T}}^U, \tag{3.11c}$$

$$s - t = x_{\mathcal{T}} - d, \tag{3.11d}$$

$$w, s, t \geq 0. \tag{3.11e}$$

The dual of (3.10) is as follows:

$$\max_{\mu_T,\mu_N,\mu_U,\mu_L,\mu_D} -(x_{\mathcal{T}}^U)^T\mu_U + (x_{\mathcal{T}}^L)^T\mu_L + d^T\mu_D \text{ s.t.} \tag{3.12a}$$

$$\mu_T + \mu_U - \mu_L - \mu_D = 0, \tag{3.12b}$$

$$\mu_N = -c_{\mathcal{N}}, \tag{3.12c}$$

$$-A_{\mathcal{T}}^T\mu_T - A_{\mathcal{N}}^T\mu_N \geq 0, \tag{3.12d}$$

$$\mu_D \geq -c_{\mathcal{T}}, \tag{3.12e}$$

$$-\mu_D \geq -c_{\mathcal{T}}, \tag{3.12f}$$

$$\mu_L, \mu_U \geq 0, \tag{3.12g}$$

and after an obvious simplification we obtain the following *reduced dual form*:

$$\max_{\mu_T,\mu_U,\mu_L,\mu_D} -(x_{\mathcal{T}}^U)^T\mu_U + (x_{\mathcal{T}}^L)^T\mu_L + d^T\mu_D \text{ s.t.} \tag{3.13a}$$

$$\mu_T + \mu_U - \mu_L - \mu_D = 0, \tag{3.13b}$$

$$A_{\mathcal{T}}^T\mu_T \geq -A_{\mathcal{N}}^T c_{\mathcal{N}}, \tag{3.13c}$$

$$-c_{\mathcal{T}} \leq \mu_D \leq c_{\mathcal{T}}, \tag{3.13d}$$

$$\mu_L, \mu_U \geq 0. \tag{3.13e}$$

We also experimented with a variant of this model in which the single target dose d was replaced by a range $[d_L, d_U]$ (with $x_{\mathcal{T}}^L \leq d_L \leq d_U \leq x_{\mathcal{T}}^U$), and penalties were imposed only if d was outside the interval $[d_L, d_U]$. The computational runtimes for this model were quite similar to those obtained for (3.10), (3.11), (3.12), (3.13), so we do not discuss it further.

3.4. Model IV: A Quadratic Programming Formulation. Model IV is similar to model III, the only difference being that we replace the ℓ_1 penalty for deviating from a prescribed dose by a sum-of-squares term.

Quadratic programming software is less widely available than linear programming software (although this situation is changing for convex quadratic programs, with the recent availability of interior-point codes) and quadratic programs have traditionally been thought of as requiring significantly more computation time than linear programs of similar dimension and sparsity. However, as our results with the CPLEX quadratic programming solver show, this view is not necessarily correct.

By modifying (3.10), defining $\gamma_{\mathcal{T}}$ to be a positive scalar, we obtain the following quadratic model (see Wu [23]):

$$\min_{w,x_{\mathcal{N}},x_{\mathcal{T}}} \frac{1}{2}\gamma_{\mathcal{T}} y^T Q y + c_{\mathcal{N}}^T x_{\mathcal{N}} \text{ s.t.} \tag{3.14a}$$

$$x_{\mathcal{T}} = A_{\mathcal{T}} w, \tag{3.14b}$$

$$x_{\mathcal{N}} = A_{\mathcal{N}} w, \tag{3.14c}$$

$$x_{\mathcal{T}}^L \leq x_{\mathcal{T}} \leq x_{\mathcal{T}}^U, \tag{3.14d}$$

$$y = x_{\mathcal{T}} - d, \tag{3.14e}$$

$$w \geq 0, \tag{3.14f}$$

where Q is a positive definite matrix. When $Q = I$, a sum-of-squares of the elements of $x_{\mathcal{T}} - d$ replaces the ℓ_1 norm of model III.

TABLE 4.1
Pancreatic Data Set: Voxels per Region.

Region—Tissue	# of voxels
Target	1244
Normal	747667
Critical—Spinal Cord	514
Critical—Liver	53244
Critical—Left Kidney	9406
Critical—Right Kidney	6158
Total	818181

By eliminating $x_{\mathcal{T}}$ and $x_{\mathcal{N}}$ we obtain the *reduced primal form*:

$$\min_{w,x_{\mathcal{N}},x_{\mathcal{T}}} \frac{1}{2}\gamma_{\mathcal{T}} y^T Q y + c_{\mathcal{N}}^T A_{\mathcal{N}} w \text{ s.t.} \tag{3.15a}$$

$$x_{\mathcal{T}}^L \le y + d \le x_{\mathcal{T}}^U, \tag{3.15b}$$

$$y = A_{\mathcal{T}} w - d, \tag{3.15c}$$

$$w \ge 0. \tag{3.15d}$$

The corresponding *reduced dual form* is as follows:

$$\max_{y,\mu_U,\mu_L,\mu_D} -\frac{1}{2}\gamma_{\mathcal{T}} y^T Q y + \mu_U^T (d - x_{\mathcal{T}}^U) + \mu_L^T (x_{\mathcal{T}}^L - d) + \mu_D^T d \text{ s.t.} \tag{3.16a}$$

$$\gamma_{\mathcal{T}} Q y + \mu_U - \mu_L + \mu_D = 0, \tag{3.16b}$$

$$-A_{\mathcal{T}}^T \mu_D \ge -A_{\mathcal{N}}^T c_{\mathcal{N}}, \tag{3.16c}$$

$$\mu_L, \mu_U \ge 0. \tag{3.16d}$$

4. Data Sets. In this section we briefly describe the data sets used in experiments with the models of Section 3. For conformal therapy data sets (with relatively few beams), both real data and randomly generated data sets were tried. We used only a real data set for the IMRT case (which has many beams and a sparser dose matrix).

4.1. Conformal Therapy (Random and Pancreatic Data Sets). Our first data set was from a patient with pancreatic cancer (the same set used in Lim et al. [15]), which contained several critical structures (liver, spinal cord, and left and right kidney). Distribution of voxels between the target, critical regions, and normal regions is shown in Table 4.1. Note that the total voxel count is slightly different from the sum of the voxels in all regions, because there are 52 voxels that are counted as belonging to both the liver and to the right kidney.

We used 36 beams in the model, where each beam is aimed from a different angle around the patient (angles separated by 10°). The beam from each angle is shaped to match the profile of the tumor, as viewed from that angle. The full dose matrix has only 36 columns (one for each beam) but more than 800000 rows (one for each voxel). We set the entry in the dose matrix to zero if its dose was less than 10^{-5} of the maximum dose in the matrix. The dose matrix has many zeros but is still quite dense, since each of the 36 beams delivers dose to a large fraction of the voxels in the treatment region.

The random data set has similar dimensions and properties to the pancreatic set. Since we have control over the various dimensions of the problem (see Table 4.2), we

TABLE 4.2
Random Data Set: Voxels per Region.

Region	# of voxels
Target	500
Normal	100000
Critical	15000
Total	115500

TABLE 4.3
IMRT Data Set: Voxels per Region.

Region—Subclass	# of voxels
Target—Target	884
Target—Regional	4246
Critical—Spinal Cord	406
Critical—Parotids	692
Normal	17772
Total	24000

are more able to infer the effects of aggregation and preprocessing performed by the software than for the real data set (indeed, this was a primary motivation for working with the random set). We used a fully dense dose matrix whose entries were drawn from a uniform distribution on the interval $[0, 1]$.

4.2. IMRT Data Set (Nasopharyngeal). In intensity modulated radiation therapy (IMRT), each beam is split into *pencil beams* or *beamlets*, usually by dividing its rectangular aperture by a rectangular mesh. A typical data set has 25-200 beamlets from each of 7-72 possible angles, where each beamlet has its own dose distribution. The solutions of the models we describe in Section 3 yield a weight for each beamlet. A postprocessing procedure known as *leaf sequencing* must then be applied to translate these individual weights into a sequence of deliverable aperture shapes.We refer the interested reader to Boland, Hamacher, and Lenzen [2], Kalinowski [10], and Engel [5] for leaf sequencing algorithms.

Our data set for IMRT is a case of a nasopharyngeal tumor, also used by Wu [23]. There are 51 beam angles, with 39 beamlets from each angle, giving a total of 1989 beamlets (that is, 1989 columns in the dose matrix). The 24000 voxels are divided into five regions, as shown in Table 4.3. The target region is subdivided into a "target" region containing the actual tumor and a "regional" part, corresponding to voxels near the tumor that we wish to control in the same way as tumor voxels (by specifying target values on their doses, for instance). The critical region is subdivided into the spinal cord and the parotids. In summary, the dose matrix A has 24000 rows and 1989 columns.

5. Computational Results. We now give details of the computational experiments with the models and formulations of Section 3 on the data sets of Section 4, using the software tools described in Section 2. Our analysis of these results indicates that the most obvious formulations and the default algorithmic parameter selections often do not yield the best execution times. We believe that our results can be used to guide the choice of formulations and algorithms for other treatment planning problems based on linear and quadratic programming models.

We have split this section into subsections for each type of model. For each model we discuss separately the results for conformal radiotherapy and IMRT. The experiments are based on comparisons of the solve time for each model by using different formulations and options in CPLEX. We apply both simplex and interior point method to models I, II, and III in turn; to the full-size and reduced variants; and to the primal and dual formulations of each model. Model IV is only solved with the interior point method, for the reduced forms of the problems.

In all the following experiments, we set the normal voxel penalty vector $c_{\mathcal{N}}$ to $e = (1, 1, \ldots, 1)^T$. For the pancreatic data set, we used $x_{\mathcal{T}}^L = 0.95e$ and $x_{\mathcal{T}}^U = 1.07e$ as bounds on the target voxel dose, and $x_{\mathcal{T}}^L = 50e$ and $x_{\mathcal{T}}^U = 75e$ for the IMRT data set. (These bounds were chosen to produce solutions with reasonable characteristics.) The lower and upper bound for the random data were generated by setting every third element of w to 1 and $x = A_{\mathcal{T}} w$. Then

$$x_{\mathcal{T}}^L = 0.65 * \frac{\sum_{i=1}^{|\mathcal{T}|} x_i}{|\mathcal{T}|} \;\; and \;\; x_{\mathcal{T}}^U = 1.35 * \frac{\sum_{i=1}^{|\mathcal{T}|} x_i}{|\mathcal{T}|}. \tag{5.1}$$

A similar technique was used to generate the threshold value b:

$$x = A_{\mathcal{T}} w, \qquad b = \left(\sum_{i=1}^{|\mathcal{C}|} x_i / |\mathcal{C}| \right) e. \tag{5.2}$$

For model III, the penalty vector $c_{\mathcal{T}}$ for the deviation from the desired dose is set to e. In model IV we used $c_{\mathcal{T}} = 2e$.

All experiments were performed on a computer running Redhat Linux 7.2 with a 1.6GHz Intel Pentium 4 processor with cache size 256 KB and total memory 1 GB.

5.1. Model I Results. The parameter specific to model I is the upper bound vector $x_{\mathcal{N}}^U$ on the normal voxel dose. To choose an appropriate value for this bound, we first solved the problem without these bounds. For the pancreatic data set, the highest doses to a voxel in each critical region (measured in relative units) were: .461 (spinal cord), .915 (liver), .111 (left kidney) and .612 (right kidney) (see Table 4.1 for a summary of the voxel distribution for this data set). To ensure that the bounds were active for at least some voxels, we set the components of the bound vectors for these different regions as follows: 1.07 (normal and target regions), .40 (spinal cord), .10 (left kidney), .55 (right kidney). For the IMRT data set, the non-bounded solution had maximum doses to the parotids of 56.15 Gy (where "Gy" denotes "Grey", the unit of radiation) and to the spinal cord of 14.13 Gy. We set the bounds as follows: 75 Gy (target and normal tissue), 50 Gy (parotids), and 10 Gy (spinal cord).

At the solution, the pancreatic data set yielded only 5 normal voxels at their upper bound. As noted earlier, this model solves in a fraction of a second if these upper bounds are removed. Hence, although the upper bounds have little effect on the actual solution of the treatment planning problem in this case, they greatly increase the time required to solve it. The IMRT data set yields a solution with only 10 normal voxels at their upper bound. For the IMRT data, removal of the normal voxel bounds improves the solution time considerably, but not by as much as for the conformal problem. (The fastest solution time for an unbounded model is about a factor-of-2 improvement on the fastest solution times reported below.) Because the dose matrix for this data set is large and sparse, the linear program is nontrivial even when the normal voxel bounds are not present in the problem.

TABLE 5.1
Model I Problem Sizes and Effects of Preprocessing.

		Before presolve		After presolve		Avg. presolve
Data	Formulation	Rows	Columns	Rows	Columns	time (sec)
Conf.	Full Primal	801815	801851	222690	222726	13.4
Conf.	Full Dual	801851	1025785	222726	446660	13.8
Conf.	Red. Primal	239058	1281	72496	1280	1.9
Conf.	Red. Dual	37	819426	36	73740	4.1
IMRT	Full Primal	23999	25604	16260	17435	3.2
IMRT	Full Dual	25174	45820	17435	37651	3.8
IMRT	Red. Primal	16263	6736	16224	6305	2.7
IMRT	Red. Dual	1606	29131	1175	21354	4.0

5.1.1. Presolving. The dimensions of the formulations before and after presolving are shown in Table 5.1. The first line shows that the number of rows in the formulation (3.1) for the Pancreatic data set is approximately equal to the total number of voxels (see Table 4.1), while the number of columns exceeds this by 36, which is the number of components in w. Presolving eliminates all those rows and columns corresponding to components of $x_{\mathcal{N}}$ that are intersected by none of the 36 beams (and therefore yield a zero row in the matrix $A_{\mathcal{N}}$). It also eliminates other rows and columns using other techniques. The dual formulation contains approximately twice as many variables after presolving (see the second line of Table 5.1) because there are two dual variables μ_N and μ_B corresponding to the normal voxels. The reduced primal form (3.2) (line 3 of the table) initially contains only as many rows as there are target voxels and normal voxels intersected by the beams; the zero rows of $A_{\mathcal{N}}$ are eliminated by GAMS prior to calling CPLEX. The number of columns equals the number of target voxels in x_T plus the number of beam weights. General presolving techniques reduce the number of rows by a factor of 3. The reduced dual formulation (3.4) (line 4 of the table) has only as many rows as there are beams. The number of columns is initially the number of normal voxels plus twice the number of target voxels, but presolving reduces this number by more than a factor of 10.

Lines 5-8 of Table 5.1 give results for the IMRT data set. Explanations of the problem dimensions before and after presolving are similar to the pancreatic data set, though the presolving reductions are less dramatic because almost all voxels are intersected by at least one of the beams. The reduction in line 8 comes from the fact that 1605 beams intersect any voxel, whereas only 1175 of these beams intersect a target voxel. The remaining beams correspond to zero rows in A_T^T, which are then eliminated in the preprocessing of the constraint (3.4b).

5.1.2. Conformal Data Set: Pancreatic. Results for CPLEX simplex codes on the Pancreatic data set are shown in Table 5.2. We start by explaining the structure of the table. It consists of two groups of nine lines each. The first group contains results for the full-size primal formulation (3.1) (columns 4 and 5) and the reduced primal formulation (3.2) (columns 6 and 7), while the second group of lines contains results for the formulation (3.1) with the `predual` option set (columns 4 and 5) and the reduced dual formulation (3.4) (columns 6 and 7). Within each group, the first four lines represent results for the dual simplex method with four different pivot rules (standard dual pricing, steepest-edge, steepest-edge with slacks, and steepest-edge with unit initial norms), and the last five give results for the primal simplex method with five different pivot rules (reduced-cost pricing, hybrid reduced-cost and devex pricing, devex, steepest-edge, and steepest-edge with slacks). For each combination of parameter setting, formulation, algorithm, and pivot rule, both the execution time

TABLE 5.2
Model I, Pancreatic Data Set, CPLEX Simplex Results.

			Full size form		Reduced form	
n	Method	Pricing	sec	iter	sec	iter
Primal formulation						
1	d spx	standard	1541.2	223135	10.3	83
2	d spx	st edge	1777.9	221606	9.9	69
3	d spx	st edge/sl	-	-	9.4	40
4	d spx	st edge/no	1766.4	221606	9.7	69
5	p spx	reduced	57.1	101	17.2	153
6	p spx	combined	54.7	101	16.9	155
7	p spx	devex	53.7	92	17.8	136
8	p spx	st edge	69.0	112	20.4	130
9	p spx	st edge/sl	65.8	115	25.0	158
Dual formulation						
10	d spx	standard	61.2	148	18.1	96
11	d spx	st edge	67.7	121	17.5	77
12	d spx	st edge/sl	72.5	121	21.4	119
13	d spx	st edge/no	95.1	189	17.7	77
14	p spx	reduced	39.0	89	13.9	122
15	p spx	combined	38.1	39	13.5	122
16	p spx	devex	37.6	39	13.9	53
17	p spx	st edge	40.2	30	13.7	32
18	p spx	st edge/sl	40.7	38	13.8	32

in seconds and the simplex iteration count are given. Preprocessing is applied in all cases.

The first observation to be made from Table 5.2 is the poor performance of the full-size formulation (3.1) when the dual simplex method is used. (Note that a time limit of 4000 seconds was used; the full-size formulation in line 3 reached this limit and was terminated.) Examination of the log of this run showed that almost all of the more than $220,000$ iterations involved swapping an artificial variable for the constraint (3.1c) with the corresponding component of $x_{\mathcal{N}}$. Apparently dual simplex was unable to form an initial basis without introducing artificial variables. The most efficient combination of formulation and algorithm was the primal reduced form and dual simplex, which required around 10 seconds of run time, regardless of pivot rule. Other results for the reduced primal and reduced dual formulations were only marginally slower. With the exceptions noted above, the full-sized formulations required between 38 and 95 seconds, approximately 13 seconds of which is taken up in the presolve phase (see Table 5.1). It is apparent from these results that the primal-formulation, dual-simplex combination must be avoided, and that the reduced formulations (primal and dual) have a significant advantage over the other full-sized formulations.

As an additional experiment, we tried formulating the full-size dual model (3.3) explicitly in GAMS, rather than using the `predual` option in GAMS to form the dual. The computations results for both primal and dual simplex were different, but generally similar to those in lines 10-18 of Table 5.2, so we do not report them here.

As mentioned earlier, we can warm-start the solution procedure for model I by first solving version of the problem without the upper bound $x^U_{\mathcal{N}}$ on the normal voxel doses. Although this simplified problem solves quickly, we found that the warm start had little effect on the total solution time for the fastest models. Some of the longest runtimes for the full-size models were reduced considerably, but since these improvements are of little practical interest, we do not report them here.

Results for the interior-point code CPLEX Barrier are shown in Table 5.3. The three lines of the table correspond to the models (3.1), (3.2), and (3.4), respectively. The number of iterations together with the total time is shown. We also report times for the major stages of the solution process: presolve time (similar to the times

TABLE 5.3
Model I, Pancreatic Data Set, CPLEX Barrier Results.

Formulation	iter	sec	presolve	barrier	crossover
Full size	75	133.3	14.1	93.3	9.7
Reduced primal	23	20.2	1.9	11.8	3.6
Reduced dual	37	26.5	4.1	13.0	1.1

reported in Table 5.1), time for the interior-point algorithm, and time required in the "crossover" phase in which a simplex-like method is used to move the interior-point solution to a vertex of the feasible region.

The total time required by the reduced primal and reduced dual formulations (lines 2 and 3 of the table) is 20 and 26 seconds, respectively—competitive with the best simplex run times from Table 5.2. The full-sized primal formulation required considerably more iterations (75) and about six times more CPU time. The poor performance of the full-sized variant is due to the 13 seconds of presolving time and the fact that the symmetric-positive-definite matrix to be factored at each iteration has three times as many rows as in the reduced primal case and approximately twice as many nonzeros in the Cholesky factor. (This matrix has the form ADA^T, where A is the constraint matrix and D is a diagonal scaling matrix.) In both the full-sized and reduced primal case, the factorization routine correctly detected that 36 columns in the constraint matrix (specifically, the columns corresponding to w) were dense, and it handled them accordingly. In the reduced dual case, the matrix to be factored at each interior-point iteration has just 36 rows, However, formation of this matrix ADA^T is expensive, since A has many columns.

TABLE 5.4
Model I, IMRT Data Set, CPLEX Simplex Results.

			Full size form		Reduced form	
n	Method	Pricing	sec	iter	sec	iter
Primal formulation						
1	d spx	standard	2046	67091	132	14008
2	d spx	st edge	2826	37400	34	1598
3	d spx	st edge/sl	2579	23034	63	1570
4	d spx	st edge/no	-	-	35	1598
5	p spx	reduced	153	12933	182	16097
6	p spx	combined	153	12933	179	16097
7	p spx	devex	89	11262	98	12873
8	p spx	st edge	209	5407	176	4628
9	p spx	st edge/sl	208	5590	174	4628
Dual formulation						
10	d spx	standard	257	12382	418	26341
11	d spx	st edge	498	7124	108	5918
12	d spx	st edge/sl	441	6282	317	7174
13	d spx	st edge/no	663	9177	107	5918
14	p spx	reduced	53	5992	30	6586
15	p spx	combined	40	2596	30	6586
16	p spx	devex	40	2596	48	2816
17	p spx	st edge	49	1048	46	1192
18	p spx	st edge/sl	90	2510	46	1192

5.1.3. IMRT Data Set. Results for the CPLEX Simplex codes on the IMRT data set are shown in Table 5.4. As for the conformal model, the worst performance is turned in by the dual simplex method on the full-size formulation (3.1). Dual simplex with dual-steepest-norm pricing, reported on line 4 of the table, reaches the runtime limit of 4000 seconds. Remarkably, this happens to be the default combination of algorithm/parameter chosen by GAMS/CPLEX when run without an option file.

This observation suggests that it is crucial to make explicit choices of algorithm and parameter for these models, and not to rely on the default choices of the modeling system and optimization software.

As in the conformal data sets, dual simplex applied to the full-size primal models spends many iterations swapping artificial variables for the constraint (3.1c) with the corresponding component of $x_{\mathcal{N}}$. Since this data set is so different in nature from the conformal data sets (the dose matrix is sparse with 1175 columns, instead of dense with 36 columns), the more efficient formulations and algorithms do not give quite as dramatic an improvement in runtime as in Table 5.2. However, there is still almost a two-order-of magnitude difference between runtime for the dual simplex/full-size primal models and runtime for the best reduced model.

As in Table 5.2, the fastest runtimes were obtained with the reduced models. For the reduced primal model, dual simplex was distinctly faster than primal simplex, while for the reduced dual, primal simplex is better. (These same combinations were the best for the conformal data set too, but for the IMRT data set the advantages are more dramatic.) The choice of pivot rule also had a significant effect on runtime in some cases. For the dual simplex algorithm applied to the reduced primal formulation, two of the steepest-edge rules led to a solution being found in around 35 seconds, while the standard rule required more than 130 seconds. For primal simplex applied to the reduced dual, on the other hand, standard reduced-cost pricing gave slightly faster runtimes than devex or steepest-edge rule—about 30 seconds as opposed to 45-50 seconds.

TABLE 5.5
Model I, IMRT Data Set, CPLEX Barrier Results.

Formulation	*iter*	*sec*	*presolve*	*barrier*	*crossover*
Full	3	time limit reached			
Red. Primal	0	memory exceeded			
Red. Dual	19	101	4.0	94.8	1.1

Results for CPLEX Barrier applied to the IMRT data set are shown in Table 5.5. We obtained a solution to just one of the three models with this code: the reduced dual model (3.4). The full-size model (line 1 of the table) performed three interior-point iterations before reaching the time limit of 4000 seconds. After presolving, the problem had 16,260 rows and 17,345 columns, with 147 columns being designated as "dense" by the linear factorization routine in CPLEX Barrier. This code reported that the Cholesky factor of the ADA^T matrix required about 1.3×10^8 storage locations and about 1.5×10^{12} operations to compute. In the reduced primal formulation (line 2), 150 dense columns were detected, and the estimated storage and computation requirements for the factorization were similar to the full-size case. However, this model terminated with an "out-of-memory" message before performing the first iteration. The reduced dual formulation solved in about 100 seconds, using 19 iterations. Dimensions of the reduced LP were modest (1175×21354) as were storage and computation times for the Cholesky factorization (7×10^5 locations and 5×10^8 operations). Given the large number of columns, however, computation of the ADA^T matrix also contributes significantly to the total computation time. We conclude that the interior-point approach applied to the reduced dual formulation is a reasonable approach, requiring about a factor of 3 more in runtime than the best simplex formulations.

5.2. Model II Results. Model II, which is targeted at formulations with DV constraints, contains two important formulation parameters that affect the difficulty of the problem considerably. One is the threshold value b (3.6f). Lower values of b in our model tend to force more voxels to be above-threshold, thus causing more of the components of the excess vector $x_{\mathcal{E}}$ to be positive. The second important parameter is $c_{\mathcal{E}}$, the penalties applied to the excess doses. Larger values of the components of $c_{\mathcal{E}}$ tend to force more of the voxels below the threshold. For each of our three data sets, we report on values of these parameters that give a wide range of proportions of above-threshold voxels, to illustrate the performance of our methods under these different conditions.

5.2.1. Presolving. The dimensions of our various formulations before and after presolving are shown in Table 5.6. The effects of presolving on the random data set are particularly instructive, since elimination of rows and columns for this set are made possible only by structural considerations and not for any reasons of sparsity in the dose matrices (which are deliberately chosen to be dense). The first row of Table 5.6 suggests that, prior to presolving, the full primal model contains the variables $x_{\mathcal{N}}$, $x_{\mathcal{E}}$, $x_{\mathcal{T}}$, and w; the critical dose vector $x_{\mathcal{C}}$ has apparently been eliminated by GAMS. Presolving eliminates the $x_{\mathcal{N}}$ vector as well, by means of the constraint (3.6c), at a considerable overhead in computation time (approximately 36 seconds). The second row of Table 5.6 indicates that in the dual formulation, the presolved model retains constraints (3.8b) and (3.8f), enforcing (3.8e) as a bound rather than a general constraint. Again, the overhead in computational time is significant. Our reduced primal formulation (3.7) (line 3 in Table 5.6) appears to be identical to the formulation obtained by presolving the full-sized primal formulation (3.6), while our reduced dual formulation (line 4) differs from the presolved version of the full dual by the absence of the $\mu_{\mathcal{T}}$ variables by means of (3.8b). As we see below, the computational benefits obtained by performing this additional elimination by hand appears to be considerable.

TABLE 5.6
Model II, Problem Sizes and Effects of Preprocessing. (Conformal Therapy Used on Random and Pancreatic Data.)

		Before presolve		*After presolve*		*Avg. presolve*
Data	*Formulation*	*Rows*	*Columns*	*Rows*	*Columns*	*time (sec)*
Random	Full primal	115501	115531	15500	15530	35.8
Random	Full dual	100531	131501	530	16500	35.9
Random	Red. primal	15501	15531	15500	15530	0.5
Random	Red. dual	31	16001	30	16000	0.5
Pancreatic	Full primal	858373	858409	29464	29500	16.1
Pancreatic	Full dual	832394	886876	1280	29747	16.2
Pancreatic	Red. primal	70515	70551	29464	29500	0.6
Pancreatic	Red. dual	37	71759	36	30708	0.6
IMRT	Full primal	24001	25606	6228	7403	3.0
IMRT	Full dual	24077	35790	6304	16488	3.2
IMRT	Red. primal	6229	7834	6228	7403	0.9
IMRT	Red. dual	1204	11359	1175	11358	2.2

Presolving has similar effects on the dimensions for the other two data sets (lines 5-12 of Table 5.6), with additional reductions due to sparsity in the dose matrices. We note that the computational times for preprocessing the full models are considerably less than for the random data set, again because of the sparsity of the matrices involved.

5.2.2. Conformal Data Set: Random. Table 5.7 shows results for CPLEX's simplex codes on the random data set, for different values of the parameters b and c_ε, various formulations, and various pivot rules. We first explain the structure of the table briefly. It is divided into four groups of nine lines each. The first two groups (lines 1-14) give results for the parameter setting $b = 0.8$, for which about 53% of the voxels in the critical region exceed b at the solution. Columns 3-6 contain the results obtained by setting $c_\varepsilon = e$, where e is the vector of all 1s of appropriate dimension, while columns 7-10 are for $c_\varepsilon = 8e$.

The third and fourth groups (lines 15-28) give results for $b = 1.0$ (approximately 2.5% of voxels over threshold). The first and third groups present results for the full primal formulation (3.6) (including presolving) and the reduced primal formulation (3.7). The second and fourth groups give results obtained by the GAMS-constructed full-size dual formulation (using the `predual` option), and from the reduced dual formulation (3.9). Note that for each group we also ran experiments using (a) dual simplex/steepest-edge with unit norms and (b) primal simplex/steepest-edge with slacks. We do not report these results in this table or in Tables 5.9 and 5.11 since they add little of interest to our conclusions.

The most striking feature of the results is the wide variation in performance for different formulations, algorithms, and pivot rules. As seen in columns 3-4 and 7-8 of the table, the most "obvious" formulation—the full-size primal model (3.6), with preprocessing—gives inferior performance, even if the best possible pivot rule is chosen, and even if the `predual` option is used to formulate its dual automatically. For all parameter settings, the best execution time is obtained by formulating the reduced dual (3.9) by hand, and using primal simplex with the simplest pivot rule (reduced-cost pricing); see row 4, columns 5-6 and 7-8 in the second and fourth groups. For all parameter settings, this strategy led to a runtime of no more than 5 seconds, while runtimes of more than 10 minutes were observed for other choices of formulation and pivot rule. The number of simplex pivots performed by the optimal strategy in just a few seconds seems remarkable; for example, line 11 shows 14,411 pivots in 3 seconds for $b = 0.8$ and $c_\varepsilon = e$. Closer examination shows that since the constraint matrix has just 30 rows, each pivot requires an update of a factorization of a 30×30 matrix—an inexpensive operation. Moreover, a refactorization of the matrix occurs only about once every 50 iterations. Potentially, the most expensive operation in each simplex iteration is pricing, since there are approximately 16,000 columns in the constraint matrix. Apparently, the partial pricing strategy used by CPLEX ensures that the pricing operation does not examine this entire matrix at each iteration, so this part of the per-iteration cost is also inexpensive in absolute terms.

Several other general observations about the results of Table 5.7 follow.

- For the primal formulations (the first and third groups of lines), the number of iterations required by the presolved version of the full model (3.6) is identical to the number of iterations required by the reduced primal formulation (3.7) that was calculated by hand. This observation confirms that the reductions obtained automatically by the presolver correspond exactly to the ones we found manually in (3.7), and that the initialization of the model was the same in both cases. The additional cost of 40-50 seconds per instance for the full model arises from the cost of preprocessing and the cost of recovering the eliminated variables $x_\mathcal{N}$ after the solution has been obtained.
- The use of primal formulations (3.6) or (3.7) is almost uniformly worse than using the dual formulation (3.9).

TABLE 5.7
Model II, Random Data Set, CPLEX Simplex Results.

		Full size form		Reduced form		Full size form		Reduced form	
Method	*Pricing*	*sec*	*iter*	*sec*	*iter*	*sec*	*iter*	*sec*	*iter*
Primal form, b=0.8		$c_\mathcal{E} = e$, 53% over threshold				$c_\mathcal{E} = 8e$, 52.9% over threshold			
d spx	reduced	102	14569	61	14569	182	15289	141	15289
d spx	st edge	147	10277	102	10277	182	8845	142	8845
d spx	st edge/sl	152	8914	112	8914	193	8406	150	8406
p spx	reduced	465	51989	429	51989	564	60986	516	60986
p spx	combined	492	59019	439	59019	451	56641	420	56641
p spx	devex	328	57177	281	57177	356	60457	303	60457
p spx	st edge	688	51307	647	51307	716	51734	667	51734
Dual form, b=0.8		$c_\mathcal{E} = e$, 53% over threshold				$c_\mathcal{E} = 8e$, 52.9% over threshold			
d spx	reduced	239	22220	136	22315	340	22224	137	22314
d spx	st edge	237	19764	118	19695	200	19768	118	19696
d spx	st edge/sl	411	19698	192	20114	330	19701	193	20123
p spx	reduced	46	16460	3	14411	48	16145	5	14036
p spx	combined	46	16460	3	14411	48	16145	5	14036
p spx	devex	134	15380	71	14681	129	11307	106	11019
p spx	st edge	66	9786	46	8772	116	8659	84	8991
Primal form, b=1.0		$c_\mathcal{E} = e$, 2.5% over threshold				$c_\mathcal{E} = 8e$, 2.5% over threshold			
d spx	reduced	47	1567	6	1567	50	1303	9	1303
d spx	st edge	53	798	12	798	55	874	14	874
d spx	st edge/sl	52	626	11	626	53	602	12	602
p spx	reduced	524	56502	464	56502	486	53483	440	53483
p spx	combined	440	57855	396	57855	436	56943	387	56943
p spx	devex	283	53834	227	53834	270	53711	232	53711
p spx	st edge	593	47720	542	47720	571	46945	526	46945
Dual form, b=1.0		$c_\mathcal{E} = e$, 2.5% over threshold				$c_\mathcal{E} = 8e$, 2.5% over threshold			
d spx	reduced	54	996	8	1042	54	993	8	1042
d spx	st edge	53	891	8	1016	53	892	8	1018
d spx	st edge/sl	57	848	10	884	58	851	10	888
p spx	reduced	45	3689	2	3874	45	2720	2	2263
p spx	combined	44	3689	2	3874	44	2720	2	2263
p spx	devex	45	1493	3	1165	49	1753	8	1561
p spx	st edge	45	1169	3	791	47	824	5	746

- While the best formulation/algorithm/pivot combination was always reduced dual formulation/primal simplex/reduced-cost pivoting, the difference between this combination and the other reduced dual formulations decreases markedly as the number of above-threshold voxels falls. In the fourth group of lines, the runtimes are similar for all algorithm/pivot combinations, while in the second group of lines, variations of more than an order of magnitude can be observed.
- For the dual formulations (3.8) and (3.9), primal simplex uniformly performs better than dual simplex. This is because dual simplex needs to use a phase I procedure to find an initial point, whereas an initial point for primal simplex is obvious (from (3.8c)) and no phase I is needed. The problem becomes simpler when b increases, so the importance of phase I iterations decreases and the run times for dual simplex become more competitive.
- For the reduced primal form (the first and third groups of lines), phase I iterations also explain the runtime differences between primal and dual simplex. In these cases, it is *primal* simplex that requires phase I. For example, in lines 4-7 for $c_\varepsilon = e$ the phase I iterations vary from 25408 to 30554 iterations.
- The "combined" pivot rule in primal simplex generally reverted to the reduced-cost rule, with only occasional devex pivots. Thus, runtimes for the combined rule were almost as good as for the reduced-cost rule.
- There are occasional surprising differences in runtime between similar instances. For example, in line 1, the reduced formulation for $c_\varepsilon = e$ performs

14,569 iterations in 61 seconds while the reduced formulation for $c_\varepsilon = 8e$ performs 15,289 iterations in 141 seconds—a considerably slower rate of iterations/second. The difference seems due mainly to the lesser number of basis refactorizations required in the first case (18 factorizations) as compared to the second case (33 factorizations).

Results for the CPLEX Barrier method for these data sets and parameter choices are shown in Table 5.8. We consider the same choices of problem parameters as in Table 5.7 ($b = 0.8, 1.0$ and $c_\varepsilon = e, 8e$), and the full-size, reduced primal, and reduced dual formulations (3.6), (3.7), and (3.9). Presolving was performed, with the same reductions in dimension as seen in Table 5.6. (The presolve is independent of whether the Simplex or Barrier solvers are used.) Similarly to Table 5.3, the columns of Table 5.8 show the number of interior-point iterations, the total time required, the time spent in presolving, the time spent in the barrier algorithm itself, and finally the time required for the crossover phase.

TABLE 5.8
Model II, Random Data Set, CPLEX Barrier Results.

		$c_\varepsilon = e$					$c_\varepsilon = 8e$				
Form.	*b*	*iter*	*sec*	*pre*	*barr*	*cross*	*iter*	*sec*	*pre*	*barr*	*cross*
Full size	0.8	21	48.0	35.8	4.6	3.4	18	48.0	35.6	4.1	4.0
Full size	1.0	25	48.5	35.6	5.3	3.3	19	47.6	35.7	4.3	3.4
R. primal	0.8	21	6.5	0.1	4.6	1.1	18	7.0	0.5	4.1	1.6
R. primal	1.0	25	7.5	0.5	5.3	1.0	19	6.5	0.5	4.3	1.0
R. dual	0.8	19	4.8	0.5	3.5	0.2	20	5.0	0.5	3.7	0.2
R. dual	1.0	15	3.9	0.5	2.6	0.1	16	4.2	0.5	3.0	0.2

We note that for the reduced models, the total CPLEX Barrier runtimes are similar to the runtimes for the best variants of CPLEX Simplex—just a few seconds in total. The full size model requires about 40 seconds longer, but the difference is entirely due to the cost of preprocessing and of recovering the eliminated variables after solving (the latter is included in "crossover" time). The reduced dual formulation is slightly faster. Note that the matrix to be factored at each iteration of the reduced dual formulation is only 30×30 and completely dense, so the row/column ordering is not an issue. For the primal formulations, the matrix is considerably larger, but it has a special structure—a rank-30 update of an extremely sparse matrix—arising from the fact that the dose matrix A has just 30 dense columns. The factorizer in CPLEX is able to detect and exploit this structure, so formation and factorization of the linear system at each interior-point iteration does not cost much more than for the more compact dual formulation.

5.2.3. Conformal Data Set: Pancreatic. Results for the CPLEX Simplex codes on the pancreatic data set, presented in Table 5.9, showed many of the same patterns observed in the random data set. We report on two different values of the parameter b (.02 and .21), and two different settings for the penalty vector c_ε (e and $8e$). These values produced a range in the proportion of above-threshold critical voxels from about 5% to about 32%. Unlike the random data set, the penalty vector had a considerable effect on the proportion of above-threshold voxels.

Again, we see that the reduced dual formulation (3.9), in conjunction with primal simplex and reduced-cost pricing, yielded the most effective strategy for solving the problem. This strategy required less than 10 seconds of runtime for each of the four parameter combinations tried. As in the random data set, the performance difference between this strategy and other strategies based on the dual formulation fades as the

TABLE 5.9
Model II, Pancreatic Data Set, CPLEX Simplex Results.

		Full size form		Reduced form		Full size form		Reduced form	
Method	*Pricing*	*sec*	*iter*	*sec*	*iter*	*sec*	*iter*	*sec*	*iter*
Primal form, b=0.02		$c_\mathcal{E} = e$, 31.9% over threshold				$c_\mathcal{E} = 8e$, 9.7% over threshold			
d spx	reduced	300	43809	280	43809	286	33333	248	33333
d spx	st edge	332	28995	297	28995	256	15053	225	15053
d spx	st edge/sl	519	28641	498	28641	296	14266	265	14266
p spx	reduced	305	42005	268	42005	211	32060	179	32060
p spx	combined	348	60249	315	60249	232	41896	201	41896
p spx	devex	282	56497	249	56497	178	41774	156	41774
p spx	st edge	545	36123	495	36123	276	20057	228	20057
Dual form, b=0.02		$c_\mathcal{E} = e$, 31.9% over threshold				$c_\mathcal{E} = 8e$, 9.7% over threshold			
d spx	reduced	212	30943	237	45110	207	30966	233	45090
d spx	st edge	201	23536	113	23538	201	23638	112	23648
d spx	st edge/sl	206	17656	96	13479	181	17654	95	13330
p spx	reduced	52	58002	9	54183	59	50343	7	39549
p spx	combined	53	58002	9	54183	59	50343	8	39549
p spx	devex	108	29221	94	29021	169	29302	85	16675
p spx	st edge	224	41201	108	24909	198	20750	109	13897
Primal form, b=0.21		$c_\mathcal{E} = e$, 11.3% over threshold				$c_\mathcal{E} = 8e$, 5.6% over threshold			
d spx	reduced	235	32120	207	32120	139	13784	104	13784
d spx	st edge	213	17799	180	17799	143	8230	110	8230
d spx	st edge/sl	297	16320	262	16320	177	7919	142	7919
p spx	reduced	219	27470	186	27470	210	25509	174	25509
p spx	combined	332	50119	305	50119	226	32673	191	32673
p spx	devex	335	60427	292	60427	165	32191	132	32191
p spx	st edge	556	39652	516	39652	302	24329	270	24329
Dual form, b=0.21		$c_\mathcal{E} = e$, 11.3% over threshold				$c_\mathcal{E} = 8e$, 5.6% over threshold			
d spx	reduced	107	12988	68	13550	107	13115	65	13615
d spx	st edge	109	9998	48	8964	105	10053	46	8937
d spx	st edge/sl	119	8581	72	8810	120	8657	69	8832
p spx	reduced	47	29863	6	35187	49	25222	6	24881
p spx	combined	47	29863	7	35187	49	25222	6	24881
p spx	devex	86	13245	64	14904	108	13090	45	8115
p spx	st edge	138	14166	95	12674	125	10144	64	7135

number of above-threshold voxels drops, but the difference is still up to an order of magnitude in the fourth group of lines. The reason is probably that there are still $5 - 11\%$ of voxels above threshold in this part of the table, whereas for the fourth group of lines in Table 5.7, only 2.5% of critical voxels remain above threshold. Note too that in Table 5.9, results for even the worst dual formulation cases are better than the best cases for the primal formulation.

As for the random data set, the full size model and the reduced primal model became identical after preprocessing, so the number of simplex iterations for these two variants is the same in the first and third groups of lines. The differences in times can again be accounted for by the cost of presolving and of recovering the eliminated variables.

TABLE 5.10
Model II, Pancreatic Data Set, CPLEX Barrier Results.

		$c_\mathcal{E} = e$					$c_\mathcal{E} = 8e$				
Form.	*b*	*iter*	*sec*	*pre*	*barr*	*cross*	*iter*	*sec*	*pre*	*barr*	*cross*
Full size	0.02	34	47	16.1	6.6	7.2	234	77	16.3	40.1	4.0
Full size	0.21	86	52	16.2	14.2	3.9	100	62	16.4	16.4	12.0
R. primal	0.02	34	14	0.7	6.6	5.1	234	44	0.7	40.0	1.9
R. primal	0.21	86	18	0.6	14.3	1.8	100	28	0.7	16.4	9.8
R. dual	0.02	295	38	0.6	35.6	1.1	195	25	0.6	3.6	0.2
R. dual	0.21	29	5	0.6	3.5	0.2	75	10	0.6	8.8	0.2

Table 5.10 shows results obtained with CPLEX Barrier on the same four parameter combinations as reported in Table 5.9. As for the random data set, the results

obtained for the full-sized formulation (lines 1-2 of the table) differ from those obtained from the reduced primal formulation (lines 3-4) only in the cost of presolving and recovery of the eliminated variables. We do however see some marked difference with the random data set (Table 5.8) and also with the IMRT data set (Table 5.12), in that the number of interior-point iterations is unusually large for some parameter combinations—up to 295 iterations, for instance, for the reduced dual formulation applied to $b = 0.02$ and $c_\varepsilon = e$. Still, for all but the latter parameter combination, the reduced dual formulation gave the best solution times. The best CPLEX Barrier times (obtained for the reduced dual formulation) are generally slower than the best CPLEX Simplex times, though still much faster than many of non-optimal CPLEX Simplex variants.

In summary, the results for the random and pancreatic data sets suggest that the best strategies in general for model II applied to conformal data sets are (i) reduced dual formulation/primal simplex/reduced-cost pivoting; and (ii) reduced dual formulation/interior-point method.

TABLE 5.11
Model II, IMRT Data Set, CPLEX Simplex Results.

		Full size form		Reduced form		Full size form		Reduced form	
Method	*Pricing*	*sec*	*iter*	*sec*	*iter*	*sec*	*iter*	*sec*	*iter*
Primal form, b=30		$c_\mathcal{E} = e$, 56.4% over threshold				$c_\mathcal{E} = 8e$, 21% over threshold			
d spx	reduced	55	7434	48	7318	54	6776	58	7629
d spx	st edge	26	2366	22	2322	30	2624	25	2559
d spx	st edge/sl	51	2673	52	2429	56	2907	54	2654
p spx	reduced	193	36830	186	36830	210	38514	201	38514
p spx	combined	161	32950	156	32950	181	35766	172	35766
p spx	devex	161	38741	157	38741	187	44032	183	44032
p spx	st edge	289	18951	286	18951	259	16988	251	16988
Dual form, b=30		$c_\mathcal{E} = e$, 56.4% over threshold				$c_\mathcal{E} = 8e$, 21% over threshold			
d spx	reduced	313	24980	425	31113	294	21956	398	27814
d spx	st edge	170	6303	107	6839	171	6281	109	6715
d spx	st edge/sl	205	6839	207	6931	204	6768	205	6614
p spx	reduced	61	13666	26	9406	97	21803	28	10028
p spx	combined	59	9048	27	9406	91	13701	27	10028
p spx	devex	60	9048	29	3496	90	13701	36	4059
p spx	st edge	67	3936	45	1961	72	3924	44	1774
Primal form, b=50		$c_\mathcal{E} = e$, 12.6% over threshold				$c_\mathcal{E} = 8e$, 2.9% over threshold			
d spx	reduced	48	6949	40	6366	35	4404	27	3980
d spx	st edge	17	1387	17	1964	18	1522	14	1462
d spx	st edge/sl	34	1776	39	2046	32	1589	32	1660
p spx	reduced	186	35700	181	35700	152	29009	151	29009
p spx	combined	156	31489	151	31489	136	26565	132	26565
p spx	devex	155	38051	150	38051	149	35994	145	35994
p spx	st edge	302	20139	299	20139	222	14582	216	14582
Dual form, b=50		$c_\mathcal{E} = e$, 12.6% over threshold				$c_\mathcal{E} = 8e$, 2.9% over threshold			
d spx	reduced	216	19058	277	22245	216	19368	277	23015
d spx	st edge	176	6688	52	4241	164	6230	53	4496
d spx	st edge/sl	179	6064	157	5501	188	6380	160	5658
p spx	reduced	31	6319	22	7379	35	7106	20	6522
p spx	combined	25	3021	22	7379	27	3503	21	6522
p spx	devex	24	3021	24	2864	27	3503	24	2551
p spx	st edge	35	1723	37	1575	37	1796	33	1304

5.2.4. IMRT Data Set. The performance results obtained from the IMRT data are quite different in character from those of the conformal data sets. Again, we tried four parameter combinations—two values of b and two values of c_ε, resulting in between 3% and 56% of critical voxels being above-threshold at the solution. As in the pancreatic data set (but unlike the random data set), the choice of c_ε made a significant difference in the number of over-threshold voxels.

Table 5.11 shows performance results for CPLEX Simplex. While the best strat-

egy for the conformal data sets—reduced dual formulation/primal simplex/reduced-cost pivoting—remains a good choice for the IMRT data, other strategies are as good or better. For *any* choice of pivot rule, the use of reduced dual formulation with primal simplex gives good results; see lines 11-14 and 25-28 of the table. More surprisingly, the use of a *primal* formulation in conjunction with dual simplex gives competitive and even superior performance; see lines 1-3 and 15-17.

Note for this data set that the times per iteration have generally increased over those reported in Tables 5.7 and 5.9, since we are dealing with sparse factorizations of much larger basis matrices than in these conformal data sets.

Interestingly, for the CPLEX Simplex method applied to Model II for all three data sets (Tables 5.7, 5.9, and 5.11), the solution time for the best cases is not sensitive to changes in the parameters b and c_ε. On the other hand, the solution time for the poorly performing simplex variants varies greatly. This observation illustrates the importance of choosing the right combination of method and options.

Note that the proportion over threshold reported in Table 5.11 are obtained from the results for the full-sized formulation. In this case (and only this case) we observed that there were multiple solutions to the problem, which gave rise to slight variations in the number of voxels above threshold.

TABLE 5.12
Model II, IMRT Data Set, CPLEX Barrier Results.

		$c\varepsilon = e$					$c\varepsilon = 8e$				
Form.	*b*	*iter*	*sec*	*pre*	*barr*	*cross*	*iter*	*sec*	*pre*	*barr*	*cross*
Full size	30	13	844	3.03	838	2.04	11	730	3.05	724	1.77
Full size	50	11	735	3.01	729	1.78	13	849	2.88	843	2.34
Red. primal	30	13	865	0.95	862	1.29	11	745	0.95	742	1.25
Red. primal	50	11	745	0.94	742	1.30	13	868	0.94	865	1.60
Red. dual	30	15	53	2.20	49	0.64	17	58	2.20	54	0.67
Red. dual	50	15	52	2.01	48	0.70	18	60	2.14	57	0.32

Results for CPLEX Barrier applied to the IMRT data set are given in Table 5.12. As in the conformal cases, the full-size formulations (lines 1-2) and the reduced primal formulations (lines 3-4) reduce to the same problem after preprocessing. However, both of these formulations are far less efficient than the reduced dual formulation (lines 5-6), which yields runtimes of between 50 and 65 second on the four parameter combinations. These times are within factor of 3 of the best simplex results from Table 5.11. The large difference between the performance of the reduced primal and reduced dual models in Table 5.12 can be accounted for by the much greater time per interior-point iteration associated with the primal formulations. In the primal formulations, the matrix to be factored at each interior-point iteration has dimension 6228 and is dense. Although the dose matrix (which forms 1175 columns of the constraint matrix, is sparse) its outer product with itself is quite dense. The columns of the dose matrix cannot be extracted and handled separately by the linear algebra routines, as happens to the 30 or 36 dense columns in the conformal models. In the reduced dual formulation, on the other hand, the dense matrix to be factored at each iteration has dimension 1175. As can be observed by comparing lines 5-6 of Table 5.12 with lines 1-4, the cost of forming and factoring this matrix is more than an order of magnitude less than the cost of working with the much larger matrix arising in the primal formulations.

5.2.5. Sampling from the Critical Structures. In comparing the results for model I with those for model II, we see that imposition of DV constraints on the critical

TABLE 5.13
Model II, Sampling from the Pancreatic Data Set with Excess Penalty e.

Method	*Form.*	*Pricing*	*b*	*Sampled*	*Prop*	*sec*	*iter*
p spx	red. pri.	devex	0.02	0.125	0.321	10	6569
p spx	red. pri.	devex	0.02	0.250	0.323	12	9877
p spx	red. pri.	devex	0.02	0.500	0.311	68	29160
p spx	red. pri.	devex	0.02	1.000	0.319	245	56497
d spx	red. pri.	st. edge	0.21	0.125	0.156	3	2928
d spx	red. pri.	st. edge	0.21	0.250	0.167	13	7454
d spx	red. pri.	st. edge	0.21	0.500	0.148	45	11309
d spx	red. pri.	st. edge	0.21	1.000	0.113	180	17799
p spx	red. dual	reduced	0.02	0.125	0.321	1	6168
p spx	red. dual	reduced	0.02	0.250	0.323	1	13345
p spx	red. dual	reduced	0.02	0.500	0.311	3	25259
p spx	red. dual	reduced	0.02	1.000	0.319	9	54183
p spx	red. dual	reduced	0.21	0.125	0.156	1	7322
p spx	red. dual	reduced	0.21	0.250	0.167	2	12862
p spx	red. dual	reduced	0.21	0.500	0.148	3	18088
p spx	red. dual	reduced	0.21	1.000	0.113	7	35187

structure voxels makes the linear program considerably harder to solve. Lim et al. [16] showed that a sampling strategy (in which just a subset of critical voxels is included in the formulation, or adjoining voxels are aggregated into a single larger voxel) has the potential to reduce the size and solution time considerably while degrading the quality of the solution only minimally. We used the conformal pancreatic data set to study the effects of these sampling strategies on solve time. We tested only those methods and pivot rules that gave the best results in columns 3-6 of Table 5.9 (for which the critical voxel weight vector c_ε is set to e).

Results are shown in Tables 5.13. For each case, we tried sampling four different fractions of the critical voxels in the constraint (3.6f): 0.125, 0.25, 0.5, and 1.0. (Of course, the fraction 1 is simply the original problem reported in Table 5.9.) Steepest-edge with norms pricing was used in lines 5-8. As each group of four lines in these tables shows, sampling caused a dramatic reduction in runtime in each case; in fact, the runtime appears to grow superlinearly with the size of the sample fraction.

The third-last column of the tables shows the proportion of critical voxels over the threshold value. Note that these values fluctuate as the sample size changes. For instance, when $b = 0.21$, 15.6% are over threshold when a sample fraction of .125 is used, compared to a proportion of 11.3% when the full set of voxels is used (sample fraction 1.0). Not surprisingly, the solution is changing as the sample size changes. Since the small-sample formulation is so inexpensive to solve, however, it is cheaper to adjust the parameter b and re-solve this model to achieve the target proportion-over-threshold value than to work with the full model.

5.3. Model III results. Most parameters for model III have already appeared in earlier models, and the same values are used in the experiments reported in this section. The exception is the desired dose d for the tumor voxels. For the random data and the pancreatic data, we set $d = 1.0$. For the IMRT data, the desired dose is based on the sets shown in Table 4.3. The set "Regional" has $d = 60.0$ while "Target" has $d = 70.0$.

Dimensions of the problems and formulations before and after preprocessing are shown in Table 5.14. Note here that the preprocessor was able to find the same reductions that we applied to obtain the reduced primal and dual models from the full-sized primal and dual models; after presolving, the reduced models had the same dimensions as the full models. For the IMRT data set, the presolver was able to eliminate

TABLE 5.14
Model III, Problem Sizes and Effect of Preprocessing. (Conformal Therapy Used on Random and Pancreatic Data.)

		Before presolve		*After presolve*		*Avg. presolve*
Data	*Formulation*	*Rows*	*Columns*	*Rows*	*Columns*	*time (sec)*
Random	Full primal	116001	116531	1000	1530	1.9
Random	Full dual	115531	118001	530	2000	2.0
Random	Red. primal	1001	1531	1000	1530	0.0
Random	Red. dual	531	2001	530	2000	0.0
Pancreatic	Full primal	819426	820706	2488	3768	3.5
Pancreatic	Full dual	818218	824402	1280	4976	3.6
Pancreatic	Red. primal	2489	3769	2488	3768	0.0
Pancreatic	Red. dual	1281	4977	1280	4976	0.0
IMRT	Full primal	29131	35866	10260	16565	1.9
IMRT	Full dual	25175	49198	6304	19636	2.1
IMRT	Red. primal	10261	16996	10260	16565	0.4
IMRT	Red. dual	6306	20521	6304	19636	0.3

approximately 400 variables from the reduced primal model and 900 variables from the reduced dual model.

TABLE 5.15
Model III, Random Data Set, CPLEX Simplex Results.

			Full size		*Reduced*	
Form	*Method*	*Pricing*	*sec*	*iter*	*sec*	*iter*
primal	d spx	reduced	8.9	672	0.5	704
primal	d spx	st edge	8.4	656	0.5	639
primal	d spx	st edge/sl	8.7	585	0.6	574
primal	d spx	st edge/no	8.2	656	0.5	639
primal	p spx	reduced	9.4	1305	1.2	1305
primal	p spx	combined	8.7	1088	1.1	1088
primal	p spx	devex	9.4	1211	1.3	1211
primal	p spx	st edge	9.5	955	1.2	955
primal	p spx	st edge/sl	8.7	853	1.1	853
dual	d spx	reduced	8.7	197	0.6	197
dual	d spx	st edge	8.7	174	0.6	174
dual	d spx	st edge/sl	8.8	174	0.6	174
dual	d spx	st edge/no	8.8	179	0.6	179
dual	p spx	reduced	8.6	651	0.5	193
dual	p spx	combined	8.2	657	0.5	173
dual	p spx	devex	8.7	657	0.5	173
dual	p spx	st edge	8.5	583	0.5	85
dual	p spx	st edge/sl	8.0	634	0.5	115

5.3.1. Conformal Data Sets. Results for the random data set appear in Table 5.15 and in the first 3 lines of Table 5.18. From Table 5.15, we see that for the reduced models the simplex method always required less than 2 seconds to solve this model, no matter what variant of algorithm and pivot rule is used. When the full-size model is passed to the solver, approximately 8 seconds are required for preprocessing. Although, as noted above, both full-size and reduced models result in the same formulation after preprocessing, the iteration counts are sometimes different. This may be due to the use of different starting points. The barrier method (Table 5.18) also takes approximately one second, with an additional 8 seconds required for preprocessing of the full-size model.

The pancreatic data set also yields very fast solutions times, again less than 2 seconds for any simplex variant, with an additional 20 seconds approximately for preprocessing the full-size model; see Table 5.16. Results for the barrier code are shown in the second section (lines 4-6) of Table 5.18. The primal models (full-size and reduced) were not competitive in this case, requiring 12 seconds for the reduced

TABLE 5.16
Model III, Pancreatic Data Set, CPLEX Simplex Results.

			Full size form		*Reduced form*	
Form	*Method*	*Pricing*	*sec*	*iter*	*sec*	*iter*
primal	d spx	reduced	21.6	2444	0.3	2459
primal	d spx	st edge	21.7	1371	0.6	1339
primal	d spx	st edge/sl	21.8	1391	1.2	1405
primal	d spx	st edge/no	21.2	2179	0.6	2219
primal	p spx	reduced	22.6	1476	1.8	1476
primal	p spx	combined	22.8	1375	1.6	1375
primal	p spx	devex	21.3	754	1.1	754
primal	p spx	st edge	22.8	727	1.8	727
primal	p spx	st edge/sl	24.7	1822	3.6	1822
dual	d spx	reduced	21.5	260	0.6	260
dual	d spx	st edge	21.4	141	0.6	141
dual	d spx	st edge/sl	21.3	128	0.6	128
dual	d spx	st edge/no	21.9	253	0.8	253
dual	p spx	reduced	21.8	2489	0.4	1679
dual	p spx	combined	20.7	2316	0.4	1106
dual	p spx	devex	21.6	2316	0.3	1106
dual	p spx	st edge	20.9	842	0.4	1029
dual	p spx	st edge/sl	21.1	2187	0.4	979

model and 34 seconds for the full-size model. The reason for these poor times was the high cost of factoring the coefficient matrix of the linear system to be solved at each interior-point iteration. This matrix has dimension 2488 and its lower triangular factor has approximately 7.7×10^5 nonzeros. The dual formulation yielded a matrix with only 1280 rows, with 4.5×10^3 nonzeros in the L factor. Hence, each interior-point iteration for the reduced dual problem was much less expensive and the code required less than 1 second to solve this formulation.

TABLE 5.17
Model III, IMRT Data Set, CPLEX Simplex Results.

			Full size form		*Reduced form*	
Form	*Method*	*Pricing*	*sec*	*iter*	*sec*	*iter*
primal	d spx	reduced	215	25768	216	25768
primal	d spx	st edge	120	10545	124	10545
primal	d spx	st edge/sl	176	11314	172	11314
primal	d spx	st edge/no	165	10946	110	10946
primal	p spx	reduced	2795	149734	2838	149734
primal	p spx	combined	1220	63747	1225	63747
primal	p spx	devex	1542	240848	1555	240848
primal	p spx	st edge	1266	76789	1238	76789
primal	p spx	st edge/sl	1908	119596	1924	119596
dual	d spx	reduced	204	14198	226	15797
dual	d spx	st edge	225	9746	211	8431
dual	d spx	st edge/sl	193	8666	181	7134
dual	d spx	st edge/no	279	12473	228	11054
dual	p spx	reduced	161	36914	183	39015
dual	p spx	combined	131	21024	152	20464
dual	p spx	devex	131	21024	151	20464
dual	p spx	st edge	180	13261	264	17376
dual	p spx	st edge/sl	488	35290	459	30681

5.3.2. IMRT. The solution times for the IMRT data set were considerably longer than for the conformal data sets, due mainly to the considerably larger size of these models after preprocessing. Results for the simplex method are shown in Table 5.17. The most obvious feature of these results was the poor performance of the primal simplex method on the primal formulation, which yielded run times almost an order of magnitude higher on average than the other formulations and algorithms. The best time was turned in by the dual simplex applied to the reduced primal formu-

TABLE 5.18
Model III, CPLEX Barrier Results.

Data	*Form.*	*iter*	*sec*	*presolve*	*barrier*	*crossover*
Random	Full size	10	9	2.01	0.96	2.67
	R. primal	10	1	0.02	0.94	0.02
	R. dual	16	0	0.02	0.16	0.01
Pancreatic	Full size	12	34	3.70	12.66	2.06
	R. primal	12	12	0.05	12.18	0.09
	R. dual	18	1	0.06	0.62	0.05
IMRT	Full size	10	400	1.88	394.24	2.16
	R. primal	10	408	0.78	405.00	1.35
	R. dual	31	88	0.87	83.69	2.99

lation, using the steepest edge/norms pivot rule (110 seconds). However, most other run times were within a factor of three of this number. In general, the safest strategy is to use the dual formulation, since all run times for this formulation (with the possible exception of primal simplex with the steepest edge/slacks rule) were reasonable.

Results for the barrier code applied to this data set are shown in the last three lines of Table 5.18. For the reduced dual formulation, only 88 seconds are required—faster than any simplex variant. The matrix to be factored at each barrier iteration has dimension 6304 and its lower triangular factor contains about 1.2×10^6 nonzeros. For the primal formulation, the lower triangular factor contains more than ten times as many nonzeros, so even though fewer barrier iterations are required, the total run time is higher.

It is particularly interesting to compare the relative behavior of models II and III on all data sets. For the conformal data set, model III is invariably easy to solve, while model II is fast only for some choices of algorithm and pivot rule (though even then considerably slower than model III). For the IMRT data set, on the other hand, model III is much challenging from a computational point of view. Part of the explanation lies in the dimensions of the problems. For the conformal data sets in model II, the number of columns is between 15,000 and 31,000. Even though the number of rows is as small as 30 or 36 (for the reduced dual formulations), the large number of columns results in many thousands of simplex iterations. Moreover, pricing is more expensive, even when reduced pricing strategies are used. In model III, at most 5,000 columns appear in the formulations, and the number of simplex iterations is rarely greater than 1000. For the IMRT data set, on the other hand, the size of the model III formulations is significantly larger than the model II formulations, and the number of simplex iterations is correspondingly larger as well.

5.4. Model IV results. Model IV is closely related to model III, so in this section we will tend to compare the computational results of these two models rather than to evaluate model IV results in absolute terms. Because of the quadratic objective function in model IV, we cannot use CPLEX's Simplex solver, only the Barrier solver. Since the full-size model results are uniformly slower than those obtained for the reduced model, we report only the latter here.

We set the Hessian Q to be the identity in these tests. The penalty factor $\gamma_{\mathcal{I}}$ was set to 2 while the normal penalty vector $c_{\mathcal{N}}$ was $1e$.

Table 5.19 shows the problem sizes before and after presolving for each data set and formulation. By comparing the reduced primal formulations for model III (3.11) and model IV (3.15) we see that there are $|\mathcal{T}|$ fewer constraints in (3.15). There are also $|\mathcal{T}|$ fewer variables, because the single violation variable y in model IV replaces the two variables s and t in model III. The dimensions of the reduced dual

TABLE 5.19
Model IV, Problem Sizes and Effects of Preprocessing.

Data	*Formulation*	*Before presolve*		*After presolve*	
		Rows	*Columns*	*Rows*	*Columns*
Random	Red. primal	501	531	500	530
Random	Red. dual	531	2001	530	2000
Pancreatic	Red. primal	1245	1281	1244	1280
Pancreatic	Red. dual	1281	4977	1280	4976
IMRT	Red. primal	5131	6736	5130	6305
IMRT	Red. dual	6306	20521	6304	20520

TABLE 5.20
Model IV, CPLEX Barrier Results.

Data set	*Formulation*	*sec*	*iter*
Random	Red. primal	1.3	10
	Red. primal	1.3	10
	Red. primal	1.2	10
	Red. dual	0.5	18
	Red. dual	0.5	18
	Red. dual	0.5	17
Pancreatic	Red. primal	11.8	12
	Red. dual	0.9	22
IMRT	Red. primal	326	8
	Red. dual	47	13

models (3.13), (3.16) are identical, as the variable μ_T in model III is replaced by y in model IV. The preprocessor had essentially no effect in these cases; many of the preprocessing techniques that can be used for linear programs cannot be applied to quadratic programs.

Table 5.20 shows the results for all data sets using model IV. For the random data set, the barrier method converges in 10-18 iterations, and the runtimes are all less than 2 seconds. By comparing with Tables 5.15 and 5.18, we see that model IV appears to be just as easy to solve as model III for this data set.

For the pancreatic data set, the reduced primal model required almost 12 seconds to solve, while the reduced dual model required less than one second. Both times were similar to the times required for the corresponding model III formulations reported in Table 5.18. Again, introduction of the quadratic term in model IV does not appear to have made the problem more difficult to solve than model III.

For the IMRT data set (the last two lines of Table 5.20), the reduced dual model gave the shorter run time; in fact, the time required was only about half of the corresponding model III formulation (see the last line of Table 5.18) and more than a factor of three faster than the best results for the simplex method on model III (see Table 5.17). This exceptionally good result is due in part to the small number of interior-point iterations required—only 13, as opposed to 31 for the corresponding model III result. (Since the number of interior-point iterations required for a given problem is hard to predict or explain in general, we do not speculate on the reasons for this improvement here.) Similarly to the results for the conformal-pancreatic data set, the reduced primal form is much slower to solve than the reduced dual form.

The key observations to be made regarding model IV are as follows. First, despite being a quadratic programming model rather than model III's linear program, it appears to be no more difficult to solve. Second, the reduced dual formulation, while a less obvious way to pose the problem than the reduced primal formulation, gives vastly better computational results.

6. Conclusions. We have performed an extensive computational study of the linear and quadratic programs that arise in treatment planning problems. Our most important conclusions are that the choice of formulation, algorithm, and pivot rule can be crucial to the efficiency of the solution procedure, and that the "default" choices are sometimes unacceptable. We also noted that interior-point methods are often a good choice, and that quadratic programming formulations can be solved (using interior-point methods) as fast or faster than the corresponding linear programming formulations.

It is important to note that solution of linear programs such as those described here represents only a part of the computational and human effort required in the treatment planning process. Calculation of the dose matrices and the cost of setting the model up in GAMS are significant. Moreover, enhancements of the models that require the introduction of binary or integer variables (such as limits on the number of beams or directions that can be used in the treatment plan) open up a new collection of algorithmic and computational issues.

Acknowledgments. We thank our colleagues Steven Dirkse, Michael Ferris, and Jin-Ho Lim for their invaluable advice and help with many aspects of this study. This work was supported in part by National Science Foundation Grants #0196485, #0296033, #0127857, #0113051, #0330538, and #0427689.

REFERENCES

[1] G.K. Bahr, J.G. Kereiakes, H. Horwitz, R. Finney, J. Galvin, and K. Goode. The method of linear programming applied to radiation treatment planning. *Radiology*, 91:686–693, October 1968.

[2] N. Boland, H. W. Hamacher, and F. Lenzen. Minimizing beam-on time in cancer radiation treatment using multileaf collimators. Report Wirtschaftsmathematik, Department of Mathematics, University Kaiserslautern, 2002.

[3] A. Brooke, D. Kendrick, A. Meeraus, and R. Raman. *GAMS: A User's Guide*. GAMS Development Corporation, 1998. `http://www.gams.com`.

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

[5] K. Engel. A new algorithm for optimal multileaf collimator field segmentation. Technical report, Universität Rostock, 2002.

[6] J. Forrest and D. Goldfarb. Steepest-edge simplex algorithms for linear programming. *Mathematical programming*, 57:341–374, 1992.

[7] `http://www.gams.com/contrib/qpwrap/qpwrap.htm`.

[8] P. M. J. Harris. Pivot selection methods of the Devex LP code. *Mathematical programming*, 5:1–28, 1973.

[9] L. Hodes. Semiautomatic optimization of external beam radiation treatment planning. *Radiology*, 1974.

[10] T. Kalinowski. An algorithm for optimal multileaf collimator field segmentation with interleaf collison constraint. Technical report, Universität Rostock, 2003.

[11] M. Langer, R. Brown, M. Urie, J. Leong, M. Stracher, and J. Shapiro. Large scale optimization of beam weights under dose-volume restrictions. *Int. J. Radiation Oncol. Biol. Phys.*, 18(4):887–893, April 1990.

[12] M. Langer and J. Leong. Optimization of beam weights under dose-volume restrictions. *Int. J. Radiation Oncol. Biol. Phys.*, 13(8):1255–1260, August 1987.

[13] M. Langer, S. Morrill, R. Brown, O. Lee, and R. Lane. A comparison of mixed integer programming and fast simulated annealing for optimizing beam weights in radiation therapy. *Medical Physics*, 23(6):957–964, 1996.

[14] E. K. Lee, T. Fox, and I. Crocker. Integer programming applied to intensity-modulated radiation therapy treatment planning. *Annals of Operations Research*, 119:165–181, 2003.

[15] J. Lim. *Optimization in radiation treatment planning*. PhD thesis, Industrial Engineering Department, University of Wisconsin-Madison, 2002.

[16] J. Lim, M. C. Ferris, S. J. Wright, D. M. Shepard, and M. A. Earl. An optimization framework

for conformal radiation treatment planning. Technical report, University of Wisconsin-Madison, 2002.

[17] S.M. Morrill, I.I. Rosen, R.G. Lane, and J.A. Belli. The influence of dose constraint point placement on optimized radiation therapy treatment planning. *Int. J. Radiation Oncol. Biol. Phys.*, 19(1):129–141, July 1990.

[18] F. Preciado-Walters, R. Rardin, M. Langer, and V. Thai. A coupled column generation, mixed integer approach to optimal planning of intensity modulated radiation therapy for cancer. *Mathematical Programming*, 101(2):319–338, 2004.

[19] I. I. Rosen, R. G. Lane, S. M. Morrill, and J. A. Belli. Treatment plan optimization using linear programming. *Medical Physics*, 18(2):141–152, 1991.

[20] D. M. Shepard. Private communication.

[21] D. M. Shepard, M. C. Ferris, G. H. Olivera, and T. R. Mackie. Optimizing the delivery of radiation therapy to cancer patients. *SIAM Review*, 41(4):721–744, 1999.

[22] D. Sonderman and P. G. Abrahamson. Radiotherapy treatment design using mathematical programming models. *Operations Research*, 33(4):705–725, 1985.

[23] C. Wu. *Treatment planning in adaptive radiotherapy*. PhD thesis, Medical Physics Department, University of Wisconsin-Madison, 2002.