Comput Optim Appl (2017) 68:209–242
DOI 10.1007/s10589-017-9919-4

An interior-point implementation developed and tuned for radiation therapy treatment planning

Sebastiaan Breedveld[1] · **Bas van den Berg**[1] · **Ben Heijmen**[1]

Received: 8 February 2016 / Published online: 28 June 2017

Abstract While interior-point methods share the same fundamentals, the implementation determines the actual performance. In order to attain the highest efficiency, different applications may require differently tuned implementations. In this paper we describe an implementation specifically designed for *optimisation in radiation therapy*. These problems are large-scale nonlinear (and sometimes nonconvex) constrained optimisation problems, consisting of both sparse and dense data. Several application-specific properties are exploited to enhance efficiency. Permuting, tiling and mixed precision arithmetic allow the algorithm to optimally process the mixed dense and sparse data matrices (making this step 2.2 times faster, and overall runtime reduction of 55%) and scalability (16 threads resulted in a speed-up factor of 9.8 compared to singlethreaded performance, against a speed-up factor of 7.7 for the less optimised implementation). Predefined cost-functions are hard-coded and the computationally expensive second derivatives are written in canonical form, and combined if multiple cost-functions are defined for the same clinical structure. The derivatives are then computed using a scaled matrix–matrix product. A cheap initialisation strategy based on the background knowledge reduces the number of iterations by 11%. We also propose a novel combined Mehrotra–Gondzio approach. The algorithm is extensively tested on a dataset consisting of 120 patients, distributed over 6 tumour sites/approaches. This test dataset is made publicly available.

Electronic supplementary material The online version of this article (doi:10.1007/s10589-017-9919-4) contains supplementary material, which is available to authorized users.

✉ Sebastiaan Breedveld
s.breedveld@erasmusmc.nl

Ben Heijmen
b.heijmen@erasmusmc.nl

[1] Department of Radiation Oncology, Erasmus University Medical Center - Cancer Institute, PO Box 2040, 3000 CA Rotterdam, The Netherlands

Keywords Nonlinear optimisation · Radiation therapy · Large-scale · Performance optimised · Interior-point · Tiled matrix algebra · Multiple precision arithmetic · Initialisation · Higher-order methods

Mathematics Subject Classification 90C06 · 90C25 · 90C26 · 90C30 · 90C51 · 90C90 · 97M40 · 97M60

1 Introduction

Cancer is diagnosed in 15 million patients globally each year. Despite advances in prevention, detection and treatment, 1.7 million people die yearly of cancer. One of the main (and cost-efficient) modalities for treatment is radiation therapy, which is used in $\approx$50% of cases, mainly when the cancer is localised in a single part of the body as a *tumour*. Radiation therapy is successfully used both for curing patients and giving important symptom relief. However, even with state-of-the-art practice, damage to healthy tissue is unavoidable, and may lead to radiation-induced side-effects. These can have a profound, long-lasting negative impact on the patient's well-being, resulting in high socio-economic costs, e.g. due to required medical treatment or loss of working days, and even in worsened overall survival.

For each patient, a so-called treatment plan is produced based on a 3D Computer Tomography (CT) scan of the patient. The treatment plan contains personalised settings of the applied treatment device, and a predicted patient dose distribution for these settings, projected on the planning CT-scan. The dose describes the probability of physical damage from irradiation. The goal is to deliver a sufficient amount of dose to the tumour for curation, while minimising the (unavoidable) dose to other regions of interest (healthy organs and other tissue), see Fig. 1.

Radiation therapy treatment planning is a multi-criteria problem, where minimising the doses delivered to different regions of interest are conflicting [9,12,54]. Functionality of some organs have a higher influence on the quality-of-life than others, and thus have a higher priority in locally reducing dose. This multi-criterial aspect results in solving a series of optimisation problems, either for Pareto-navigation [19,36] or through automated treatment planning [13,15,33,34,37,68,70].

Another problem in some forms of radiation therapy is the placement of the ionising beam directions [4,15,21,51]. This is a combinatorial problem, often solved through heuristics, requiring solving hundreds or thousands smaller and larger optimisation problems before finding an acceptable solution. When the treatment is based on protons instead of photons, an additional discrete degree of freedom is added to the problem, namely keeping the number of spots and energy layers as low as possible. This is again handled by solving multiple problems [69].

In radiation therapy, optimisation time is also an important aspect. Once the patient is scheduled for treatment, there is a single day at most to complete the treatment plan. Depending on the type of treatment and planning approach, this requires solving one large or multiple smaller problems. An ideal workflow for the clinic is that once the physician (clinician/medical doctor) has delineated the tumour and other important structures (see Fig. 1), a final plan is computed in the order of *minutes*. This allows the

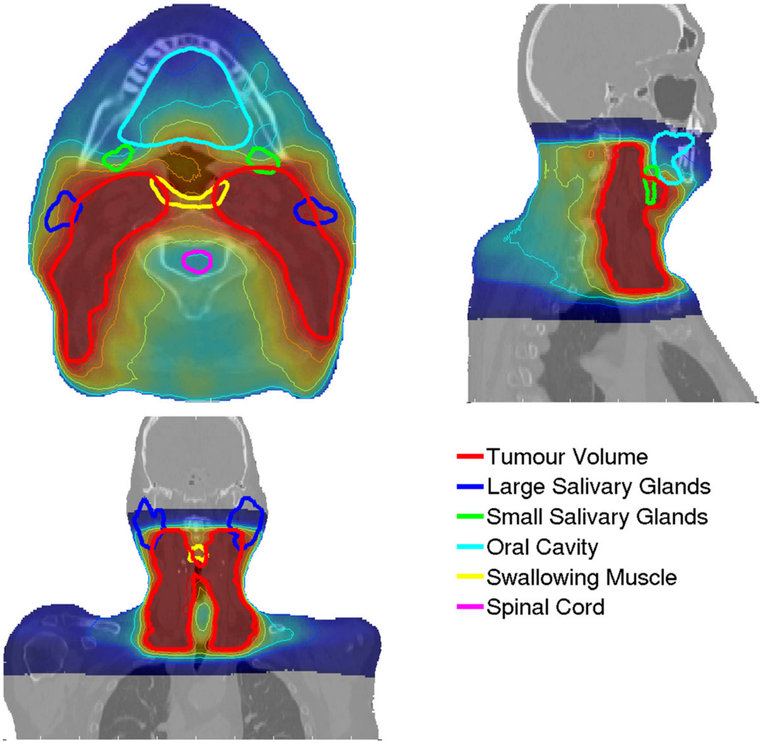

Fig. 1 Different cross-sections of a Computer Tomography-scan with organs-at-risk and tumour in the head-and-neck region delineated. The predicted delivered dose (i.e. treatment plan) is projected onto the CT. *Red* high dose, *blue* low dose (Color figure online)

physician to directly verify the treatment plan, with the patient's background information still in mind, rather than recollecting this the next day. If the optimisation time can be further reduced to the order of *seconds*, a highly desired treatment planning application comes within reach: *online treatment planning*. Radiation therapy is not delivered once, but spread out over up to 40 fractions/days to take advantage of the biological property that healthy cells recover faster than malignant cells. However, the patient's anatomy differs daily, and it is therefore desired to compute a new treatment plan matching the current anatomical situation. Ideally, a new plan should be generated in less than 15 s.

The focus on speed, the traditional interactive form of treatment planning, and the large-scale origin of the problem resulted in a plethora of custom implementations, often favouring speed over optimality (for which [2,3,8,14,18,29,39,53,62,72,74] is a very incomplete list). Few implementations are interior-point based [2,3,53], where [2] is used in a clinical (commercial) treatment planning system (Elekta AB, Sweden). Our decision to use an interior-point method was the support for constrained nonlinear optimisation, stability and robustness for different types of problems, optimality of the solution, the acceptable size of the decision-space, and the leniency towards nonconvex problems. However, full Newton-based implementations are notorious for

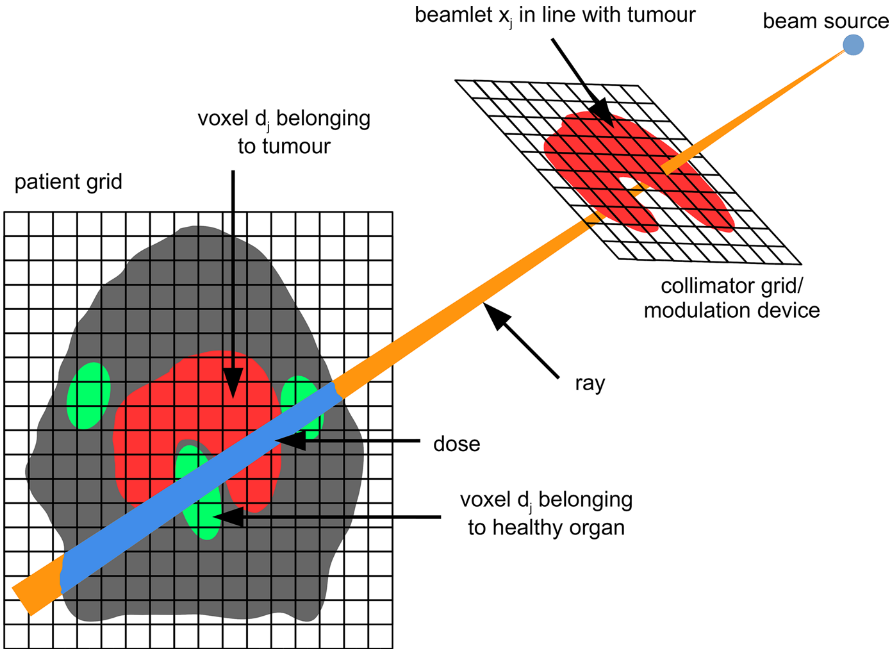
beamlet x_j in line with tumour
beam source
voxel d_j belonging to tumour
patient grid
collimator grid/ modulation device
ray
dose
voxel d_j belonging to healthy organ

Fig. 2 Radiation therapy problem decomposition. Ionising radiation originates from the beam source point and falls onto a collimator. The collimation device allows shaping the beam in different forms and intensities, and is discretised in *beamlets*. The longer a beamlet is "open", the higher the intensity through that beamlet, and the higher the resulting dose in the patient. As soon as the beam enters the patient, the ionising radiation interacts with the tissue, leading to dose (cell damage). The patient is discretised in *voxels*

their computational burden. In this paper, we describe our approaches to improve the computational efficiency, effect on scalability on multicore computer systems, and approaches to reduce the number of iterations.

The approach described in this paper is the mathematical solver used in *Erasmus-iCycle*, our in-house developed software package for multi-criterial radiation therapy treatment plan optimisation, including several extensions to support different clinical techniques and approaches. Erasmus-iCycle is used in the clinical workflow since 2010, and proven to generate treatment plans which are of equal, but often of higher quality compared to the traditional manual trial-and-error treatment planning [56,58, 67–69].

1.1 Data structure and cost-functions

The numerical decomposition of the radiation therapy problem is described in Fig. 2. In general, the *beamlets* (machine parameters) are the *decision-variables* and *cost-functions* are evaluated on the dose in the patient, who is discretised in *voxels*.
The relation between given 2D beamlet intensities x and the 3D dose distribution d is linearly related by:

$$d = Ax \tag{1}$$

where A is called the *pencil-beam matrix*. A single beamlet influences the intensity of an ionising ray, resulting in dose in several parts of the patient (Fig. 2). Rather than working with the dose to the entire patient, the dose is separated per region/structure-of-interest/organ-at-risk, and optimised on by one or more cost-functions.

All clinical cost-functions in radiation therapy take arguments in the dose domain. Therefore, a typical mathematical formulation looks like:

$$\begin{aligned} \underset{x}{\text{minimise}} \quad & f(d_1) \\ \text{subject to} \quad & g_1(d_2) \leq b_1 \\ & d_2 \leq b_2 \\ & b_3 \leq d_3 \leq b_4 \\ & -b_5 \leq B_1 x \leq b_5 \\ & x^T B_2 x \leq b_6 \\ & x \geq 0 \\ \text{where} \quad & d_1 = A_1 x \\ & d_2 = A_2 x \\ & d_3 = A_3 x. \end{aligned} \tag{2}$$

In this formulation, d_1, d_2 and d_3 reflect the doses in different structures, and f and g_1 are general nonlinear nonconvex cost-functions. A typical interpretation is that the dose d_1 to structure 1 is minimised as far as possible, subject to both a nonlinear constraint g_1 and a linear constraint $d_2 \leq b_2$ on structure 2. Structure 3 is the tumour, for which the dose d_3 is at least b_3 (minimum constraint) but does not exceed b_4 (maximum constraint). The relations $d_i = A_i x$ are kept explicit for computational efficiency in the case that multiple cost-functions are imposed on a single structure (see Sect. 4.2).

In addition to the dosimetric problem, there are also constraints on the beamlets x. First, intensities cannot be negative (energy cannot be drawn from body tissue, only added to), so $x \geq 0$ is a fundamental constraint to the problem. Depending on the type of radiation therapy application, there are additional constraints on the values adjacent beamlets can take, due to limitations of the *modulation device* (see Fig. 2). It goes beyond the scope of this paper to describe the physical constraints, but an acceptable approximation is to smooth the adjacent beamlets, preventing spiked beamlet profiles [14]. These Smoothing constraints consists of a linear ($-b_5 \leq B_1 x \leq b_5$) and quadratic ($x^T B_2 x \leq b_6$) type.

Much of the preprocessing steps are done at the highest level. Interior-point methods are known to behave badly if there are duplicate (linear) constraints, or duplicate rows in the data matrices. Because the problem is composed at application-level, the problems of duplicate rows/columns, multiple or redundant applied constraints are unlikely to occur at the level of the mathematical solver.

1.2 Overview

The paper is organised as follows. In Sect. 2, the framework for a general primal-dual interior-point method is derived, including some basic choices relevant to our radiation therapy application. Section 3 introduces some typical extensions, such as initialisation and handling our nonconvex problem. Section 4 describes analytical improvements in constructing the dual-normal matrix, whereas Sect. 5 describes efficient computation for the dual-normal matrix from a numerical point of view. Results are presented in Sect. 6, with discussion of the results and other approaches in Sect. 7.

1.3 Notation

Throughout the paper, the following notation is used. The vector e is the vector of all-ones of appropriate dimension. For a vector d, the capital D denotes a diagonal matrix with d on its diagonal.

Cost-functions are indicated by $f(x)$ for objectives and $g(x)$ for constraints, for which we often drop the dependency on x for readability. The number of constraints is indicated by m, the number of decision-variables x by n.

2 Optimisation model

In this section, we give a concise derivation of our primal-dual interior-point optimisation method for completeness. Our model is based on LOQO [6,57,64]. The approach derived in Sects. 2.1 and 2.2 will be referenced as the *default* method, to distinguish from the higher-order *Mehrotra* and *Gondzio* approaches (Sect. 2.3).

2.1 Primal-dual interior-point method

The basis is the following nonlinear inequality constrained problem with nonnegativity constraint on x:

$$\begin{aligned} \text{minimise} \quad & f(x) \\ \text{subject to} \quad & g(x) \le b \\ & x \ge 0. \end{aligned} \tag{3}$$

Here $f(x)$ and $g(x)$ can contain both linear and nonlinear cost-functions. In practice, $f(x)$ is a scalarisation v_i of cost-functions $f_i(x)$, thus $f(x) = \sum_i v_i f_i(x)$, where each $f_i(x)$ can either be linear or nonlinear. This will be used explicitly in Sect. 4.

The equivalent Fiacco–McCormick logarithmic barrier problem of (3) is:

$$\text{minimise} \quad f(x) - \mu \sum_{i=1}^{m} \log w_i - \mu \sum_{i=1}^{n} \log x_i \tag{4}$$

$$\text{subject to} \quad g(x) + w = b \tag{5}$$

where w is an added slack variable, and the scalar μ represents the duality gap. The Lagrangian of this problem is:

$$L_\mu(x, y, w) = f(x) - \mu \sum_{i=1}^{m} \log w_i - \mu \sum_{i=1}^{n} \log x_i + y^T (g(x) + w - b) \tag{6}$$

where the vector y is a dual variable called the *Lagrange multiplier*. The corresponding Newton system is [64,71]:

$$\begin{pmatrix} -Z^{-1}X & & -I & \\ & W^{-1}Y & & I \\ -I & & H & \nabla g^T \\ & I & \nabla g & \end{pmatrix} \begin{pmatrix} \Delta z \\ \Delta w \\ \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} \gamma_z \\ \gamma_w \\ \gamma_x \\ \gamma_y \end{pmatrix} \tag{7}$$

where z is introduced as a dual component for x. Vector γ represents the first-order optimality conditions, and H is the Hessian:

$$\begin{aligned} \gamma_z &= -\mu Z^{-1} e + xc \\ \gamma_w &= \mu W^{-1} e - y \\ \gamma_x &= -\nabla f - \nabla g^T y + z \\ \gamma_y &= b - g - w \end{aligned} \tag{8}$$

$$H = \nabla^2 f + \sum_{i=1}^{m} y_i \nabla^2 g_i. \tag{9}$$

At a certain iteration, the duality gap is estimated by:

$$\mu = \frac{w^T y + z^T x}{m + n}. \tag{10}$$

After solving (7) (see Sect. 2.2), the step direction $(\Delta z, \Delta w, \Delta x, \Delta y)$ is known for the current point (z, w, x, y). The maximum steplength α is determined such that (z, w, x, y) stays positive. If rescaling for a direction is required, we additionally shorten the steplength by $\tau = 0.995$. For purely linear problems, we use $\alpha_x = \alpha_w = \min(\alpha_x, \alpha_w)$ and $\alpha_y = \alpha_z = \min(\alpha_y, \alpha_z)$. Otherwise, the steplength is chosen as $\alpha = \min(\alpha_z, \alpha_w, \alpha_x, \alpha_y)$, see [64]. Additionally, further steplength reduction may be required for nonconvex or ill-initialised problems, see Sect. 3.3.

The updated point $k + 1$ is then computed by:

$$\begin{pmatrix} z_{k+1} \\ w_{k+1} \\ x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} z_k \\ w_k \\ x_k \\ y_k \end{pmatrix} + \alpha \begin{pmatrix} \Delta z \\ \Delta w \\ \Delta x \\ \Delta y \end{pmatrix}. \tag{11}$$

Based on the steplength, the reduction σ of the duality gap μ (10) for the next iteration is:

$$\sigma = \left(\frac{\alpha - 1}{\alpha + 10}\right)^2. \tag{12}$$

Finally, primal and dual infeasibility are given by:

$$\delta_p = \frac{||\gamma_y||}{||b|| + 1}, \qquad \delta_d = \frac{||\gamma_x||}{||\nabla f(x)|| + 1}. \tag{13}$$

We consider the problem converged when the ℓ^2-norm of the first order optimality conditions (8) is less than $\mathcal{O}(10^{-4})$.

2.2 Solving system (7)

The most time-consuming part of the interior-point iterations are the construction and solution of system (7). Straightforward elimination of Δz and Δw by choosing:

$$\Delta z = -ZX^{-1}(\gamma_z + \Delta x) \tag{14}$$

$$\Delta w = WY^{-1}(\gamma_w - \Delta y) \tag{15}$$

leads to the quasidefinite *reduced Karush–Kuhn–Tucker* system:

$$\begin{pmatrix} H + ZX^{-1} & \nabla g^T \\ \nabla g & -WY^{-1} \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} \gamma_x - ZX^{-1}\gamma_z \\ \gamma_y - WY^{-1}\gamma_w \end{pmatrix}. \tag{16}$$

For purely linear problems, H equals 0, and for quadratic problems H is symmetric positive definite and static. For nonlinear nonquadratic convex problems, H changes in every iteration, and is positive definite. For nonconvex problems, H *may* be indefinite (see Sect. 4.3). ∇g is usually very large (dimensions m by n), and of mixed dense and sparse matrices. ∇g is therefore never constructed explicitly, but kept as separate data matrices, similar to the canonical form for H (Sect. 4).

If the problem is sparse and linear, system (16) can be solved directly using a sparse indefinite Cholesky decomposition [24,64]. Because the radiation therapy problem is fairly dense, the Karush–Kuhn–Tucker system is impractical to construct and solve due to the memory requirements. Therefore, we further reduce this system by eliminating Δy:

$$\Delta y = \gamma_w - W^{-1}Y(\gamma_y - \nabla g \Delta x) \tag{17}$$

and defining:

$$N = H + ZX^{-1} + \nabla g^T (W^{-1}Y) \nabla g \tag{18}$$

$$r = \nabla g^T (W^{-1}Y\gamma_y - \gamma_w) + \gamma_x - ZX^{-1}\gamma_z. \tag{19}$$

The normal equation to solve in each interior-point iteration is:

$$N\varDelta x = r. \tag{20}$$

An alternative is to eliminate $\varDelta x$ instead [26]. This results in a system of size m, but is generally ill-conditioned for radiation therapy problems. It also requires the inverse of $H + ZX^{-1}$, which is expensive to compute for dense H.

Compared to the alternative representations, N as (18) is well-conditioned, and a practical form for this application of nonlinear optimisation problems [57]. For nonconvex problems there is a high probability that the nonconvex part of the problem is dominated by the convex part, still resulting in a positive definite system.

We solve this system by using the Cholesky decomposition, which also scales well accross multiple threads. This decomposition may fail for nonconvex or ill-conditioned problems. In that case, we restore positive definiteness by adding λI to N, starting with $\lambda = 10^{-6}$ and successively doubling λ until success [65].

2.3 Higher-order methods: Mehrotra and novel Gondzio

Higher-order methods aim to reuse the computation and factorisation of N (20) for different right hand sides r, in order to reduce the number of overall iterations. A successful second-order method (predictor–corrector) was proposed by Mehrotra [41,42]. Later, Gondzio et al. [25] extended this to multiple corrections, creating a higher-order approach. In our experience, Gondzio's method did not work well for our application with respect to structurally reducing the number of iterations. We therefore use a novel combined Mehrotra–Gondzio approach, where we first take a full Mehrotra step before applying the Gondzio update scheme. Details can be found in the Supplementary material.

3 Extensions

In this section we describe three extensions to the standard interior-point approach (Sect. 2). Section 3.1 describes an efficient initialisation approach based on the problem background. Section 3.2 describes the use of a special nonconvex cost-function in the optimisation. Steplength control is discussed in Sect. 3.3.

3.1 Initialisation

Proper initialisation of the primal and dual variables are essential for reasonable convergence. In this section we describe our initialisation approaches, by making use of the application's background. We describe both a simple and advanced approach for choosing x_0.

In radiation therapy, the *tumour* is the defining component in the optimisation problem. Without the tumour, the optimal radiation therapy treatment plan would be an all-zero dose. The amount of dose prescribed to the tumour is directly related to the shape of the dose distribution. We therefore focus on the dose to the tumour in our initialisation strategy.

The simple approach is to find a uniform initialisation of x_0. We cannot simply initialise e.g. $x_0 = 10e$ because this could potentially result in very large values. The dose to the tumour is usually modelled using the *Logarithmic Tumour Control Probablity* [2], which exponentially penalises underdosage:

$$\text{LTCP} = \frac{1}{m}\sum_{j=1}^{m} e^{-\alpha(d_j - d^p)} \tag{21}$$

where α sets the cell-sensitivity. A homogeneous dose to the tumour ($d_j = d^p \ \forall j$) results in an LTCP value of 1. If x_0 is too small, the LTCP easily attains values of $\mathcal{O}(10^{10})$, resulting in a badly scaled interior-point problem which will often not even start. Our simple strategy is therefore to start with $x_0 = 100e$, then iteratively increase by a factor 1.5 until all LTCPs are <1000.

The advanced approach includes two treatment plan preferences in the initialisation: a homogeneous dose to the tumour, and a smooth intensity profile. Let $d = A_t x$ be the dose to the tumour, and B_2 a smoothing matrix used by the problem (2). We then solve the following least-squares problem:

$$\begin{pmatrix} A_t \\ B_2 \end{pmatrix} x = \begin{pmatrix} d^p \\ 0 \end{pmatrix} \tag{22}$$

which aims for both properites. This results in the initialisation x_0 given by:

$$x_0 = \max\left\{0, (A_t^T A_t + B_2)^{-1} A_t^T d^p\right\} \tag{23}$$

where we simply add B_2 instead of $B_2^T B_2$, since B_2 is already symmetric and positive definite in our problem definition. Finally, we rescale by:

$$x_0 = \frac{1}{2}x_0 + \frac{1}{2}\text{mean}(x_0) \tag{24}$$

and set elements <5 equal to 5.

An additional effect of the smoothing matrix B_2 is that it also increases the condition of the least-squares problem. For proton-based radiation therapy, smoothing is not required (i.e. each element of x can have a strongly different value from its neighbours), but we add $B_2 = I$ as a regulator, which simultaneously aims at minimising the overall beamlet intensity.

If a patient has multiple tumour volumes with different prescriptions d^p, these matrices are simply concatenated in (22).

The initial solution for the dual variable y is computed using its first order optimality condition γ_x (8):

$$\nabla f(x) + \nabla g(x)^T y - z = 0 \tag{25}$$

where we assume that $\mu = 0$, resulting in $z = \frac{\mu}{x} = 0$. Using x_0 we can compute $\nabla f(x_0)$ and $\nabla g(x_0)$. This leads to the following problem:

$$\nabla g(x_0)^T y_0 = -\nabla f(x_0) \tag{26}$$

which we solve using an iterative least-squares implementation [48]. The iterative least-squares method is implemented in such a way that the required multiplications with ∇g are decomposed in a series of individual matrix multiplications (using the techniques presented in Sect. 4).

Once x_0 and y_0 have been computed, we compute z_0 by using (25):

$$z_0 = \nabla f(x_0) + \nabla g(x_0)^T y_0 \tag{27}$$

and similar for w_0 by again using the first order optimality condition γ_y (8):

$$w_0 = b - g(x_0). \tag{28}$$

Similar to x_0 we set elements of y_0, z_0 and w_0 which are <5 to 5.

3.2 Nonconvex cost-function

For some organs, the remaining functionality after radiation therapy is strongly correlated with the volume that receives a certain amount of dose. For example, the liver is considered functional if at least 700 *cc* receives a dose less than 15 *Gy* (*Gy* or Gray, measure for absorbed dose). If a larger volume is damaged, there is a considerable probability that the liver has insufficient capacity to maintain functionality. Lungs and kidneys behave similarly.

A visual example of such a case is given in Fig. 3a. This constraint is called *dose-volume*, *partial volume* or *coverage* constraint. The coverage is determined by counting the number of voxels d_i receiving a dose $d_i > d^c$, d^c being the *critical dose level*.

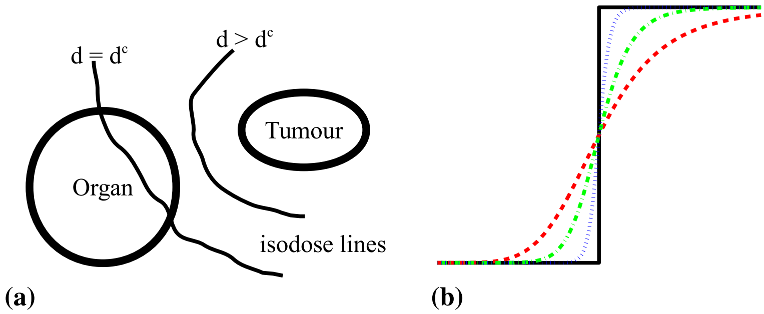

Fig. 3 **a** Example of the coverage cost-function. The tumour receives the highest dose, which decreases with distance from the tumour (see also Fig. 1). The two *isodose lines* represent the levels where the dose takes the same value. At the left of *isodose line* d^c, dose is lower than d^c. In this case, ≈ 20% of the organ receives a dose higher than d^c. **b** Sigmoid-like approximations for the indicator function. The inflexion point is at the critical dose level d^c

The formal description is an accumulation of indicator functions:

$$DV_{\text{exact}}(d) = \frac{1}{m}\sum_{i=1}^{m} I_{d_i<d^c}(d_i) \tag{29}$$

resulting in the fraction of the volume (containing m voxels) receiving dose d less than the critical dose d^c.

The problem is that this cost-function is not convex and not smooth. As a result of the degeneracy of the radiation therapy problem [1] and the fact that (physical) dose is continuous and differentiable, the number of local minima is expected to be limited [40]. There has been active research in directly incorporating dose-volume cost-functions, including mixed-integer programming [31], conditional value at risk (CvaR) [53] or using a series of generalised mean constraints [45,75].

We have adopted the more direct approach proposed by [2], who uses a smoothed version of the indicator version (29). The slope can be set with the parameter p:

$$DV_{\text{approx}}(d) = \frac{1}{m}\sum_{i=1}^{m} \frac{\left(\frac{d_i}{d^c}\right)^p}{1+\left(\frac{d_i}{d^c}\right)^p} \tag{30}$$

where a higher parameter results in a more accurate approximation of the indicator function, see Fig. 3b. In this work, we use a parameter of $p = 5$.

In our approach, we assume that the dose-volume cost-function is used as *constraint*. The reason is as follows: the dose-volume cost-function prescribes a very precise volumetric treatment objective. If one simply wants to minimise the volume below a certain critical dose, it is better to use a convex approach, such as the generalised mean:

$$f_{\text{gmean}}(x) = \left(\frac{1}{m}\sum_{i=1}^{m} d_i(x)^a\right)^{\frac{1}{a}}. \tag{31}$$

The parameter $a \geq 1$ is used to focus on certain part of the dose values, where a higher a controls the high part of the dose distribution. In practice, f_{gmean} has proven sufficient to replace most of the clinical dose-volume guidelines.

3.3 Steplength control and step rejection

By default, interior-point methods take a full step $\alpha = \alpha_{max}$ in (11) the descent directions as given in Sect. 2.1, which is only scaled to prevent the slack and dual variables from becoming negative. For nonconvex problems, steplength control is essential for the algorithm to converge to a local minimum [6,65]. For convex problems, we introduce another type of steplength reduction to reduce the effect of ill-initialised problems.

3.3.1 Markov filter

For nonconvex problems, we use the Markov filter as described in [6]. Let the barrier function b_μ be defined by:

$$b_\mu = f(x) - \mu \sum_{i=1}^{m} \log w_i - \mu \sum_{i=1}^{n} \log x_i \tag{32}$$

and the infeasiblity given by:

$$\gamma_y = b - g(x) - w. \tag{33}$$

The Markov filter [6] aims to find a steplength $\alpha_{k+1} \in (0, \alpha_{max}]$ that ensures a reduction in either the barrier function or infeasibility, satisfying an Armijo condition. Thus for $\mu = \mu_k$ and $\epsilon = 10^{-6}$, either

$$b_{\mu_k}^{(k+1)} - b_{\mu_k}^{(k)} < \epsilon \alpha_{k+1} \begin{pmatrix} \nabla_x b_{\mu_k}^{(k)} \\ \nabla_w b_{\mu_k}^{(k)} \end{pmatrix}^T \begin{pmatrix} \Delta x_k \\ \Delta w_k \end{pmatrix} \tag{34}$$

or

$$||\gamma_y^{(k+1)}||_2^2 - ||\gamma_y^{(k)}||_2^2 < -2\epsilon\alpha_{k+1}||\gamma_y^{(k)}||_2^2 \tag{35}$$

should be satisfied if $||\gamma_y^{(k+1)}||_2^2$ is not already smaller than the requested primal precision ($<10^{-3}$). The last expression is achieved by carefully working out the Armijo condition for γ_y [65], resulting in a computationally cheap filter. We successively scale α_{k+1} by 0.9 for at most 10 times.

It depends on the type of higher-order update method whether or not this rejected step is used. If the default method is used (without higher-order updates), then the small step is simply taken as this may result in escaping the local region where the quadratic approximation of the nonlinear problem is poor. When the step is rejected in the Mehrotra approach, the Default direction is computed and a new steplength is determined for that direction. For Gondzio, we simply stop the updating scheme if a step is rejected.

3.3.2 Ratio control

For convex problems, enabling the Markov filter for steplength control results in general in unnecessary steplength reductions, and consequently to a higher number of iterations. For some problems however, steplength reduction resulted in less iterations. We observed that the objective function value often increased by more than a factor 1000, with only a negligible decrease in the infeasibility. In other cases, the infeasibility increased in favour of a small reduction in the objective function value. Inspired by a suggestion in [6], we search for a step α_{k+1} that satisfies:

$$\frac{f^{(k+1)}}{f^{(k)}} \frac{\gamma_y^{(k+1)}}{\gamma_y^{(k)}} < 20. \tag{36}$$

We scale α_{k+1} by 0.9 for at most 20 times, and only apply the ratio control during the 10 first iterations for performance reasons, which seem to be sufficient.

4 Construction of the dual-normal matrix

The construction of the dual-normal matrix N is the most time-consuming part of a full-Newton interior-point iteration. For completeness, substitute H from Eq. (9) with Eq. (18), and assume that the objective function is actually a scalarisation with weights v_i of several cost-functions:

$$N = \sum_{i=1}^{k} \nabla^2 v_i f_i + \sum_{i=1}^{m} y_i \nabla^2 g_i + \sum_{i=1}^{m} \nabla g_i^T w_i^{-1} y_i \nabla g_i + ZX^{-1}. \tag{37}$$

Here, k is the number of objective functions and m the number of constraints (both linear and nonlinear). The third term computes for each constraint i the scaled rank-1 update of ∇g_i^T.

This computation can be simplified to the *condensed* representation:

$$N = A^T DA + Q + T \tag{38}$$

where D and $T = ZX^{-1}$ are diagonal matrices, Q a symmetric matrix and A the matrices originating from the problem data. A generally represents the pencil-beam dose matrices, $d_i = A_i x$ [see (2)]. This representation is used to efficiently compute the Hessian for the (nonlinear) objectives and constraints [first 2 terms of (37)] and the rank updates for the first order derivatives of the constraints [third term of (37)]. This representation allows a very efficient computational implementation (Sect. 5), but we will first describe the analytical advantages in the next section.

4.1 Expansion

First we expand (37) in explicit terms of *linear*, *nonlinear* and *symmetric positive definite* (resulting from quadratic objectives or constraints). Let $\mathcal{L}_o$ and $\mathcal{L}_c$ be the set of linear objectives and constraints respectively, $\mathcal{N}_o$ and $\mathcal{N}_c$ the sets of nonlinear cost-functions (except quadratic), and $\mathcal{Q}_o$ and $\mathcal{Q}_c$ the sets of quadratic cost-functions. For linear constraints, it is assumed that there are $\mathcal{L}_c$ "big" matrices A_i, containing $\mathcal{J}_i$ rows. Equation (37) then becomes:

$$\begin{aligned} N = {} & \sum_{i\in\mathcal{L}_o} v_i \nabla^2 f_i && + \sum_{i\in\mathcal{N}_o} v_i \nabla^2 f_i && + \sum_{i\in\mathcal{Q}_o} v_i \nabla^2 f_i \\ & + \sum_{i\in\mathcal{L}_c} y_i \nabla^2 g_i && + \sum_{i\in\mathcal{N}_c} y_i \nabla^2 g_i && + \sum_{i\in\mathcal{Q}_c} y_i \nabla^2 g_i \\ & + \sum_{i\in\mathcal{L}_c} \sum_{j\in\mathcal{J}_i} \nabla g_{ij}^T w_{ij}^{-1} y_{ij} \nabla g_{ij} && + \sum_{i\in\mathcal{N}_c} \nabla g_i^T w_i^{-1} y_i \nabla g_i && + \sum_{i\in\mathcal{Q}_c} \nabla g_i^T w_i^{-1} y_i \nabla g_i \\ & + ZX^{-1}. \end{aligned} \tag{39}$$

For linear cost-functions the second derivative equals 0, so they can be removed from the equation.

The cost-functions used in radiation therapy take their arguments in the dose domain, which has a linear relation to the decision-variables x (2). For a certain cost function $h(d_i)$ (either used as an objective or constraint), where:

$$d_i = A_i x \tag{40}$$

then the first and second derivatives can be expressed in the following canonical form [32]:

$$\begin{aligned} \nabla h &= c_{i1}(x) A_i^T x \\ \nabla^2 h &= c_{i2}(x) A_i^T E_i(x) A_i + c_{i3}(x) \nabla h \nabla h^T \end{aligned} \tag{41}$$

where $c_{i1},\ c_{i2}$ and c_{i3} are scalars and E_i a diagonal matrix. For the remainder of this section, we drop the dependence on x for readability.

For quadratic functions, we use the canonical form:

$$h(x) = \frac{1}{2} x^T B x + b^T x + c \tag{42}$$

resulting in:

$$\begin{aligned} \nabla h &= Bx + b \\ \nabla^2 h &= B \end{aligned} \ . \tag{43}$$

Substituting the linear relation (40) and derivatives (41–43) into (39) gives:

$$\begin{array}{llll} N = & \sum_{i \in \mathcal{N}_o} v_i c_{i2} A_i^T E_i A_i + v_i c_{i3} \nabla f_i \nabla f_i^T & + \sum_{i \in \mathcal{Q}_o} v_i B_i \\ + & \sum_{i \in \mathcal{N}_c} y_i c_{i2} A_i^T E_i A_i + y_i c_{i3} \nabla g_i \nabla g_i^T & + \sum_{i \in \mathcal{Q}_c} y_i B_i \\ + \sum_{i \in \mathcal{L}_c} A_i^T W_i^{-1} Y_i A_i + & \sum_{i \in \mathcal{N}_c} \nabla g_i^T w_i^{-1} y_i \nabla g_i & + \sum_{i \in \mathcal{Q}_c} \nabla g_i^T w_i^{-1} y_i \nabla g_i \\ + Z X^{-1}. & & \end{array} \tag{44}$$

Rearranging to group terms that have similar operations results in:

$$\begin{array}{llll} 1. & N = \sum_{i \in \mathcal{N}_o} v_i c_{i2} A_i^T E_i A_i & + \sum_{i \in \mathcal{N}_c} y_i c_{i2} A_i^T E_i A_i & + \sum_{i \in \mathcal{L}_c} A_i^T W_i^{-1} Y_i A_i \\ 2. & \quad + \sum_{i \in \mathcal{N}_o} v_i c_{i3} \nabla f_i \nabla f_i^T & + \sum_{i \in \mathcal{N}_c} y_i c_{i3} \nabla g_i \nabla g_i^T & \\ 3. & \quad + \sum_{i \in \mathcal{N}_c} \nabla g_i^T w_i^{-1} y_i \nabla g_i & + \sum_{i \in \mathcal{Q}_c} \nabla g_i^T w_i^{-1} y_i \nabla g_i & \end{array}$$

$$\begin{array}{lll} 4. & +\sum_{i\in\mathcal{Q}_o} v_i B_i & +\sum_{i\in\mathcal{Q}_c} y_i B_i \\ 5. & +ZX^{-1}. & \end{array} \tag{45}$$

This formulation shows different layers of the underlying computational complexity. The rows contain consecutively: 1. matrix-matrix products, 2. vector outer-products, 3. vector dot-products, 4. symmetric positive definite terms, and 5. a diagonal term. The matrix Q is described by rows 3 and 4, and T by row 5. The advantage of this formulation is described in the next section.

4.2 Computational efficiency

The most time-consuming part of Eq. (45) are the matrix-matrix products. Any reduction in computational effort gained here is directly measurable. Remember that each matrix A_i represents the dose to a certain structure (i.e. the tumour, or one of the healthy organs). It often occurs that for a single structure, multiple cost-functions are used. For example, the tumour can both have a minimum and maximum dose, and has a nonlinear objective that minimises the probability of recurrence (i.e. due to insufficient irradiation). This problem is formalised as follows:

$$\begin{array}{rl} \text{minimise} & f(d_1) \\ \text{subject to} & b_1 \leq d_1 \leq b_2 \\ & x \geq 0 \\ \text{where} & d_1 = A_1 x. \end{array} \tag{46}$$

The first row of Eq. (45) for this problem is:

$$v_1 c_{12} A_1^T E_1 A_1 + A_1^T W_2^{-1} Y_2 A_1 + A_1^T W_3^{-1} Y_3 A_1 \tag{47}$$

where E_1 is concerned with the Hessian of the objective $f(d_1)$ (Eq. 41), $W_2^{-1} Y_2$ the Lagrange multipliers for the linear minimum constaint $b_1 \leq d_1$ and $W_3^{-1} Y_3$ with the linear maximum constraint $d_1 \leq b_2$. In a naive implementation this would require three expensive matrix-matrix products, but a straightforward factorisation results in:

$$\begin{array}{l} A_1^T D_1 A_1 \\ D_1 = v_1 c_{12} E_1 + W_2^{-1} Y_2 + W_3^{-1} Y_3 \end{array} \tag{48}$$

which represents the first term of the condensed form (38).

The second term of the condensed form, the symmetric matrix Q consists of the matrices B_i (43), vector outer-products, and a small matrix-matrix product resulting from the gradients of the nonlinear cost-functions. For computational efficiency, the k outer-products of the gradients of the nonlinear cost-functions are concatenated in a matrix, and computed by a matrix-matrix product. Third row of (45) becomes:

$$[\nabla g_1\ \nabla g_2\ \cdots\ \nabla g_k]^T \begin{bmatrix} w_1^{-1} y_1 & & & \\ & w_2^{-1} y_2 & & \\ & & \ddots & \\ & & & w_k^{-1} y_k \end{bmatrix} [\nabla g_1\ \nabla g_2\ \cdots\ \nabla g_k]. \quad (49)$$

Usually, the number of nonlinear cost-functions is in the order of $10 - 20$.

4.3 Properties

The condensed form:

$$N = A^T DA + Q + T \quad (50)$$

with A being the data matrices of the problem, expressed as:

$$A^T = [A_1\ A_2\ A_3 \ldots]^T \quad (51)$$

has the following properties:

- A has unique rows if the problem was initialised correctly
- D is positive semidefinite for convex problems
- D may contain negative elements for nonconvex problems
- T is positive semidefinite
- Q is symmetric
- N is symmetric
- Computational complexity depends on the number of *unique structures* (expressed in A_i and its corresponding dose d_i), not on the number of cost-functions or constraints

For convex problems, N is positive definite. For nonconvex problems, N may still be positive definite due to the other elements of D, Q and T.

The quality of the matrix A depends on the correct configuration of the radiation therapy problem. If no overlapping voxels are selected during the discretisation phase of the patient (see Fig. 2), which is easily avoidable, it is highly likely that A is full rank. However, 2 neighbouring voxels may be numerically identical due to limited representation of machine precision. Careful preprocessing to ensure full rank A's did not reveal any improvements above compared to no preprocessing.

5 Computational advances

In this section, computational approaches to efficiently compute the matrix-matrix product of (38) are presented, including permutation, tiling and multiple precision arithmetic.

Table 1 Matrix properties of a typical radiation therapy head-and-neck case (subset of Head-and-Neck 01)

Structure	Rows	#Nonzero	Sparsity (%)
Tumour	5096	9,142,738	18.0
Spinal cord	3529	4,682,162	13.3
Brainstem	3757	1,843,820	4.9
Parotid (R)	3976	6,038,167	15.2
Parotid (L)	3975	5,602,446	14.1
SMG (R)	1406	2,640,389	18.8
SMG (L)	1769	3,252,371	18.4
Oral cavity	5298	8,259,205	15.6
Larynx	5263	10,353,470	19.7
MCS	1	4448	44.6
MCM	1	4514	45.2
MCI	1	5208	52.2
MCP	1	3465	34.7
Oesophagus	1	3316	33.2
Patient	10,917	12,627,463	11.6
PTV shell 0 mm	4908	8,173,765	16.7
PTV shell 5 mm	4954	7,719,957	15.6
PTV shell 15 mm	4823	6,147,460	12.8
PTV shell 30 mm	4805	4,298,559	9.0
PTV shell 40 mm	4726	2,795,977	5.9
External ring 20 mm	5346	1,586,997	3.0

The number of columns in each matrix is 9977 elements

5.1 Symmetric matrix-matrix product

The data matrices A_i in (51) are stored separately and are not combined into a single matrix. The sparsity of the matrices differ greatly. Table 1 shows matrix properties of a typical problem. Structures with only a single row are only evaluated by the *mean* cost-function. The number of voxels is determined by the size and type of the structure. This table shows that the matrices are not particularly dense, but also not sufficiently sparse to be handled efficiently by sparse algorithms. Figure 4 shows the sparsity pattern for the tumour and oral cavity matrices. For the computation of $A_i^T D_i A_i$ we use three separate procedures, described in the following sections.

5.1.1 Dense and nonnegative diagonal

For computing

$$N = N + A_i^T D_i A_i \tag{52}$$

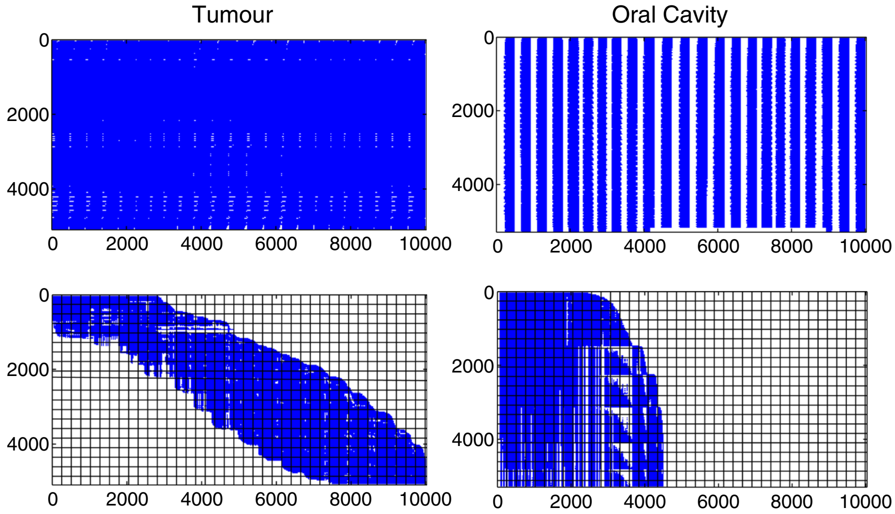

Fig. 4 Sparsity patterns for typical pencil-beam matrices A_i in radiation therapy. *Top panels* show the original data, *bottom panels* show the permuted and tiled version. While the matrix for the tumour (*top-left*) seems almost completely dense, only 18% of the elements are used

for $D_i > 0$ (all elements of the diagonal are nonnegative), we compute a matrix B as:

$$B = D_i^{\frac{1}{2}} A_i \tag{53}$$

which is a scalar product of the square-root of each element of d_i with the respective column of the matrix A_i. As a column-wise operation is linear in memory access, this operation is as good as free.

The product is then computed by a symmetric matrix-matrix product, using the *symm* BLAS function.

$$N = N + B^T B \tag{54}$$

5.1.2 Dense and general diagonal

If the diagonal D_i has negative elements, the square-root cannot be used. Instead, we pre-multiply the matrix by the diagonal and perform a general matrix-matrix operation, using the *gemm* BLAS function:

$$\begin{aligned} B &= D_i A_i \\ N &= N + B^T A. \end{aligned} \tag{55}$$

5.1.3 Sparse

For sparse A_i we use a custom implementation based on the sparse matrix multiplication algorithm as described by [30]. We know that the resulting (dual-normal) matrix will be a dense matrix, and can therefore skip the symbolic part of sparse algebra. For this algorithm, the left matrix has to be pre-transposed, and is directly multiplied with the diagonal matrix D_i [14]. The formal operation is identical to (55). We also exploit the fact that the result is symmetric: the transposition algorithm ensures an ordered sparse representation, allowing the multiplication algorithm to skip operations for the lower triangle of N.

5.2 Permutation and tiling

Table 1 and Fig. 4 suggest that neither the dense nor sparse format is particularly efficient for the matrix-matrix product. The tipping point for computational efficiency for the matrix-matrix product lies around a sparsity level of 15% when using the method described in the previous section (depending on matrix dimensions). However, we can exploit the problem structure by permuting the rows and columns of each matrix. This results in constructing a *tiled* matrix format with dense, sparse and all-zero tiles.

There are many partitioning algorithms available. We tested several and found that *Gibbs–Poole–Stockmeyer* [23,59] from the *Scotch* [49] library resulted in the most efficient partition for the matrix-matrix product, while keeping the partitioning time to a minimum. As this algorithm only works on symmetric matrices (bipartite graphs), we converted the rectangular matrices to symmetric square ones using:

$$\begin{bmatrix} 0 & A_i^T \\ A_i & 0 \end{bmatrix} \tag{56}$$

before partitioning.

The permuted and tiled sparsity pattern is also shown in Fig. 4. For each tile, it is determined whether it is best handled dense or sparse. All-empty tiles are tagged as "empty", and are not stored.

The matrix multiplication now becomes a tiled matrix multiplication, either between dense-dense tiles, sparse-sparse tiles, dense-sparse tiles, or any with an all-zero tile. The tiled multiplications are then distributed over the available CPU-cores and executed single-threadedly. For the dense-dense algebra we use the *gemm* function from the BLAS library, for the other combinations algorithms from [14,30,50] are used.

Heuristics determine whether or not tiling and/or permutation is an efficient investment. Matrices over 80% dense are only tiled, not permuted. Tiling allows matrices to be efficiently processed and multithreaded using the tiled multiplication algorithm, which is efficient even for dense matrices [38]. Dense matrices with few rows, or large, sparse matrices are not tiled at all, as their size results in better performance when not multithreaded.

5.3 Multiple precision arithmetic

For the data, single precision is sufficient to model the radiation therapy problem, but double precision is required for successful convergence of the interior-point algorithm. The matrix-matrix product of the condensed form (38) is the most time-consuming step in the algorithm, so it is effective to perform this operation as much as possible in single precision. Due to the relative few rows in computing Q (49), the time required to compute Q and diagonal T is negligible.

Because the background of the problem is known, we can make some (but not limiting) assumptions on the data matrices: (1) all elements and all matrices are of similar order of magnitude (generally $\mathcal{O}(10^{-3})$), and (2) all elements are nonnegative (except occasionally a column of -1 resulting from minimax reformulations). As a consequence, we can focus solely on the magnitude of the elements of the diagonal D, without taking into account the (possibly) different scalings of the rows of A.

The magnitude of the elements in the diagonal D vary between very large ($>10^8$) and very small ($<10^{-10}$), especially in the final steps. A good indicator is the ratio between the largest and the smallest element of D. If this ratio is $<10^{-5}$ (single precision accuracy is around $1.19 \cdot 10^{-7}$), then the matrix multiplication can be performed in single precision, otherwise double precision is required for a sufficiently accurate multiplication.

Our strategy is to keep N in double precision, and determine per matrix which precision is possible. This is even propagated further down to the tiled matrix implementation (Sect. 5.2), where the decision for the precision is made *per tile* rather than per matrix. In this event, the number of diagonal elements to compare is greatly reduced, increasing the probability of more single precision operations. It is possible that when using the tiled approach, 80% of the multiplications can be performed using single precision.

One other advantage of the canonical form is that D is often a combination of $W^{-1}Y$ and E (41) if both linear and nonlinear constraints are used. The addition of these result in diminishment of extremely small elements, increasing the single precision ratio.

6 Results

6.1 Implementation and hardware

A concise overview of the algorithm is given in Algorithm 1. The solver is implemented in C++ and compiled using the GNU C++ Compiler (version 4.6.3). For basic linear algebra operations we used the Intel Math Kernel Library (version 11.3) wherever possible. For the tiled matrix-vector and matrix-matrix product, as well as sparse matrix algebra, custom functions are implemented in C++ [14,30,50]. Special attention is given to multi-threading, which is achieved using OpenMP.

The hardware used is a dual CPU system, consisting of 2 octocore Intel Xeon E5-2690 CPUs, running at 2.90 GHz (and up to 3.8 GHz singlethreaded) and has 20 MiB cache. The maximum memory bandwidth is 51.2 GiB/s. Unless stated otherwise, all results are achieved using all 16 cores, with hyperthreading disabled.

Algorithm 1 Concise overview of the algorithm. Here, $\mathbf{x} = (z, w, x, y)$.

```
load data
permute/tile matrices, store in single/double precision (Sect. 5)
initialise $\mathbf{x}_0$ (Sect. 3.1)
rewrite minimax problems to generic form (3)
for interior-point iterations $k = 1, \ldots$ do
    compute $N$ according to (37) using arithmetic from Sect. 5
    decompose $N$ using Cholesky decomposition
    if method == Default then
        find $\Delta x_k$ by solving (20)
        determine maximum step $\alpha_k$ according to Sect. 3.3
    else
        find $\Delta\hat{x}_k$ for predictor step (Sect. 2.3)
        determine maximum steplength $\alpha_k$ according to Sect. 3.3
        find $\Delta\check{x}_k$ for corrector step (Sect. 2.3)
        set $\Delta\hat{x}_k \leftarrow \Delta\hat{x}_k + \Delta\check{x}_k$
        determine maximum steplength $\alpha_k$ according to Sect. 3.3
        if step rejected then
            compute $\Delta x_k$ by doing a Default step (above)
        else
            set $\Delta x_k \leftarrow \Delta\hat{x}_k$
            if method == Gondzio then
                while step is acceptable do
                    aim to take step $\check{x}_k = x_{k-1} + 1.1\alpha_k \Delta x_k$
                    find $\Delta\check{x}_k$ for Gondzio update step (Sect. 2.3)
                    set $\Delta\hat{x}_k \leftarrow \Delta x_k + \Delta\check{x}_k$
                    determine maximum steplength $\hat{\alpha}_k$ according to Sect. 3.3
                    if step rejected OR $\hat{\alpha}_k < 1.07\alpha_k$ then
                        stop updates and use $\Delta x_k$
                    else
                        accept step and set $\Delta x_k \leftarrow \Delta\hat{x}_k$
                    end if
                end while
                determine maximum steplength $\alpha_k$ according to Sect. 3.3
            end if
        end if
    end if
    take step $x_k = x_{k-1} + \alpha_k \Delta x_k$
    compute convergence criteria and terminate if norm of (8) $< 10^{-4}$
end for
```

6.2 Problem data

We used the TROTS dataset [10,11] to demonstrate the performance of the solver. This dataset consists of 120 patients (47 GiB) for different treatment sites and problem definitions, representing real clinical problems.

To focus on the performance and characteristics of the solver, the multi-criteria aspect of the radiation therapy problem has been reduced to a single-criteria formulation. We used our lexicographic multi-criteria optimisation approach for automated treatment planning, which is a form of sequential ϵ-constraint programming [12,15]. The final constrained problem is then transformed to a weighted-sum by setting the Lagrange multipliers of the constrained objectives as their corresponding weights [12].

Table 2 Computational complexity, average per group. Shown are the number of initial decision-variables/constraints and the numbers after reconfiguring the problem, such as handling minimax constraints

	Initial		Final				
	n	m	n	m	Quadratic	Convex	Not convex
Prostate CK	2937	51,321	2943	97,439	24–25	3	0
Prostate VMAT	2697	67,959	2701	94,602	23	3–4	0
Head-and-Neck	8302	75,508	8309	97,917	23	3	0
Head-and-Neck Alt	8302	75,508	8309	97,917	23	3	0
Protons	1803	108,277	1823	408,294	0	0	0
Liver	1615	91,884	1621	119,680	15	0	3

The latter relates to the "computational burden", i.e. final size of the data matrices. The number of constraints *m* are the total constraints, including nonlinear. The number of nonlinear constraints are given per type

Our rationale for doing so is to (1) create problems of realistic complexity, and (2) the result is a realistic treatment plan.

For details regarding the source of the data, treatment protocols, usage and visualisation of the TROTS data we refer to [10,11]. Here, we only provide details relevant to the optimisation. Problem complexities are summarised in Table 2.

The *Prostate CK*, *Prostate VMAT*, *Head-and-Neck* and *Head-and-Neck Alt* patients are all regular convex problems for intensity-modulated photon therapy. The *Head-and-Neck Alt* patients are identical to the normal group, but are equipped with a more accurate dose model, resulting in denser matrices. This allows investigating the effect of the denseness of the data on runtime, as the problems are identical. The *Liver* cases contain 3 nonconvex dose-volume constraints (Sect. 3.2). The *Protons* patients are patients who are treated with intensity-modulated proton therapy. This modality has different dosimetric properties, resulting in different types of dose matrices. In addition, these problem configurations are fully linear.

6.3 Performance

In the following, all runtimes are reported in *seconds*, *excluding* loading the data from disk, but *including* initialisation, permutation, etc., and achieved by using the *Mehrotra* method, unless stated otherwise. All problems were optimised using the same termination criteria: optimality conditions (8) $<10^{-4}$ and a maximum number of iterations of 300. Steplength control in the form of the Markov filter (Sect. 3.3.1) was only enabled for the nonconvex problems, otherwise, ratio control was used.

All tables contain summarised results, detailed results can be found in the supplementary material. The reported *Average per Iteration* is computed by averaging to the times per iteration for *each* problem, rather than based on the accumulated results. All problems converged successfully, including the nonconvex Liver problems. Only the *Head and Neck 12* problem in the 8-threaded naive run did not converge and terminated after 128 iterations.

Table 3 Effect of second-order and higher-order interior-point methods by number of iterations and runtime

	Iterations			Time (s)		
	Default	Mehrotra	Gondzio	Default	Mehrotra	Gondzio
Prostate CK	2863	1284	1319	3501.9	1732.1	2031.3
Prostate VMAT	3007	1310	1458	4685.3	2221.3	2758.2
Head-and-Neck	1198	615	610	5660.3	3110.3	3226.0
Head-and-Neck Alt	1202	608	596	12,895.0	6756.1	6937.7
Protons	2151	1401	1063	1738.9	1321.7	1274.0
Liver	989	435	392	685.0	369.8	412.6
Total	11,410	5653	5438	29,166.3	15,511.2	16,639.8
Average per iteration				2.9	3.0	3.3

In Table 3 we report the number of iterations and runtimes for the Default (Sect. 2.1), the Mehrotra, and the novel Gondzio methods (Sect. 2.3). Compared to the Default method, Mehrotra's method reduces the number of iterations by 50%, and Gondzio's method slightly more to 52%. Gondzio used 4370 extra updates on top of the standard Mehrotra corrector update. While the runtime is reduced accordingly between Default and Mehrotra, the runtime for Gondzio's method is slightly higher. This is because the additional higher-order corrections require additional time to compute. Only for the Protons problems there is a strong and structural reduction.

The effect of the computational advances (Sect. 5) is demonstrated in Table 4. Runtimes are reported for different combinations of permutation/tiling and/or multiple precision arithmetic. The effect of multiple precision arithmetic is negligible without permutation/tiling. This is because the single precision heuristic checks the diagonal elements per multiplication. Smaller matrices have a higher probability of having all elements in the same order of magnitude, while a single outlier can ruin this approach. For large (untiled) matrices, it is likely that all multiplications are performed in double precision. Permutation/tiling results in a reduction in time of 37% (per iteration). Using in addition multiple precision arithmetic, the runtimes are further reduced to 54%. We also observe a drop in number of iterations: this results in a total runtime reduction of 55%.

An additional advantage of the optimised approach is that it improves scalability, see Table 5. Here we compare singlethreaded and multithreaded (16 cores, divided over 2 CPUs) performance for the naive (nonoptimised) and the optimised approach. Where the naive approach is only 7.7 times faster when using 16 threads compared to singlethreaded, the optimised approach scales with a factor 9.8.

The effects of our initialisation and ratio control strategies is demonstrated in Table 6. Initialisation decreases the number of iterations by 9% compared to the simpler approach. When initialisation is used, the ratio control only has a negligible advantage, but for ill-initialised problems, the use of ratio control results in a reduction of 11%.

Figure 5 and Table 7 show the fraction of the runtime spent in each module. The time to permute, tile matrices and initialise the problem is $5\% - 10\%$ of the runtime.

Table 4 Iterations and runtime in seconds for different computational approaches (Sect. 5)

	No Tile No Perm No MPA	Tile No Perm No MPA	No Tile No Perm MPA	Tile Perm No MPA	Tile Perm MPA
Iterations					
Prostate CK	1285	1291	1285	1291	1284
Prostate VMAT	1365	1313	1365	1313	1310
Head-and-Neck	661	613	661	613	615
Head-and-Neck Alt	619	611	619	611	608
Protons	1565	1416	1565	1411	1401
Liver	430	437	430	437	435
Total	5925	5681	5925	5676	5653
Time (s)					
Prostate CK	3060.3	2655.0	3130.8	2468.4	1732.1
Prostate VMAT	4956.0	3528.9	5017.9	3184.5	2221.3
Head-and-Neck	9981.9	8805.3	10, 382.6	3831.9	3110.3
Head-and-Neck Alt	12,974.9	14,912.8	13,385.9	9755.4	6756.1
Protons	3125.1	1381.0	3117.1	1467.4	1321.7
Liver	556.4	479.6	571.0	463.1	369.8
Total	34,654.7	31,762.6	35,605.3	21,170.7	15,511.2
Average per iteration	6.5	6.3	6.7	4.1	3.0

Each column indicates whether permutation/tiling and/or multiple precision arithmetic (MPA) is used. In the second column, the matrices are only *tiled*, not permuted. Most right column is equal to Mehrotra in Table 3

Clearly, the construction of the dual-normal matrix N (38) consumes $>70\%$ of the time, of which only a fraction is used for the function evaluations and derivatives, and construction of the diagonals D, T and matrix Q (48-49). The time required for solving the system (20) (the backsolves) is between $5\% - 15\%$. This is almost entirely due to the two matrix-vector computations with $\nabla g(x)$: one transposed to construct the right-hand-side (19), and one to compute Δy once the system is solved (17). An interesting observation can be made when comparing the Head-and-Neck and Head-and-Neck Alt problems. As both problems are equal in size n (decision-variables), the time required for the Cholesky decomposition is similar for both approaches, but the time required for the other algebra strongly increases due to the denser matrices of the alternative problem. By comparing the overall runtime from Table 3 with accumulated times in Table 7, we learn that the total overhead is 414.1 s (non-measured time, which is spent on function/library calls, I/O handling, etc.).

7 Discussion and conclusions

This paper described an interior-point implementation specifically designed and tuned for a single application. This allows exploiting the specific problem structure, hard-

Table 5 Scalability of the solver to 16 threads, comparing the standard algebra (no permutation/tiling, no multiple precision arithmetic) with optimised algebra

	No Perm/No MPA (naive)		Perm/MPA	
	1 thread	16 threads	1 thread	16 threads
Iterations				
Prostate CK	1290	1285	1284	1284
Prostate VMAT	1411	1365	1314	1310
Head-and-Neck	905	661	612	615
Head-and-Neck Alt	637	619	610	608
Protons	1530	1565	1409	1401
Liver	434	430	435	435
Total	6207	5925	5664	5653
Time (s)				
Prostate CK	17,961.2	3060.3	14,921.7	1732.1
Prostate VMAT	32,406.9	4956.0	19,971.5	2221.3
Head-and-Neck	101,727.2	9981.9	30,958.6	3110.3
Head-and-Neck Alt	122,267.7	12,974.9	73, 019.5	6756.1
Protons	7712.0	3125.1	5782.0	1321.7
Liver	2470.8	556.4	2345.7	369.8
Total	28,4545.7	34,654.7	146,998.9	15,511.2
Average per iteration	50.3	6.5	29.4	3.0
Speed-up factor		7.7		9.8

Table 6 Effect on the number of iterations when (not) using the initialisation strategy and/or ratio control

Iterations	Initialise ratio	No initialise ratio	Initialise no ratio	No initialise no ratio
Prostate CK	1284	1385	1272	1404
Prostate VMAT	1310	1597	1335	2152
Head-and-Neck	615	716	637	857
Head-and-Neck Alt	608	678	619	762
Protons	1401	1437	1401	1437
Liver	435	410	435	410
Total	5653	6223	5699	7022

coding cost-functions, optimising linear algebra and initialisation. Additionally, a typical nonconvex cost-function is supported.

The use of the Mehrotra higher-order method resulted in a higher reduction of iterations than expected: 50%, whereas [42] reported 35% and [57] only 15%. The added value of higher-order methods therefore seems problem-dependent.

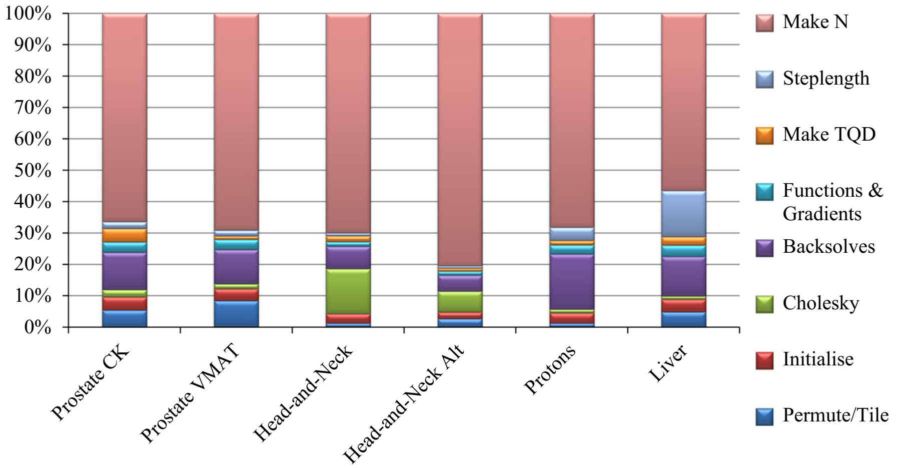

Fig. 5 Time spent in each module (see also Table 7). The ordering of the stacked *bars* is the same as in the legend

Gondzio's method is able to further reduce the number of iterations for linear problems only (Protons), or problems with few nonlinear cost-functions (Liver). For general convex problems, Gondzio was not advantageous, especially not when comparing runtime. Tuning of the parameters (see Supplemantary material) demonstrated a sensitive behaviour in the number of iterations for the nonlinear problems, while most parameter settings worked relatively stable for the linear problems. The Protons problems show a very slow convergence for all methods. When using Gondzio, we see more higher-order updates in the later iterations. It remains to be investigated if Gondzio will always be the preferred choice for full linear problems. We did not show results for the original Gondzio approach without the Mehrotra corrector step, because we were unable to find acceptable parameters. With our best setting, the Protons problems still required between 150–250 iterations.

Interestingly, the matrix permutation, tiling and multiple precision arithmetic reduces the number of iterations. Apparently, these steps improve the numerical stability. For cases which have difficulty converging, the steps were smooth until the problem started converging (optimality condition (8) around $\mathcal{O}(10^{-2})$). The Cholesky decomposition started failing because the dual-normal matrix N was not positive definite. For the single case that did not converge, the starting value of $\lambda = 10^{-6}$ added to the diagonal of N seems to be ill-chosen: either larger 10^{-4} or smaller 10^{-8} worked. However, since the problem is convex, the problem should not become indefinite. The problem with indefiniteness is related to not tiling the matrices, and does not seem to be related to the multiple precision arithmetic (Table 4). We are unsure exactly why, but one could argue that tiled matrix-matrix multiplication is able to better handle the badly scaled diagonal D in $A^T DA$ (38). In the final iterations, these elements can range from 10^{-80} to 10^{10}, although setting the smallest elements to 0 did not solve this issue. Tiling of the matrices is not new: this is a standard procedure in every matrix-matrix multiplication library, and used to fit each tile in the CPU's cache prior

Table 7 Fraction spent in each module, where the last column shows the time spent in making N when no permutation/tiling is used

	Permute/ Tile	Initialisation	Cholesky	Backsolves
Prostate CK	89.1	68.7	38.9	197.7
Prostate VMAT	179.4	81.7	33.6	232.1
Head-and-Neck	37.0	91.8	430.1	226.0
Head-and-Neck Alt	179.7	138.8	442.3	341.0
Protons	16.3	40.4	14.3	215.0
Liver	16.3	13.3	3.4	50.7
Total	517.6	434.8	962.6	1262.5

	Functions & Gradients	Make TQD	Steplength	Make N	No Perm Make N
Prostate CK	54.8	72.1	36.1	1110.8	2309.1
Prostate VMAT	68.5	24.0	39.2	1489.2	4007.1
Head-and-Neck	46.7	56.2	27.8	2122.9	8376.4
Head-and-Neck Alt	85.4	58.0	46.4	5364.9	11248.4
Protons	35.2	17.6	53.0	834.3	2261.5
Liver	12.3	9.6	57.1	197.4	364.3
Total	302.8	237.6	259.6	11119.5	28566.8

to multiplication to avoid latency in memory look-ups [5,17]. Similar to [16,38], we tile the matrices once.

The prolonged time required for starting the problem due to permutation, tiling and initialisation of x_0 is fully compensated by the reduction in runtime and required iterations. This demonstrates that proper initialisation is extremely important for the performance of interior-point methods. The application of a simple ratio steplength control mechanism is capable of reducing the impact of an unfortunate chosen initial point, which may either be x_0 or one of the other variables. In our previous implementation, we had set the minimum of w_0 and y_0 to 10 instead of 5, which turned out to have a high impact on the number of iterations (6369 in total compared to 5653 now). During this implementation, the ratio control was developed, which resulted in reducing the number of iterations to 6069. Further research may be beneficial, for example using warmstarting techniques based on the initialised variables [22,28,47,73].

The extensions for nonconvex interior-point optimisation (Sect. 3.3) were only recently introduced in our solver, and with success: all nonconvex problems converged without problems. Using the parameter $p = 5$ in the approximation of the dose-volume constraint (30) resulted in a correspondence of within 1%-point of the exact dose-volume function (29) for most cases. It remains to be investigated how many dose-volume constraints can be included and how accurate we can make the approximation by using higher values for the p parameter, and if further extensions to the algorithm are necessary.

We also compiled the C++ source code using the Intel C++ Compiler, but this did not result in improved performace. The majority of the runtime is spent on linear algebra, for which we utilise highly optimised libraries. The tiled matrix-matrix multiplication is parallellised on the distribution of tiles (each thread computes a tile-by-tile multiplication). Consequently, each multiplication is performed singlethreadedly, significantly simplifying the implementation of our custom (sparse) linear algebra routines.

The approach presented in this paper is the result of over 9 years development and tuning of the interior-point method and algebra. The solver runs on a cluster with currently over 350 computing cores, and is estimated to have solved (sometimes without success) over 7 million problems (estimated based on scheduler statistics). This high number is obtained because the more complex radiation therapy problems require solving thousands of problems for a single patient. Aside from computing treatment plans for patients currently under treatment, our Erasmus-iCycle framework is an ideal research environment, where novel treatment protocols are developed and evaluated [55,56,58,61,66–69]. In this case, many configurations with different parameters are being tested on (large) groups of previously treated patient data, resulting in this high number of solved mathematical problems. During these years, searches for higher performance were ever ongoing, and included investigating alternative classes of optimisation methods.

The most straightforward ones are quasi-Newton methods, where the Hessian is approximated by a low-rank matrix [46]. However, quasi-Newton methods only apply to nonlinear functions, thus the many (linear) pointwise minimum- or maximum-constraints need to be replaced by a surrogate function, such as e.g. the log-sum-exp. This has a practical disadvantage: if a maximum constraint of 50 is desired, achieving this by the surrogate maximum may result in a real maximum of e.g. 53 (which is unacceptable) or 47. In the latter, the remaining freedom within the constraints is not used, thereby limiting reduction of dose (sparing) elsewhere. For the same reason of remodelling the problem to few (nonlinear) constraints, sequential quadratic programming (SQP) is also less interesting. Computationally, the disadvantage of quasi-Newton methods is the increased number of iterations required, whereas the time reduction per iteration is already small due to the pre-combination of different cost-functions for the same matrix (Sect. 4) and the permutation/tiling/multiple precision arithmetic (Sect. 5). In [29] it is argued and demonstrated that also for larger problems in radiation therapy, full-Newton methods are preferred above quasi-Newton and SQP methods with respect to quality and performance.

Matrix-free methods avoid building the dual-normal matrix N, and solve system (20) by a series of matrix-vector multiplications, where our matrix is kept in condensed form (38) [26,60]. The main problem with this approach was preconditioning the iterative method, which has to be done in each iteration. Without preconditioning the system often did not converge. A simple preconditioning method that worked was based on taking the 1000–3000 largest elements of D (38), compute N and factorise it to obtain the preconditioner. The downside of this approach is that a Cholesky decomposition is still required, and that multiplying 1000–3000 non-sequential rows of A is much slower than 3000 sequential rows. Another disadvantage is that each backsolve is expensive, so the reduction in iterations acquired by higher-order methods result in longer runtimes. Our best implementation hardly rivalled the full matrix-matrix multiplication approach, and did also not show stable convergence for different problems.

Column generation (CG) is another approach for solving large problems by dimensionality reduction [52], and proposed to be used in radiation therapy to optimise directly on treatment device parameters to obtain efficient and short treatment times by [43]. Unfortunately, column generation does not apply well to our class of constrained problems. In column generation, the problem is initially reduced to a single decision-variable, resulting in an infeasible constrained problem: a plan respecting both minimum and maximum dose constraints for the tumour is impossible. Gondzio *et. al.* [27] proposes to solve such problems up to some predefined duality gap μ, and reducing the *dual*-infeasibility as far as possible. This is to obtain the most relevant Lagrange multipliers required for selecting the subset of the decision-variables to add for the next iteration. In the next iteration, a smaller estimate for μ is made. We have implemented this in an automated adaptive fashion where the problem is optimised up to some pre-set μ, and once attained μ is iteratively reduced after which the interior-point method continues from this point on. This has the advantage of obtaining the smallest possible μ without having to restart the optimisation multiple times. Unfortunately, achieving feasibility and optimality using the CG approach requires a significant number of CG iterations, where much time is spent in recreating the data matrices for the reduced problems. Consequently, the aim of attaining an efficient treatment plan is lost when too many columns are generated [43].

The TROTS dataset used in this paper [10,11] was specifically constructed to evaluate the performance for medium- to large-scale problems consisting of dense data. Other sets frequently used in benchmarking solvers, such as the Netlib, CUTEst, Hock and Schittkowski, and Vanderbei's set [20,35,44,63], often provide problems which are challenging to solve, but are in general too small or too sparse to be used to test performance and scalability of denser problems [7]. By making this dataset available we offer the field of operational research data for extensive performance testing, but also to test stability (stable number of iterations for similar problems) of the solvers.

Acknowledgements The authors want to thank Jacek Gondzio (University of Edinburgh, Scotland) and Bram Gorissen (VU University Amsterdam, The Netherlands) for their insightful and interesting discussions on interior-point methods and other solver approaches, and Markus Alber (Aarhus University Hospital, Denmark) for sharing his original ideas on the dose-volume cost-function. We also thank the anonymous reviewers for their time and many suggestions for improvement of the algorithm and the manuscript. From the Erasmus MC we thank András Zolnay for his endless support in writing a rigid C++ implementation, and Rens van Haveren for refining the analytical canonical form for the derivatives. We also thank Steven van de Water, Linda Rossi and Peter Voet for sharing their research data.

Compliance with ethical standards

Conflict of inerest This work was in part made possible by funding from Accuray Inc, Sunnyvale, USA and Elekta AB, Stockholm, Sweden.

References

1. Alber, M., Meedt, G., Nüsslin, F.: On the degeneracy of the IMRT optimization problem. Med. Phys. **29**, 2584–2589 (2002). doi:10.1118/1.1500402
2. Alber, M., Reemtsen, R.: Intensity modulated radiotherapy treatment planning by use of a barrier-penalty multiplier method. Optim. Methods Softw. **22**, 391–411 (2007). doi:10.1080/10556780600604940
3. Aleman, D.M., Glaser, D., Romeijn, H.E., Dempsey, J.F.: Interior point algorithms: guaranteed optimality for fluence map optimization in IMRT. Phys. Med. Biol. **55**, 5467–5482 (2010). doi:10.1088/0031-9155/55/18/013
4. Aleman, D.M., Romeijn, H.E., Dempsey, J.F.: A response surface approach to beam orientation optimization in intensity modulated radiation therapy treatment planning. INFORMS J. Comput. **21**, 62–76 (2009). doi:10.1287/ijoc.1080.0279
5. ATLAS: Building the general matrix multiply from the L1 cache-contained multiply. http://www.netlib.org/atlas/developer/atlas_contrib/node11.html
6. Benson, H., Shanno, D., Vanderbei, R.: Interior-point methods for nonconvex nonlinear programming: filter methods and merit functions. Comput. Optim. Appl. **23**, 257–272 (2002). doi:10.1023/A:1020533003783
7. Benson, H., Shanno, D., Vanderbei, R.: A comparative study of large-scale nonlinear optimization algorithms. In: High Performance Algorithms and Software for Nonlinear Optimization, pp. 95–127. Springer, New York (2003). doi:10.1007/978-1-4613-0241-4_5
8. Bokrantz, R.: Multicriteria optimization for managing tradeoffs in radiation therapy treatment planning. Ph.D. thesis, KTH Royal Institute of Technology, Sweden (2013)
9. Breedveld, S., Craft, D., van Haveren, R., Heijmen, B.: Multi-criteria optimisation and decision-making in radiotherapy (submitted) (2017)
10. Breedveld, S., Heijmen, B.: TROTS - The Radiotherapy Optimisation Test Set. http://www.erasmusmc.nl/radiotherapytrots/ (2016)
11. Breedveld, S., Heijmen, B.: Data for TROTS—the radiotherapy optimisation test set. Data Br. **12**, 143–149 (2017). doi:10.1016/j.dib.2017.03.037
12. Breedveld, S., Storchi, P., Heijmen, B.: The equivalence of multi-criteria methods for radiotherapy plan optimization. Phys. Med. Biol. **54**, 7199–7209 (2009). doi:10.1088/0031-9155/54/23/011

13. Breedveld, S., Storchi, P., Keijzer, M., Heemink, A.W., Heijmen, B.: A novel approach to multi-criteria inverse planning for IMRT. Phys. Med. Biol. **52**, 6339–6353 (2007). doi:10.1088/0031-9155/52/20/016
14. Breedveld, S., Storchi, P., Keijzer, M., Heijmen, B.: Fast, multiple optimizations of quadratic dose objective functions in IMRT. Phys. Med. Biol. **51**, 3569–3579 (2006). doi:10.1088/0031-9155/51/14/019
15. Breedveld, S., Storchi, P., Voet, P., Heijmen, B.: iCycle: integrated, multicriterial beam angle, and profile optimization for generation of coplanar and noncoplanar IMRT plans. Med. Phys. **39**, 951–963 (2012). doi:10.1118/1.3676689
16. Buttari, A., Langou, J., Kurzak, J., Dongarra, J.: A class of parallel tiled linear algebra algorithms for multicore architectures. Parallel Comput. **35**, 38–53 (2009). doi:10.1016/j.parco.2008.10.002
17. Catalyürek, Ü., Aykanat, C.: Hypergraph-partitioning-based decomposition for parallel sparse-matrix vector multiplication. IEEE Trans. Parallel Distrib. **10**, 673–693 (1999). doi:10.1109/71.780863
18. Chen, W., Craft, D., Madden, T., Zhang, K., Kooy, H., Herman, G.: A fast optimization algorithm for multicriteria intensity modulated proton therapy planning. Med. Phys. **37**, 4938–4945 (2010). doi:10.1118/1.3481566
19. Craft, D.L., Halabi, T.F., Shih, H.A., Bortfeld, T.R.: Approximating convex Pareto surfaces in multi-objective radiotherapy planning. Med. Phys. **33**, 3399–3407 (2006). doi:10.1118/1.2335486
20. CUTEst: CUTEst—a constrained and unconstrained testing environment on steroids. http://ccpforge.cse.rl.ac.uk/gf/project/cutest/wiki/
21. Dong, P., Lee, P., Ruan, D., Long, T., Romeijn, E., Yang, Y., Low, D., Kupelian, P., Sheng, K.: 4Pi non-coplanar liver SBRT: a novel delivery technique. Int. J. Radiat. Oncol. Biol. Phys. **85**, 1360–1366 (2012). doi:10.1016/j.ijrobp.2012.09.028
22. Gertz, M., Nocedal, J., Sartenaer, A.: A starting-point strategy for nonlinear interior methods. Appl. Math. Lett. **17**, 945–952 (2004). doi:10.1016/j.aml.2003.09.005
23. Gibbs, N., Poole, W., Stockmeyer, P.: A comparison of several bandwidth and profile reduction algorithms. ACM Trans. Math. Softw. **2**(4), 322–330 (1976). doi:10.1145/355705.355707
24. Gondzio, J.: Implementing Cholesky factorization for interior point methods of linear programming. Optimization **27**, 121–140 (1993). doi:10.1080/02331939308843876
25. Gondzio, J.: Multiple centrality corrections in a primal-dual method for linear programming. Comput. Optim. Appl. **6**, 137–157 (1996). doi:10.1007/BF00249643
26. Gondzio, J.: Matrix-free interior-point method. Comput. Optim. Appl. **51**, 457–480 (2012). doi:10.1007/s10589-010-9361-3
27. Gondzio, J., González-Brevis, P., Munari, P.: New developments in the primal-dual column generation technique. Eur. J. Oper. Res. **224**, 41–51 (2013). doi:10.1016/j.ejor.2012.07.024
28. Gondzio, J., Grothey, A.: A new unblocking technique to warmstart interior point methods based on sensitivity analysis. SIAM J. Optim. **19**, 1184–1210 (2008). doi:10.1137/060678129
29. Gorissen, B.L.: On Newton based algorithms for inverse planning of intensity-modulated proton therapy (submitted) (2016)
30. Gustavson, F.G.: Two fast algorithms for sparse matrices: multiplication and permuted transposition. ACM Trans. Math. Softw. **4**, 250–269 (1978). doi:10.1145/355791.355796
31. Halabi, T., Craft, D., Bortfeld, T.: Dose-volume objectives in multi-criteria optimization. Phys. Med. Biol. **51**, 3809–3818 (2006). doi:10.1088/0031-9155/51/15/014
32. Van Haveren, R., Breedveld, S.: A canonical and computationally efficient form for the gradient and Hessian of simple composite functions in nonlinear programming (submitted) (2017)
33. Van Haveren, R., Breedveld, S., Keijzer, M., Voet, P., Heijmen, B., Ogryczak, W.: Lexicographic extension of the reference point method applied in radiation therapy treatment planning. Eur. J. Oper. Res. (in press) (2017). doi:10.1016/j.ejor.2017.04.062
34. Van Haveren, R., Ogryczak, W., Verduijn, G., Keijzer, M., Heijmen, B., Breedveld, S.: Fast and fuzzy multi-objective radiotherapy treatment plan generation for head-and-neck cancer patients with the lexicographic reference point method (LRPM). Phys. Med. Biol. **62**, 4318 (2017). doi:10.1088/1361-6560/62/11/4318
35. Hock, W., Schittkowski, K.: Test Examples for Nonlinear Programming Codes. Springer, New York (1981)
36. Hoffmann, A.L., Siem, A.Y.D., den Hertog, D., Kaanders, J.H.A.M., Huizenga, H.: Derivative-free generation and interpolation of convex Pareto optimal IMRT plans. Phys. Med. Biol. **51**, 6349–6369 (2006). doi:10.1088/0031-9155/51/24/005

37. Jee, K.W., McShan, D.L., Fraass, B.A.: Lexicographic ordering: intuitive multicriteria optimization for IMRT. Phys. Med. Biol. **52**, 1845–1861 (2007). doi:10.1088/0031-9155/52/7/006
38. Kurzak, J., Ltaief, H., Dongarra, J., Badia, R.: Scheduling dense linear algebra operations on multicore processors. Concurr. Comput. Pract. Exp. **22**, 15–44 (2009). doi:10.1002/cpe.1467
39. Li, R., Xing, L.: An adaptive planning strategy for station parameter optimized radiation therapy (SPORT): Segmentally boosted VMAT. Med. Phys. **40**, 050701 (2013). doi:10.1118/1.4802748
40. Llacer, J., Deasy, J.O., Bortfeld, T.R., Solberg, T.D., Promberger, C.: Absence of multiple local minima effects in intensity modulated optimization with dose-volume constraints. Phys. Med. Biol. **48**, 183–210 (2003). doi:10.1088/0031-9155/48/2/304
41. Lustig, I., Marsten, R., Shanno, D.: On implementing Mehrotraś predictor-corrector interior-point method for linear programming **2**, 435–449 (1992). doi:10.1137/0802022
42. Mehrotra, S.: On the implementation of a primal-dual interior point method. SIAM J. Optim. **2**, 575–601 (1992). doi:10.1137/0802028
43. Men, C., Romeijn, E., Taşkin, C., Dempsey, J.: An exact approach to direct aperture optimization in IMRT treatment planning. Phys. Med. Biol. **52**, 7333–7352 (2007). doi:10.1088/0031-9155/52/24/009
44. Netlib: Netlib. http://www.netlib.org/lp/
45. Niemierko, A.: Reporting and analyzing dose distributions: a concept of equivalent uniform dose. Med. Phys. **24**, 103–110 (1997). doi:10.1118/1.598063
46. Nocedal, J., Wright, S.: Numerical Optimization. Springer, New York (2000)
47. Pagès, A., Gondzio, J., Nabona, N.: Warmstarting for interior point methods applied to the long-term power planning problem. Eur. J. Oper. Res. **197**, 112–125 (2009). doi:10.1016/j.ejor.2008.05.022
48. Paige, C.C., Saunders, M.A.: LSQR: an algorithm for sparse linear equations and sparse least squares. ACM Trans. Math. Softw. **8**, 43–71 (1982). doi:10.1145/355984.355989
49. Pellegrini, F.: Software package and libraries for sequential and parallel graph partitioning, static mapping and clustering, sequential mesh and hypergraph partitioning, and sequential and parallel sparse matrix block ordering. http://www.labri.fr/perso/pelegrin/scotch/ (2012)
50. Pissanetsky, S.: Sparse Matrix Technology. Academic, London (1984)
51. Rocha, H., Dias, J., Ferreira, B., Lopes, M.: Noncoplanar beam angle optimization in IMRT treatment planning using pattern search methods. J. Phys. Conf. Ser. **616**, 12014–12023 (2015)
52. Romeijn, H.E., Ahuja, R.K., Dempsey, J.F., Kumar, A.: A column generation approach to radiation therapy treatment planning using aperture modulation. SIAM J. Optim. **15**, 838–862 (2005). doi:10.1137/040606612
53. Romeijn, H.E., Ahuja, R.K., Dempsey, J.F., Kumar, A., Li, J.G.: A novel linear programming approach to fluence map optimization for intensity modulated radiation therapy treatment planning. Phys. Med. Biol. **48**, 3521–3542 (2003). doi:10.1088/0031-9155/48/21/005
54. Romeijn, H.E., Dempsey, J.F., Li, J.G.: A unifying framework for multi-criteria fluence map optimization models. Phys. Med. Biol. **49**, 1991–2013 (2004). doi:10.1088/0031-9155/49/10/011
55. Rossi, L., Breedveld, S., Aluwini, S., Heijmen, B.: Non-coplanar beam angle class solutions to replace time-consuming patient-specific beam angle optimization in robotic prostate SBRT. Int. J. Radiat. Oncol. Biol. Phys. **92**, 762–770 (2015). doi:10.1016/j.ijrobp.2015.03.013
56. Rossi, L., Breedveld, S., Heijmen, B.J.M., Voet, P.W.J., Lanconelli, N., Aluwini, S.: On the beam direction search space in computerized non-coplanar beam angle optimization for IMRT - prostate SBRT. Phys. Med. Biol. **57**, 5441–5458 (2012). doi:10.1088/0031-9155/57/17/5441
57. Shanno, D.F., Vanderbei, R.J.: Interior-point methods for nonconvex nonlinear programming: orderings and higher-order methods. Math. Program. Ser. B **87**, 303–316 (2000). doi:10.1007/s101070050116
58. Sharfo, A.W., Voet, P., Breedveld, S., Mens, J.W., Hoogeman, M., Heijmen, B.: Comparison of VMAT and IMRT strategies for cervical cancer patients using automated planning. Radiother. Oncol. **114**, 395–401 (2015). doi:10.1016/j.radonc.2015.02.006
59. Simon, H.: Partitioning of unstructured problems for parallel processing. Comput. Syst. Eng. **2**, 135–148 (1991). doi:10.1016/0956-0521(91)90014-V
60. Sonneveld, P., van Gijzen, M.B.: IDR(s): a family of simple and fast algorithms for solving large nonsymmetric linear systems. SIAM J. Sci. Comput. **31**, 1035–1062 (2008). doi:10.1137/070685804
61. Thörnqvist, S., Hysing, L.B., Zolnáy, A.G., Söhn, M., Hoogeman, M.S., Muren, L.P., Bentzen, L., Heijmen, B.J.M.: Treatment simulations with a statistical deformable motion model to evaluate margins for multiple targets in radiotherapy for high-risk prostate cancer. Radiother. Oncol. **109**, 344–349 (2013). doi:10.1016/j.radonc.2013.09.012

62. Tian, Z., Peng, F., Folkerts, M., Tan, J., Jia, X., Jiang, S.: Multi-GPU implementation of a VMAT treatment plan optimization algorithm. Med. Phys. **42**, 2841–2852 (2015). doi:10.1118/1.4919742
63. Vanderbei, R.: Nonlinear optimization models. http://orfe.princeton.edu/%7Ervdb/ampl/nlmodels/
64. Vanderbei, R.J.: LOQO: an interior point code for quadratic programming. Optim. Methods Softw. **11**, 451–484 (1999). doi:10.1080/10556789908805759
65. Vanderbei, R.J., Shanno, D.F.: An interior point algorithm for nonconvex nonlinear programming. Comput. Optim. Appl. **13**, 231–252 (1999). doi:10.1023/A:1008677427361
66. Voet, P., Breedveld, S., Dirkx, M., Levendag, P., Heijmen, B.: Integrated multi-criterial optimization of beam angles and intensity profiles for coplanar and non-coplanar head and neck IMRT and implications for VMAT. Med. Phys. **39**, 4858–4865 (2012). doi:10.1118/1.4736803
67. Voet, P., Dirkx, M., Breedveld, S., Al-Mamgani, A., Incrocci, L., Heijmen, B.: Fully automated VMAT plan generation for prostate cancer patients. Int. J. Radiat. Oncol. Biol. Phys. **88**, 1175–1179 (2014). doi:10.1016/j.ijrobp.2013.12.046
68. Voet, P., Dirkx, M., Breedveld, S., Fransen, D., Levendag, P., Heijmen, B.: Towards fully automated multi-criterial plan generation: a prospective clinical study. Int. J. Radiat. Oncol. Biol. Phys. **85**, 866–872 (2013). doi:10.1016/j.ijrobp.2012.04.015
69. Van de Water, S., Kooy, H., Heijmen, B., Hoogeman, M.: Shortening delivery times of intensity modulated proton therapy by reducing proton energy layers during treatment plan optimization. Int. J. Radiat. Oncol. Biol. Phys. **92**, 460–468 (2015). doi:10.1016/j.ijrobp.2015.01.031
70. Wilkens, J.J., Alaly, J.R., Zakarian, K., Thorstad, W.L., Deasy, J.O.: IMRT treatment planning based on prioritizing prescription goals. Phys. Med. Biol. **52**, 1675–1692 (2007). doi:10.1088/0031-9155/52/6/009
71. Wright, S.J.: Primal-Dual Interior-Point Methods. SIAM Publishers, Philadelphia (1997)
72. Wu, Q., Mohan, R.: Algorithms and functionality of an intensity modulated radiotherapy optimization system. Med. Phys. **27**, 701–711 (2000). doi:10.1118/1.598932
73. Yildirim, E.A., Wright, S.J.: Warm-start strategies in interior-point methods for linear programming. SIAM J. Optim. **12**, 782–810 (2002). doi:10.1137/S1052623400369235
74. Ziegenhein, P., Kamerling, C., Bangert, M., Kunkel, J., Oelfke, U.: Performance-optimized clinical IMRT planning on modern CPUs. Phys. Med. Biol. **58**, 3705–3715 (2013). doi:10.1088/0031-9155/58/11/3705
75. Zinchenko, Y., Craig, T., Keller, H., Terlaky, T., Sharpe, M.: Controlling the dose distribution with gEUD-type constraints within the convex radiotherapy optimization framework. Phys. Med. Biol. **53**, 3231–3250 (2008). doi:10.1088/0031-9155/53/12/011