

# NEW Delta4 Insight

Independent verification of your TPS calculations

**We provide medical physicists with patient QA  
from prescription to final fraction!**

Delta4 Insight is an independent secondary 3D dose calculation software that utilizes a Monte Carlo engine to verify the calculations of a clinic's Treatment Planning System (TPS).

"Our ambition when developing Delta4 Insight was to extend our Delta4 family of products into a full end-to-end verification solution where the customer can be confident that Delta4 provides the true values and not just the deviations."

Görger Nilsson, CTO, and founder

Delta4 Insight completes the ScandiDos patient QA product portfolio which now covers  
- patient QA from prescription to final fraction.

Learn more



[delta4family.com](https://delta4family.com)

# Distributed and scalable optimization for robust proton treatment planning

Anqi Fu<sup>1</sup>  | Vicki T. Taasti<sup>2</sup>  | Masoud Zarepisheh<sup>1</sup> 

<sup>1</sup>Department of Medical Physics, Memorial Sloan Kettering Cancer Center, New York, New York, USA  
(Email: [zarepism@mskcc.org](mailto:zarepism@mskcc.org))

<sup>2</sup>Department of Radiation Oncology, Maastricht University Medical Center, Maastricht, The Netherlands  
(Email: [vicki.taasti@maastro.nl](mailto:vicki.taasti@maastro.nl))

## Correspondence

Anqi Fu, Department of Medical Physics, Memorial Sloan Kettering Cancer Center, New York, NY, USA.  
Email: [fua@mskcc.org](mailto:fua@mskcc.org)

## Funding information

National Institutes of Health, Grant/Award Number: P30 CA008748

## Abstract

**Background:** The importance of robust proton treatment planning to mitigate the impact of uncertainty is well understood. However, its computational cost grows with the number of uncertainty scenarios, prolonging the treatment planning process.

**Purpose:** We developed a fast and scalable distributed optimization platform that parallelizes the robust proton treatment plan computation over the uncertainty scenarios.

**Methods:** We modeled the robust proton treatment planning problem as a weighted least-squares problem. To solve it, we employed an optimization technique called the alternating direction method of multipliers with Barzilai–Borwein step size (ADMM-BB). We reformulated the problem in such a way as to split the main problem into smaller subproblems, one for each proton therapy uncertainty scenario. The subproblems can be solved in parallel, allowing the computational load to be distributed across multiple processors (e.g., CPU threads/cores). We evaluated ADMM-BB on four head-and-neck proton therapy patients, each with 13 scenarios accounting for 3 mm setup and 3.5% range uncertainties. We then compared the performance of ADMM-BB with projected gradient descent (PGD) applied to the same problem.

**Results:** For each patient, ADMM-BB generated a robust proton treatment plan that satisfied all clinical criteria with comparable or better dosimetric quality than the plan generated by PGD. However, ADMM-BB's total runtime averaged about 6 to 7 times faster. This speedup increased with the number of scenarios.

**Conclusions:** ADMM-BB is a powerful distributed optimization method that leverages parallel processing platforms, such as multicore CPUs, GPUs, and cloud servers, to accelerate the computationally intensive work of robust proton treatment planning. This results in (1) a shorter treatment planning process and (2) the ability to consider more uncertainty scenarios, which improves plan quality.

## KEYWORDS

distributed optimization, proton treatment planning, robust optimization

## 1 | INTRODUCTION

Proton treatment planning has been an active topic of research over the last decade. The sharp dose fall-off of protons, which makes proton therapy an appealing treatment modality, also renders it vulnerable to errors and uncertainties during treatment planning. Consequently, proton plans are usually developed using robust optimization.<sup>1</sup> The dose distribution for each potential

uncertainty scenario (e.g., due to setup or range estimation errors) is computed, then the plan is optimized to fulfill clinical objectives even in these scenarios.

To produce the most robust plan, we would like to consider as many scenarios as possible, covering all potential sources of uncertainty. This presents a number of computational challenges. First, the more scenarios we include in the optimization problem, the longer it will take to solve the problem. This could prolong the

treatment planning process and limit the number of parameter adjustments (e.g., objective weights, penalties) that we make, which may result in a suboptimal plan. Second, the memory required to handle all the desired scenarios could exceed the resources available to us, especially on a single computing platform. It is thus of great clinical importance to develop a fast, distributed, and scalable method for robust proton treatment planning.

Given the large size of robust optimization problems in proton therapy, first-order methods like gradient descent have been the de facto solution technique. In particular, projected gradient descent (PGD) enjoys widespread popularity, as it can handle upper and lower bounds on spot intensities. However, while gradient descent is widely used, it possesses several weaknesses that make it undesirable for solving complex, data-intensive optimization problems. First, it requires the calculation of the gradient, which may be mathematically cumbersome or intractable. Second, its convergence depends heavily on the step size (i.e., the distance moved in the direction of the gradient), which is typically chosen via a line search. This line search is computationally demanding and scales poorly with the size of the problem. Moreover, the very serial format of the line search makes gradient descent unamenable to parallelization. Alternative methods exist for step size selection,<sup>2–5</sup> but in general, optimization practitioners turn to the class of distributed algorithms to solve very large problems.

In distributed optimization, multiple agents (e.g., CPUs) collaborate to solve an optimization problem. A typical algorithm will decompose the problem into parts and distribute each part to an agent, which carries out its computation using local information. The agents then combine their results to produce a solution. Due to the use of many agents, these algorithms scale remarkably well: Their computational demands increase very modestly with the size and complexity of the problem. Consequently, distributed optimization has been an active topic of research for decades,<sup>6–9</sup> and distributed algorithms have been applied in a variety of fields.<sup>10–13</sup>

The alternating direction method of multipliers (ADMM) is a distributed optimization algorithm dating back to the 1970s.<sup>14–16</sup> It has seen renewed interest over the last few years, thanks to its success in solving large-scale optimization problems that arise in data science and machine learning.<sup>17–21</sup> The key to ADMM's success is its ability to split a problem into smaller subproblems, which can be solved independently from each other, making it ideal for parallel computation. A clever split can also yield mathematically simple subproblems that permit a closed-form solution. Recently, Zarepisheh, Xing, and Ye<sup>22</sup> introduced a new variant of ADMM with improved convergence properties and evaluated its performance on fluence map optimization problems in intensity modulated radiation therapy (IMRT). They showed that ADMM outperforms other optimization techniques, including an active-set

method (FNNLS), a gradient-based method (SBB), and an interior-point solver (CPLEX). This demonstrates the potential of ADMM to efficiently handle large treatment planning problems.

Our paper extends the application of ADMM to robust proton treatment plan optimization. In particular, we exploit the special structure of the robust optimization problem, which enables us to reformulate the problem and split it into smaller subproblems, each corresponding to a separate uncertainty scenario. A major difference between our study and the aforementioned study by Zarepisheh, Xing, and Ye is that we implement a *fully parallelized* version of ADMM that distributes the workload required to solve these subproblems across multiple CPU threads/cores. This allows us to achieve a greater and more consistent speed advantage over PGD. More importantly, we show that our ADMM algorithm scales well with the number of scenarios, making it possible to include many different sources of uncertainty in the treatment planning process.

## 2 | METHODS

### 2.1 | Problem formulation and gradient descent

Suppose we have  $N$  scenarios, including the nominal scenario, represented by dose-influence matrices  $A_s \in \mathbf{R}_+^{m \times n}$  for  $s = 1, \dots, N$ . Here  $m$  is the number of voxels and  $n$  is the number of beamlets. Let  $a_{s,i} \in \mathbf{R}_+^n$  be row  $i$  of  $A_s$ , corresponding to voxel  $i$ . The prescribed dose to each voxel is given in the form of a vector  $p \in \mathbf{R}_+^m$ , where  $p_i$  is equal to the prescription (i.e., a constant scalar  $D_{\text{pres}} > 0$ ) for target voxels and zero for nontarget voxels. Our objective is to find spot intensities  $x \in \mathbf{R}^n$  that minimize the deviation between the actual and prescribed dose across all scenarios.

Formally, let us define the *scenario-specific objective function* to be

$$f_s(x) := \sum_{i=1}^m w_i \|a_{s,i}^T x - p_i\|_2^2, \quad (1)$$

where  $w_i \geq 0$  is the relative importance weight on voxel  $i$ . While in principle, different voxels within a structure can admit different weights, in our model, we assign the same weight to all voxels in a structure and only allow weights to vary across structures. Our goal is to solve the optimization problem

$$\begin{aligned} & \text{minimize} && f(x) := \sum_{s=1}^N f_s(x) \\ & \text{subject to} && x \geq 0 \end{aligned} \quad (2)$$

with variable  $x \in \mathbf{R}^n$ . Here, we have assumed all scenarios occur with equal probability. It is straightforward

to incorporate differing probabilities by replacing  $w_i$  with scenario-specific weights  $\tilde{w}_{s,i} := c_s w_i$ , where  $c_s \in [0, 1]$  is the probability of scenario  $s$ .

PGD starts from an initial estimate of the spots  $x^{(0)} \in \mathbf{R}_+^n$  and iteratively computes

$$\nabla f(x^{(k)}) = \sum_{s=1}^N \sum_{i=1}^m 2w_i a_{s,i} (a_{s,i}^T x^{(k)} - p_i), \quad (3)$$

$$x^{(k+1)} = \max(x^{(k)} - \rho^{(k)} \nabla f(x^{(k)}), 0), \quad (4)$$

where  $\rho^{(k)} > 0$  is the step size in iteration  $k$ , selected using a line search technique such as *Armijo* [23, section 3.1] to ensure improvement in the objective value. *Armijo* line search starts from an initial step size  $\rho^{(k)} = \alpha$  and shrinks the step size by a factor of  $\gamma$  until it produces  $x^{(k+1)}$  that satisfies

$$f(x^{(k+1)}) \geq f(x^{(k)}) + \beta \nabla f(x^{(k)})^T (x^{(k+1)} - x^{(k)}). \quad (5)$$

Here,  $\alpha > 0$ ,  $\beta \in (0, 1)$ , and  $\gamma \in (0, 1)$  are fixed parameters. PGD proceeds until some stopping criterion is reached, typically when the change in the objective value,  $|f(x^{(k)}) - f(x^{(k+1)})|$ , falls below a user-defined cutoff.

The gradient of each scenario's objective function,

$$\nabla f_s(x^{(k)}) = \sum_{i=1}^m 2w_i a_{s,i} (a_{s,i}^T x^{(k)} - p_i), \quad (6)$$

can be computed in parallel, that is, simultaneously on different processing units (CPUs, GPUs, etc), and summed up to obtain the gradient of the full objective,  $\nabla f(x^{(k)})$ . However, the *Armijo* condition 5 must be checked and updated serially, creating a bottleneck in any parallel implementation of PGD.

## 2.2 | Distributed ADMM

To apply ADMM to problem 2, we must first reformulate it so the objective function is separable. The current objective  $f(x)$  consists of a sum of scenario-specific objectives  $f_s(x)$ , which share the same variable  $x$ . We will replace  $x$  with new variables  $x_1, \dots, x_N \in \mathbf{R}^n$ , representing the scenario-specific spots, and a so-called consensus variable  $z \in \mathbf{R}^n$ . We then introduce a consensus constraint  $x_s = z$  for  $s = 1, \dots, N$  to ensure the spot intensities are equal. This results in the mathematically equivalent formulation

$$\begin{aligned} & \text{minimize} && \sum_{s=1}^N f_s(x_s) \\ & \text{subject to} && x_s = z, s = 1, \dots, N, \\ & && z \geq 0. \end{aligned} \quad (7)$$

The objective function in problem 7 is separable across scenarios: The only link between  $\{f_s(x_s)\}_{s=1}^N$  is the requirement that  $x_1 = \dots = x_N = z$ . However, even with the consensus constraint, ADMM is able to split this problem into independent subproblems. The reader is referred to S. Boyd et al. (2011)<sup>15</sup> for a detailed description of ADMM.

ADMM starts from an initial estimate of the spot intensity vector  $z^{(0)} \in \mathbf{R}_+^n$ , the constant parameter  $\rho > 0$ , and the dual variables  $y_s^{(0)} \in \mathbf{R}^n$  for  $s = 1, \dots, N$ . (The value  $1/\rho$  is known as the step size.) In each iteration  $k = 0, 1, 2, \dots$

1. For  $s = 1, \dots, N$ , solve for the scenario spot intensities

$$x_s^{(k+1)} = \arg \min_{x_s} f_s(x_s) + \frac{\rho}{2} \|x_s - z^{(k)} + y_s^{(k)} / \rho\|_2^2. \quad (8)$$

2. Project the average of the scenario spot intensities onto the nonnegative orthant:

$$z^{(k+1)} = \max\left(\frac{1}{N} \sum_{s=1}^N x_s^{(k+1)}, 0\right). \quad (9)$$

3. For  $s = 1, \dots, N$ , update the dual variables

$$y_s^{(k+1)} = y_s^{(k)} + \rho(x_s^{(k+1)} - z^{(k+1)}). \quad (10)$$

4. Terminate if stopping criterion 14 is satisfied and return  $x^* = z^{(k+1)}$  as the solution.

Notice that the scenario subproblems 8 can be solved in parallel.

### 2.2.1 | Solving the subproblems

Subproblem 8 is an unconstrained least-squares problem with a closed-form solution  $x_s^{(k+1)}$ , which can be obtained by solving a system of linear equations of the form  $B^{(s)} x_s^{(k+1)} = c^{(k)}$ , where  $B^{(s)} \in \mathbf{R}_+^{n \times n}$  is an iteration-independent symmetric positive definite matrix and  $c^{(k)} \in \mathbf{R}_+^n$ . This system is sometimes referred to as the normal equations. Since the value of  $B^{(s)}$  remains the same throughout the ADMM loop, we can solve subproblem 8 efficiently by forming the Cholesky factorization of  $B^{(s)}$  once prior to the start of ADMM and caching it, then using backward substitution to calculate  $x_s^{(k+1)}$  each iteration. This reduces the computational load and ensures our subproblem step is still parallelizable. See the Appendix for mathematical details.

## 2.2.2 | ADMM with Barzilai–Borwein (BB) step size

While ADMM will converge for any  $\rho > 0$ , the value of  $\rho$  often has an effect on the practical convergence rate. In some cases, allowing  $\rho$  to vary in each iteration will result in faster convergence. One popular method for selecting a variable step size is the BB method.<sup>24–26</sup> This method has shown great success when combined with ADMM.<sup>22</sup> The BB step size in iteration  $k$  is given by

$$\beta^{(k)} = -\frac{\Delta d^{(k)T} \Delta y^{(k)}}{\|\Delta d^{(k)}\|_2^2}, \quad (11)$$

where  $\Delta y^{(k)} = y^{(k)} - y^{(k-1)}$  and  $\Delta d^{(k)} = (x^{(k+1)} - z^{(k+1)}) - (x^{(k)} - z^{(k)})$ . If  $\beta^{(k)} > 0$ , we use it in place of  $\rho$  in our dual variable update 10. Otherwise, we revert to the user-defined constant  $\rho > 0$ .

Note that subproblem 8 always uses the default  $\rho$ , since for speed purposes, we do not recompute the Cholesky factorization each iteration. In addition, the BB step size is calculated using simple linear algebra, not the serial loop required by a backtracking line search (e.g., *Armijo*). As we will see in Section 3, this gives ADMM a significant advantage over PGD in terms of runtime and memory efficiency.

## 2.2.3 | Stopping criterion

The convergence of ADMM with variable step size is still an active topic of research.<sup>27–29</sup> In our algorithm, we employ a stopping criterion based on the optimality conditions of problem 7, which we have observed works well in practice. This translates into checking whether the residuals associated with these conditions,

$$r_{\text{prim}}^{(k)} = (x_1^{(k)} - z^{(k)}, \dots, x_N^{(k)} - z^{(k)}), \quad (12)$$

$$r_{\text{dual}}^{(k)} = (\rho(z^{(k)} - z^{(k-1)}), \dots, \rho(z^{(k)} - z^{(k-1)})), \quad (13)$$

are close to zero. (Here,  $r_{\text{prim}}$  is referred to as the primal residual and  $r_{\text{dual}}$  as the dual residual.) Thus, a reasonable stopping criterion is

$$\|r_{\text{prim}}^{(k)}\|_2 \leq \epsilon_{\text{prim}}, \quad \|r_{\text{dual}}^{(k)}\|_2 \leq \epsilon_{\text{dual}} \quad (14)$$

for tolerances  $\epsilon_{\text{prim}} > 0$  and  $\epsilon_{\text{dual}} > 0$ . These tolerances are typically chosen with respect to user-defined cutoffs  $\epsilon_{\text{abs}} > 0$  and  $\epsilon_{\text{rel}} > 0$  using

$$\epsilon_{\text{prim}} = \epsilon_{\text{abs}} \sqrt{Nn} + \epsilon_{\text{rel}} \max(\|x^{(k)}\|_2, \sqrt{N}\|z^{(k)}\|_2), \quad (15)$$

$$\epsilon_{\text{dual}} = \epsilon_{\text{abs}} \sqrt{Nn} + \epsilon_{\text{rel}} \|y^{(k)}\|_2, \quad (16)$$

**TABLE 1** Head-and-neck cancer patient information

	Patient			
	1	2	3	4
Beam configuration	40°, 320°	220°, 320°	30°, 60°	30°, 300°
CTV volume (cm <sup>3</sup> )	91.4	2.8	72.9	59.5
Number of voxels ( $m$ )	98 901	50 728	119 258	126 072
Number of spots ( $n$ )	5762	707	4697	3737

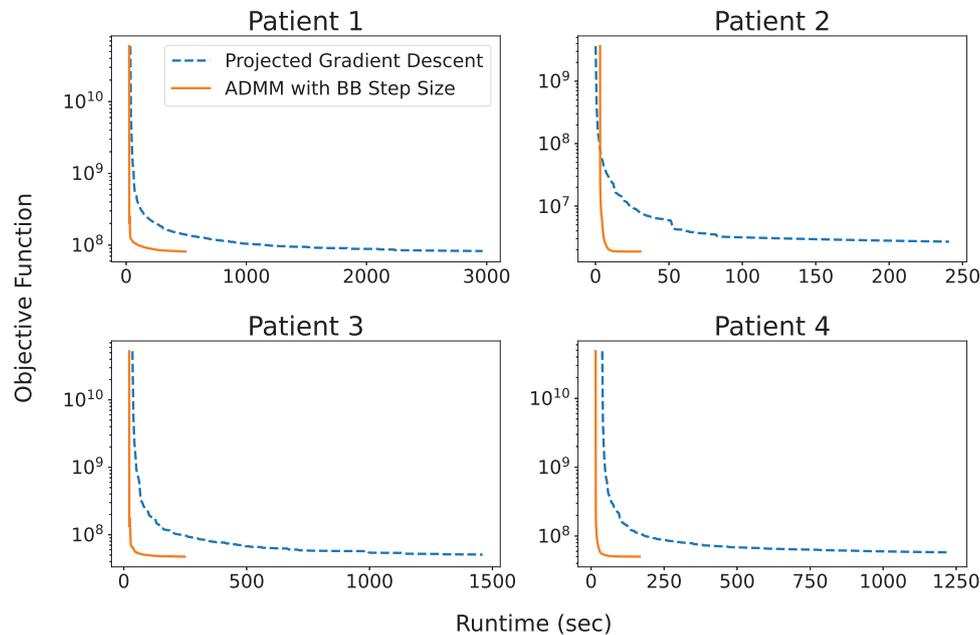
where  $x^{(k)} := (x_1^{(k)}, \dots, x_N^{(k)})$  and  $y^{(k)} := (y_1^{(k)}, \dots, y_N^{(k)})$ . Intuitively, we want to stop when (1) the scenario-specific spots are approximately equal ( $r_{\text{prim}}^{(k)} \approx 0$  implies  $x_1^{(k)} \approx \dots \approx x_N^{(k)} \approx z^{(k)}$ ) and (2) the most recent iteration of ADMM yielded little change in the spot intensities ( $r_{\text{prim}}^{(k)} \approx 0$  implies  $z^{(k)} \approx z^{(k-1)}$ ). This indicates that the optimality conditions for problem 7 have been fulfilled. We refer the reader to section 3.3 of S. Boyd et al. (2011)<sup>15</sup> for a derivation of the stopping criterion.

## 2.3 | Patient population and computational framework

We tested our ADMM algorithm with BB step size (ADMM-BB) on four head-and-neck cancer patient cases from The Cancer Imaging Archive (TCIA).<sup>32,33</sup> For each patient, only the primary clinical target volume (CTV) was considered in the optimization, with a prescribed dose of  $D_{\text{pres}} = 70$  Gy delivered in 35 fractions of 2 Gy per fraction. Dose calculations were performed in MATLAB using the open-source software MatRad and the pencil beam dose calculation algorithm.<sup>34,35</sup> The proton spots were placed on a rectangular grid, covering the planning target volume (PTV) region plus 1 mm out from its perimeter with a spot spacing of 5 mm. All patients were planned using two coplanar beams. See Table 1 for more details.

To create uncertainty scenarios, we simulated range over- and undershoots by rescaling the stopping power ratio (SPR) image  $\pm 3.5\%$  following typical range margin recipes used in proton therapy.<sup>36,37</sup> We also simulated setup errors by shifting the isocenter  $\pm 3$  mm in the  $x$ ,  $y$ , and  $z$  direction. Combining the range and setup errors gave us a total of 13 scenarios, including the nominal scenario. For each scenario, we computed the dose-influence matrix using a relative biological effectiveness (RBE) of 1.1.

We implemented PGD and ADMM-BB in Python using the built-in `multiprocessing` library. This library supports parallel computation across multiple CPUs and CPU threads/cores. The algorithms were executed on a server with 2 Intel Xeon Gold 6248 CPUs @ 2.50 GHz / 20 cores and 128 GB RAM. We terminated PGD when the relative change in the objective value was about  $10^{-4}$  to  $10^{-5}$ . For ADMM-BB, we combined this same



**FIGURE 1** Objective function value versus algorithm runtime (s) for all patient cases

criterion with the residual stopping criterion 14 using  $\epsilon_{\text{abs}} = \epsilon_{\text{rel}} = 10^{-6}$ .

### 3 | RESULTS

#### 3.1 | Algorithm runtime comparisons

Figure 1 shows the objective function value over the course of the total runtime of PGD and ADMM-BB, for all four patients. Clearly for all patients, ADMM-BB converges much faster than PGD to approximately the same objective value. On patient 1, a relatively large case, ADMM-BB took 8.3 min to converge, whereas PGD required 50 min to achieve roughly the same objective. For a smaller case like patient 2, ADMM-BB converged in 31 s, while PGD required 241 s. Altogether across all patients, ADMM-BB converged on average six to seven times faster than PGD to a very similar objective and spot intensities—a significant speedup.

#### 3.2 | Treatment plan comparisons

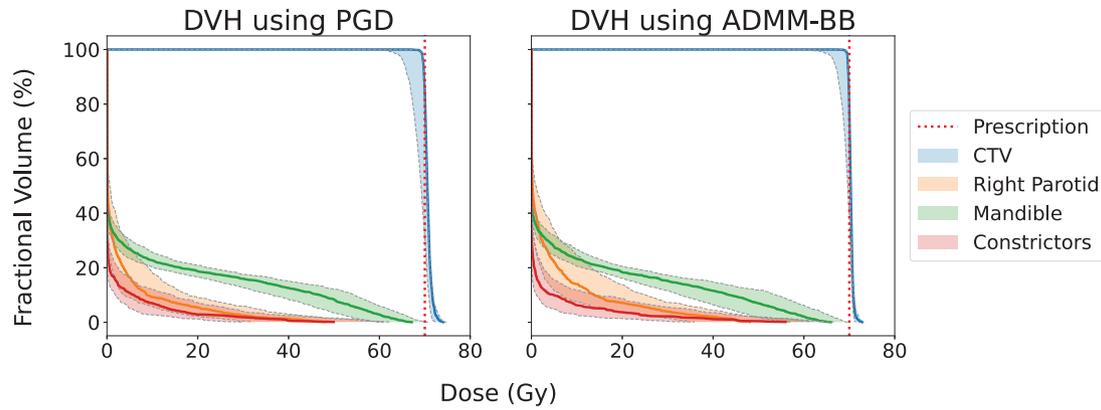
Figures 2 and 3 compare the treatment plans produced by PGD and ADMM-BB for patient 2. Figure 2 depicts the DVH bands: Each band, color-coded to a particular structure, represents the range of DVH curves across all 13 scenarios, while the corresponding solid curve is the DVH in the nominal scenario. The vertical dotted line marks the prescription  $D_{\text{pres}} = 70$  Gy. It is clear from an inspection of the figure that PGD and ADMM-BB converge to near-identical DVH bands. The only difference is that ADMM-BB achieves a tighter CTV band

around  $D_{\text{pres}}$  at the expense of a slight increase in the mean dose to the right parotid (Figure 2). Figure 3 shows the box plots of a few dose-volume clinical metrics for this patient. The box plots depict the value of the metric for each of the 13 uncertainty scenarios. The dotted line represents the clinical constraint for the metric; this line is a lower bound for D98% on the CTV and an upper bound for all other metrics. (See Table 2 for a full description of the criteria.) It is apparent from the plots that both PGD and ADMM-BB produce robust plans that respect the clinical constraints in nearly all scenarios. On the OAR metrics, PGD and ADMM-BB achieve similar performance: The mean doses to the right parotid and the constrictors are nearly identical, while the maximum dose to the mandible is slightly lower for ADMM-BB. However, ADMM-BB outperforms PGD on the CTV metrics.

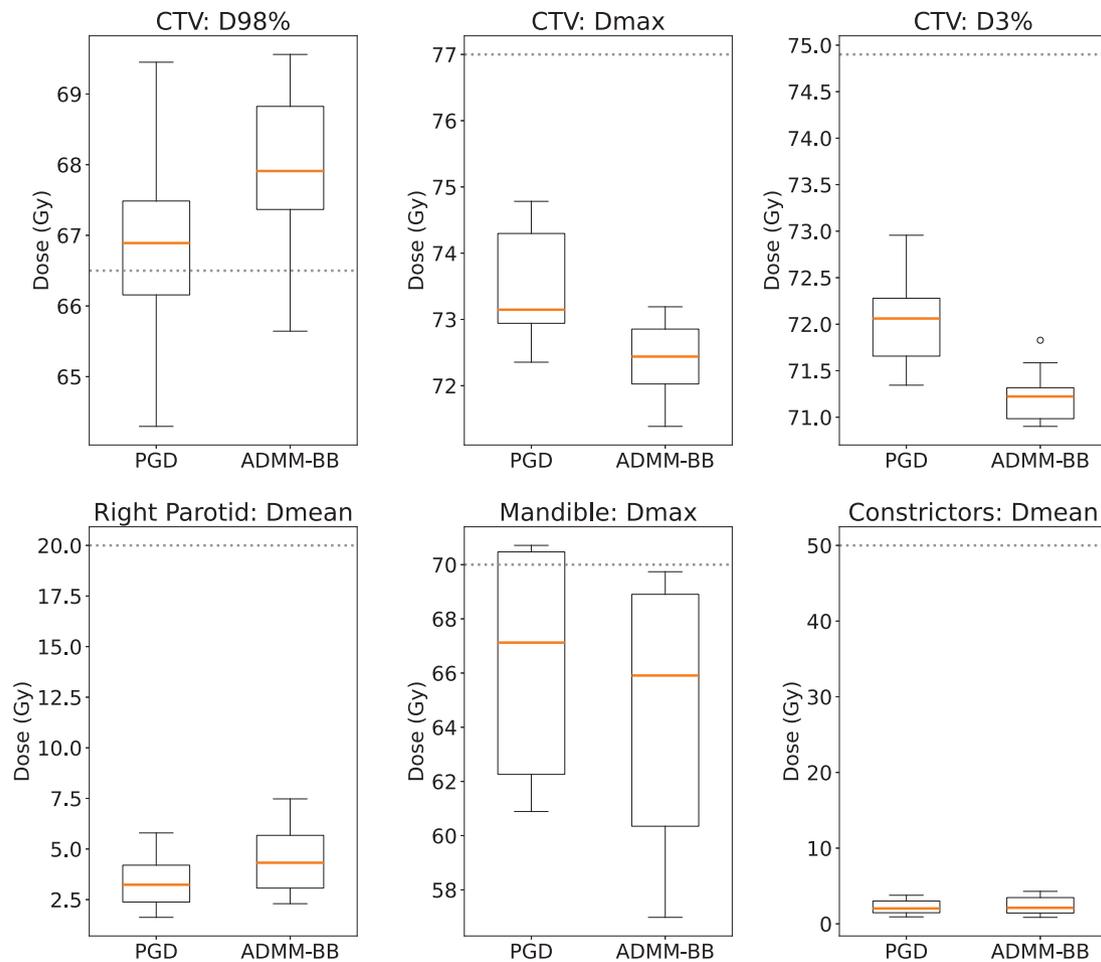
Figures 4 and 5 depict the DVH bands and clinical metrics for patient 4. Again, the PGD and ADMM-BB treatment plans are nearly identical. ADMM-BB

**TABLE 2** Recommended clinical dose bounds for head-and-neck cancer patients<sup>38</sup>

Structure	Dose constraint
CTV	D98% > 66.5 Gy
	Dmax < 77 Gy
	D3% < 74.9 Gy
Parotid	Dmean < 20 Gy
Mandible	Dmax < 70 Gy
Larynx	Dmean < 45 Gy
Constrictors	Dmean < 50 Gy



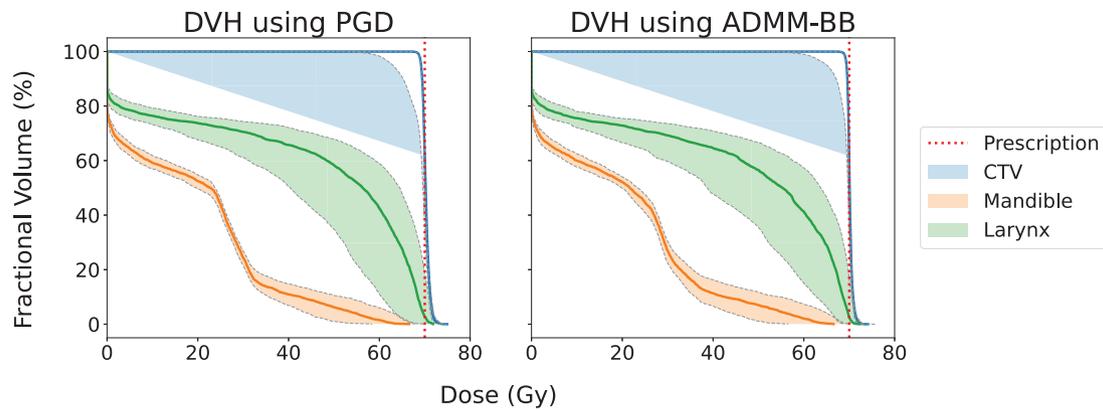
**FIGURE 2** Dose-volume histogram (DVH) bands across all scenarios for patient 2



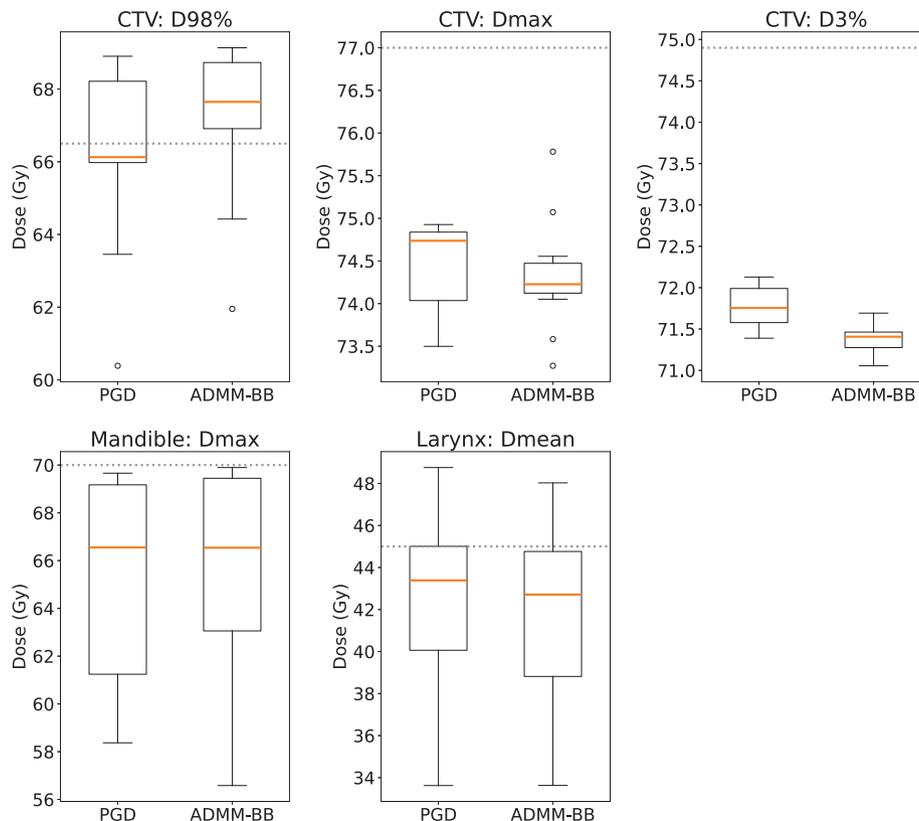
**FIGURE 3** Dose-volume clinical metrics for patient 2. The box plot spans the values over the 13 uncertainty scenarios, with the orange line indicating the median. The dotted lines mark the clinical constraints: Higher is better for D98% on the CTV, while lower is better for all other metrics.

produces a tighter CTV band below 70 Gy, indicating superior robustness, at the expense of a somewhat wider DVH band on the mandible. The box plots of the dose-volume metrics for the ADMM-BB plan are also largely within the desired clinical constraints, and the

CTV box plots in particular show an improvement over those of PGD (Figure 4), similar to what we saw in patient 2. The PGD and ADMM-BB treatment plans for other patients reflect the same pattern; we have omitted them here for brevity. Overall, we see that ADMM-BB



**FIGURE 4** Dose-volume histogram (DVH) bands across all scenarios for patient 4



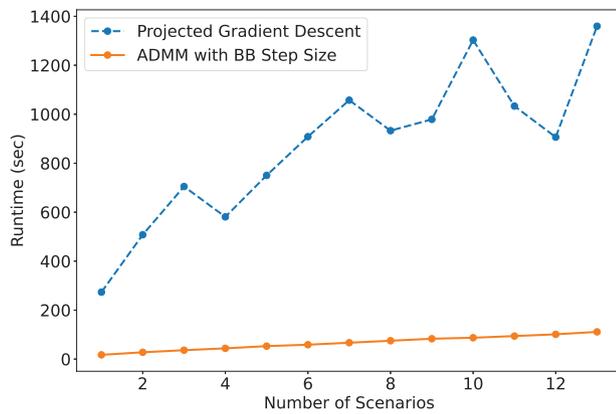
**FIGURE 5** Dose-volume clinical metrics for patient 4. The box plot spans the values over the 13 uncertainty scenarios, with the orange line indicating the median. The dotted lines mark the recommended clinical constraints: Higher is better for D98% on the CTV, while lower is better for all other metrics.

produces plans identical to or better than the plans produced by PGD, but in a fraction of the runtime.

### 3.3 | Scalability over multiple scenarios

Not only is ADMM-BB faster than PGD, it also scales better with the number of scenarios. Figure 6 shows the time required by PGD and ADMM-BB to produce a treatment plan for patient 4 for different numbers of

scenarios in the optimization problem. Specifically, we solved problem 2 for  $N \in \{1, 2, \dots, 13\}$  and recorded the total runtime in each case. Parallelization in ADMM-BB was carried out across the included scenarios. It is clear from the figure that ADMM-BB outperforms PGD by a wide margin. ADMM-BB's runtime increases modestly from 17.3 s at  $N = 1$  to 111.2 s at  $N = 13$ . By contrast, PGD goes from 273.4 s at  $N = 1$  to 1359.4 s at  $N = 13$ . At its peak, ADMM-BB is over 12 times faster than PGD. This pattern holds true for all the other patients as well.



**FIGURE 6** Algorithm runtime (s) versus number of scenarios (N) for patient 4's treatment plan

## 4 | DISCUSSION

In this study, we presented a distributed optimization method for solving robust proton treatment planning problems. Our method splits the problem into smaller parts, which can be handled on separate machines/processors, allowing us to reduce the overall planning time and overcome the limits (e.g., memory) of single-machine computation. As a result, we can efficiently plan large patient cases with many voxels and spots. We can also consider more uncertainty scenarios for any given patient, enabling us to produce more robust treatment plans.

We have used PGD as our benchmark for comparison because it is commonly employed to solve unconstrained treatment planning problems.<sup>39–42</sup> More advanced versions of gradient descent, such as the non-linear conjugate gradient (CG) method and Nesterov-accelerated gradient descent, can also be applied to problem 2. While these algorithms may work better than PGD, they possess the same dependence on the line search step,<sup>23,43</sup> which was the main source of computational burden in our experiments. A variant of PGD exists that uses the BB step size.<sup>44</sup> The removal of line search does speed up PGD, but experiments show that ADMM-BB still generally converges faster in a nondistributed setting.<sup>22</sup> In sum, both the ADMM subproblem decomposition and the BB step size combine to give ADMM-BB a clear advantage over gradient descent.

Another advantage of ADMM is that it supports a wide variety of constraints. While PGD can only handle box constraints (i.e., upper and lower bounds on the spot intensities), ADMM can easily be modified to handle any type of linear constraint, such as maximum and mean dose constraints. More advanced versions of ADMM can also handle nonlinear constraints.<sup>45–47</sup> Other optimization algorithms exist that support linear/nonlinear constraints (e.g., interior point method,<sup>48</sup> sequential quadratic programming<sup>23</sup>), but generally, they require the calculation of the Hessian matrix, which is

computationally expensive for robust treatment planning problems.

In our formulation, we have chosen to split problem 2 by scenarios. It is also possible to split the problem by voxels, for example, create a separate subproblem for each target and organ at risk. Indeed, there are multi-block versions of ADMM that allow arbitrary splitting along rows and columns,<sup>49–52</sup> so we can theoretically accommodate any partition of voxels/spots. This is a natural direction for future research.

Finally, we plan to extend the implementation of ADMM-BB to other platforms. Our current software implementation parallelizes computation across multiple CPUs and CPU cores. We intend to add support for parallelization across GPUs, which should produce further speed improvements. With the rise of cloud computing, data storage and processing are increasingly moving to remote high performance computing (HPC) clusters. Our long-term goal is to implement ADMM-BB on these clusters, so that each subproblem (and its corresponding portion of the data) is handled by a separate machine or group of machines in the cluster. This will allow us to solve treatment planning problems of immense size. We foresee ADMM-BB's application in many data-intensive treatment environments, such as beam angle optimization and volumetric modulated arc therapy (VMAT), as well as more recent treatment planning modalities like  $4\pi$ <sup>53</sup> and station parameter optimized radiation therapy (SPORT).<sup>54</sup>

## 5 | CONCLUSIONS

We have developed a fast, distributed method for robust proton treatment planning. Our method splits the treatment planning problem into smaller subproblems, which can then be solved in parallel, improving runtime and memory efficiency. Moreover, we showed that the method scales well with the number of uncertainty scenarios. These advantages allow us to (1) shorten the time required by the treatment planning process, (2) incorporate more uncertainty scenarios into the robust optimization problem, and (3) improve plan quality by exploring a larger space of parameters.

## ACKNOWLEDGMENTS

We thank Joseph O. Deasy for his helpful comments and suggestions on the paper. This work was partially supported by MSK Cancer Center Support Grant/Core Grant from the NIH (P30 CA008748).

## CONFLICT OF INTEREST

The authors have no relevant conflicts of interest to disclose.

## ORCID

Anqi Fu  <https://orcid.org/0000-0002-2876-2942>

Vicki T. Taasti   
<https://orcid.org/0000-0002-4588-9769>  
 Masoud Zarepisheh   
<https://orcid.org/0000-0001-6616-5949>

## REFERENCES

- Unkelbach J, Alber M, Bangert M, et al. Robust radiotherapy planning. *Phys Med Biol*. 2018;63:22TR02.
- Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. *J Mach Learn Res*. 2011;12:2121-2159.
- Kingma DP, Ba J. Adam: A method for stochastic optimization. In: *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. San Diego, CA; *arXiv:1412.6980*. 2015. <https://dare.uva.nl/search?identifier=a20791d3-1aff-464a-8544-268383c33a75>
- Tan C, Ma S, Dai Y-H, Qian Y. Barzilai-Borwein Step Size for Stochastic Gradient Descent: Proceedings of the 30th Conference on Neural Information Processing Systems (NIPS). Barcelona, Spain; 2016;29. <https://papers.nips.cc/paper/2016/hash/c86a7ee3d8ef0b551ed58e354a836f2b-Abstract.html>
- Qiao Y, van Lew B, Lelieveldt BPF, Staring M. Fast automatic step size estimation for gradient descent optimization of image registration. *IEEE Trans Med Imag*. 2016;35:391-403.
- Bertsekas DP, Tsitsiklis JN. *Parallel and Distributed Computation: Numerical Methods*. Prentice-Hall; 1989.
- Sayed AH. Adaptation, learning, optimization over networks. *Found Trends Mach Learn*. Atlantis, Paradise Island, Bahamas; 2014;7:311-801.
- Nedić A, Liu J. Distributed optimization for control. *Annu Rev Control Robot Auton Syst*. 2018;1:77-103.
- Yang T, Yi X, Wu J, et al. A survey of distributed optimization. *Annu Rev Control*. 2019;47:278-305.
- Raffard R, Tomlin C, Boyd S. Distributed Optimization for Cooperative Agents: Application to Formation Flight: Proceedings of the 43rd IEEE Conference on Decision and Control (CDC). 2004:2453-2459. <http://cdc2004.ieeecss.org/>
- Rabbat M, Nowak R. Distributed Optimization in Sensor Networks: Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks. Berkeley, CA; 2004:20-27. <https://dl.acm.org/doi/proceedings/10.1145/984622>
- Wang Y, Wang S, Wu L. Distributed optimization approaches for emerging power systems operation: a review. *Electr Power Syst Res*. 2017;144:127-135.
- Fu A, Xing L, Boyd S. Operator splitting for adaptive radiation therapy with nonlinear health constraints. *Optim Methods Softw*. 2022;1-24. <https://doi.org/10.1080/10556788.2022.2078824>
- Gabay D, Mercier B. A dual algorithm for the solution of nonlinear variational problems via finite element approximations. *Comput Math Appl*. 1976;2:17-40.
- Boyd S, Parikh N, Chu E, Peleato B, Eckstein J. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found Trends Mach Learn*. 2011;3:1-122.
- Fang EX, He B, Liu H, Yuan X. Generalized alternating direction method of multipliers: new theoretical insights and applications. *Math Program Comput*. 2015;7:149-187.
- Huang Z, Hu R, Guo Y, Chan-Tin E, Gong Y. DP-ADMM: ADMM-based distributed learning with differential privacy. *IEEE Trans Inf Forensics Security*. 2019;15:1002-1012.
- Gu Y, Fan J, Kong L, Ma S, Zou H. ADMM for high-dimensional sparse penalized quantile regression. *Technometrics*. 2018;60:319-331.
- Ramdas A, Tibshirani RJ. Fast and flexible ADMM algorithms for trend filtering. *J Comput Graph Stat*. 2016;25:839-858.
- Dhar S, Congruí Y, Ramakrishnan N, Shah M. ADMM Based Machine Learning on Spark: IEEE International Conference on Big Data. Santa Clara, CA; 2015:1174-1182. <https://dblp.org/db/conf/bigdataconf/bigdataconf2015.html>
- Sawatzky A, Xu Q, Schirra CO, Anastasio MA. Proximal ADMM for multi-channel image reconstruction in spectral X-ray CT. *IEEE Trans Med Imag*. 2014;33:1657-1668.
- Zarepisheh M, Xing L, Ye Y. A computation study on an integrated alternating direction method of multipliers for large scale optimization. *Optim Lett*. 2018;12:3-15.
- Nocedal J, Wright SJ. *Numerical Optimization*. Springer-Verlag; 2006.
- Barzilai J, Borwein JM. Two-point step size gradient methods. *IMA J Numer Anal*. 1988;8:141-148.
- Raydan M. The Barzilai and Borwein gradient method for the large scale unconstrained minimization problem. *SIAM J Optim*. 1997;7:26-33.
- Dai Y, Fletcher R. Projected Barzilai-Borwein methods for large-scale box-constrained quadratic programming. *Numer Math*. 2005;100:21-47.
- Xu Z, Figueiredo M, Yuan X, Studer C, Goldstein T. Adaptive Relaxed ADMM: Convergence Theory and Practical Implementation: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Honolulu, HI; 2017:7389-7398. <https://cvpr2017.thecvf.com/>
- Xu Y, Liu M, Lin Q, Yang T. ADMM without a fixed penalty parameter: faster convergence with new adaptive penalization. In: *Adv Neural Inf Process Syst (NIPS)*. 2017;30. <https://proceedings.neurips.cc/paper/2017/hash/e97ee2054defb209c35fe4dc94599061-Abstract.html>
- Boj RI, Csetnek ER. ADMM for monotone operations: convergence analysis and rates. *Adv Comput Math*. 2019;45:327-359.
- Eckstein J, Bertsekas DP. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Math Program*. 1992;55:293-318.
- Gabay D. Applications of the method of multipliers to variational inequalities. In: Fortin M, Glowinski R, eds. *Augmented Lagrangian Methods: Applications to the Solution of Boundary-Value Problems*. North-Holland; 1983:299-331. <https://www.sciencedirect.com/bookseries/studies-in-mathematics-and-its-applications/vol/15/suppl/C>
- Clark K, Vendt B, Smith K, et al. The Cancer Imaging Archive (TCIA): maintaining and operating a public information repository. *J Digit Imaging*. 2013;26:1045-1057.
- Nolan T. Head-and-Neck Squamous Cell Carcinoma Patients with CT Taken During Pre-Treatment, Mid-Treatment, and Post-Treatment (HNSCC-3DCT-RT). Accessed 18 June 2022 <https://doi.org/10.7937/K9/TCIA.2018.13upr2xf>
- Wieser H-P, Cisternas E, Wahl N, et al. Development of the open-source dose calculation and optimization toolkit matRad. *Med Phys*. 2017;44:2556-2568.
- Wieser H, Wahl N, Gabrys H, et al. MatRad—an open-source treatment planning toolkit for educational purposes. *Med Phys Int*. 2018;6:119-127.
- Taasti VT, Bäumer C, Dahlgren CV, et al. Inter-centre variability of CT-based stopping-power prediction in particle therapy: survey-based evaluation. *Phys Imaging Radiat*. 2018;6:25-30.
- Paganetti H. Range uncertainties in proton therapy and the role of Monte Carlo simulations. *Phys Med Biol*. 2012;57:R99-R117.
- Taasti VT, Hong L, Deasy JO, Zarepisheh M. Automated proton treatment planning with robust optimization using constrained hierarchical optimization. *Med Phys*. 2020;47:2779-2790.
- Bortfeld T. Optimized planning using physical objectives and constraints. *Semin Radiat Oncol*. 1999;9:20-34.
- Hristov D, Stavrev P, Sham E, Fallone BG. On the implementation of dose-volume objectives in gradient algorithms for inverse treatment planning. *Med Phys*. 2002;29:848-856.
- Ghobadi K, Ghaffari HR, Aleman DM, Jaffray DA, Ruschin M. Automated treatment planning for a dedicated multi-source intracranial radiosurgery treatment unit using projected gradient and grassfire algorithms. *Med Phys*. 2012;39:3134-3141.
- Yao R, Templeton A, Liao Y, Turian J, Kiel K, Chu J. Optimization for high-dose-rate brachytherapy of cervical cancer with

adaptive simulated annealing and gradient descent. *Brachytherapy*. 2014;13:352-360.

43. Gonzaga CC, Karas EW. Fine tuning Nesterov's steepest descent algorithm for differentiable convex programming. *Math Program*. 2013;138:141-166.
44. Kim D, Sra S, Dhillon IS. A non-monotonic method for large-scale non-negative least squares. *Optim Methods Softw*. 2013;28:1012-1039.
45. Gao W, Goldfarb D, Curtis FE. ADMM for multiaffine constrained optimization. *Optim Methods Softw*. 2020;35:257-303.
46. Latorre F, Eftekhari A, Cevher V. Fast and provable ADMM for learning with generative priors. *Adv Neural Inf Process Syst (NIPS)*. 2019;32. <https://proceedings.neurips.cc/paper/2019/hash/4559912e7a94a9c32b09d894f2bc3c82-Abstract.html>
47. Benning M, Knoll F, Schonlieb C-B, Valkonen T. Preconditioned ADMM with Nonlinear Operator Constraint: IFIP Conference on System Modeling and Optimization. 2015:117-126. <https://dblp.org/db/conf/ifip7/csmo2015.html>
48. Boyd S, Vandenberghe L. *Convex Optimization*. Cambridge University Press; 2004.
49. Parikh N, Boyd S. Block splitting for distributed optimization. *Math Program Comput*. 2014;6:77-102.
50. Sun R, Luo Z-Q, Ye Y. On the expected convergence of randomly permuted ADMM. *arXiv:1503.06387*. 2015. <https://opt-ml.org/oldopt/opt15/cfp.html>
51. Deng W, Lai M-J, Peng Z, Yin W. Parallel multi-block ADMM with  $o(1/k)$  convergence. *J Sci Comput*. 2017;71:712-736.
52. Mihaljčić K, Zhu M, Ye Y. Managing randomization in the multi-block alternating direction method of multipliers for quadratic optimization. *Math Program Comput*. 2021;13:339-413.
53. Dong P, Lee P, Ruan D, et al.  $4\pi$  Noncoplanar stereotactic body radiation therapy for centrally located or larger lung tumors. *Int J Radiat Oncol Biol Phys*. 2013;86:407-413.
54. Zarepisheh M, Li R, Ye Y, Xing L. Simultaneous beam sampling and aperture shape optimization for SPORT. *Med Phys*. 2015;42:1012-1022.

**How to cite this article:** Fu A, Taasti VT, Zarepisheh M. Distributed and scalable optimization for robust proton treatment planning. *Med Phys*. 2022;1-10. <https://doi.org/10.1002/mp.15897>

## APPENDIX

We are interested in solving problem 8:

$$\text{minimize } \sum_{i=1}^m w_i \|a_{s,i}^T x_s - p_i\|_2^2 + \frac{\rho}{2} \|x_s - z^{(k)} + y_s^{(k)}/\rho\|_2^2 \quad (\text{A1})$$

with respect to  $x_s \in \mathbf{R}^n$ , where  $z^{(k)} \in \mathbf{R}_+^n$ ,  $y_s^{(k)} \in \mathbf{R}^n$ , and  $\rho > 0$  is a parameter. This is an unconstrained least-squares problem. Expand the objective function out to get

$$\begin{aligned} & \sum_{i=1}^m w_i \|a_{s,i}^T x_s - p_i\|_2^2 + \frac{\rho}{2} \|x_s - z^{(k)} + y_s^{(k)}/\rho\|_2^2 \\ &= \|\tilde{A}_s x_s - \tilde{p}\|_2^2 + \left\| \sqrt{\frac{\rho}{2}} x_s - \sqrt{\frac{\rho}{2}} (z^{(k)} - y_s^{(k)}/\rho) \right\|_2^2 \\ &= \left\| \begin{pmatrix} \tilde{A}_s \\ \sqrt{\frac{\rho}{2}} I \end{pmatrix} x_s - \begin{pmatrix} \tilde{p} \\ \sqrt{\frac{\rho}{2}} (z^{(k)} - y_s^{(k)}/\rho) \end{pmatrix} \right\|_2^2, \end{aligned} \quad (\text{A2})$$

where we define

$$\tilde{A}_s := \text{diag}(\sqrt{w}) A_s, \quad \tilde{p} := \text{diag}(\sqrt{w}) p. \quad (\text{A3})$$

Here  $\text{diag}(\sqrt{w})$  is a diagonal matrix with  $(\sqrt{w_1}, \dots, \sqrt{w_m})$  on the diagonal.

By the normal equations, a solution  $x_s^{(k+1)}$  of problem 8 must satisfy

$$\underbrace{(\tilde{A}_s^T \tilde{A}_s + \frac{\rho}{2} I)}_{B_s} x_s^{(k+1)} = \underbrace{\tilde{A}_s^T \tilde{p} + \frac{\rho}{2} (z^{(k)} - y_s^{(k)}/\rho)}_{c_s^{(k)}}. \quad (\text{A4})$$

Since  $B_s$  is positive definite (because  $\rho > 0$ ), such a solution always exists. One way to find  $x_s^{(k+1)}$  is to form the Cholesky decomposition of  $B_s$  and apply backward substitution to the triangular matrices. More precisely, we find an upper triangular matrix  $U \in \mathbf{R}^{n \times n}$  that satisfies  $B_s = U U^T$ . Then, we solve the system of equations  $U v = c_s^{(k)}$  for  $v$  and  $U^T x_s^{(k+1)} = v$  for  $x_s^{(k+1)}$ . These two solves can be done very quickly via backward substitution. Notice that  $U$  need only be computed once at the start of ADMM, as  $B_s$  remains the same across iterations, so subsequent solves of problem 8 simply require us to update  $c_s^{(k)}$  and solve the two triangular systems of equations.