Deep BOO: Automating Beam Orientation Selection in Intensity Modulated Radiation Therapy. Part I: Methodologies

Olalekan P. Ogunmolu*

Department of Radiation Oncology, UT Southwestern Medical Center,
2280 Inwood Road, Dallas, Texas , USA
olalekan.ogunmolu@utsouthwestern.edu

Abstract. Finding the optimal beam angle from a robot's tool frame (gantry angle) from which to aim radiation to cancer tumor(s) whilst simultaneously minimizing radiation intensity to organs-at-risks during intensity modulated radiation therapy is an exhaustive combinatorial search/human-based parameters-tuning process. So far, traditional optimization approaches for solving this problem have not yielded real-time feasible results. Therefore, we introduce techniques from (i) pattern recognition, (ii) monte carlo evaluations, (iii) game theory, and (iv) neuro-dynamic programming to find admissible policies for beamlets selection duringh IMRT treatment planning. In this sentiment, we optimize a deep neural network policy which guides a monte carlo simulation/evaluation of promising beamlets within the gantry's phase space. To obtain a pair of saddle point equilibrium strategy for the system, two fictitious players, within a zero-sum markov game framework, alternatingly play a best response strategy to their opponent's mixed strategy profile in games of neural self-play. At inference time, we run the trained policy on a patient's target volume geometry to predict feasible beam angles. This paper provides initial results on example coplanar beam orientation cases.

1 Introduction

Intensity-modulated Radiation Therapy (IMRT) is a state-of-the-art cancer treatment method that modulates the intensity of radiation delivered across a robot's beam eye-view and the geometric field-shape of the resulting beam in order to produce conformal dose distributions around a tumor. A good IMRT plan should simiulataneously maximize and minimize the intensities of radiation delivered to tumor(s) and healthy organs respectively, and it should produce sharp dose gradients at the transition between tumors and normal tissues. Before actual treatment, a doctor contours the *critical structures* (or tumors) and *organs-at-risk* (or OARs) within a *target volume* (the patient's CT scan), and then prescribes doses that must be delivered to the respective structures. Each beam is divided into beamlets, where each beamlet is aimed from the same angle; each beamlet in a beam may be of a different intensity from that of its neighbors – helping to achieve prescription dose for the target while sparing OARs. Generally, the radiation intensities are delivered from about 5-7 different beam orientations with multiple collimator units. The process of determining what intensity (influence) to give each beamlet so as to meet the doctor's objective is termed *fluence map optimization*.

Maximizing dose to tumors while simultaneously minimizing dose to critical structures poses a conflicted objective because it often occurs that the tumor(s) intersect(s) with some of the OARs. As such, finding good candidate beam angles in practice is often a trial-and-error process, with success depending on the dosimetrist/treatment planner's experience and the amount of time spent on tuning the angles. The process of selecting the best beam angles among all available beam orientations from which radiation is to be delivered in a treatment plan (TP), and optimizing the resulting chosen beams to fit a doctor's prescribed fluence profile is termed *beam orientation optimization* (BOO). Human intuition-based beam angle selection is laden with the subjectivity of the dosimetrist's judgment, not to mention the long time it takes in settling on a good set of candidate beam angles. Thus, having an automated framework for choosing beam angles is desirable since it will reduce the trial and error process, and save treatment time.

The second phase of the treatment planning process involves finding the curvature of beamlet intensities from which a beam can be shaped with respect to the geometrical shape and fluence across e.g. a high resolution multi-leaf collimator (MLC) field; this is so as to deliver a fixed or moving nonuniform photon beams or charged particles. Solving for good candidate beam angles is a nonconvex problem with many local minima (e.g. [1], [2]). It is at best a combinatorial optimization process – seeking to find an optimal set of beamlet candidates within the gantry of a robot's beam phase space (see figure above) – the workspace configuration and robot angle variables where the dynamics of the system is contained. When just the gantry (the robot's tool frame) of the linear accelerator (LINAC) machine is rotated in order to deliver beams to a patient (see figure above), we have *coplanar beams*, $i.e.$ all other angles are fixed while only the gantry rotates – resulting in a set of coplanar beams that are swept out by the gantry. In this paper, we focus on finding good coplanar beams in beam orientation optimization problem as more often than not, only coplanar beams are employed when delivering radiation.

IMRT TPS setup. ©Cyberknife.

Given the different ways a set of beamlets can be chosen and how a beam can be shaped with respect to the geometrical shape and fluence field, the dimensionality of the parameters to be optimized using traditional optimization methods increases to the extent that classical methods are no longer real-time feasible. This is coupled with the nonconvexity of the solution surface of the BOO process as the physics of dose deposition changes with each beam orientation. Therefore, we resort to an approximate dynamic programming formulation to derive approximately optimal control laws/policies in high dimensional state spaces. Particularly, we leverage on recent machine learning breakthroughs [3] to guide a monte carlo tree search [4–6] of promising beam angle candidates by rapidly exploring different parts the beam space.

1.1 Contributions

Finding the right combination of beam angles within the configuration space of the gantry is a hard computational problem given the high branching factor, tree depth, and complexity of finding a good known heuristic value function for non-terminal beam angle positions. This article provides the first comprehensive description of Monte-Carlo tree search within the framework of beam orientation optimization. It adds new pseudocode that transforms a BOO problem into a game planning strategy in order to recover an appropriate fluence for a patient during IMRT. With neural fictitious self-play, and monte carlo lookout strategy, we continually refine the predictions from a neural network policy so as to drive the weights of the network to a **saddle equilibrium**. This can be seen as a form of minimax zero-sum markov game e.g. [8] with the second player being randomly sampled previous iterations of the current neural network. To this end,

- we device a deep neural network to model the nonlinear dynamical system and to generate control policies that guide MCTS simulations for two markov decision process (MDP) players
- the MCTS tree lookout strategy guides transition from one beam angle set to another during each episodic setting for either player
- in a zero-sum two-player markov game of perfect information, we seek to find an alternating best response strategy to the current player's average strategy so as to drive the weights of the network policy toward an approximate **saddle equilibrium**
- this aids the network in finding an approximately optimal beam angle candidate set that meets the doctor's dosimetric requirements.

1.2 Related Work

There have been attempts at providing a mathematical formulation for the choice of feasible beam angles from which a good clinical plan can be achieved. For example, in [1], the author locally tune a set of beam angles within a beam's continuous state space by using techniques from linear programming duality theory and gradient descent. They reported that the BOO problem for a simple 2D pancreatic case has many local minima (≈ 100) when the local optimization is warm-started from 100 different beam angles.

Building on Craft's work in [1], Bertsimas et. al [9] resolved to a two meta-step algorithm: the BOO problem is solved by dynamically selecting beam angle candidates (based on the linear accelerator machine precision) within the continuous beams' phase space via a local minimum search with gradient descent, and then exploring a different portion of the solution space by taking finite steps of simulated annealing. Bertsimas et. al [9] define a linear programming problem with constraints that capture a doctor's preference for dose delivery. While this approach takes advantage of global and local search with improvements in solutions obtained from equispaced angles, it has the drawback of presenting the objective function as convex and assuming the doctor's preferences can be represented as a set of linear constraints.

Jia et. al [10] split the BOO problem into two consecutive phases: first, by progressively identifying non-consequential beam angles through the evaluation of a combinatorial

objective function, and secondly, by optimizing the fluence on beam angles that are assigned a high confidence by a dosimetric objective component in their formulated objective. Heuristic search strategies have also previously developed e.g. [11–13]. Other lines of work have treated IMRT treatment planning as an inverse optimization problem, with techniques ranging from adaptive l_{21} optimization [10], mixed integer linear programming [14–16] and simulated annealing [17, 18].

1.3 Notations and Definitions

First, let us formalize definitions and notations to guide our problem formulation. The state of the dynamical system, will be denoted by $s \in \mathcal{S}$, and it is to be controlled by a discrete action $a \in \mathcal{A}$. States evolve according to the (unknown) dynamics $p(s_{t+1}|s_t, a_t)$,

Notation	**Definition/Examples**	**Notation**	**Definition/Examples**
m	dimensionality of a node's beam set, e.g. $m = 5$	n	dimension of discretized beam angles, e.g. $n = 180$ for 4° angular resoultion
$\boldsymbol{\Theta}$	discretized beam angle set e.g. equally spaced angles between 0° and 360°, spaced apart at 4°	$a_t \in \mathcal{A}$	control or action, $a_t \in \mathcal{A}$ at time step $t \in [1, T]$ used in determining the probability of transitioning from a beam angle subset to another within $\boldsymbol{\Theta}$
$\theta^j \subseteq \boldsymbol{\Theta}$	beam angles selected from $\boldsymbol{\Theta}$ e.g. $\theta_k \in \mathbb{R}^m$	$s_t \in \mathcal{S}$	markovian system state at time step, $t \in [1, T]$ e.g. patient contour, beam angle candidates; dimensionality $2,727,936$ to $3,563,520$
γ	discount factor e.g. 0.99	$\boldsymbol{f_\psi}$	parametric function approximator (deep neural network policy) for state s_t
$v_\psi(s)$	value estimate of state, s_t, as predicted by $\boldsymbol{f_\psi}$	$\boldsymbol{p(s)}$	probability distribution over current state, s generated by neural network policy
$Q(s, a)$	action-state values that encode the "goodness" of a beam-angle set, $\theta_k \in \mathbb{R}^m$, where m is the number of beams considered for a fluence generation, e.g. $m = 5$	$\boldsymbol{B}_{X_t}$	a concatenation of beams in consideration at time step, t, as a block of beams being fed to the neural network policy
$\mathcal{D}_{ij}(\theta_k)$	dose matrix containing dose influence to voxel i from beam angle, θ_k, $\forall\, k \in \{1, 2, \ldots, n\}$ where n is range of the beam set $\mathcal{B}$	$\boldsymbol{D}_t$	dose mask for target volume in consideration at time step , t

Table 1: Table of notations commonly used in this article

which we want to learn. We pose the learning problem within a discrete-time finite-time horizon, T, which is without a discount factor γ.

A beam angle combination search task in this setting can be defined by a reward function $r(s_t, a_t)$ which can be found by recovering a policy $p(a_t|s_t;\psi)$ which specifies a distribution over actions conditioned on the state parameterized by some vector ψ. The parameters, ψ, might correspond to the weights of a neural network or parameters of a Gaussian process, for example. Recovering the optimal weights may consist of the maximization problem,

$$\psi^{\star} = \arg\max_{\psi} \sum_{t=1}^{T} \mathbb{E}_{(s_t,a_t)\sim p(s_t,a_t|\psi)} \left[r(s_t, a_t)\right],$$

where the optimization problem is to determine an optimal vector $\psi^{\star}$ that maximize the sum of the respective instantaneous rewards $\sum_t r(s_t, a_t)$ of the policy, $p(a_t|s_t, \psi)$. For clarity, we will denote the action conditional $p(a_t|s_t, \psi)$ as $\boldsymbol{\pi}_{\psi}(a_t|s_t)$.

2 Methods and Materials

Our design philosophy draws inspiration from approximate dynamic programming, optimal control theory, and game theory. For games with perfect information, there is an optimal value function, $v^{\star}(s)$, that decides the outcome of the game for every possible state under perfect play ($i.e.$ a game with an optimal min-max state-action value function, $Q(s, a)$, produced by a policy $\boldsymbol{\pi} = \{\boldsymbol{\pi}_{min}, \boldsymbol{\pi}_{max}\}$). It follows that one can devise a planning strategy that guides the search for optimistic beam angle configurations within the global beam angle space by using a probability distribution, $p(s, a)$, over a set of deterministic *pure strategies* for the tree [7].

2.1 Overview

The search for an *approximately* optimal beam angle is performed by optimizing a set of deep neural network parameters, ψ which are then used to repeatedly guide an MCTS simulation of 'best-first' beam angle combinations for a sufficiently long number of iterations. Essentially, the MCTS performs a sparse lookout simulation, recursively expanding the sub-nodes under a tree's root node, and selectively adjusts beam angle(s) that contribute the least to an optimal fluence within a beam angle set. Successor nodes beneath a terminal node are **approximated** with a value, $v(s)$, to assure efficient selectivity.

We model the decision process for optimal beam selection by maintaining a probability distribution over the set of possible states, based on a set of observations and observation probabilities for the underlying Markov Decision Process. The state of the system, s_t, is markovian containing the concatenation of the patient ct mask as well as a subset of the beams in consideration at the current time step t. The policy that guides the search for the optimal beam angle selection is computed by a deep neural network policy, consisting of multiple residual blocks fashioned similar to modern neural network architectures [19, 20]. Each beam angle is represented as gantry angle images (see Fig. 1). Depending on the patient in question, the state at time step, t, varies between $2,700,000 \sim 3,600,000$ in number of parameters.

(a) angle 10° (b) angle 82° (c) angle 166°

Fig. 1: Example 2D gantry angle representations

2.2 Data Preprocessing

This section describes the data preparation for the masks that we use in training the network.

Patient Mask. We obtained 77 anonymized patient CT scans and associated data from our clinic. The scans relate to prostate cases used in previous IMRT cancer treatment planning. The prostate data contain scans of six organs, namely the patient's body, bladder, left femoral head, right femoral head, rectum and planning target volume (PTV) or tumor. Each patient's scan is represented in 3D as $N \times W \times H$, where N is the total number of slices, W is the image width and H is the image height. We resized each slice within each patient's ct mask, $\boldsymbol{D}$, to a square-shaped 2D matrix of size $N \times 64 \times 64$, where N is the number of slices in the mask (note that this varies per patient, but the network is so designed to handle different number of slices).

Beam Orientation Representation: We discretize the gantry angle space at a 4° regular grid from 0° through 360°. Since the beam angles and patient's geometry are spatially related, we use deep convolutional networks to encode the spatial relationship between the pose of the patient from each beam's eye view. We generate 3D images that represent the orientation of the gantry with respect to the patient for each discretized beam angle as shown in (Fig. 1).

2.3 Neural Network Architecture

In addition to the resized masks, $\boldsymbol{D}$, we define five feature planes, $\boldsymbol{X}_t$ as a block of beam configurations: $\boldsymbol{B}_{\boldsymbol{X}_t}$, denotes the beam angles that generate the current fluence. For example, if we are considering five beams for the geometric shape for the fluence, $\boldsymbol{B}_{\boldsymbol{X}_t}$ would contain the 3D RGB images of the beams being considered at time step t. We then augment the state with a memory of five previously used beam blocks, $\{\boldsymbol{B}_{\boldsymbol{X}_t}, \ldots, \boldsymbol{B}_{\boldsymbol{X}_{t-5}}\}$, during the search process; we make this augmentation due to the partial observability of the patient-gantry setup from a single set of features, in order to mitigate the uncertainty of decisions under our POMDP formulation.

The dose masks and beam blocks are as shown in Fig. 2 so that the input planes to the network are sized as $T \times N \times H \times W$; T is the total number of input planes ($T = 6$ structures $+ 5$ beams $+ 5 \times 5$ regressed beams). Thus, the input to the network are arranged as: $s_t = [\boldsymbol{D}_t, \boldsymbol{B}_{\boldsymbol{X}_t}, \boldsymbol{B}_{\boldsymbol{X}_{t-1}}, \boldsymbol{B}_{\boldsymbol{X}_{t-2}}, \boldsymbol{B}_{\boldsymbol{X}_{t-3}}, \boldsymbol{B}_{\boldsymbol{X}_{t-4}}, \boldsymbol{B}_{\boldsymbol{X}_{t-5}}]$. To address robust learning of patient geometry and beam angle orientations, we use a modern neural

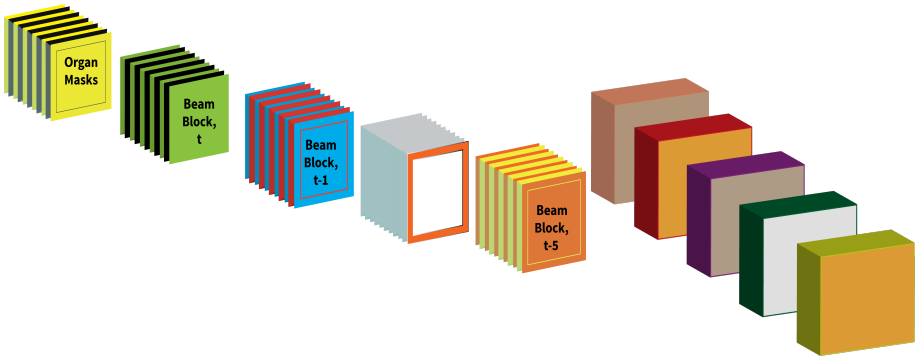

Fig. 2: [Left]: Arrangement of the target volume masks with the beam angles before feeding the input planes to the residual tower neural network. The first six planes (top-most mask of left figure) contain the delineated organs and the PTV. This is concatenated with a block of m beams from the current time step, regressed to the previous 5 time steps (here, 5 was heuristically chosen). [Right] Each beam angle in a beam block is represented as shown. Together with the target volume, these form an input plane of size $36 \times N \times W \times H$ to the policy/value neural network tower of residual blocks.

network architecture with many residual blocks [20]; each layer of the network fits a residual nonlinear mapping to its input data; we end up with a deeply stacked network whose input features, s_t, are processed by 34 residual blocks described as follows: (i) a 3D convolution with $64 \times l$ filters, a square kernel of width 7, and double strided convolutions, where l is the depth of the stack in the network; (ii) a 3D batch normalization layer [22]; (iii) nonlinear rectifiers [23]; (iv) a 3D convolution of $64 \times l$ filters; (v) a 3D batch normalization layer; (vi) a skip connection from the input to the block, in order to facilitate efficient gradients' propagation; and (vii) nonlinear rectifiers.

We then split the output of the network into two heads: (i) the first head is a probability distribution over which angle in the current beam block contributed the least to an optimal fluence cost at the current tiome step, while (ii) the second head estimates the *value* (in an *ADP sense*) of the tree beneath the terminal state This probabilty head is comprised of two residual blocks, each containing: (i) a 3D convolution of $64 \times l$ filters, followed by a square kernel of size 1, and a single strided convolution; (ii) a 3D batch normalization layer; (iii) nonlinear rectifiers; (iv) a 3D convolution of $64 \times l$ filters (v) a 3D batch normalization layer; (vi) a skip connection from the input to the block (vii) nonlinear rectifiers (viii) a fully connected layer that maps the resulting output to the total number of discretized beam angle grids, 180; and (ix) a softmax layer then maps the neuron units to logit probabilities $p_i(s|a)$ for all beam angles.

The value head applies the following transformation modules (i) a 3D convolution of $64 \times l$ filters, a square kernel of size 1, and a single strided convolution; (ii) a 3D batch normalization layer; (iii) nonlinear rectifiers; (iv) a second 3D convolution of $64 \times l$ filters; (v) a 3D batch normalization layer; (vi) a skip connection from the input to the block (vii) nonlinear rectifiers and a sequential layer consisting of:

- a linear layer that maps the output of the above connections to a 512-layer hidden unit
- a linear layer that maps the output of the above connections to a 256 hidden unit followed by rectified nonlinearities
- a linear that maps to a single value again followed by rectified nonlinearities
- a tanh nonlinearity that maps the output to the closed interval $[-1, 1]$.

The parameters of the neural network were initialized using the initialization method proposed in [24]. Since decisions about what beam angle combination is optimal at the current time step is made under uncertainty, we use a *probability model* to choose among **lotteries**, where the lotteries in this case are probability distributions over a set of beamlets to choose. Each state during our learning process is constructed by appending the beam block at the current time step to a history of beam blocks for the previous five time steps. We append a new beam block to the current set of beam blocks and target volume dose mask (see Fig. 2) with a FIFO policy so that we minimize the partial observability of our MDP system. When we transition to a new state, the beam block that has been in the state set for the longest time ($i.e.$ at the **head** of the queue) is deleted first, and the new state's beam block is enqueued at the tail as in a **queue** data structure. Let us now introduce some definitions.

Definition 1. *A **beam block** is a concatenation of beamlets, $\{\theta_1, \theta_2, \ldots, \theta_m\}$, at time step t, as a tensor of dimension $m \times C \times H \times W$ (see Fig. 2) that together with the patient's ct mask form the state s_t at time step t.*

Definition 2. *Every **node**, $\boldsymbol{x}$ has the following fields: (i) a pointer to the parent that led to it, $\boldsymbol{x}.p$; (ii) the beamlets, $\boldsymbol{x}_b$, stored at that node where $b = \{1, \ldots, m\}$; (iii) a set of move probabilities prior, $p(s, a)$; (iv) a pointer, $\boldsymbol{x}.r$, to the reward, r_t, for the state s_t (v) a pointer to the state-action value $Q(s, a)$ and its upper confidence bound $U(s, a)$ (1) (vi) a visit count, $\mathbb{N}(s, a)$, that indicates the number of times that node was visited; and (vii) a pointer $\boldsymbol{x}.child_i$ to each of its children nodes.*

In order for the state definition to capture as much information as possible under uncertainty, we define the state broadly enough to capture all subjective unknowns that might influence the payoff/reward to be received by an agent (which is a rational decision maker). We want an adaptive allocation rule for determining the transition between states such that as the search process progresses, and achieves convergence as the simulation grows, the worst possible bias is bounded by a quantity that converges to zero. We leverage on the *upper confidence bound* algorithm of Agrawal [25] that assures an asymptotic logarithmic regret behavior. Since we do not know what node may yield the best bandit, a player might be biased towards always selecting the beams set with the maximum value. Therefore, we attach a regret term, $U(n(s))$ to the $Q(s, a)$-value so as to ensure the optimal beam does not evade the simulation $i.e.$ $Q(s, a) - U(n(s)) \leq Q(s, a) \leq Q(s, a) - U(n(s))$; the width of this confidence bound guides the exploration strategy for states that are momentarily unpromising in values but may later emerge as promising states as the number of games evolve in length.

Definition 3. *We define an **upper confidence bound**, $U(s, a)$, on $Q(s, a)$ that adds an exploration bonus that is highest for seldomly visited state-action pairs so that the tree*

policy selectes the action $a^\star$ that maximizes the augmented value:

$$\bar{Q}(s,a) = Q_j(s,a) + c\sqrt{\frac{2\ ln\ n(s)}{\mathbb{N}(s,a)}}, \quad \textit{where } a^\star = \arg\max_a \bar{Q}(s,a). \tag{1}$$

*$Q(s,a)$ is the highest average observed reward from node j – encouraging exploitation of the current node, and ln $n(s)$ is the natural logarithm of the total number of roll-outs through state s [26]. The second term in the (1) encourages exploration of other beam angles and c is a scalar exploration constant. Note that (1) is a a version of the **UCB1** algorithm [6]. This estimates the expected reward of a set of beam canditates once we compute its upper confidence.*

2.4 Fluence Map Optimization

Suppose $\mathcal{X}$ is the total number of discretized voxels of interest (VOI's) in a target volume, and $\mathcal{B}_1 \cup \mathcal{B}_2 \cup \ldots \cup \mathcal{B}_n \subseteq \mathcal{B}$ represents the partition subset of a beam $\mathcal{B}$, where n is the total number of beams from which radiation would be delivered. Suppose we let $\mathcal{D}_{ij}(\theta_k)$ be the matrix that describes each dose influence, d_i, delivered to a discretized voxel, i, in a volume of interest, VOI_h $(h = 1, \ldots, \mathcal{X})$, from a beam angle, θ_k, which is within the beamlet $k \in \{1, \ldots, n\}$, then it follows that one can compute the matrix $\mathcal{D}_{ij}(\theta_k)$ by calculating each d_i for every bixel, j, at every φ°, $j = 1, \ldots, k$ resolution. Doing this, we end up with a large but sparse matrix, $\mathcal{D}_{ij}(\theta_k)$, which consists of the dose to every voxel, i, incident from a beam angle, θ_k at every $360^\circ/\varphi^\circ$ (in our implementation, we set φ to 4°).

Let us introduce a decision variable, x_j, which represents the intensities of beamlets, $j \in \mathcal{B}$; then it is trivial to find the dose influence, d_i, that relates the bixel intensities, x_j, to the voxels of interest, VOI_h. The fluence problem aims to find the values of x_j for which the d_i to the tumor is maximized, while simultaneously minimizing the d_i to critical structures. These objectives are conflicted by definition, and it is typical for dosimetrists to spend hours before finding a good set of candidate beams. For the voxels in all OARs, a weighted quadratic objective minimizes the l_2 distance between a pre-calculated dose $\boldsymbol{Ax}$, and a doctor's prescribed dose, $\mathbf{b}$, while a weighted quadratic objective maximizes the l_2 distance between $\boldsymbol{Ax}$ and $\mathbf{b}$. The pre-calculated dose term is given by $\boldsymbol{Ax} = \{\sum_s \frac{w_s}{v_s}\mathcal{D}^s_{ij}\boldsymbol{x}_s \,|\, \mathcal{D}_{ij} \in \mathbb{R}^{n\times l}, n > l\}$, which is a combination of the dose components that belong to OARs and those that belong to PTVs. $w_s = \{\underline{w}_s, \bar{w}_s\}$ are the respective underdosing and overdosing weights for the OARs and PTVs, and v_s represents the total number of voxels in each structure. The cost function is

$$\frac{1}{v_s}\sum_{s\in\text{OARs}} \|(b_s - \underline{w}_s\mathcal{D}^s_{ij}\boldsymbol{x}_s)_+\|_2^2 + \frac{1}{v_s}\sum_{s\in\text{PTVs}} \|(\bar{w}_s\mathcal{D}^s_{ij}\boldsymbol{x}_s - b_s)_+\|_2^2 \tag{2}$$

where the underdosing weights are typically set as $\underline{w}_s = 0$ since it is desirable to deliver as little dose as possible to critical structures, while the overdosing weights are chosen to deliver the prescribed dose to the tumor; $(\cdot)_+$ denotes a Euclidean projection onto the nonnegative orthant $\mathbb{R}_+$. We can succintly rewrite the objective function above, subject

to nonnegative bixel intensity constraints, as

$$\text{minimize } \frac{1}{2}\|A\mathbf{x}-\mathbf{b}\|_2^2 \quad \text{subject to } x \geq 0,$$

with the attraction of being differentiable and convex, penalizing dose volumes that exceed a doctor's prescribed constraint. The Lagrangian becomes

$$L(\boldsymbol{x},\boldsymbol{\lambda}) = \text{minimize } \frac{1}{2}\|A\mathbf{x}-\mathbf{b}\|_2^2 - \boldsymbol{\lambda}^T\boldsymbol{x}.$$

The above problem can be solved with dual gradient descent (DGD) but DGD has the drawback that the primal and dual updates are not robust to objective's constraints [27]. The alternating direction method of multipliers (ADMM) [27] tackles the robustness problem by adding a quadratic penalty term to the Lagrangian and alternatingly updating the $\boldsymbol{x}$ and $\boldsymbol{\lambda}$ variables in a "broadcast and gather" process. This turns out to be attractive since we will be solving a large-scale learning problem for the optimal beam angle set combination. Introducing an auxiliary variable $\mathbf{z}$, the above is transformed into the following

$$\min_{\boldsymbol{x}} \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\mathbf{b}\|_2^2, \quad \text{subject to } \boldsymbol{z}=\boldsymbol{x},\ \boldsymbol{z}\geq 0.$$

so that the Lagrangian can be written as,

$$\min_{\boldsymbol{x},\boldsymbol{z}} \frac{1}{2}\|\boldsymbol{A}\boldsymbol{x}-\mathbf{b}\|_2^2 - \boldsymbol{\lambda}^T(\boldsymbol{z}-\boldsymbol{x}) + \frac{\rho}{2}\|\boldsymbol{z}-\boldsymbol{x}\|_2^2, \tag{3}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^n$ is a multiplier and $\rho > 0$ is an ADMM penalty parameter. The $\boldsymbol{x}$ subproblem to (3) yields

$$\min_{\boldsymbol{x}} \quad \frac{1}{2}\boldsymbol{x}^T(\boldsymbol{A}^T\boldsymbol{A}+\rho\boldsymbol{I})\boldsymbol{x} + (\boldsymbol{\lambda}^T - \boldsymbol{A}^T\boldsymbol{b} - \rho\boldsymbol{z}^T)\boldsymbol{x},$$

so that the $\boldsymbol{x}$-update (due to the convex quadratic nature of the problem),

$$\boldsymbol{x}^{k+1} = \left(\boldsymbol{A}^T\boldsymbol{A}+\rho\boldsymbol{I}\right)^{-1}\left(\boldsymbol{A}^T\boldsymbol{b}+\rho\boldsymbol{z}^k-\boldsymbol{\lambda}^k\right). \tag{4}$$

Similarly, the $\boldsymbol{z}$-update for (3) can be found by the z-minimization subproblem

$$\min_{\boldsymbol{z}} -\boldsymbol{\lambda}^T\boldsymbol{z} + \frac{\rho}{2}\|\boldsymbol{z}-\boldsymbol{x}\|_2^2 := \min_{\boldsymbol{z}} \frac{\rho}{2}\|\boldsymbol{z}-\boldsymbol{x}-\frac{1}{\rho}(\boldsymbol{\lambda})\|_2^2.$$

Using the soft-thresholding operator, $S_{\boldsymbol{\lambda}/\rho}$, we find that

$$\boldsymbol{z}^{k+1} = S_{\boldsymbol{\lambda}/\rho}\left(\boldsymbol{x}^{k+1}+\boldsymbol{\lambda}^k\right), \tag{5}$$

where $S_{\boldsymbol{\lambda}/\rho}(\tau) = (x-\boldsymbol{\lambda}/\rho)_+ - (-\tau-\boldsymbol{\lambda}/\rho)_+$. $\boldsymbol{\lambda}$ is updated as

$$\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k - \gamma(\boldsymbol{z}^{k+1}-\boldsymbol{x}^{k+1}), \tag{6}$$

where γ is a parameter that controls the step length. The inverse operation in (4) can be carried out with any iterative solver e.g. conjugate gradient. We use an over-relaxation parameter, $\alpha^k = 1.5$, and set the quadratic penalty to $\rho = 1.5$, in the $\boldsymbol{z}$ and $\boldsymbol{\lambda}$ updates: $\alpha^k \boldsymbol{A}\, \boldsymbol{x}^{k+1} - (1-\alpha^k)\boldsymbol{z}^k$. The stopping criterion is met when the primal and dual residuals are sufficiently small, $i.e.$,

$$r^k = \|\boldsymbol{x}^k - \boldsymbol{z}^k\|_2 \leq \epsilon^{\text{pri}} \;\text{ and }\; s^k = \| - \rho\,(\boldsymbol{z}^{k+1} - \boldsymbol{z}^k)\|_2 \leq \epsilon^{\text{dual}},$$

with,

$$\epsilon^{\text{pri}} = \sqrt{\rho}\epsilon^{\text{abs}} + \epsilon^{\text{rel}} \max\{\|\boldsymbol{x}^k\|_2, \| - \boldsymbol{z}\|_2\}, \text{ and}$$
$$\epsilon^{\text{dual}} = \sqrt{n}\epsilon^{\text{abs}} + \epsilon^{\text{rel}}(\rho\boldsymbol{\lambda}^{\boldsymbol{k}}),$$

where $\epsilon^{\text{pri}} > 0$, $\epsilon^{\text{dual}} > 0$ are the primal and dual feasibility tolerances for the primal and dual feasibility conditions (see [27, §3.3]). We set $\epsilon^{\text{abs}} = 10^{-4}$ and $\epsilon^{\text{rel}} = 10^{-2}$.

2.5 Game Tree Simulation

Our tree-search policy is based on a sparse look-ahead search strategy for large state-space Markov Decision Problems (MDPs). The game simulation is one of **perfect recall** $i.e.$ each node's current information state implies knowledge of the ancestral node sequences that led to it.

The algorithm can be seen as a set of sequential simulation of different beam angle combinations guided by probabilities obtained from a two-player zero-sum game of neural fictitious self-play (FSP); the network roll-out policy is used to guide the search for a *best-first* set of beam angle candidates. The best-first leaf node encountered is given by the child node with the highest reward along a tree that is expanded. What follows is a set of sampling-based lookout algorithm that focuses learning on regions of the state space that are likely to have a good fluence profile and computes action-value pairs based on selectively sampled angles.

We randomly sample beam angles within the gantry angle space by carrying out a lookahead search at fixed depth; we restrict sampling to 90 beam grid in $\boldsymbol{\Theta}$ at every stage of our simulation. A partial tree of beam angle candidates is randomly constructed, starting from a root node, and we progressively add child nodes to the current node in a ***selection*** process guided by move probabilities, $p(s, a)$. At the first iteration, the subset of beam angles are randomly sampled from $\boldsymbol{\Theta}$. In our expansion phase (see algorithm 1), to prevent angles from being too close to one another, we define a minimum pairwise distancem, $\bar{d}_i \in \mathbb{R}^+$ between the beamlets in a beam block, given by $\|\theta_i - \theta_j\| \geq \bar{d}_i, \forall\{j \in m \setminus i\}$ where we set $\bar{d}_i = 20$. By repeatedly performing roll-outs, we maintain a history of state-action value pairs, stored along the edges of the tree. This aids faster convergence if the same state is encountered more than once: we can bias an action selection based on old actions that were chosen.

When we finish each simulation, a 'best move' for the current beam block is selected, as computed by the tree search; we exponentiate the move probabilities by a temperature slightly larger than unity to encourage diversity in early play; specifically, we compute

$$p(a|s_0;\psi) = \frac{N(s_0,a)^{1/\tau}}{\sum_b N(s,b)^{1/\tau}},$$

where τ is the temperature factor that diversifies the move probabilities. The UCT [6] algorithm applied to optimal beam angle selection is presented in algorithm 1.

Definition 4. *A minimizer player is at a **terminal state** if the FMO cost for the beams at a leaf node is greater than the cost of the beams at its direct ancestor node.*

Definition 5. *A maximizer player has reached a **terminal state** if the FMO cost for the beams at a leaf node is less than the FMO cost for the beams at its direct ancestor node.*

Definitions § 4 and § 5 help preserve the neighborhood search for the beam improvement tuning as specified earlier. When angles are at the edges $i.e.$ 0° or 360° and an angle change outside the range $0 \leq \theta \leq 360$ is recommended, we "wrap" around to enforce cyclicity. The index of the angle to select is determined by taking an argmax of the mixed strategy produced by the tree policy $\boldsymbol{\pi}_{\psi_{t-1}}$; this policy is constantly being optimized in a separate thread. Similarly, the direction in which to modify the angle is determined by an argmax of the third head of the tree policy $\boldsymbol{\pi}_{\psi_{t-1}}$ as discussed in § 2.3. The **EXPANSION** and **DEFAULTPOLICY** procedures of the MCTS simulation in algorithm 1 can be seen as a form of Add/Drop simulated annealing [18]. The **BESTCHILD** procedure compares the quality of all beam angle sets traversed as the root node is expanded to one or more child nodes; the greedy approach of selecting the beam angle set with the highest reward results in a locally optimal set of beam angles candidates at each iteration of running a game.

2.6 Self-Play Neuro-Dynamic Programming

This section describes the approximate dynamic programming scheme and self-play formulation for generating saddle point equilibrium strategies for an *approximately* optimal outcome (value) by playing a weakened fictitious self-play (FSP) between two neural network behavior strategies. Weakened fictitious self-play applies machine learning to determine an *approximate* best response strategy to an opponent's mixed strategy in games of self-play. An agent continually plays a game against an opponent and the outcome of the game is given by averaging the outcome of all individual plays.

We apply FSP in a machine learning framework to a game's strategy profile so that each player plays an *approximate* best response strategy to their opponent's mixed strategy at each iteration. In our formulation, each player does not necessarily know the strategy of its opponent ahead of time $i.e.$ their security levels do not necessarily coincide. To ensure that equilibrium can be found within pure strategies, we let one player act after observing the decision outcome of the other player. The game thus has a **dynamic** property since the strategy of the successive acting player explicitly depends on that of the first player [28, pp.22].

Each player's action is governed by a mixed strategy – obtained by adding a Gaussian random walk sequence with standard deviation 2 to the prior probability distribution predicted by the neural network policy or computed by the tree policy; this is then normalized by the sum of the resulting noised distribution. We do this to encourage further exploration and to transform each player's pure strategy to a mixed strategy. As such, a player's mixed strategy is simply a random variable whose values are the player's

Algorithm 1 Policy-Guided Monte Carlo Tree Search

function MCTS(s_0)
  with state, s_0, create a root node, $\boldsymbol{x}_0(s_0)$
  while search time $<$ budget **do**
    $\bar{\boldsymbol{x}} \leftarrow$ **EXPAND_POLICY**($\boldsymbol{x}_0$)
    $\bar{\boldsymbol{x}}.r \leftarrow$ **FMO_POLICY**
    BACKUP($\bar{\boldsymbol{x}}, \bar{\boldsymbol{x}}.r$)
  end while
  return BEST_CHILD($\boldsymbol{x}_0, c$)
end function

function EXPAND_POLICY($\boldsymbol{x}$)
  while $\boldsymbol{x}$ is nonterminal **do**
    if $\boldsymbol{x}$ is not fully expanded **then**
      return EXPAND ($\boldsymbol{x}$)
    else
      $\boldsymbol{x} \leftarrow$ **BEST_CHILD**($\boldsymbol{x}, c$)
    end if
  end while
  return $\boldsymbol{x}$
end function

function FULLY_EXPANDED($\boldsymbol{x}$, $\boldsymbol{d}$)
  $d_i \leftarrow$ pairwise_distance($\boldsymbol{x}.s$)
  min_elem $\leftarrow \min(\boldsymbol{d})$
  if min_elem $<$ $\boldsymbol{d}$ **then**
    return True
  else
    return False
  end if
end function

function FMO_POLICY($\boldsymbol{x}$)
  find optimal fluence objective, $h^\star(\boldsymbol{x}(s)|\cdot)$
  return $r = -h^\star(\boldsymbol{x}(s)|\cdot)$
end function

function EXPAND($\boldsymbol{x}$)
  create $\bar{\boldsymbol{x}}$ with tree policy $\pi_{\psi_{t-1}}$
  while not $\bar{\boldsymbol{x}}$ in children of $\boldsymbol{x}_0$ **do**
    add child $\bar{\boldsymbol{x}}$ to $\boldsymbol{x}$
  end while
  return $\bar{\boldsymbol{x}}$
end function

function BACK_UP($\boldsymbol{x}, c$)
  while $\bar{\boldsymbol{x}}$ is not null **do**
    $N(\bar{\boldsymbol{x}}) \leftarrow \bar{\boldsymbol{x}} + 1$
    $Q(\bar{\boldsymbol{x}}) \leftarrow Q(\bar{\boldsymbol{x}}) + \bar{\boldsymbol{x}}.r$
    $\bar{\boldsymbol{x}} =$ parent of $\bar{\boldsymbol{x}}$
  end while
end function

function SELECT_MOVE($\boldsymbol{x}, c$)
  if p_1 to play **then**
    return $\arg\max_{\bar{\boldsymbol{x}} \in \text{ children of } \boldsymbol{x}} Q(\bar{\boldsymbol{x}}) + c\sqrt{\frac{2 \ln n(\bar{\boldsymbol{x}}.s)}{N(\bar{\boldsymbol{x}}.s, a)}}$
  else
    return $\arg\min_{\bar{\boldsymbol{x}} \in \text{ children of } \boldsymbol{x}} Q(\bar{\boldsymbol{x}}) - c\sqrt{\frac{2 \ln n(\bar{\boldsymbol{x}}.s)}{N(\bar{\boldsymbol{x}}.s, a)}}$
  end if
end function

pure strategies. Players $p_1,$ and p_2's mixed strategies are independent random variables, repeatedly implemented during game simulations such that their actual decisions are based on chance events. However, as the number of times the game is played gets larger, the frequency with which different actions for p_1 and p_2 are chosen will converge to the probability distribution that characterize their random strategies [28, pp.24]. For a game Γ, suppose that $y = \{y_1, \ldots, y_m | \sum_{i=1}^{m} y_i = 1\}$ and $z = \{z_1, \ldots, z_n | \sum_{i=1}^{n} z_i = 1\}$ are the respective probability distributions for players p_1 and p_2, defined on the n and $m-$dimensional simplices respectively, the average value of the game will correspond to player p_1 minimizing a cost $\mathcal{J}(y, z) = y^T \Gamma z$ and player p_2 maximizing $\mathcal{J}(y, z)$.

The fictitious self-player samples from previous episodes of the game of self-play at each iteration k. For each ***pure strategy***, $p(s, a)$, predicted by the neural network policy, $\boldsymbol{f_\psi}$, each player makes a decision governed by a probability distribution on the space of its pure strategies. The opposing agent's policy is randomly sampled from a previous checkpoint of the training scheme. The goal of the two-player zero-sum neural fictitious self-play is to drive the network to an approximate saddle equilibrium, via experiential learning, thus guaranteeing that the realization of an approximate optimal value function for the optimal beam angle set that generates a desired fluence/DVH profile.

3 Conclusion

Beam orientation optimization is a key component of conformal IMRT radiotherapy TPS. It has a nonconvex solution surface given the way the dose coefficients can significantly alter based on the gantry angle from which beams are aimed to a target volume. As such, in modern clinics, this problem is typically solved with many hours of planning, hand-tuning and refinement – usually by experienced dosimetrists and radiation therapists.

Given the long time traditional optimization approaches take in solving this problem, we adopt a hybrid approach, leveraging on monte carlo simulation of games in high dimensional state-spaces, neuro-dynamic programming, convex optimization of fluence profiles of a target radiation, and game theory to arrive at good candidate beamlets. Our monte carlo tree search formulation is the first, to the best of our knowledge, that transforms the BOO problem into a monte carlo tree search strategy; and provides a pseudocode that shows how we implement the tree search to navigate the complex state space of respective beamlets. In the part II of this paper, we elucidate on the end-to-end optimization pipeline and provide our initial results based on a set of numerical simulations for select anonymized patients data in our clinic.

References

1. Craft, D.: Local beam angle optimization with linear programming and gradient search. Physics in Medicine & Biology **52**(7), N127 (2007) 2, 3
2. Södertröm, S., Brahme, A.: Optimization of the Dose Delivery In A Few Field Techniques Using Radiobiological Objective Functions. Medical physics **20**(4), 1201–1210 (1993) 2
3. LeCun, Y., Bengio, Y., Hinton, G.: Deep Learning. Nature **521**(7553), 436–444 (2015) 2
4. Gelly, S., Silver, D.: Monte-Carlo tree search and rapid action value estimation in computer Go. Artificial Intelligence **175**, 1856–1875 (2011) 2

5. Coulom, R.: Efficient Selectivity and Backup Operators in Monte-Carlo Tree Search. International Conference on Computers and Games (2006) 2
6. Kocsis, L., Szepesvári, C.: Bandit based Monte-Carlo Planning. European Conference on Machine Learning (2006) 2, 9, 12
7. Heinrich, J., Lanctot, M., Silver, D.: Fictitious self-play in extensive-form games. In: International Conference on Machine Learning, pp. 805–813 (2015) 5
8. Ogunmolu, O., Gans, N., Summers, T.: Minimax Iterative Dynamic Game : Application to Nonlinear Robot Control. IEEE International Conference on Intelligent Robots and Systems (2018) 3
9. Bertsimas, D., Cacchiani, V., Craft, D., Nohadani, O.: A Hybrid Approach To Beam Angle Optimization In Intensity-modulated Radiation Therapy. Computers & Operations Research **40**(9), 2187–2197 (2013) 3
10. Jia, X., Men, C., Lou, Y., Jiang, S.B.: Beam Orientation Optimization For Intensity Modulated Radiation Therapy Using Adaptive L2,1-minimization. Physics in Medicine and Biology **56**(19), 6205–6222 (2011) 3, 4
11. Bortfeld, T., Schlegel, W.: Optimization of beam orientations in radiation therapy: Some theoretical considerations. Physics in Medicine & Biology **38**(2), 291 (1993) 4
12. Djajaputra, D., Wu, Q., Wu, Y., Mohan, R.: Algorithm and Performance Of A Clinical Imrt Beam-angle Optimization System. Physics in Medicine & Biology **48**(19), 3191 (2003) 4
13. Pugachev, A., Xing, L.: Computer-assisted selection of coplanar beam orientations in intensity-modulated radiation therapy. Physics in Medicine & Biology **46**(9), 2467 (2001) 4
14. Wang, C., Dai, J., Hu, Y.: Optimization of beam orientations and beam weights for conformal radiotherapy using mixed integer programming. Physics in Medicine & Biology **48**(24), 4065 (2003) 4
15. Lim, G.J., Ferris, M.C., Wright, S.J., Shepard, D.M., Earl, M.A.: An optimization framework for conformal radiation treatment planning. INFORMS Journal on Computing **19**(3), 366–380 (2007) 4
16. D D'Souza, W., Meyer, R.R., Shi, L.: Selection of beam orientations in intensity-modulated radiation therapy using single-beam indices and integer programming. Physics in Medicine & Biology **49**(15), 3465 (2004) 4
17. Hou, Q., Wang, J., Chen, Y., Galvin, J.M.: Beam orientation optimization for imrt by a hybrid method of the genetic algorithm and the simulated dynamics. Medical Physics **30**(9), 2360–2367 (2003) 4
18. Aleman, D.M., Kumar, A., Ahuja, R.K., Romeijn, H.E., Dempsey, J.F.: Neighborhood Search Approaches to Beam Orientation Optimization i Intensity Modulated Radiation Therapy Treatment Planning. Journal of Global Optimization **42**(4), 587–607 (2008) 4, 12
19. Huang, G., Liu, Z., Weinberger, K.Q., van der Maaten, L.: Densely Connected Convolutional Networks. arXiv preprint arXiv:1608.06993 (2016) 5
20. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning For Image Recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016) 5, 7
21. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602 (2013)
22. Ioffe, S., Szegedy, C.: Batch Normalization: Accelerating Deep Network Training By Reducing Internal Covariate Shift. arXiv preprint arXiv:1502.03167 (2015) 7
23. Hahnloser, R.H., Sarpeshkar, R., Mahowald, M.A., Douglas, R.J., Seung, H.S.: Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. Nature **405**(6789), 947 (2000) 7
24. He, K., Zhang, X., Ren, S., Sun, J.: Delving Deep into Rectifiers: Surpassing Human-level Performance on Imagenet Classification. In: Proceedings of the IEEE international conference on computer vision, pp. 1026–1034 (2015) 8

25. Agrawal, R.: Sample mean based index policies by o (log n) regret for the multi-armed bandit problem. Advances in Applied Probability **27**(4), 1054–1078 (1995) 8
26. Soo Chang, H., Fu, M.C., Hu, J., Marcus, S.I., Fu Robert H Smith, M.C.: An Adaptive Sampling Algorithm for Solving Markov Decision Processes An Adaptive Sampling Algorithm for Solving Markov Decision Processes Area of review: Stochastic Models. Source: Operations Research **53**(1), 126–139 9
27. Boyd, S., Parikh, N., Chu, E., Peleato, B., Eckstein, J.: Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. Foundations and Trends® in Machine learning **3**(1), 1–122 (2011) 10, 11
28. Basar, T., Olsder, G.J.: Dynamic noncooperative game theory, vol. 23. Siam (1999) 12, 14