# P³IMRT

## Direct machine parameter optimization

Björn Hårdemark, MSc
Anders Liander, MSc
Henrik Rehbinder, PhD
Johan Löf, PhD
RaySearch Laboratories AB
Stockholm, Sweden

In traditional IMRT inverse planning, fluence profiles are optimized to produce a desired dose distribution. These profiles are then converted to MLC settings, with a possible degradation of plan quality. Furthermore, this two-step process may produce more segments and MUs than necessary.

With direct machine parameter optimization (DMPO), MLC settings are produced directly within the optimization process. Therefore, there is no need for conversion, filtering, weight optimizations or other kinds of post-processing, and the plan quality will not degrade.

### Traditional inverse IMRT planning

The starting point for inverse IMRT planning is the dose, $d$, that the physician would like the patient to receive. The physician's preferences are described by an objective function, $f(d)$, and possibly a set of constraints, $c(d)$.

The dose, in turn, is a function of the fluence distributions in the beams. The fluences are modeled by grids that divide the beams' cross sections into small elements. The opening density matrices (ODM) describe the transmission through each such element. If nothing else is in a beam's path, the ODM will closely resemble the fluence incident on the patient surface, and for convenience, we will refer to the ODM as the beam's fluence.

The transmission coefficients of all relevant ODM elements are collected into a vector, $\tau$. The dose engine provides the function $d(\tau)$ that calculates the dose resulting from the ODMs. The dose engine will also consider other modulators in the beam, such as wedges and boluses.

This is where traditional IMRT optimization stops. The problem posed and solved is:

$$
(1) \quad \min_{\tau} \quad F(\tau) \\
\text{s.t.} \quad \begin{cases} C(\tau) \leq 0 \\ \tau \leq 0 \end{cases}
$$

where $F(\tau)=f(d(\tau))$ and $C(\tau)=c(d(\tau))$. The condition $\tau \geq 0$ ensures non-negative fluences.

The fluences, and hence the dose, resulting from the optimization problem (1) cannot be delivered to the patient directly. Instead, a conversion to control points is necessary. This conversion is done without taking the physician's preferences into account, and the resulting dose distribution may not be desirable.

# PHILIPS

## Direct machine parameter optimization

Direct machine parameter optimization avoids this problem by taking the optimization problem one step further [1]. Given a beam model, the fluence, $\tau$, can be calculated from a set of control points described by the leaf positions, $x$, and the segment weights, $w$. This model is described by the function $\tau(x,w)$. By incorporating this function, the optimization problem can be restated as:

$$(2) \quad \min_{x,w} \quad F(x,w)$$
$$\text{s.t.} \quad \begin{cases} C(x,w) & \leq & 0 \\ Ax & \leq & b \\ w & \geq & 0 \end{cases}$$

where $F(x,w)=f(d(\tau(x,w)))$, and similarly $C(x,w)=c(d(\tau(x,w)))$. The condition $w \geq 0$ ensures non-negative fluence and $Ax \leq b$ describes user and machine-specific requirements on the leaf positions. These requirements can be formulated as linear constraints and include minimum gaps, interdigitation, maximum tip differences, segment areas, and more.

## Optimization process overview

The direct machine parameter optimization problem (2) is harder to solve than the fluence based (1). It is not convex, has a greater degree of non-linearity and parameter coupling, and is subject to numerous linear constraints. Therefore, a good starting point for this problem is needed to ensure a high quality plan. The following are the basic steps of the optimization:

1. The first few iterations are used to find an initial set of control points that meet the user and machine specific requirements.
2. During the remainder of the iterations, the MLC leaf positions and segment weights are optimized. Throughout this process, the plan is feasible for delivery. When the optimization is finished, no post-processing is needed.

## Optimization engine

The optimization is performed by RayOptimizer [2] and its optimization core, NPSOL[3].

NPSOL is a sequential quadratic programming (SQP) algorithm. It can handle both non-linear and linear constraints as well as variable bounds. In ODM optimization, the number of variable bounds is very large, and rigorous handling of them may be time consuming. In order to alleviate this, some approximate handling of the bounds is allowed for many problem types.

In direct machine parameter optimization, machine constraints and user preferences may lead to a very large number of linear constraints. These are also handled in an approximate manner when possible. The final result, however, is guaranteed to fulfill the specified constraints.

## Gradients

The SQP algorithm is gradient based. In every iteration, the optimizer will use the gradient of the objective function with respect to the optimization parameters (leaf positions and weights) to find an update of the parameters that improves the objective function.

As illustrated below (Figure 1), the objective function, and hence the gradient, is calculated in several steps, starting in the dose, going up through the fluence and into the treatment head.

The expressions for the objective gradients are:

$$\frac{\partial f}{\partial x} = \frac{df}{dd}\frac{dd}{d\tau}\frac{\partial\tau}{\partial x}$$
$$\frac{\partial f}{\partial w} = \frac{df}{df}\frac{dd}{d\tau}\frac{\partial\tau}{\partial w}$$
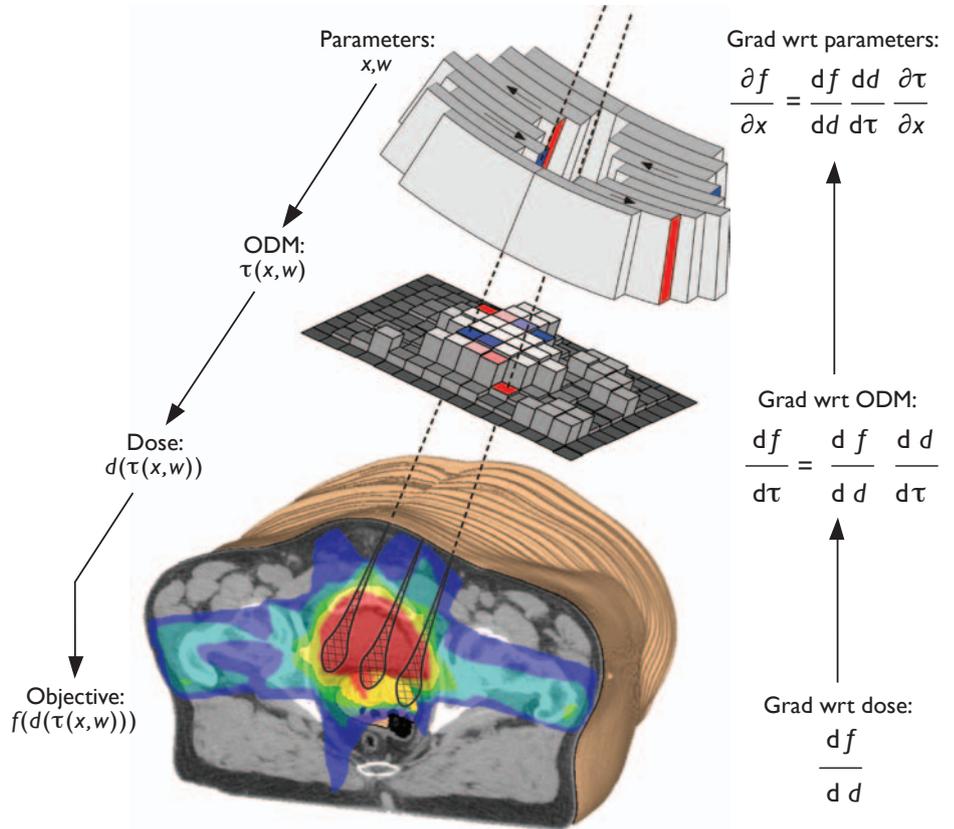
and similarly for the constraint functions.



Parameters:
$x,w$

ODM:
$\tau(x,w)$

Dose:
$d(\tau(x,w))$

Objective:
$f(d(\tau(x,w)))$

Grad wrt parameters:
$$\frac{\partial f}{\partial x} = \frac{df}{dd}\frac{dd}{d\tau}\frac{\partial\tau}{\partial x}$$

Grad wrt ODM:
$$\frac{df}{d\tau} = \frac{df}{dd}\frac{dd}{d\tau}$$

Grad wrt dose:
$$\frac{df}{dd}$$

Figure 1: An objective function and its gradients are calculated through several steps.

## Dose calculation

During optimization, dose is computed through an ODM. In order to be able to optimize on machine parameters, the software translates control points into equivalent ODMs during optimization. This approximation, and the dose engine itself, introduces a small error in the dose. In order to minimize the error, a collapsed cone (CC) dose calculation performed after the leaf sequencing is used as an accurate starting point, and only differences from this position are calculated with the dose engine. The final dose is computed with the CC dose engine.

## Starting from control points

If the beams already have control points, for example from an earlier optimization or from forward planning, these control points can be used for optimization, and step one in the optimization process can be omitted. In that case, all leaves that are exposed by the jaws, except those being closed at their minimum gap, are used in the optimization. Otherwise, control points are generated automatically as described in the next section.

## Finding initial control points

During optimization of machine parameters, only leaf positions and segment weights are optimized. The jaw positions, number of segments, and leakage avoidance, such as flag poles, remain constant. It is therefore important for the optimizer to have a good starting point for the machine parameter optimization. The process of finding such a point is described below.

## Optimizing the fluence distributions

The first few iterations are very similar to ordinary intensity modulated optimization. An initial estimate is produced by assigning a uniform fluence to the beam's eye view of the target for each beam, and the ODM elements are then optimized.

## Conversion

RayOptimizer is equipped with an integrated leaf sequencer that is designed specifically to produce a suitable starting point for the subsequent machine parameter optimization.

The focus is therefore on the user and machine-specific requirements. The sequencer will aim to produce nearly, or just as many, segments as the user has allowed. In distributing segments between the beams, both beam weights and beam complexity are taken into account.

The sequencer ensures that all segments have a certain degree of uniqueness, and it will also aim to produce regular aperture shapes in order to avoid the tongue-and-groove effect.

The leaf sequencing process consists of the following basic steps:

1. The ODMs are resampled into a grid that matches the MLC leaves.
2. The fluence values are constrained to a number of equidistant levels, and the fluences are then decomposed into smaller elements using a combination of the "close in" (Figure 2) and "leaf sweep" (Figure 3) techniques [4].
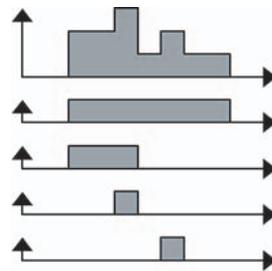


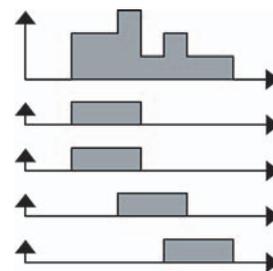Figure 2: "Close in" technique



Figure 3: "Leaf sweep" technique

3. The elements are used to build the segments.
   In this process, all requirements on the leaf positions
   are considered, including shape regularity.

4. Jaw positions are assigned to the segments, and the
   remaining requirements are processed, including
   segment uniqueness. Segments that cannot fulfill all
   requirements are discarded.

The number of segments produced depends on the
number of fluence intensity levels that are used in
step two. The sequencer will try several different values
in order to produce a number of segments close to, but
not exceeding, the user-specified maximum number.
The sequencer will avoid unnecessary leakage by using
the jaws, separately or in combination. For a typical
Varian machine, closed leaf pairs will be positioned
so that they are covered by the jaws (Figure 4). For a
typical Elekta machine, the jaws will conform as close as
possible to the MLC aperture (Figure 5). If needed, the
flag pole technique of positioning the leaves and jaws is
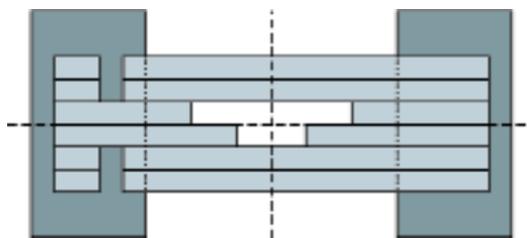incorporated to minimize leakage.

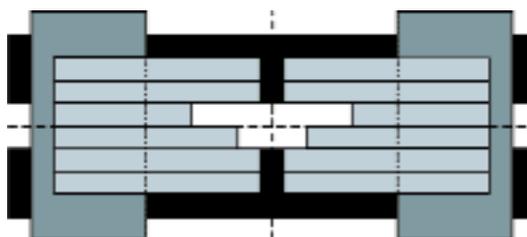Figure 4: Leakage avoided by moving leaves

Figure 5: Leakage avoided by conforming jaws

### References

[1] J. Löf. *Development of a general framework for
optimization of radiation therapy.* PhD thesis, Stockholm
University, 2000.

[2] J. Löf, H. Rehbinder, T. McNutt, and S. Johnson. P³IMRT:
*Inverse Planning Optimization.* Pinnacle³ White Paper.

[3] P. E. Gill, W. Murray, MA. Saunders, et al. *User´s Guide
for NPSOL (Version 4.0): A Fortran package for nonlinear
programming.* Report No. SOL 86-2, 1992.

[4] T. R. Bortfeld, D. L. Kahler, T. J. Waldron, and A. L. Boyer.
X-ray field compensation with multileaf collimators.
I*nternational Journal of Radiation Oncology*Biology*Physics*
28(3):723-730, 1994.

**PHILIPS**