MATRAD - AN OPEN-SOURCE TREATMENT PLANNING TOOLKIT FOR EDUCATIONAL PURPOSES

H.P. Wieser[1,2,3], N. Wahl[1,2,4], H.S. Gabryś[1,2,3], L.R. Müller[1,2,4], G. Pezzano[1,2], J. Winter[1,2,4], S. Ulrich[1,2], L. Burigo[1,2] , O. Jäkel[1,2] and M. Bangert[1,2]

[1] Department of Medical Physics in Radiation Oncology, German Cancer Research Center - DKFZ, Im Neuenheimer Feld 280, D-69120 Heidelberg, Germany
[2] Heidelberg Institute of Radiation Oncology - HIRO, Im Neuenheimer Feld 280, D-69120 Heidelberg, Germany
[3] Medical Faculty, University of Heidelberg - Im Neuenheimer Feld 672, D-69120 Heidelberg, Germany
[4] Physics and Astronomy Faculty, University of Heidelberg, Im Neuenheimer Feld 226, D-69120 Heidelberg, Germany

Abstract— **We present educational aspects of matRad – an open-source treatment planning toolkit for three-dimensional intensity-modulated radiotherapy treatment planning supporting photons, scanned protons and scanned carbon ions. matRad is publicly available for download on GitHub and does not require payable software-products to run or to change its source code. This manuscript helps new users to get familiar with the basic concept, the matRad GitHub environment, and potential applications. Specifically we discuss seven novel workflow examples that illustrate usage of matRad's code base and we introduce three practical treatment planning examples from a planner's point of view. The workflow examples and the treatment planning tutorial are available in the form of Matlab scripts and documented with pdf files and wiki pages, respectively. They are intended as both learning and teaching material, e.g., in a classroom setting. The provided examples range from simple to complex treatment planning scenarios and can all be executed in a couple of minutes on a standard desktop computer.**

Keywords— **radiotherapy, treatment planning, DICOM, open-source, optimization, dose calculation**

I. INTRODUCTION

According to the World Health Organization nearly every sixth death in 2015 was caused by cancer. There are various approaches to treat cancer, such as surgery, chemotherapy, radiotherapy, immunotherapy or target agents. More than 50% of all cancer patients are either treated with radiation therapy as primary treatment or in combination with aforementioned approaches [1]. The most common type of radiation therapy is external beam therapy, which is characterized by directing ionizing radiation from the outside into the patient's body in order to sterilize cancerous tissue.

Integral part of radiation therapy is a computer-aided process called treatment planning, which is performed upfront to the actual patient treatment by the usage of specialized software. The aim of the treatment planning process is firstly to simulate the dose deposition within the patient body and secondly to optimize the dose distribution according to clinical objectives and constraints that define a trade-off between tumor irradiation and normal tissue sparing.

In the past decade, the advancement of radiation therapy in general and treatment planning in particular correlated strongly with the technical developments of computer hardware. This led to conceptually more complex treatment techniques, such as intensity- (IMRT) or volumetric-modulated arc radiation therapy (VMAT) which are accompanied by more sophisticated treatment planning software and algorithms.

Nowadays, the software landscape in radiation therapy is dominated by commercial closed-source products, also due to safety regulations. This limits a flexible application for education and research. Besides the closed-source characteristic, such specialized treatment planning software is usually very expensive, thereby limiting its availability for research and education in developed and even more in developing countries.

Consequently, there is a need for educational open-source treatment planning software in the medical physics community to ease the way of teaching basic workflow principles and concepts along with their algorithms. In the past, various open-source systems evolved in the radiotherapy area, whereby most of them focus on specific treatment planning aspects [6, 12, 14].

Recently, we have introduced matRad [17], an open-source dose calculation and optimization toolkit for intensity-modulated radiation therapy with photons, scanned protons, and scanned carbon ions. Here we report on potential applications of the matRad toolkit in an educational setting. Hence, this manuscript will not be a typical research paper but rather a description of matRad's functionalities and characteristics and how they can be used for teaching. Furthermore, we provide learning material to support beginners getting started with matRad and course material for medical physics instructors.

II. METHODS

A. Overview

matRad is an open-source multi-modality treatment planning toolkit that is entirely written in the interpreted numerical programming language Matlab and covers a wide range of functionalities from DICOM import, photon, scanned proton and scanned carbon ion dose calculation, dose optimization to plan visualization. matRad's code base corresponds to a flexible and modular set of functions containing well established radiotherapy algorithms with clearly structured function interfaces.

The software package is published under the GNU public license[1] and the complete source code along with a standalone version and open-source patient data sets can be downloaded for free from the file hosting service GitHub[2]. Moreover, the source code, excluding the graphical user interface (GUI) is compatible, with GNU Octave.

matRad was developed with the intention (i) to support gaining experience with radiation therapy treatment planning, (ii) to allow for algorithmic developments on top of the basic toolkit, (iii) to be used as a learning and teaching tool, and (iv) to enable its application in research.

Throughout the manuscript we will highlight matRad data, i.e., Matlab variables, and algorithms, i.e., Matlab functions, using the Courier New font (`e.g., matRad_fluenceOptimization`). In section II.C we start explaining the semantics of the main variables used in matRad, before we focus on specific matRad functionalities and aspects, e.g., DICOM import, plan visualization, base data format, and educational characteristics of the source code repository in II.D-II.I. In section III, we present online course material of detailing different matRad workflows.

B. Prerequisites

In order to run matRad, we recommend to use a system having at least a 64-bit processor with 1 GHz, 4 GB RAM and 100 MB hard disk space. **Besides the image processing toolbox, which is required for the DICOM import, no other Matlab toolbox is needed to run matRad in its entirety including inverse optimization. We have actively tested compatibility with Matlab version ≥ 8.3 (R2014a).**

This manuscript refers to matRad version 3.0.0. Being under active development, function- and variable-names may change in future releases.

C. Understanding and setting up matRad

matRad models a radiotherapy treatment planning workflow by consecutive function calls and organizes the relevant data for treatment planning in multiple structures in the base workspace of Matlab as illustrated in Table 1. The main sequential treatment planning steps are briefly described next.

Patient data can either be imported using matRad's import interface (compare section II.E) or by loading one of the existing open-source patient data sets. Two variables (`ct`, `cst`) have to be available in the Matlab workspace in order to proceed. The first variable (`ct`) holds the computed tomography (CT)/imaging data stored as a structure array; the corresponding segmentations of volumes of interests (VOI) are stored in a cell array labeled `cst`. Apart from the voxel index list for each VOI, the `cst` contains meta information for dose optimization, such as the overlap priority and constraints/objectives used to calculate the objective function value during inverse optimization.

In the next step, meta treatment planning parameters, e.g. the radiation modality, beam angles, and bixel spacing, have to be defined in the `pln` structure array by the user before the beam geometry information is generated in `matRad_generateStf` and stored in the `stf` structure array.

Table 1: matRad variables as of version 3.0.0.

Variable name	Description
`ct`	ct images and ct meta information
`cst`	segmentations along with constraints & objectives for dose optimization
`pln`	treatment plan information
`stf`	beam geometry information
`dij`	dose influence information
`resultGUI`	dose distribution with corresponding bixel/pencil beam weights

Once the beam geometry information is available, photon or particle dose calculation in water for unit fluences can be carried out (`matRad_calcPhotonDose` or `matRad_calcParticleDose`). The dose calculation function outputs a structure array labeled `dij`, which holds the dose contribution for each individual bixel/pencil beam. Next, the fluence optimization function (`matRad_fluenceOptimization`) can be called in order to optimize individual bixel/pencil beam weights such that the resulting dose distribution produces

[1]The license agreement is available on https://github.com/e0404/matRad/blob/master/LICENSES.txt

[2] http://www.matrad.org/

a minimal objective function value according to given constraints and objectives. The optimized dose distribution is ultimately stored in a structure array called `resultGUI`, which allows plotting of dose volume histograms (DVHs) and computation of quality indicators, such as mean dose and maximum dose. If the optimization does not produce acceptable results, a re-optimization using different parameters can be carried out until the dose distribution is deemed satisfying.

For photons, we also provide an experimental multileaf collimator sequencing and a direct aperture optimization algorithm.

To illustrate the sequential workflow and the compactness of the code, Figure 1 presents the complete source code for an intensity-modulated scanned proton treatment plan that consists of dose calculation, inverse dose optimization and visualization.

```
load('TG119.mat'); % load ct and cst
%% define treatment plan
pln.radiationMode = 'protons';
pln.machine       = 'Generic';
pln.numFractions  = 30;
pln.propStf.bixelWidth    = 5;
pln.propStf.gantryAngles = [0 45 315];
pln.propStf.couchAngles  = [0  0   0];
pln.propStf.numOfBeams    = 3;
pln.propOpt.bioOptimization = 'const_RBExD';
pln.propStf.isoCenter = ones(3,1) *  ...
matRad_getIsoCenter(cst,ct,0);
%% generate geometrical steering information
stf       = matRad_generateStf(ct,cst,pln);
%% perform dose calculation
dij       = matRad_calcParticleDose(ct,cst,stf,pln);
%% perform optimization
resultGUI = matRad_fluenceOptimization(dij,cst,pln);
%% start visualization
matRadGUI
```

Figure 1: matRad code generating a proton treatment plan on phantom TG119

In principle, there are three different ways to run matRad. The first possibility is given by executing scripts directly from the command prompt in the integrated development environment (IDE) of Matlab. For instance, users can start the script `matrad.m` in the root folder of matRad, which defines a default intensity-modulated photon treatment planning workflow. Alternatively (or complementary), users may want to work with the GUI by executing `matRadGUI` from the command prompt.

The second option to run matRad is given by using GNU Octave instead of Matlab, which is explained in detail in section II.F.

The third option is intended for users not familiar with the scripting language or users who do not intend to make code changes. For this group of users, we provide a matRad standalone executable for Windows 7, 8, and 10, which is able to run all functionalities via the GUI. The standalone does not require a Matlab product license. Only the freely available system-specific Matlab runtime package[3] is needed and automatically downloaded during the installation of the matRad standalone.

D. Base data format

matRad 3.0.0 comprises three open-source base data sets for treatment planning that can be used for analytical photon, proton and carbon ion dose calculation, respectively. As a side note, for photons we provide the possibility to interface to the Monte Carlo (MC) photon dose calculation VMC++.

Each base data set is stored in a separate file using Matlab's file format (*.mat) and is named according to the concatenation of the radiation type and the treatment machine (e.g., `protons_Generic.mat`). Since we provide base data sets for generic treatment machines, all base data sets end with `Generic`. The differentiation of radiation modality and treatment machine allows to keep multiple base data sets in parallel, whereas each depicts a different radiation modality or a different treatment machine.

Base data sets are stored as Matlab structure arrays. The base data sets contain, apart from the actual beam data, also meta information such as the name of the selected treatment machine, the utilized lateral beam model, the creation date and radiation modality. In the following, we explain the photon and particle specific base data set properties separately.

Photon base data format

matRad facilitates a singular value decomposed pencil beam algorithm for analytical photon dose calculation [3]. This reduces the beam description of a fully depth-dependent convolution pencil kernel to three depth-independent radial kernels. These kernels are stored in the `photons_Generic.mat` file as a structure together with the primary fluence of the accelerator and geometrical information about the treatment device. The photon base data set provided for free with matRad describes a 6 MeV SIEMENS Artiste 3; detailed instructions how to establish the base data set for your own accelerator can be found in [3].

The photon base data set was obtained from the clinically approved photon dose calculation engine PDC++ for 501 different source to surface distances (from 500 mm to 1000 mm in 1 mm steps). Besides the source to axis distance (SAD), the source to collimator distance (SCD) is also stored in the base data set. These distances are required during dose calculation to determine the treatment-specific source to surface distance (SSD) in order to select the corresponding scattering kernels and depth dose components.

[3] http://www.mathworks.com/products/compiler/mcr

Particle base data format

matRad facilitates a conventional pencil beam algorithm for analytical particle dose calculation [15] supporting protons and carbon ions. In total, we provide for each modality around 120 different beam energies, which are all stored in the multi-dimensional substructure `data`. Figure 2 depicts an overview of the first three proton energies of the generic base data with ranges of

Last, we keep for each initial beam energy a substructure named `initFocus`. This structure allows to store lookup tables for the beam widening in air as function of the geometrical distance for multiple initial beam widths. As we provide one focus in the public base data set, only one lookup table can be found in here. However, if desired, the beam widening in air of multiple initial beam widths can be added to each initial beam energy in the base data set. The actual beam width on the patient surface is then calculated considering the

Fields	range	energy	depths	Z	peakPos	sigma	offset	initFocus	LET
1	10	31.7290	51x1 double	51x1 double	7	51x1 double	0	1x1 struct	51x1 double
2	13	36.7986	51x1 double	51x1 double	10	51x1 double	0	1x1 struct	51x1 double
3	16	41.3788	51x1 double	51x1 double	13	51x1 double	0	1x1 struct	51x1 double

Figure 2: The first three entries of the proton base data set

10, 13 and 16 mm, respectively. For each individual beam energy, we store the range [mm], initial beam energy [MeV], peak position [mm] and an offset [mm] as scalar values. The offset value measured in water equivalent path length allows to shift the entire beam to model additional material in the beam line. Moreover, we store the integrated depth dose (IDD) profiles [MeV cm^2 / g / 10^6 primaries] and the lateral spread (sigma) [mm] due to Multiple Coulomb scattering using a single Gaussian approximation at given depth values as vectors.

If a single Gaussian lateral beam model is used, as it is the case for the public base data set, the lateral spread is stored in a subfield named `sigma`. If a double Gaussian lateral beam model is facilitated, then the component modelling the inner core is stored in the subfield `sigma1` and the broader low dose component is saved in `sigma2`. Their relative contribution is controlled using the subfield `weight`.

For protons and carbon ions there are, compared to photons, additional parameters available to accurately model the beam divergence in air and to consider multiple initial beam widths (also known as beam foci).

First, the `SAD` parameter corresponds again to the geometrical source to axis distance in millimeter. The second one, `BAMStoIsoDist` denotes the geometrical distance from the beam application monitoring system (BAMS)/beam nozzle to the isocenter. Next, a lookup table `LUT_bxWidthmimFWHM` is stored, which determines the initial beam width to be used during treatment planning. Let N be number of depth values used in the lookup table, then `LUT_bxWidthmimFWHM` is of size 2xN. In the case of the public base data set, this table contains only four values (2x2 matrix). As we provide only one beam width, we set the minimum required full width half maximum (FWHM) to a constant value for lateral pencil beam spacings from 0 to infinity. This lookup table allows users to specify, the usage of a certain beam width given a certain lateral pencil beam spacing.

constraint given by `LUT_bxWidthmimFWHM`, the SSD and the spread in air of the utilized beam width. A detailed description of the particle base data set format can be found in the wiki[4].

For protons, we additionally provide the linear energy transfer (LET) according to [18], which allows the future usage of phenomenological variable relative biological effectiveness (RBE) models or LET-based optimization. In contrast, for carbon ions we supplementary provide dose-averaged radio sensitivity parameters of the linear quadratic (LQ) model to enable variable RBE calculations based on the local-effect model (LEM) IV for a generic early and late responding tissue. For a detailed explanation of the carbon base data set we refer to [17].

If users want to perform treatment planning mimicking a specific particle treatment machine, then Monte Carlo particle transport simulations (e.g., using TOPAS, FLUKA) need to be performed considering beamline specific geometries and machine specific characteristics. Simulating the dose deposition of individual pencil beams for various initial beam energies in water allows in a further step to extract the IDD profiles. In addition to the IDD, the lateral beam profile represented by either a single or a double Gaussian needs to be fitted to the lateral dose distribution in each depth, respectively.

E. Import and Export

The matRad code base comprises the open-source CORT dataset, i.e., three segmented patient CTs, [5] as well as two phantom CTs. We provide the data as Matlab files, organized in the matRad data structures `ct` and `cst`.

Furthermore, own patient data may be imported either via a DICOM import interface based on Matlab's image

[4] https://github.com/e0404/matRad/wiki/Particle-Base-Data-File

processing toolbox or via custom import interfaces for binary data formats (e.g. NRRD).

matRad's DICOM import functionality allows to import CT images, radiation therapy (RT) structure set, RT Dose, and RT Plan files. In order to load patient data, the user has to specify the input directory. The directory can contain files of multiple patients and multiple image series of a single patient. After reading the input directory, the user is able to choose the desired patient and select suitable RT DICOM files. The use of a Hounsfield unit look up table (HLUT) is recommended and can be provided in the directory *.../dicomImport/hlutLibrary*. Specifically, HLUTs need to be accessible by matRad to convert Hounsfield units in case of photon dose calculation to electron densities relative to water or in case of particles to stopping powers relative to water. If no HLUT can be found for the corresponding CT, a generic table will be used independent of the radiation modality.

By default, all images (segmentation and dose) are resampled at the import resolution of CT. The user may chose the grid of the CT, the grid of the dose or specify any resolution in x, y, and z directions. When a plan file was selected for import, the user is asked to specify a proper machine base data set to be able to perform a dose re-calculation. After successful import of the DICOM files, the user is prompted to save a binary Matlab file containing `ct`, `cst`, and - if applicable `pln` as well as `resultGUI` structures corresponding to CT images, RT structures, and plan data. This file can be loaded directly to matRad for treatment planning.

Other than DICOM, binary data export and import supports only a limited set of file formats. So far, matRad can export any image/cube to the established formats NRRD [19], MetaImage (MHA/MHD) [20], and VTK image [2]. Import is possible only for NRRD files with basic functionality; an image file for the CT and binary images for the segmentations are required, with the meta-information (i.e. resolution, dimensionality etc.) being collected from the file headers. Users must take care of matching resolutions and dimensionality, and additional tweaking of the resulting workspace variables might be necessary.

F. Visualization and Plan Evaluation

The GUI includes a visualization engine to display axial, sagittal and coronal slices of the CT image set. Using the GUI, different colormaps, value windows (with available presets), and units (i.e. Hounsfield units or electron density/stopping power) may be selected, depending on the CT in use. Segmentations are displayed as (pre-computed) contours. Optimized or imported dose distributions can be displayed as a transparent colorwash overlay with a set of available colormaps, and users can define which isodose lines should be displayed explicitly. For particles, matRad calculates besides the total dose distribution also the dose distribution of each individual beam.

Complementary to the slice view in the GUI, a 3D view can be opened, rendering VOIs as isosurfaces and additionally visualizing the plan geometry. While the matRad GUI contains sliders and input fields to adjust the color mapping as well as selectors to hide or show certain entities, the visualization may be modularly executed from a script or the command line, allowing to draw the desired sub-images also within other Matlab figures/axes apart from the GUI to create graphics for publication or presentation purposes. Alternatively, the GUI provides a "screenshot" functionality to export the current slice view directly to an image or Matlab figure file. Apart from visualization of the optimized or imported dose distributions, calculation of volume-based clinical endpoints, i.e., quality indicators, is available in matRad. This includes basic indicators like mean and standard deviation of the dose in a VOI, DVHs [7] and the corresponding dose-volume points, as well as conformity and homogeneity indices [10, 13].

G. Matlab and GNU Octave

Matlab is a proprietary high-level programming language developed by The MathWorks, Inc. primarily intended for numerical computations. The matRad toolkit is also compatible with GNU Octave [8, 9] which is an open-source alternative to Matlab. Since Octave is part of the GNU project, it is free software under the terms of the GNU General Public License. Octave development is intended to be mostly compatible with the Matlab language. Similar to Matlab, GNU Octave also contains an integrated development environment in addition to the traditional command line interface and a graphical user interface with limited functionality compared to Matlab.

There are two main limitations of using matRad with GNU Octave. First, the matRad GUI is not supported in Octave. While matRad's GUI is not mandatory for the treatment planning workflow, it provides a valuable graphical interface for data visualization and handling of DICOM data. The second limitation is related to the dynamically loadable MEX-files of the Matlab interface for the Ipopt (Interior Point OPTimizer) software package [16]. The Matlab interface of Ipopt available from COIN-OR under the Eclipse Public License is linked against the linear solver MA57 included in the Matlab software. In order to use Ipopt with GNU Octave, the Ipopt package as well as a linear solver, e.g., MUMPS[5], needs to be compiled for the user's platform and the Matlab interface of Ipopt has to be compiled and linked with GNU Octave. This procedure is also described in detail on our wiki page (see section H).

[5] http://mumps.enseeiht.fr

H. How to access and contribute

We host our complete source code on the web-based version control system GitHub. The code itself is stored and structured as part of a project repository.[6] This allows for version control and the management of different code bases, i.e. branches.

The most stable and tested code can always be found in the 'master' branch, whereas the development branch 'dev' combines the latest feature developments. Individual feature developments can be found in 'dev_feature1' before being merged into the 'dev' and then further into the 'master' branch. By using GitHub, we can publicly perform transparent source code management keeping track of all changes.

When using the versioning control system git, the source code can either be directly cloned from the public project repository or it can be forked into the user's public GitHub environment and then be cloned to the users local system. Alternatively, the source code can simply be downloaded without version control as zip file from the corresponding code branch.

Besides source code management and public accessibility, GitHub also provides bug tracking, project/task management and a wiki functionality.

The issue page on our GitHub project page serves, besides matRad@dkfz.de, as a contact point to report bugs, to ask questions, and to leave ideas for feature requests. Furthermore, GitHub enables transparent code development and clear authorship tracking. Since matRad is intended to be a medical physics community toolkit, we encourage also others to contribute to the public code base either via bug fixes, code improvements, or new features.

In particular, the GitHub 'pull request' workflow provides a suitable solution to easily integrate code contributions from others. Once a pull request has been made, we as the repository owner can review, comment on the new code contribution and can then further accept, reject, or leave the pull request open. In most cases, an iterative code review process is started to further improve the new code contribution. After successful code integration into the public project repository, the contributor is automatically listed in the public URL[7] and is associated as author of the corresponding code lines.

I. Wiki and training material

Along with the actual source code, GitHub allows to host a project wiki.[8] The matRad project wiki is a collaborative effort to provide various kinds of information about matRad. As of now, the wiki comprises 29 pages. It is the best starting point for new users to get familiar with matRad. The wiki page can be edited directly by any GitHub user in the web browser by using the markup language.

The wiki is mainly comprised of three parts. The first part is a quick overview page about what matRad actually is. The second part provides a quick-setup guide intended to help new users getting started with matRad for the first time. The third part of the wiki includes detailed technical documentation describing among others, the general workflow, the main variables used in matRad, and the implemented algorithms along with their functionality.

Furthermore, the wiki links to a 40-minute long webinar which was given by Dr. Mark Bangert for the medical physics brown bag seminar in summer 2016 at the Massachusetts General Hospital.

For additional information, we also want to explicitly refer to two previously published manuscripts about matRad. The first paper [4] describes the initial version and core functionalities of matRad in 2015 and the second paper [17], published in 2017, reports on recent developments and the validation of matRad's core components (ray tracing, photon and particle dose engine).

III. RESULTS

In the first part of the result section, we present selected planning workflow examples based on matRad. In particular, we focus on executing functions from a script and explain how to run a dose calculation, how to trigger dose optimization, how to manipulate variables using the Matlab command prompt, and lastly how to analyze and visualize the resulting dose distributions.

The second part depicts practical radiotherapy treatment planning examples for photons and scanned protons from a treatment planner perspective aiming to teach the ability to create reasonable treatment plans.

A. Workflow Examples

The workflow examples are designed for new users who want to learn how to use matRad. They showcase treatment planning tasks and provide inspiration for customized usage of matRad's functionality. In total, we introduced seven use cases; corresponding Matlab scripts are located in the subfolder 'examples' of the matRad code base.

The first workflow example explains how to create a user-specific cubical or spherical phantom geometry (`ct`) in line with a corresponding segmentation, i.e., `cst` variable. The intention of this example is to develop an understanding for the `ct` and `cst` variable. In addition, generic phantoms might be valuable later on for testing user-specific code implementations.

[6] https://github.com/e0404/matRad
[7] https://github.com/e0404/matRad/graphs/contributors
[8] https://github.com/e0404/matRad/wiki

The second workflow example generates two treatment plans for intensity-modulated radiation therapy with photons for different beam ensembles on the open-source phantom TG119. In the end, a visual comparison of the resulting dose cubes is performed, whereas in the analysis, the dose to 95% of the volume of interest (D95) of both treatment plans is compared.

The third workflow example represents an intensity-modulated photon treatment plan for the open-source head and neck patient. Besides a multifield optimization, multileaf sequencing is applied to translate the continuously optimized fluences into multiple deliverable static segments. Lastly, direct aperture optimization is carried out followed by calculating quality indicators and DVHs.

The fourth workflow example shows a Monte Carlo photon dose calculation based on the VMC++ algorithm for the generic box phantom. The treatment plan is only comprised of a single beam with zero gantry- and zero couch-angle. In the end, the histogram of dose values belonging to the target structure is visualized.

The fifth workflow example demonstrates an intensity-modulated scanned proton treatment plan for the open-source prostate patient using two laterally opposing beams. After dose calculation and optimization, we simulate an isocenter shift and re-calculate the dose (forward dose calculation) using the previously optimized fluence to mimic a patient shift. The resulting dose cubes are then visually and quantitatively compared using the `matRad_plotSliceWrapper` and the `matRad_gammaIndex` [11] function. Exemplary, Figure 3 illustrates the visualization of the unshifted proton dose of the prostate case in the axial isocenter slice using the function `matRad_plotSliceWrapper`.

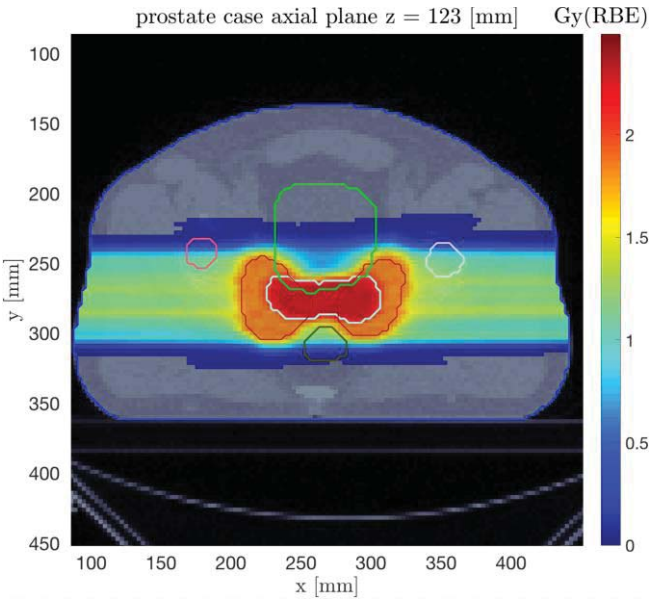

Figure 3: Prostate treatment plan with two laterally opposing proton beams

The sixth workflow example is similar to the one before but instead of shifting the isocenter, we manipulate the stopping power of the CT by adding 3.5% to the initial values in order to simulate a simple range undershoot.

The seventh and last workflow example shows a scanned carbon ion treatment plan for the open-source liver patient. We define a single beam with 300° gantry- and 0° couch-angle. After a dose calculation and optimization of the RBE-weighted dose, we change the radiobiological characteristic of the patient by assuming a different radio-sensitivity and re-calculate the RBE-weighted dose utilizing the existing pencil beam intensities.

Each workflow example has been designed to cover a different aspect of planning, data manipulation, data visualization and evaluation. In addition to the scripts, we provide one pdf per workflow for validation that conveniently combines source code, console output, and figures. The pdfs have been created using the publish command in Matlab.

B. Practical Treatment planning examples

Overall, we provide three practical treatment planning examples hosted on our wiki[9]. A key aspect of these practical examples is that they are consistently carried out using the graphical user interface and not the command line tool.

The first example depicts an intensity-modulated photon treatment on the phantom TG119. After the beam angles are determined, a maximum dose constraint is additionally defined to the existing default objectives. In the next step, optimization and analysis in form of a DVH and quality indicators are performed. The functionality 'Save To GUI' allows to store optimized dose distribution within the GUI in order to facilitate a comparison with a dose distributions based on different treatment plan settings.

The second more advanced practical treatment planning example aims to find an intensity-modulated photon beam setup and objective/constraint definitions that fulfill four given specifications. Exemplary, one of the four is to keep the total mean dose to the parotid glands below 20 Gy.

The third example is targeted to particle therapy and aims to create a simple intensity-modulated proton and carbon ion treatment plan. In the analysis, a comparison of the dose to the OAR, located adjacent at the distal edge of the spread out Bragg peak (SOBP), is performed.

[9] https://github.com/e0404/matRad/wiki/Practical-treatment-planning

IV. DISCUSSION & OUTLOOK

This paper introduces matRad with a particular focus on educational purposes and available training material. We describe various workflow examples and a treatment planning tutorial that have been made publicly available on our Github page.

Further, we outline the general setup of the matRad project on the GitHub webpage. In particular, we explain how to access the source code, retrieve additional information and how to contribute to matRad's development.

We aim to provide a toolkit with well-established and trusted algorithms to lower the burden for beginners entering this research area. Since treatment planning became a highly computerized process, one can observe that the ability to read and write computer code gets increasingly important. The clear software design and the intuitive Matlab language allows for rapid prototyping. Along with the relatively clean syntax for matrix computations, in comparison to other programming languages, this allows for a steep matRad learning curve, not only as users but also as active developers.

matRad 3.0.0 is the first version that supports seamless operation with Octave; the program code can now smoothly be executed without liability to pay for Matlab licenses. Moreover, the relative low hardware requirements make matRad a suitable tool for self-studying or teaching radiotherapy principles in a classroom setting in both low and high-income countries. The new workflow examples comprise only a few lines of code and demonstrate the full potential of matRad. The examples can be executed by everybody in a matter of several minutes due to optimized vectorized calculations.

As can been seen on the GitHub source code repository, matRad is under constant code development with an increasing number of contributing authors. In the near future, we are planning to include current experimental code for worst case, probabilistic and 4D treatment planning together with variable RBE models for protons in matRad's main release. In a collaborative effort with colleagues from Carleton University we are also working on the integration of optimization functionality for volumetric modulated arc therapy (VMAT).

V. CONCLUSIONS

The open-source software project matRad, which comprises dose calculation and optimization functionality for intensity-modulated radiation therapy with photons, protons, and carbon ions, is ideally suited for educational purposes.

Here we introduce seven workflow examples and a treatment planning tutorial to help students and new users to get started with radiation therapy treatment planning. This material is freely available online on our Github page, which further facilitates wide-spread use of our software.

We outline how matRad can be used independently from the proprietary numerical computation environment Matlab using GNU Octave. This allows to run matRad without any software costs - which may be particularly interesting for students and centers with budget limitations.

matRad not only significantly facilitates access to well established radiotherapy algorithms but also lowers the burden of entering into the research field of radiotherapy treatment planning.

VI. ACKNOWLEDGMENT

We gratefully acknowledge financial support by the German Research Foundation (DFG), grant No. BA 2279/3-1.

Further we want to thank the growing matRad community on GitHub in particular Eric Christiansen at Carlton University and Josefine Handrack at the German Cancer Research Center.

VII. REFERENCES

1. Atun R, Jaffray DA, Barton MB, et al (2015) Expanding global access to radiotherapy. Lancet Oncol 16(10):1153–1186
2. Avila LS, Barre S, Blue R, Geveci B, Henderson A, Hoffman WA, King B, Law CC, Martin KM, Schroeder WJ (eds) (2010) VTK File Formats. VTK User's guide. Kitware, Inc., New York, pp 469–506
3. Bortfeld T, Schlegel W, Rhein B (1993) Decomposition of pencil beam kernels for fast dose calculations in three-dimensional treatment planning. Med Phys 20(2 Pt 1):311–318
4. Cisternas E, Mairani A, Ziegenhein P, Jäkel O, Bangert M (2015) matRad - a multi-modality open source 3D treatment planning toolkit. In: Jaffray D (ed) IFMBE Proceedings. Springer, pp 1608–1611
5. Craft D, Bangert M, Long T, Papp D, Unkelbach J (2014) Shared data for intensity modulated radiation therapy (IMRT) optimization research: the CORT dataset. Gigascience 3(1):37
6. Deasy JO, Blanco AI, Clark VH (2003) CERR: a computational environment for radiotherapy research. Med Phys 30(5):979–985
7. Drzymala RE, Mohan R, Brewster L, Chu J, Goitein M, Harms W, Urie M (1991) Dose-volume histograms. Int J Radiat Oncol Biol Phys 21(1):71–78
8. Eaton JW, Bateman D, Hauberg S (2008) GNU Octave: A High-level Interactive Language for Numerical Computations. Network Theory.
9. Eaton JW, Bateman D, Hauberg S (2015) The Gnu Octave 4.0 Reference Manual 2/2: Free Your Numbers.
10. Kataria T, Sharma K, Subramani V, Karrthick KP, Bisht SS (2012) Homogeneity Index: An objective tool for assessment of conformal radiation treatments. J Med Phys 37(4):207–213
11. Low DA, Harms WB, Mutic S, Purdy JA (1998) A technique for

the quantitative evaluation of dose distributions. Med Phys 25(5):656–661

12. Pinter C, Lasso A, Wang A, Jaffray D, Fichtinger G (2012) SlicerRT: radiation therapy research toolkit for 3D Slicer. Med Phys 39(10):6332–6338

13. Riet AV, Mak ACA, Moerland MA, Elders LH, van der Zee W (1997) A conformation number to quantify the degree of conformality in brachytherapy and external beam irradiation: Application to the prostate. International Journal of Radiation Oncology*Biology*Physics 37(3):731–736

14. Sánchez-Parcerisa D, Kondrla M, Shaindlin A, Carabe A (2014) FoCa: a modular treatment planning system for proton radiotherapy with research and educational purposes. Phys Med Biol 59(23):7341–7360

15. Schaffner B, Pedroni E, Lomax A (1999) Dose calculation models for proton treatment planning using a dynamic beam delivery system: an attempt to include density heterogeneity effects in the analytical dose calculation. Phys Med Biol 44(1):27–41

16. Wächter A, Biegler LT (2006) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math Program 106(1):25–57

17. Wieser H-P, Cisternas E, Wahl N, et al (2017) Development of the open-source dose calculation and optimization toolkit matRad. Med Phys 44(6):2556–2568

18. Wilkens JJ, Oelfke U (2003) Analytical linear energy transfer calculations for proton therapy. Med Phys 30(5):806–815

19. Teem: nrrd. http://teem.sourceforge.net/nrrd/. Accessed 5 Oct 2017

20. ITK/MetaIO/Documentation - KitwarePublic. https://itk.org/Wiki/ITK/MetaIO/Documentation. Accessed 5 Oct 2017

Corresponding author: Hans-Peter Wieser
Institute: German Cancer Research Center DKFZ - Department of Medical Physics in Radiation Oncology
Street: Im Neuenheimer Feld 280
City: 69120 Heidelberg
Country: Germany
Email: h.wieser@dfkz.de