PortPy: An Open-Source Initiative for Radiotherapy Treatment Planning Optimization Including Benchmark Data and Algorithms and Integration with Commercial TPS

Gourav Jhanwar[1], Mojtaba Tefagh[2], Qijie Huang[1], Vicki Trier Taasti[3], Seppo Tuomaala[4], Saad Nadeem[1], and Masoud Zarepisheh[1]

[1]Department of Medical Physics, Memorial Sloan-Kettering Cancer Center, New York, NY, USA
[2]Sharif University of Technology, Tehran, Iran
[3]Department of Clinical Medicine - Danish Center for Particle Therapy, Aarhus University Hospital, Denmark
[4]Varian Medical Systems Inc, CA, USA

Abstract We have developed PortPy (Planning and Optimization for Radiation Therapy in Python), an innovative open-source software package designed to expedite the research and clinical translation of novel treatment planning techniques in radiation therapy. PortPy addresses the limitations of existing packages by offering several key features: 1) Interoperability with Commercial Systems: PortPy is designed to work seamlessly with a commercial treatment planning system, Eclipse. This interoperability allows for the objective evaluation of new techniques in comparison to the current best clinical practices. 2) Inclusion of Benchmark Dataset: The package includes a benchmark dataset, currently 50 lung patients, complete with clinical plans and all necessary data for treatment planning optimization, such as voxels, beamlets, and the dose influence matrix. This data is extracted from Eclipse using its API, facilitating comprehensive and realistic testing scenarios. 3) Advanced Benchmark Algorithms: PortPy encompasses benchmark algorithms capable of identifying "global" optimal solutions, using computationally intensive mixed integer programming, for various non-convex optimization problems encountered in treatment planning. This includes challenges such as Dose Volume Histogram (DVH) constraints, direct machine parameter optimization for Volumetric Modulated Arc Therapy (VMAT), and beam orientation optimization for Intensity Modulated Radiation Therapy (IMRT). PortPy enables the design, testing, and clinical validation of both classical optimization-based and cutting-edge AI-based treatment planning techniques. It promotes transparency, reproducibility, and community-driven development in the field of radiation therapy treatment planning.

1 Introduction

The importance of open-source packages in advancing research, promoting transparency, and encouraging community-driven development cannot be overstated. In the field of radiotherapy treatment planning, existing Matlab-based open-source packages like CERR [1] and MatRad [2] have made significant contributions. The rising popularity of Python as a free, open-source programming language has inspired the development of Python-based packages such as OpenTPS, focusing on proton treatment planning, and PyMedPhys, mainly aimed at machine and plan quality assurance (QA). However, a key limitation of these existing packages is their inability to objectively evaluate new treatment planning techniques against current clinical practices. This limitation stems from the difficulty in fully replicating clinical environments, such as detailed linear accelerator configurations and commercial dose calculation engines. Consequently, plans generated using novel techniques, such as AI-based automated treatment planning, cannot be objectively compared with clinical plans used in patient treatment. We seek to overcome this barrier with the development of PortPy (Planning and Optimization for Radiation Therapy in Python), a new Python-based open-source package. PortPy is currently compatible with the Varian Eclipse treatment planning system (TPS), and future expansions to other TPSs are planned. It is important to emphasize that PortPy can be utilized without access to any TPS. However, access to a TPS is necessary if a final evaluation within the TPS is desired. Furthermore, PortPy addresses the absence of a benchmark dataset and algorithms in current packages, providing a crucial tool for researchers to evaluate novel techniques.
Figure 1 illustrates the inspiration behind PortPy's development, drawing from successful open-source practices in the AI and computer science community. Open-source tools such as PyTorch and TensorFlow, alongside benchmark datasets like ImageNet, and benchmark algorithms like AlexNet, have revolutionized AI and data science. Our goal is to replicate this successful model, equipping researchers with comprehensive tools (PortPy integrated with commercial TPS and the popular 3DSlicer imaging package), a benchmark dataset (currently 50 curated lung patients with expert-selected beams, all necessary optimization data, and benchmark IMRT plans generated using our in-house automated planning system, ECHO), and benchmark algorithms (including Mixed Integer Programming [MIP] algorithms for global optimal solutions).

2 Materials and Methods

Figure 2 illustrates the PortPy design and its three main modules: "Data Management", "Plan Generation", and "Plan Evaluation", which are discussed in the next sections.

2.1 Data Management

This module provides access to the curated benchmark PortPy dataset, which currently comprises data from 50 lung patients.

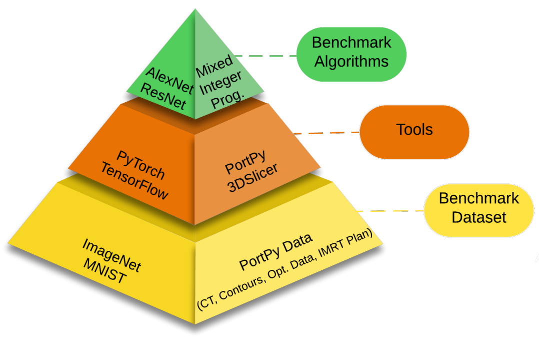


Figure 1: The development of PortPy has been inspired by the recent advancements in AI, facilitated by new tools, benchmark datasets, and algorithms. This analogy is depicted in this figure.

It allows researchers to test and develop their algorithms using the same dataset. The available data includes: 1) CT images and contours, 2) all necessary data for optimization for 72 uniformly selected beams (e.g., beamlets, voxels, dose influence matrix) extracted from Eclipse (version 16.1), 3) expert-selected beams for each patient, 4) an IMRT plan for each patient, generated using our in-house automated planning system, ECHO [3] (Expedited Constrained Hierarchical Optimization). ECHO is integrated with the Eclipse system and is currently in routine use in our clinic, having treated over 10,000 patients to date. It is important to note that the plans included in our dataset are not the clinical plans used for actual patient treatment, and therefore have not undergone the routine clinical scrutiny. However, they serve as valuable benchmark plans for research purposes.

2.2 Plan Generation

This module facilitates the generation of treatment plans using either classical optimization methods or emerging AI-based techniques. For optimization tasks, PortPy has been integrated with CVXPy [4], a widely-used open-source package. CVXPy enables the high-level formulation of optimization problems and offers out-of-the-box access to a range of of free (e.g., SCIP, SCIPY) and commercial (e.g., MOSEK, CPLEX) optimization engines (available for free for research purposes). Additionally, the PortPy.AI module is equipped with essential functionalities for AI-based planning. These include data access, data pre-processing, model training and testing, and patient-specific 3D dose prediction. The availability of both optimization and AI-based planning modules within PortPy allows researchers to not only compare these techniques but also explore their complementary aspects (Use Case 3, Section 3.3).

2.3 Plan Visualization and Evaluation

PortPy is equipped with basic built-in visualization tools, such as Dose-Volume Histograms (DVH) and dose distribution maps, as depicted in Figures 3, 4, 5. For more advanced visualizations, it integrates seamlessly with the popular open-source 3DSlicer package. When it comes to plan evaluation, PortPy provides a suite of methods to quantify optimized plans using clinically relevant metrics (for example, mean, maximum, DVH) and to compare these plans against established clinical criteria, as shown in Figure 4. Additionally, PortPy enables the export of plans in the DICOM RT plan format, allowing users to import these plans into any commercial TPS for final dose calculation and assessment. Figure5 illustrates how the results from PortPy align closely with Eclipse after importing the plan in Eclipse and running final dose calculation. It should be highlighted that although DICOM RT plan can be imported into various TPS, discrepancies are anticipated with TPSs other than Eclipse, due to the fact that our current data are derived from Eclipse, and there may exist variations in dose calculation methods between Eclipse and other TPSs. For IMRT, PortPy also supports exporting the optimal fluence map in a format compatible with Eclipse, facilitating the import into Eclipse for final leaf sequencing and dose calculation.
Furthermore, PortPy includes a collection of benchmark algorithms, useful for evaluating novel treatment planning optimization algorithms. These algorithms employ mixed integer programming (MIP) and are able to identify the best global optimal solutions for non-convex problems, including DVH constraints, VMAT, and beam angle orientations. While these algorithms are highly computationally demanding (typically taking days or weeks to solve) and not suited for real clinical application, they can be run on powerful computers and in the absence of time constraints to find global optimal solutions. These solutions then serve as benchmarks to validate new, computationally efficient algorithms (see Use Case 2, Section 3).

3 Results

In this section, we present three use cases to demonstrate the capabilities of PortPy. Additional examples are available on our GitHub page, provided in user-friendly Jupyter Notebook formats for ease of access and experimentation.

3.1 Use case 1: Developing and validating new treatment planning algorithms

The details of this simple and educational use case are provided in a Jupyter Notebook available on our GitHub page. To illustrate how researchers can develop new treatment planning techniques and validate them using PortPy, we performed the following tasks:

1. Utilizing the "Data Management" module, we loaded a lung patient case from the PortPy dataset, using the expert selected beams, which are also included in the dataset. We then employed the "Plan Generation" module and "Optimization" class to create a new IMRT plan using constraint optimization. We applied max/mean

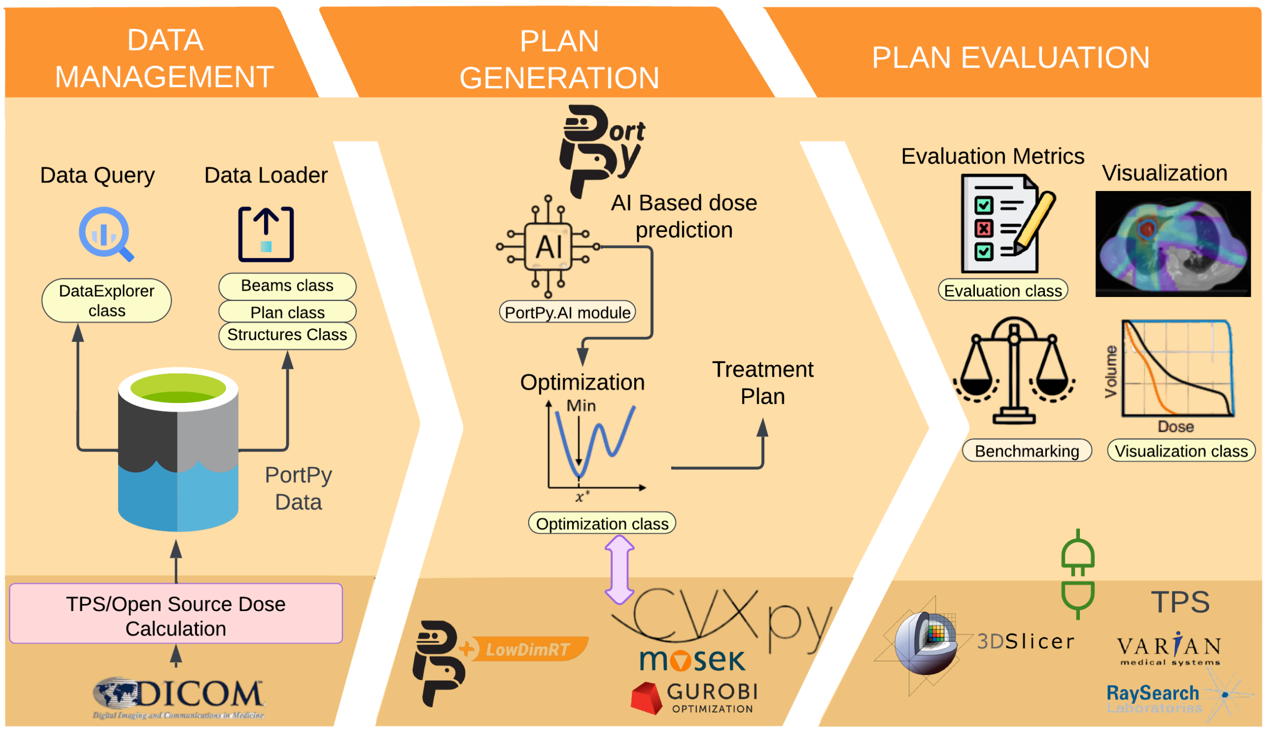


Figure 2: (PortPy workflow) PortPy comprises three primary modules: 1) "Data Management" for accessing the benchmark dataset, 2) "Plan Generation" for creating treatment plans using either classical optimization techniques or AI-based planning, and 3) "Plan Evaluation" for visualizing and assessing the generated plans.

hard constraints in line with our clinical criteria and developed a composite quadratic objective function, adjusting the weights manually. The integration with CVXPy facilitated the formulation of this optimization problem. Subsequently, we asked CVXPy to use the commercial optimization engine MOSEK (for which we obtained a free academic license) for solving the problem. Following this, we utilized PortPy's leaf sequencing algorithm to segment the resulting optimal fluence. Alternatively, the optimal fluence can be used without leaf sequencing.

2. The "Plan Evaluation" module and "Visualization" class were used to display the dose distribution and fluence profile of a beam in PortPy (see Figure 3). We then utilized the "Evaluation" class for a quantifiable assessment of the plan against our clinical criteria (refer to Figure 4).

3. We imported the DICOM RT plan file generated by PortPy, including the optimal control points, into Eclipse for the final dose calculation. Figure 5 shows that the results in Eclipse and PortPy are aligned, with minor discrepancies attributable to sampling inaccuracies and specific details regarding scattering and leaf transmissions.

4. We conducted a comparison of the plan with the PortPy IMRT benchmark plan (shown in Figure 6), which was generated using our in-house automated planning system, ECHO.

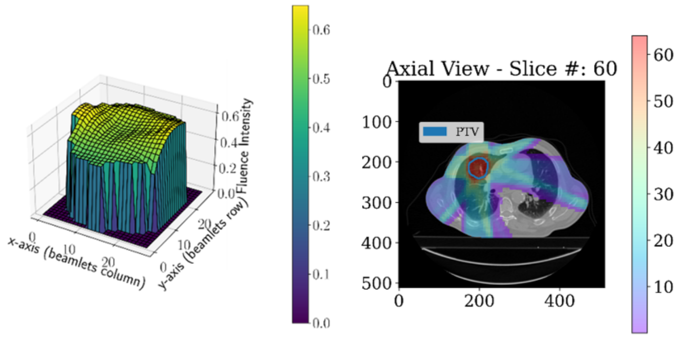
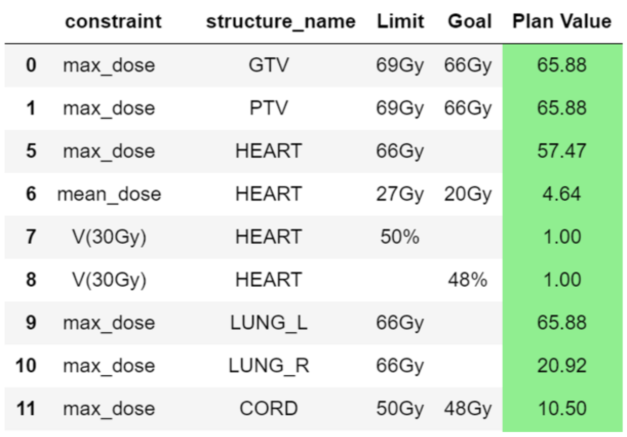


Figure 3: PortPy enables visualization of the generated plans, and for advanced visualizations, it offers integration with 3DSlicer and commercial treatment planning systems.



	constraint	structure_name	Limit	Goal	Plan Value
0	max_dose	GTV	69Gy	66Gy	65.88
1	max_dose	PTV	69Gy	66Gy	65.88
5	max_dose	HEART	66Gy		57.47
6	mean_dose	HEART	27Gy	20Gy	4.64
7	V(30Gy)	HEART	50%		1.00
8	V(30Gy)	HEART		48%	1.00
9	max_dose	LUNG_L	66Gy		65.88
10	max_dose	LUNG_R	66Gy		20.92
11	max_dose	CORD	50Gy	48Gy	10.50

Figure 4: PortPy allows for the evaluation of generated plans through quantifiable metrics, enabling comparisons with established clinical criteria. The color green is used to indicate areas where limits and goals are met.

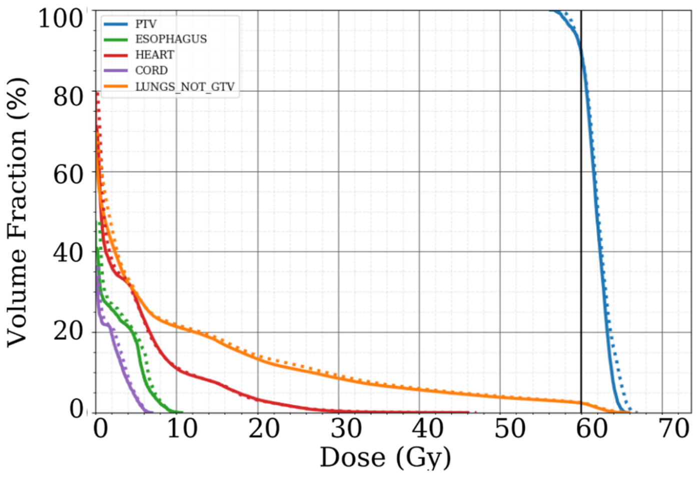


Figure 5: The DVH curves generated in PortPy (solid lines) for the optimized plan closely align with the DVH curves generated in Eclipse (dotted lines) following the final dose calculation

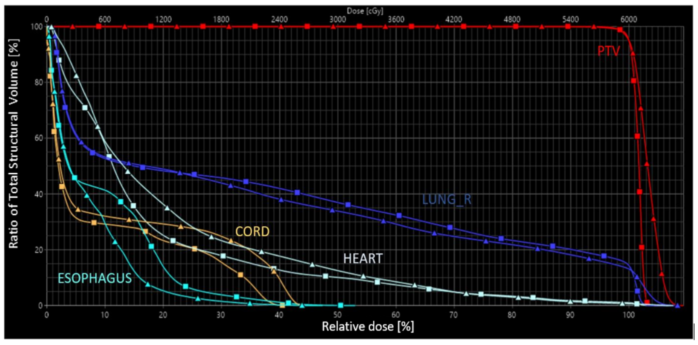


Figure 6: The plan generated in PortPy (lines with square marks) is imported into Eclipse and compared against the benchmark IMRT plan that is available in the PortPy dataset (lines with triangle marks). The benchmark plan in the dataset was generated using our in-house automated planning system, ECHO.

3.2 Use case 2: Bench-marking novel algorithms against the global optimal solutions

Certain non-convex optimization problems common in this field can be solved to global optimality through proper formulation using Mixed Integer Programming (MIP), which is notably computationally intensive. Figure 7 displays plans generated using both expert-selected beams, as provided in the dataset, and globally optimal beams. The global optimal beams were determined by solving the MIP-based problem, a process that required over four days of computation on our system. These plans can act as benchmarks for validating new, more computationally efficient algorithms for beam orientation optimization. Additionally, PortPy includes global optimal solutions for both VMAT and DVH constraints.

3.3 Use case 3: AI-based automated planning

The PortPy.AI module facilitates the end-to-end development and validation of innovative AI-based planning techniques. For instance, it enables the training of deep learning models using the PortPy benchmark dataset and benchmark IMRT

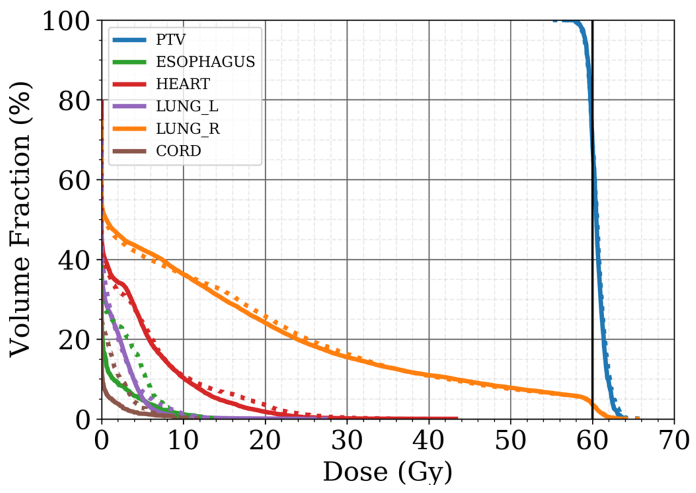


Figure 7: DVH curves generated with expert-selected beams (solid lines) are compared to those from globally optimal beams (dashed lines), the latter obtained by solving a mixed integer programming problem (> 4 days to solve).

plans. Subsequently, these models can predict the 3D dose distribution for a new patient. Following this prediction, the "Optimization" module can be employed for dose mimicking optimization, and the optimal control points (or fluence map) can be imported into a TPS for final evaluation. Illustrative workflows and examples are provided in our GitHub repository's Jupyter Notebooks.

4 Discussion and Conclusion

PortPy has been developed to address the limitations of the current packages, with a primary aim to advocate transparency, reproducibility, and foster community-driven development within the realm of radiotherapy treatment planning. PortPy stands as an open-source community project, under continuous enhancement, and we invite contributions and involvement from the broader scientific community. PortPy has been downloaded over 7,000 times in the past four months, a testament to its adoption within the scientific community.

References

[1] J. O. Deasy, A. I. Blanco, and V. H. Clark. "CERR: a computational environment for radiotherapy research". *Medical physics* 30.5 (2003), pp. 979–985.

[2] H.-P. Wieser, E. Cisternas, N. Wahl, et al. "Development of the open-source dose calculation and optimization toolkit matRad". *Medical physics* 44.6 (2017), pp. 2556–2568.

[3] M. Zarepisheh, L. Hong, Y. Zhou, et al. "Automated and clinically optimal treatment planning for cancer radiotherapy". *INFORMS journal on applied analytics* 52.1 (2022), pp. 69–89.

[4] S. Diamond and S. Boyd. "CVXPY: A Python-embedded modeling language for convex optimization". *The Journal of Machine Learning Research* 17.1 (2016), pp. 2909–2913.