Benchmark and Problem Bank for OptimalControl.jl Package

Yassine Hajem

***Abstract*—This paper aims to benchmark OptimalControl.jl, a Julia-based package for solving optimal control problems, by comparing it to the established JuMP package. To achieve this, a problem bank, OptimalControlProblems.jl, was developed, enabling fair comparisons between the two solvers across identical setups and evaluating key metrics such as solution accuracy and computational speed. The results indicate that OptimalControl.jl performs well in handling complex optimal control problems, though there are areas where it could be improved. This benchmark highlights OptimalControl.jl's potential in applications requiring optimal control and offers a foundation for guiding further development and enhancements.**

***Keywords*—Optimal Control, Benchmarking, Julia Programming, Control Toolbox, Computational Efficiency, Solver Comparison**

I. Introduction

Optimal control solvers are central to many engineering, economics, and system optimization tasks, where the goal is to determine a control strategy that minimizes or maximizes a given performance criterion (cost function) while satisfying dynamic constraints. These problems require sophisticated computational tools for their solution.

In recent years, Julia [1] has emerged as a high-performance programming language particularly well-suited for complex and scientific computing due to its speed and ease of use. Among the packages available in Julia, ControlToolbox [2] is a versatile library designed to modelize various optimal control problems. it provides a range of tools and algorithms for formulating, analyzing, and solving control problems.

The goal of the project was twofold: First, to create a diverse repository of test cases (problem bank) that can be used for future evaluations and enhancements of the package in Sec. II. Second, to rigorously assess the efficiency, accuracy, and robustness of the package when solving different types of optimal control problems in Sec. III. The benchmarks aim to provide insights into the package's strengths and limitations.

II. The Problem bank : OptimalControlProblems

To establish a robust framework for evaluating Julia-based approaches to solving optimal control problems, this study compares the well-established JuMP [3] package with the newer OptimalControl.jl solver from ControlToolbox. To facilitate this, a dedicated problem bank, called OptimalControlProblems.jl, has been developed, containing various real-world problems. This package allows for a direct, fair comparison of each framework's performance across identical problem.

Each problem within the OptimalControlProblems.jl package has been formulated to address realistic challenges. The problems are implemented in both JuMP and OptimalControl.jl syntax to facilitate side-by-side comparisons of solver's speed and accuracy. Additionally, each problem includes metadata attributes—such as problem name, default number of discretization points, and optimization objective (minimization or maximization) —to simplify problem selection, management, and filtering based on specific parameters.

However, it is important to note that while all problems in the problem bank have been written in both formats, some problems are not yet fully solvable within OptimalControl.jl. As the package evolves and incorporates new features and capabilities, it is expected that the complete problem set will eventually become solvable within OptimalControl.jl. This progressive enhancement will allow for an even more accurate comparison between the two packages. Table I below summarizes the current status of each problem's solvability with JuMP and OptimalControl.jl, highlighting areas where future updates to OptimalControl.jl could broaden the scope of the comparison.

Problem	With JuMP	With OptimalControl
beam	✓	✓
bioreactor	✓	✓
cart_pendulum	✓	✓
chain	✓	✓
dielectrophoretic_particle	✓	✓
double_oscillator	✓	✓
ducted_fan	✓	✓
electrical_vehicle	✓	✓
glider	✓	✓
insurance	✓	✓
jackson	✓	✓
moonlander	✓	X
quadrotor	✓	X
robbins	✓	✓
robot	✓	✓
rocket	✓	✓
space_shuttle	✓	X
steering	✓	✓
truck_trailer	X	X
vanderpol	✓	✓

TABLE I

List of Problems in OptimalControlProblems and its feasibility with JuMP and OptimalControl [4]

III. The Benchmark Package : OptimalControlBenchmarks

To evaluate the performance of these solvers, two main metrics were examined: accuracy and computational speed.

A. Accuracy evaluation

In optimal control applications, accuracy is critical, as even small deviations from the optimal solution can lead to significant issues. Accurate solutions ensure that objectives are achieved while constraints are respected, which is particularly important in high-stakes domains where precise results are paramount. To assess accuracy, this study compares objective function values and state and control functions between JuMP and OptimalControl.jl across various problems. For each problem, a dedicated notebook was created to outline the setup and execution of both solvers.

First, each problem from OptimalControlProblems.jl was called in both OptimalControl.jl and JuMP syntax with identical parameters for discretization points and solver settings. This approach ensures that the comparison between the two solvers reflects differences in solution accuracy, rather than variations in problem setup. Each problem was solved using the same nonlinear solver (Ipopt [5]) and linear solver (HSL MA57 [6]). Visualizations were then generated to compare accuracy aspects, such as the state and co-state trajectories and the control function for each solver. Additionally, the iterative steps of each solver were documented to demonstrate convergence progress, illustrating how effectively each solver minimizes residuals at each step.

B. *Speed evaluation*

In time-sensitive applications, especially those involving real-time control, rapid computation of optimal control solutions is essential. To assess the practical responsiveness of OptimalControl.jl and JuMP packages, computational speed was evaluated across each problem in OptimalControlProblems.jl under varying levels of discretization. This setup enabled a clear comparison of each solver's capability to handle increased complexity.

Problems were identically configured to ensure that differences in speed were attributable to solver performance, not problem formulation. Julia's native timing tools were used to measure total solution time, specifically leveraging the @time and @btime macros from the BenchmarkTools.jl [7] package. @time provides a quick estimate of execution time and memory usage, while @btime runs multiple evaluations. These tools provided a comprehensive view of the computational demands of each solver, revealing insights into performance and scalability.

The benchmarking process was automated, allowing users to specify only discretization ranges, which each solver then executed across the problems. The speed of solving was automatically recorded, and the results compiled into a PDF file for easy review and comparison(an example of the benchmark output is shown below in Figure 1). This automated setup ensured consistency and accuracy across tests.

In summary, the comparison framework developed here provides a structured, consistent basis for evaluating OptimalControl.jl against JuMP across diverse problems and performance metrics.

Benchmark models with JuMP and OptimalControl
max iter = 1000, tol = 1.0e-8, constr viol tol = 1.0e-6, solver = ma57

Model	Discretization	Total Time JuMP	Total Time OC	Iterations JuMP	Iterations OC	Allocations JuMP(1e6)	Allocations OC(1e6)
steering	100	0.02	0.05	11	11	4.3	9.4
steering	400	0.08	0.2	17	17	18.0	53.8
steering	500	0.11	0.2	17	17	22.1	67.2
steering	1000	0.28	0.44	18	18	44.4	141.1
rocket	100	0.05	0.1	19	21	9.9	26.5
rocket	400	0.24	5.12	40	79	39.3	410.8
rocket	500	0.53	8.78	59	77	50.3	484.4
rocket	1000	0.36	27.07	24	54	97.2	700.5
chain	100	0.01	0.03	7	7	3.4	5.6
chain	400	0.02	0.08	7	8	13.4	24.7
chain	500	0.05	0.16	8	14	16.9	51.2
chain	1000	0.07	0.17	6	6	34.0	48.3
electrical_vehicle	100	0.01	0.06	5	16	2.7	21.0
electrical_vehicle	400	0.01	0.69	5	17	10.6	103.4
electrical_vehicle	500	0.05	13.78	5	1000	13.3	4001.5
electrical_vehicle	1000	0.1	40.28	5	370	27.3	4195.1

Fig. 1. Benchmark Results

IV. Conclusion

In conclusion, this project explored the benchmarking of OptimalControl.jl, comparing it to JuMP for solving various optimal control problems. By creating a problem bank and automating performance evaluation, both solvers were tested for accuracy and speed under consistent setups. The benchmarks showed where OptimalControl.jl shines and where it needs work, giving a solid foundation for future updates.

V. Feedback

During this experience, my performance was positively reviewed, particularly in terms of my analytical skills, commitment, and collaborative abilities. According to my tutor's evaluation, I demonstrated a high level of proficiency in benchmarking and in developing an accessible tools. My ability to model, test, analyse, and document autonomously was noted as "advanced".

In self-reflection, I would assess my strengths similarly, acknowledging good analytical abilities and adaptability in real-world scenarios. My project management skills, as reflected in structuring and documenting complex comparisons, highlight my ability to face challenging tasks into manageable pieces while maintaining clarity and focus.

For improvement, I aim to enhance my communication skills, especially in presenting technical concepts in a way that's clear and accessible to diverse audiences. Improving in this area will help me convey complex insights more effectively, which I believe is essential for successful collaboration and shared understanding in team settings.

References

[1] "Julia," *https://julialang.org/*.

[2] "Controltoolbox," *https://control-toolbox.org/*.

[3] "Jump," *https://jump.dev/*.

[4] "List of problems," *https://control-toolbox.org/OptimalControlProblems.jl/stable/list_of_problems.html#List-of-Problems-2*.

[5] "Ipopt documentation," *https://coin-or.github.io/Ipopt/*.

[6] "Libhsl," *https://licences.stfc.ac.uk/product/libhsl*.

[7] "Benchmarktools," *https://juliaci.github.io/BenchmarkTools.jl/*.