

Spring 2021



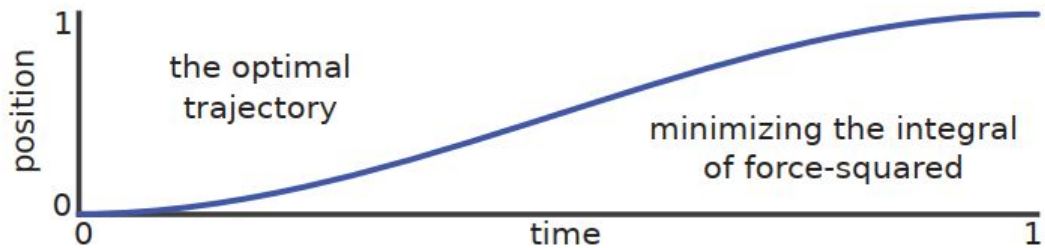
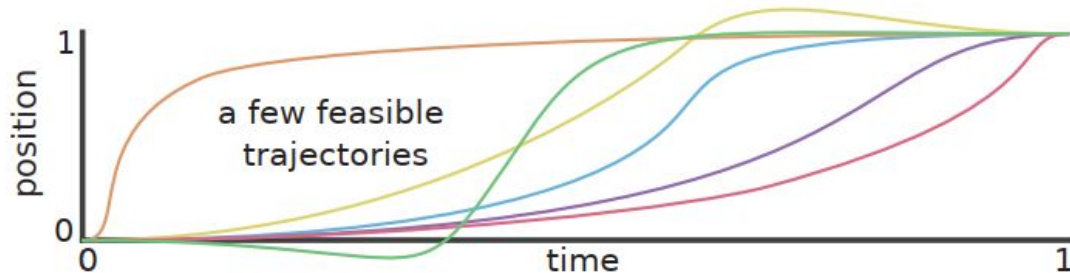
Pydcol: Control of ODE Systems with “Minimal Effort”

May 6th, 2021

Authors: John D’Angelo & Shreyas Sudhaman

Overview

- Introduction
- Methodology
- Examples
- Demonstration & Documentation
- Conclusion



Ref. [2]

Introduction

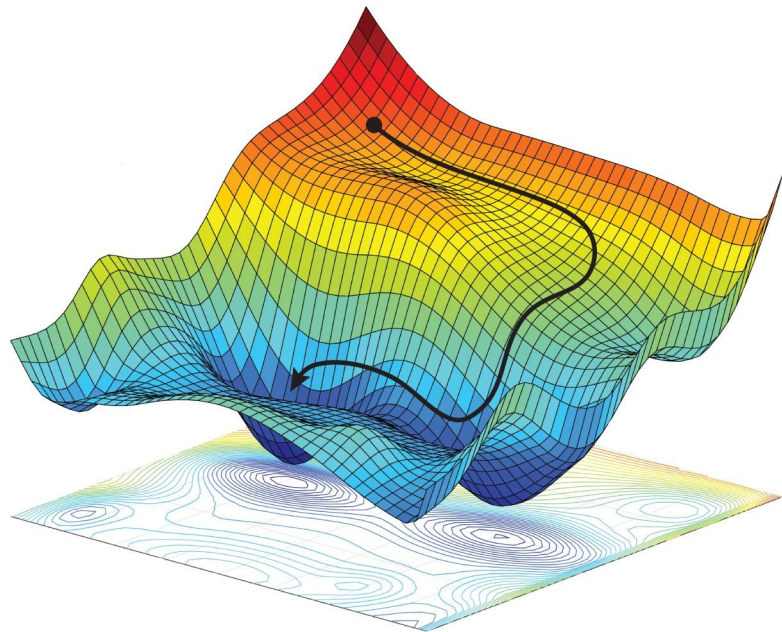
- pydcol automates direct collocation problem for
 - ODE systems: $\dot{X} = f(X, u)$
 - Fixed final time and state
 - Objective: Minimal control effort:

$$\min_{X, u} \int_{t_0}^{t_f} u^2 dt$$

$$s. t. X(t_0) = X_{start}, X(t_f) = X_{goal}$$

Introduction

- Simultaneous Discretization
- Non-linear Program (NLP)
- Design a library to handle
 - Symbolic manipulation
 - Multiple integration methods
 - Choice of optimizer



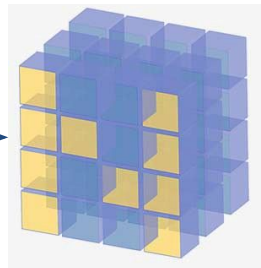
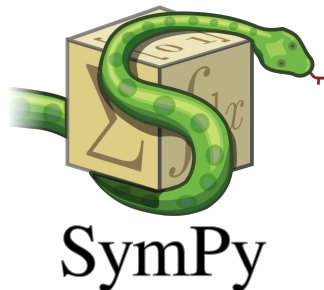
Ref. [4]

Methodology

Sym Objective
Sym Constraints
Sym Derivatives

Num Objective
Num Constraints
Num Derivatives

$$\begin{aligned}
 \dot{X} &= f(X, U) \\
 X(t_0), X(t_f) \\
 N
 \end{aligned}$$



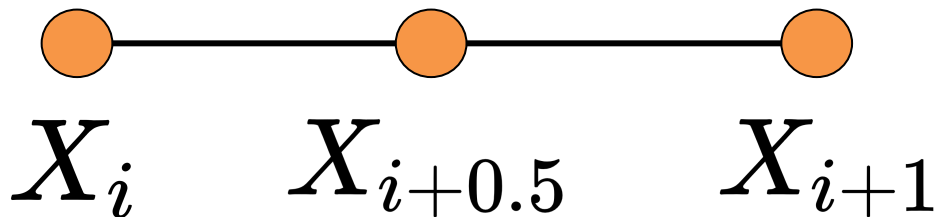
NLP Solver

User Input:
 Problem Definition

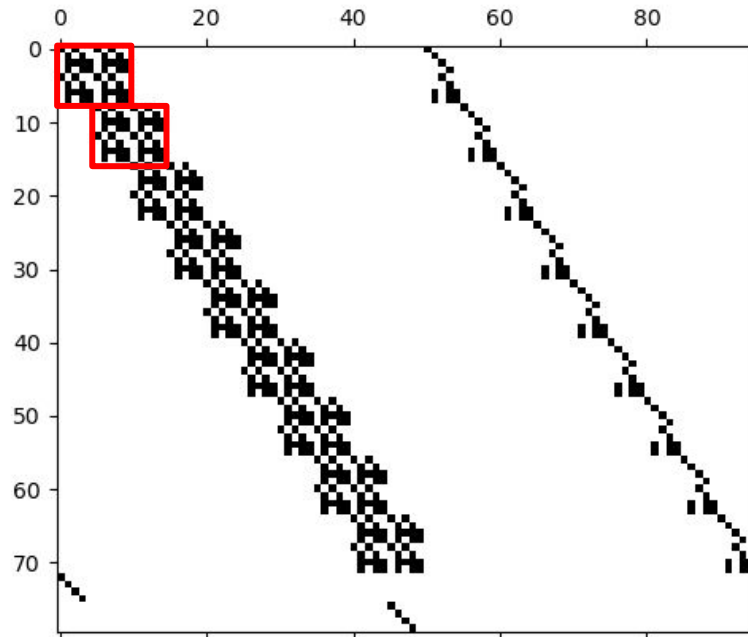
Output: Locally-
 Optimal X,U
 Trajectory

Methodology

- General equation for a node used instead of writing N equations
- Reduces computational cost of Jacobian and Hessian



Sparsity Pattern for Jacobian of Equality Constraints

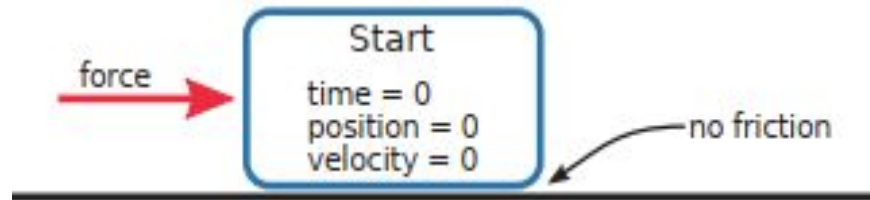
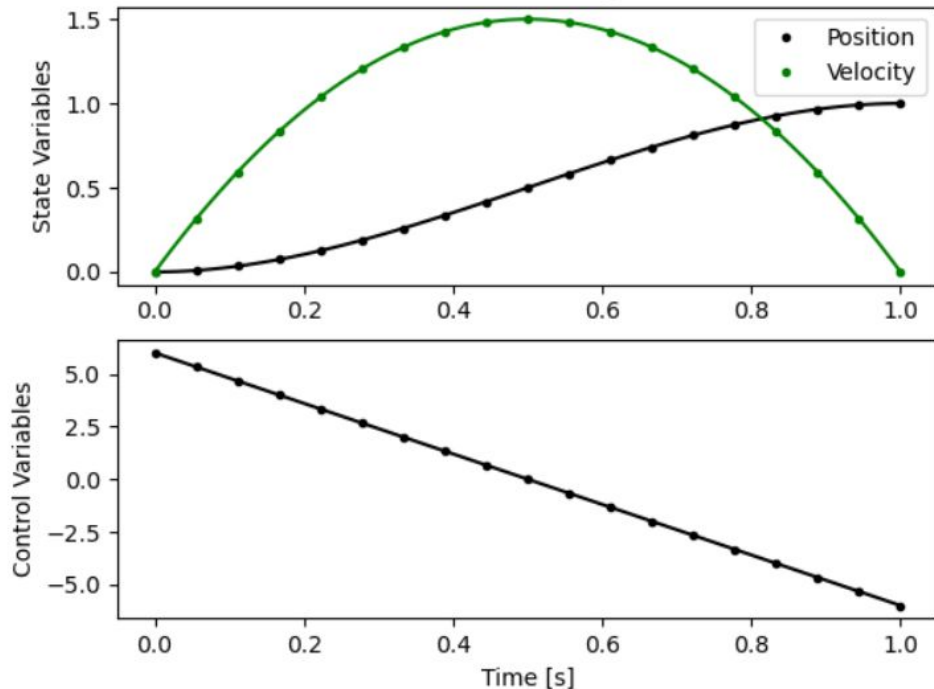


Examples

- A set of test problems was selected (sorted by difficulty to solve):
 - Box move (linear)
 - Cartpole swing-up (slightly nonlinear)
 - Double pendulum swing-up (very nonlinear)
 - Lunar lander (stiff, multiple inputs)

Box Move

Collocation Points vs. Integration Results

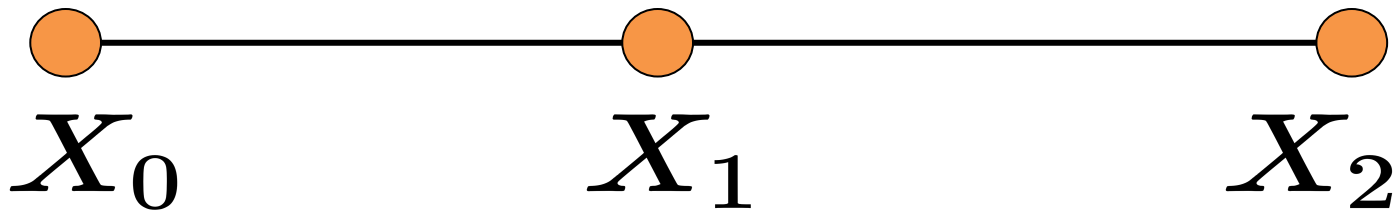


Ref. [2]

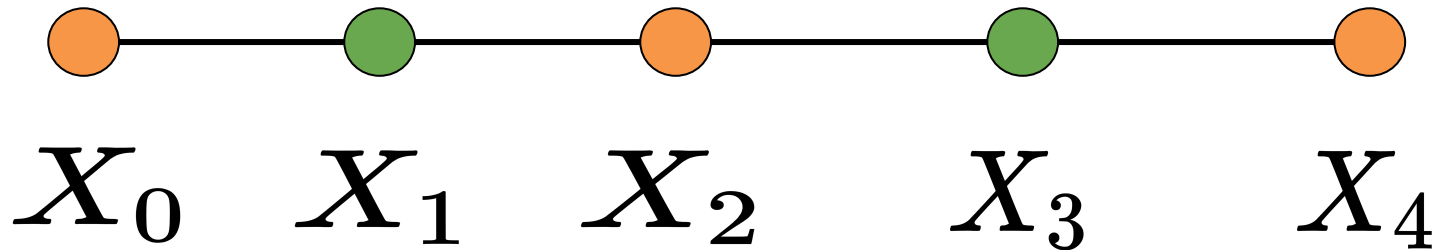
Dots = Collocation, Lines = IVP

Error Analysis

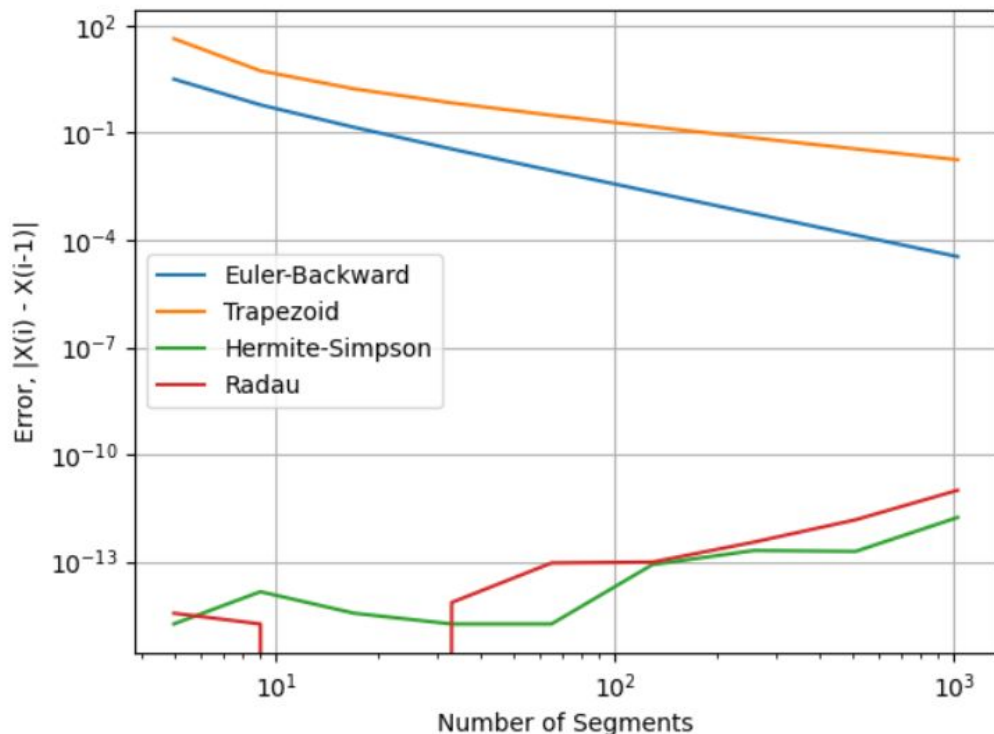
Iteration 1:



Iteration 2:



Box Move - Revised Error Analysis



$$e_i = |Obj_i - Obj_{i-1}|$$

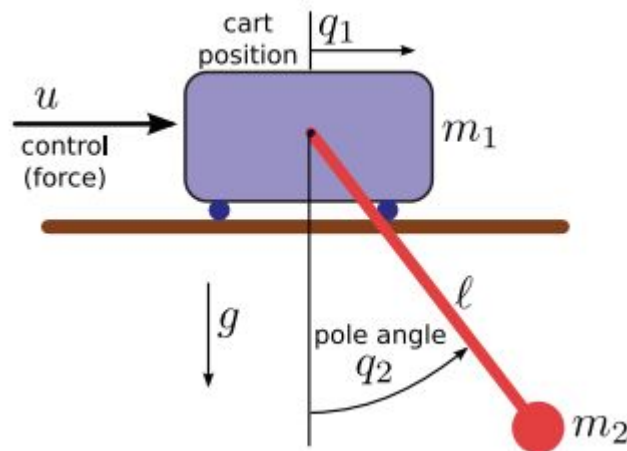
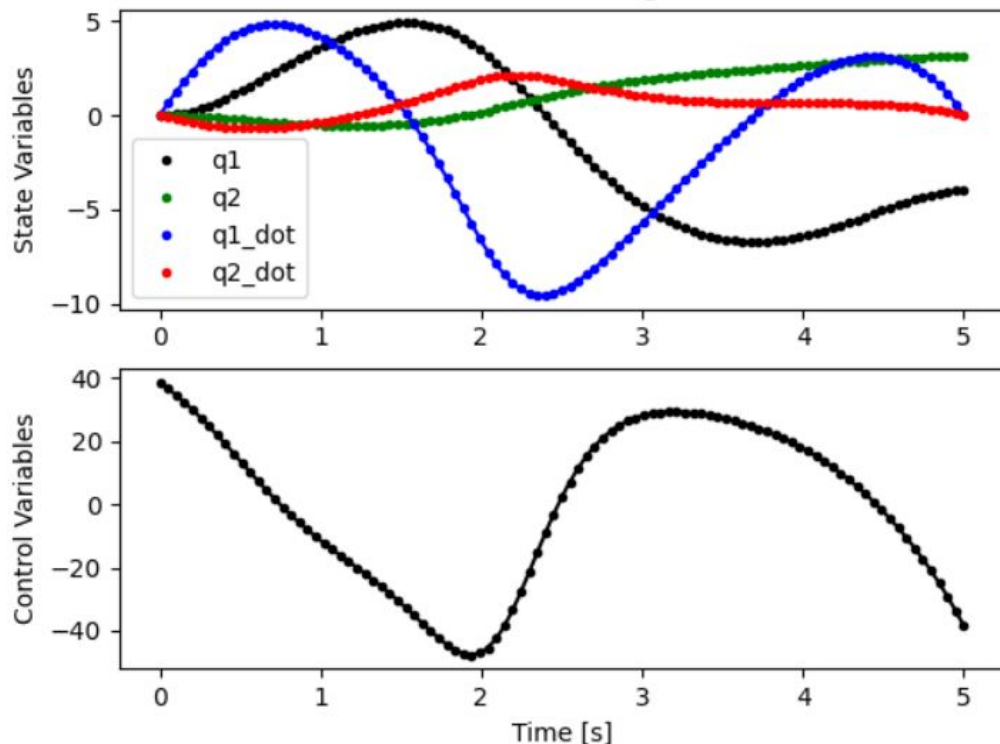
$$O(e) \approx \log\left(\frac{e_{i-1}}{e_i}\right) / \log(2)$$

Integration Method	Estimated Order of Accuracy
Euler Backward	2
Trapezoid	1
*Hermite-Simpson	N/A
*Radau IIA	N/A

*Radau and Hermite-Simpson converged too quickly for this analysis to be meaningful.

Cartpole Swing-up

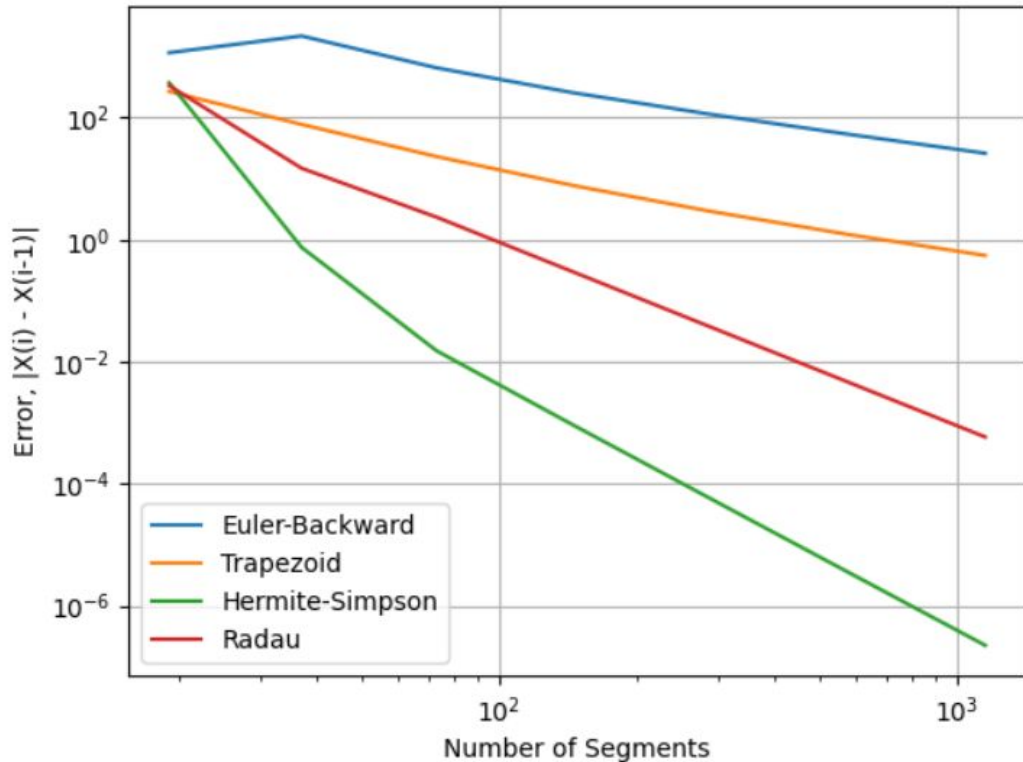
Collocation Points vs. Integration Results



Ref. [2]

Dots = Collocation, Lines = IVP

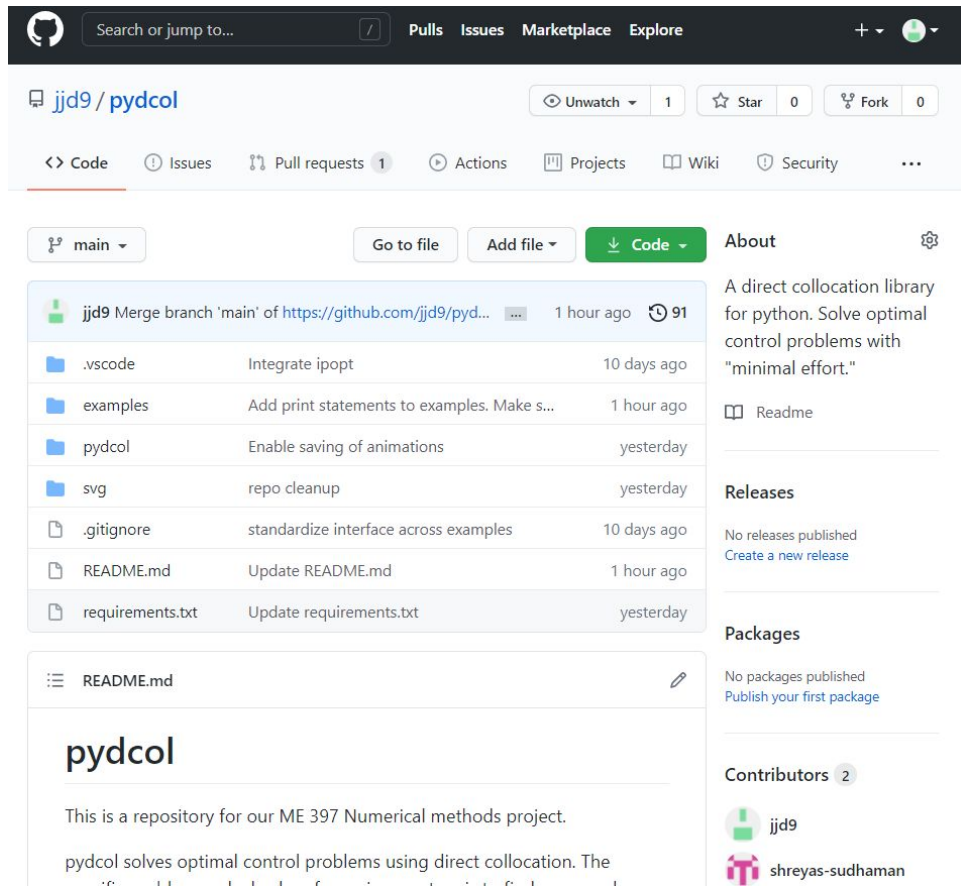
Cartpole - Revised Error Analysis



Integration Method	Estimated Order of Accuracy
Euler Backward	1
Trapezoid	1
Hermite-Simpson	4
Radau IIA	3

Documentation

- The documentation and source code for pydcol are available on Github
- We welcome feedback through issues or pull requests



The screenshot shows the GitHub repository page for `jjd9/pydcol`. The repository is in the `main` branch. The file list includes `.vscode`, `examples`, `pydcol`, `svg`, `.gitignore`, `README.md`, and `requirements.txt`. The `README.md` file is selected, showing the title `pydcol` and the description: "This is a repository for our ME 397 Numerical methods project. pydcol solves optimal control problems using direct collocation. The".

Repository: `jjd9 / pydcol`

Navigation: `<> Code` `Issues` `Pull requests 1` `Actions` `Projects` `Wiki` `Security`

Branch: `main` `Go to file` `Add file` `Code`

Recent commits:

File	Commit Message	Time
<code>.vscode</code>	Integrate ipopt	10 days ago
<code>examples</code>	Add print statements to examples. Make s...	1 hour ago
<code>pydcol</code>	Enable saving of animations	yesterday
<code>svg</code>	repo cleanup	yesterday
<code>.gitignore</code>	standardize interface across examples	10 days ago
<code>README.md</code>	Update README.md	1 hour ago
<code>requirements.txt</code>	Update requirements.txt	yesterday

About: A direct collocation library for python. Solve optimal control problems with "minimal effort."

Releases: No releases published. [Create a new release](#)

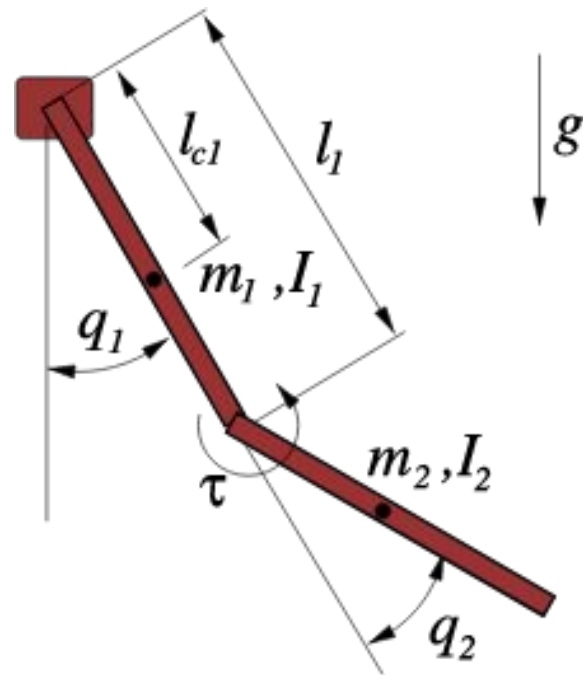
Packages: No packages published. [Publish your first package](#)

Contributors: 2

- `jjd9`
- `shreyas-sudhaman`

Double Pendulum Swing-up

Go to the code demo!



Ref. [1]

Conclusions

- pydcol is a tool for solving a common variation of the optimal control problem
- Our testing showed that pydcol's hermite-simpson method is well suited for optimal control of mechanical systems
- Developing this tool yielded some practical insights into direct collocation (how you integrate the objective matters, sparsity is great but gets complex quickly, etc.)

References

- [1] Russ Tedrake. Underactuated Robotics: Algorithms for Walking, Running, Swimming, Flying, and Manipulation (Course Notes for MIT 6.832). Downloaded on [date] from <http://underactuated.mit.edu/>
- [2] Kelly, M. An Introduction to Trajectory Optimization: How to Do Your Own Direct Collocation. SIAM Rev. 2017, 59 (4), 849–904.
<https://doi.org/10.1137/16M1062569>
- [3] Assignment 4: Lunar Lander Solution
https://web.aeromech.usyd.edu.au/AMME3500/Course_documents/material
- [4] A. Amini et al. “Spatial Uncertainty Sampling for End-to-End Control”. NeurIPS Bayesian Deep Learning 2018.

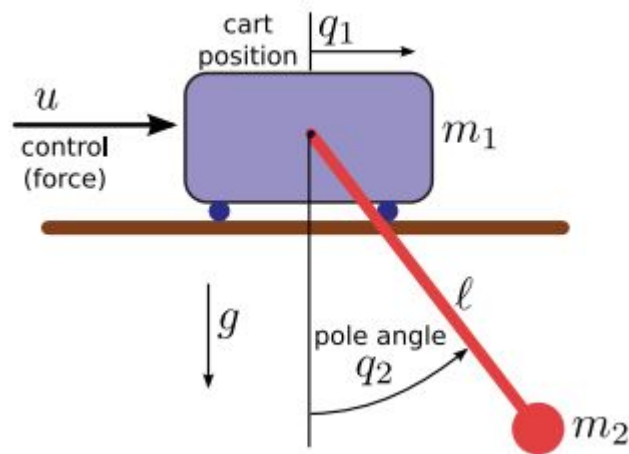
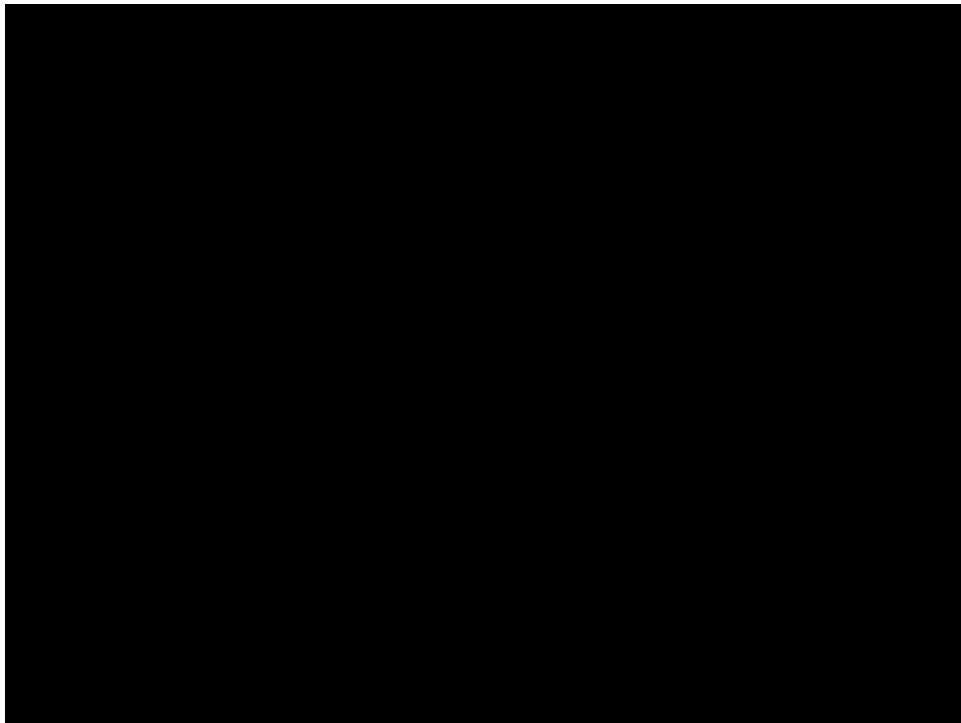
Acknowledgement

Thank you to Professor Subramanian for his instruction in this course and his assistance with this project.

Thank you!

Questions?

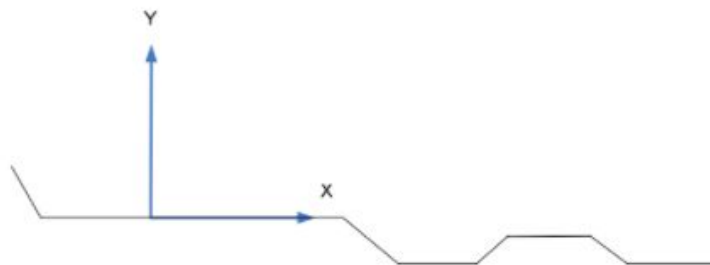
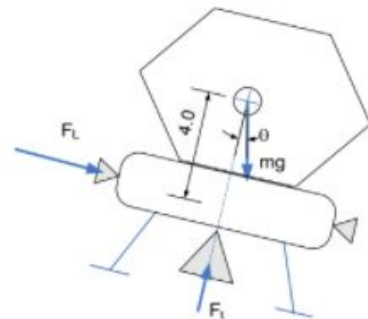
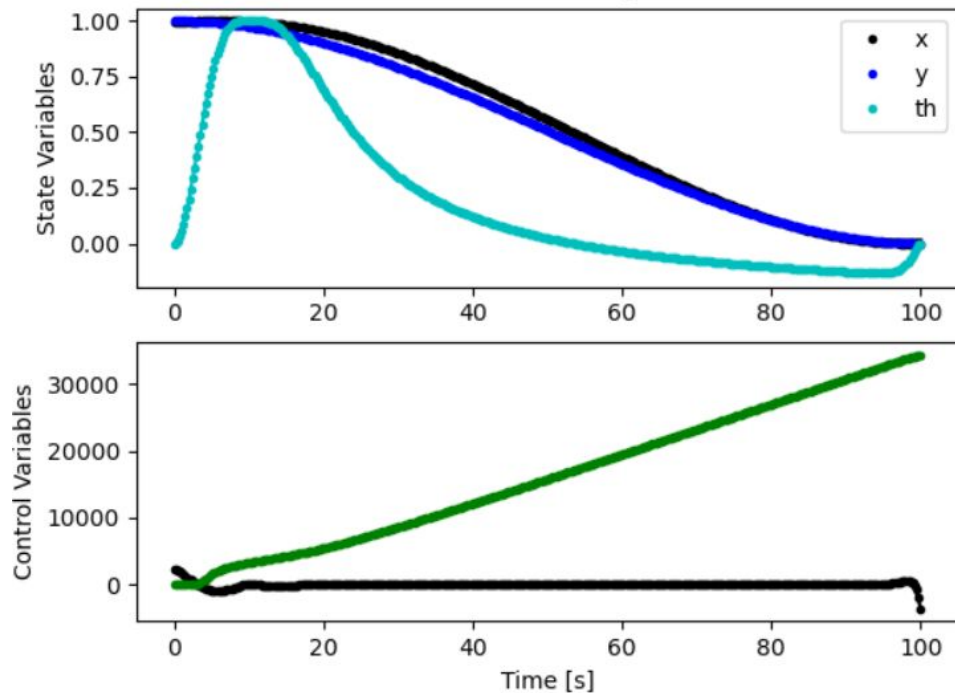
Cartpole Swing-up



Ref. [2]

Lunar Lander

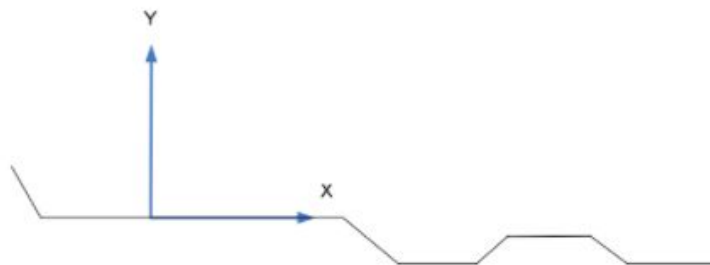
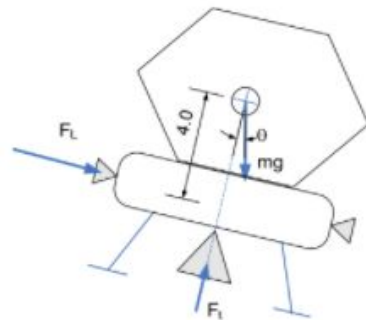
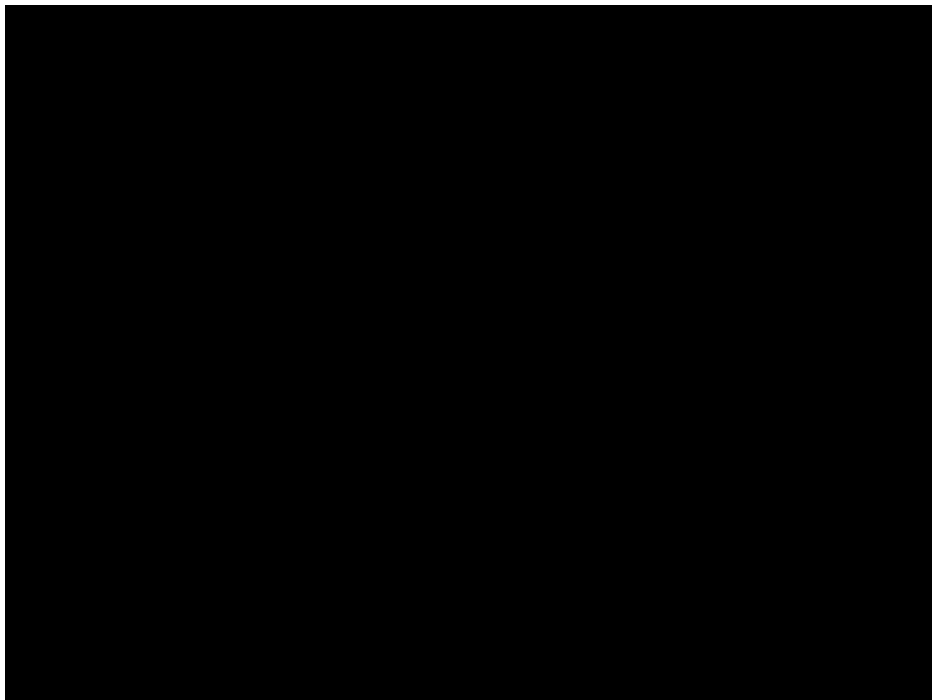
Collocation Points vs. Integration Results



Ref. [3]

Dots = Collocation, Lines = IVP

Lunar Lander

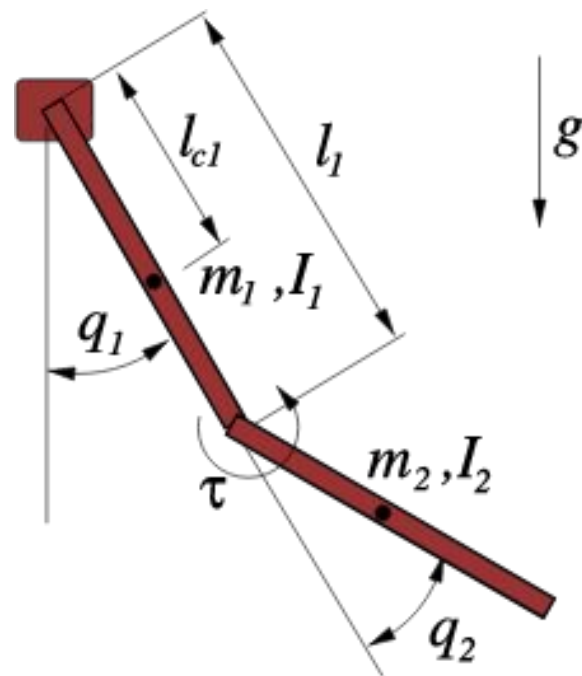
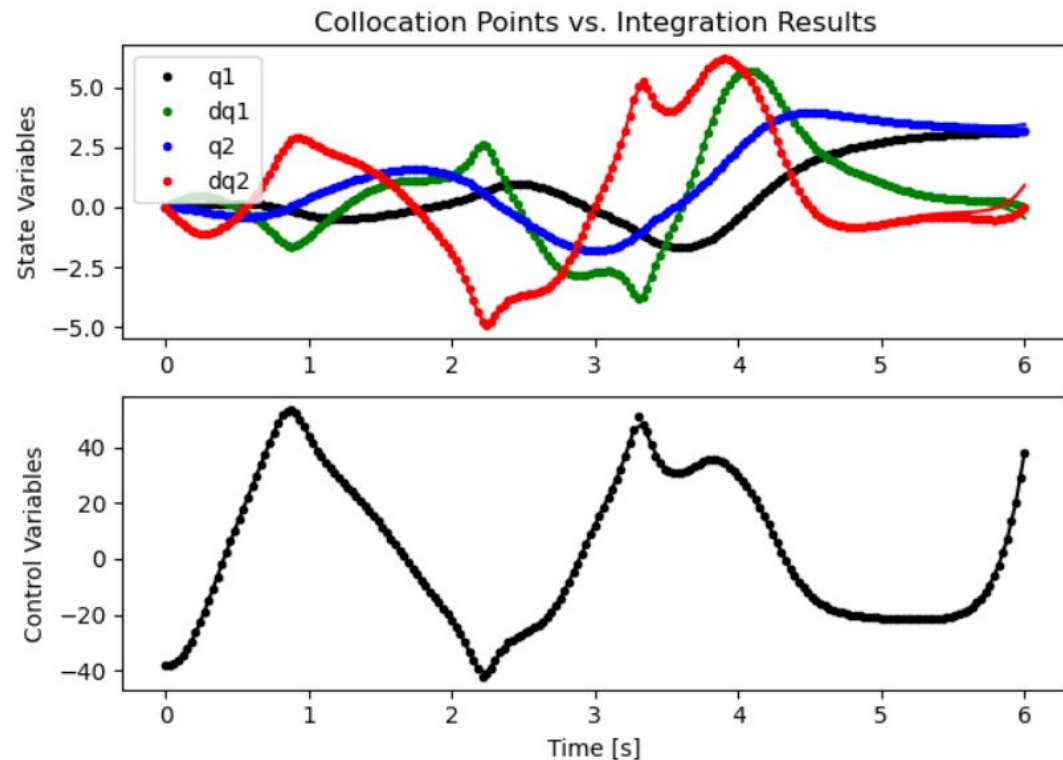


Ref. [3]

Room for Improvement

- Using final time as an optimization variable
- Interfacing with a Sequential Quadratic Programming solver
- Handling arbitrary objectives and integration schemes
- Handling DAE's

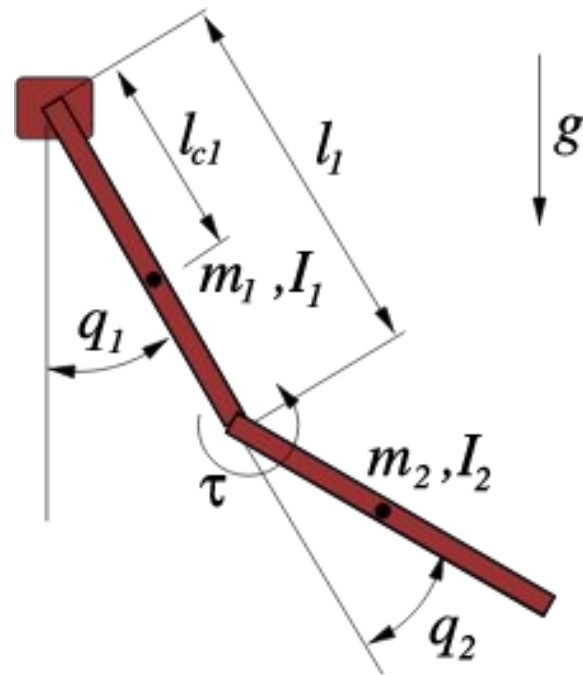
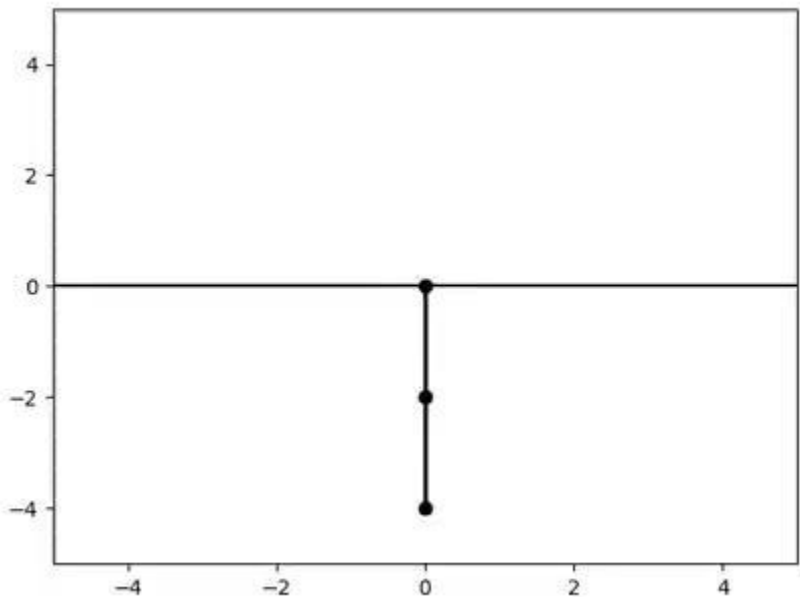
Double Pendulum Swing-up



Ref. [1]

Dots = Collocation, Lines = IVP

Double Pendulum Swing-up



Ref. [1]