Numerical Algorithms

Analysing Cart Pole Swing Up - Team 26

Abhinav Raundhal 2022101089
Mayaank Ashok 2022111022

Faculty - Pawan Kumar

10th May 2024

Problem Statement

The cart-pole system comprises a cart that travels along a horizontal track and a pendulum that hangs freely from the cart. There is a motor that drives the cart forward and backward along the track. It is possible to move the cart in such a way that the pendulum, initially hanging below the cart at rest, is swung up to a point of inverted balance above the cart. The problem is to numerically compute the minimum-force trajectory to perform this so-called “swing-up” maneuver.

Theory

- The goal is to minimize the objective function J by finding the optimal control force $u(\tau)$:

$$J = \int_0^T u^2(\tau)d\tau$$

- What can we solve? Any Non Linear Program (NLP)
- `fmincon` in Matlab can solve any NLP of the form:

$$\begin{aligned} \text{minimize } f(x) \\ \text{subject to } g_i(x) \leq 0,\ i \in 1...m \\ h_j(x) = 0,\ j \in 1...p \\ x \in X \end{aligned}$$

System Dynamics

We can formalize the system dynamics as follows:

$$\ddot{q_1} = \frac{Lm_2 \sin(q_2)\dot{q_2}^2 + u + m_2 g \cos(q_2)\sin(q_2)}{m_1 + m_2(1 - \cos^2(q_2))}$$

$$\ddot{q_2} = -\frac{Lm_2 \cos(q_2)\sin(q_2)\dot{(q_2)}^2 + u\cos(q_2) + (m_1 + m_2)g\sin(q_2)}{Lm_1 + Lm_2(1 - \cos^2(q_2))}$$

The state variables are defined as:

$$x = \begin{bmatrix} q_1 \\ q_2 \\ \dot{q}_1 \\ \dot{q}_2 \end{bmatrix} \qquad \dot{x} = f(x, u) = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix}$$

Now it remains to discretize the continuous function such that we are left with a set of equations in a form that can be solved by the NLP.

Trapezoidal Collocation

The trapezoidal collocation method treats the continuous function as a set of linear splines.

$$J = \sum_{k=0}^{N-1} \frac{h_k}{2}(u_k^2 + u_{k+1}^2)$$

$$\frac{1}{2}h_k(f_{k+1} + f_k) = x_{k+1} - x_k$$

We are trying to find the decision variables:

$$x_0 \dots x_N, \quad u_0 \dots u_N$$

After adding boundary conditions, this is now a NLP problem.

Hermite Simpson

The Hermite-Simpson collocation method treats the continuous function as a set of quadratic splines. We also have collocation points at the midpoints of each segment.

$$J = \sum_{k=0}^{N-1} \frac{h_k}{s}\left(u_k^2 + 4u_{k+\frac{1}{2}}^2 + u_{k+1}^2\right)$$

$$\frac{1}{6}h_k\left(f_{k+1} + 4f_{k+\frac{1}{2}} + f_k\right) = x_{k+1} - x_k$$

We have to interpolate at the midpoints:

$$x_{k+\frac{1}{2}} = \frac{1}{2}\left(x_k + x_{k+1}\right) + \frac{h_k}{8}\left(f_k - f_{k+1}\right)$$

Interpreting the results

The NLP solver returns the decision variables $x_0 ... x_N, \quad u_0 ... u_N$. We have to convert them into continuous functions $x(t)$ and $u(t)$ For trapezoidal collocation we use linear splines for the control function and quadratic splines for the state.

$$u(t) = u_k + \frac{\tau}{h_k}(u_{k+1} - u_k)$$

$$x(t) \approx x_k + f_k \tau + \frac{\tau^2}{2h_k}(f_{k+1} - f_k)$$

For Hermite-Simpson we use quadratic splines for the control and cubic splines for the state

$$u(t) = \frac{2}{h_k^2}(\tau - \frac{h_k}{2})(\tau - h_k)u_k - \frac{2}{h_k^2}\tau(\tau - h_k)u_{k+\frac{1}{2}} + \frac{2}{h_k^2}\tau(\tau - \frac{h_k}{2})u_{k+1}$$

$$x(t) = x_k + f_k(\frac{\tau}{h_k}) + \frac{1}{2}(-3f_k + 4f_{k+\frac{1}{2}} - f_{k+1})(\frac{\tau}{h_k})^2 + \frac{1}{3}(2f_k - 4f_{k+\frac{1}{2}} + 2f_{k+1})(\frac{\tau}{h_k})^3$$

The error at the knot points are zero but there is non-zero error between the knot points.
Therefore the accumulated error is calculated as the integral of the errors between the knot points.

MATLAB Code

- The provided code, the `OptimTraj` module, allows us to specify the dynamics of a system that is then automatically converted into a NLP problem and then optimized.
- Using this we can simulate the optimal trajectory of the cart-pole system for minimizing the total force and plot the results.
- The results we got are shown below:

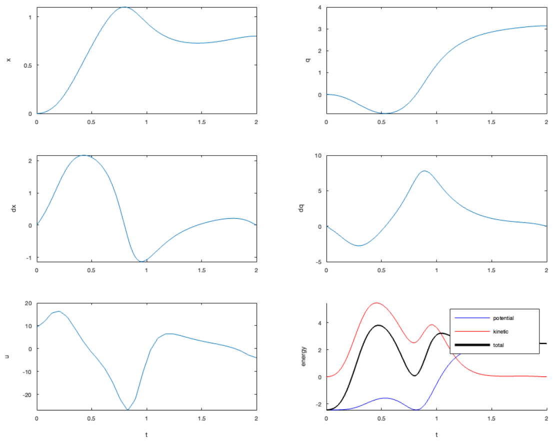

Figure 1: Optimal State

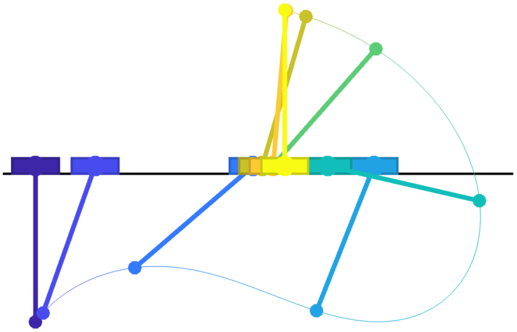

Figure 2: Snapshots of the trajectory

Error Variation

- We varied the number of grid points in the collocation method. And plotted the time taken and the total accumulated error as a function of the number of grid points.
- As we can see in the below graphs, as we increase the number of grid points, there is a tradeoff observed. The error decreases, but the execution time of the code increases.
- At the 25-30 grid-points, the accumulated error starts to taper off, while still maintaining a low execution time. Hence this is an optimal value for the number of grid-points

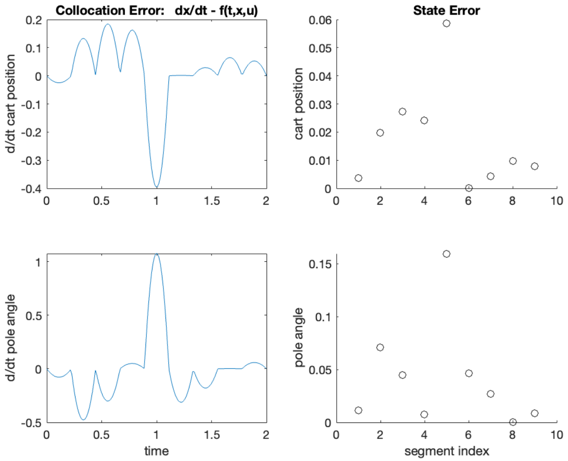

Figure 3: 10 Grid Points

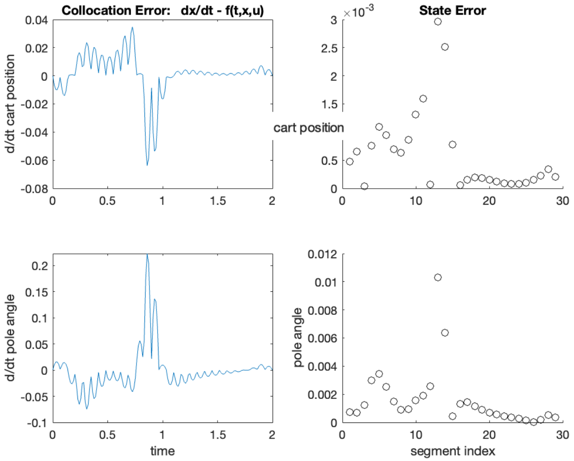

Figure 4: 30 Grid Points

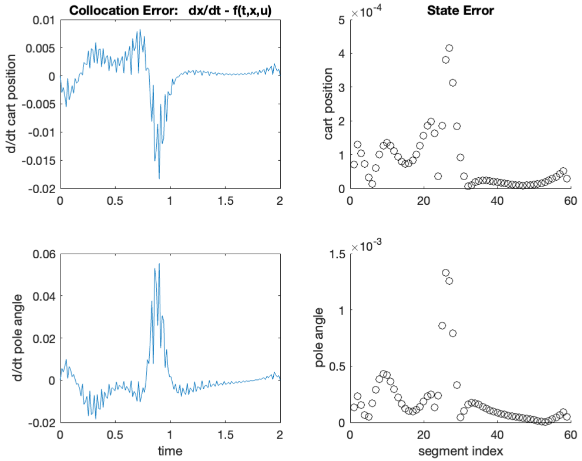

Figure 5: 60 Grid Points

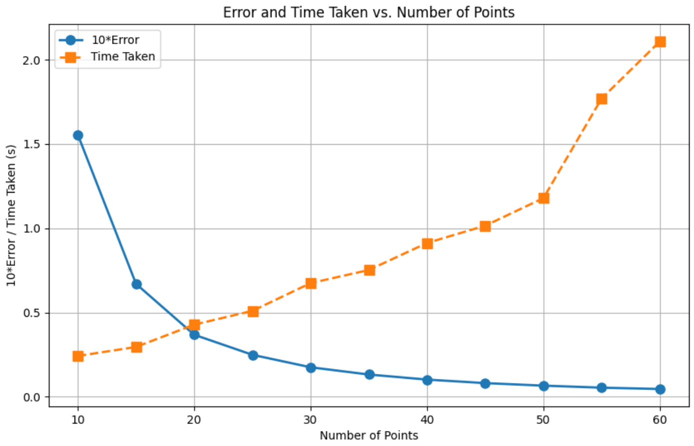

Figure 6: Trend

Troubles with perturbations

- We tried to implement the effects of perturbation on the stability of the system. However we faced some challenges that prevented us from getting accurate error data. The solver returns both the state as a function of time and the control force applied to achieve that state.
- Upon re-running the simulation using only the control force to update the state at each time-step, a tiny error was accumulated each time. This, exacerbated by the inherent instability of the system lead to an unacceptable error even in the baseline case.
- Further the solver doesn’t support a variable time-step between points in the code to discretize the grid-points. Hence creating an adaptive mesh-size isn’t possible.