NTNU
Institutt for teknisk kybernetikk

Masters Project

Cool name

A novel approach to reinforcement learning

Thomas A. Frekhaug [1]

supervised by

Prof. Sebastian Gross[2]

October 11, 2019

[1]M.SC student, Cybernetics and Robotics 4^{th} year @Norwegian University of Science and Technology
[2]Director of R&D @Andøya Space Centre

Contents

List of Figures

List of Tables

1 Project description

2 Theory

2.1 MPC

2.2 Reinforcement learning

2.3 Sensitivity analysis

2.3.1 Integrators, IVP

Integrators, in this setting, are numerical methods for solving ODE and IVP's. They come in many different forms, where the most known and simple integrator is the Eulers method. This is a explicit forward method, and is defined as the following

$$y'(t) \approx \frac{y(t+h) - y(t)}{h} \tag{1}$$

$$y(t+h) \approx y(t) + hf(t, y(t)) \tag{2}$$

Explicit methods are purely based on of already known values. This is in opposition to the implicit methods, that implores that an equation must be solved in order for the system to be solved. implicit solvers are used when the eigenvalues of the systems becomes so small that a small enough timestep cannot be chosen, or when there is a large spread in the eigenvalues of the jacobian. Such systems are called stiff.
A forward solver solves from time zero to final time, while a backwards solver soles from final time to inital time.

3 Implementation

The first complete setup will be a simple 2 state SIMO system with LQR cost. We define the initial NLP problem as following

$$\begin{aligned} \min_{u,x,\sigma} \quad & x_N^T P_N x_N + \sum_{k=0}^{N-1} x_k^T Q x_k + u_k^2 R + w^T \sigma_k \\ \text{s.t} \quad & x_{k+1} = x_k + \frac{T}{N}(Ax_k + Bu_k),\ k \in [0, N-1], \\ & u_k \geq -4, \\ & u_k \leq 4, \\ & h(x_k, u_k) \leq \sigma_k. \end{aligned} \tag{3}$$

Which can be rewritten as

$$\begin{aligned} \min_{u,x,\sigma} \quad & x_N^T P_N x_N + \sum_{k=0}^{N-1} l_\theta(x_k, u_k) + w^T \sigma_k \\ \text{s.t} \quad & x_{k+1} = f_\theta(x_k, u_k),\ k \in [0, N-1], \\ & g(u_k) \leq 0, \\ & h(x_k, u_k) \leq \sigma_k. \end{aligned} \tag{4}$$

Note, The P_N matrix is by the ricatti equation, where K is the infinite horizon gain, and for simplicity is simply calculated by the LQR-command of matlab

$$P = (A - BK)^T P (A - BK) + K^T R K + Q \tag{5}$$

l_θ	LQR stage cost, parametrized by theta
σ	Slack variables, to prevent states exceeding max limits
$g(u_k)$	Pure constraints on our input u
$h(x_k, u_k)$	Mixed constraints, typically only max/min of x with slack variables
w	Slack variables weighting matrix

Now, to expand this for the RL case The value function of the RL becomes the objective function of the NLP, however, a few alterations must be done to the objective function in order to properly retrieve the valuefunction. Namely, the RL discounted stage cost must be added to the objective function, since the value function is defined as

$$\hat{V_\infty} = \min_\pi E(\sum_{k=0}^{N-1} \gamma^k \hat{(}L(s, \pi))) \tag{6}$$

Where the discounted stage function does not nesecarily need to be the same as the objective function stage cost. Thereby giving the modified scheme:

$$\begin{aligned} V = \min_{u,x,\sigma} \quad & x_N^T P_N x_N + \sum_{k=0}^{N-1} (l_\theta(x_k, u_k) + w^T \sigma_k) + \sum_{k=0}^{N-1} (\frac{1}{2}\gamma^k (x_k^T Q_{rl} x_k + u_k^2 R_{rl} + w^T \sigma_k) \\ \text{s.t} \quad & x_{k+1} = f_\theta(x_k, u_k),\ k \in [0, N-1], \\ & g(u_k) \leq 0, \\ & h(x_k, u_k) \leq \sigma_k. \end{aligned} \tag{7}$$

This can be rewritten on a different form with a few alterations, thereby not making the two equal, but similar in the sense of values and goals.

$$\begin{aligned} \min_{u,x,\sigma} \quad & V_0 + \frac{\gamma^N}{2} x_N^T P_N x_N + \sum_{k=0}^{N-1} (f^t[x_k, u_k]^T) + \sum_{k=0}^{N-1} (\frac{1}{2}\gamma^k (x_k^T Q_{rl} x_k + u_k^2 R_{rl} + w^T \sigma_k) \\ \text{s.t} \quad & x_{k+1} = f_\theta(x_k, u_k),\ k \in [0, N-1], \\ & g(u_k) \leq 0, \\ & h(x_k, u_k) \leq \sigma_k. \end{aligned} \tag{8}$$

Note that now, all the RL-parameters to be learned are contained within the f-function, which is NOT the same function as the stage transition function. V_0 has also been added to add a base level for the value function. For simplicity, f_θ is chosen constant and equal to an imperfect model.

The RL-algorithm should now learn two parameters, namely f, V_0. In addition, no constraints are put on x, thereby removing the need for slack variables and the mixed inequality constraint. The goal should be to converge parameters to zero of a stochastic model on the form

$$x = Ax + Bu + D\epsilon, \epsilon\ N(1,1) \tag{9}$$

Which leads to the resulting NLP problem

$$\begin{aligned} \min_{u,x} \quad & V_0 + \frac{\gamma^N}{2} x_N^T P_N x_N + \sum_{k=0}^{N-1} (f^t[x_k, u_k]^T) + \sum_{k=0}^{N-1} (\frac{1}{2}\gamma^k (x_k^T Q_{rl} x_k + u_k^2 R_{rl}) \\ \text{s.t} \quad & x_{k+1} = x_k + \frac{T}{N}(Ax_k + Bu_k),\ k \in [0, N-1], \\ & g(u_k) \leq 0, \\ & x_0 = x_0; V_0 = V_0. \end{aligned} \tag{10}$$

Some notes of importance

- Normally, the stage cost is parametrized by $x_{k+1}^T Q x_{k+1}^T$, summing from $[0, N-1]$. However, here the final value is parametrized by the infinite horizon, and as such is removed from the stage sum. Therefore, the quadratic term in the sum becomes $x_k^T Q x_k^T$,
- the f to be estimated is a vector, and as such, we only try to modify the stage cost in a linear sense.
- The quadratic cost is replaced by the discounted cost.

3.1 Implementation of RL and MPC

First, we do a simple linear test on a MSD system. The stage cost function l_ϕ becomes

$$l_\theta = V_0 + \frac{\gamma^N}{2} x_N^T P_N x_N + \sum_{k=0}^{N-1} (f^t[x_k, u_k]^T) + \sum_{k=0}^{N-1} (\frac{1}{2}\gamma^k (x_k^T Q_{rl} x_k + u_k^2 R_{rl}) \tag{11}$$

The real system dynamics of MSD is quite simple, and is given as

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \epsilon \qquad \epsilon \approx N(1,1) \tag{12}$$

3.2 Test2

The current problem is formulated as

$$\min_{u,x} \quad \frac{\gamma^N}{2} x_N^T P_N x_N + \sum_{k=0}^{N-1} (f^t[x_k, u_k]^T) + \sum_{k=0}^{N-1} (\frac{1}{2}\gamma^k (x_k^T Q_{rl} x_k + u_k^2 R_{rl}) \tag{13}$$

$$s.t \tag{14}$$

$$x_{k+1} = x_k + \frac{T}{N}(Ax_k + Bu_k + E),\ k \in [0, N-1] \tag{15}$$

$$x_0 = s \tag{16}$$

The real system dynamics is stated as

$$\begin{bmatrix} \dot{x_1} \\ \dot{x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \epsilon \qquad \epsilon \approx N(1,1) \tag{17}$$

The RL-algorithms acts upon 10 parameters, $\theta = [q_1, q_2, q_3, q_4, a_1, a_2, a_3, a_4, e1, e2]$, that is the elements of the MPCmodel A and E, and the quadratic stage cost matrix Q. The RL-algorithm is as follows

$$L = x_s^T x_s + u_s^T u_s \tag{18}$$

$$TD = L + \gamma V_{s+1} - V_s \tag{19}$$

$$\theta_{s+1} = \theta_s + \alpha TD \nabla_\theta \mathcal{L} \tag{20}$$

The sequence of operations is

$$x_s, u_s, \theta_s, V_s, \nabla_{\theta,s}\mathcal{L} \underset{simulation}{\rightarrow} x_{s+1} \underset{optimization}{\rightarrow} u_{s+1}, V_{s+1}, \nabla_{\theta,s+1}\mathcal{L} \tag{21}$$

$$\downarrow \tag{22}$$

$$x_{s+1}, u_{s+1}, \theta_{s+1}, V_{s+1}, \nabla_{\theta,s+1}\mathcal{L} \underset{progression}{\leftarrow} (L(x_s, u_s) + \gamma V_{s+1} - V_s)\alpha + \theta_s = \theta_{s+1} \underset{RL-update}{\leftarrow} \tag{23}$$

$V(s)$ is defined as the solution to the problem of 13 - 16, and $Q(s,a)$ is defined as the solution to the same problem 13 - 16, only with the added constraint $u0 = a$. Since the input a is deterministic, then the solution to both of these problems becomes the same, and for simplicity, $V(s)$ is used instead of $Q(s,a)$

MPCParameters:

The MPC model is simulated by discrete eulersmethod, where the dt is given by the prediction horizon N and time horizon T. The vector f is set to zero, and the discount is set to $\gamma = 0.9$. A and B is initialized as the real system dynamics, and E is initialized as zero. In addition, $T = 1$ & $N = 5$.

Other Parameters:

The system as a whole is simulated by 4th order runge-kutte with timestep of 0.01. The RL-algorithm has parameters $\gamma = 0.9, \alpha = 0.0001$

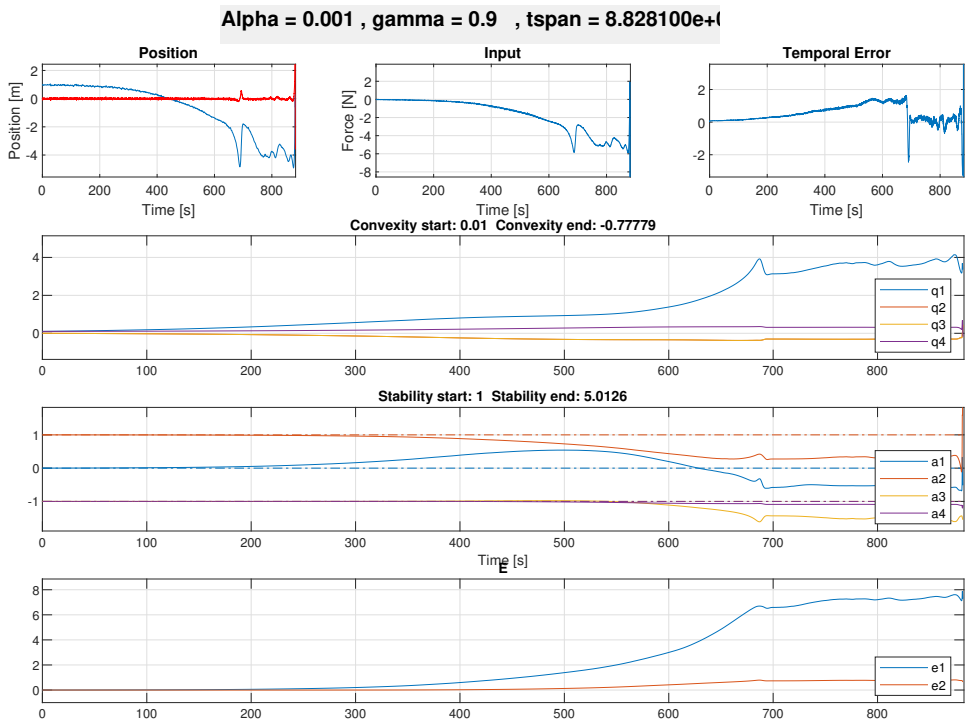

Figure 1: Caption

4 Discussion

5 Results

6 Known Problems

References