

Adaptive Control and Reinforcement Learning HW

March 7, 2009

1 Problem 2: Differential Dynamic Programming

This problem is designed to help you understand DDP. Each question to be addressed is in the file *ACRL-HW2b.m*. The question requires you to setup the trajectory for DDP, tweak parameters of the setup for DDP, evaluate the results, and comment on the particular implementation.



Figure 1: Stanford's autonomous performing the nose-in funnel maneuver you will build here. From [1]

2 Problem 3: Inverse Optimal Control

A. Theoretical

Constraint

Suppose you are given a controller that you can execute on the (linear) helicopter simulation, possibly from multiple starting trajectories. Assuming the controller is optimal, what constraints (ala Maximum Margin Planning) should hold? Write this in terms of Q and R as we did in class.

Subgradient update rule

Take the sub-gradient of this loss function. Write down the update rule using this subgradient and a step-size.

Non-negativity Projection

Note that the update rule you derive should naturally ensure Q and R are symmetric matrices. Note that it *may not* ensure the matrices remain positive definite, or even semidefinite. You can ensure R is definite by redefining it as $I + \tilde{R}$, where $\tilde{R}$ is only semi-definite. However, this still leaves open the issue as to how to ensure Q and $\tilde{R}$ are positive semidefinite.

A standard trick that provably ensures convergence and regret bounds for gradient algorithms is to “project” a variable that doesn’t meet a constraint onto the constraint set. The constraint set here is the positive definite cone of matrices. [2] Define projection as minimizing the Frobenius norm between the original matrix and the “projected” matrix that lies inside the positive definite cone. Give Q that is a symmetric but indefinite matrix, how can you compute this projection? Prove this is the correction projection to minimize the Frobenius norm.

B. Practical

You will find a controller for hover around the nominal controls as $K_{ss.mat}$. Use the approach you derived above to (a) derive a cost function for which the controller is optimal and (b) derive a controller for that cost function again using LQR.

How does the cost function differ from the one you used in HW2a to hover? (Remember that the relative scales of costs are all that really matter here.)

Describe any problems you had.

Comment on whether the non-negativity fixing is necessary in practice for this problem.

References

- [1] Pieter Abbeel, Adam Coates, Morgan Quigley, and Andrew Y. Ng. An application of reinforcement learning to aerobatic helicopter flight. In *In Advances in Neural Information Processing Systems 19*, page 2007. MIT Press, 2007.

- [2] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004.