Acrobot Region of Attraction

Acrobot Region of Attraction (Part I)

0.0/5.0 points (graded)

Prerequisites for this problem: You will need a semidefinite program solver. For this, you can either use SeDuMi (found here) or MOSEK (found here). MOSEK is typically a lot faster than SeDuMi and we recommend installing it. If you are using a precompiled version of Drake, you can install the solver of your choice from the above links. If you built drake from source, you can just do

```
cd /path/to/drake-distro
make options
```

and then set WITH_MOSEK or WITH_SEDUMI to ON. Then hit "c to configure", "g to generate and exit", and finally, from the drake-distro folder, do:

```
make
```

In this problem, we will use Drake to approximate the region of attraction (ROA) for a LQR controller that makes the Acrobot balance at the upright configuration. We have provided most of the code you'll need to compute ROAs using Drake in **acrobotROA.m** (found here). Please download this file and **place it in the examples/Acrobot folder** in Drake.

The code we have provided first sets up the Acrobot model and computes a LQR controller to balance the robot at the upright position. It also calls the Drake function **regionOfAttraction.m** to compute approximations of the ROA of the resulting closed-loop system. Please try to familiarize yourself with the different functions in regionOfAttraction.m and make sure you understand (at least at a high level) what they do. The regionOfAttraction.m code uses Sums-of-Squares (SOS) Programming to approximate ROAs. Your only task is to fill in the lines in the block of code that says "FILL THIS IN" (see below for instructions on this) in acrobotROA.m.

(a) Set the degree of the Lyapunov function to 2 and the degree of the Lagrange multipliers to 2 also. Next, use the "sampling" method to approximate the ROA. This option fixes the Lyapunov function (the cost-to-go function from LQR in our case) and searches for the largest sub-level set of the function such that $\dot{V} < 0$ for all points in the sub-level set. This is done by random sampling rather than SOS programming. Type in the value of the level set found below (i.e., the last value of "rho" printed out).

Answer: 6.81

Explanation
We set options.degL1 = 2 and options.degV = 2. Setting options.method = 'sampling' and running the code gives us a rho of 6.81.

(b) Now use the binary search method to approximate the ROA. This option again fixes the Lyapunov function and searches for the largest sub-level set of the function such that $\dot{V} < 0$ for all points in the sub-level set. However, this search is now done using SOS programming. Please enter the value of the level set found below (i.e., the last value of "rho" printed out).

Answer: 6.124

Explanation
Setting options.method = 'binary' and running the code gives us a rho of 6.124.

Submit You have used 0 of 3 attempts

Answers are displayed within the problem

Acrobot Region of Attraction (Part II)

0.0/10.0 points (graded)
(c) Finally, we will use SOS programming to search for both the Lyapunov function and the sub-level set. This is done using the "bilinear" option. The final Lyapunov function is printed to the MATLAB workspace (the variable "V"). The 1 sub-level set (i.e., the set of points x such that $V(x) < 1$) is the estimated ROA.

Find a point x_0 such that $V(x_0) < 1.0$ (i.e., a point inside the verified ROA). Ideally, choose a point close to the boundary of the verified ROA. The easiest way to do this is probably to use the plot of the ROA generated by the code. This is a plot of a two-dimensional slice of the

four-dimensional ROA, with the angular velocity coordinates set to 0. In other words, if we have a point (θ_1, θ_2) inside the gray region of the plot, the state $[\theta_1, \theta_2, 0, 0]$ lies inside the verified ROA.

Simulate the system from the point you found (we have provided you code for this in acrobotROA.m). Check that the Lyapunov function starts off less that 1 and then decreases to almost 0 (the plot of the value of the Lyapunov function vs. time will appear in Figure 2). Type in the Lyapunov function **V_grade** that gets printed to the MATLAB window and the 100 x 4 matrix x_grade below. Note: You do not have to modify V_grade in any way (e.g., taking out parentheses, etc. or changing the names of variables). Simply copy it below.

```
1 syms q1 q2 v1 v2
2 V_grade = ;
3 x_grade = [   ];
4
```

Unanswered

```
% For this problem, we set options.method = 'bilinear'.
% A point that lies in the ROA is x0 = [2.938;0.45;0;0].
```

Run Code

Submit You have used 0 of 3 attempts

Answers are displayed within the problem