Defense Assignment

Mart Veldkamp

I. Introduction

This assignment aims to implement a defense mechanism according to the strategy proposed in scientific publication "Constraining Attacker Capabilities Through Actuator Saturation".

II. Reproduce the results in Figure 1

In this assignment, I've used the state and input matrix from section IV of the paper. Which were:

$$F = \begin{bmatrix} 0.85 & 0.23 \\ -0.47 & 0.12 \end{bmatrix} \tag{1}$$

$$G = \begin{bmatrix} 0.07 & 0.3 \\ 0.23 & 0.1 \end{bmatrix} \tag{2}$$

A. *Evaluate Problem (8)*

The formal definition of a described in the paper is $a \in (0, 1)$, so any value between 0 and 1. So how does one choose a value, I've found the following observations:

- For lower values of a, the ellipsoid tends the elongate or stretch in one direction, these shapes can even reach the boundaries of the red constraint (or dangerous region) more readily because one axis of the ellipsoid extends further. However, such a shape may be overly conservative in one direction while underestimating the volume in other directions.
- With values like $a = 0.9$, the ellipsoid becomes more uniformly shaped around the origin. This ensures that the ellipsoid encloses the reachable set more evenly and is better suited to handle multiple constraints (such as both the red and red+green constraints). It avoids excessive elongation in any single direction and generally provides a tighter, more balanced bound on the reachable set.

My conclusion was that if the goal is to avoid dangerous regions that lie in a specific direction (as captured by the red constraint), a lower aa might make the ellipsoid "reach out" to those limits.

While if your goal is to uniformly protect against constraints, a higher a is better.

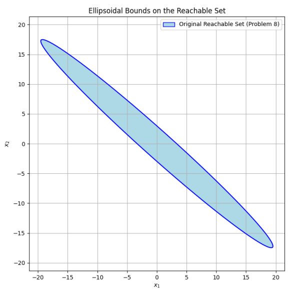

Fig. 1. Reachable set, $a = 0.3$.

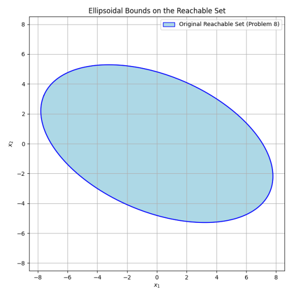

Fig. 2. Reachable set, $a = 0.9$.

Another thing I observed from 3 is that the ellipsoid seems to be rotated, so to get the best possible result, I've added a rotation of $-\frac{\pi}{8}$, given the result of 2.

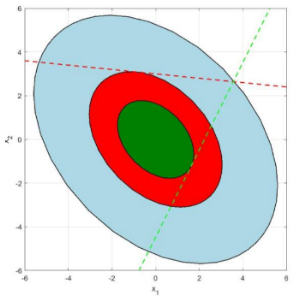

Fig. 3. figure from the assignment.

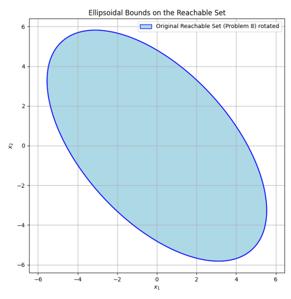

Fig. 4. $a = 0.85$ and rotated.

As for P, I've got the following values:

$$P = \begin{bmatrix} 0.05785381 & 0.03909993 \\ 0.03909993 & 0.12351144 \end{bmatrix} \tag{3}$$

With the eigenvalues of P being: $[0.039628380.14173687]$.

Mart Veldkamp is a master's students at TU Delft, The Netherlands. E-mail:`mmveldkamp@student.tudelft.nl`.

B. Evaluate Problem (13), red constraints

For the following problem, we were given $a = 0.65$ together with the constraint:

$$D_1 = \{x \mid 0.1x_1 + x_2 \geq 3\} \tag{4}$$

Implementing this, gives the following result:

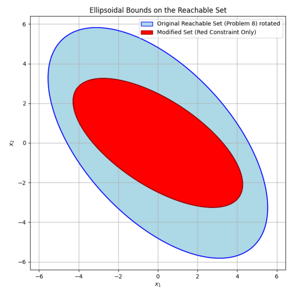

Fig. 5. constrained, no rotation.

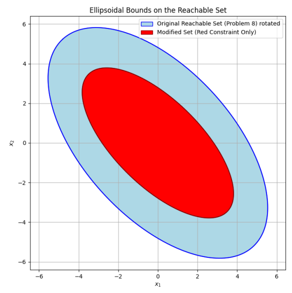

Fig. 6. constrained, with rotation.

As can be seen, also here adding a rotation makes it look more like the figure from the assignment.

Next to rotating, changing the state and input matrix (the main way I would say), increasing a and scaling will give a more similar plot.

The results were:

$$Y^{-1} = \begin{bmatrix} 0.18651704 & 0.15787854 \\ 0.15787854 & 0.32207655 \end{bmatrix} \tag{5}$$

And $\hat{R} = 0.32263149763899 19$

C. Evaluate Problem (13), red + green constraints

In this experiment, the dangerous set is defined as the union of two half-space constraints:

$$D_2 = \{x \mid 0.1x_1 + x_2 \geq 3\} \cup \{x \mid -2x_1 + x_2 \leq -2\sqrt{5}\} \tag{6}$$

The first constraint, $0.1x_1 + x_2 \geq 3$, defines a half-space where the linear combination of x_1 and x_2 exceeds 3, representing a dangerous boundary in one region of the state space. The second constraint, $-2x_1 + x_2 \leq -2\sqrt{5}$, defines a different half-space that restricts the state space in another direction. Together, these constraints ensure that the reachable set is limited in both directions, preventing the system from entering either of these dangerous regions.

Which will result in a smaller reachable set, as can be seen in 7.

Also here we have the same as before, that the resulting reachable set is a lot thinner and longer then the original figure we were suppose to make. With results:

$$Y^{-1} = \begin{bmatrix} 0.60319878 & 0.20121935 \\ 0.20121935 & 0.89221785 \end{bmatrix} \tag{7}$$

And $\hat{R} = 2.906942218987064$

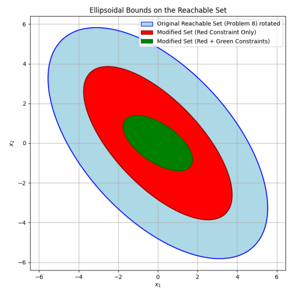

Fig. 7. double constrained, no rotation.

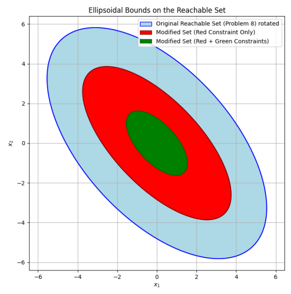

Fig. 8. double constrained, with rotation.

And also here rotating, changing the matrices, changing a and scaling the plot will make it more in line with 3.

To show this, I create custom input and state matrices, and plotted those. As can be seen from 10, this worked quite well to get a good result:

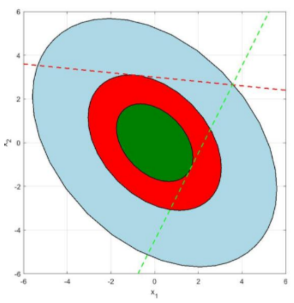

Fig. 9. figure from the assignment.

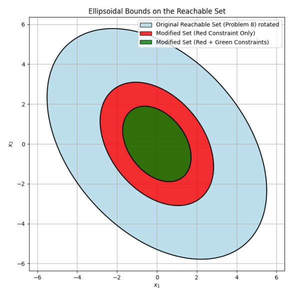

Fig. 10. best attempt with custom matrices.

While it still different from the original image, with enough tweaking you could get it to the same point.

III. Reproduce the results in Figure 2

A. Evaluate Problem (8)

To find an optimal a, we can also simulate all the values between 0 and 1 and run the optimizer for every value. And therefore find an optimal value for a and P.

My results from simulation showed $a \approx 0.8558$ to be an optimal value of a. With matrix P being:

$$P \approx \begin{bmatrix} 0.015 & -0.008 & -0.003 & 0.007 & -0.001 \\ -0.008 & 0.016 & 0.001 & -0.007 & 0.004 \\ -0.003 & 0.001 & 0.016 & -0.01 & 0.0002 \\ 0.007 & -0.006 & -0.009 & 0.036 & -0.01 \\ -0.001 & 0.003 & 0.0001 & -0.011 & 0.018 \end{bmatrix} \tag{8}$$

with eigenvalues: 0.63109, 0.1998, 0.17894, 0.04962, 0.030114.

I had to approximate P for this assignment, otherwise it would fit on the page, for a more accurate matrix, please run the simulation in python.

B. With red constraints

Now lets suppose we take $a = 0.85$, we simulate with the following constraints:

$$D_i = \{\bar{d}_i \mid \bar{b}_i \leq -1\}, i = 1, 2$$

The results can be seen in 12.

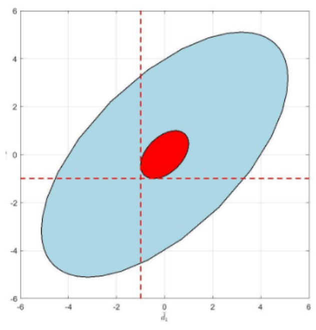
Fig. 11. example figure, what is expected.

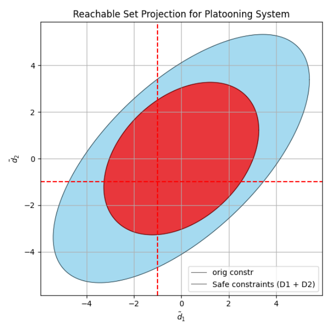

Fig. 12. reachable set with and without constraints.

This results deviates from our original system, as the reachable set with constraints seems to be a lot larger.

The values correcsponding to 12 are:

$$Y^{-1} = \begin{bmatrix} 3.56720905 & 1.3334777 \\ 1.3334777 & 3.56636293 \end{bmatrix} \tag{9}$$

$$\hat{R} = [\ 24.76506489,\ 17.61920056,\ 24.73053313\ ] \tag{10}$$

C. Gains of the controller

Now we are considering initial conditions: $[4, 4, 16, 16, 16]^T$. From LQR1, obtained from the solution of Problem (8). We got the following gain:

$$K \approx \begin{bmatrix} -0.333 & -0.144 & 0.483 & -0.101 & -0.224 \\ 0.188 & -0.188 & -0.101 & 0.360 & -0.101 \\ 0.144 & 0.333 & -0.224 & -0.101 & 0.483 \end{bmatrix}$$

with eigenvalues: $\lambda \approx [0.384 + 0.250j,\ 0.384 - 0.250j,\ 0.623 + 0.278j,\ 0.623 - 0.278j,\ 0.821]$

For LQR2, which is obtained from the solution of Problem (13). We got the following gain:

$$K \approx \begin{bmatrix} -0.070 & -0.061 & 0.190 & -0.029 & -0.160 \\ 0.025 & -0.0249 & -0.042 & 0.084 & -0.042 \\ 0.061 & 0.079 & -0.160 & -0.030 & 0.190 \end{bmatrix}$$

with eigenvalues: $\lambda \approx [0.471 + 0.194j,\ 0.471 - 0.194j,\ 0.712 + 0.228j,\ 0.712 - 0.228j,\ 0.899$

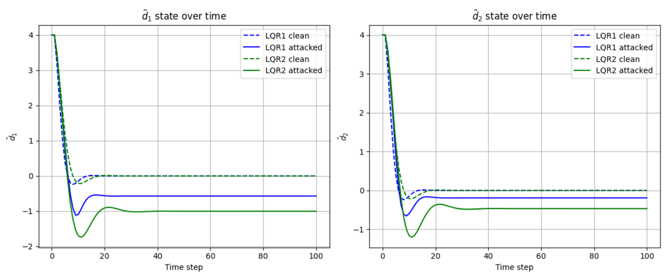

Fig. 13. state over time with and without attack.

D. Introducing an attacker

Now we can implement an attacker on w_1 and start injecting the following constant value $w_1^A = 0.6881$, in 13 we can see the state over time. Which clearly shows that with an attacker, there is a offset in both LQR1 and LQR2 in both $\hat{d}_1$ and $\hat{d}_2$.

These plots show how the controller reacts to the attack. LQR1 reacts aggressively to the added value w_1^A , especially in u_1. The input overshoots which can drive the system out of bounds if actuator limits are present. LQR2 is designed with these limits. The control signals are more careful. As a result, the input stays in the safe limits.

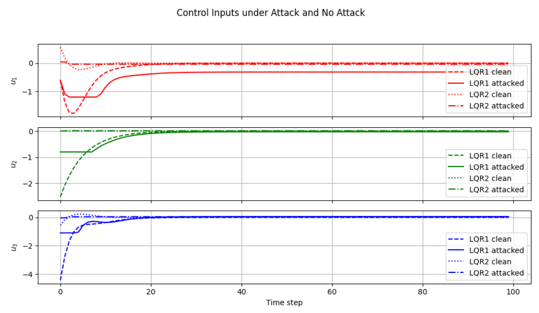

Fig. 14. control inputs with and without attack.