Optimization and Algorithms
Part 1 of the Project

João Xavier and Cláudia Soares
Instituto Superior Técnico
October 2020

Contents

1 Transferring a robot

In this section, we explore the following question: how should we act on a robot—using simple control signals—so that the robot passes as near as possible to given intermediate *points* along its journey?

Section 1.1 states the problem, and then each section 1.2, 1.3, and 1.4 explores a different way to formulate it as an optimization problem.

1.1 The setup

We want to act on a robot such that the robot behaves in a desired way over a finite-time horizon $\{0, 1, 2, \ldots, T\}$.

Where is our robot? The state of the robot. The position of the robot at time t in $\{0, 1, \ldots, T\}$ is denoted by $p(t)$; its velocity, by $v(t)$. Because we consider a robot that moves in a plane, the position and velocity vectors are two-dimensional, that is, $p(t) \in \mathbf{R}^2$ and $v(t) \in \mathbf{R}^2$ for $0 \leq t \leq T$. The *state* $x(t)$ of the robot at time t is

$$x(t) = \begin{bmatrix} p(t) \\ v(t) \end{bmatrix}.$$

Note that $x(t)$ is a four-dimensional vector: $x(t) \in \mathbf{R}^4$, for $0 \leq t \leq T$.

How do we act on the robot? The control signal. We act on the robot by applying a force at each time t in $\{0, 1, \ldots, T-1\}$. (These forces are applied in fact by the motor in the robot.) The force that we apply at time t, denoted by $u(t)$, is two-dimensional, that is, $u(t) \in \mathbf{R}^2$ for $0 \leq t \leq T-1$. The sequence of forces applied throughout time, $\{u(t) \colon t = 0, \ldots, T-1\}$, is called the *control signal*. This is the signal we want to design.

How does the robot move? The dynamics of the robot. Each time we apply a force to the robot, we push it a bit, thereby changing its state (position and velocity). More precisely, if $x(t)$ is the current state of the robot and we apply the force $u(t)$ to the robot, then its state changes to

$$x(t+1) = Ax(t) + Bu(t), \tag{1}$$

where $A \in \mathbf{R}^{4\times 4}$ and $B \in \mathbf{R}^{4\times 2}$ are known matrices. These matrices depend on physical constants such as the mass of the robot and the drag coefficient of the environment (if you're curious about how to obtain A and B, see slides 17—24 in Part 1, "The art of formulating optimization problems"). For this project, let

$$A = \begin{bmatrix} 1 & 0 & 0.1 & 0 \\ 0 & 1 & 0 & 0.1 \\ 0 & 0 & 0.9 & 0 \\ 0 & 0 & 0 & 0.9 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0.1 & 0 \\ 0 & 0.1 \end{bmatrix}.$$

What do we want the robot to do? We want to act on the robot such that many wishes are fulfilled:

- **Wish 1: transfer.** We wish the robot, which is resting at $t = 0$ in a given initial position $p_{\text{initial}} \in \mathbf{R}^2$, to transfer to a desired position at time $t = T$, denoted $p_{\text{final}} \in \mathbf{R}^2$, and to rest there. That is, the initial state of the robot is $x(0) = x_{\text{initial}}$, where

  $$x_{\text{initial}} = \begin{bmatrix} p_{\text{initial}} \\ 0 \end{bmatrix} \in \mathbf{R}^4,$$

  and we wish the final state to be $x(T) = x_{\text{final}}$, where

  $$x_{\text{final}} = \begin{bmatrix} p_{\text{final}} \\ 0 \end{bmatrix} \in \mathbf{R}^4;$$

- **Wish 2: bounded control.** Because the motor of the robot is limited, we can apply forces only up to a certain magnitude, say $U_{\max}$. That is, we wish the control signal to satisfy $\|u(t)\|_2 \leq U_{\max}$ for $0 \leq t \leq T-1$, where
$$\|x\|_2 = \left(x_1^2 + \cdots + x_n^2\right)^{1/2}$$
is the usual Euclidean norm of a vector x in $\mathbf{R}^n$ with $x = (x_1, \ldots, x_n)$. This norm is also commonly called the ℓ_2 norm;

- **Wish 3: waypoints.** As the robot transfers from its initial position to its final position, we also would like it to pass as near as possible to given intermediate locations—called *waypoints*—at given appointed times. For example, we could wish the robot to get data from information sources at those locations, via wireless; because wireless signals weaken with distance, the closer the robot is to each of the locations at the appointed times, the better.

  The set of waypoints is $\{w_1, \ldots, w_K\}$, and the associated set of appointed times is $\{\tau_1, \ldots, \tau_K\}$. Hence our wish is to have the position of the robot at each time τ_k to be close to w_k, that is, we wish $p(\tau_k) \simeq w_k$ for $1 \leq k \leq K$;

- **Wish 4: simple control.** Finally, we wish to have a "simple" control signal, that is, a control signal that is constant, or, at least, a control signal that changes just a few times during the time horizon $\{0, 1, \ldots, T-1\}$. This means that we wish to have $u(t) = u(t-1)$ for most values of t in $\{1, \ldots, T-1\}$.

1.2 The ℓ_2^2 regularizer

We now convert the wish list at the end of section 1.1 into an optimization problem. We encode the first two wishes (transfer and bounded control) as constraints of the optimization problem; to deal with the last two wishes (waypoints and simple control), we create a cost function that penalizes deviations from those wishes.

Optimization problem. We formulate the problem as follows:

$$\begin{array}{ll} \underset{x,u}{\text{minimize}} & \sum_{k=1}^{K} \|Ex(\tau_k) - w_k\|_2^2 + \lambda \sum_{t=1}^{T-1} \|u(t) - u(t-1)\|_2^2 \\ \text{subject to} & x(0) = x_{\text{initial}} \\ & x(T) = x_{\text{final}} \\ & \|u(t)\|_2 \leq U_{\max}, \quad \text{for } 0 \leq t \leq T-1 \\ & x(t+1) = Ax(t) + Bu(t), \quad \text{for } 0 \leq t \leq T-1. \end{array} \tag{2}$$

The variables to optimize are x and u, where x stands for $(x(0), x(1), \ldots, x(T))$, which is the sequence of states of the robot, and u stands for $(u(0), u(1), \ldots, u(T-1))$, which is the control signal we apply.

Thus, in the optimization problem (2), we are composing the state trajectory and the control signal, jointly. Of course, these variables are linked by the physics of the problem, that is, the robot dynamics; this link is expressed by the last constraint in (2).

The first two wishes (transfer and bounded control). The first two wishes are enforced by the first three constraints, which force the initial and final state to be the given ones and the magnitude of the control signal to be upper bounded by the given maximum admissible value.

The last two wishes (waypoints and simple control). The matrix E in the cost function of (2) is

$$E = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},$$

therefore $Ex(\tau_k)$ gives the position of the robot at time τ_k: $Ex(\tau_k) = p(\tau_k)$. Hence the first sum in the cost function,

$$\sum_{k=1}^{K} \|Ex(\tau_k) - w_k\|_2^2\,, \tag{3}$$

penalizes deviations of the position of the robot from the waypoints, at the given appointed times. This sum takes care of the third wish (waypoints).

The fourth wish (simple control) is dealt with by the last sum in the cost function,

$$\sum_{t=1}^{T-1} \|u(t) - u(t-1)\|_2^2\,, \tag{4}$$

which penalizes deviations of the control signal at time t from its previous value at time $t-1$, across the time horizon. The sum in (4) is the *regularizer.* We call it a ℓ_2^2 regularizer because it uses the *square* of the ℓ_2 norm of the control increments.

The two sums (3) and (4) are mixed additively in the cost function of (2) by a weight $\lambda > 0$, the *regularization parameter.* Increasing λ gives more importance to the fourth wish, therefore playing down the third wish; decreasing λ does the opposite.

Constants. For problem (2), consider $T = 80$, $p_{\text{initial}} = (0, 5)$, and $p_{\text{final}} = (15, -15)$. Also, consider the $K = 6$ waypoints

$$w_1 = \begin{bmatrix} 10 \\ 10 \end{bmatrix}, w_2 = \begin{bmatrix} 20 \\ 10 \end{bmatrix}, w_3 = \begin{bmatrix} 30 \\ 10 \end{bmatrix}, w_4 = \begin{bmatrix} 30 \\ 0 \end{bmatrix}, w_5 = \begin{bmatrix} 20 \\ 0 \end{bmatrix} \text{ and } w_6 = \begin{bmatrix} 10 \\ -10 \end{bmatrix},$$

with appointed times

$$\tau_1 = 10, \tau_2 = 25, \tau_3 = 30, \tau_4 = 40, \tau_5 = 50, \text{ and } \tau_6 = 60.$$

The maximum control magnitude is $U_{\max} = 100$.

Task 1. Use the software `CVX` from `http://cvxr.com/cvx` to solve problem (2) for $\lambda \in \{10^{-3}, 10^{-2}, 10^{-1}, 10^{0}, 10^{1}, 10^{2}, 10^{3}\}$. (Hint: For squared-Euclidean norm terms such as $\|\cdot\|_2^2$, check the function `square_pos` of CVX.) So, you solve 7 problems (one per value of λ in the list). After each problem is solved, do the following:

(a) plot the optimal positions of the robot from $t = 0$ to $t = T$, marking its positions at the appointed times τ_k, for $1 \leq k \leq K$;

(b) plot the optimal control signal $u(t)$, where $u(t) = (u_1(t), u_2(t))$, from $t = 0$ to $t = T - 1$;

(c) report how many times the optimal control signal changes from $t = 1$ to $t = T - 1$: we say that the control signal changes at time t if $\|u(t) - u(t-1)\|_2 > 10^{-4}$; this tells us how much the fourth wish of a simple control is ignored;

(d) report the mean deviation from the waypoints, defined as

$$\frac{1}{K} \sum_{k=1}^{K} \|Ex(\tau_k) - w_k\|_2 \,, \tag{5}$$

which tells us how much the third wish (waypoints) is ignored.

Note: keep in mind that in matlab the entries of vectors (and columns of matrices) are indexed from 1 onwards. But, as you see in (2), the state vectors x are indexed from 0 onwards. So, in writing your matlab code, adjust accordingly. (In particular, the appointed times in the matlab code should be shifted by one unit.)

Example. So that you can have an example to check your code, we give the answer for the case $\lambda = 10^{-1}$: Figure 1 answers part (a), and figure 2 answers part (b). The answer to part (c) is 79, that is, the optimal control signal changes at each time instant $t \in \{0, 1, \ldots, T-1\}$ (recall that $T = 80$); you can also confirm this from figure 2. Finally, the answer to part (d), the mean waypoint deviation (5), is 2.1958.

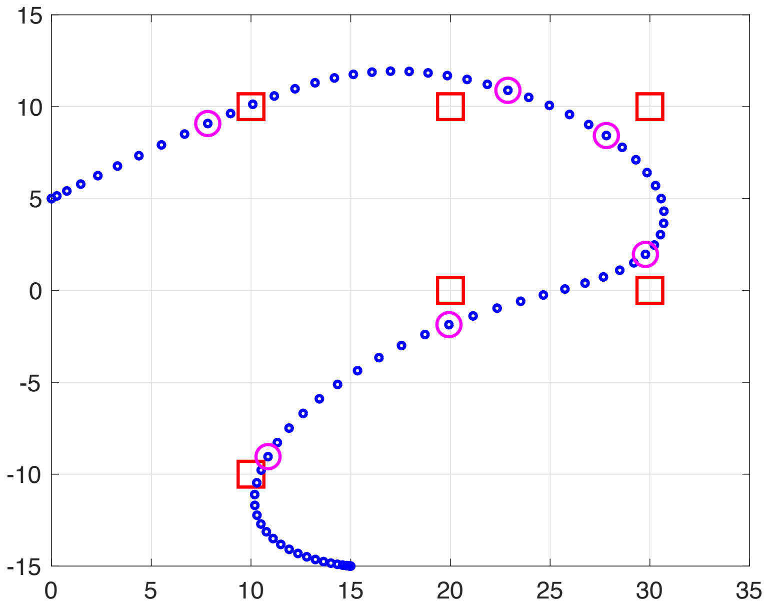

Figure 1: Positions of the robot from $t = 0$ to $t = T$ are the small blue circles; the positions at appointed times τ_k, for $1 \leq k \leq K$, are the large magenta circles. The waypoints are the red squares. Case $\lambda = 10^{-1}$ with ℓ_2^2 regularizer—see problem (2).

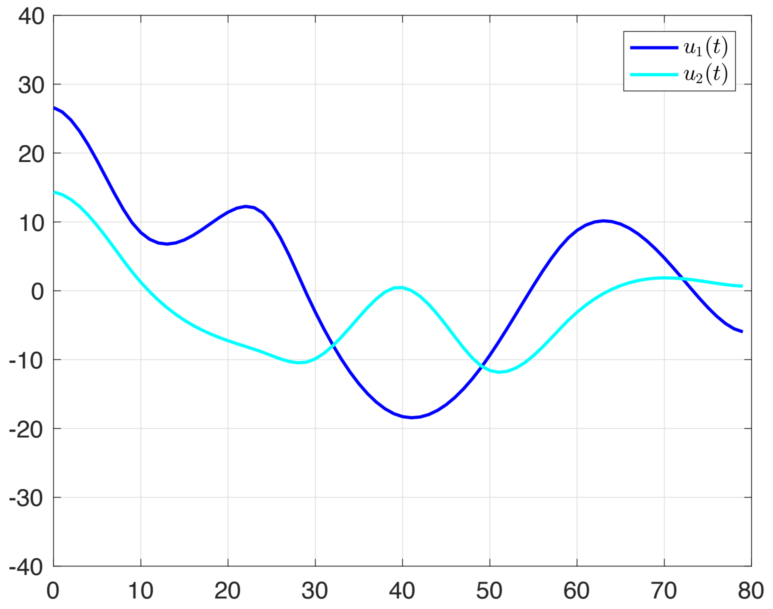

Figure 2: The two components, $u_1(t)$ and $u_2(t)$, of the control signal $u(t)$ from $t = 0$ to $t = T - 1$. Case $\lambda = 10^{-1}$ with ℓ_2^2 regularizer—see problem (2).

1.3 The ℓ_2 regularizer

We now remove the square from the ℓ_2 regularizer in (2) and formulate the problem as follows:

$$\begin{array}{ll} \underset{x,u}{\text{minimize}} & \sum_{k=1}^{K} \|Ex(\tau_k) - w_k\|_2^2 + \lambda \sum_{t=1}^{T-1} \|u(t) - u(t-1)\|_2 \\ \text{subject to} & x(0) = x_{\text{initial}} \\ & x(T) = x_{\text{final}} \\ & \|u(t)\|_2 \leq U_{\max}, \quad \text{for } 0 \leq t \leq T-1 \\ & x(t+1) = Ax(t) + Bu(t), \quad \text{for } 0 \leq t \leq T-1. \end{array} \tag{6}$$

Task 2. Redo task 1 for the optimization problem (6).

Example. We give the answer for $\lambda = 10^{-1}$, which you should reproduce. The answers to parts (a) and (b) are in figures 3 and 4, respectively. The answer to part (c) is 8, and the answer to part (d) is 0.7021.

1.4 The ℓ_1 regularizer

As a final way to tackle the fourth wish (simple control), we use a regularizer based on the ℓ_1 norm. For a vector $x \in \mathbf{R}^n$ with $x = (x_1, \ldots, x_n)$, the ℓ_1 norm is

$$\|x\|_1 = |x_1| + \cdots + |x_n|.$$

We formulate the problem as follows:

$$\begin{array}{ll} \underset{x,u}{\text{minimize}} & \sum_{k=1}^{K} \|Ex(\tau_k) - w_k\|_2^2 + \lambda \sum_{t=1}^{T-1} \|u(t) - u(t-1)\|_1 \\ \text{subject to} & x(0) = x_{\text{initial}} \\ & x(T) = x_{\text{final}} \\ & \|u(t)\|_2 \leq U_{\max}, \quad \text{for } 0 \leq t \leq T-1 \\ & x(t+1) = Ax(t) + Bu(t), \quad \text{for } 0 \leq t \leq T-1. \end{array} \tag{7}$$

Task 3. Redo task 1 for the optimization problem (7).

Example. We give the answer for $\lambda = 10^{-1}$. Parts (a) and (b) are in answered in figures 5 and 6, respectively. The answer to part (c) is 14, and the answer to part (d) is 0.8863.

Task 4. Comment on what you have observed from Tasks 1 to 3 (for example, compare the impact of the three regularizers on the optimal control signal that they each induce).

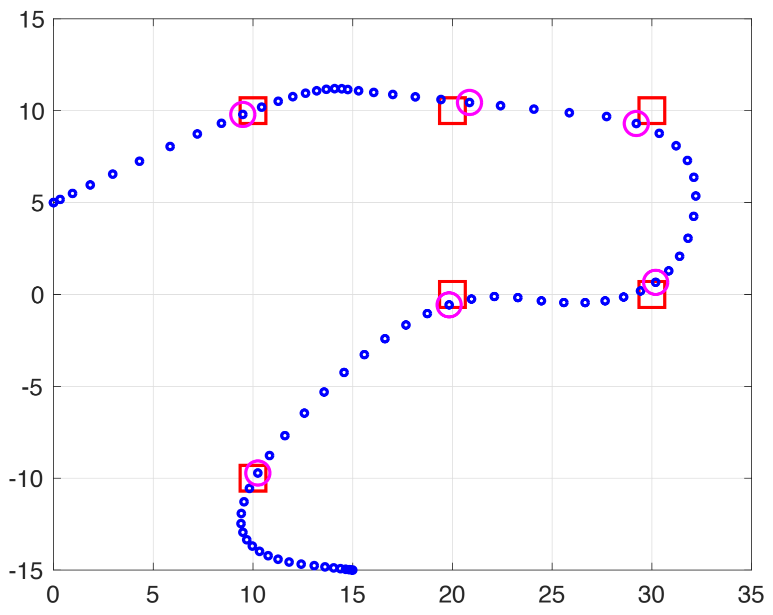

Figure 3: Positions of the robot from $t = 0$ to $t = T$ are the small blue circles; the positions at appointed times τ_k, for $1 \leq k \leq K$, are the large magenta circles. The waypoints are the red squares. Case $\lambda = 10^{-1}$ with ℓ_2 regularizer—see problem (6).

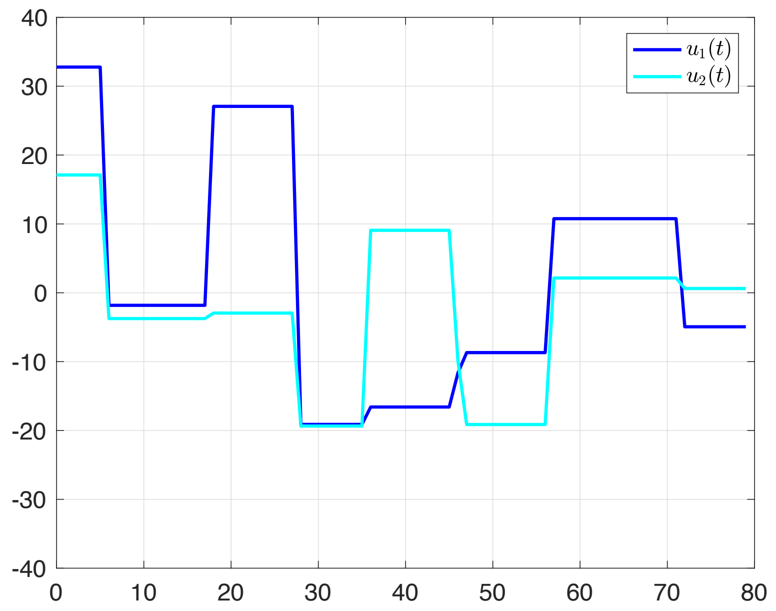

Figure 4: The two components, $u_1(t)$ and $u_2(t)$, of the control signal $u(t)$ from $t = 0$ to $t = T - 1$. Case $\lambda = 10^{-1}$ with ℓ_2 regularizer—see problem (6).

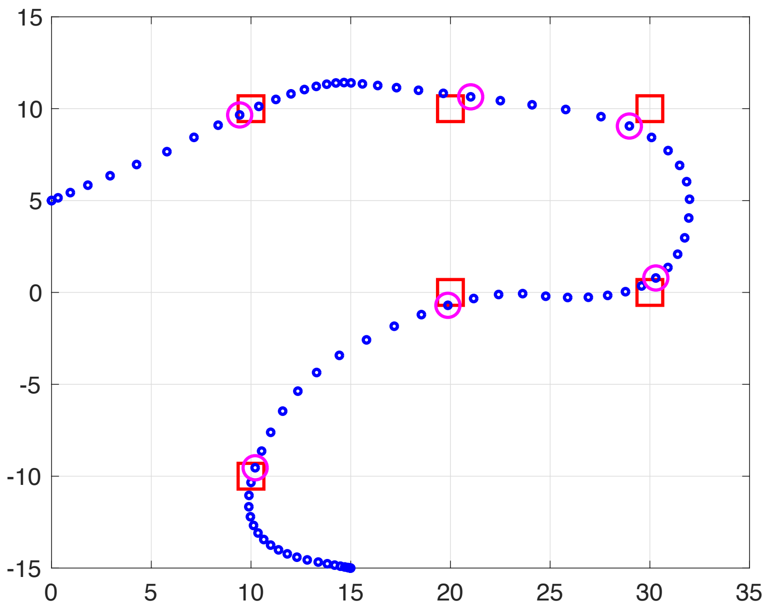

Figure 5: Positions of the robot from $t = 0$ to $t = T$ are the small blue circles; the positions at appointed times τ_k, for $1 \leq k \leq K$, are the large magenta circles. The waypoints are the red squares. Case $\lambda = 10^{-1}$ with ℓ_1 regularizer—see problem (7).

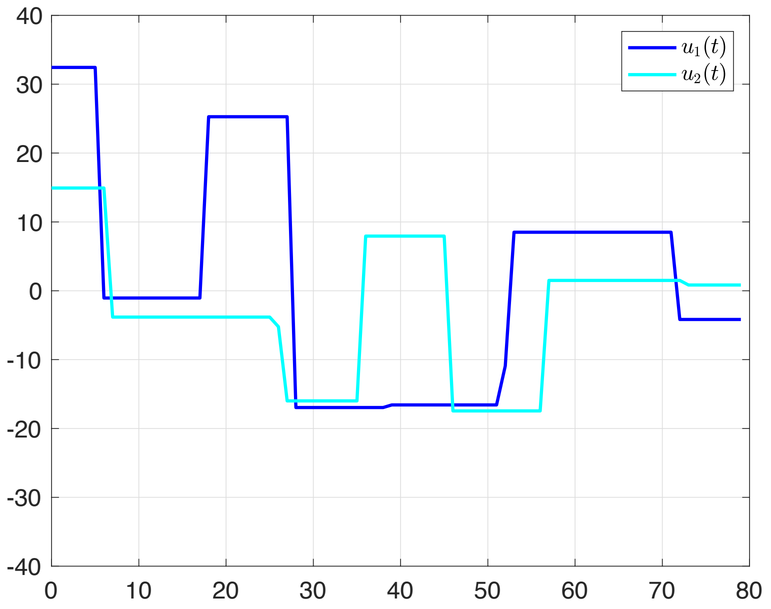

Figure 6: The two components, $u_1(t)$ and $u_2(t)$, of the control signal $u(t)$ from $t = 0$ to $t = T - 1$. Case $\lambda = 10^{-1}$ with ℓ_1 regularizer—see problem (7).

2 Locating a moving target

2.1 The setup

Suppose a target moves with constant velocity, that is, we have

$$p(t) = p_0 + tv, \tag{8}$$

where $p(t)$ is the position of the target at time t, the vector p_0 is its initial position, and the vector v is its velocity. We consider a target that moves in the plane $\mathbf{R}^2$, therefore the vectors $p(t)$, p_0, and v are two-dimensional.

We don't know either the initial position p_0 or the velocity vector v of the target. We have only partial knowledge about the position of the target at some given past time instants $t_1, \ldots, t_K$: specifically, we know that the target was inside a given disk $D(c_k, R_k)$ at time t_k, for each $1 \leq k \leq K$, where the symbol $D(c, R)$ denotes a disk with center c and radius R, $D(c, R) = \{x \in \mathbf{R}^2 \colon \; \|x - c\|_2 \leq R\}$.

2.2 Distance from a critical point

We want to know how close the target can be to a given critical point $x^\star \in \mathbf{R}^2$ at a given future time $t^\star$. Take $x^\star = (6, 10)$ and $t^\star = 8$.

Example. Consider $K = 5$ and the data in Table 1. We can visualize the data available

k	t_k	c_k	R_k
1	0	(-1.7210, -4.3454)	0.9993
2	1	(1.0550, -3.0293)	1.4618
3	1.5	(2.9619, -1.5857)	2.2617
4	3	(3.8476, 1.2253)	1.0614
5	4.5	(7.1086, 4.9975)	1.6983

Table 1: Data for the example in Section 2.

in this table as in Figure 7.

We wish to know how close a moving target can be to $x^\star$ at time t^*, for a target that moves as in (8) and that was inside the disk $D(c_k, R_k)$ at time t_k, for $1 \leq k \leq K$.

It turns out that, for this example (which you should reproduce), the closest that such a moving target can be from $x^\star$ at time $t^\star$ is 4.4948. You can check that the choice

$$p_0 = (-0.7596, -4.0725), \quad v = (1.3855, 1.9125) \tag{9}$$

in (8) puts the target in the given disks at their associated times, and, at time $t^\star$, it puts the target at $(10.3240, 11.2274)$ (whose distance from $x^\star$ is 4.4948). See Figure 8.

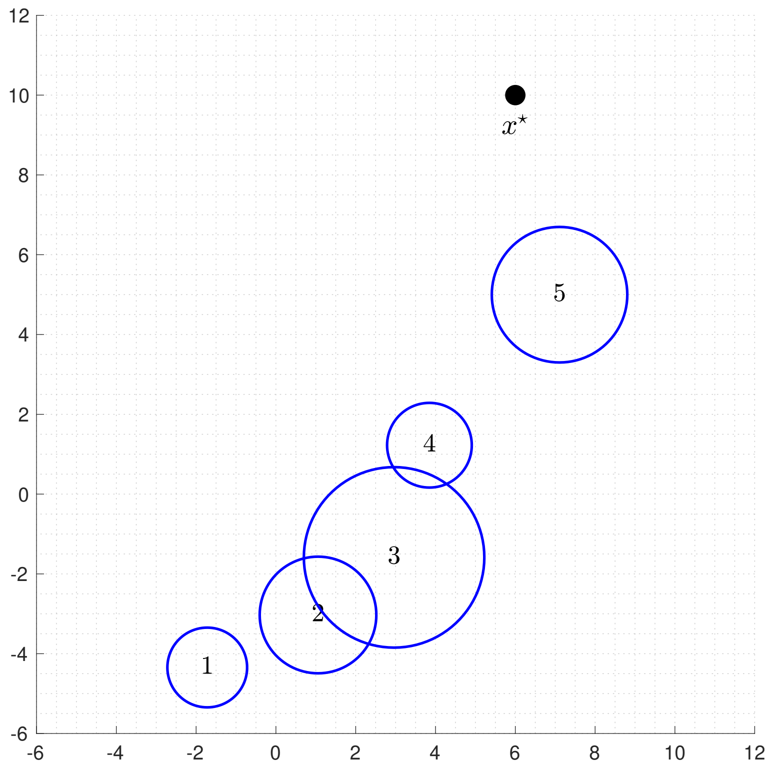

Figure 7: Visualization of the data available for the example in Table 1. At time t_k, we know only that the target was in the disk k. The boundaries of the disks are in blue. The critical point $x^\star$ is the black dot.

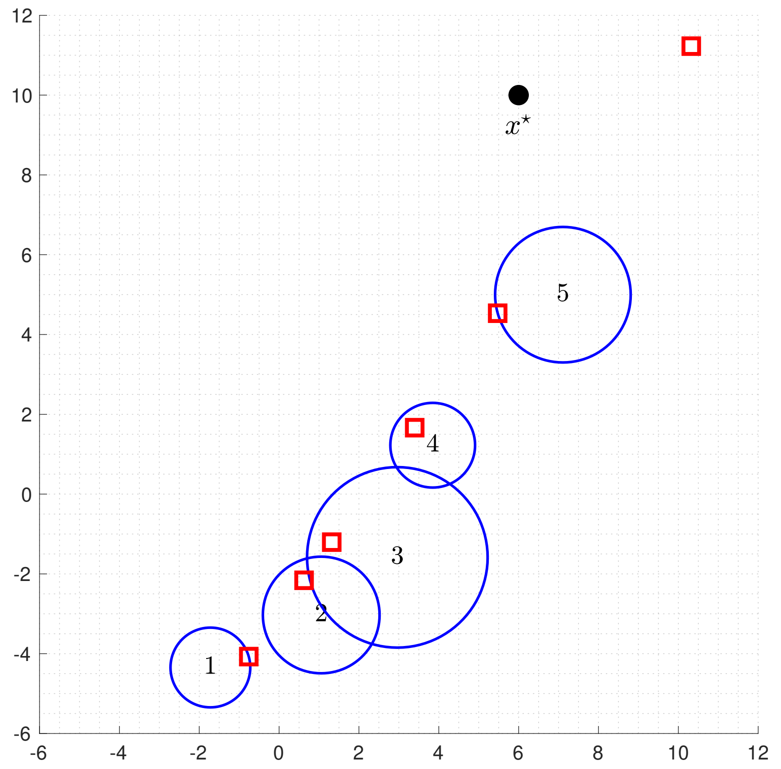

Figure 8: The solution for the data in Table 1. The red squares inside the circles are the positions at times $t_1, \dots, t_K$ of a target that moves with parameters as in (9) . The red square in the top right is the position of that target at time $t^\star$.

Task 5. Consider now the data in Table 2, which is visualized in Figure 9. Find out how close a target (which complies with the data in Table 2) can be to $x^\star$ at time $t^\star$. Describe how you formulate this task as an optimization problem and plot the solution like in Figure 8.

k	t_k	c_k	R_k
1	0	(0.6332, -3.2012)	2.2727
2	1	(-0.0054, -1.7104)	0.7281
3	1.5	(2.3322, -0.7620)	1.3851
4	3	(4.4526 3.1001)	1.8191
5	4.5	(6.1752, 4.2391)	1.0895

Table 2: Data for the Task 5.

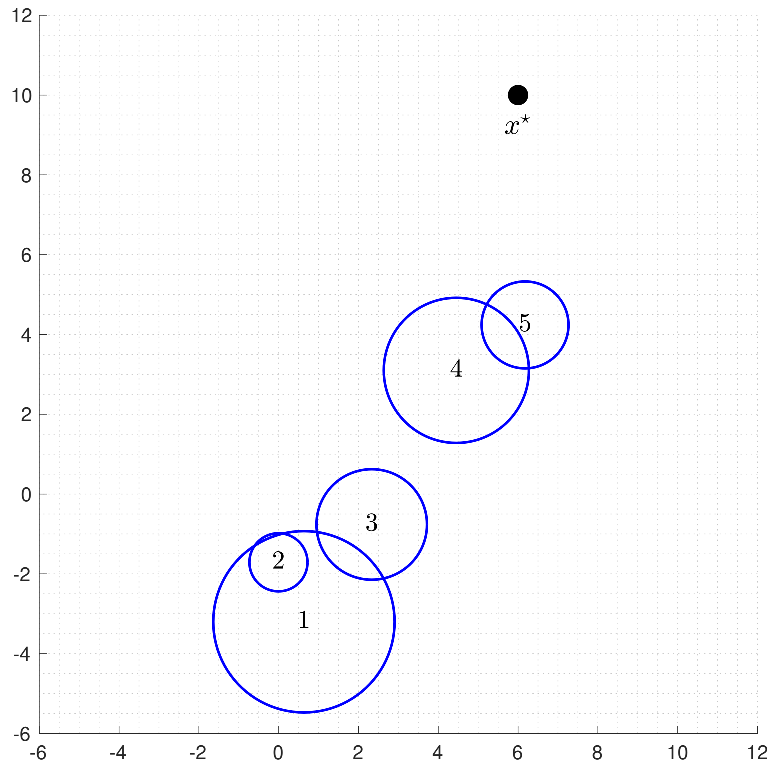

Figure 9: Visualization of the data available for the example in Table 2. At time t_k, we know only that the target was in the disk k. The boundaries of the disks are in blue. The critical point $x^\star$ is the black dot.

2.3 Smallest enclosing rectangle

Consider the setup in section 2.1. Now we want to find a rectangle that, for sure, contains the position of the target at time $t^\star = 8$. That is, we want to find a_1, a_2, b_1, b_2 such that the rectangle $R(a_1, a_2, b_1, b_2) = \{(x_1, x_2) \in \mathbf{R}^2 \colon a_1 \leq x_1 \leq a_2, b_1 \leq x_2 \leq b_2\}$ contains all possible positions for a target that moves as in (8) and that was in disk $D(c_k, R_k)$ at time t_k, for $1 \leq k \leq K$. Naturally, we want to find the *smallest* such rectangle.

Task 6. Consider the data in Table 2. Report to us your choice of a_1, a_2, b_1, b_2 and describe which optimization problems you solved to get a_1, a_2, b_1, b_2. Also, display your rectangle $R(a_1, a_2, b_1, b_2)$ on Figure 9.