CS 4758/6758: Robot Learning: Homework 4

Due: Mar 19, 5pm

1 Logistic Regression: Gradient Descent (20pts)

A robot is trying to classify whether it is seeing a obstacle or not. (I.e., $y \in \{0, 1\}$.) Given input $x \in \Re^k$, we will use logistic regression, i.e., $p(y|x;\theta) = (h_\theta(x))^y(1 - h_\theta(x))^{1-y}$, where $h_\theta(x) = \frac{1}{1+\exp -\theta^T x}$.

(a) For learning the optimal value of θ, we will use gradient descent. Assume that the m training examples were generated independently. Use the maximum likelihood principle, $\max_\theta \prod_{i=1}^m p(y^{(i)}|x^{(i)};\theta)$, to derive the update for the parameters θ using gradient descent.

(b) During testing phase, given x and θ, write an expression to estimate the output $\hat{y}$.

(c) Write the name of the function in some language (Matlab,C++, etc.) that you would use for logistic regression.

2 Kernels (15pts)

One method to improve performance is to use kernels with SVM. A kernel is a function of two input datapoints x and z, but not all functions are valid kernels. One method to prove that a kernel $K(x, z)$ is a valid kernel, is by proving that $K(x, z)$ can be written as a dot product in high dimensions, i.e., $K(x, z) = \phi(x)^T\phi(z)$ for some $\phi(x)$.

In this problem, we consider original data is two-dimensional, i.e., $x = (x_1, x_2) \in \Re^2$ and $z = (z_1, z_2) \in \Re^2$. For $K_1(x, z) = (x^T z + c)^2$, prove that it is a valid kernel by showing that $K_1 = \phi(x)^T\phi(z)$ for $\phi(x) = (x_1x_1, x_1x_2, x_2x_1, x_2x_2, \sqrt{2c}x_1, \sqrt{2c}x_2, c)$.

3 Linear Systems (10 pts)

1. We define the state of a robot (e.g., for a helicopter to stabilize its height) by $x = (h, \dot{h})$, where h is the height of the helicopter and $\dot{h}$ would be the velocity in vertical direction (rate of change of height). We model it as a *continuous* time linear system $\dot{x} = Ax$, where $A \in \Re^{2\times2}$ would describe the linear system.

   We bought two helicopters, based on measurements, we found out that their effective dynamics matrix is given as follows.

$$A_1 = \begin{pmatrix} 0 & 1 \\ 0.1 & 1 \end{pmatrix}$$

$$A_2 = \begin{pmatrix} 0 & 1 \\ -0.1 & -1 \end{pmatrix}$$

   Please determine explain which helicopter(s) will be passively stable out of the box.[1]

2. Now you were hired to comment on a autonomous flying plane robot a company is building. The plane's state in this case is determined by $x = (u, v, \theta, q)$, where: u is the velocity of aircraft along body axis; v is the velocity of aircraft perpendicular to body axis (down is positive); θ is the angle between body axis and horizontal (up is positive); q is the angular velocity of aircraft (pitch rate).

[1]Hint: Look at the eigenvalues.

The plane engineers told you that the continuous time linearized dynamics matrix of the plane flying at 774ft in this case is given by:

$$A_1 = \begin{pmatrix} 0.003 & 0.039 & 0 & 0.322 \\ 0.065 & 0.319 & 7.74 & 0 \\ 0.020 & 0.101 & 0.429 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

Please explain if their plane is stable or not? (Hint: Use eigenvalues again!)

4 Arm Path Planner (55 points)

Overview

In this question, you have to build a path planner for the two degree-of-freedom arm at right.

You know the locations of all obstacles but need to find a path around them. The path should be specified in configuration space–that is, as a set of θ_1, θ_2 angles. Write a program, using any planning algorithm you like, that can accomplish this. A successful path will:

1. Move the arm's tip from the specified start to finish
2. Keep the arm's joints within the range $-90° \leq \theta_i \leq 90°$
3. Prevent the arm from colliding with any obstacles

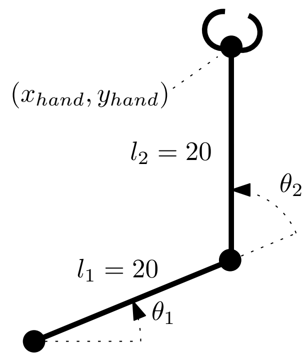


Figure 1: the arm

Steps:
(a) First convert the problem to configuration space, i.e., (θ_1, θ_2). Plot the obstacle locations, and start/goal locations of the arm in this new space. (The start and end locations will be point in this space.)
(b) Use a potential field attractor function to attract the arm to the goal.
(c) Use a potential function to repel the arm from the obstacle areas.
(d) Run gradient descent (or a similar algorithm) to find a path.

Specification

In the file `easy.txt` (Figure 2a) and `medium.txt` (Figure 2b), you are given desired (x, y) in the first line, followed by $(x_{min}, y_{min}, x_{max}, y_{max})$ defining a rectangle in the next line

The arm always starts from $\theta_1 = 90°, \theta_2 = 0°$.

You should produce a third csv file as output, containing an acceptable path for the arm. The path is written as a set of configurations, one per line. That is, each line should contain the angles θ_1, θ_2 in degrees and separated by commas. (So for example, the point $\theta_1 = 30°, \theta_2 = 45°$ would be written as: $30, 45$.) The arm should be able to smoothly move from one configuration to the next without hitting anything. Please give at least 10 configurations (i.e. your file should have at least 10 lines).

Please do not deviate from the specified format–we will be testing your paths.

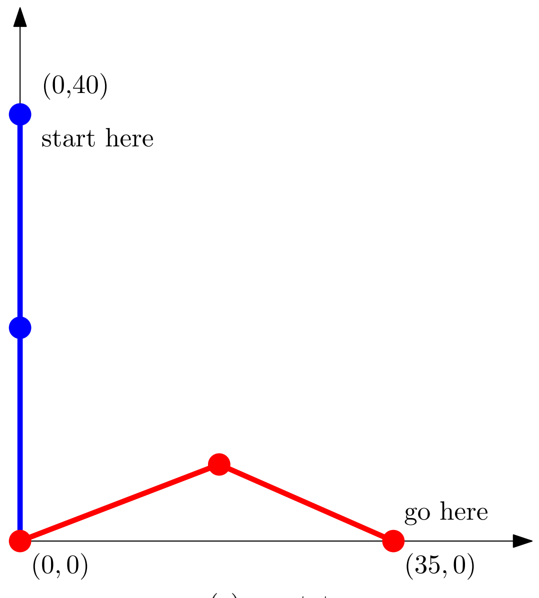
Submission

Using the subject line [**hw4armanswer**], send an email to `cs4758.qa@gmail.com` with your program's output (i.e., csv file) for `easy` and `medium` scenarios. attached.

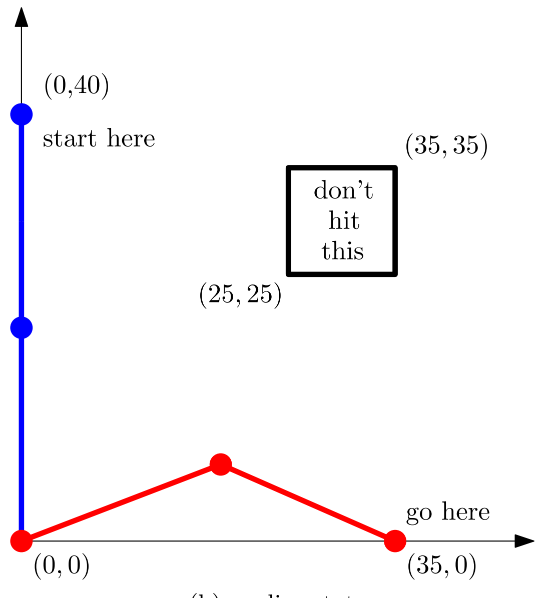
Also, on paper, submit:

1. If you deviate from the procedure above, write a high-level description of your algorithm (note that it should be able to handle planning for obstacles of similar difficulty)

2. Which scenarios would it work in? Would it work if there were more obstacles (such as Figure 3)? You do not need to solve for this case.
3. A printout of your source code
4. One graph each for `easy.txt` and `medium.txt` case for θ_1, θ_2 from start to finish.



(a) easy.txt



(b) medium.txt

Figure 2: Find paths for each of these two cases.

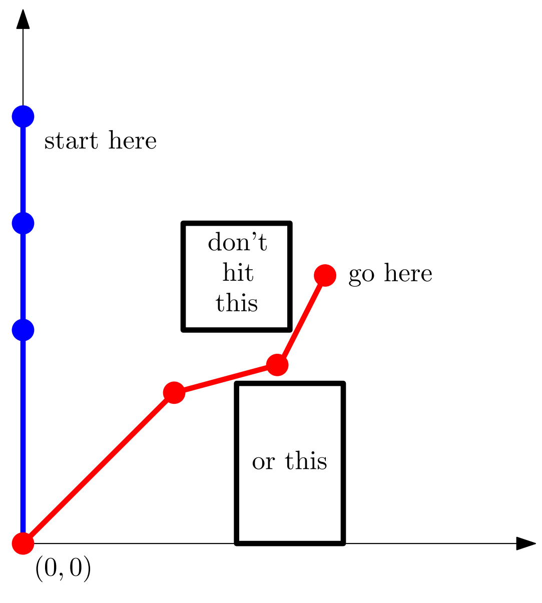


Figure 3: a more challenging task