**PSL432 Theoretical Physiology**
**Department of Physiology**
**University of Toronto**

# ASSIGNMENT 1

Assigned 28 January 2020
Due 11 February 2020

This assignment is worth 25% of your final mark in the course. Please do all parts of all 3 questions. For each question you will submit computer code as an m-file. Please email these m-files to douglas.tweed@utoronto.ca and ali.mojdeh@mail.utoronto.ca by 11:59 p.m. on 11 Feb. Your code will be graded partly on concision and clarity, so keep it brief and orderly, and make your m-files self-contained so Ali and I can run them in Matlab on their own, i.e. don't write any other files that are called from them.

## 1. Feedback linearization

Your job is to program a feedback-linearization policy for a 2-joint arm reaching to a jumping target. That is, you will choose linear desired dynamics for the arm and then solve analytically for the policy that produces those dynamics.

In continuous time, the arm's state dynamics are described by the differential equation,

$$\begin{aligned}\boldsymbol{a} &= \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{\Gamma}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \\ &= \begin{bmatrix} \psi_1 + 2\psi_2\cos(q_2) & \psi_3 + \psi_2\cos(q_2) \\ \psi_3 + \psi_2\cos(q_2) & \psi_3 \end{bmatrix}\ddot{\boldsymbol{q}} + \psi_2\sin(q_2)\begin{bmatrix} -\dot{q}_2 & -(\dot{q}_1 + \dot{q}_2) \\ \dot{q}_1 & 0 \end{bmatrix}\dot{\boldsymbol{q}}\end{aligned}$$

where $\boldsymbol{a}$ is the action, $\boldsymbol{q}$ is the configuration vector, containing the shoulder and elbow angles in that order, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ are velocity and acceleration, and $\psi_1$, $\psi_2$, and $\psi_3$ are constants: $\psi_1 = 2.06$, $\psi_2 = 0.65$, $\psi_3 = 0.56$. In your code, you will integrate the dynamics using Euler's method, with $\Delta t = 0.01$ s.

You should also prevent the joints going outside their motion ranges, i.e. define `q_min = [-1.5; 0]`, `q_max = [1.5; 2.5]`, and after Euler integration write:

```
q_bounded = max(q_min, min(q_max, q));
q_vel = (q == q_bounded).*q_vel;
q = q_bounded;
```

Make the target joint angles (the elements of $\boldsymbol{q}^*$) jump once every 0.5 s, keeping them inside the joint ranges like this:

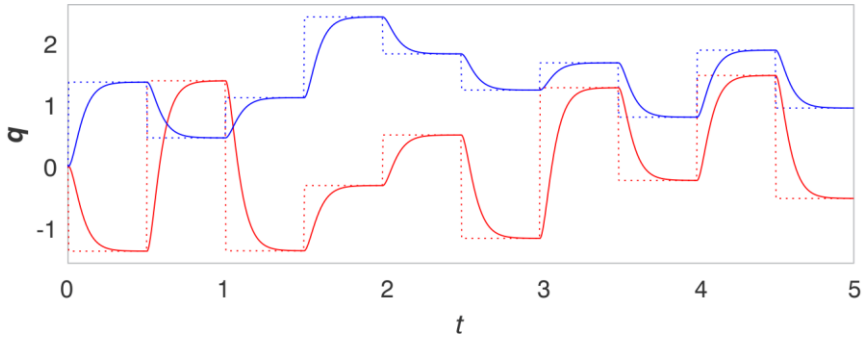
```
q_star = q_min + (q_max - q_min).*rand(n_q, 1);
```

Simulate 5 s in all, i.e. 10 reaches to 10 targets.

Choose desired dynamics that get both joints to their targets within 0.5 s, and find the policy that produces those dynamics. Assume the agent knows the dynamics of the arm (including $\psi_1$, $\psi_2$, and $\psi_3$) and the state (including $\boldsymbol{q}$, discrete-time velocity $\boldsymbol{q}^{(1)}$, and $\boldsymbol{q}^*$). Also choose your desired dynamics to keep both elements of $\boldsymbol{a}$ smaller than about 4000 in absolute value (you needn't prove mathematically that $\boldsymbol{a}$ never crosses those bounds — just look at your simulations to check that it normally stays in about that range).

In the handout notes, we derived policies using scalar operations, but here you will need vectors and matrices, because $\boldsymbol{q}$ has more than one element. For vector and matrix operations anywhere in your code, use Matlab's built-in functions, e.g. if you need to multiply $\boldsymbol{A}$ and $\boldsymbol{B}$ or invert $\boldsymbol{A}$, then write `A*B` or `inv(A)` — *don't* write out the complicated, element-wise formulae for these operations.

Display your results in a figure with 2 panels. In the upper panel, plot shoulder and elbow angles (as red and blue solid lines) and their targets (as red and blue dotted lines) versus time from $t = 0$ to 5 s, with y-axis limits set using `ylim(1.05*[min(q_min), max(q_max)])`. In the lower panel, plot both elements of $\boldsymbol{a}$ versus time. If all goes well, your upper-panel plot should look something like this:




Name your variables as in these pages, the notes, and the sample code, e.g. `q`, `q_star`, `q_vel`, `q_acc`, `psi`, `M`, `GAMMA`, `a`. Submit an m-file called Yoursurname_Yourgivennameinitial_A1_Q1, e.g. Tweed_D_A1_Q1.m.

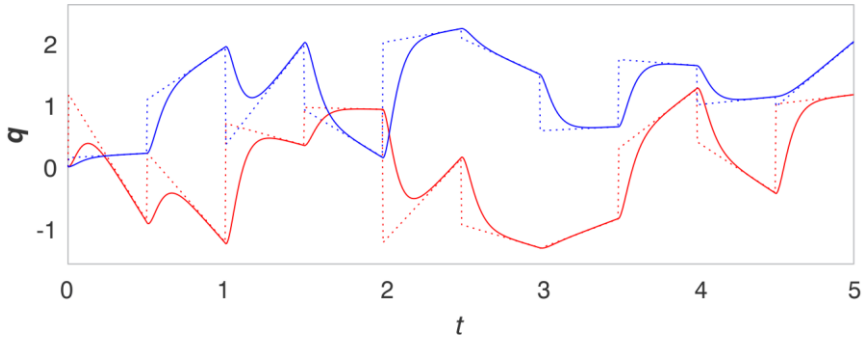
## 2. Internal feedback

Here you will again code a feedback-linearization policy for the arm of Question 1, but this time, your agent will rely on *internal* feedback. In Question 1, the agent knew $\psi_1$, $\psi_2$, $\psi_3$, $\boldsymbol{q}$, and $\boldsymbol{q}^{(1)}$. Now, it uses *estimates* of $\psi_1$, $\psi_2$, and $\psi_3$ in its policy and also to estimate $\boldsymbol{q}$ and $\boldsymbol{q}^{(1)}$. You can use the sample code Saccadic_internal.m as a partial guide, but your agent should not assume that actual arm velocity equals desired arm velocity (that assumption is safe for eye control but not for arm control, because arms are much more likely than eyes to carry unexpected loads). The agent still knows everything about arm dynamics *except* $\psi_1$, $\psi_2$, $\psi_3$, $\boldsymbol{q}$, and $\boldsymbol{q}^{(1)}$ — it knows the form of the differential equation and where $\psi_1$, $\psi_2$, $\psi_3$, $\boldsymbol{q}$ and velocity plug into it, and it knows the joint bounds. In your code, give the agent correct initial estimates of $\boldsymbol{q}$ and $\boldsymbol{q}^{(1)}$, and reset those estimates to their correct values every time the target jumps.

Apart from these differences, organize everything as in Question 1, e.g. use the same $\Delta t$ and duration, bound the joint angles in the same ranges, and plot your results in the same format. Submit an m-file called Yoursurname_Yourgivennameinitial_A1_Q2.

### 3. Moving target

Here you will program the target to glide at a random, constant velocity between jumps, with speeds ranging between 0 and about 4 or 5 rad/s, always staying inside the joint ranges, like this:




And program a policy that tracks this type of target based on internal feedback, assuming the agent has correct estimates of $\psi_1$, $\psi_2$, and $\psi_3$ and it knows the target's configuration and velocity at each moment.

Apart from this difference, you should again organize everything as in Questions 1 and 2. Submit an m-file called Yoursurname_Yourgivennameinitial_A1_Q3.