

Creating a Reinforcement Learning Controller for Functional Electrical Stimulation Control of a Human Arm

Philip Thomas¹; Antonie van den Bogert, Ph.D.^{2,3}; Kathleen Jagodnik^{2,3}; Michael Branicky, Sc.D.¹

¹) Department of Electrical Engineering and Computer Science, Case Western Reserve University, ²) Department of Biomedical Engineering, Case Western Reserve University, ³) Department of Biomedical Engineering, Lerner Research Institute, Cleveland Clinic

INTRODUCTION

Functional Electrical Stimulation (FES)

- People with spinal cord injury (SCI) are often unable to move their limbs, though most of their nerves and muscles may be intact.
- Functional Electrical Stimulation (FES) can activate these muscles to restore movement

Closed-Loop FES Control

- Open-loop control systems have been successful, but their abilities are limited. Closed-loop controllers allow tighter control, less user interaction, and can compensate for modeling errors.
- K. Jagodnik et al. (2008) have successfully implemented a Proportional Derivative (PD) Controller to control the right arm of a paralyzed patient.
- The PD Controller was trained using Dynamic Arm Simulator 1, which simulates the two-dimensional movement of a human arm given stimulation to three muscle pairs (six muscles total).

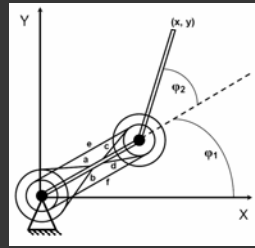


Figure 1: 2-joint, 6-muscle biomechanical arm model used by DAS 1. Antagonistic muscle pairs are as follows, listed as (flexor, extensor): (a) anterior deltoid, (b) posterior deltoid, (c) brachialis, (d) triceps brachii, (e) biceps brachii, (f) triceps brachii

Reinforcement Learning

- Reinforcement Learning (RL) allows the controller to adapt to changes in the dynamics of the arm, such as the shift from simulation to the subject, or to compensate for muscle fatigue over time.
- This project involves developing a Reinforcement Learning controller that adapts to simulated changes in arm dynamics.

Actor-Critic Reinforcement Learning Architecture

Continuous Actor-Critic

- The Actor-Critic architecture (Sutton and Barto, 1998) simplifies the RL problem of learning the value of taking each action in each state by splitting the problem in two. The actor learns the policy (which action is best in a state), while the critic learns the value of each state. This cuts the dimensionality of the problem in half.
- The actor-critic architecture differs for a continuous or discrete state space. Both follow Figure 2.
- Our implementation of the continuous actor-critic follows Doya (2000).

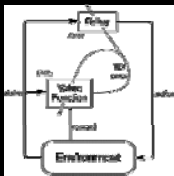


Figure 2: Actor-critic architecture diagram from Sutton and Barto (1998). The actor takes an action that changes the state of the environment and also gives a reward. The critic uses these two to compute the Temporal Difference (TD) error, which represents the difference between the expected and observed state and reward.

In our application, the environment is DAS 1, the reward is a function of the muscle forces and the distance between the desired arm position and the current one.



Figure 3: A simple (two-dimensional discrete grid) environment was setup to study the learning characteristics of the actor-critic model. The top graph shows the sum (over every possible state) of the squared error between the actor's policy and the optimal policy. The bottom graph shows the sum (over every possible state) of the squared error between the critic's value function and the optimal value function.

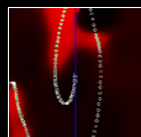
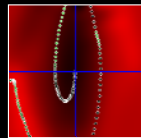


These graphs show a characteristic of the actor-critic that was observed during all trials: the actor does not converge toward the optimal policy until the critic has converged. In fact, the actor usually diverges until the critic has stabilized.

Figure 5: The actor-critic is capable of quickly learning a successful policy when given a suitable reward. An example of this is the pendulum swing-up task specified in Doya (2000), in which the agent attempts to swing up a pole initially hanging straight down ($\theta=0$), with limited torque. The images on the right depict the utility (top) and policy (bottom) after 1,000 episodes of training, each lasting 20 seconds. In both, the trajectory is overlaid.

In the utility graph (top), brighter red represents higher value (expected reward). In the policy graph (bottom), brighter red represents a larger clockwise torque, and darker represents a larger counter-clockwise torque.

In both, the x-axis represents the angle and the y-axis represents the angular velocity.



Actor-Critic Performance on Dynamic Arm Simulator 1

Setup

- Initial tests showed that the actor-critic was capable of learning a decent policy without any a priori knowledge. Practically, the actor-critic was pre-trained to mimic the PD controller developed by Jagodnik et al. (2008) before being tested for its ability to adapt to different arm dynamics.
- Both actor and critic were artificial neural networks. The actor had 10 neurons in its hidden layer; the critic had 20.
- Each episode lasts 2 seconds, during which the arm must move between random initial and desired goal states.

Parameters

- Several parameters of the actor-critic can be modified, including the learning rate and the amount of exploration done by the actor to find better policies (the amount and shape of the noise injected into the actor's output).
- If improperly set, the actor-critic will not learn.
- The set of parameters that do learn well in this environment is broad. In some, the actor flops the arm around in a way seemingly irrelevant to achieving the goal state. In doing so, it is learning the dynamics of the arm, so when the exploration noise is removed it has learned the correct policy and moves quickly and smoothly to the goal. With other parameters, the exploration noise hardly changes the actor's actions at all, yet the actor-critic still learns.

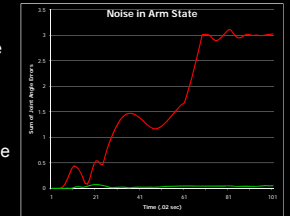


Figure 7: The sum of the absolute joint angle errors (shoulder and elbow) shown above are the difference between the arm position with and without noise. The parameters that created the red curve have significant noise, while the parameters that created the green curve do not. Both sets of parameters learn quickly and smoothly.

Adaptive Abilities

- To evaluate the performance of each controller, an evaluation scheme was devised that sums rewards as a function of position error and muscle forces over 256 fixed movements.
- Even without changing the arm dynamics, the actor-critic was able to improve slightly upon the PD controller's policy.
- Two tests were run to evaluate the actor-critic's ability to adapt to different arm dynamics.

Baseline Biceps Stimulation

- The biceps was given a constant additional stimulation of 20% of the maximum. This approximates the conditions of an observed patient.
- This changes the steady state of the arm when using the PD's policy.



Figure 8: The steady state after training for 0, 10, 50, 100, and 200 episodes, each getting progressively closer to a goal state (red).

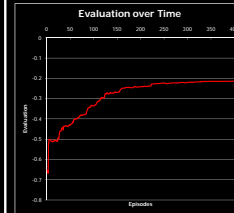


Figure 9: Plot of actor-critic's performance over time. For reference, the PD controller achieves an evaluation of -65 .

After 200 training episodes, the arm makes a smooth motion to its steady state, which is close to the goal.

Fatigued or Atrophied Triceps

- The triceps muscle was weakened by 80% to represent a fatigued or atrophied muscle.
- The steady state does not change, but (when starting clockwise of the goal) the arm overshoots initially.

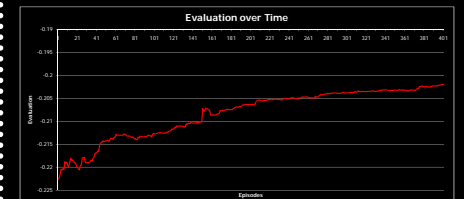


Figure 10: Plot of the actor-critic's performance over time. For reference, the actor-critic achieves an evaluation of -21 (which is equivalent to the PD controller's evaluation without any triceps weakness) after approximately 150 training episodes.

Conclusion

- An actor-critic reinforcement learning architecture is capable of learning to adapt to different dynamics in the arm in a reasonable amount of time.
- For further information contact Philip Thomas, pst5@case.edu

Future Work

- Different function approximators, such as a locally weighted regression model, or radial basis functions may perform better as the actor or the critic.
- In human trials, what parameters (specifically exploration noise size and shape) work best?

REFERENCES

1. Doya K (2000) Reinforcement Learning in Continuous Time and Space. *Neural Computation*, 12(1):219-245.
2. Jagodnik KM, Kirsch RF, van den Bogert AJ (2008) A Proportional Derivative Controller for Planar Arm Movement. Poster at ShowCASE 2008.
3. Sutton RS, Barto AG (1998) *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA

ACKNOWLEDGEMENTS: This work was funded by NIH Grant R21HD049662 and Predoctoral Fellowship F31HD049326 (Jagodnik). The authors thank Dr. Robert Kirsch for his helpful input.