CoSMo Motor Control Matlab Session 1: Optimal Control of Movements

August 4, 2014

1 Exercise 1: Distribution of effort across muscles*

Choose a set of 5 pulling directions roughly uniformly distributed from 0 to 360^o and create a 5×2 matrix P that represents the pulling direction vectors for each muscle.
The motor commands (forces) to these muscles can be represented as a 5-element vector $\mathbf{u}$. Compute the endpoint force $\mathbf{f}$ associated with a given set of motor commands.
Now determine the 'optimal' set of motor commands to achieve a specified force vector (you need only consider forces of unit size in different directions). Assume that 'optimal' means to minimize the sum of the squared motor commands, $\Sigma \mathbf{u}_i^2$. Remember that muscles can only pull, not push ($\mathbf{u}_i > 0$). (Hint: you may find the matlab function `quadprog` helpful).
Plot the 'tuning function' for each muscle.
Now artificially 'weaken' one muscle by reducing the length of its pulling direction vector. Examine the impact on the tuning function and compare to the results of de Rugy et al.(2012).
Do you agree with the conclusion of de Rugy et al. (2012), that muscle coordination is not optimal?

2 Exercise 2: Minimum Endpoint Variance Control**

Simulate a discrete-time dynamical system model of the eye:

$$\mathbf{x}_t = A\mathbf{x}_{t-1} + \mathbf{b}u_t + \epsilon_t, \tag{1}$$

with

$$\epsilon_t \sim N(0, \kappa^2 u_t^2 Q) \tag{2}$$

Suitable values for A and $\mathbf{b}$, given a time discretization of 1ms are:

$$A = \begin{pmatrix} 0.9998 & 0.0010 & 0.0002 \\ -0.3298 & 0.9217 & 0.2913 \\ 0 & 0 & 0.7788 \end{pmatrix}, \tag{3}$$

$$\mathbf{b} = \begin{pmatrix} 0 \\ 0.0385 \\ 0.2212 \end{pmatrix}, \tag{4}$$

$$Q = \begin{pmatrix} 0 & 0.0007 & 0.0027 \\ 0.0007 & 1.9227 & 8.1782 \\ 0.0027 & 8.1782 & 49.1837 \end{pmatrix}. \tag{5}$$

$$\kappa = 10^{-3} \tag{6}$$

We will assume that the eye begins at $\mathbf{x}_0 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$.
Simulate the result of a constant motor input $u = 1$.
The mean and variance of the eye's state at timestep n can be expressed as linear and quadratic functions of the time-series of motor commands $\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix}$.

$$E[\mathbf{x}_n] = A^n \mathbf{x}_0 + \sum_{i=1}^{n} A^{n-i} B u_i$$

and variance:

$$\begin{aligned} Var[\mathbf{x}_n]/\kappa &= E[\mathbf{x}_n \mathbf{x}_n^T] - E[\mathbf{x}_n] E[\mathbf{x}_n]^T \\ &= \sum_{i=1}^{n} A^{n-i} Q \left(A^{n-i}\right)^T u_i^2 \end{aligned}$$

Build a matrix G that maps the motor commands to the time-varying position $\mathbf{p}$ of the eye for a saccade that lasts 150ms:

$$\mathbf{p} = G\mathbf{u}. \tag{7}$$

(NB, we only need keep track of the position at each time. Hint: You only need compute 1 row of this matrix, other rows are shifted versions of the first row)
Check that the result of this matrix multiplication agrees with your prior simulation.
Build a matrix H that can be used to compute the endpoint variance of the eye:

$$\mathrm{Var}[\mathbf{p}_n] = \mathbf{u}^T H \mathbf{u}. \tag{8}$$

Use these matrices to determine the motor commands that minimize endpoint variance subject to the constraint that the eye remains on the target for the last 10ms (so the actual saccade will last 140ms.) HINT: Use lagrange multipliers to solve the constrained optimization problem:

$$\min \frac{1}{2} \mathbf{u}^T H \mathbf{u} + \boldsymbol{\lambda}^T \left(\bar{G}\mathbf{u} - \mathbf{k}\right), \tag{9}$$

where $\bar{G}$ is a sub-matrix of G computed earlier, and $\mathbf{k}$ is the desired endpoint of the saccade (30^o).
After setting derivatives equal to zero leads to

$$\begin{aligned} H\mathbf{u} + \bar{G}^T \boldsymbol{\lambda} &= \mathbf{0} \\ \bar{G}\mathbf{u} = \mathbf{k}. \end{aligned}$$

These equations can be combined into a single, easily-solvable matrix equation:

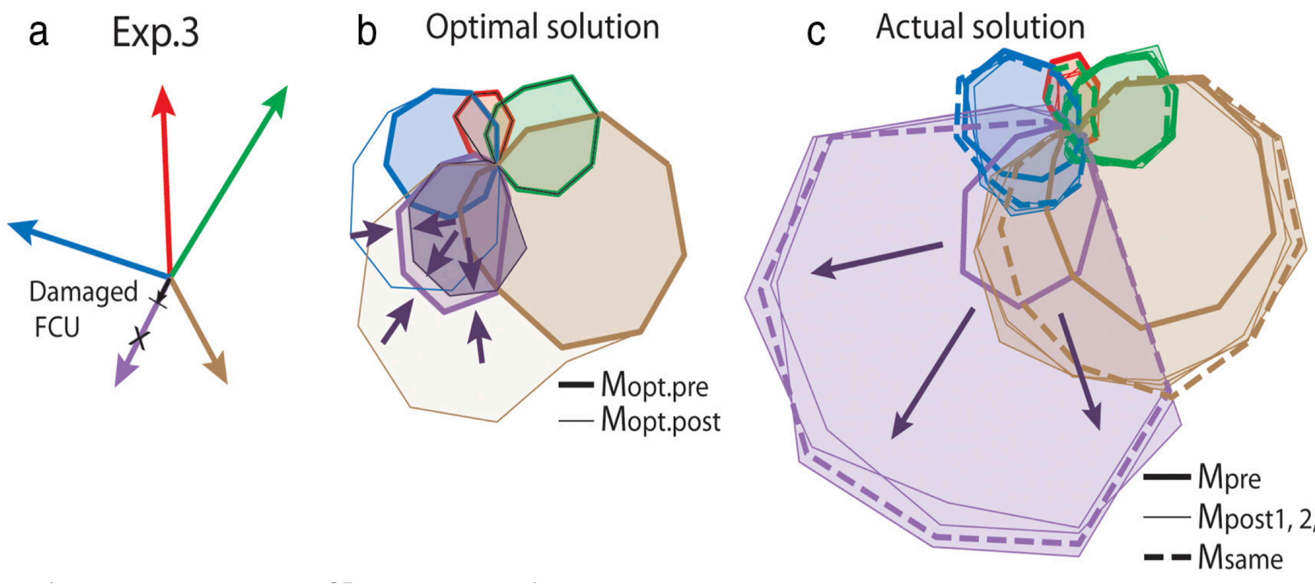
$$\begin{pmatrix} H & \bar{G}^T \\ \bar{G} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{k} \end{pmatrix} \tag{10}$$

Simulate the saccade with the optimal motor commands and plot the velocity profile.
Do the same for saccades of different durations and amplitudes and compare to the figure in Harris and Wolpert, 1998.

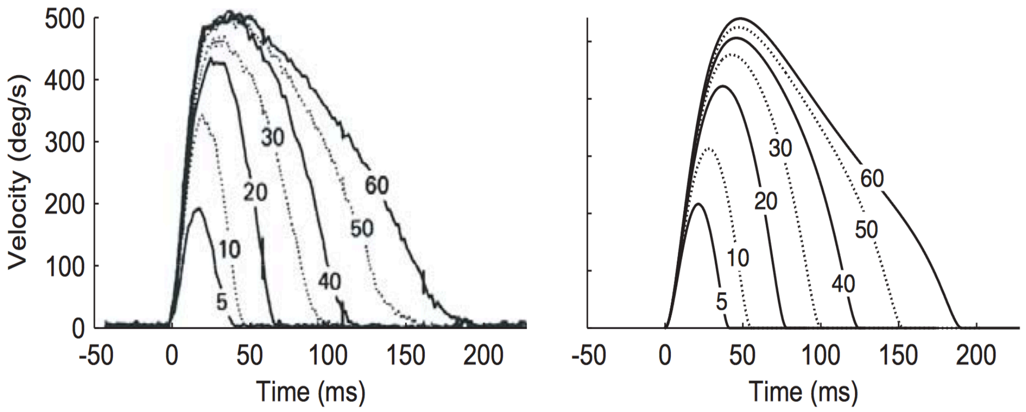
3 Exercise 3: Optimal Saccade Duration*

The supplied matlab function `saccadeOpt` performs essentially the same optimization from the previous exercise. This function also returns the endpoint variance of the saccade.
Determine the probability of success for a saccade of 40^o amplitude as a function of movement duration. (Hint: use the error function)
Determine the rate of reward obtained as a function of movement duration with an inter-trial-interval of 1s and identify the optimal movement duration for a 40^o saccade.
Do the same for a range of saccade amplitudes to reproduce Figure 1C from Haith et al., 2012.
Explore the effect of varying ITI on the optimal saccade duration.

Exercise 1: Effect of muscle weakness on recruitment:

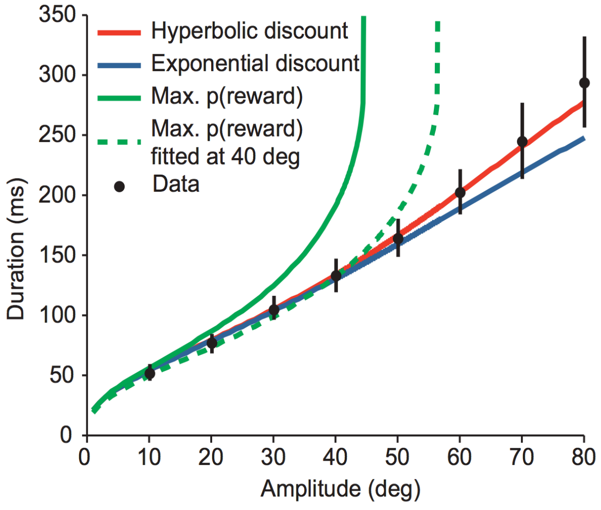


Exercise 2: Trajectories of saccades of varying amplitude/duration:



Note that it is the duration rather than amplitude per se that alters the predicted shape of the saccade

Exercise 3: Optimizing saccade duration through rate of reward



The red line represents the movement duration predicted by rate of reward with an ITI of 1s