Hit and Run and Stuff

Brian Cohn May Szedlák

March 15, 2015

Abstract

The brain must select its control strategies among an infinite set of possibilties, thereby solving an optimization problem. While this set is infinite and lies in high dimensions, it is bounded by kinematic, neuromuscular, and anatomical constraints, within which the brain must select optimal solutions. We use data from a human index finger with 7 muscles, 4DOF, and 4 output dimensions. For a given force vector at the endpoint, the feasible activation space is a 3D convex polytope, embedded in the 7D unit cube. It is known that explicitly computing the volume of this polytope can become too computationally complex in many instances. We generated random points in the feasible activation space using the Hit-and-Run method, which converged to the uniform distribution. After generating enough points, we computed the distribution of activation across each muscle, shedding light onto the structure of these solution spaces- rather than simply exploring their maximal and nimimal values. We also visualize the change in these activation distributions as we march toward maximal feasible force production in a given direction. Using the parallel coordintes method, we visualize the connection between the muscle activations. Once can then explore the feasible activation space, while constraining certain muscles. Although this paper presents a 7 dimensional case of the index finger, our methods extend to systems with up to at least 40 muscles. We challenge the community to map the shapes distributions of each variable in the solution space, thereby providing important contextual information into optimization of motor cortical function in future research.

1 Author Summary

2 Introduction

Described in a mathematical way the feasible activation set is expressed as follows. For a given force vector $f \in \mathbb{R}^m$, which are the activations that satisfy

$$\mathbf{f} = A\mathbf{a}, \mathbf{a} \in [0,1]^n?$$

In our 7-dimensional example $m = 4$ and $n = 7$, typically n is much larger than m. The constraint $\mathbf{a} \in [0,1]^n$ describes that the feasible activation space lies in the n-dimensional unit cube (also called the n-cube). Each row of the constraint $\mathbf{f} = A\mathbf{a}$ is a $n-1$ dimensional hyperplane. Assuming that the rows in A are linearly independent (which is a safe assumption in the muscle system case), the intersection of all m equality constraints constraints is a $(n-m)$-dimensional hyperplane. Hence the feasible activation set is the polytope given by the intersection of the n-cube and the $(n-m)$-dimensional hyperplane. Note that this intersection is empty in the case where the force f can not be generated.

3 Materials and Methods

Exact volume calculations for polygons can only be done in reasonable time in up to 10 dimensions [2, 5, 6]. We therefore use the so called Hit-and-Run approach, which samples points in a given polygon uniformly at random. Given the points for a feasibale activation space, this method gives us a deeper understanding of its underlying structure. In this section we introduce

3.1 Hit-and-Run

The Hit-and-Run algorithm used for sampling in a convex body K, was introduced by Smith in 1984 [9]. The mixing time is known to be $\mathcal{O}^*(n^2R^2/r^2)$, where R and r are the radii of the inscribed and cicumscribed ball of K respectively [1, 7]. In the case of the muscles of a limb, we are interested in the polygon P that is given by the set of all possible activations $\mathbf{a} \in \mathbb{R}^n$ that satisfy

$$\mathbf{f} = A\mathbf{a}, \mathbf{a} \in [0,1]^n,$$

where $\mathbf{f} \in \mathbb{R}^m$ is a fixed force vector and $A = J^{-T}RF_m \in \mathbb{R}^{m\times n}$. P is bounded by the unit n-cube since all variables a_i, $i \in [n]$ are bounded by 0 and 1 from below, above respectively.

Consider the following 1×3 example.

$$1 = \frac{10}{3}a_1 - \frac{53}{15}a_2 + 2a_3$$
$$a_1, a_2, a_3 \in [0,1],$$

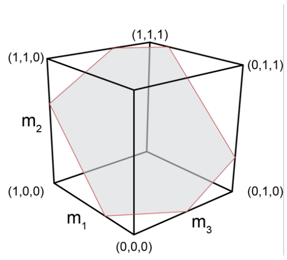


Figure 1: Feasible Activation

the set of feasible activations is given by the shaded set in Figure 1.

The Hit-and-Run walk on P is defined as follows (it works analogously for any convex body).

1. Find a given starting point $\mathbf{p}$ of P (Figure 2a) .

2. Generate a random direction through $\mathbf{p}$ (uniformly at random over all directions) (Figure 2b).

3. Find the intersection points of the random direction with the n-unit cube (Figure 2c).

4. Choose the next point of the sampling algorithm uniformly at random from the segment of the line in P (Figure 2d).

5. Repeat from (b) the above steps with the new point as the starting point .

The implementation of this algorithm is straight forward except for the choice of the random direction. How do we sample uniformly at random (u.a.r.) from all directions in P? Suppose that $\mathbf{q}$ is a direction in P and $p \in P$. Then by definition of P, $\mathbf{q}$ must satisfy $\mathbf{f} = A(\mathbf{p} + \mathbf{q})$. Since $\mathbf{p} \in P$, we know that $\mathbf{f} = A\mathbf{p}$ and therefore

$$\mathbf{f} = A(\mathbf{p} + \mathbf{q}) = \mathbf{f} + A\mathbf{q}$$

and hence

$$A\mathbf{q} = 0.$$

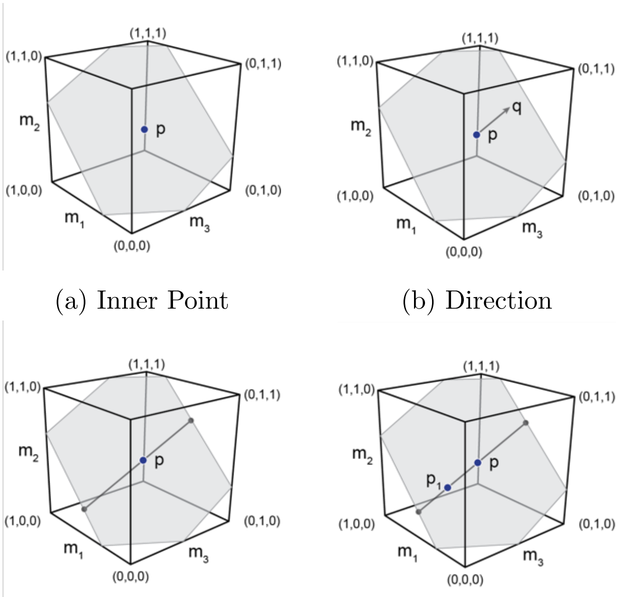
We therefore need to choose directions uniformly at random from all directions in the vectorspace

$$V = \{\mathbf{q} \in \mathbb{R}^n | A\mathbf{q} = 0\}.$$

As shown by Marsaglia this can be done as follows [8].

1. Find an orthonormal basis $b_1, \ldots, b_r \in \mathbb{R}^n$ of $A\mathbf{q} = 0$.

2. Choose $(\lambda_1, \ldots, \lambda_r) \in \mathcal{N}(0,1)^n$ (from the Gaussian distribution).



(a) Inner Point (b) Direction

(c) Endpoints (d) New Point

Figure 2: Hit-and-Run Step

3. $\sum_{i=1}^{r} \lambda_i b_i$ is a u.a.r. direction.

A basis of a vectorspace V is a minimal set of vectors that generate V, and it is orthonormal if the vectors are pairwise orthogonal (perpendicular) and have unit length. Using basic linear algebra one can find a basis for $V = \{A\mathbf{q} = 0\}$ and orthogonalize it with the well known Gram-Schmidt method (for details see e.g. [3]). Note that in order to get the desired u.a.r. distribution the basis needs to be orthonormal. For the limb case we can safely assume that the rows of A are linearly independent and hence the number of basis vectors is $n - m$.

3.2 Mixing and Stopping Time

In this section we discuss the stopping time of the Hit and Run algorithm. How many steps are necessary to reach an approximate uniform distribution? The theoretical bound $\mathcal{O}^*(n^2R^2/r^2)$ given in [7] has a very large hidden coefficient which makes the algorithm almost infeasible in lower dimensions.

These bounds hold for general convex sets. For convex polygons, as in our case, Ge et al. showed experimentally that up to about 40 dimensions, 10 million random points suffice to get a close to uniform discussion [4]. For our case we generate 10 million points and also test whether the mean of each coordinate converges and whether the upper and lower bounds for each coordinate are met. In detail for the mean we see that it converges after ?? steps. For the upper and lower bounds of the activation we can solve two linear

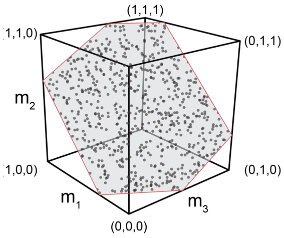
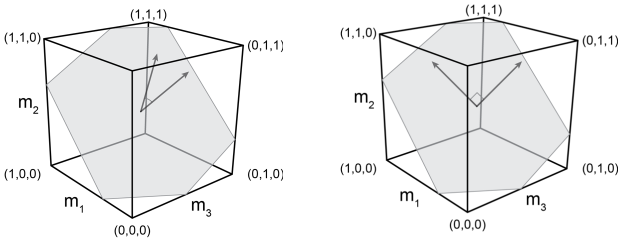


Figure 3: Uniform Distribution



(a) Some Basis (b) Direction

Figure 4: Find Orthonormal Basis

program for each coordinate of $\mathbf{a}$ to find the upper and lower bounds of each a_i. We see that those theoretical bounds match the experimentally obtained bounds.

3.3 Starting Point

To find a starting point in

$$\mathbf{f} = A\mathbf{a}, \mathbf{a} \in [0,1]^n,$$

we only need to find a feasible activation vector. For the hit and run algorithm to mix faster, we do not want the starting point to be in a vertex of the activation space. Therefore we solve the the following linear program.

$$\begin{array}{lrcl} \text{maximize} & \sum_{i=1}^{n} \epsilon_i & & \\ \text{subject to} & \mathbf{f} & = & A\mathbf{a} \\ & a_i & \in & [\epsilon_i, 1-\epsilon_i], \quad \forall i \in \{1,\ldots,n\} \\ & \epsilon_i & \geq & 0, \quad \forall i \in \{1,\ldots,n\}. \end{array} \tag{1}$$

This approach can still fail in theory, but this method has the choose $\epsilon_i > 0$ and therefore $a_i \neq 0$ or 1. Since for all vertices of the feasible activation space lie on the boundary of the n-cube, at least $n - m$ muscles must have activation 0 or 1. Documentation is included in our supplementary information.

3.4 Parallel Coordinates: Visualization of the Feasible Activation Space

Citation A common way to visualize higher dimensional data is using parallel coordinates. To show our sample set of points in the feasible activation space we draw n parallel lines, which representing the activations of the n muscles. Each point is then represented by connecting their coordinates by $n-1$ lines.

How many points do we use?

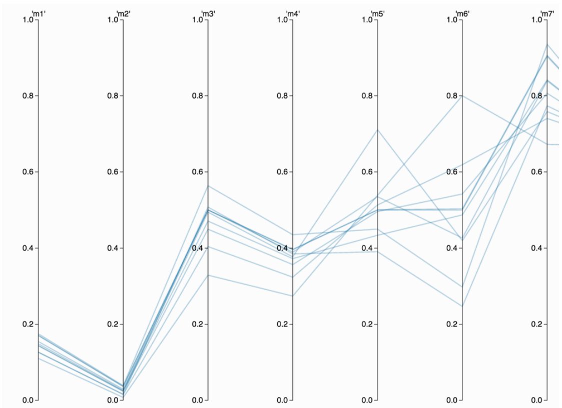


Figure 5: Feasible Activation

Using an interactive surface one can now restrict each muscle function to any desired interval, e.g., figure ??.

NICE FIGURE OF RESTRICTED PARALLEL COORDINATES

For the l_1, l_2 and l_3 norm respectively, we added an additional line to represent the corresponding weight. E.g. for a given point $\mathbf{a} \in \mathbb{R}^n$ we are interested in $\sum_{i=1}^n a_i$, $\sqrt{\sum_{i=1}^n a_i^2}$ and $\sqrt[3]{\sum_{i=1}^n a_i^3}$. As for the muscles one can restrict the intervals of the weight functions, to explore the corresponding feasible activation space.

NICE PICTURE WITH WEIGHTS INCLUDED

4 Results

Many nice figures

1. Histograms
2. Histograms 3 directions
3. PC

4.1 Activation Distribution on a Fixed Force Vector

4.2 Changing Output Force in 3 Directions

We discuss different forces into three different directions, which are given by the palmar direction (x-direction), the distal direction (y-direction) and the sum of them. The maximal forces into each direction are given by ??, ?? and ?? respectively. For $\alpha = 0.1, 0.2, \ldots, 0.9$, we give the histograms where the force is $\alpha \cdot F_{\max}$, where $F_{\max}$ is the maxium output force in the corresponding direction.

4.3 Parallel Coordinates

5 Discussion

Mostly to be written by Brian

5.1 Distributions

- Bounding box away from 0 and 1 means muscle is really needed → Already known from the bounding boxes
- High density → most solutions in that area

5.2 Parallel Coordinates

- Parallel lines in PC indicate opposite direction of muscles
- Crossing lines indicate similar direction

5.3 Running Time

The step of the algorithm which are time consuming are finding a starting point, which solves a linear program and can take exponential running time in worst case. For each fixed force vector we only have to find a starting point and an orthonormal basis once, and are hence not of concern for the running time.

Running one loop of the hit and run algorithm only needs linear time, therefore the method will extend to higher dimesions with only linear factor of additinal running time needed.

6 Acknowledgments

References

[1] M. Dyer, A. Frieze, and R. Kannan. A random polynomial time algorithm for approximating the volume of convex bodies. *Proc. of the 21st annual ACM Symposium of Theory of Computing*, pages 375–381, 1989.

[2] M. Dyer and M. Frieze. On the complexity of computing the volume of a polyhedron. *SIAM Journal on Computing*, 17:967–974, 1988.

[3] T. S. Blyth E. F. Robertson. *Basic Linear Algebra.* Springer, 2002.

[4] C. Ge, F. Ma, and J. Zhang. A fast and practical method to estimate volumes of convex polytopes. *Preprint: arXiv:1401.0120*, 2013.

[5] L.G. Khachiyan. On the complexity of computing the volume of a polytope. *Izvestia Akad. Nauk SSSR Tekhn. Kibernet*, 216217, 1988.

[6] L.G. Khachiyan. The problem of computing the volume of polytopes is np-hard. *Uspekhi Mat. Nauk 44*, 179180, 1989.

[7] L. Lovász. Hit-and-run mixes fast. *Math. Prog.*, 86:443–461, 1998.

[8] G. Marsaglia. Choosing a point from the surface of a sphere. *Ann. Math. Statist.*, 43:645–646, 1972.

[9] R.L. Smith. Efficient monte-carlo procedures for generating points uniformly distributed over bounded regions. *Operations Res.*, 32:1296–1308, 1984.