Structure of the set of feasible neural commands for complex motor tasks

Cohn BA[1], Szedlák M[2], Fukuda K[2], Gärtner B[2], and Valero-Cuevas FJ[1‡]

May 8, 2015

Abstract

Multidimensional optimal control models of neuromuscular control have shown how coordination has an infinite set of possibilities, and due to their non-exhaustive approaches (often for the sake of computational tractability), they only engage with the local structure of the activation set, rather than evaluating the entire space. While this set of feasible solutions is infinite and lies in high dimensions, it is bounded by kinematic, neuromuscular, and anatomical constraints, within which the brain must select optimal solutions. That is, the set of feasible activations is well structured. To date there is no method to describe and quantify the entire structure of high-dimensional solution spaces, other than bounding boxes or dimensionality reduction algorithms that do not capture its full structure. We present a novel approach based on the well-known Hit-and-Run algorithm in computational geometry to extract the structure of the feasible activations that produce various finger tip forces into palmar direction. It is known that explicitly computing the volume of this polytope can become too computationally complex in many instances. However, using the Hit-and-Run algorithm, we are able to uniformly sample points across the feasible activation space. We visualize this space by using histograms across each dimension, illustrating the distributions over each muscle. We use a realistic model of a static human index finger with 7 muscles, 4DOF, and 4 output dimensions, and for a progression of increasing palmar force (from 10% to 100% of maximal), we examine the distribution across each muscle's activation across all points sampled, as the feasible activation space shrinks to the maximal solution. An integration of these densities illustrates the relative number of solutions in different parts of activation space.We simulated a 40% reduction in activation ability for three muscles innervated by the deep head of the ulnar nerve, and to explore alternative constraints, we designed an interactive parallel coordinate visualization of the space. The computed distribution of activation across each

*This work was supported by NIH NIAMS R01AR050520 and R01AR052345 grants, and SNF Project 200021-150055-1.

†[1]Departments of Biomedical Engineering and Computer Science at the University of Southern California Viterbi School of Engineering, Los Angeles, CA 90089, USA brianaco@usc.edu

‡[2]Department of Computer Science, ETH Zurich, Switzerland

muscle shed light onto the structure of these solution spaces, rather than simply exploring their maximal and minimal values. Finally, we apply six non weighted and weighted static force cost functions and compare their effect upon the shape of the polytope. Although this paper presents a 7-dimensional case of the index finger, our methods extend to systems with up to at least 40 muscles. Our sampling technique for an activation space allows us to articulate the structure within which coordination Bayesian priors exist, thereby providing a foundation to the contextual neuromechanical framework which constrains learning, optimization, and adaptation of motor patterns in future research.

1 INTRODUCTION

Muscle redundancy is the term used to describe the underdetermined nature of neural control of musculature. The classical notion of muscle redundancy proposes that, faced with an infinite number of possible muscle activation patterns for a given task, the nervous system optimizes in some fashion to select one solution. Here, each of N muscles represents a dimension of control on an end effector, and at any moment of a task, a muscle activation pattern exists as a point in $[0,1]^n$,— the n-dimensional hypercube — where each muscle's maximal activation is normalized to 1 [27]. Thus researchers often seek to infer the optimization approach and the cost functions the nervous system utilizes to select effective points in activation space to produce natural behavior [4, 21, 22, 25, 6, 14].

Implicit in these optimization procedures is the notion that there exists a well structured set of feasible solutions. Thus several of us have focused on describing and understanding those high-dimensional subspaces embedded in $[0,1]^n$ [17, 18, 24, 27, 15].

For the case of static force production with a limb, the muscle redundancy problem is phrased in computational geometry: Find the structure of the set of all feasible muscle activations, given the limb mechanics and the task constraints [2, 27, 26, 15]. We aim to explore what the solution space looks like, and uncover the structure of the feasible activation space for given static force tasks. If each muscle's maximal activation is normalized to 1, the constraint $\mathbf{a} \in [0,1]^n$ describes that the feasible activation space lies in the n-dimensional unit hypercube (also called the n-cube).

1.1 High dimensionality difficulties

Consider a model of a static fingertip force, with 7 muscles arculating the index finger's 4 degrees of freedom (DoF), which will be further described in Section 2.5. Assuming independent control of each muscle (non-synergistic model), each muscle has a unique force vector at the endpoint; the end effector has 7 unique vectors it can linearly combine to generate any vector of static force. This yields a unit 7-cube in charge of producing a 4-dimensional output wrench. On order

to uncover the structure and relationship of these spaces, we cannot visualize all dimensions simultaneously.

The solution of the above system is a convex polytope is called the *feasible activation set* (see Section 2). To date, the structure of this high-dimensional polytope is inferred by its bounding box [17, 24, 15]. But the bounding box of a convex polytope excludes the details of its shape, thereby precluding comparison, since the polytope is a lower dimensional object embedded into $[0,1]^n$. Empirical dimensionality-reduction methods have also been used to calculate a basis vectors for such subspaces [5, 7, 16]. But those basis vectors only provide a description of the dimension, orientation, and aspect ratio of the polytope, but not of its boundaries or internal structure.

Here we present a novel application of the well-known Hit-and-Run algorithm [23] to describe the internal structure of these high-dimensional feasible activation sets (see Section 2.2. The input to Hit-and-Run procedure is a task force, along with the system's endpoint Jacobian, maximal tendon forces, and a moment arm matrix [26].

We applied our approach to two separate musculoskeletal models:

1. A fabricated schematic system, which we designed to have three muscles articulating one DOF, and one dimension of output force.

2. A realistic model, with seven muscles articulating four DOFs, and four dimensional output force [27].

With this, below are the key ideas and findings we present with this paper:

- The high-dimensional space serves as a Bayesian posterior probability distribution, with a nonparametric, piecewise shape.
- The bounding box exceptionally misconstrues the actual shape of the feasible activation space.
- Our approach provides a more granular context to the space within which the central nervous system optimizes.
- Hit-and-Run sampling of the solution space is computationally tractable.
- We apply six different cost functions (post-hoc) to all solutions, thereby providing spatial context to where 'optimal' solutions lie within the space.
- We designed an interactive parallel coordinates platform for visualizing and manipulating constraints to the solution space, such as muscle dysfunction, muscle hyperactivity, as well as constraining the upper and lower bounds for six different cost functions. We can compare cost functions side-by-side and view subsets of the dataset after applying cost function constraints.

2 METHODS

In the case of a tendon-driven limb with n muscles, the feasible activation space is the unit n-hypercube (as muscles can only be activated positively from 0 to a maximal normalized value of 1). As explained in [26], when task constraints are introduced to the system, the feasible activation set is further reduced; in this context, a task is a static force vector produced at the endpoint of the limb, which is represented as a set of inequality and equality constraints. Thus if this simple limb meets all constraints, the feasible activation set is given by the polytope P containing all $a \in \mathbb{R}^n$, that satisfy

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

where $\mathbf{f} \in \mathbb{R}^m$ is a fixed output force vector and $A = J^{-T}RF_o \in \mathbb{R}^{m\times n}$— where J, R and F_0 are the matrices of the Jacobian of the limb, the moment arms of the tendons, and the strengths of the muscles, respectively [27, 26]. P is bounded by the unit n-cube since all variables a_i, $i \in [n]$ are in the interval $[0,1]$. Each constraint of $\mathbf{f} = A\mathbf{a}$, is a hyperplane in $n-1$ dimensions. If $\mathbf{f}$ is a is a feasible submaximal output force and there are no linear depenencies on the constraints, the feasible activation set is a $(n-m)$-dimensional space embedded into the n-dimensional unit cube. Note that in our applications, it is safe to assume that no linear dependencies exist. Consider the following 1×3 fabricated example, where the task is a 1N unidimensional force. The set of feasible activations is given by the shaded set in Figure 2. Since there are three muscles and one constraint the output space is of dimension $3-1=2$.

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

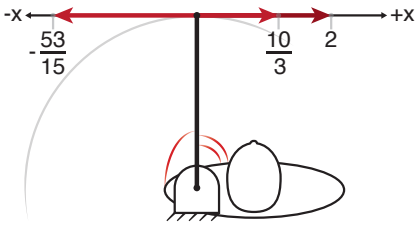

Figure 1: One imagined visualization of the fabricated tendon driven system, with 3 generators.

2.1 Difficulties of Volume Computation in Higher Dimension

Exact volume computations for polytopes is known to be #P-hard [9]. Several algorithms have been surveyed and implemented, but can only handle up to 10 dimen-

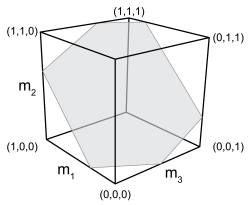

Figure 2: The feasible activation set for a three-muscle system meeting one functional constraint is a polygon in $\mathbb{R}^3$.

sions [3]. Recent muscle system models we have used have been 31 dimensional [15], and other muscle models have over 40 muscles involved [1, 18, 13, 8], thereby limiting the feasibility of using direct volume computations. Instead, we chose to uniformly sample the continuous space, effectively approximating the shape of the polytope by calculating point densities.

2.2 Hit-and-Run algorithm

We chose to sample the activation space with the Hit-and-Run method that is known to converge to the uniform distribution across any convex body K [23]. It is a generalization of the discrete Markov chains and recursively samples a sequence of points in K as described below. The mixing time is known to be $\mathscr{O}^*(n^2R^2/r^2)$, where r and R are the radii of the inscribed and circumscribed ball of K respectively [9, 19]. I.e., after $\mathscr{O}^*(n^2R^2/r^2)$ steps, the Hit-and-Run algorithm has sampled a point uniformly at random (u.a.r.) in K. Unfortunately the hidden constant is large, which makes the problem practically almost infeasible. However experimental results suggest that a number of points linear w.r.t. to the dimension suffices; this will be discussed in Section 2.4. As the feasible activation space of the muscles are given by a convex polytope, this method can be directly applied for our problem. Notably, there are other methods which sample uniformly, such as the Grid Walk or Ball walk [28]. We chose the Hit-and Run because of its easy structure and mixing guarantee, however it would be interesting to compare with other methods.

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

1. Find a starting point $\mathbf{p}$ of P.
2. Generate a random direction from $\mathbf{p}$ in P (uniformly at random over all directions) (Figure 3a).
3. Find the intersection points of the random direction with the boundary of the polytope (Figure 3b).

4. Choose a point u.a.r. on this line segment given by the intersection points (Figure 3c).

5. Repeat from 1. the above steps with the new point as the starting point .

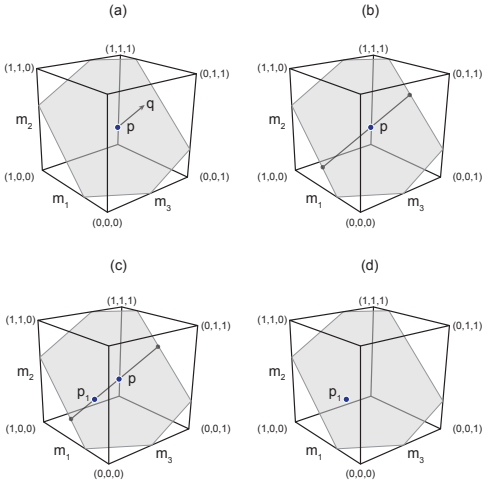

Figure 3: Graphical description of the Hit-and-Run algorithm.

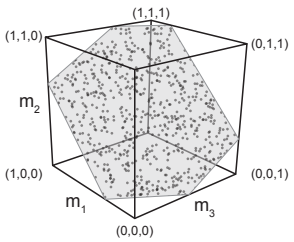

Figure 4: Uniform distribution across the feasible activation space. In the schematic arm example, the distribution is represented within a 2D plane.

2.3 Implementation of Hit-and-Run

To find a starting point in P the polytope given by

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

we only need to find a feasible activation vector. For the Hit-and-Run algorithm to mix faster we want the starting point not close to a vertex of the polytope [19]. We use the following standard trick with slack variables ε_i, which for applications often gives a good starting point.

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

The rest of the implementation of the Hit-and-Run algorithm is straight forward except for the choice of the random direction. How do we sample u.a.r. from all directions in P? Suppose that $\mathbf{q}$ is a direction in P and $\mathbf{p} \in P$. 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

$$\mathbf{f} = A(\mathbf{p}+\mathbf{q}) = \mathbf{f} + A\mathbf{q} \Rightarrow A\mathbf{q} = 0.$$

and hence 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 [20].

1. Find an orthonormal basis $b_1, \ldots, b_r \in \mathbb{R}^n$ of $A\mathbf{q} = 0$.
2. Choose $(\lambda_1, \ldots, \lambda_r) \in \mathcal{N}(0,1)^r$ (from the Gaussian distribution).
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 $\{q | A\mathbf{q} = 0\}$ and orthogonalize with the well known Gram-Schmidt method (for details see e.g. [10]). Note that in order to get the desired u.a.r. sample 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 $n - m = r$.

2.4 Mixing Time

From a given starting point, how many iterations of the Hit-and-Run method are necessary to reach a u.a.r. point? For convex polygons in higher dimensions up to 40, experimental results suggest that $\mathcal{O}(n)$ steps of the Hit-and-Run algorithm are sufficient. In particular Emiris and Fisikopoulos paper suggest that $(10 + \frac{10}{n})n$ steps are enough to converge upon the uniform distribution [11], while in Ge et al.'s paper every point of the Hit-and-Run algorithm is used in the sample [12].

2.5 Realistic index finger model

We used our published model in [27] to find matrix $A \in \mathbb{R}^{4\times 7}$, where $\mathbf{a} \in \mathbb{R}^7$; the four degrees of freedom were ad-abduction, flexion-extension at the metacarpophalangeal joint, and flexion-extension at the proximal and distal interphalangeal

joints. The force direction we simulated are visible in Figure 5. In this model, for each input we collected 1.000.000 points and sampled every 100th point. In addition to the Hit-and-Run algorithm we computed the theoretical maximum and minimum activation for each muscle for the given force; the theoretical and observed bounds for all muscles are close, such that they are superimposed **??**.

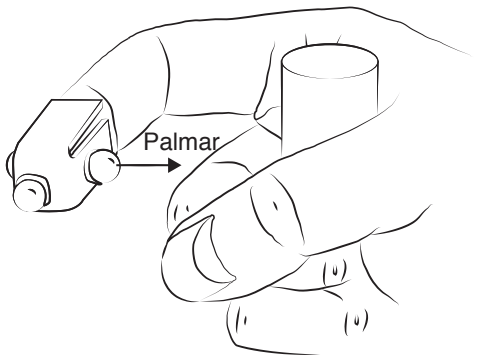

Figure 5: The index finger model simulated force production in the palmar direction. Adapted from [27].

2.6 Solution projection histograms

Figure 6 shows the activation distribution of each muscle, when activated with 50% of the maximal activation force. We also show the observed activation upper and lower bounds for each muscle (vertical dotted lines). In Section 3.2 we consider the distributions for different forces into palmar direction.

2.7 Parallel coordinates visualization

A common way to visualize higher dimensional data is using parallel coordinates[briantodo citations]. To show our sample set of points in the feasible activation space we draw n parallel lines for each of the n muscles. With the axis labels of the line set between 0 and 1, each point is then represented by connecting their coordinates by $n-1$ lines. Using an interactive surface we restrict each muscle function to any desired interval- see Figures 8 and 9. We decided to simulate a 50% reduction in activation (feasible tendon force production) in three of the muscles innervated by the deep branch of the ulnar nerve- PI, DI, and FDP.

2.8 Muscle-metabolic and neural drive cost functions

For every solution collected, we used popularly-used cost functions: we computed activation l_1, l_2 and l_3 norms, and the tendon-force l_1^w, l_2^w and l_3^w norms.

Six additional vertical lines were added to the parallel coordinates plot to represent each cost function. With the same parallel coordinates framework as de-

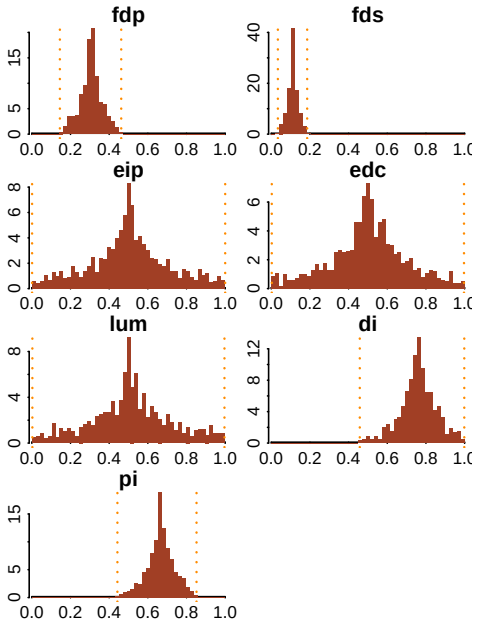

Figure 6: Distribution of feasible activations for 50% of the computed maximal force output in the palmar direction.

veloped with muscle activation, we can restrict and subset solutions which fall into desired cost-function ranges, thereby masking sub-optimal solutions and highlighting only those meeting the criteria. For a given point $\mathbf{a} \in \mathbb{R}^n$ we are interested in the associated cost of every solution collected through Hit-and-Run. We developed and tested our code in Ubuntu 14.04, Windows 8.1, and OSX Yosemite, using Scala 2.11.6 [briantodo cite] for our implementation of Hit-and-Run, R 3.1.3 [briantodo cite] for histograms and plots, and using Sygmatic Parcoord[briantodo cite] and d3.js[briantodo cite] for our interactive parallel coordinate visualization. All code used to develop this publication is readily available at https://github.com/bcohn12/space.

3 RESULTS

3.1 Density projection upon one dimension

Using Hit-and-Run to sample feasible activation sets, Figure 6 shows the distributions of activation solutions for a palmar submaximal force resulting from [briantodo number] solutions computed with Hit-and-Run sampling. This is the first time (to our knowledge) that the internal structure of the feasible activation set has been visualized for a sub-maximal force. Notice also that the lower and upper bounds of the activations (i.e., the dashed lines that indicate their bounding box), are uniquely uninformative of the actual density of distribution of feasible activa-

Name	Cost function
l_1	$\sum_{i=1}^{n} a_i$
l_2	$\sqrt{\sum_{i=1}^{n} a_i^2}$
l_3	$\sqrt[3]{\sum_{i=1}^{n} a_i^3}$
l_1^w	$\sum_{i=1}^{n} a_i F_{0i}$
l_2^w	$\sqrt{\sum_{i=1}^{n} (a_i F_{0i})^2}$
l_3^w	$\sqrt[3]{\sum_{i=1}^{n} (a_i F_{0i})^3}$

Table 1: Cost functions and their usage, where a_i and F_{0i} represent a muscle's activation in a given solution and that muscles MIC [briantodo: what is MIC?], respectively.

tions. Note also that the activation needed for the maximal force output (thick gray line) is very often not the mode of the activations at [briantodo select correct number]50%[maytodo: can we set this as a variable and use throughout?] of output.

Talk about the function of each muscle, with respect to the moment arm matrix and the relevant cell of the A matrix.

The density integrals perpendicular to each muscle are provably unimodal due to the convexity [maytodo cite or add supporting evidence], and therefore it would be inadvisable to fit a normal distribution as the probability density function.

3.2 Activation spaces for increasing force

For maximal force into any direction, there is a unique activation vector satisfying all constraint. The maximal force into palmar direction is given by ????? and its unique activation vector ????. [briantodo insert values] In Figure 7 we give the distributions of the activations for increasing force, starting with 10% of the maximal force, increasing in 10% steps until maximal force is reached.

The solution polytope converges as the difficulty of the task increases; the rate of convergence is different across muscles. Whereas for example fds already has a small range of feasible activations at 10%, eip has feasible activation $[0,1]$ up to 80%. For some muscles (such as X and Y) the convergence only occurs in the last [briantodo insert maximal feasible palmar force * 0.90], while others converge earlier-in lower forces of maximal (X and Y are examples of this)[briantodo fill out these descriptive statistics].

It is imperative to keep in mind that every histogram (regardless of its convergence) is composed of the distribution of all 10,000 points; when the distribution is compressed, the relative percentage of the bars will increase, as we fixed break width (Δx) remain constant to 2% of maximal contraction. [briantodo cite http://library.msri.org/books/Book31/files/ball.pdf]

The peaks seen in these figures is the perpendicular slice that has the largest

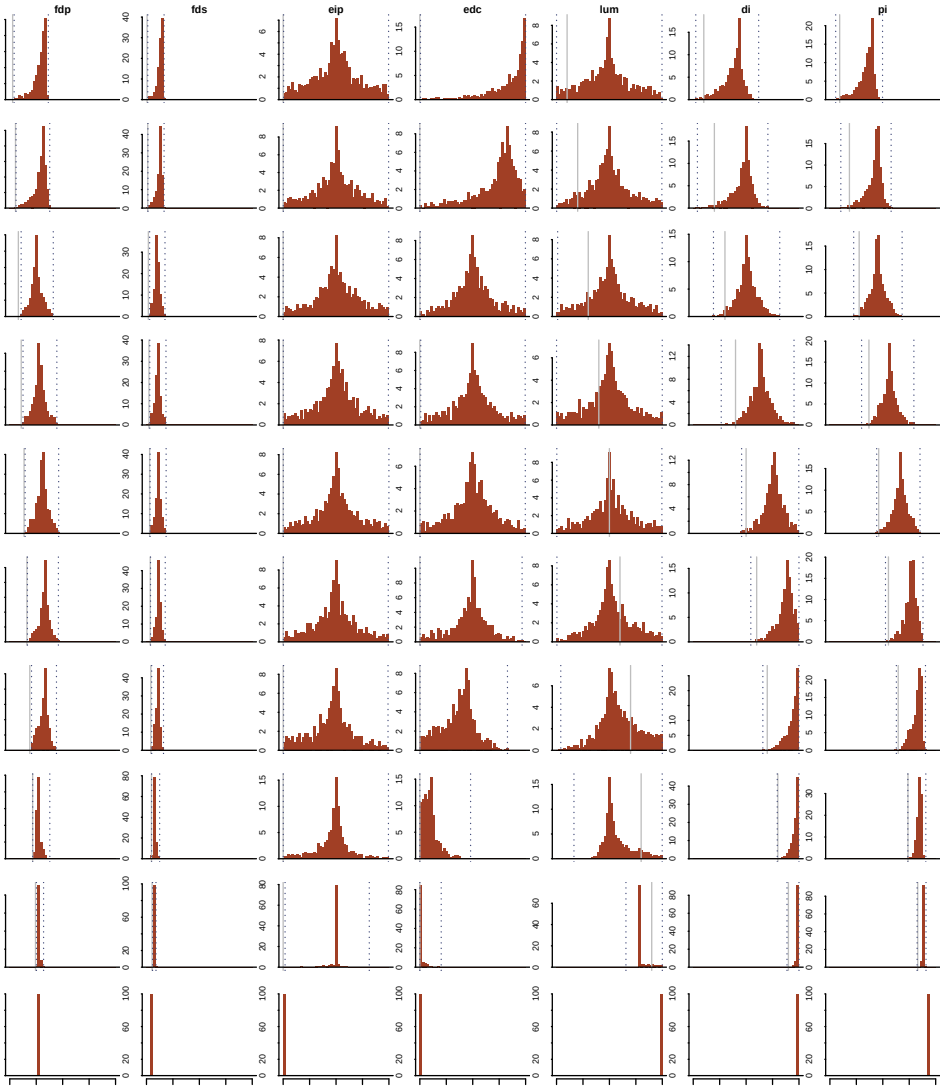

Figure 7: Distribution of activations in the palmar direction and changing force. Each row of histograms uses a Hit-and-Run set. The height of each bar visualizes the percentage of 10,000 solutions found within a given 0.02 span of activation; the shape is more meaningful than the magnitude of the y-axis, as we expect convergence (and therefore peak increase within few bins) towards maximal contractions. We computed the upper and lower bounds of activation for each muscle (vertical dotted lines).

relative area; within the same muscle it does not have to be symmetric between the bounds, and can shift over differing tasks. As expected, the unique solution at 100% activation appears as a single peak representing 100% of the sampled points; the bounds and the muscle's unique activation are superimposed. We explain why these distributions must be unimodal, non-normal, more formally in the discussion, along with information about the structure of the probability density function across a given muscle dimension. Consider the activation distributions between 70%and 80% for lum, where the median changed by less than 4% while the lower bound increased by nearly 13%.

First, we observe that a meaningful cross-muscle comparison of point distributions cannot be ascertained by the bounding box. For example, at a task of 10% of maximal palmar force production, epi and edc both have lower and upper bounds of 0, and 1, respectively, yet their distributions are thoroughly distinct; the shape of eip is more symmetric (lower 25%, median, upper 75%: 0.36, 0.5029, and 0.62), while 75% of the solutions sampled have edc higher than 0.74 7.

Second, we find this holds not only for inter-muscle distribution comparisons, but intra-muscular distributions. Consider the significant change in the shape of the distributions across the progression for edc until the 60% task; the lower and upper bounds change less than 1% and 4%, respectively, while the median shifted by nearly 40%. In the most extreme case, the median activation can be exceptionally narrow, while the bounds are wide- for example, eip at a 90% task; although all solutions are bounded between 1% and 81%, the space between 0.49 and 0.51 house approximately 79% of the solutions.

Next, we see that if one muscle had to be fixed throughout the entire force progression, di and pi would fail; their bounding boxes of tasks below 40% do not include the unique solution at 100%. We also placed a vertical grey line for the scaled unique solution at maximal, denoted $\mathbf{a}^*$, (e.g. lum converges to an activation of 1 at maximal palmar force, so we put a grey line at 0.8 for 80% of maximal palmar force.) Since $\alpha f_{\max} = \alpha A\mathbf{a}^*$, $\alpha\mathbf{a}^*$ is a solution of the feasible activation set at α-fraction of the maximal force. However we observe that for some muscles, these points can lie arbirtraliy in the distribution i.e. do not have to lie close to the corresponding peaks (e.g. musle di and eip).

3.3 Parallel Coordinates Visualization

To maintain realtime interactivity of the plot, we used only the first 1000 points collected for each task, from $\alpha = 0.1$ to $\alpha = 1\%$ (maximal force).

On the histograms you can predict how many solutions would be lost if you put one constraint on one muscle. Parallel coordinates allow us to visualize not only the amount of solutions lost, but the location of the remaining solutions across the other muscle dimensions. Furthermore one can constraint any number of muscles and still visualize the remaining amount of solutions, whereas in the histograms one can not make any predictions.

Figure 8 is a modified screen-shot of the platform, constrain activation for a

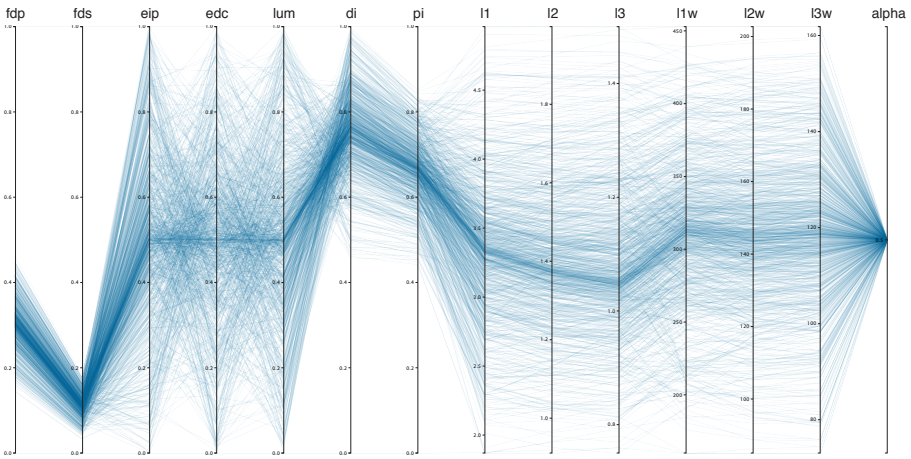

Figure 8: This figure is a snapshot of the interactive platform for visualizing all solutions. This parallel coordinates plot visualizes, where each feasible activation set is strewn across each dimension's axis as a line. While α could be set to view all tasks, we set alpha to a palmar force of 50% of the computed maximal feasible force in this direction. Here we show 1000 points accrued from Hit-and-Run on a task of $\alpha = 0.5$.

given muscle or cost function from above, below, or both. The resulting number of lines represents how the solution space changed across all dimensions recorded. Note how α is also a constraint- we can explore each level of force output, or multiple at a time within a range of magnitude in the palmar direction.

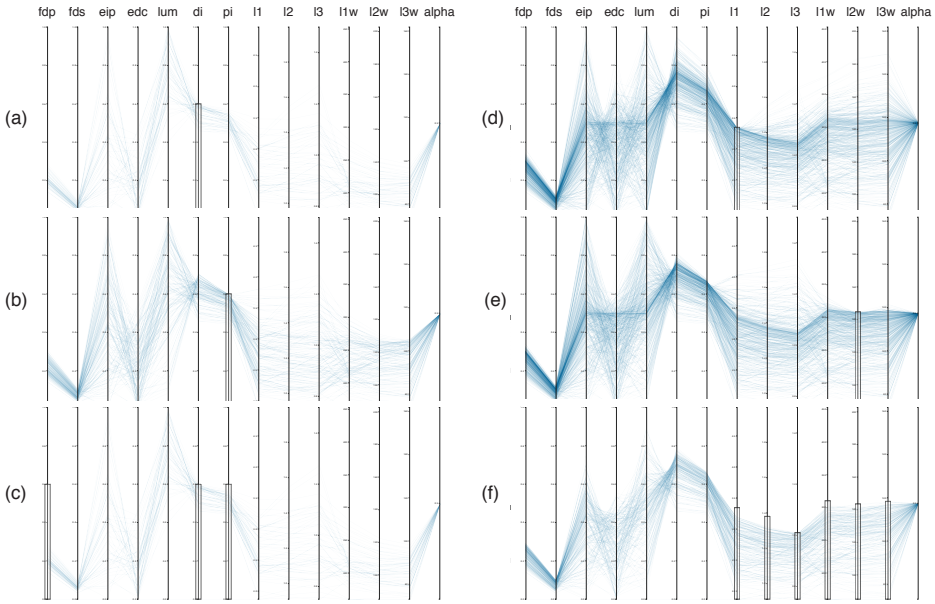

Figure 9: These snapshots show the use of the interactive parallel coordinate visualization of solutions across the activation space. For the task set to 50% of maximal in the palmar direction, we show the (a) remaining 166 solutions when pi $<$ 60% (b) remaining 57 solutions when di $<$ 60%, and the (c) remaining 57 solutions when we constrain pi $<$ 60% and di $<$ 60%. We also show the (d) remaining 502 solutions when we select the lower 50% of l_1 costs, (e) remaining 498 solutions when we select the lower 50% of l_2^w costs, and the (f) remaining 220 solutions when we select the lower 50% across all cost functions in Table 1

Constraining pi $<$ 60% still leaves a solution space about three times as large as di $<$ 60%. Observe that constraining di $<$ 60% already implies that pi $<$ 60% as seen in Figure 9 (b), (c). Constraining one muscles does have different effects on the other muscles as for example if ei is constrained below 60% activation, we see that the bounding box of pi is significantly restricted; however, the same constraint has no effect on the bounding box of eip.

discuss what happens when you bring each of those muscles down, using the R produced stats. talk about how when you add X as a constraint, most of the solutions are distributed across the other muscles between X and X. Say which ones go up, which go down- which ones become clustered and which ones lose their peak/spike.

As there are only few not steep crossings between l_1, l_2 and l_3, the parallel

coordinates suggest correlation of those three weights. Similar correlation is observed for the respective weighted functions. We provide a scatter plot for their relationships in Appendix Figures 10 and 11.

4 DISCUSSION

4.1 Level of trust of the polytope approximation

Our approximations show sufficiently accurate views of the polytope in slices perpendicular to each axis. Had we performed exact volume computations, we would have had more accurate relative volumes; that said, the level of error generated through approximation is exceptionally small in comparison to error derived from measuring/predicting the musculoskeletal parameters to define the force generators A. Our code only solves one linear equation to find the starting point, and the time-cost of each point thereafter is linear; therefore this method can be used for tendon driven models in very high-dimensional systems.

Sohn et. al. have identified how the bounding box can illustrate the bounds of feasible activation; essentially reducing the effective number of solutions a neuromuscular system can select from [24]. Our research here looks to further constrain this space by observing the cost-agnostic distribution of solutions across the set of all feasible coordination patterns.

4.2 Activation Progression

The only way that the bounding box would contain the entire set of information regarding a given muscle's activation distribution would be if that end effector is completely unaffected by any activation of that muscle i.e. the muscle's linear endpoint force is the 0 vector. As such, muscles which are nearly uninvolved in the end effector's actions will form near-uniform distributions, as their involvement does not affect the activation space.

4.3 Cost distributions

In further studies one could put activation constraints directly into the A, instead of the unit interval bounds. As long as the resulting dataset is large enough for our purpose, there is no advantage to this, as we can not compare with the original feasible activation space. Since l_1 and l_1^w are linear, one can also put corresponding constraints into A. Our implementation does not support directly inputting bounds on the higher degree weight functions l_2, l_3, l_2^w and l_3^w.

We note that the activation and metabolic classes of cost function are fundamentally different, and do not explore correlations between these two classes. We do, however, note that when all of the involved cost functions are 'minimized' to the bottom half of all solution costs, the union maintains a very high number of

solutions (22%). With this we can note how all of these cost functions are similar in nature across the polytope (as expected).

4.4 Concluding remarks

Our results clearly show:

- The Hit-and-Run algorithm can explore the feasible activation space for a realistic 7-muscle finger in a way that is computationally tractable.
- We find that the bounding box exceptionally misconstrues the internal structure of the feasible activation set.
- The Hit-and-Run algorithm is cost-agnostic in the sense that no cost function is needed to predict the distribution of muscle activation patterns. Therefore, we can provide spatial context to where 'optimal' solutions lie within the solution space; this approach can be used to explore the consequences of different cost functions.
- The distribution of muscle activations often show and strong modes that will critically affect the learning of motor tasks.

In comparison to traditional bounding-box representations, our application of Hit-and-Run in this context is decisively superior in capability for meaningful visualization, value in extracting associations between solutions, and computational tractability, in addition to being veritable of the true solution distributions within the feasible activation set. Our bodies exist within a feasible activation space, and once we enter this space then optimization is possible. In this way, we can think of the solutions space as an effective model for exporation-exploitation. This can help us in comparing the structure of the activation space as a set of high-dimensional Bayesian priors which are narrowed/shifted over time to compensate for learning and skill-development. Essentially, once you enter into task-independent variation, it becomes a question of identifying the region of the activation space which both satisfies the spatiotemporal constraints, but also approaches optimality under efficiency/speed demands. We want to 'close the loop' between the nervous system commands, and the mechanical output, thereby uncovering how the CNS collaborates with Newtonian physics to select neural commands which effectively coordinate multi-link limbs, so we can act, play, and dance in the real world. Mechanical demands constrain the total space of musculoskeletal coordination options, thus, motile organisms first 'explore' coordination strategies conducive to the desired movement, and recursively redefine the more optimal subspaces. Once a desired task is mapped to an effective coordination strategy (as in, it gets the job done), then training and experience (exploration-exploitation) can aid in finding the best coordination. As many tasks are similar (i.e. they require the similar force generation or torque production over the course of a movement), the activation patterns for similar actions must be similar as well. Optimal coordination strategies for one

task may be near-optimal for a similar task. On the other hand, a suboptimal coordination strategy that achieves one task, may be furiously off-target for a very similar task.

References

[1] Edith M Arnold, Samuel R Ward, Richard L Lieber, and Scott L Delp. A model of the lower limb for analysis of human movement. *Annals of biomedical engineering*, 38(2):269–279, 2010.

[2] D. Avis and K. Fukuda. A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. *Discrete & Computational Geometry*, 8(3):295–313, 1992.

[3] B. Büeler, A. Enge, and K. Fukuda. Exact volume computation for polytopes: A practical study. *Polytopes: Combinatorics and Computation*, 29:131–154, 2000.

[4] E Y Chao and K N An. Graphical interpretation of the solution to the redundant problem in biomechanics. *Journal of Biomechanical Engineering*, 100:159–67, 1978.

[5] R. H. Clewley, J. M. Guckenheimer, and F. J. Valero-Cuevas. Estimating effective degrees of freedom in motor systems. *IEEE Trans Biomed Eng*, 55:430–442, Feb 2008.

[6] R.D. Crowninshield and R.A. Brand. A physiologically based criterion of muscle force prediction in locomotion. *Journal of Biomechanics*, 14(11):793–801, 1981.

[7] Andrea d'Avella and Emilio Bizzi. Shared and specific muscle synergies in natural motor behaviors. *Proceedings of the National Academy of Sciences of the United States of America*, 102(8):3076–3081, 2005.

[8] Vincent De Sapio, Darren Earl, Rush Green, and Katherine Saul. Human factors simulation using demographically tuned biomechanical models. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, volume 58, pages 944–948. SAGE Publications, 2014.

[9] 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.

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

[11] Ioannis Z Emiris and Vissarion Fisikopoulos. Efficient random-walk methods for approximating polytope volume. *arXiv preprint arXiv:1312.2873*, 2013.

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

[13] Samuel R Hamner, Ajay Seth, and Scott L Delp. Muscle contributions to propulsion and support during running. *Journal of biomechanics*, 43(14):2709–2716, 2010.

[14] JS Higginson, RR Neptune, and FC Anderson. Simulated parallel annealing within a neighborhood for optimization of biomechanical systems. *Journal of biomechanics*, 38(9):1938–1942, 2005.

[15] Valero-Cuevas F. J., Cohn B. A., Yngvason H. F., and Lawrence E. L. Exploring the high-dimensional structure of muscle redundancy via subject-specific and generic musculoskeletal models. *J Biomech*, In press, 2015.

[16] Vijaya Krishnamoorthy, Simon Goodman, Vladimir Zatsiorsky, and Mark L Latash. Muscle synergies during shifts of the center of pressure by standing persons: identification of muscle modes. *Biological cybernetics*, 89(2):152–161, 2003.

[17] J. J. Kutch and F. J. Valero-Cuevas. Muscle redundancy does not imply robustness to muscle dysfunction. *Journal of Biomechanics*, 44(7):1264–1270, 2011.

[18] Jason J Kutch and Francisco J Valero-Cuevas. Challenges and new approaches to proving the existence of muscle synergies of neural origin. *PLoS computational biology*, 8(5):e1002434, 2012.

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

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

[21] B. I. Prilutsky. Muscle coordination: the discussion continues. *Motor Control*, 4(1):97–116, 2000. 0 1087-1640 Journal article.

[22] Stephen H Scott. Optimal feedback control and the neural basis of volitional motor control. *Nature Reviews Neuroscience*, 5(7):532–546, 2004.

[23] Robert L Smith. Efficient monte carlo procedures for generating points uniformly distributed over bounded regions. *Operations Research*, 32(6):1296–1308, 1984.

[24] M Hongchul Sohn, J Lucas McKay, and Lena H Ting. Defining feasible bounds on muscle activation in a redundant biomechanical task: practical implications of redundancy. *Journal of biomechanics*, 46(7):1363–1368, 2013.

[25] Emanuel Todorov and Michael I Jordan. Optimal feedback control as a theory of motor coordination. *Nature neuroscience*, 5(11):1226–1235, 2002.

[26] F. J. Valero-Cuevas. A mathematical approach to the mechanical capabilities of limbs and fingers. *Adv. Exp. Med. Biol.*, 629:619–633, 2009.

[27] F. J. Valero-Cuevas, F. E. Zajac, and C. G. Burgar. Large index-fingertip forces are produced by subject-independent patterns of muscle excitation. *J Biomech*, 31:693–703, Aug 1998.

[28] S. Vempala. Geometric random walks: A survey. *Combinatorial and Computational Geometry, MSRI Publications*, 52:573–612, 2005.

5 APPENDIX

5.1 Finger model data

$F_0 = (123.0, 219.0, 23.52, 91.74, 21.6, 124.8, 129.6)$

$$JR = \begin{pmatrix} -0.08941 & -0.0447 & -0.009249 & 0.03669 & 0.1421 & 0.2087 & -0.2138 \\ -0.04689 & -0.1496 & 0.052 & 0.052 & 0.0248 & 0.0 & 0.0248 \\ 0.06472 & 0.001953 & -0.1518 & -0.1518 & 0.2919 & 0.0568 & 0.2067 \\ 0.003081 & -0.002352 & -0.0001649 & -0.0001649 & -0.0004483 & 0.0001578 & -0.000685 \end{pmatrix}$$

$task_x = (1.0, 0.0, 0.0, 0.0)$

$task_y = (0.0, 1.0, 0.0, 0.0)$

Palmar force is $task_z = (0.0, 0.0, 1.0, 0.0)$

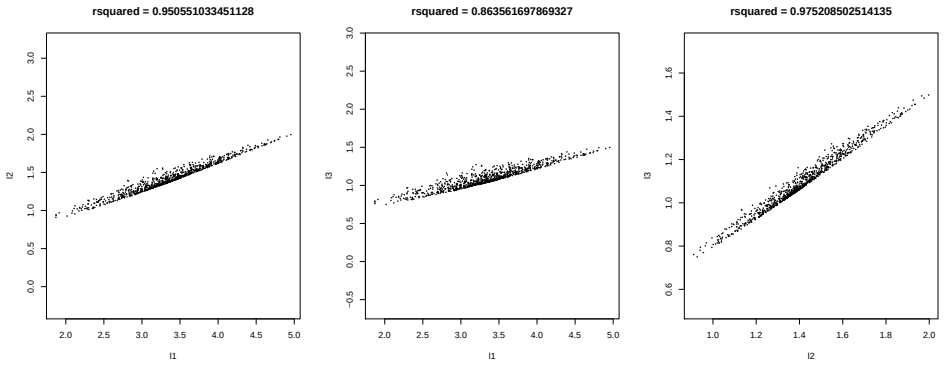

Figure 10: Non weighted cost functions

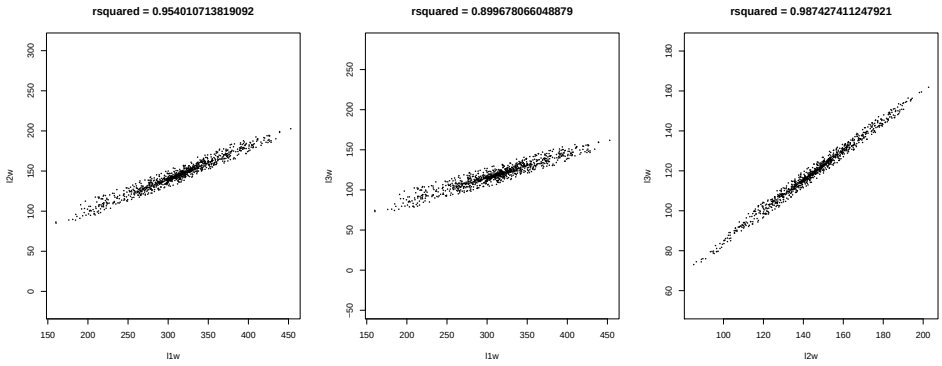

Figure 11: Weighted cost functions