Feasible Sets Analysis of Musculoskeletal Systems

An Oral Defense Presented for the Degree of Doctor of Philosophy

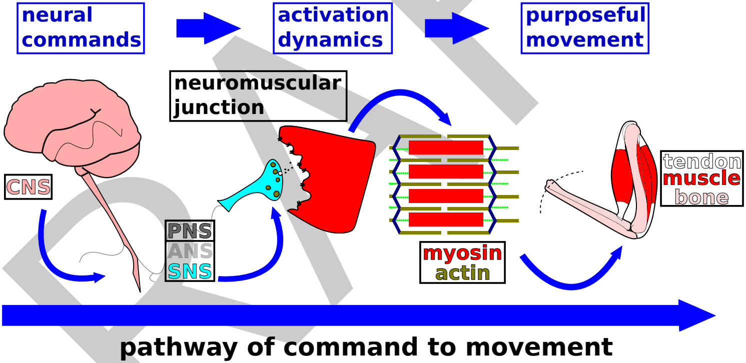
Aravind Sundararajan

University of Tennessee

July 23, 2020

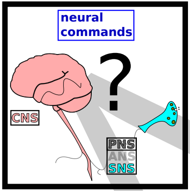
flex! how did you do it?

control of biological systems involves many components

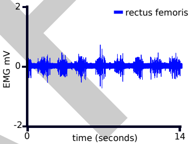


but what is the brain doing?

neural control is a black box



→



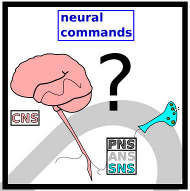
our understanding is still in **infancy** ...

- brain-machine interfaces
- prosthetics

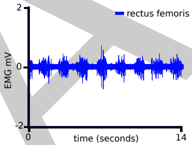
... and our tools are **less than ideal**

- Surface Electromyography (sEMG)
- dynamometry
- computed control

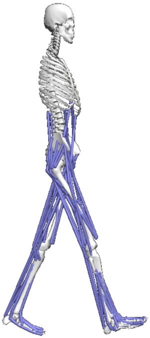
So how do we move in the world?



→

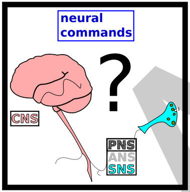


→



how the brain and body work together to produce purposeful movement is not yet fully understood

there are many conflicting theories about control



- **muscle synergies** - brain organizes muscles in groups
- **minimization** - brain optimizes some value
- **task prioritization** - brain decomposes complex behaviors into tasks

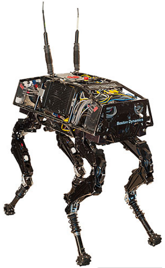
biomechanists conventionally look for **optimized** or **minimized** solutions...

OpenSim modeling software used in this dissertation even has **two** algorithms that optimize controls (CMC, SO)

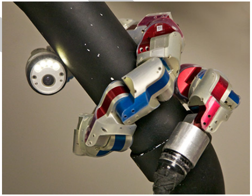
but why?

biomechanists have to borrow tools from roboticists!

optimal controls are **desirable** for roboticists and biomechanists borrow their algorithms for **computed control** ...



BigDog (Boston Dynamics)



modular snake robot (Carnegie Mellon)

... but biological systems aren't robots [1]

[1] https://www.bostondynamics.com/legacy, http://biorobotics.ri.cmu.edu/

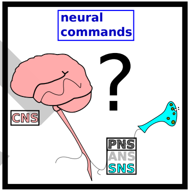
my contribution

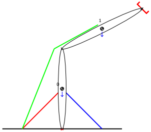
biomechanists need **specialized tools** to investigate the control of biological systems without assuming optimization of the commands according to arbitrary objectives

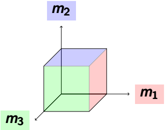
the **objective** of this dissertation was to design tools that explore the solution space where control can happen

this dissertation is a **unifying platform** that other analyses of control can be layered

this dissertation serves as a **vehicle** for machine learning in musculoskeletal modeling





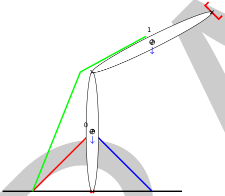


Let's probe the possibilities!

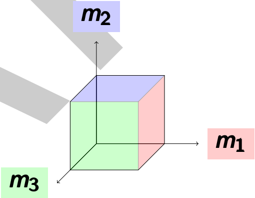
a "feasible set" is that space of possibilities

feasible set is also known as **feasible region**, **search space**, or **solution space**

Different representations:



$$0 \leq m_1 \leq 1$$
$$0 \leq m_2 \leq 1$$
$$0 \leq m_3 \leq 1$$



$\mathcal{H}$ is halfspace representation

$$\mathcal{H} = [b| - A] \geq 0 = \begin{array}{cccc} b & m_1 & m_2 & m_3 \end{array} \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 1 & 0 & 0 & -1 \end{bmatrix}$$

$\mathcal{V}$ is vertex representation

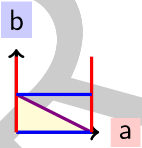
$$\mathcal{V} = \begin{array}{ccc} m_1 & m_2 & m_3 \end{array} \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{bmatrix}$$

how to find feasible space

vertex enumeration is the process of finding the map from $\mathcal{H} \to \mathcal{V}$

\$1 apples a and \$2 bananas b with \$10 in my pocket:

$$0 \leq 1a \leq 10$$
$$0 \leq 2b \leq 10$$
$$1a + 2b \leq 10$$

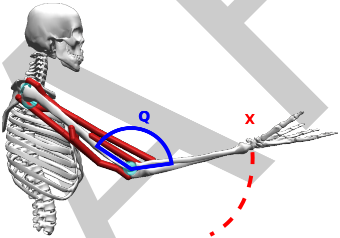


$$\text{Vertices: } (0,0), (10,0), (0,5)$$

feasible activation space (FAS) is the set of all possible muscle activations that satisfy some constraint

how do we apply feasible sets analysis to multibody systems?

first, let's review dynamics



we can think about the motion of systems in terms of the locations of end effectors in $\mathbb{O}$ or configurations of joints in $\mathbb{C}$

equations in terms configuration space		equations in terms operational space
$\mathcal{Q} \in \mathbb{C}$	$\longrightarrow$	$\mathcal{X} \in \mathbb{O}$

dynamics crash course

configuration space : $\mathbb{C}$

- generalized coordinate: $\mathcal{Q}$
- generalized force: $\boldsymbol{\Gamma}$
- mass matrix: $\boldsymbol{M}$
- centrifugal + Coriolis: $\boldsymbol{C}$
- Gravity: $\boldsymbol{G}$
- $M(\mathcal{Q})\ddot{\mathcal{Q}} + C(\mathcal{Q}, \dot{\mathcal{Q}})\dot{\mathcal{Q}} + G(\mathcal{Q}) = \boldsymbol{\Gamma}$

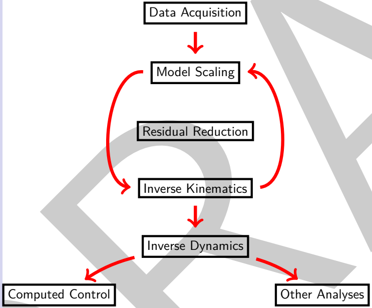
operational space : $\mathbb{O}$

- position in space: $\boldsymbol{\mathcal{X}}$
- force: $\boldsymbol{F}$
- kinetic energy: $\boldsymbol{\Lambda}$
- centrifugal + Coriolis: $\boldsymbol{\mu}$
- Gravity: $\boldsymbol{p}$
- $\Lambda(\boldsymbol{\mathcal{X}})\ddot{\boldsymbol{\mathcal{X}}} + \mu(\boldsymbol{\mathcal{X}}, \dot{\boldsymbol{\mathcal{X}}}) + p(\boldsymbol{\mathcal{X}}) + \boldsymbol{J_{ext}F_{ext}} = \boldsymbol{F}$

$\boldsymbol{J}$ is a jacobian ($\dot{\boldsymbol{\mathcal{X}}} = J(\mathcal{Q})\dot{\mathcal{Q}}$)
$\boldsymbol{J_{ext}}$ is the jacobian to the applied external forces
$(\boldsymbol{M} = \boldsymbol{J^T \Lambda J})$[2]

[2] Khatib 1987, IEEE Journal of Robotics and Automation

Biomechanical Modeling involves finding Γ from experimental data [a]



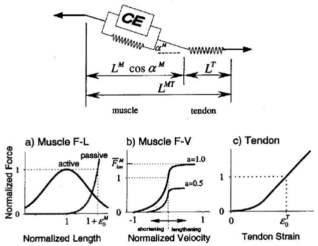
- we use recorded **motion capture** and **force plate** data to make subject specific physics-based models
- **computed control** involves finding the muscle activations that contribute to Γ (ID torques)
- how complex does the **muscle model** have to be?

[a] https://news.cision.com/vicon

isolate away the dynamic contributions, how does the muscle physiology influence force?

assume statics ($\sum F = 0, \sum M = 0$) over each discrete time of the dynamic task and iteratively pose the model

what muscle parameters do we have to consider? ($\mathbf{F_0}$,$\mathbf{l_m}$,$\mathbf{v_m}$)



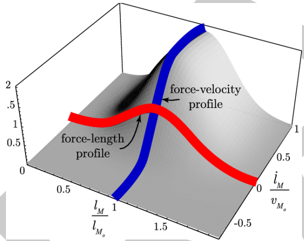
Muscle force:

$$F_m = \mathbf{F_0}(\mathbf{a} f^L(\mathbf{l_m}) f^V(\mathbf{v_m}) + f^{PE}(\mathbf{l_m})) \cos \alpha$$

F_0 peak isometric force, l_m muscle fiber length, v_m fiber velocity, a activation, α pennation angle. F^L curve, F^V curve, F^{PE} curve [3]

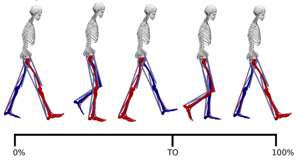
[3] Thelen DG (2003) Adjustment of muscle mechanics model parameters to simulate dynamic contractions in older adults

muscles operate on a F-L-V surface



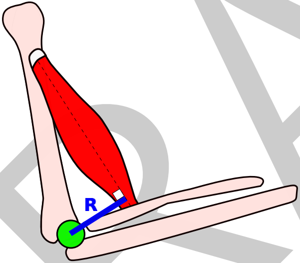
4

how do muscle fiber length and muscle fiber velocity effects influence the force generating capacity during gait? How do they work together?



[4] De Sapio (2008) Least action principles and their application to constrained and task-level problems in robotics and biomechanics

muscles induce moments about joints



this is the shortest euclidean distance to the joint, but can involve complicated routing

muscle induced moments:

$$\boldsymbol{\tau} = \boldsymbol{R} \odot \boldsymbol{F}$$

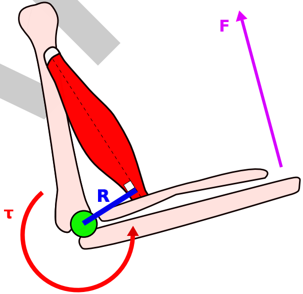
$\boldsymbol{R}$: muscle moment arms matrix
$\boldsymbol{F}$: muscle force

map muscle forces to static output force

step one: To map joint moments to forces, use J^{-T}

$$\boldsymbol{F_{6\times m}} = \begin{bmatrix} M_x^1 & \dots & M_x^m \\ M_y^1 & \dots & M_y^m \\ M_z^1 & \dots & M_z^m \\ F_x^1 & \dots & F_x^m \\ F_y^1 & \dots & F_y^m \\ F_Z^1 & \dots & F_z^m \end{bmatrix}$$

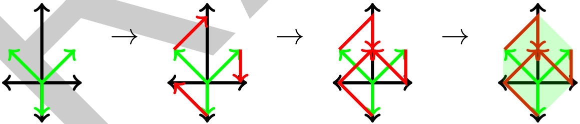
$$= \boldsymbol{J^{-T}\tau}$$



finding all possible muscle contributions to end effector forces

step two: Minkowski sum the columns (pretend the columns are the spanning set and make positive linear combinations)

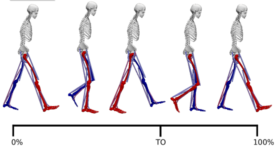
$$\boldsymbol{F_{6\times m}} = \begin{bmatrix} M_x^1 & \dots & M_x^m \\ M_y^1 & \dots & M_y^m \\ M_z^1 & \dots & M_z^m \\ F_x^1 & \dots & F_x^m \\ F_y^1 & \dots & F_y^m \\ F_z^1 & \dots & F_z^m \end{bmatrix}$$



pseudo-static analysis experimental design

using existing gait data set available from simTK.org [5]

Subjects walk at each of **4** self-selected speeds: xslow,slow,free,fast



3 muscle physiological considerations:

- ***F*0** only
- ***F*0** and l_m
- ***F*0**, l_m, and v_m

Spline each dataset to **0%** → **100%** of gait.

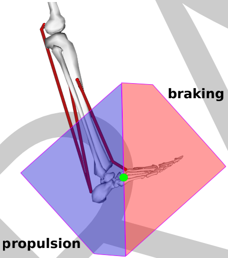
No between-subjects variables in the design

statistical analysis performed with the GLM procedure with SPSS

[5] Liu et al (2008), Muscle contributions to support and progression over a range of walking speeds

OpenSim and MATLAB analysis

post hoc analysis performed with OpenSim and MATLAB with CMC states



feasible force space split into propulsive and braking forces

muscle model change for physiological consideration

$\boldsymbol{F_0}$:

$F_m = \boldsymbol{F_0} \boldsymbol{a} \cos\alpha$

$\boldsymbol{l_m}$:

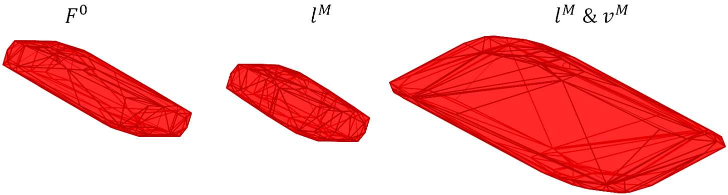
$F_m = \boldsymbol{F_0}(\boldsymbol{a} f^L(\boldsymbol{l_m}) + f^{PE}(\boldsymbol{l_m})) \cos\alpha$

$\boldsymbol{l_m}$ and $\boldsymbol{v_m}$:

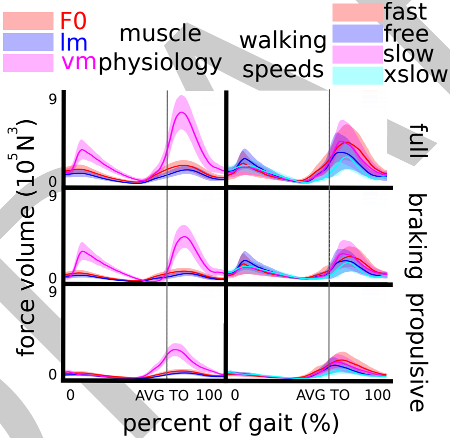
$F_m = \boldsymbol{F_0}(\boldsymbol{a} f^L(\boldsymbol{l_m}) f^V(\boldsymbol{v_m}) + f^{PE}(\boldsymbol{l_m})) \cos\alpha$

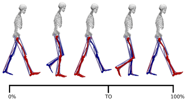
compute force volumes for each type of space

muscle physiology changes feasible force space



significant effects of l_m&v_m on volumes





take-aways: muscle physiology significantly changes force generating capacity

for the rest of this dissertation, equations that deal with muscles will include both the l_m and v_m effects

postural differences from different gait speeds were **not significant**

dynamical consideration

this section

next section

dynamical considerations and projection operators

a downstream parameter is an activations-dependent parameter

if we already have the set of dynamically consistent muscle activations (**inverse problem solution**), can we map back to the downstream parameter (joint moments, accelerations, etc)?

YES!

strategy using homogeneous coordinates

1. construct a projector reflecting the $\boldsymbol{a}$-dependent components of force
2. project to a dimension $n+1$
3. return to n and translate by the $\boldsymbol{a}$-independent components of force

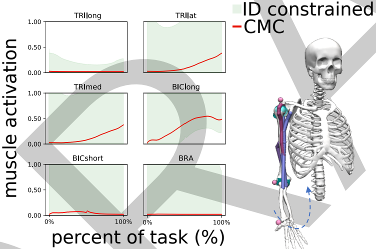
$$\boldsymbol{P} = \begin{bmatrix} \boldsymbol{P}_{n\times m}^{a-\text{dependent}} & 0 \\ \boldsymbol{P}_{1\times m}^{a-\text{independent}} & 1 \end{bmatrix}$$

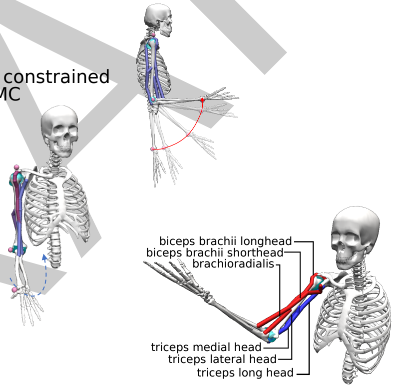
We can map feasible activations back to moments by: $\boldsymbol{\Gamma} = R(\mathcal{Q}) \odot \boldsymbol{Fa}$

We can map the feasible activations to induced accelerations by: $\ddot{\mathcal{X}} = J(\mathcal{Q})M(\mathcal{Q})^{-1}(R(\mathcal{Q}) \odot \boldsymbol{Fa} - J_{ext}^{T}\boldsymbol{F_{ext}})$

inverse problem solution (we will revisit this)

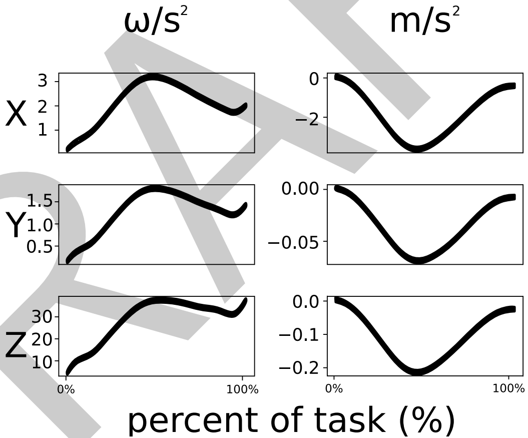
This isn't nullspace projection. Every activation set in FAS maps to **one** possible acceleration.



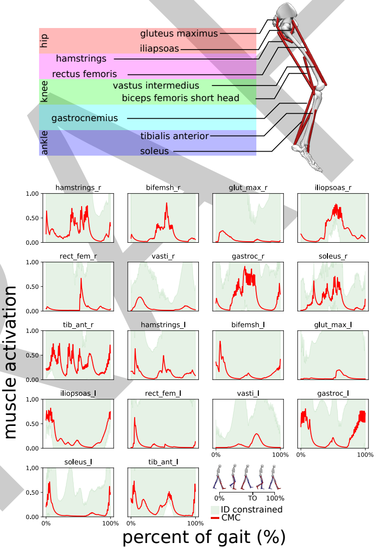


mapping FAS to hand accelerations

every activation maps to one set of accelerations

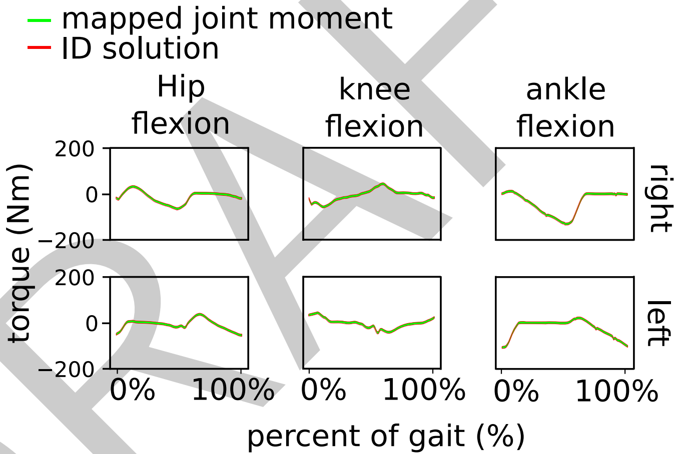


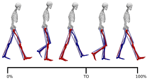
inverse problem solution (we will revisit this)



mapping FAS to joint moments

every activation maps to one set of joint moments





sanity test came up OKAY

nullspace of a task are the space of possible $\mathcal{Q}$ that that don't change the end effector position

these methods can be used to obtain feasible downstream parameters in the nullspace of a task

computing feasible activation space (FAS)

typical approach (CMC/SO) to computed control is to find an **optimized** solution constrained by ID ($\boldsymbol{\Gamma}_{task}$) according to a quadratic objective

instead, let's find the space of every possible solution

if we have ID and the kinematics, can we find the boundaries of possible solutions of muscle activation?

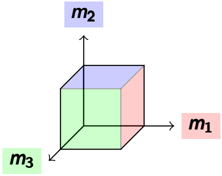
YES!

inverse problem boundaries

boundaries of activation space:

$$H_{\text{bounds}} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 1 & -1 & 0 & \dots & 0 & 0 \\ 1 & 0 & -1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 0 & 0 & \dots & -1 & 0 \\ 1 & 0 & 0 & \dots & 0 & -1 \end{bmatrix}$$

this is just a unit hypercube with $2^{(\text{n muscles})}$ vertices



Methods

step one: construct task constraints:

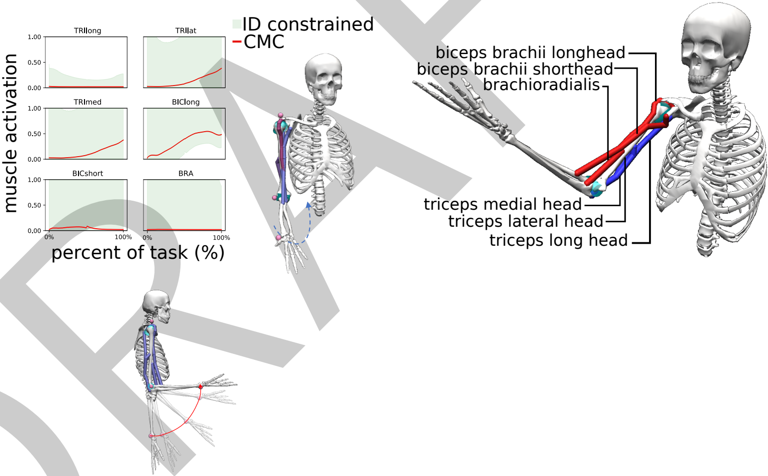
$$H_{task} = \begin{bmatrix} -\boldsymbol{\Gamma}_{\textbf{task}} + \sum F_m^{\text{pmf}} \boldsymbol{R_{m\times c}} + \tau_{ext} & \sum F_{m_m}^{\text{amf}} \boldsymbol{R_{1\times c}} a_m \\ \boldsymbol{\Gamma}_{\textbf{task}} - \sum F_m^{\text{pmf}} \boldsymbol{R_{m\times c}} - \tau_{ext} & \sum -F_{m_m}^{\text{amf}} \boldsymbol{R_{1\times c}} a_m \end{bmatrix}$$

$$\boldsymbol{\mathcal{H}}_{\text{FAS}} = \begin{bmatrix} H_{\text{task}} \\ H_{\text{bounds}} \end{bmatrix}$$

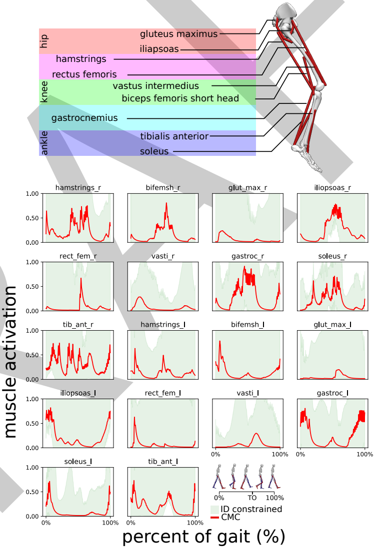
step two: use vertex enumeration on $\boldsymbol{\mathcal{H}}_{\text{FAS}}$ to find $\boldsymbol{\mathcal{V}}_{\text{FAS}}$

step three: move forward one step in Δt, repeat steps 1 and 2 till task-completion.

I can bound CMC's solution!

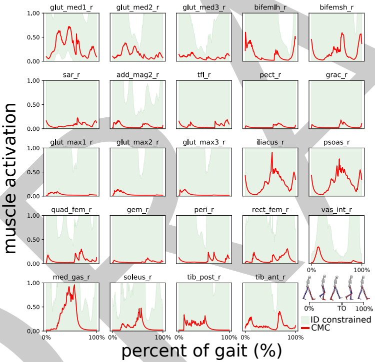


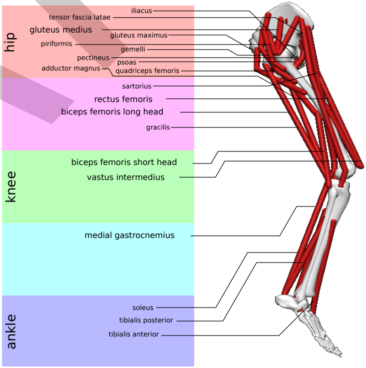
nonzero lower bounds indicates necessity



adding muscles with overlapping functions reduces necessity

23 DOF 54 muscles model freely available with OpenSim

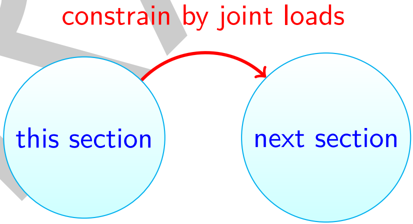




I need a better way to constrain H-FAS

I developed a method of calculating $\mathcal{V}_{\mathrm{FAS}}$ over each discrete time of a dynamic task for arbitrary models

previous research has assumed statics or ignored muscle parameters



constraining FAS by joint contact forces

FAS from the inverse problem was pretty big

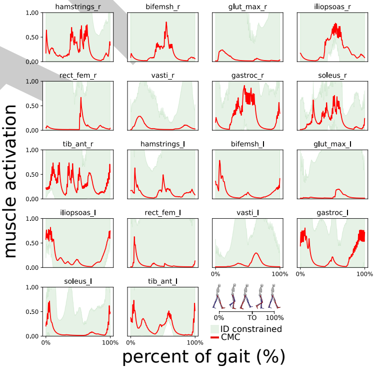
many activations had $\text{range} = [0, 1]$ over progression of the task

the **joint loads** are downstream parameters with analytical expressions that we can use along with $\boldsymbol{\Gamma_{task}}$

if we have ID solution, the IK solution, and joint loads can we **further** bound the possible solutions?

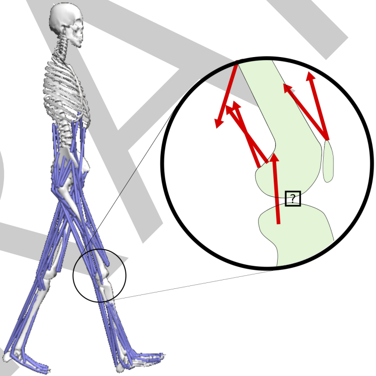
YES!

most muscles were unnecessary, even for simple models



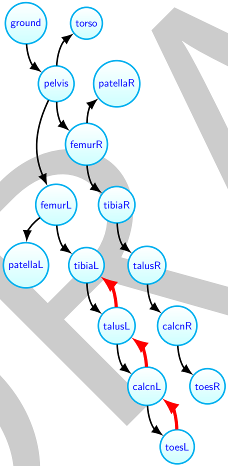
what are joint loads?

Not the same as Γ_{task}!
muscles apply tension to bodies along **lines of action** (LoA).
Sum the force vectors along LoA around the joint to find $\boldsymbol{F_j}$



OpenSim can't give us analytical expressions!

We have to sum expressions working up the **kinematic tree**.



if we know the **system topology** (kinematic chains), then we can write an algorithm that determines the joint loading expression as a function of muscle parameters.

$$
\begin{aligned}
F_j = \quad & F_{m_{amf_{b_1}}} + \cdots + F_{m_{amf_{b_b}}} + \\
& F_{m_{pmf_{b_1}}} + \cdots + F_{m_{pmf_{b_b}}} + \\
& F_{ext_{b_1}} + \cdots + F_{ext_{b_b}} + \\
& a_{b_1} m_{b_1} + \cdots + a_{b_b} m_{b_b}
\end{aligned}
$$

constructing the H-FAS with joint contact forces constraint

step one: construct task + joint loads constraints:

$$H_{task} = \begin{bmatrix} -\mathbf{\Gamma_{task}} + \sum F_m^{\text{pmf}} \boldsymbol{R_{m\times c}} + \tau_{ext} + \epsilon & \sum F_{m_m}^{\text{amf}} \boldsymbol{R_{1\times c}} a_m \\ \mathbf{\Gamma_{task}} - \sum F_m^{\text{pmf}} \boldsymbol{R_{m\times c}} - \tau_{ext} + \epsilon & \sum -F_{m_m}^{\text{amf}} \boldsymbol{R_{1\times c}} a_m \end{bmatrix}$$

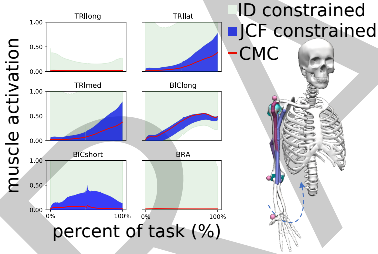
$$H_{\text{jcf}} = \begin{bmatrix} -\sum F_j + \epsilon - F_{m_{\text{pmf}_{1\to b}}} - F_{ext_{1\to b}} & \sum F_{m_{\text{amf}_{1\to b}}} \\ \sum F_j + \epsilon - F_{m_{\text{pmf}_{1\to b}}} - F_{ext_{1\to b}} & \sum F_{m_{\text{amf}_{1\to b}}} \end{bmatrix}$$

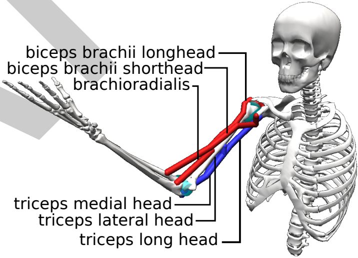
$$\mathcal{H}_{\text{FAS}} = \begin{bmatrix} H_{\text{task}} \\ H_{\text{jcf}} \\ H_{\text{bounds}} \end{bmatrix}$$

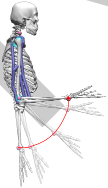
step two: use vertex enumeration on $\mathcal{H}_{\text{FAS}}$ to find $\mathcal{V}_{\text{FAS}}$

step three: move forward one Δt, repeat steps 1 and 2 till task-completion.

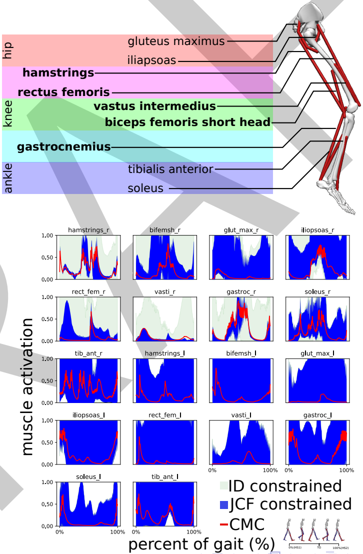
I can use elbow loads to capture the CMC solution with better accuracy



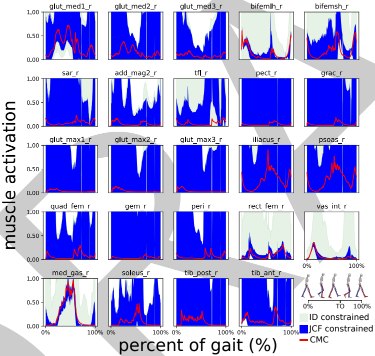


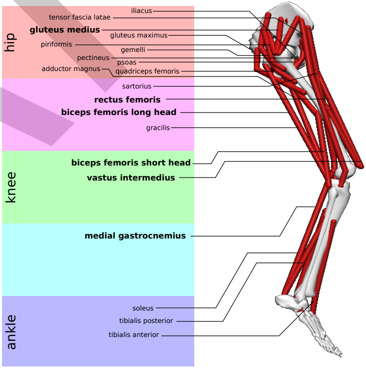


constraining planar gait model FAS by JCF



constraining complex gait model FAS by JCF

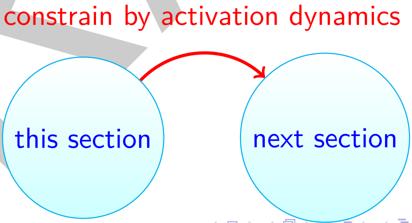




can we navigate H-FAS without computing V-FAS?

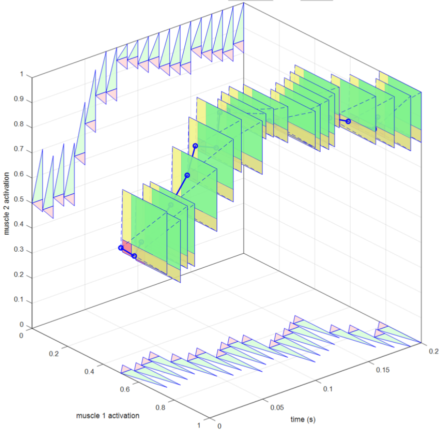
I developed a method of calculating $\mathcal{V}_{\text{FAS}}$ constrained by joint loading by procedurally constructing the analytical expression

this method works for arbitrary models and can be expanded to **any** muscle-dependent parameter as long as there's an analytical expression for it!



can we tie $\mathcal{H}$-FAS together in time?

this has only been **theorized** [6]



[6] Cohn (2018), Feasibility Theory Reconciles and Informs Alternate Approaches to Neuromuscular Control

Feasible Sets Analysis of Musculoskeletal Systems

Aravind Sundararajan

introduction

Pseudo-static

Methods

Results

Forward Problem

Methods

Results

Inverse Problem

Methods

Results

Loads Constrained

Methods

Results

FAS Trajectories

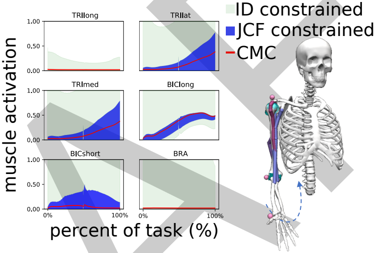
Methods

Results

Conclusion

acknowledgements

Probabilistic Computed Control



bounds plots are a little deceiving...

if we have the ID solution, the IK solution, joint loads, and first order activation dynamics, can we **even further** bound the possible solutions?

YES!

1st-order activation dynamics from Thelen:

$$\Delta a(a,0) = \frac{0 - a}{\tau_{\mathrm{deact}}}$$

$$\Delta a(a,1) = \frac{1 - a}{\tau_{\mathrm{act}}}$$

Lower Bound: $\boldsymbol{a}_{lb} = \boldsymbol{a} + \Delta t \Delta a(a,0)$

Upper Bound: $\boldsymbol{a}_{ub} = \boldsymbol{a} + \Delta t \Delta a(a,1)$

boundaries of activation space (again, again):

$$H_{\text{bounds}} = \begin{bmatrix} 0 & 1 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 \\ 1 & -1 & 0 & \dots & 0 & 0 \\ 1 & 0 & -1 & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 0 & 0 & \dots & -1 & 0 \\ 1 & 0 & 0 & \dots & 0 & -1 \end{bmatrix}$$

1st-order activations modification to H_{bounds}

$$H_{\Delta t \Delta a} = \begin{bmatrix} -a_{lb}^{1} & 1 & 0 & \dots & 0 \\ -a_{lb}^{2} & 0 & 1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -a_{lb}^{m} & 0 & 0 & \dots & 1 \\ a_{ub}^{1} & -1 & 0 & \dots & 0 \\ a_{ub}^{2} & 0 & -1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{ub}^{m} & 0 & 0 & \dots & -1 \end{bmatrix}$$

now H_{bounds} is **state-dependent**

step one: $\mathcal{H}_{\mathrm{FAS}}$ formation

$$H_{task} = \begin{bmatrix} -\mathbf{\Gamma}_{\mathbf{task}} + \sum F_m^{\mathrm{pmf}} \boldsymbol{R}_{\boldsymbol{m}\times\boldsymbol{c}} + \tau_{ext} + \epsilon & \sum F_{m_m}^{\mathrm{amf}} \boldsymbol{R}_{\boldsymbol{1}\times\boldsymbol{c}} a_m \\ \mathbf{\Gamma}_{\mathbf{task}} - \sum F_m^{\mathrm{pmf}} \boldsymbol{R}_{\boldsymbol{m}\times\boldsymbol{c}} - \tau_{ext} + \epsilon & \sum -F_{m_m}^{\mathrm{amf}} \boldsymbol{R}_{\boldsymbol{1}\times\boldsymbol{c}} a_m \end{bmatrix}$$

$$\mathcal{H}_{\mathrm{FAS}} = \begin{bmatrix} H_{\mathrm{task}} \\ H_{\Delta t \Delta a} \end{bmatrix}$$

now $\mathcal{H}_{\mathrm{FAS}}$ is **state-dependent**

we can also do joint constraints!

step one: $\mathcal{H}_{\mathrm{FAS}}$ formation

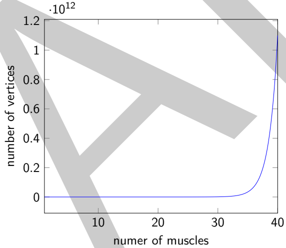
$$H_{task} = \begin{bmatrix} -\mathbf{\Gamma_{task}} + \sum F_m^{\mathrm{pmf}} \boldsymbol{R_{m\times c}} + \tau_{ext} + \epsilon & \sum F_{m_m}^{\mathrm{amf}} \boldsymbol{R_{1\times c}} a_m \\ \mathbf{\Gamma_{task}} - \sum F_m^{\mathrm{pmf}} \boldsymbol{R_{m\times c}} - \tau_{ext} + \epsilon & \sum -F_{m_m}^{\mathrm{amf}} \boldsymbol{R_{1\times c}} a_m \end{bmatrix}$$

$$H_{\mathrm{jcf}} = \begin{bmatrix} -\sum F_j - F_{m_{\mathrm{pmf}_{1\to b}}} - F_{ext_{1\to b}} & \sum F_{m_{\mathrm{amf}_{1\to b}}} \\ \sum F_j - F_{m_{\mathrm{pmf}_{1\to b}}} - F_{ext_{1\to b}} & \sum F_{m_{\mathrm{amf}_{1\to b}}} \end{bmatrix}$$

$$\mathcal{H}_{\mathrm{FAS}} = \begin{bmatrix} H_{\mathrm{task}} \\ H_{\mathrm{jcf}} \\ H_{\Delta t \Delta a} \end{bmatrix}$$

step two: find an interior point of $\mathcal{H}_{\text{FAS}}$

calculating $\mathcal{V}_{\text{FAS}}$ is extremely computationally costly



how do I get inside $\mathcal{H}_{\text{FAS}}$ without computing $\mathcal{V}_{\text{FAS}}$?

iterative method

conic optimization

iterating to the center

vertex center is like the center of geometry (avg. vertices in $\mathcal{V}$)

analytical center a_{ac} is like the center of mass

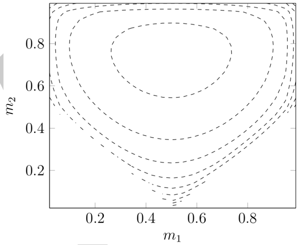
newton's method approach for finding the a_{ac}:

$$\begin{aligned} & \delta_{nt} = (A^T S^{-2} A)^{-1} A^T y \\ \text{s.t.} \quad & S = diag(\frac{1}{y}) \\ & y_i = b_i - A_i a_i \end{aligned}$$

$(A^T S^{-2} A)^{-1}$ is an inverse hessian $\mathcal{D} = J(\nabla)$

centers of FAS

another way to find a_{ac},
take the log barrier of $\mathcal{H}_{\mathrm{FAS}}$ and maximize it.



this is a conic optimization in the domain of the exponential cone:

$$\begin{aligned} & \max \sum \log\left(b_i - A_i^T a\right) \\ \text{s.t.} \quad & Aa \leq b \\ & 0 \leq a \leq 1 \end{aligned}$$

step three: walk to a new point in FAS

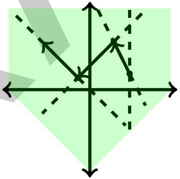
many interior point options!

previously in the literature for statics: **Hit-and-Run** (HAR)

unexplored for computed control: **Dikin Walk** (DW)

Hit-and-Run procedure

step one: pick a random unitary direction inside $\mathcal{H}_{\mathrm{FAS}}$ from a Gaussian distribution
step two: draw a line through the current point along the unitary direction
step three: pick any interior point along the line from a uniform distribution
step four: repeat steps one to three with the new interior point

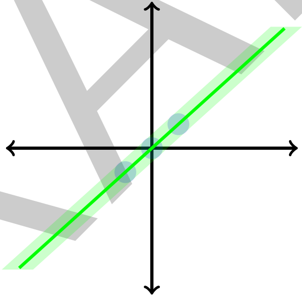


HAR has several complications

HAR is great for **statics**, but it tends to get trapped locally in thin feasible spaces like $\mathcal{H}_{\mathrm{FAS}}$

HAR approaches the uniform distribution in at most $\mathcal{O}(d^2\gamma_\kappa^2)$ where d is rows of $\mathcal{H}_{\mathrm{FAS}}$

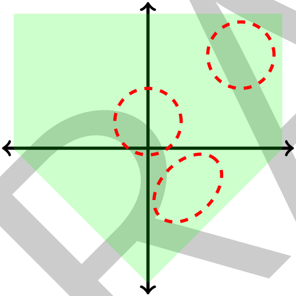
γ_κ is the **matrix condition number**



Could use scaling/damping methods, but why not use a better algorithm?

Dikin Walk is a better alternative to HAR

DW approaches the sampling distribution in $\mathcal{O}(nd)$



Hessian of the log barrier defines an ellipsoid inside $\mathcal{H}_{\mathrm{FAS}}$

step one: calculate the hessian of the log barrier $\mathcal{D}_a = \nabla \mathcal{F}_a$
step two: select a new activation from $\{u \in R^d | (u-a)^T \mathcal{D}_a (u-a) \leq R\}$
select u from the multivariate Gaussian $g(z)$ centered at a with user-selected radius r and covariance $\frac{r^2}{n}\mathcal{D}_a^{-1}$:

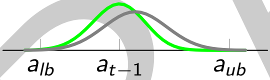
$$z = a + \frac{r}{\sqrt{n}}(\mathcal{D}_a)^{-\frac{1}{2}}\boldsymbol{g}, a = z$$

where $\boldsymbol{g}$ is a vector sampled from the standard Gaussian
step three: repeat steps one and two

1st-order activation dynamics heavily skews toward high activation

The Skew Normal Distribution:

$$f(a) = 2\phi(a)\Phi(\alpha a)$$



$a_{\mathrm{lb}} \to a_{t-1}$ and $a_{t-1} \to a_{\mathrm{ub}}$ is **not symmetrical**

Where $\Phi(a)$ is the cumulative distribution function:

$$\Phi(a) = \frac{1}{2}(1 + \mathrm{erf}(\frac{\alpha a}{\sqrt{2}}))$$

and $\phi(a)$ is the probability density function:

$$\phi(a) = \frac{1}{\sqrt{2\pi}} e^{-\frac{a^2}{2}}$$

$\mathrm{erf}(a)$ is also known as the Gaussian error function

I developed a modified DW with **multivariate skew normal** to account for this

method of moments
maximum likelihood estimate of the shape parameter

strategy: estimate the quartiles and use Bowley's Skewness Estimate as input to an MLE function

$$\gamma = \frac{Q_3 + Q_1 - 2Q_2}{Q_3 - Q_1}$$

$$Q_1 = a_{t-1} - \frac{.67\Delta t \tau_{deact}}{2}$$
$$Q_2 = a_{t-1}$$
$$Q_3 = a_{t-1} + \frac{.67\Delta t \tau_{act}}{2}$$

Method of Moments:

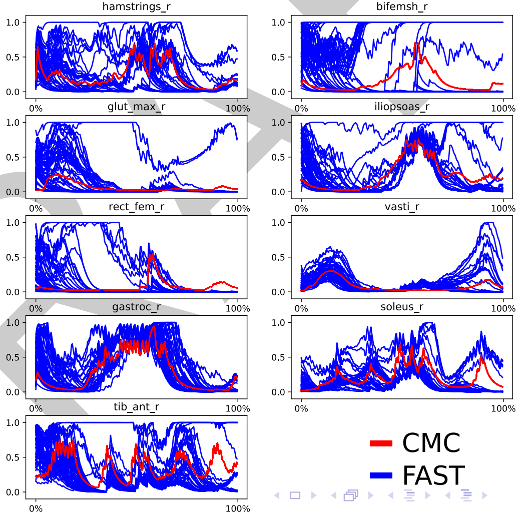
$$|\delta| = \sqrt{\frac{\pi}{2} \frac{|\gamma|^{\frac{2}{3}}}{|\gamma|^{\frac{2}{3}} + \left(\frac{4-\pi}{2}\right)^{\frac{2}{3}}}}$$

and finally:

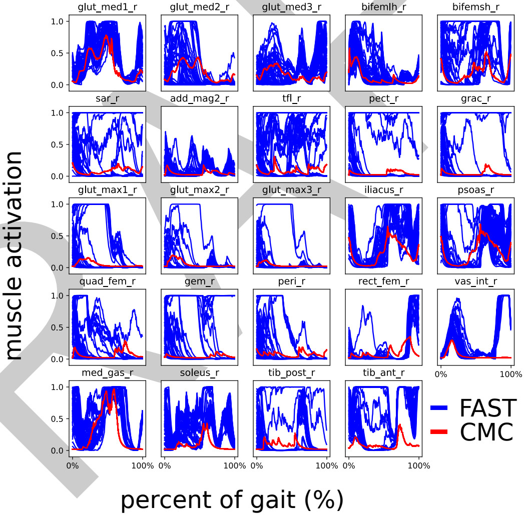
$$\hat{\alpha} = \frac{\delta}{\sqrt{1-\delta^2}}$$

now let's apply Sunny's Walk to the computed controls problem!

FAST analysis of the planar gait model



FAST analysis: increasing the model complexity



FAST is FAST!

for the models tested in this dissertation,
FAST was **up to 30 times faster** than CMC

concluding remarks

1. this dissertation was an expansion and synthesis of tools that can be used to investigate the **boundaries of control** over the course of a dynamic task
2. these methods are **generalized** to work for most models and tasks
3. I developed a **comprehensive software platform** for performing feasible sets analysis
4. these tools are an **umbrella** for other analyses (muscle synergies, task prioritization, minimization)
5. these tools can be used by researchers interested in neural nets and **machine learning** with computed control

Feasible Sets Analysis of Musculoskeletal Systems

Aravind Sundararajan

acknowledgements