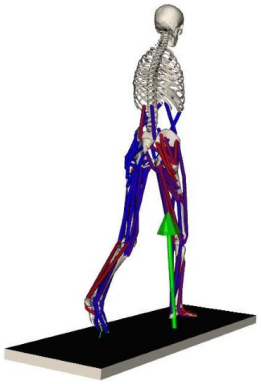
Slide 1

Forward Simulation

Slide 2

Forward Dynamics

- Validation: do forces estimated from inverse dynamics reproduce the observed motion?
- Understanding: how do muscle forces generate motion – what are the “cause and effect” relationships?
- Prediction: “what if” a muscle or joint is altered, how will performance change?

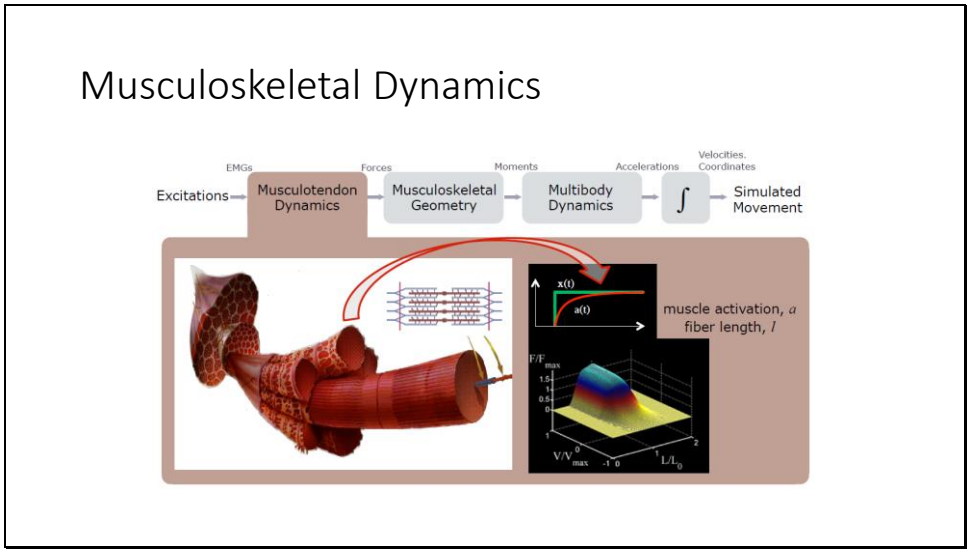


Slide 3

Key Concepts

- Musculoskeletal model dynamics
- States of a musculoskeletal model
- Controls of a musculoskeletal simulation
- Numerical integration of dynamical equations

Slide 4



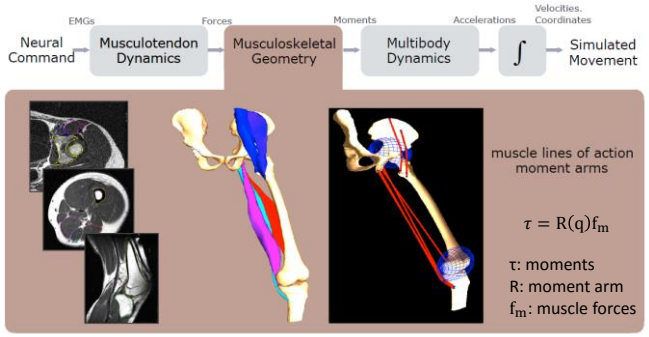
The Forward Dynamics Tool uses the model together with the initial states and controls to run a muscle-driven forward dynamics simulation. A forward dynamics simulation is the solution (integration) of the differential equations that define the dynamics of a musculoskeletal model.

Muscle excitations are not directly related to EMGs, but the latter is used to validate quantitatively.

Typically, muscle excitations are generated using the Computed Muscle Control (CMC) tool. As a pre-cursor to running CMC, the Residual Reduction Algorithm (RRA) is used to minimize the effects of modeling and marker data processing errors that aggregate and lead to large nonphysical compensatory forces called residuals. Thus the typical workflow for generating a muscle-driven simulation after importing experimental data is Scale->IK->RRA->CMC->Forward Dynamics.

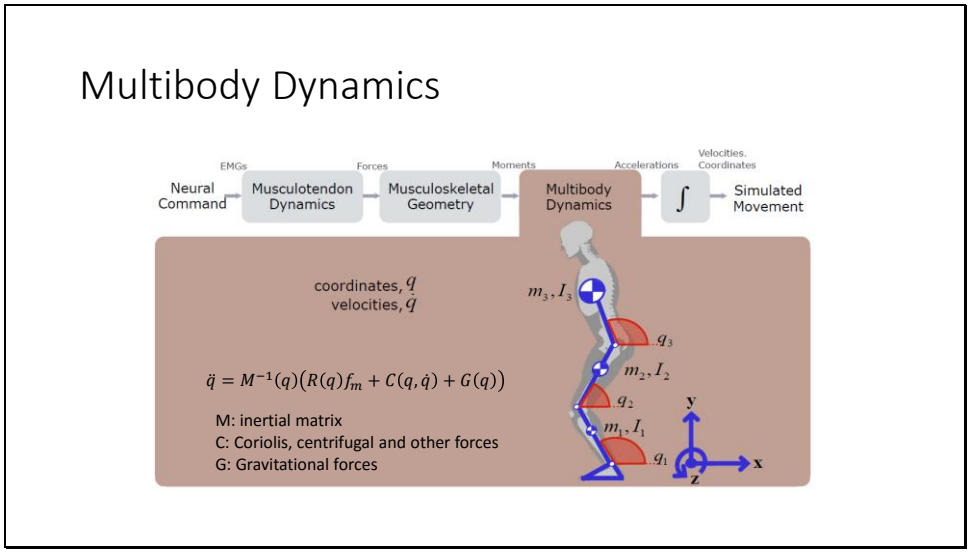
Slide 5

Musculoskeletal Geometry



Muscle forces must be mapped to moments around the joints. The moment arm is estimated geometrically from the pose of the body, and the muscle path.

Slide 6

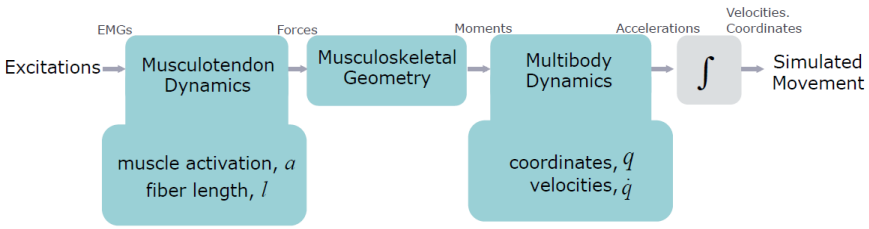


In contrast to inverse dynamics where the motion of the model was known and we wanted to determine the forces and torques that generated the motion, in forward dynamics, a mathematical model describes how coordinates and their velocities change due to applied forces and torques (moments). From Newton's second law, we can describe the accelerations (rate of change of velocities) of the coordinates in terms of the inertia and forces applied on the skeleton as a set of rigid-bodies.

Slide 7

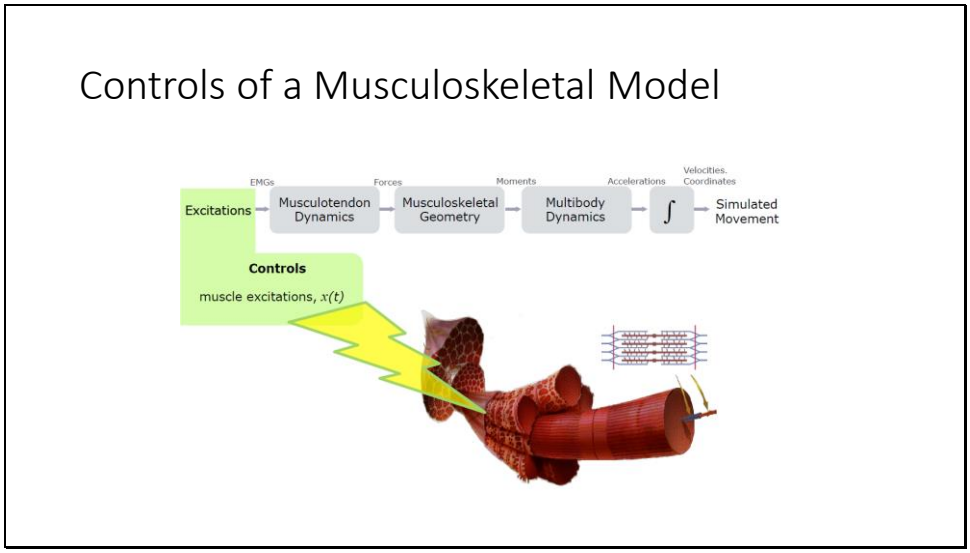
States of a Musculoskeletal Model

- States are model variables that are governed by the dynamics
- All measures of interest can be calculated from the states



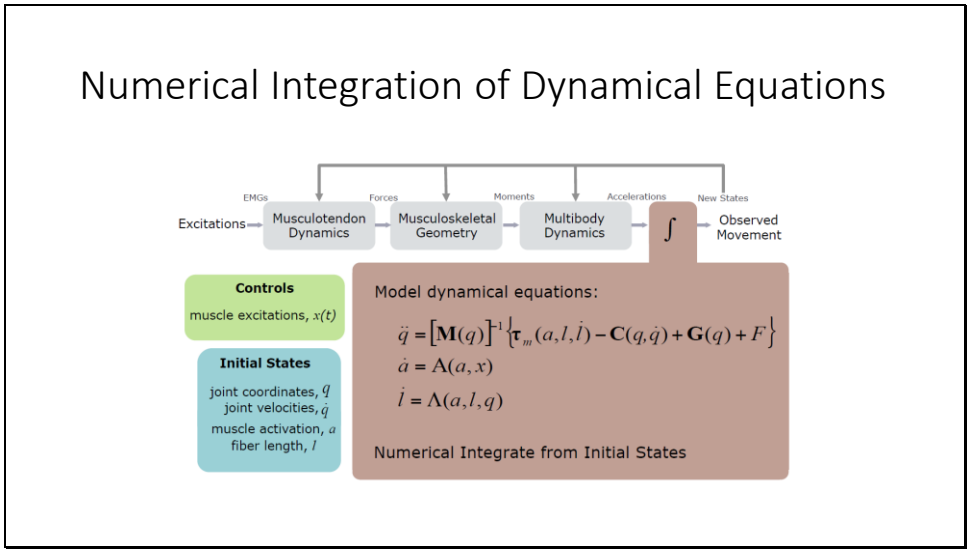
The *state* of a model is the collection of all model variables defined at a given instant in time that are governed by dynamics. The model dynamics describe how the model will advance from a given state to another through time. In a *musculoskeletal model* the states are the coordinates and their velocities and muscle activations and muscle fiber lengths. The dynamics of a model require the state to be known in order to calculate the rate of change of the model states (joint accelerations, activation rates, and fiber velocities) in response to forces and controls.

Slide 8



The forces (e.g., muscles) in a *musculoskeletal model* are governed by dynamics and have inputs that affect their behavior. In OpenSim, these inputs are called the *controls* of a model, which can be excitations for muscles or torque generators. Ultimately, controls determine the forces and/or torques applied to the model and therefore determine the resultant motion.

Slide 9



A simulation is the integration of the musculoskeletal model's dynamical equations starting from a user-specified initial state. After applying the controls, the activation rates, muscle fiber velocities, and coordinate accelerations are computed. Then, new states at small time interval in the future are determined by numerical integration. A 5th-order Runge-Kutta-Feldberg integrator is used to solve (numerically integrate) the dynamical equations for the trajectories of the musculoskeletal model states over a definite interval in time. The Forward Dynamics Tool is an open-loop system that applies muscle/actuator controls with no feedback, or correction mechanism, therefore the states are not required to follow a desired trajectory.

Forward dynamics simulations are sensitive to initial conditions and it is good practice to double check that they are appropriate for the desired simulation.

If the Forward Tool fails gracefully (i.e., without crashing OpenSim) or the output of the Forward Tool drifts too much (i.e., the model goes crazy), shorten the interval over which the Forward Tool runs (i.e., make initial_time and final_time closer to each other in the Forward Tool setup dialog box or setup file). Open-loop forward dynamics tends to drift over time due to the accumulation of numerical errors during integration.

Slide 10

Forward Dynamics Exercise

A forward dynamics simulation is

A. a musculoskeletal model
B. muscle-driven
C. a simulation that uses feedback
D. the integration of dynamical equations

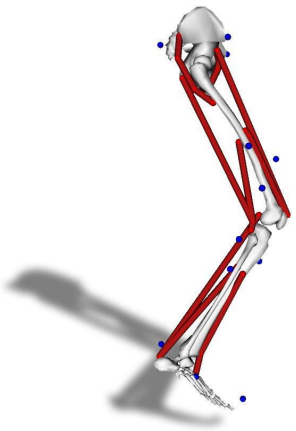
D.

Slide 11

Forward Dynamics Exercise

The musculoskeletal model for this tutorial (leg39) has how many states?

A. 3
B. 9
C. 12
D. 24



The model has 3DOFs and 9 muscles.

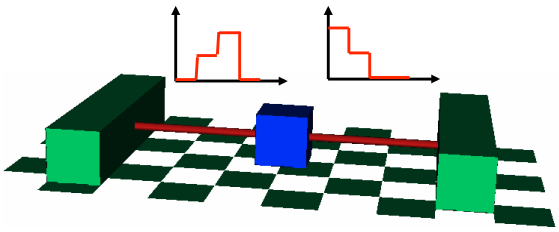
D. Each coordinate has two states q, u (coordinate and speed), while the muscle has also two states, namely the muscle length and activation.

Slide 12

Forward Dynamics Exercise

Given the model below with two identical muscles and their levels of excitation plotted versus time, which way will the block initially move if starting from rest?

A. To the left
B. Does not move
C. To the right
D. Upward



B.

Slide 13

Forward Dynamics Exercise

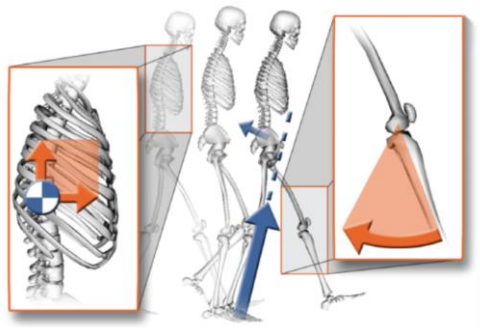
Given initial q and $\dot{q}$ and muscle a and l, how are these states determined at a small instant ahead in time?

A. Specify controls and compute $\dot{a}$, $\dot{l}$, and $\ddot{q}$ from model dynamics
B. Numerically integrate forces and controls from model differential equations
C. Numerically integrate $\dot{a}$, $\dot{l}$, and $\ddot{q}$
D. Numerically differentiate forces and controls from the dynamical equations
E. A & C

E.

Slide 14

Reducing Residuals



The purpose of Residual Reduction is to minimize the effects of errors in modeling and marker kinematics that lead to significant nonphysical forces called residuals. Specifically, residual reduction slightly adjusts the mass properties of a subject-specific model and the joint kinematics from inverse kinematics to improve dynamic consistency with respect to the ground reaction force data.

Residual reduction is a form of forward dynamics simulation that uses a tracking controller to follow model kinematics determined from the inverse kinematics. Computed muscle control (CMC) serves as the controller, but without muscles the skeleton of the model can be used to determine a mass distribution and joint kinematics that are more consistent with ground reaction forces.

Slide 15

What are residuals?

Non-physical forces that account for inconsistencies between experimental GRFs and joint accelerations estimated from experimental markers.

$$F = ma + R$$

Inconsistencies due to:

1. noise in marker and joint angle data (differentiating angles for accelerations)
2. inaccuracies in model geometry and mass distribution

Residual reduction is primarily intended for gait, i.e., movements like walking and running where the model is displaced relative to the ground while subject to ground reaction forces and torques. An example gait model (gait2354_simbody.osim) consisting of ten rigid segments (bones) where 17 of the 23 generalized coordinates (degrees of freedom) of the model represent angles for the joints connecting the rigid segments together. Each of these 17 degrees of freedom is actuated by a single torque actuator.

The remaining 6 generalized coordinates represent the 6 degrees of freedom (3 translational, 3 rotational) between the model's pelvis and the ground. To simulate walking, we need some way of representing how the model propels itself forward relative to the ground. One way would be to use a foot-ground contact mechanism.

Instead, we present a simpler solution: represent the 6 degrees of freedom between the pelvis and the ground as a 6-degree-of-freedom joint between the pelvis and the ground, and actuate each degree of freedom with its own torque actuator. Each of these 6 actuators is called a *residual actuator*. Now the model has 23 degrees of freedom and 23 actuators, i.e., exactly one actuator per degree of freedom. The three residuals that actuate the 3 translational degrees of freedom between the pelvis and the ground are the *residual forces*, whose values we denote by *Fx*, *Fy*, and *Fz*. The 3 rotational degrees of freedom are actuated by the *residual torques* (or *moments*), whose values we denote by *Mx*, *My*, and *Mz*. *Fx* is the force applied along the X (forward) axis, *Fy* is the force applied along the Y (vertical) axis, *Mx* is the torque applied about the X (forward axis), and so on.

Typically, modeling assumptions (e.g., having a model with no arms), noise, and other errors from motion capture data lead to *dynamic inconsistency*; essentially, the ground reaction forces and acceleration estimated from measured marker kinematics for a subject do not satisfy Newton's Second Law (F = ma).

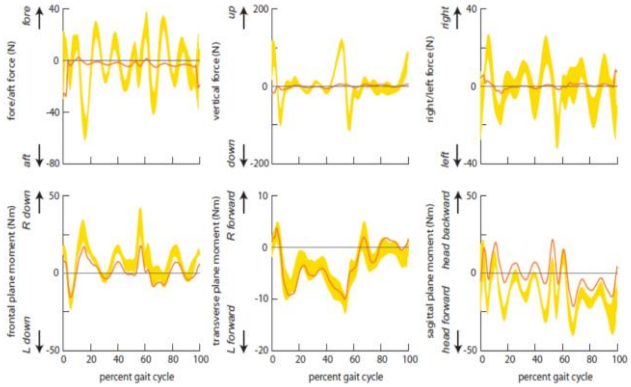
Roughly speaking, the 6 residuals amount to adding a new force to the equation that accounts for inconsistency: F +R = ma

Why reduce residuals?

1. Residuals are non-physical and necessary only to account for errors
2. Want muscles to account for all movement
3. To have confidence in muscle contributions

Slide 17

Sample residual reduction during gait

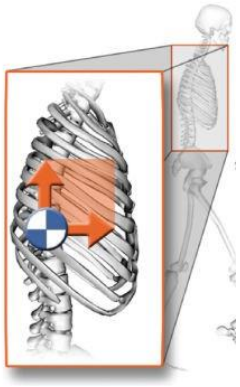


Slide 18

How can you reduce residuals?

- Torso is most massive and error prone to estimate
- Location of Torso mass center also difficult to estimate

1. Adjust mass distribution including Torso COM location



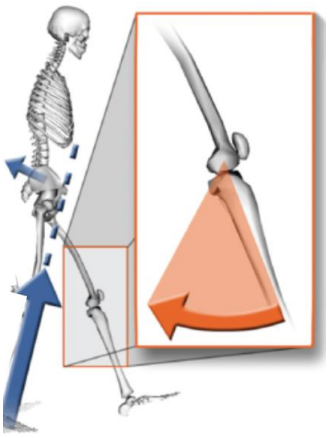
Slide 19

How can you reduce residuals?

- Joint kinematics estimated from marker position has inaccuracies
- Differentiation of kinematics can yield non-physical accelerations

1. Adjust mass distribution including Torso COM location
2. Adjust kinematics slightly while satisfying equations of motion

RRA tracks kinematics in a forward dynamics simulation



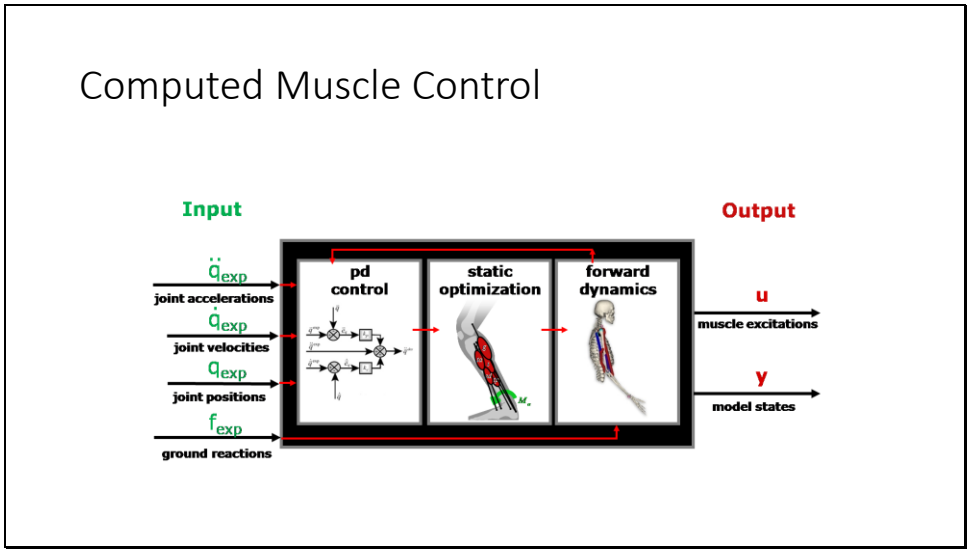
Slide 20

Tips and Tricks

- Keep optimal forces for residuals low (increase control bounds if necessary)
- Lower weight on kinematics that track closely or have low con!dence in measurement
- Make mass adjustments and run RRA again - repeat until residuals no longer change

- RMS difference in joint angle during the movement should be less than 2-5º (or less than 2 cm for translations).
- Peak Residual Forces should be less than 10-20 N.
- Compare the residual moments from RRA to the moments from Inverse Dynamics. You should see a 30-50% reduction in peak residual moments.
- Compare the joint torques/forces to established literature (if available). Try to find data with multiple subjects. Your results should be within one standard deviation of the literature.
- Optimal forces for residuals should be low to prevent the optimizer from "wanting" to use residual actuators (an actuator with large optimal force and low excitation is "cheap" in the optimizer cost).
- To help minimize residuals, make an initial pass with default inputs, then check residuals and coordinate errors. To reduce residuals further, decrease tracking weights on coordinates with low error. You can also try decreasing the maximum excitation on residuals or the actuator optimal force.
- If RRA is failing, try increasing the max excitation for residuals by 10x until the simulation runs. Then try working your way back down while also "relaxing" tracking weights on coordinates.
- If residuals are very large (typically, this is greater than 2-3x BW, depending on the motion), there is probably something wrong with either (i) the scaled model, (ii) the IK solution, or (iii) the applied GRFs. To double check that forces are being applied properly, visualize GRFs with IK data (you can use the “Preview motion data” function in the GUI).

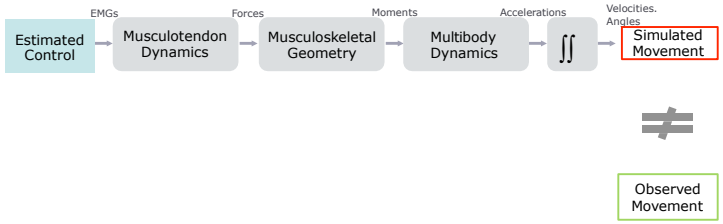
Slide 21



The purpose of Computed Muscle Control (CMC) is to compute a set of muscle excitations (or more generally actuator controls) that will drive a dynamic musculoskeletal model to track a set of desired kinematics in the presence of applied external forces (if applicable).

Slide 22

Muscle Driven Forward Simulation

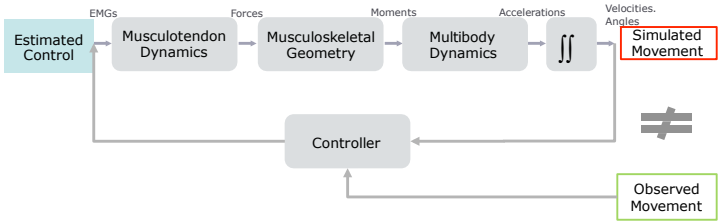


- Static optimization muscle model differs from dynamical model in forward simulation
- Acceleration data is discrete and noisy
- A nonlinear dynamical systems can be chaotic

If we use the muscle activations estimated form Static Optimization as inputs to forward dynamics, the simulated movement will be different with respect to the observed movement.

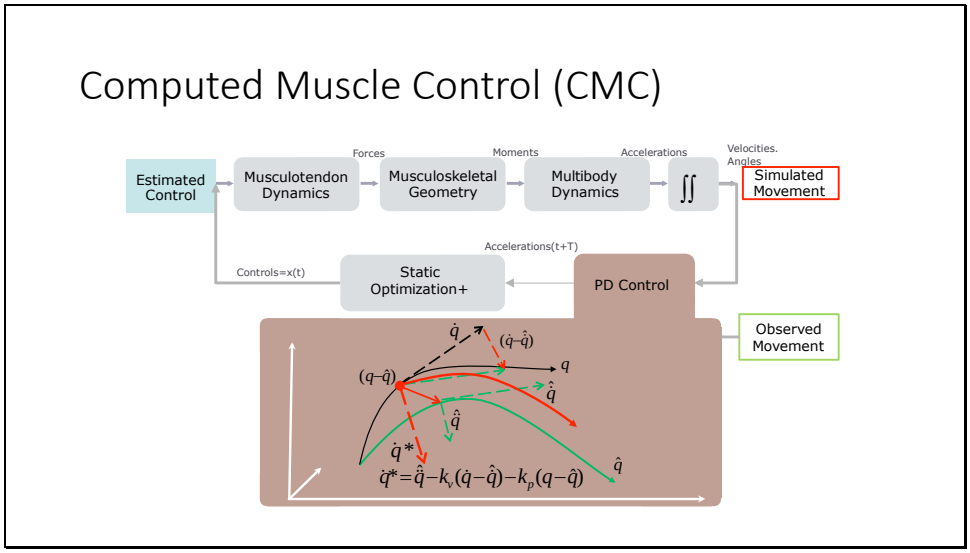
Slide 23

Muscle Driven Forward Simulation



- Close the loop and try to track the observed movement in a forward dynamic manner: Computed muscle control

Slide 24



The CMC tool computes muscle excitation levels that will drive the generalized coordinates (e.g., joint angles) of a dynamic musculoskeletal model towards a desired kinematic trajectory. CMC does this by using a combination of proportional-derivative (PD) control and static optimization.

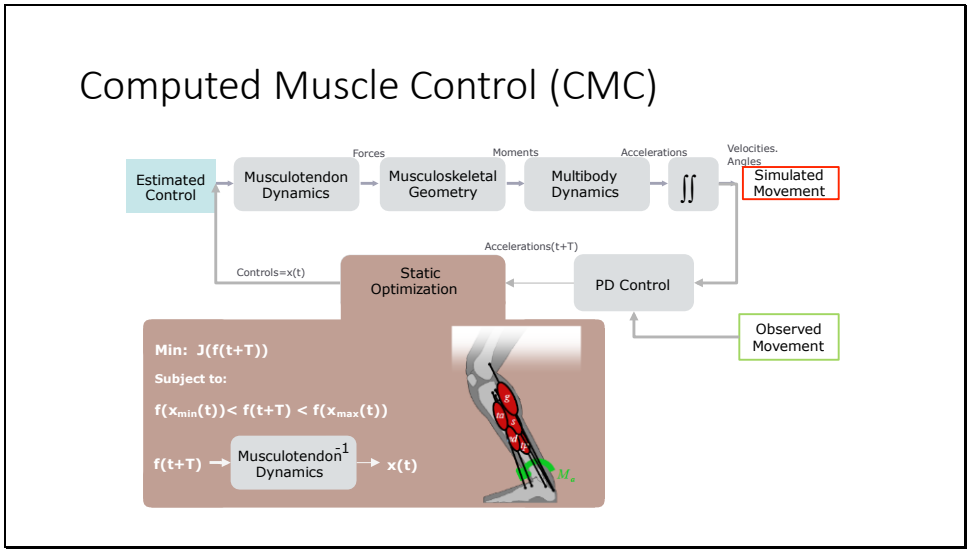
The first step in the CMC algorithm is to compute a set of desired accelerations $\vec{q}^{\,**}$ which, when achieved, will drive the model coordinates $\vec{q}$ toward the experimentally-derived coordinates $\vec{q}$ exp. The desired accelerations are computed using the following PD control law.

The $\vec{k}$ *v* and $\vec{k}$ *p* are the feedback gains on the velocity and position errors, respectively. Because the forces that muscles apply to the body cannot change instantaneously, the desired accelerations are computed for some small time *T* in the future. For musculoskeletal models, *T* is typically chosen to be about 0.010 seconds. This time interval is short enough to allow adequate control, but long enough to allow muscle forces to change. If these desired accelerations are achieved, errors between the model coordinates and experimentally-derived coordinates will be driven to zero. To drive these errors to zero in a critically damped fashion (i.e., without over-shooting or over-damping), the velocity gains can be chosen using the following relation: $\vec{k}$ *v*=2 * sqrt($\vec{k}$ *p).* For musculoskeletal models, it works well if the error gains are chosen to drive any errors to zero slowly. The error gains $\vec{k}$ *v*=20 and $\vec{k}$ *p*=100 will reduce tracking errors.

Before starting the CMC algorithm, initial states for the model are computed. The states comprise the generalized coordinates (joint angles), generalized speeds (joint

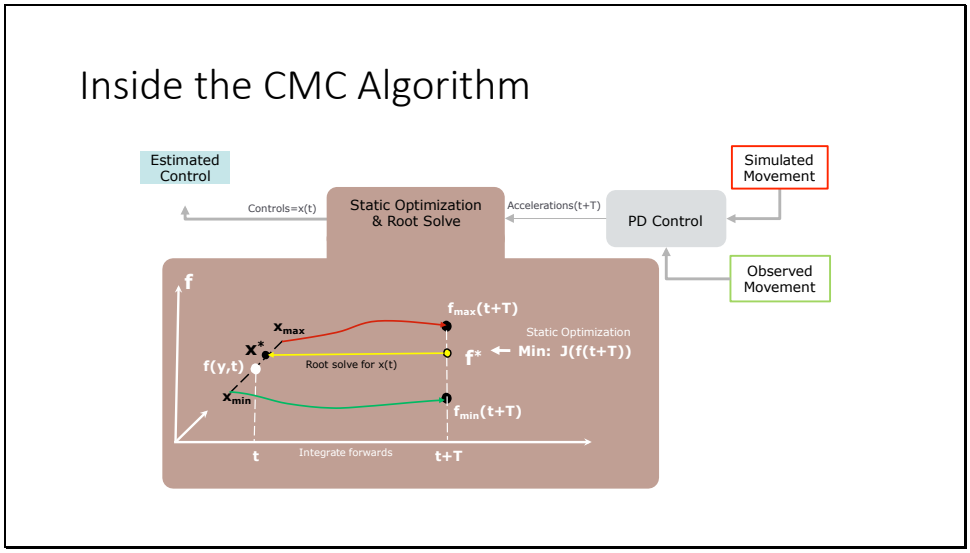
angular velocities), plus any muscle states (e.g., muscle activation levels and fiber lengths). While the initial values of the generalized coordinates and speeds can be taken from the desired kinematics that you specify, the initial values of the muscle states are generally unknown. To compute viable starting muscle states, CMC is applied to the first 0.030 seconds of the desired movement. Because the muscle states are generally out of equilibrium and muscle forces can change dramatically during this initial time interval, the simulation results during this interval are generally not valid. Therefore, you should be sure to start CMC at least 0.030 seconds prior to the interval of interest.

Slide 25



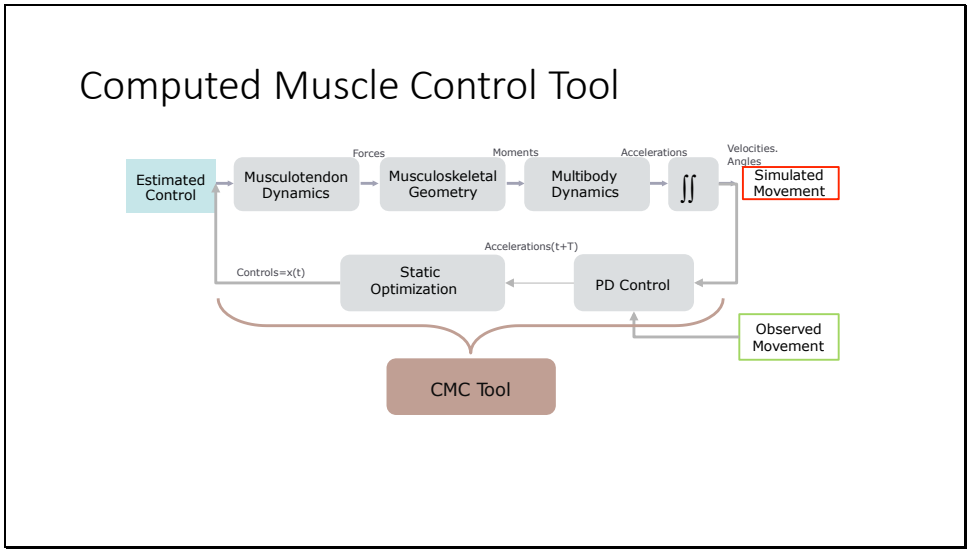
The next step in CMC is to compute the actuator controls $\vec{x}$ that will achieve the desired accelerations $\ddot{q}^*(t+T)$. Most of the time, the controls are predominantly comprised of muscle excitations, but this is not required. Any kind of actuator can be used with CMC (e.g., idealized joint moments). Static optimization is used to distribute the load across synergistic actuators. It is called "static" optimization because the performance criterion (i.e., the cost index) is confined to quantities that can be computed at any instant in time during a simulation. Using criteria like jump height or total metabolic energy over a gait cycle, for example, are not possible because these require simulating until the body leaves the ground or until the end of the gait cycle is reached.

Slide 26



Once CMC finishes execution, you will typically want to compare the computed muscle excitation patterns with prototypical or measured electromyographical measurements. If desired, constraints can be placed on the upper and lower bounds of the controls $\vec{x}$ as a function of simulation time. The bounds on the controls $\vec{x}$ are specified in an XML input file. For muscle excitations, the default upper bound is typically 1.0 (full excitation), and the default lower bound is typically a small number just above 0.0 (no excitation), such as 0.01 or 0.02. The lower bound is not set at precisely 0.0 because mathematical models of muscle are often not as well-behaved when excitation goes all the way to 0.0. The format of the control constraints is shown in the following section.

Slide 27



The final step in the CMC algorithm is to use the computed controls to conduct a standard forward dynamic simulation, advancing forward in time by *T*. These steps—computing the desired accelerations, static optimization, and forward dynamic simulation—are repeated until time is advanced to the end of the desired movement interval.

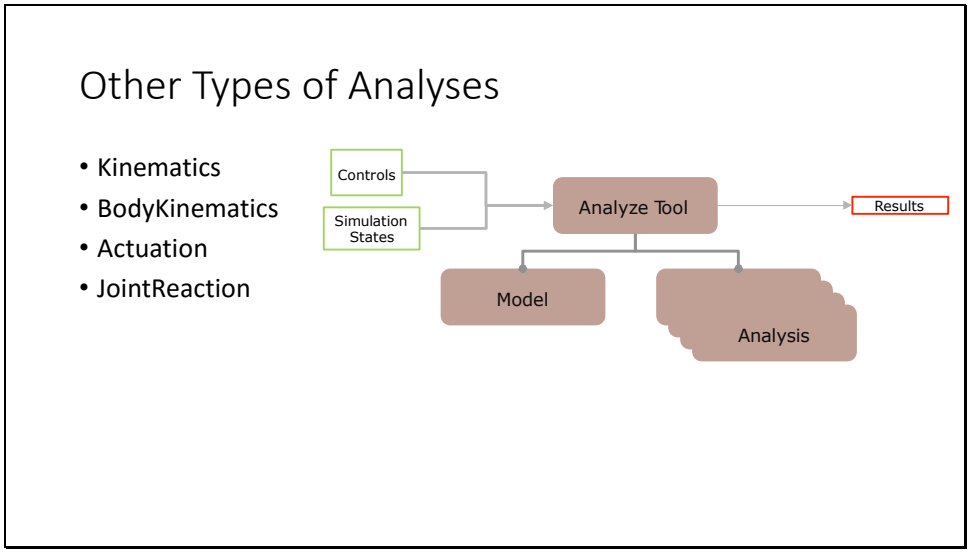
Slide 28

Tips and Tricks

- You can use results from IK or RRA. For best results, track RRA output not IK
- Increase max excitation of reserves if CMC is failing
- Compare to EMG and constrain excitations where there is a mismatch
- Command Line: cmc –S cmc_setup_!le.xml

1. Compare the simulated activations to experimental EMG data (either recorded from your subject or from the literature). Activations should exhibit similar timing and magnitude to EMG data.
2. Peak reserve actuators torques should typically be less than 10% of the peak joint torque.
3. If performing Perturbation or Induced Acceleration Analysis, you should verify that reserves and residuals contribute less than 5% to the net acceleration of interest.
4. Compare your results (muscle activations or forces) to other simulations, if available.
5. To help minimize reserve torques, make an initial pass with default inputs, and then check reserves, residuals, and joint angle errors. To reduce reserves further, decrease tracking weights on coordinates with low error.
6. Optimal forces for reserves should be low to prevent optimizer from "wanting" to use reserve actuators (an actuator with large optimal force and low excitation is "cheap“ in the optimizer cost). Increase the maximum control value of residuals so they can generate sufficient force, but are penalized for doing so.
7. If CMC is failing, try increasing the max excitation for reserves and residuals by 10x until the simulation runs. Then try working your way back down while also "relaxing“ tracking weights on coordinates.

Slide 29



The Analyze Tool steps in time through a set of input data specifying the state of a model; at each time step, the tool runs a set of analyses on the model. Available analyses include:

Kinematics: Records the generalized coordinates (q's), generalized speeds (u's), and the accelerations (i.e., derivatives of the generalized speeds: du/dt)
BodyKinematics: Records the configuration (center of mass position and orientation) of each body, as well as their velocities (linear and angular) and accelerations (linear and angular). Additionally, it records the overall center of mass of the model, as well as the velocity and acceleration of this center of mass.
Actuation: Records the generalized force, speed, and power developed by each actuator of the model. The generalized force can either be a force (with units N) or a torque (with units Nm). The actuator speed is the rate at which the actuator shortens. Depending on the actuator, a speed can be either a translational speed (m/s) or an angular speed (deg/s). An actuator power (Watts) is the rate at which an actuator does work. Positive work means that the actuator is delivering energy to the model; negative power means that the actuator is absorbing energy from the model.
JointReaction: Reports the joint reaction loads from a model. For a given joint, the reaction load is calculated as the forces and moments required to constrain the body motions to satisfy the joint as if the joint did not exist. The reaction load acts at the joint center (mobilizer frame) of both the parent and child bodies and either force can be reported and expressed in either the child, parent or ground frames. The default behavior is to express the force on the child in the ground frame.

The analyses can be added to the list to run via the GUI or by editing a setup file. The input data typically are loaded from files, and may come from experiments or

simulations. The results are collected and written to file, usually storage (.sto) files, which you can open with other programs, such as Matlab or Microsoft Excel, for further analysis or plotting. Any analysis available in OpenSim can be included in the list of analyses to run.

Depending on the analyses you would like to run, there are four types of input that may be needed to analyze a model. These are described below.

states States are the variables of a model that are governed by differential equations and thus are integrated during a simulation. The most common examples of states are the generalized coordinates (q; e.g., joint angles) and speeds (u; e.g., joint angular velocities) that specify the configuration of a model. Every model has them. The states are not limited to the coordinates and speeds, however. Muscles frequently have states. Muscle activation and fiber length are common examples of muscle states.

controls Controls are independent variables used to control the behavior of a model. Muscle excitations are an example. They are not governed by differential equations, but typically are free to take on any value between zero (no excitation) and one (full excitation). The controls of a model are frequently the variables used as the control parameters in optimization problems.

external loads External loads are forces or torques applied between the ground (or world) and the bodies of a model. For example, you might have force plate data that were recorded during a gait experiment. The recorded forces and moments can be applied to the right and left feet of the model by inputting the data as external loads. In OpenSim 1.5, external loads can be applied to at most two bodies.