Tutorial 3 - Scaling, Inverse Kinematics, and Inverse Dynamics

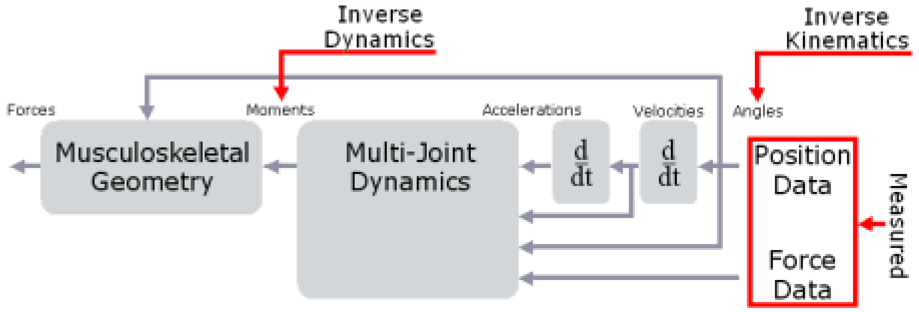
I. Objectives

Purpose

The purpose of this tutorial is to demonstrate how OpenSim solves an inverse kinematics problem and an inverse dynamics problem using experimental data. To diagnose movement disorders and study human movement, biomechanists frequently ask human subjects to perform movements in a motion capture laboratory and use computational tools to analyze these movements. A common step in analyzing a movement is to compute the joint angles and joint moments of the subject during movement. OpenSim has tools for computing these quantities:

1. *Inverse kinematics* is used to compute joint angles.
2. *Inverse dynamics* is used to compute net joint reaction forces and net joint moments.

Inverse kinematics computes the joint angles for a musculoskeletal model that best reproduce the motion of a subject. *Inverse dynamics* then uses joint angles, angular velocities, and angular accelerations of the model, together with the experimental ground reaction forces and moments, to solve for the net reaction forces and net moments at each of the joints. The schematic below shows an overview of the inverse kinematics and inverse dynamics problems.



In this tutorial, you will:

- Become familiar with OpenSim's Scale, Inverse Kinematics and Inverse Dynamics tools
- Solve an inverse kinematics and inverse dynamics problem using experimental data
- Interpret the results of the inverse dynamics solution
- Investigate the dynamic inconsistencies that arise during inverse dynamics

Format

Each section of the tutorial guides you in using certain tools within and asks you to answer a few questions. The menu titles and option names you must select and any commands you must type to run OpenSim will appear in **bold face**. The questions can be answered based on information from OpenSim and basic knowledge of the human musculoskeletal system. After you complete the tutorial, feel free to explore OpenSim and the other analysis tools further on your own. Depending on the amount of exploration you do, this tutorial should take 1-2 hours to complete.

II. Generic Musculoskeletal Model

In this tutorial, you will be using a generic musculoskeletal model with 23 degrees of freedom and actuated by 54 muscles entitled *3DGaitModel23 54*. It is a simplified version of the lower-extremity model of Delp et al. [1], modified to include a torso and back joint based on the model of Anderson and Pandy [2].

To load the generic musculoskeletal model into OpenSim:

- Click the **File** menu and select **Open Model**.
- Find the ***Models*** folder, which is located under your OpenSim installation directory, e.g., *C:\OpenSim3.2. (The actual directory name*

depends on version used and can be changed during installation).

- Open the ***Gait2354_Simbody*** folder, select the file ***gait2354_simbody.osim***, and click **Open**.

The experimental gait data were collected by Jill Higginson and Chand John in the Neuromuscular Biomechanics Lab at the University of Delaware as part of the study cited below. The data include marker trajectories and ground reaction forces for an adult male walking at a self-selected speed on an instrumented split-belt treadmill. Please note that the data distributed with OpenSim is from a different subject than the one described in the paper. Data collection protocols were the same for both subjects.

Chand T. John, Frank C. Anderson, Jill S. Higginson & Scott L. Delp (2012): Stabilisation of walking by intrinsic muscle properties revealed in a three-dimensional muscle-driven simulation, Computer Methods in Biomechanics and Biomedical Engineering.

III. Scaling A Musculoskeletal Model

The purpose of scaling a generic musculoskeletal model is to modify the anthropometry, or physical dimensions, of the generic model so that it matches the anthropometry of a particular subject. Scaling is one of the most important steps in solving inverse kinematics and inverse dynamics problems because these solutions are sensitive to the accuracy of the scaling step. In OpenSim, the scaling step adjusts both the mass properties (mass and inertia tensor), as well as the dimensions of the body segments. Scaling can be performed using a combination of two methods:

> *(1) Measurement-based Scaling*: This type of scaling determines scale factors for a body segment by comparing distance measurements between specified landmarks on the model, known as *virtual markers*, and the corresponding *experimental marker* positions.
>
> *(2) Manual Scaling*: This type of scaling allows the user to scale a segment based on some predetermined scale factor. Manual scaling is sometimes necessary when suitable marker data are not available, or if the scale factors were determined using an alternative algorithm.

To scale the generic model:

- Click the **Tools** menu and select **Scale Model**.
- At the bottom of the *Scale Tool* dialog, click **Load** to input a settings file.
- In the file browser, ensure that you are in the ***Gait2354_Simbody*** folder, select the file ***subject01_Setup_Scale.xml*** and click **Open**.

This *xml* file contains pre-configured settings to scale the generic musculoskeletal model to the dimensions of the subject. Notice all of the textboxes in the dialog were filled in appropriately.

Questions

1. Based on information in the Scale Tool dialog, what is the mass of the generic musculoskeletal model? What was the mass of the subject?

2. To see the loaded scale factors, click on the **Scale Factors** tab. *Which body segments were scaled manually?*

3. *Based on information in the Inverse Kinematics Tool dialog, at what frequency was the experimental motion data captured? Hint: Look for the box titled Marker Data.*

To complete the scale step:

- In the *Scale Tool* dialog, click **Run**. Then click **Close**.
- To save the scaled model, click **File** and select **Save Model.**
- Ensure that you are in the ***Gait2354_Simbody*** folder, type ***gait2354_scaled.osim*** into the **File name** textbox, and click **Save**.

When complete, a new, scaled model entitled *subject01* will appear in *View* window. Notice the blue spheres around the new model. These blue spheres graphically represent the *experimental markers* from the motion capture data used in the measurement-based scaling.

At this point, you may close the generic model (right-click the model name in the Navigator window, and select "close") or hide the model (right-click the model name, and select Display >> Hide).

IV. Inverse Kinematics

Kinematics is the study of motion without considering the forces and moments that produce that motion. Thus, to perform kinematical analyses, such as inverse kinematics, mass and inertia properties are not needed. The purpose of inverse kinematics is to find the joint angles of the model that best reproduce the experimental kinematics of a particular subject. In this tutorial, the experimental kinematics used by the inverse kinematics tool are based on *experimental marker* positions*.*

The inverse kinematics tool goes through each time step, or frame, of recorded motion and computes the set of joint angles that put the model in a configuration that "best match" the experimental kinematics. OpenSim determines this "best match" by solving a weighted least squares optimization problem with the goal of minimizing *marker error*. For more information about optimization and least-squares problems, see Chapter 1 of *Convex Optimization* by Stephen Boyd and Lieven Vandenberghe (http://www.stanford.edu/~boyd/cvxbook/).

Marker error is defined as the distance between an *experimental marker* and the corresponding *virtual marker*. Each marker has an associated weighting value, specifying how strongly that marker's error term should be minimized in the least squares problem. In each frame, the inverse kinematics tool solves for a vector of generalized coordinates (e.g., joint angles), **q**, that minimizes the weighted sum of marker errors, which is expressed as,

$$\min_{q}\left[\sum_{i\in\text{markers}} w_i \left\| \mathbf{x}_i^{\text{exp}} - \mathbf{x}_i(\mathbf{q}) \right\|^2\right]$$

where **q** is the vector of generalized coordinates (e.g., joint angles), $\mathbf{x}_i^{\mathbf{exp}}$ is the position of *experimental marker i*, $\mathbf{x}_i(\mathbf{q})$ is the position of the corresponding *virtual marker i* (which depends on **q**), and w_i is the weight associated with marker *i*.

To solve the inverse kinematics problem:

- Click the **Tools** menu and select **Inverse Kinematics**.
- In the *Inverse Kinematics Tool* dialog, click **Load** to load an Inverse Kinematics setup file.
- In the file browser, ensure that you are in the ***Gait2354_Simbody*** folder, select the file ***subject01_Setup_IK.xml*** and click **Open**.

This *xml* file contains pre-configured settings to solve the inverse kinematics problem for the scaled model. Notice all of the textboxes in the dialog were filled in appropriately.

Questions

4. Click the **Weights** tab and scroll through the list of markers in the top half of the dialog. *Which markers have weighting values less than one? Why?*
Hint: Think about joints that have not been modeled.

To complete inverse kinematics step:

- In the *Inverse Kinematics Tool*, click **Run**. Then click **Close**.
  Note: Even though you closed the dialog, the Inverse Kinematics tool is still running. Notice the progress bar in the lower right-hand corner of the program. ***Wait until the bar disappears before proceeding.***

The model will begin to move slowly, as the inverse kinematics problem is being solved for each frame of the experimental data.

To compare experimental marker data with inverse kinematics results, in the *Navigator* panel, go to **Motions** and right-click on **Results** (which are what the Inverse Kinematics Tool just generated). Then choose **Associate Motion Data...** from the drop down menu. Choose *subject_01_walk1.trc* and click **Open**. Model markers are *virtual markers* shown in pink and *experimental markers* are shown in blue. Hit play in the Motion Toolbar. The *virtual markers* should correspond closely to the *experimental marker* locations as the animation proceeds.
*Note: If using a Virtual Machine on a Mac, Command + Ctrl + Left Click on each motion.

When completed, examine the accuracy of the inverse kinematics solution:

- Click the **Window** menu and select **Messages**.
- The *Messages* window records details of the all steps you have performed. Take a minute to explore the Messages window. Then, **scroll** to the very bottom.
- The next to last line provides the markers errors and model coordinate errors (e.g., joint angle errors) associated with the last frame of the motion.
  Note: All marker errors have units in meters, and all coordinate errors have units in radians.

Questions

5. *Based on information in the Messages window, what is the root-mean-squared (RMS) error of all the markers in the last frame of the motion?*

Include units. Does this seem reasonable? Explain.

6. *What was the value of the maximum marker error in the last frame? Include units. Which marker had this maximum error, and why?*
Hint: Think about the weighted least squares problem.

To visualize the inverse kinematics solution, animate the model by using the *motion slider* and *video controls*. The model should walk through one full gait cycle.
Remember: *You can **loop***
🔁
and control the speed of the animation.

The inverse kinematics solution is saved to *subject01_walk1_ik.mot* as specified in the setup file.
Note: Be sure to use the exact file name given, as this file is used to complete the next step.

V. Inverse Dynamics

Dynamics is the study of motion *and* the forces and moments that produce that motion. Thus, to perform dynamical analyses, such as inverse dynamics, estimation of mass and inertia is required. The purpose of inverse dynamics is to estimate the forces and moments that cause a particular motion, and its results can be used to infer how muscles are utilized for that motion. To determine these forces and moments, equations of motion for the system are solved iteratively [3]. The equations of motion are derived using the kinematic description and mass properties of a musculoskeletal model. Then, using the joint angles from inverse kinematics and experimental ground reaction force data, the net reaction forces and net moments at each of the joints are calculated such that the dynamic equilibrium conditions and boundary conditions are satisfied [3].

Note: Joint reaction, or inter-segmental, force is the total force acting across a particular joint in a model. This should not be confused with joint bone-on-bone force, which is the actual force seen across the articulating surfaces of the joint and include the effect of muscle activity. For a thorough discussion on this topic see pp 77-79 in [4].

To solve the inverse dynamics problem:

- Click the **Tools** menu and select **Inverse Dynamics**.
- In the *Inverse Dynamics Tool* dialog, click **Load** to load an Inverse Dynamics setup file.
- In the file browser, ensure that you are in the ***Gait2354_Simbody*** folder, select the file ***subject01_Setup_InverseDynamics.xml*** and click **Open**.
  *Note: If the **Motion From File** textbox appears red, this means the textbox was filled with an inappropriate file name. Make sure the motion file was saved with the correct file name in the Inverse Kinematics section.*
- Note the folder listed in the **Directory** textbox, located in the **Output** section of the dialog. The storage file containing the inverse dynamics results will be saved in this folder: *examples\Gait2354_Simbody\ResultsInverseDynamics*
- Click **Run** at the bottom of the dialog. Then click **Close**.

When completed, examine the results of the inverse dynamics solution by plotting the net moments at the left and right ankles:

- Click **Tools** and select **Plot**.
- In the *Plotter* window, click the **Y-Quantity** button and select **Load File**.
- In the file browser, go to the *ResultsInverseDynamics* folder, select the file ***inverse_dynamics.sto***, and click **Open**.
- In the menu, select **ankle_angle_r_moment** and **ankle_angle_l_moment** by clicking the corresponding checkboxes, then click **OK**.
  Note: To quickly find these quantities, type ankle into the pattern text box.
- Click the **X-Quantity** button, select **time**, and click **OK**.
- Back in the *Plotter* window, click **Add** to add the moment curves to the plot.
- Print your plot by **right clicking** on the plot and selecting **Print**.
  *Note: To export the plot as an image by **right-clicking** the plot and selecting **Export Image**.*
- After printing the plot and answering the following questions, **close** the *Plotter* window.

Questions

7. *On your plot of the ankle moments, identify when heel strike, stance phase, toe off, and swing phase occur for each curve (i.e., left leg and right leg).*

8. *Based on your plot and the angle convention for the ankle, give an explanation of what is happening at the ankle just before toe-off.*
Hint: It may be useful to use the Coordinate sliders to understand the angle convention for the ankle.

In solving the inverse dynamics problem, both kinematical data *and* force plate data were used, making this an over-determined problem. In other words, the problem has more equations than unknowns (i.e., degrees of freedom). Due to errors in the experimental motion data and inaccuracies in the musculoskeletal model, it turns out that Newton's second law is violated, or

$$\vec{F}_{\text{exp}} \neq m \cdot \vec{a}$$

[3]. One method to handle this dynamical inconsistency is to compute and apply residual forces and moments to a particular body segment in the model, such that Newton's second law becomes:

$$\vec{F}_{\text{exp}} + \vec{F}_{residual} = m \cdot \vec{a}^{\dagger}$$

† An analogous equation relates the ground reaction moment,

$\vec{M}_{\text{exp}}$

to the residual moment,

$\vec{M}_{residual}$.

.

In this musculoskeletal model, the residuals are applied to the pelvis segment.

To see the residuals from the inverse dynamics solution:

- Click **Tools** and select **Plot**.
- In the Plotter window, click the **Y-Quantity** button and select **Load File**.
- In the file browser, go to the *ResultsInverseDynamics* folder, select the file ***inverse_dynamics.sto***, and click **Open**.
- In the menu, select **pelvis_tx_force**, **pelvis_ty_force**, and **pelvis_tz_force** by clicking the corresponding **checkboxes**, then click **OK**.
- Click the **X-Quantity** button, select **time**, and click **OK**.
- Click **Add** to add the curves to the plot.

Questions

9. *What are the maximum magnitudes of the residual forces? Using the mass of the subject from Question 1, what fraction of body weight are the maximum residual forces?*

While computing and applying residual forces and moments makes the model's motion dynamically consistent with the external forces (i.e., $\vec{F} = m \cdot \vec{a}$), this strategy is undesirable because the residuals can be large. More advanced strategies have been developed to deal with the problem of residuals and dynamic inconsistencies, such as least-squares optimization [3], the Residual Elimination Algorithm (REA) [5], and the Residual Reduction Algorithm (RRA) [6]. Additionally, OpenSim implements a Residual Reduction Algorithm as part of its workflow for generating muscle-actuated simulations [7]. Reference [6] is attached at the end of this tutorial. For additional information on these strategies, please also refer to [3], [5], and [7].

References

1. Delp, S.L., Loan, J.P., Hoy, M.G., Zajac, F.E., Topp E.L., Rosen, J.M. An interactive graphics-based model of the lower extremity to study orthopaedic surgical procedures. *IEEE Transactions on Biomedical Engineering*, vol. 37, pp. 757-767, 1990.

2. Anderson, F.C., Pandy, M.G. A dynamic optimization solution for vertical jumping in three dimensions. *Computer Methods in Biomechanical and Biomedical Engineering*, vol. 2, pp. 201-231, 1999.

3. Kuo, A.D. A least squares estimation approach to improving the precision of inverse dynamics computations, *Journal of Biomechanical Engineering*, vol. 120, pp. 148-159, 1998.

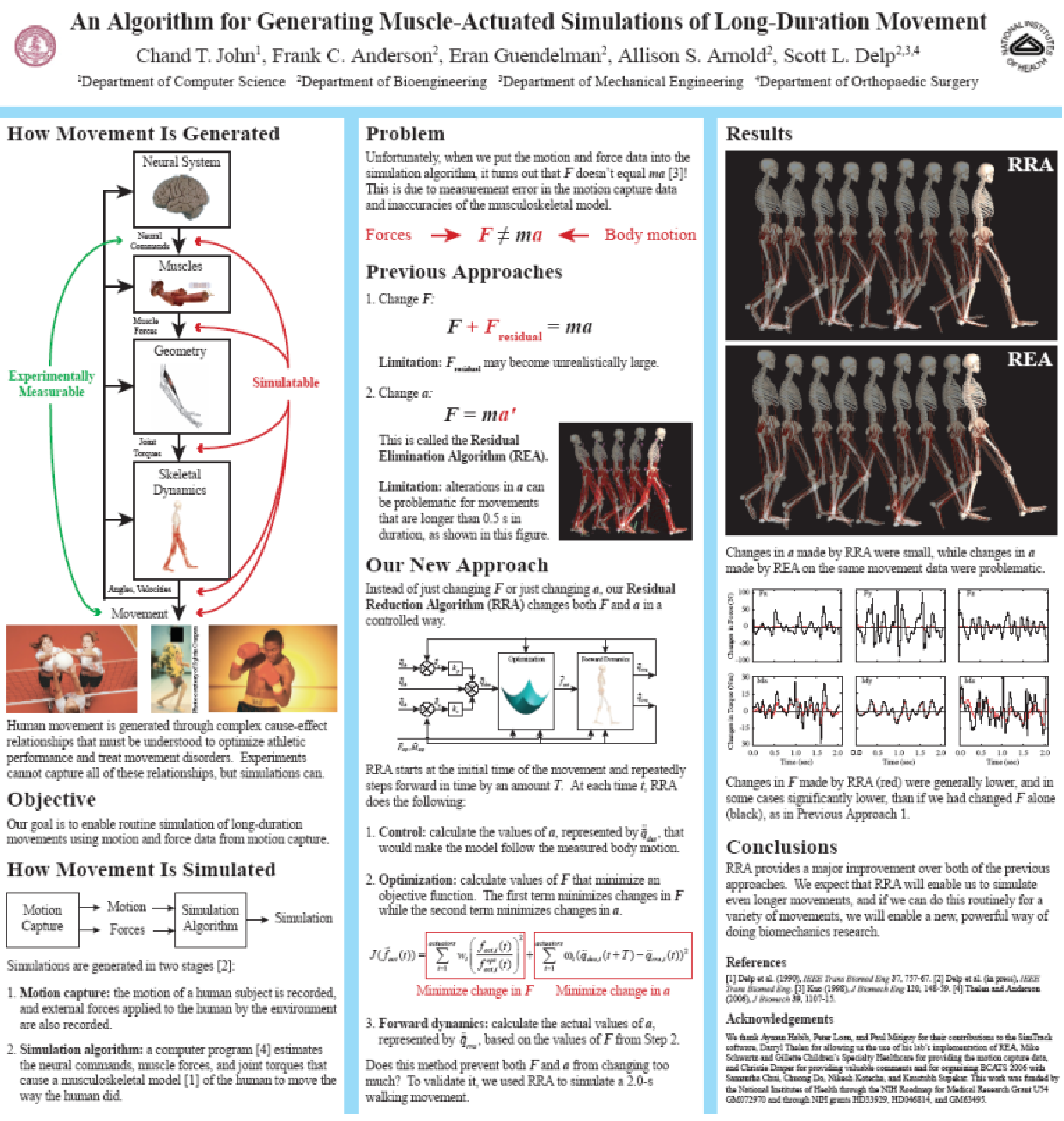
4. Winter, D.A. *Biomechanics and Motor Control of Human Movement*, Wiley and Sons, pp. 77-79, 1990.

5. Thelen, D.G., Anderson, F.C. Using computed muscle control to generate forward dynamic simulations of human walking from experimental data, *Journal of Biomechanics*, vol. 39, pp. 1107-1115, 2006.

6. John, C.T., Anderson, F.C., Guendelman, E., Arnold, A.S., Delp, S.L. An algorithm for generating muscle-actuated simulations of long-duration movements, *Biomedical Computation at Stanford (BCATS) Symposium*, Stanford University, 21 October 2006, Poster Presentation.

7. Delp, S.L., Anderson, F.C., Arnold, A.S., Loan, P., Habib, A., John, C.T., Guendelman, E., Thelen, D.G. OpenSim: Open-source software to

create and analyze dynamic simulations of movement. *IEEE Transactions on Biomedical Engineering*, vol. 55, pp. 1940-1950, 2007.



An Algorithm for Generating Muscle-Actuated Simulations of Long-Duration Movement

Chand T. John[1], Frank C. Anderson[2], Eran Guendelman[2], Allison S. Arnold[2], Scott L. Delp[2,3,4]

[1]Department of Computer Science [2]Department of Bioengineering [3]Department of Mechanical Engineering [4]Department of Orthopaedic Surgery

How Movement Is Generated

Neural System
Neural Commands
Muscles
Muscle Forces
Geometry
Joint Torques
Skeletal Dynamics
Angles, Velocities
Movement
Experimentally Measurable
Simulatable

[redacted]

Human movement is generated through complex cause-effect relationships that must be understood to optimize athletic performance and treat movement disorders. Experiments cannot capture all of these relationships, but simulations can.

Objective

Our goal is to enable routine simulation of long-duration movements using motion and force data from motion capture.

How Movement Is Simulated

Simulations are generated in two stages [2]:

1. **Motion capture:** the motion of a human subject is recorded, and external forces applied to the human by the environment are also recorded.
2. **Simulation algorithm:** a computer program [4] estimates the neural commands, muscle forces, and joint torques that cause a musculoskeletal model [1] of the human to move the way the human did.

Problem

Unfortunately, when we put the motion and force data into the simulation algorithm, it turns out that *F* doesn't equal *ma* [3]! This is due to measurement error in the motion capture data and inaccuracies of the musculoskeletal model.

$$\text{Forces} \rightarrow F \neq ma \leftarrow \text{Body motion}$$

Previous Approaches

1. Change *F*:

$$F + F_{\text{residual}} = ma$$

Limitation: F_{residual} may become unrealistically large.

2. Change *a*:

$$F = ma'$$

This is called the **Residual Elimination Algorithm (REA).**

Limitation: alterations in *a* can be problematic for movements that are longer than 0.5 s in duration, as shown in this figure.

Our New Approach

Instead of just changing *F* or just changing *a*, our **Residual Reduction Algorithm (RRA)** changes both *F* and *a* in a controlled way.

RRA starts at the initial time of the movement and repeatedly steps forward in time by an amount *T*. At each time *t*, RRA does the following:

1. **Control:** calculate the values of *a*, represented by $\ddot{q}_{des}$, that would make the model follow the measured body motion.
2. **Optimization:** calculate values of *F* that minimize an objective function. The first term minimizes changes in *F* while the second term minimizes changes in *a*.

$$J(\vec{f}_{act}(t)) = \boxed{\sum_{i=1}^{actuators} w_i \left(\frac{f_{act,i}(t)}{f_{act,i}^{opt}(t)}\right)^2} + \boxed{\sum_{j=1}^{actuators} \omega_j\left(\ddot{q}_{des,j}(t+T) - \ddot{q}_{sim,j}(t)\right)^2}$$

Minimize change in *F* Minimize change in *a*

3. **Forward dynamics:** calculate the actual values of *a*, represented by $\ddot{q}_{sim}$, based on the values of *F* from Step 2.

Does this method prevent both *F* and *a* from changing too much? To validate it, we used RRA to simulate a 2.0-s walking movement.

Results

Changes in *a* made by RRA were small, while changes in *a* made by REA on the same movement data were problematic.

Changes in *F* made by RRA (red) were generally lower, and in some cases significantly lower, than if we had changed *F* alone (black), as in Previous Approach 1.

Conclusions

RRA provides a major improvement over both of the previous approaches. We expect that RRA will enable us to simulate even longer movements, and if we can do this routinely for a variety of movements, we will enable a new, powerful way of doing biomechanics research.

References

[1] Delp et al. (1990), *IEEE Trans Biomed Eng* 37, 757-67. [2] Delp et al. (in press), *IEEE Trans Biomed Eng*. [3] Kuo (1998), *J Biomech Eng* 120, 148-59. [4] Thelen and Anderson (2006), *J Biomech* 39, 1107-15.

Acknowledgements

We thank Ayman Habib, Peter Loan, and Paul Mitiguy for their contributions to the SimTrack software, Darryl Thelen for allowing us the use of his lab's implementation of REA, Mike Schwartz and Gillette Children's Specialty Healthcare for providing the motion capture data, and Christine Draper for providing valuable comments and for organizing BCATS 2006 with Samantha Chui, Chuong Do, Nikesh Kotecha, and Kaustubh Supekar. This work was funded by the National Institutes of Health through the NIH Roadmap for Medical Research Grant U54 GM072970 and through NIH grants HD33929, HD046814, and GM63495.