SimTK Documents

OpenSim and NMS Physiome Tutorial

June 30, 2011, ISB Technical Group on Computer Simulation, Leuven, Belgium

Websites: SimTK.org/home/opensim, opensim.stanford.edu, and biomedtown.org

Acknowledgments

OpenSim was developed as a part of SimTK and funded by the Simbios National Center for Biomedical Computing and the National Center for Simulation in Rehabilitation Research through the National Institutes of Health and the NIH Roadmap for Medical Research, Grant U54 GM072970. Information on the National Centers can be found at http://nihroadmap.nih.gov/bioinformatics.

Trademarks and Copyright and Permission Notice

Table of Contents

1 Introduction

1.1 Tutorial Overview

This tutorial provides an introduction to some of the many tools available within OpenSim, a freely available software package for musculoskeletal modeling and dynamic simulation of movement. The tutorial will also include an overview of the recent progress and future plans of the NMS Physiome project.

In the morning session we will focus on generating muscle activations for a forward dynamic simulation of the swing phase of gait. We will try to generate these muscle activations ourselves, and then compare our results to muscle activations computed with the Static Optimization and Computed Muscle Control Tools. We will also hear about the latest developments on the NMS Physiome Project. In the afternoon, we will learn about the Residual Reduction Algorithm and then you will break into teams to generate and analyze a muscle-driven simulation of the stance phase of gait.

This handout includes information about using the OpenSim workflow to generate and analyze muscle driven simulations of motion. Please use it as a resource throughout the tutorial and in the future when using OpenSim for your own research. The handout provides a brief overview of the underlying theory, inputs, outputs, best practices, and troubleshooting tips for each of the tools used in the workflow, including Scale, Inverse Kinematics, Inverse Dynamics, Static Optimization, Residual Reduction, Computed Muscle Control, and Forward Dynamics. We have also included a chapter about conducting a Joint Reactions Analysis in OpenSim.

This tutorial will involve lots of hands-on practice using OpenSim. You will receive a separate handout with information about the hands-on portion of the tutorial and the afternoon team project.

1.2 Where to Find Additional Resources and Support

There are many resources available to help you troubleshoot, locate models and simulation data, and interact with the rest of the OpenSim community.

1.2.1 OpenSim User's Guide

The User's Guide is an extensive resource for most of the features available in OpenSim. For assistance using the GUI and learning about setting up Tools in OpenSim, please refer to the OpenSim User's Guide that is available under "OpenSim Documentation" at https://simtk.org/docman/?group_id=91.

1.2.2 Model and Simulation Repository

You can create your own models of musculoskeletal structures and dynamic simulations of movement in OpenSim, as well as take advantage of computer models and dynamic simulations that other users have developed and shared. For example, you can use existing computer models of the human lower limb, upper limb, cervical spine, and whole body, which have already been developed and posted at https://simtk.org/home/nmblmodels. You can also use dynamic simulations of walking and other activities that have been developed, tested, and posted on SimTK.org. We encourage you to share your models and simulations with the research community by setting up a project on SimTK.org.

1.2.3 Online Resources

There are many resources available online for support both during and after the workshop. This includes the User's Guide and model repository described above, as well as a user forum, tutorials, examples, webinars, and many other resources. We've recently launched a new website that serves as a portal to these resources. It can be found at http://opensim.stanford.edu. You can also find more about OpenSim and related projects by other researchers around that world at the SimTK.org project site at http://simtk.org/home/opensim.

1.2.4 Publications

You can find additional information in the following article: Delp, S.L., Anderson, F.C., Arnold, A. S., Loan, P., Habib, A., John, C., Guendelman, E.G., Thelen, D.G., OpenSim: Open-source software to create and analyze dynamic simulations of movement. *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 11, pp. 1940-1950, 2007. Please cite this work in any of your own publications that use OpenSim.

2 Scaling

2.1 How it Works

The Scale Tool alters the anthropometry of a model so that it matches a particular subject as closely as possible. Scaling is typically performed based on a comparison of experimental marker data with virtual markers placed on a model. In addition to scaling a model, the scale tool can be used to adjust the locations of virtual markers so that they better match the experimental data.

In this chapter, we provide a conceptual review of the inputs and outputs of the Scale tool and a set of troubleshooting tips and best practices for scaling. The OpenSim User's Guide provides detailed information and step-by-step instructions on scaling a model (Chapter 14). Carefully scaling your model to match your subject is essential for getting good results from later tools, like Inverse Kinematics and Inverse Dynamics.

2.2 Scale Tool

The Scale Tool is accessed by selecting **Tools → Scale Model...** from the OpenSim main menu bar. Like all tools, the operations performed by the Scale Tool apply to the current model.

2.2.1 Input

The modelName_Setup_Scale.xml file is the setup file for the Scale Tool. Note that all filenames given are examples. You may use different naming conventions, if desired. The setup file contains settings, which help describe the model, data, and parameters for scaling. These include:

1. modelName.osim: A generic, unscaled model.
2. modelName_Scale_MarkerSet.xml: A virtual marker set that corresponds to your experimental marker set.
3. subject_static.trc: The experimental marker data of your subject in a static pose and the time range when the static pose was collected. The static pose should include the subject wearing the full marker set. The marker trajectories are specified in global frame.

4. modelName_Scale_MeasurementSet.xml: The measurement set for the Scale Tool, which contains pairs of experimental markers, the distance between which are used to scale the generic musculoskeletal model.
   AND/OR
   modelName_Scale_ScaleSet.xml: Scale set for the Scale Tool. Alternately, you can use manual scale factors to scale the generic musculoskeletal model.
5. modelName_Scale_Tasks.xml: In addition to scaling the model, the Scale Tool moves the virtual markers on the model so that their positions match the experimental marker locations. To do this, the Scale Tool must position the model so that it best matches the position of the subject, which requires an inverse kinematics problem to be solved. This file contains the inverse kinematics tasks describing which virtual and experimental markers should be matched up during the inverse kinematics phase. The file also contains marker weights, which are relative and determine how "well" the virtual markers track experimental markers (i.e., a larger weight will mean less error between virtual and experimental marker positions).
6. subject_static.mot (optional): Experimental generalized coordinate values (joint angles) for a trial obtained from alternative motion capture devices or other specialized algorithms. You can specify coordinate weights in the Tasks file, if joint angles are know a priori. Coordinate weights are also relative and determine how "well" a joint angle will track the specified angle.

2.2.2 Output

1. subject.osim: OpenSim musculoskeletal model scaled to the dimensions of the subject.
2. subject_static_ik.mot: A motion file containing the joint angles for the static pose.

2.3 Best Practices and Troubleshooting

1. When collecting data, take pictures of your subjects in the static pose.
2. Have your subjects perform movements to calculate functional joint centers at the hip, knee, ankle, and/or shoulders and append the joint centers to your static trial data.
3. Measure subject specifics, like height, mass, body segment lengths, mass distribution (if DXA is available), and strength (if a Biodex is available).

4. Do not use all markers from motion-capture to position and scale the model. Markers that match anatomical landmarks and functional joint centers are the only markers that can be relied on for scaling. Avoid adjusting their model positions to match experimental positions, and use the marker position errors to assess the quality of your scaling results.
5. Some segments, like the pelvis are best scaled non-uniformly.
6. In general, maximum marker errors for bony landmarks should be <2 cm.
7. Use what you know about your subject's static pose when looking at the results of Scale. For example, in a typical static posture ankle angle is generally less than 5° and hip flexion angle is less than 10°.
8. Use the "preview static pose" option in the GUI to visualize the scaled model's anatomical marker positions relative to the corresponding experimental markers to see how well the model "fits" the data before adjusting all the markers to match the experimental data.
9. If the results of scale look incorrect, you can either change the location of virtual markers or alter marker weightings to calculate static pose.
10. Use coordinate tasks (Static Pose Weights) to set joint angles for troublesome joints (commonly the ankle joint and lumbar joint) that are very sensitive to how the markers are placed. For example if it is known that the foot is flat, an ankle angle can be provided and then the markers adjusted in order to match the known pose.
11. It is common to iterate through Scale and Inverse Kinematics to fine-tune segment dimensions and marker positions that yield low marker errors for the task of interest.
12. If using coordinates from a motion capture system make sure that the joint/coordinate definitions match otherwise you may cause more harm than good.
13. The model has a built in assumption that the global Y axis is up. If your data doesn't fit this, then consider transforming it using the Preview Motion Data option in the GUI.

3 Inverse Kinematics

3.1 How It Works

The Inverse Kinematics Tool steps through each time frame of experimental data and positions the model in a pose that "best matches" experimental marker and coordinate data for that time step. This "best match" is the pose that minimizes a sum of weighted squared errors of markers and/or coordinates.

In this chapter, we provide a conceptual review of the inputs and outputs of the Inverse Kinematics (IK) tool and a set of troubleshooting tips and best practices. The OpenSim User's Guide provides detailed information and step-by-step instructions (Chapter 15). Getting accurate results from the IK tool is essential for using later tools like Static Optimization, Residual Reduction Algorithm, and Computed Muscle Control.

3.2 Inverse Kinematics Tool

To launch the IK Tool, select **Tools** → **Inverse Kinematics** from the OpenSim main menu bar.

3.2.1 Input

The primary inputs to IK are the following files:

1. subject01.osim: A subject-specific OpenSim model generated by scaling a generic model with the Scale Tool or by other means, along with an associated marker set containing adjusted virtual markers.
2. motionTrial.trc: Experimental marker trajectories for a trial obtained from a motion capture system, along with the time range of interest
3. marker_tasks.xml: A file containing marker weightings. As in the scale tool, marker weights are relative and determine how "well" the virtual markers track experimental markers (i.e., a larger weight will mean less error between virtual and experimental marker positions).
4. subject01_coords.mot (optional): Experimental generalized coordinate values (joint angles) for a trial obtained from alternative motion capture devices or other

specialized algorithms. You can optionally specify relative coordinate weights in the Tasks file, if joint angles are known a priori.

3.2.2 Output

1. subject01_ik.mot: A motion file containing the generalized coordinate trajectories (joint angles and/or translations) computed by IK.

3.3 Best Practices and Troubleshooting

1. When collecting experimental data, place three non-collinear markers per body segment that you want to track. You need at least three markers to track the 6 DOF motion (position and orientation) of a body segment.
2. Place markers on anatomical locations with minimum skin/muscle motion.
3. Weight "motion" segment markers, for example from a triad placed on the thigh segment, more heavily than anatomical markers affixed to landmarks like the greater trochanter and the acromion, which can be helpful for scaling, but are influenced by muscle and other soft tissue movements during motion.
4. Relative marker weightings are more important than their absolute values. Therefore, a weighting of 10 vs. 1 is 10 times more important whereas 20 vs. 10 is only twice as important. Markers are not necessarily tracked better because they both have higher weightings.
5. Total RMS and max marker errors are reported in the messages window. Use these values to guide changes in weightings, or if necessary to redo marker placement and possibly scaling. Maximum marker error should generally be less than 2-4cm and RMS under 2cm is achievable.
6. Compare your results to similar data reported in the literature. Your results from an unimpaired average adult should generally be within one standard deviation.

4 Inverse Dynamics

4.1 How it Works

The Inverse Dynamics Tool steps through each time frame of a motion and computes the net forces and/or torques at each joint in the model based upon/due to the experimental kinematics. The equations of motion relate the model accelerations to the forces and/or joint torques applied to the model.

In this chapter, we provide a conceptual review of the inputs and outputs of the Inverse Dynamics tool and a set of troubleshooting tips and best practices. The OpenSim User's Guide provides detailed information and step-by-step instructions (Chapter 16).

4.2 Inverse Dynamics Tool

To launch the Inverse Dynamics Tool select **Inverse Dynamics...** from the **Tools** menu. The **Inverse Dynamics Tool** dialog, like all other OpenSim tools, operates on the Current Model open and selected in OpenSim.

4.2.1 Input

1. subject01.osim: A subject-specific OpenSim model generated by scaling a generic model with the Scale Tool or by other means, along with an associated marker set containing adjusted virtual markers. The model must include inertial parameters. Note that forces like contact, ligaments, bushings, and even muscles will be applied to the model based on the kinematic state of the model and defaults for the muscle states, unless these forces are specifically excluded in the calculation.
2. subject01_ik.mot: Coordinate motion data (i.e., joint angles or translation for a motion) and time range for the motion of interest).
3. subject01_grf.xml: External load data (i.e., ground reaction forces, moments, and center of pressure location). Note that it is necessary to measure and apply or model all external forces acting on a subject during the motion to calculate accurate joint torques and forces. This file includes the name of the ground reaction force-data file (e.g. subject01_grf.mot) as well as the names of the bodies they are applied to. Options to specify the forces, point of application, and torques in a global or body

local frame (relative to the body to which the force is being applied) are also defined here.

4.2.2 Output

1. subject01_id.sto: Joint torques and forces, acting along the coordinate axes that produce the accelerations estimated (via double differentiation) from your measured experimental motion and modeled and external forces applied.

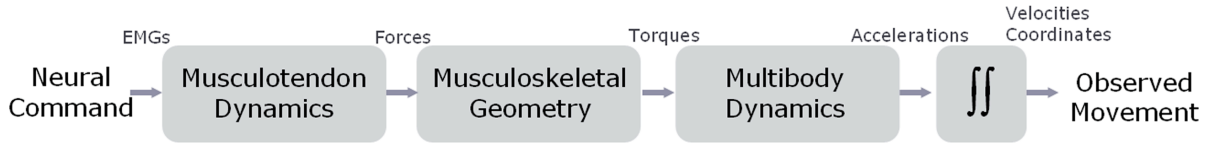
4.3 Best Practices and Troubleshooting

1. Filter your raw coordinate data, since noise is amplified by differentiation. Without filtering, the calculated forces and torques will be very noisy.
2. Compare your results to data reported in the literature. Your results should be within one s.d. of reported values.
3. Inspect results from Inverse Dynamics to check if ground reaction forces were applied correctly or not. For gait, applying ground reaction forces should help reduce the forces computed by Inverse Dynamics at the pelvis.

5 Forward Dynamics

5.1 How it Works

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.



5.1.1 Musculoskeletal Model Dynamics

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:

$$\ddot{\boldsymbol{q}} = \left[\boldsymbol{M}(\boldsymbol{q})\right]^{-1}\left\{\boldsymbol{\tau} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}\right\} \qquad \text{Multibody dynamics}$$

where $\ddot{\boldsymbol{q}}$ is the coordinate accelerations due to joint torques, $\boldsymbol{\tau}$, Coriolis and centrifugal forces, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$, as a function of coordinates, $\boldsymbol{q}$, and their velocities, $\dot{\boldsymbol{q}}$, gravity, $\boldsymbol{G}(\boldsymbol{q})$, and other forces applied to the model, $\boldsymbol{F}$, and $\left[\boldsymbol{M}(\boldsymbol{q})\right]^{-1}$ is the inverse of the mass matrix.

$$\boldsymbol{\tau}_m = \left[\boldsymbol{R}(\boldsymbol{q})\right]\boldsymbol{f}(\boldsymbol{a},\boldsymbol{l},\dot{\boldsymbol{l}}) \qquad \text{Moments due to muscle forces}$$

$$\dot{\boldsymbol{l}} = \Lambda(\boldsymbol{a},\boldsymbol{l},\boldsymbol{q},\dot{\boldsymbol{q}}) \qquad \text{Muscle contraction dynamics}$$

$$\dot{\boldsymbol{a}} = \mathrm{A}(\boldsymbol{a}, \boldsymbol{x})$$ Muscle activation dynamics

The net muscle moments, $\boldsymbol{\tau}_m$, in turn, are a result of the moment arms, $\boldsymbol{R}(\boldsymbol{q})$, multiplied by muscle forces, $\boldsymbol{f}$, which are a function of muscle activations, $\boldsymbol{a}$, and muscle fiber lengths, $\boldsymbol{l}$, and velocities, $\dot{\boldsymbol{l}}$. Muscle fiber velocities are governed by muscle contraction dynamics, Λ, which is dependent on the current muscle activations and fiber lengths as well as the coordinates and their velocities. Activation dynamics, A, describes how the activation rates, $\dot{\boldsymbol{a}}$, of the muscles respond to input neural excitations, $\boldsymbol{x}$, generally termed the model's controls. These form a set of differential equations that model *musculoskeletal dynamics*.

5.1.2 States of a Musculoskeletal Model

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.

5.1.3 Controlling a Musculoskeletal Model

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.

5.1.4 Numerical Integration of Dynamical Equations

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 5^{th}-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.

5.2 Forward Dynamics Tool

To launch the Forward Dynamics Tool select **Forward Dynamics…** from the **Tools** menu. The **Forward Dynamics Tool** dialog like all other OpenSim tools operates on the *Current Model* open and selected in OpenSim.

5.2.1 Inputs

Three items are required by the Forward Dynamics Tool:

1. subject01.osim: A subject-specific OpenSim model generated by scaling a generic model with the Scale Tool or by other means, along with an associated marker set containing adjusted virtual markers. The model must include inertial parameters (segment masses, etc.).
2. subject01_controls.xml: XML file containing the time histories of the model controls (e.g., muscle excitations) to the muscles and/or joint torques. This file may be generated by the user, Static Optimization Tool, or Computed Muscle Control Tool. If no controls are provided they are assumed to be zero for any actuators in the model.
3. subject01_states.sto: Storage file containing the initial states of the musculoskeletal model that includes coordinates and their velocities and muscle activations and muscle fiber lengths. Alternately, the simulation can begin from the pose that mode is in without providing initial states. Muscle states are estimated by solving for muscle-fiber and tendon force equilibrium.

A few *additional* items may be provided to the Forward Dynamics Tool:

1. subject01_grf.xml: External load data (i.e., ground reaction forces, moments, and center of pressure location).
2. Settings for numerical integration
3. Additionally, analyses can be added so that they are executed during a forward simulation.

5.2.2 Outputs

The Forward Dynamics Tool generates storage files containing the time histories of the model's controls and states that result from integration of the model's dynamical equations in the output directory specified.

5.3 Best Practices and Troubleshooting

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

2. 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.

6 Static Optimization

6.1 How it Works

As described in the previous chapter, the motion of the model is completely defined by the generalized positions, velocities, and accelerations. The Static Optimization Tool uses the known motion of the model to solve the equations of motion for the unknown generalized forces (e.g., joint torques) subject to one of the following muscle activation-to-force conditions:

$$\underbrace{\sum_{m=1}^{nm}\left(a_m F_m^0\right) r_{m,j} = \tau_j}_{\text{ideal force generators}} \quad \text{or} \quad \underbrace{\sum_{m=1}^{nm}\left[a_m f\left(F_m^0, l_m, v_m\right)\right] r_{m,j} = \tau_j}_{\text{constrained by force-length-velocity properties}}$$

while minimizing the objective function:

$$J = \sum_{m=1}^{nm}\left(a_m\right)^p$$

where *nm* is the number of muscles in the model; a_m is the activation level of muscle *m* at a discrete time step; F_m^0 is its maximum isometric force; l_m is its length; v_m is its shortening velocity; $f\left(F_m^0, l_m, v_m\right)$ is its force-length-velocity surface; $r_{m,j}$ is its moment arm about the j^{th} joint axis; τ_j is the generalized force acting about the j^{th} joint axis; and *p* is a user defined constant.

6.2 Static Optimization Tool

To launch the Static Optimization Tool, select **Static Optimization…** from the **Tools** menu. The *Static Optimization Tool* dialog window, like all other OpenSim tools, operates on the current model open and selected in OpenSim

6.2.1 Input

1. subject01.osim: A subject-specific OpenSim model generated by scaling a generic model with the Scale Tool or by other means, along with an associated marker set containing adjusted virtual markers. The model must include inertial parameters (segment masses, etc.).
2. subject01_ik.mot: Kinematic data (i.e., joint angles) from IK or states (i.e., joint angles AND velocities) from RRA and the time range of interest.
3. x: The exponent for the activation-based cost function, $\sum_{actuators} a^x$ to be minimized (i.e., the criteria used to solve muscle force distribution problem).
4. subject01_grf.xml: External load data (i.e., ground reaction forces, moments, and center of pressure location). Note that you must measure or model all external forces acting on a subject during the motion to calculate accurate muscle forces.

6.2.2 Output

1. subject01_actuator_forces.sto: A set of muscle and other actuator forces that produce the necessary generalized force (joint torque) for your measured motion.

6.3 Best Practices and Troubleshooting

1. You can use IK or RRA results as input kinematics. If using IK results, you usually need to filter them, either externally or using the OpenSim analyze/static optimization field. If using RRA results, you usually do not have to filter.
2. Add residual actuators to the first free joint in the model (typically the ground-pelvis joint).
3. If the actuators/muscles are weak, the optimization will take a long time to converge or never converge at all. This takes a long time. If troubleshooting a weak model and each time, optimization is slow, try reducing the parameter that defines the max number of iterations.
4. Increase the maximum control value of a residual or reserve actuator, while lowering its maximum force. This allows the optimizer to generate a large force (if necessary) to match accelerations but large control values are penalized more heavily. In static optimization, ideal actuator excitations are treated as activations in the cost function.

7 Residual Reduction Algorithm

7.1 How it Works

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.

In this chapter, we provide a conceptual review of the inputs and outputs of the Residual Reduction Algorithm (RRA) along with a set of troubleshooting tips and best practices for completing the tool. RRA is covered in detail in Chapter 19 of the OpenSim User's Guide.

7.2 Residual Reduction Algorithm Tool

The residual reduction algorithm tool is accessed by selecting **Tools → Residual Reduction Algorithm...** from the OpenSim main menu bar. Like all tools, the operations performed by the computed muscle control tool apply to the current model.

7.2.1 Input

1. subject01.osim: A subject-specific OpenSim model generated by scaling a generic model with the Scale Tool or by other means, along with an associated marker set containing adjusted virtual markers. The model must include inertial parameters.
2. subject01_ik.mot: Kinematic data (i.e., joint angles) from IK and time range.
3. subject01_RRA_tasks.xml: A tracking tasks file specifying which coordinates to track and the corresponding tracking weight (weights are relative and determine how "well" a joint angle will track the specified joint angle from IK).
4. subject01_RRA_actuators.xml: The Actuator Set specifies the residual and reserve actuators to be applied and their parameters, like maximum/minimum force and body or joint, or location, depending on the actuator type. Each degree of freedom in the model should have an ideal torque or force (reserve) actuator. This includes the 6 DOFs of the

model's base segment, which are called the residual actuators. In most circumstances, these Ideal joint actuators used to replace the muscles in the model (by checking "Replace model actuators" in the Actuators tab.

5. subject01_RRA_ControlConstraints.xml: The actuator constraints file specifying the maximum and minimum "excitation" (i.e., control signal) for each actuator. Note that the maximum/minimum force or torque generated by an ideal actuator is the product of the max/min force and max/min excitation.
6. subject01_grf.xml: External load data (i.e., ground reaction forces, moments, and center of pressure location). See Inverse Dynamics (Handout Chapter 6) for more details.
7. Integrator settings (i.e., max number of steps, max step size, min step size, and error tolerance)

7.2.2 Output

1. subject01_RRA_states.sto: Adjusted kinematics (i.e., joint angles) and corresponding model states of the simulated motion (i.e., joint angles AND velocities).
2. subject01_adjusted.osim (optional): A model with adjusted mass properties.
3. subject01_RRA_forces.sto: Actuator forces and torques (i.e., joint torques corresponding to adjusted kinematics).
4. subjct01_RRA_controls.xml: Actuator excitations (i.e., control signals needed to generate actuator forces and torques)

7.3 Best Practices and Troubleshooting

1. RMS difference in joint angle during the movement should be less than 2-5^0 (or less than 2 cm for translations).
2. Peak Residual Forces should be less than 10-20 N.
3. Compare the residual moments from RRA to the moments from Inverse Dynamics. You should see a 30-50% reduction in peak residual moments.
4. 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.
5. 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).

6. 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.
7. 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.
8. 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).

8 Computed Muscle Control

8.1 How it Works

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).

In this chapter, we provide a conceptual review of the inputs and outputs of the Computed Muscle Control (CMC) tool, along with a set of troubleshooting tips and best practices. CMC is a tool for estimating the muscle excitations that drive a subject's motion and is covered in detail in Chapter 20 of the OpenSim User's Guide.

8.2 Computed Muscle Control Tool

The computed muscle control tool is accessed by selecting **Tools → Computed Muscle Control...** from the OpenSim main menu bar. Like all tools, the operations performed by the computed muscle control tool apply to the current model.

8.2.1 Input

1. subject01.osim: A subject-specific OpenSim model generated by scaling a generic model with the Scale Tool or by other means, along with an associated marker set containing adjusted virtual markers. The model must include inertial parameters.
2. subject01_RRA_states.sto: Kinematic data (i.e., joint angles) from RRA and the time range of interest.
3. subject01_CMC_tasks.xml: A tracking tasks file specifying which coordinates to track and the corresponding tracking weight (weights are relative and determine how "well" a joint angle will track the specified joint angle from RRA).
4. subject01_CMC_actuators.xml: This includes reserve and residual actuators, as in RRA.
5. subject01_CMC_ControlConstraints.xml: The control constraints file specifying the maximum and minimum "excitation" (i.e., control signal) for each actuator. Control constraints can also be used to enforce when certain actuators are "on" or "off" and the range in which they can operate.

6. subject01_grf.xml: External load data (i.e., ground reaction forces, moments, and center of pressure location). See Inverse Dynamics (Handout Chapter 6) for more details.
7. Integrator settings (i.e., max number of steps, max step size, min step size, and error tolerance)

8.2.2 Output

1. subject01_CMC_states.sto: Model and muscle states of the simulated motion (i.e., joint angles AND velocities, muscle fiber lengths AND activations).
2. subjct01_CMC_controls.xml: Actuator (e.g., muscle) excitations (i.e., control signals needed to generate muscle forces and reserve/residual forces and torques).
3. subject01_CMC_forces.sto: Muscle forces and reserve/residual forces and torques .

8.3 Best Practices and Troubleshooting

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.

9 Analyzing a Simulation: Estimating Joint Loads

9.1 Overview

JointReaction is an OpenSim Analysis for calculating the joint forces and moments transferred between consecutive bodies in a model. These forces and moments correspond to the internal loads carried by the joint structure. These loads represent the contributions of all un-modeled joint structures that would produce the desired joint kinematics, such as cartilage contact and any omitted ligaments. The reaction load acts at the joint center (mobilizer frame) of both the parent and child bodies. The loads 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. For more detailed description of the method implementation see *Tibiofemoral Contact Force during Crouch Gait* (Steele et al, in review).

In this chapter, we provide a conceptual review of the inputs and outputs for a JointReaction Analysis along with a set of troubleshooting tips and best practices. Running an Analysis like JointReaction is covered in detail in Chapter 21 of the OpenSim User's Guide.

9.2 Performing a JointReaction Analysis

The Analyze Tool is accessed by selecting **Tools → Analyze...** from the OpenSim main menu bar. Like all tools, the operations performed by the Analyze Tool apply to the current model.

9.2.1 Input

Inputs Specific to JointReaction:

1. joint_names: List of the names of the joints of interest. JointReaction reports loads for only listed joints that exist in the model. Joint names may be repeated any number of times to allow reporting on different bodies or with respect to different reference frames. Using the keyword 'all' reports the loads for all joints in the model. Default is 'all'.

2. apply_on_bodies: List of the body (parent or child) on which the corresponding reaction occurred. If the array has only one entry, that selection is applied to all joints specified in joint_names. The default is 'child'.
3. express_in_frame: List of the frames (ground, parent, or child) in which the corresponding reaction is expressed. If the array has only one entry, that selection is applied to all joints specified in joint_names.
4. forces_file: The name of a file containing forces storage. If a file name is provided, the applied forces for all actuators will be constructed from the forces_file instead of from the states. This option should be used to calculate joint loads from static optimization results.

9.2.2 Output

JointReaction prints results to one storage file with the suffix "_ReactionLoads.sto". This file contains rows of time-stamped data containing the 3 force and 3 moment vector components of the reaction load at each joint specified. Each data column label includes all information about how the load is applied and expressed. Specifically, the form is

< joint name >_on_<body>_in_<frame>_<component>.

For example, the column containing the z-component of hip force occurring on the femur and expressed in the femur frame would be labeled "hip_on_femur_in_femur_FX" while the y-component of the knee moment occurring on the tibia and expressed in the ground frame would have the label "knee_on_tibia_in_ground_MY".

9.3 Best Practices and Troubleshooting

Input error: If JointReaction doesn't recognize a specified joint, body, or frame name, it will perform its default action. The overall default action is to report the loads on all joints as applied to the joint's child body and expressed in the ground reference frame.

Consistency of modeling inputs to the Analyze tool: The validity of the joint loads depends on modeling assumptions and correct modeling practices. JointReaction is very sensitive to the consistency of all inputs that define a dynamic trial, including the following inputs to the Analyze Tool: <model_file>, <replace_force_set>, <force_set_files>, <states_file>, <coordinates_file>, <speeds_file>, <lowpass_cutoff_frequency_for_coordinates>, <external_loads_file>, <external_loads_model_kinematics_file>, <lowpass_cuttoff_frequency_for_load_kinematics>

If any of these files are not associated with the same dynamic trial, the system of accelerations and forces will not be consistent. Therefore, JointReaction will calculate incorrect joint loads.

Special Types of Forces: Certain types of forces and actuators, called SpringGeneralizedForce, CoordinateLimitForce, and CoordinateActuator, are associated with specific degrees of freedom and treated as part of the joint structure. This means that any contribution from these components will be treated as part of the resultant loads reported by JointReaction instead of body forces. As an example, consider a reserve actuator on the hip joint of a gait model. If this reserve actuator is defined as a CoordinateActuator, its contribution is treated as a residual force provided by the joint and therefore will be added to the resultant load at the hip. However, if the hip reserve is defined as a TorqueActuator, its torque will be treated as a motor external to the joint and attached between the pelvis and femur. Therefore, its torque will not be added to the reported resultant load at the hip.

10 NMS Physiome Project

10.1 Overview

NMS Physiome: Personalised models of the neuro-musculo-skeletal system

Nearly four million osteoporotic bone fractures cost the European health system more than 30 billion Euros per year. This figure could double by 2050. The traditional reductionist approach is reaching dead ends in a number of relevant questions in musculoskeletal research, and a major challenge for the international research community is to develop a Personalised, Predictive and Integrative (PPI) approach to musculoskeletal medicine.

Through the Neuro-Musculo-Skeletal Physiome (NMS Physiome) project, the two largest international research projects developing technology for PPI musculoskeletal medicine, the European VPHOP and the United States SIMBIOS, aim to develop synergies in terms of tools, infrastructures and research activities related to this grand challenge of biomedical research. The collaboration is funded as part of the European Union's Virtual Physiological Human initiative.

10.2 Context

The musculoskeletal system is probably the organ system where the need for the integrative approach advocated by the Virtual Physiological Human (VPH) initiative is most pronounced. Neuromotor control involves the entire body, whereas the processes involved in muscle twitching, in bone and muscle adaptation, and in musculoskeletal ageing and most musculoskeletal diseases, take place at the molecular level.

The traditional reductionist approach is reaching dead ends in a number of relevant questions in musculoskeletal research, such as those related to osteoporotic fractures, the pathophysiology of growth in cerebral palsy children, the pathogenesis of rheumatoid arthritis and osteoarthritis. It is becoming more and more evident that the way out is to develop new approaches within Information and Communication Technology (ICT). Those solutions will make possible a personalised, predictive and integrative musculoskeletal medicine.

10.3 Project

Worldwide, two of the largest research projects focused on this topic are the Osteoporotic Virtual Physiological Human (VPHOP) integrated project funded by the European Commission, and the Center for Physics-based Simulation of Biological Structures (SIMBIOS) funded by the US National Institute of Health. These two projects have similar strategic objectives and are developing highly complementary technologies.
This unique situation creates a compelling opportunity for international collaboration, which will dramatically increase the international impact of the work being done by these projects and will produce the conditions for global cooperation on this grand challenge of biomedical research.

10.4 Consortium

VPHOP, formed by a consortium of 20 partner institutions led by the **Rizzoli Orthopaedic Institute**, is developing the next generation of health technologies to fight osteoporosis. As part of this endeavour, personalised modelling of the patient's neuro-musculo-skeletal system is essential.

SIMBIOS, led by **Stanford University**, provides infrastructures, software and training to help biomedical researchers understanding biological form and function as they create novel drugs, synthetic tissues, medical devices, and surgical interventions. In particular, the team focuses on the accurate modelling and simulation of the neuro-musculo-skeletal system.

10.5 Objectives

With **NMS Physiome**, the SIMBIOS and VPHOP consortia intend to establish a more organic cooperation, structured around three objectives:

Integrate the projects' communities: work towards an integration of the two community web sites, *Biomed Town* (www.biomedtown.org) and *Simtk.org* (www.simtk.org), by developing mechanisms to share news and events, providing a global search for data models and tools across the two repositories, and exploring opportunities for the exchange, co-development, and sharing of team science technologies; develop the

activity consistent with other larger integration initiatives such as the *Biositemaps* of the US National Centers for Biomedical Comupting.

Integrate the projects' tools: work towards the complete integration of neuromuscular software tools being developed by the two consortia, including MAF(www.biomedtown.org/biomed_town/MAF), OpenSim(https://simtk.org/home/opensim) and FEBio(https://simtk.org/home/febio) ensure interoperability among software, data formats, ontologies, etc. When fully integrated, the neuromuscular software tools being developed by the two consortia will form the first complete suite for personalised, predictive and integrative musculoskeletal medicine and related research. It will be based entirely on open access software and thus readily available to the global research community.

Work together on grand challenges: combine the latest research achievements of the two consortia to better face the grand challenges this problem poses. Among the challenges we would like to tackle, we want to highlight: the efficient multiscale modelling of the musculoskeletal system, the creation of accurate patient specific models from clinically available data, and the development of modelling methods to cope with the probabilistic nature of the neuromotor system.

10.6 Method

With respect to core research, both VPHOP and SIMBIOS share the same long-term goal: **Predictive, Personalised, and Integrative musculoskeletal medicine**, *PPI medicine* for short.

Predictive means that we should answer clinical questions not with indirect statistical evidence, but with direct predictions made using biomedical-based models.
Personalised means developing predictive models that are designed to maximise the likelihood of the prediction for a particular patient being based on the available patient data.
Integrative means that, whenever necessary, sub-models describing processes occurring at different temporal or dimensional scales should be composed into a hypermodel that describes all systemic interactions across scales (body, organ, tissue, cell, and molecule).

Project co-ordinator:
Rizzoli Orthopaedic Institute

Contact person:
Marco Viceconti
Tel: +39-051-6366865
Fax: +39-051-6366863
Email: viceconti@tecno.ior.it
Website: http://www.nmsphysiome.eu