I downloaded OpenSim: now what?

A gentle introduction to OpenSim

Analysis and modeling of locomotion

Dimitar Stanev, Andrea Di Russo, Alice Bruel and Auke Ijspeert

Acknowledgments

OpenSim was developed as a part of SimTK and funded by the Simbios National Center for Biomedical Computing through the National Institutes of Health and the NIH Roadmap for Medical Research, Grant U54 GM072970. OpenSim is additionally funded by the National Center for Simulation in Rehabilitation Research (NCSRR), a National Center for Medical Rehabilitation Research supported by grant R24 HD065690, and by the Mobilize Center, a National Center for Big Data to Knowledge (BD2K) supported by grant U54 EB020405. Funding for OpenSim has also been provided by the DARPA Warrior Web Program and NMS Physiome.

Other acknowledgments are owned to the authors of the OpenSim handouts and tutorial materials available on the user guide: Jeff Reinbolt, Ajay Seth, Jennifer Hicks, Scott Delp, Allison Arnold, Apoorva Rajagopal, James Dunne and Chris Carty.

Trademarks and Copyright and Permission Notice

Table of Contents

1 Overview

Computer simulation has emerged as a powerful method to investigate muscles' actions during movement, identify factors that contribute to movement disorders, and evaluate the biomechanical consequences of possible treatments. OpenSim is freely - available software for developing, analyzing, and sharing such simulations. In OpenSim, a musculoskeletal model consists of a set of rigid bodies connected by joints. Muscles span these joints and generate forces that accelerate the body, consistent with the laws of physics. OpenSim models allow users, for example, to examine the effects of musculoskeletal geometry and muscle-tendon properties on the forces and moments that individual muscles can produce. When used in combination with experimental data from motion capture, OpenSim allows users to visualize musculoskeletal structures during movement, scale models to individual subjects, run inverse and forward dynamics analyses, estimate internal loads (such as muscle forces or joint reaction loads, which are difficult to measure), and plot results. This hands-on tutorial introduces users to OpenSim's features, file formats, and documentation using clinically - relevant examples.

This tutorial was adapted from online materials provided by OpenSim community and the user guide. The sections were adapted from multiple sources to accommodate the theoretical aspects needed for processing the provided experimental data and carry out valuable analyses. New sections were added to feel gaps in the user guide through the examples provided here. Figures and text were updated to agree with the steps using a more recent version of OpenSim. Additional questions corresponding to the provided data were also added to encourage students in comprehending topics that were not covered here.

1.2 Learning objectives

By working through this tutorial, you will:

- Identify components of an OpenSim model
- Load a model and animate it using OpenSim's graphical user interface (GUI)
- Use the Plot Tool to plot data calculated from a model of the lower extremity
- How to convert C3D files from motion capture into OpenSim compatible file formats
- Use the Scale Tool to create a subject-specific model from a generic model
- Use the Inverse Kinematics Tool to solve for a subject's joint angles during walking from measured motion capture data
- Use the Inverse Dynamics Tool to solve for a subject's generalized forces that satisfy the recorded motion and externally applied forces
- Use the Static Optimization Tool to find a possible solution of muscle forces that is responsible for the recorded movement

- Locate OpenSim documentation and resources

1.3 Format

This handout assumes that you've already installed **OpenSim v4.1** and that you've downloaded a folder of files named **tutorial**.

In these exercises, you'll use the OpenSim GUI to visualize and analyze an existing lower extremity model. The menus and options that you must select to complete each exercise and any commands that you must type appear in **boldface**. Questions are provided at the end of each exercise to help confirm your understanding. As you complete each exercise, feel free to explore OpenSim and the model more on your own.

1.4 Installation

For this tutorial, we will use the OpenSim GUI, which can be downloaded from the simtk.org webpage[1]. Currently, the graphical user interface (GUI) is available on Windows and Mac. Linux users can build OpenSim from source code[2]. If you are an advanced user, it might be helpful to set up Python and MATLAB wrappings because some functionalities, such as importing C3D files, can be performed only through the application programming interface (API). To this end, one can follow the instructions in the user guide to set up everything correctly[3].

For this tutorial, **you will not be required to convert the C3D to OpenSim compatible formats**. However, the conversation scripts will be provided and explained so that you will adapt them according to your laboratory setup.

Also, please download and install the Mokka software.[4] Mokka is an open-source and cross-platform software for analysis and visualization of biomechanical data such as C3D.

1.5 Deliverable

Each subsection contains questions. In your report, all inquiries should be indicated (e.g., using headings). Answer as concisely as possible. All plots should be labeled with units. **Your report should be named OpenSim_Name_Surname.pdf.**

[1] https://simtk.org/frs/index.php?group_id=91
[2] https://github.com/opensim-org/opensim-core
[3] https://simtk-confluence.stanford.edu/display/OpenSim/Scripting
[4] http://biomechanical-toolkit.github.io/mokka/

2 Generic model

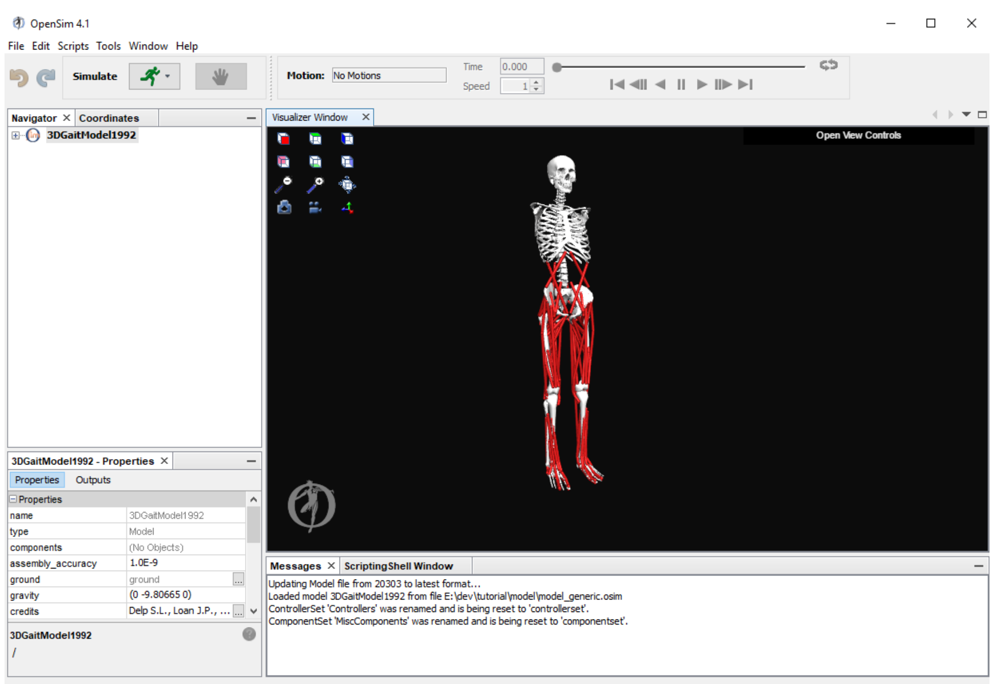
In this exercise, you'll load a lower extremity model in OpenSim and use the View, Navigator, Coordinates, and Properties windows to identify important components of the model, including bodies, joints, and muscles. The model you will use characterizes the musculoskeletal geometry and muscle force-generating capacity of an adult male with a height of 1.8 m and a mass of 75 kg. The model has 19 degrees of freedom and specifies the force-generating properties of 92 muscle-tendon units.

2.1 Loading a model

Launch OpenSim and load the model named **model_generic.osim** as follows:

- Click on the **File** menu and select **Open Model.**
- Browse and find the folder: **tutorial/model**
- Select and open the model: **model_generic.osim**

Once you've loaded this model, it will appear in the View window. The light grey skeleton shows the bodies (i.e., limb segments) of the model, and the red lines represent the muscles. The model's name, **3DGaitModel1992**, appears in the Navigator window. The Navigator window provides information about the bodies, joints, forces, and other model components. To expand one of these headings in the window, click on the plus icon (+) to its left.



2.2 Viewing a model

Click in the **View** window to make it active. Use the following commands and your mouse to view the model from different directions:

ROTATE	To rotate the view, click and hold the **left mouse button** and drag the mouse.
TRANSLATE	To translate the view, click and hold the **center mouse button** and drag the mouse.
ZOOM	To zoom, click and hold the **right mouse button**. To zoom in, drag the mouse down. To zoom out, drag the mouse up.

Also, note that there are six orienting icons located along the right side of the View window. To view the model in the –X direction, click the  icon. To view the model along other principal directions, click on the other orienting icons.

To view the axes of the OpenSim reference frame, click on the axes  icon.

To take a snapshot of the View window, click on the camera icon. You may save the image to a file.

2.3 Identifying components of a model

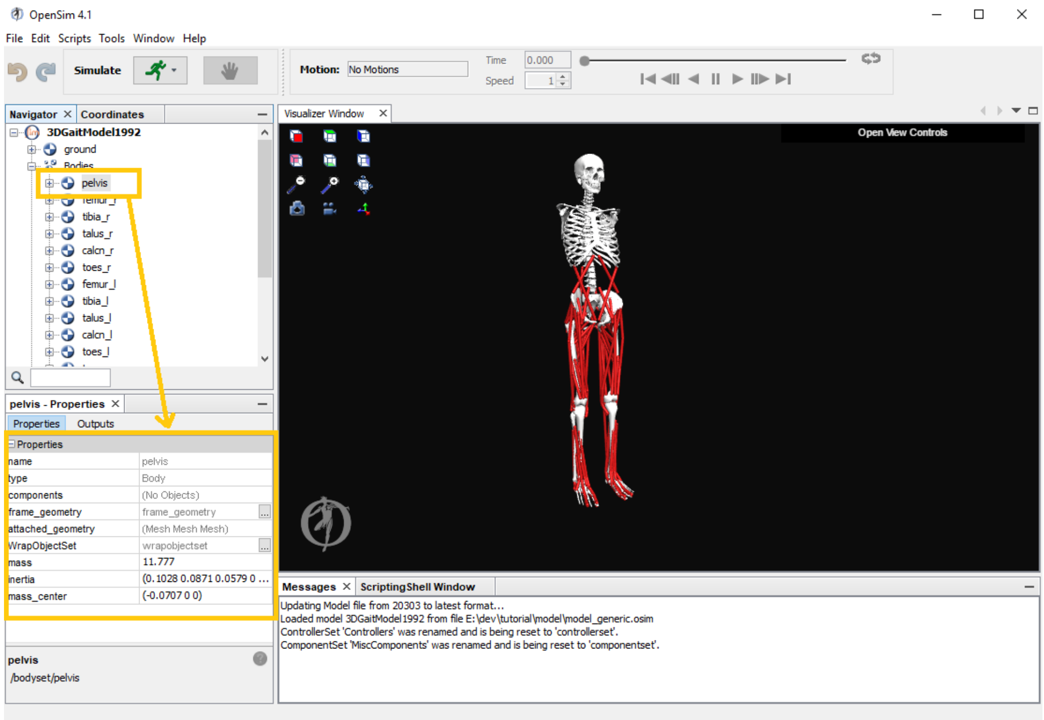
To see information about the bodies and muscles that comprise this model, click on the **plus icon** (+) next to the model's name, **model_generic**, in the **Navigator** window.

NOTE: You can choose to display the Navigator window or the Coordinates window at any time by clicking the window's title bar or by clicking the **Window** menu and selecting **Navigator** or **Coordinates**.

Bodies

An OpenSim model starts with a set of bodies (e.g., pelvis, femur, tibia). Each body is assumed to be rigid, and each body has its associated reference frame. For display purposes, bodies are typically associated with one or more "visual objects" (e.g., bone files) that describe the surface geometry. Bodies may be assigned mass and inertial properties.

- To view a list of all bodies in **model_generic**, click on the (+) next to **Bodies** in the **Navigator** window.
- To get information about a particular body, click on the body's name. Important attributes of the body, such as its mass, center of mass, and inertial properties, will appear in the **Properties** window (e.g., the pelvis body in this model has a mass of 11.777 kg).
- By default, the Properties window reports all quantities in SI units.

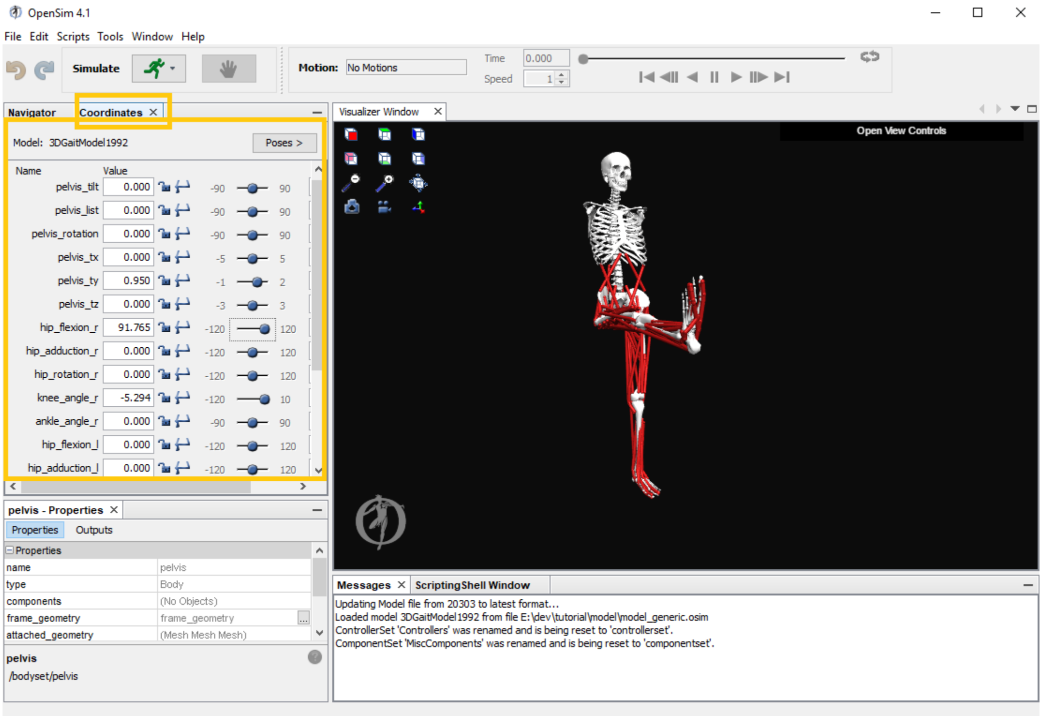


Joints

Bodies in OpenSim are connected by joints. Joints constrain bodies to move within physiological bounds. Each joint specifies how one body can translate and rotate with respect to its "parent" body, and these translations and rotations are called coordinates. For example, in **model_generic,** the femur body is connected to the pelvis body via the hip joint. The hip joint has three rotational degrees of freedom described by three coordinates: hip flexion angle, hip adduction angle, and hip rotation angle.

Every body must be connected to a parent body via a joint. The only exception is the ground body, which represents the ground reference frame. In **model_generic**, the pelvis body is connected to the ground body via a six-degree-of-freedom joint. The six coordinates labeled pelvis_rotation, pelvis_list, pelvis_tilt, pelvis_tx, pelvis_ty, and pelvis_tz, specify the location (3 translational degrees of freedom) and orientation (3 rotational degrees of freedom) of the pelvis with respect to the ground.

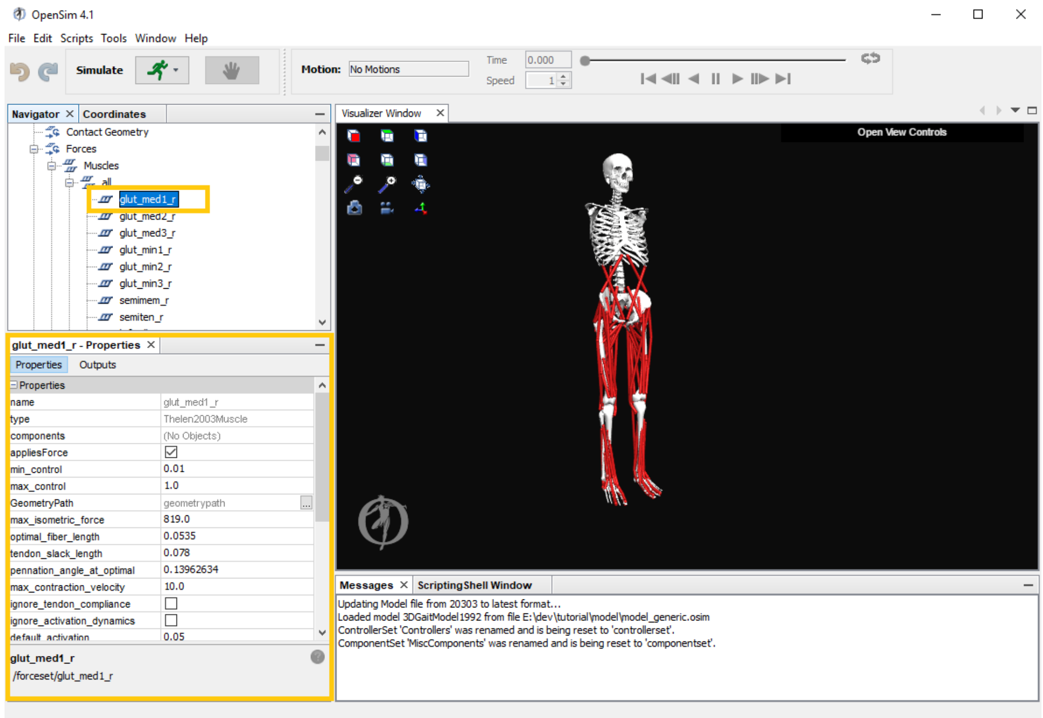
- To view a list of all the coordinates in **model_generic**, open the **Coordinates** window by clicking the window's title bar.
- The Coordinates window has sliders that correspond to each coordinate in the model. Move the sliders back and forth and examine how changing a coordinate's value changes the model's pose.
- To return the model to its default pose, click on **Poses** and select **Default**.



Muscles

Muscles in OpenSim are characterized by their geometric paths and by their force-generating capacity. The path of each muscle-tendon actuator is specified by a series of points and (sometimes) wrap objects. This geometric path determines the muscle's origin-to-insertion length and the muscle's moment arms about the spanned joint(s), all of which vary with the joint's coordinate values (e.g., a muscle's origin-to-insertion length changes as the spanned joint rotates). Each muscle-tendon actuator is also represented by a Hill-type muscle model that determines the muscle's force-generating capacity as a function of its activation, length, and velocity.

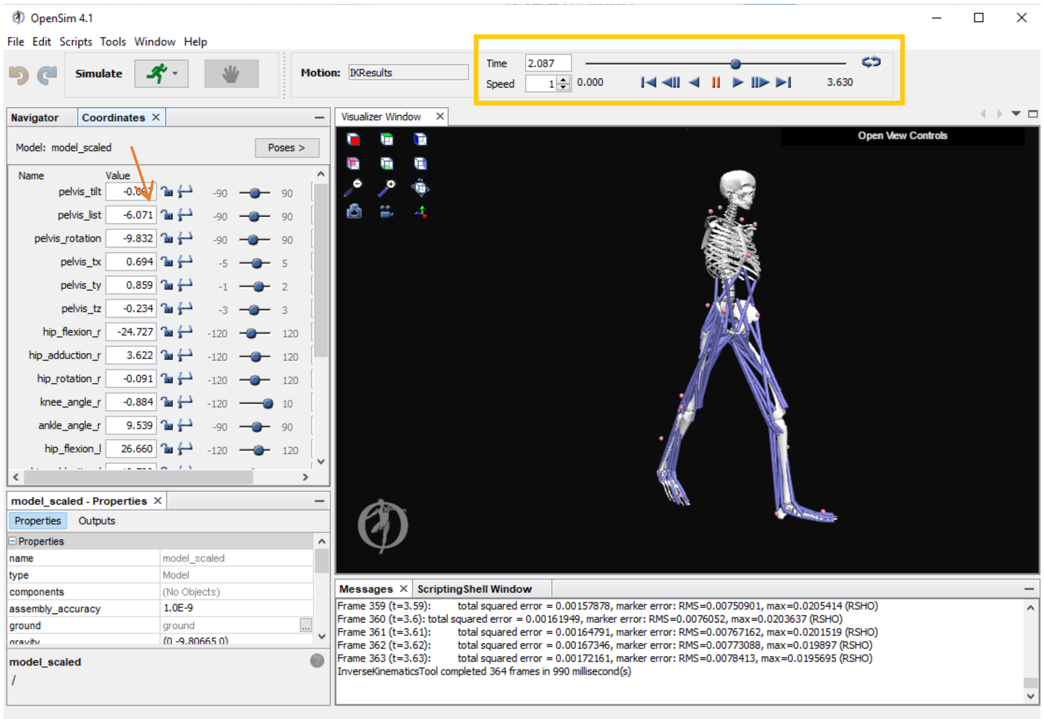
- To view a list of all the muscles in **model_generic**, click on the (+) next to **Forces** in the **Navigator** window, and then click on the (+) next to **Muscles** to expand the list of muscles.
- You may display a list of *all* muscles or a *functional group* of muscles (e.g., ***R_hip_abd***, the right hip abductors). For now, expand the list of ***all*** muscles.
- To see information about a particular muscle, click on the muscle's name. Parameters that define the muscle's force-generating capacity appear in the **Properties** window.
- You can choose whether to display the paths of one or more muscles in the model by right-clicking either the name of a muscle, the name of a *functional group*, or *all* in the **Navigator** window, selecting **Display**, then **Show, Show Only,** or **Hide**.



2.4 Animating a model

To animate the model **model_generic**, load the motion file named **task_InverseKinematics.mot** as follows:

- Click on the **File** menu and select **Load Motion.**
- Browse and find the folder: **inverse_kinematics** within the **tutorial**
- Select and open the file: **task_InverseKinematics.mot**
  This file contains joint angle data that describe the time-varying gait pattern.
- If necessary, re-center the model in the **View** window using the View commands.
- Use the **motion slider** and the **video control buttons** to play the animation and make the model "walk." Try the buttons that loop, pause, and adjust the speed of the animation. In the Coordinates window, observe that as the animation plays, the values of the model's coordinates change accordingly.



2.5 Questions

1. *How many bodies (segments) does* ***model_generic*** *have?*

2. *How many coordinates (degrees of freedom) does* ***model_generic*** *have?*

3. *Which two bodies are connected by the* ***hip_r*** *joint?*

4. *Which coordinate represents right ankle flexion? In this model, is dorsiflexion described as a positive or negative angle?*

5. *What is the maximum isometric force that the soleus muscle can produce?*

   (HINT: Find and select soleus_*r/l* in the Navigator window, then find the muscle's *max_isometric_force* in the Properties window. Forces are given in Newtons.)

6. *Some muscles in the model are represented by multiple muscle-tendon actuators. For example, the gluteus medius muscle is split into glut_med1, glut_med2, and glut_med3. Which other muscles in the model are divided into multiple compartments? Why do you think these muscles are represented this way?*

2.6 Additional resources

You can find links to a wide range of OpenSim-related resources, including documentation, tutorials, webinars, the user forum, and announcements, at:

http://opensim.stanford.edu

The OpenSim User's Guide is available online at:

http://simtk-confluence.stanford.edu:8080/display/OpenSim/User%27s+Guide

A more detailed discussion of bodies, joints, muscles, markers, and other components of an OpenSim model is available at:

http://simtk-confluence.stanford.edu:8080/display/OpenSim/OpenSim+Models

Descriptions of the various models that were distributed with your OpenSim installation, together with a list of other existing models, are available at:

http://simtk-confluence.stanford.edu:8080/display/OpenSim/Musculoskeletal+Models

The following paper provides an overview of OpenSim's features:

Seth A, Hicks JL, Uchida TK, Habib A, Dembia CL, Dunne JJ, et al. (2018) OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement. PLoS Comput Biol 14(7): e1006223. https://doi.org/10.1371/journal.pcbi.1006223

3 Tools for analyzing models and data

This exercise provides a brief introduction to some of OpenSim's commonly - used tools. From the OpenSim GUI, you will:

- use the Plot tool to plot the moment arms of muscles about the ankle, as calculated from the model **model_generic**
- use the Scale tool, together with measured marker data from a static trial, to scale **model_generic** to the dimensions of a subject
- use the Inverse Kinematics (IK) tool, together with the scaled model, to solve for the subject's joint angles during walking from measured marker trajectories
- use the Inverse Dynamics (ID) tool to solve for a subject's generalized forces that satisfy the recorded motion and externally applied forces
- use the Static Optimization (SO) tool to find a possible solution of muscle forces that is responsible for the recorded movement

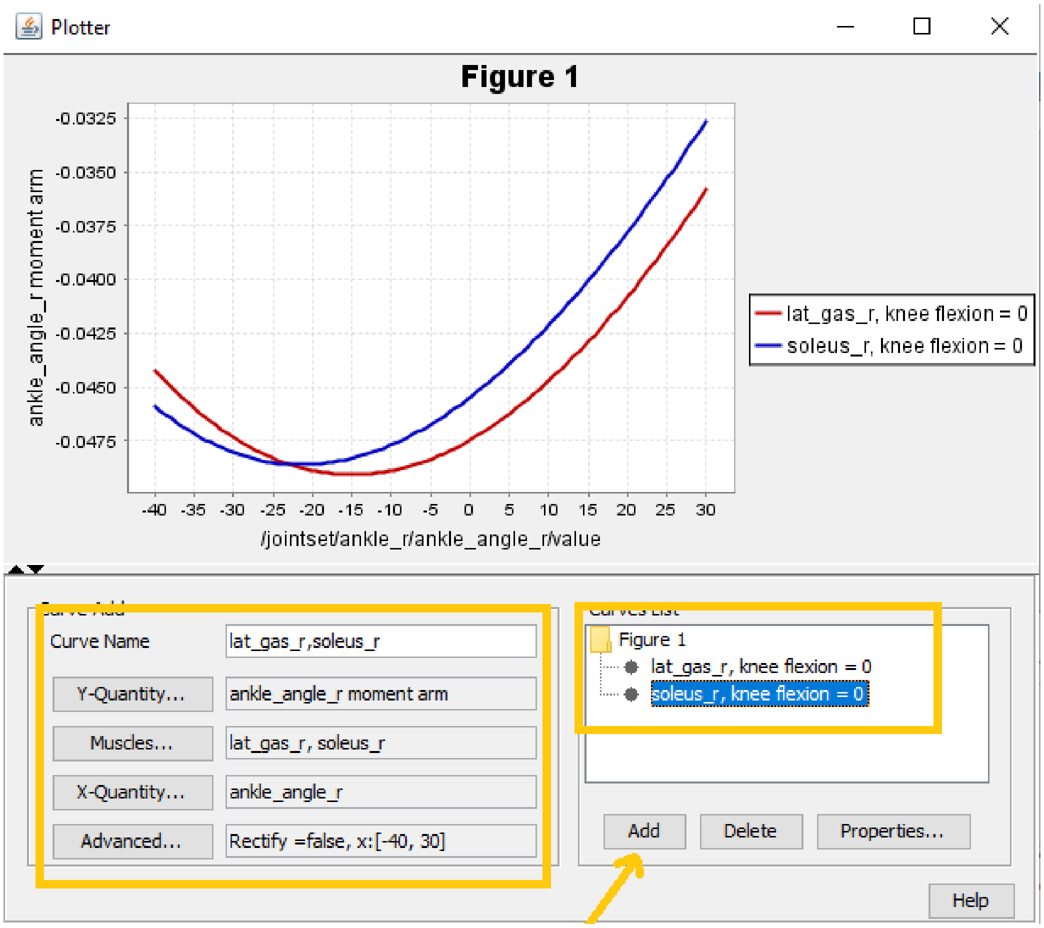
3.1 Plotting muscle moment arms using the plot tool

OpenSim's Plot tool is relatively straightforward: select a quantity to plot along the X-axis, select a quantity to plot along the Y-axis, and hit the ADD button to generate the plot. Quantities that are often plotted along the X-axis include time, the gait cycle, a relevant joint angle, etc. Quantities that may be plotted along the Y-axis include geometric data that OpenSim calculates from a model (e.g., muscle-tendon lengths and moment arms), data that OpenSim calculates from a muscle model (e.g., muscle forces and fiber lengths) as well as kinematic, kinetic, or other data that you could generate using one of OpenSim's tools (e.g., joint angles from Inverse Kinematics or joint moments from Inverse Dynamics).

To get started, let's plot the ankle plantarflexion moment arms of the gastrocnemius and soleus muscles (right leg) across a range of ankle angles using the model that you've already loaded.

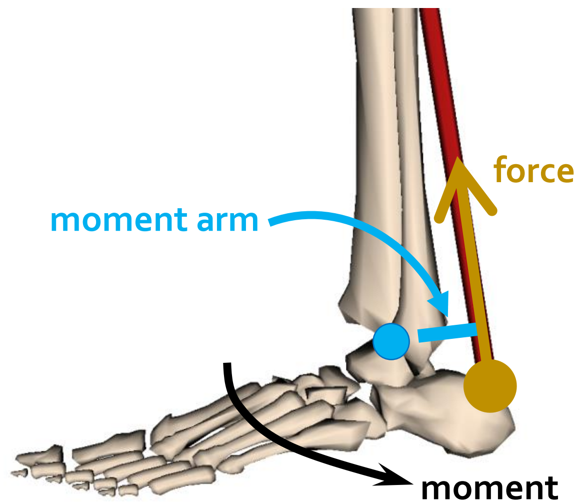
- Ensure that the model_generic is in its default pose by clicking on Poses and selecting the **Default** from the Coordinates window.
- Open a plotter window by clicking on the **Tools** menu and selecting **Plot.**
- Click on **Y-quantity** and select **moment arm -> ankle_angle_r.**
- Click on **Muscles**. Find and select ***lat_gas_r*** and ***soleus_r***, then close the *Select Muscles* window.
- Click on **X-quantity** and select **ankle_angle_r.**
- Click on the **Add** button to generate the curves.

The plotter window will show the ankle plantarflexion moment arm for the selected muscles over a range of ankle angles (from -40° plantarflexion to +30° dorsiflexion) <u>as computed from the current model at its current pose</u>.



- For reference, rename these curves as follows. **Right-click** on the name of each curve in the <u>Curves List</u>, select **Rename**, and rename the curves "**lat_gas_r, Knee Flexion = 0**" and "**soleus_r, Knee Flexion = 0**".
- To give your figure a meaningful title, **right-click** on the figure, select **Properties,** and rename the title **"Ankle Flexion Moment Arm"**. Leave this plot open since you'll return to it later.

A muscle's moment arm is the perpendicular distance from the muscle's line of action to the rotation's joint axis. A muscle's moment arm transforms the muscle's force into a moment about the joint. To produce the same moment, a muscle with a smaller moment arm will need to generate *more force* than a muscle with a larger moment arm. A muscle's moment arm determines how much the muscle's origin-to-insertion length changes when the spanned joint rotates. A muscle with a smaller moment arm will *change length less* than a muscle with a larger moment arm, thus generating force over a wider range of joint angles.



Questions

7. *Are the ankle flexion moment arms of the gastrocnemius and soleus muscles plotted as positive or negative? Given the muscles' geometric paths and the sign convention for the* ***ankle_angle_r*** *coordinate as defined in* ***model_generic****, does your answer make sense?*

8. *Your plot shows the muscles' moment arms about the ankle with the knee fully extended. Do you think the muscles' moment arms about the ankle will change if the knee is flexed?*

   One way to answer this question is to re-calculate the muscles' moment arms with the knee flexed 90°. Let's try it:

   - In the **Coordinates** window, change **knee_angle_r** to -90°.

     Note that knee flexion angles are defined as negative in this model.

   - In your same plotter window, verify that:
     Y-Quantity is still set to **ankle_angle_r moment arm**
     X-Quantity is still set to **ankle_angle_r**
     Muscles is still set to **lat_gas_r** and **soleus_r**

   - Click on the **Add** button to overlay the new curves onto the existing figure.

   - In the Curves List, **right-click** on the name of each curve that you just added and select **Rename**. Rename these curves "**lat_gas_r, Knee Flexion = 90**" and "**soleus_r, Knee Flexion = 90".**

Additional resources

The following papers describe studies in which muscle moment arms were analyzed to answer scientific or clinical questions:

Lee and Piazza (2009)

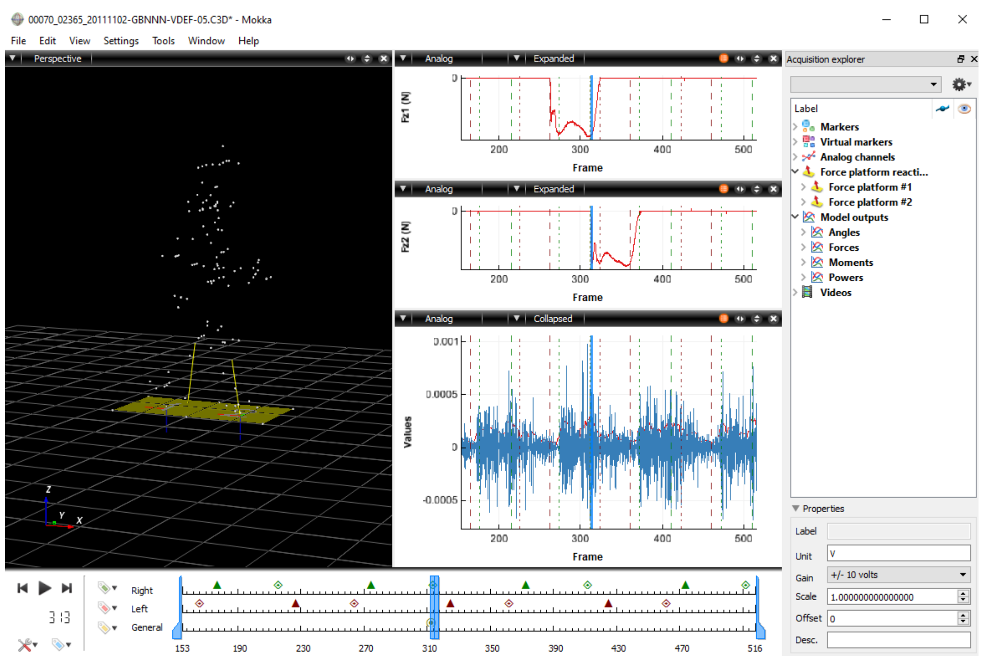
Built for speed: musculoskeletal structure and sprinting ability
Journal of Experimental Biology, vol. 212, pp. 3700-3707.

Delp, Hess, Hungerford, and Jones (1999)
Variation of rotation moment arms with hip flexion
Journal of Biomechanics, vol. 32, pp. 493-501.

Arnold and Delp (2001)
Rotational moment arms of the medial hamstrings and adductors vary with femoral geometry and limb position: implications for the treatment of internally rotated gait
Journal of Biomechanics, vol. 34, pp. 437-447.

3.2 Importing C3D into OpenSim

The C3D format is a public domain file format that has been used in biomechanics, animation, and gait analysis laboratories to record synchronized 3D and analog data since the mid-1980s. It is supported by all major 3D motion capture system manufacturers and companies in the biomechanics, motion capture, and animation industries. C3D files are a standard that contains all the information needed to read, display, and analyze 3D motion data with additional analog data from force plates, electromyography, accelerometers, and other sensors. Unlike various other 3D formats (Biovision BVH files, OpenSim .TRC and .MOT files, etc.), the .C3D format is a standard that does not change each time a manufacturer releases a new product, so data stored in the C3D format will remain readable for a long time. The Mokka software can be used to visualize C3D files and inspect their content.



Converting C3D into .trc and .mot files recognized by OpenSim

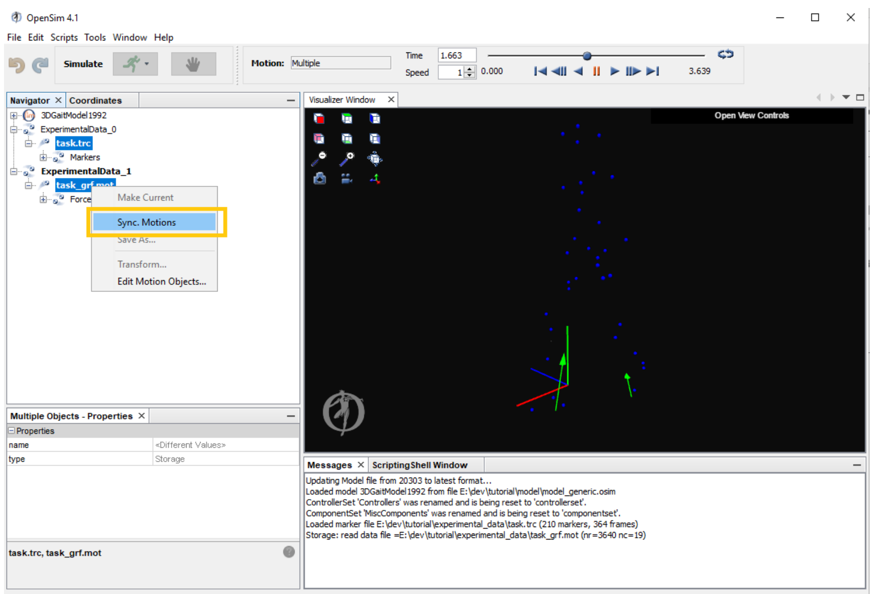
We have included a utility in Python that can be used to perform some common operations, such as rotating data, conversation of units, interpolation of missing values, filtering of noise, and writing marker and force data to OpenSim file formats. The Python file can be found in your resources **tutorial/scripts/c3d_to_opensim.py**. The backbone of the script performs the following operations:

1. Read a C3D file containing markers and forces.
2. Transform the coordinate system of the lab into the coordinate system of OpenSim, so that the up direction corresponds to Y-axis.
3. Convert units to N for forces, m for the position, and Nm for moments.
4. Interpolate to fill for missing values and remove noise artifacts using filters.
5. Write the markers to a .trc file and the forces to a .mot file.

Using the data previewer

After exporting the into OpenSim compatible file formats, the motion capture data can be previewed in the OpenSim GUI to verify that preprocessing was done correctly and that data is in agreement with the intended model. If users have multiple files representing different data pieces, OpenSim allows users to synchronize data to verify that the data was transformed consistently. The data previewer handles two types of data: marker trajectories (contained in .trc files) and measured forces (e.g., ground reactions, contained in .mot files).

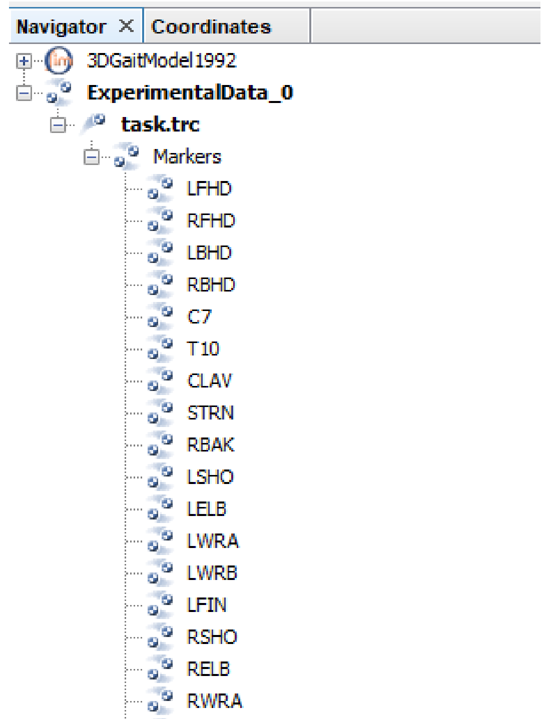
Select **Preview Motion Data… (or Associative Motion Data…)** from the **File** menu. Then browse to select the motion capture data file to be visualized. This enables the user to synchronize the motions from different motion files. These other motions could be either motion capture files or results from OpenSim tools. For example, experimentally measured marker trajectories and ground reaction forces can be synchronized and superimposed based on a forward simulation.



Visualizing marker trajectories

In OpenSim, marker trajectories are contained in a .trc file. When loaded with the data previewer, a component named "ExperimentalData_0" will contain the loaded motion file consisting of the marker data (e.g., "**task.trc**"). The "Markers" node lists each marker found in the file. Nodes corresponding to individual markers have the following options (access options by *right-clicking* the marker name):

- **Show**: Enabled only if a marker is hidden
- **Show Only**: Hides all other markers except for those selected.
- **Toggle Trail Display**: Toggles the display of a line representation of the selected marker trajectory (s).



Visualizing external forces

In OpenSim, external forces (e.g., ground reaction forces) are contained in a .mot file. The header of the selected force (.mot) file is the same as that expected by OpenSim tools. For the ground reaction forces used by the gait model, the column labels are shown below:

left_ground_force_vx	**left_ground_force_vy**	**left_ground_force_vz**
left_ground_force_px	**left_ground_force_py**	**left_ground_force_pz**
left_ground_torque_x	**left_ground_torque_y**	**left_ground_torque_z**
right_ground_force_vx	**right _ground_force_vy**	**right _ground_force_vz**
right _ground_force_px	**right _ground_force_py**	**right _ground_force_pz**
right _ground_torque_x	**right _ground_torque_y**	**right _ground_torque_z**

The data previewer expects groups of 6 columns for a force of the form "body"_"ForceName"_{vx, vy, vz, px, py, pz}, where *vi* corresponds to each component of the force and *pi* corresponds to each component of the location of the force (e.g., center of pressure) and 3 columns for a torque of the form "body"_"TorqueName"_{x, y, z}, corresponding to each component of the applied torque. Note that this naming convention is only necessary for previewing purposes. OpenSim tools employ a new user interface enabling you to specify any number of forces, along with points of application or torques to a model during any simulation or analysis.

Questions

9. *Open the task.c3d file located in **tutorial/experimental_data** with Mokka.*
    a. *Report the subject's height, weight, age, and sex by inspecting the metadata (**View -> Metadata -> SUBJECTS**).*
    b. *How does the coordinate system of the experiment (lab) differ from the coordinate system in OpenSim?*
    c. *Visualize the ground reaction forces (**Analog channels,** then drag and drop the component into an analog type plot). What do you observe in terms of the quality of the signal?*
    d. *Calculate the linear envelop (**Tools -> Analog -> Smooth / Linear Envelope**) of the right tibialis anterior' s EMG. What is the linear envelop?*
10. *Preview the markers (**task.trc**) and ground reaction forces (**task_grf.mot**) in the OpenSim GUI by synchronizing them. Check if the forces are correctly applied to the foot. Which labels in the ground reaction force data correspond to the left and right leg?*

Additional resources

More information about the process of preparing the data for analysis in OpenSim:

https://simtk-confluence.stanford.edu/display/OpenSim/Preparing+Your+Data

3.3 Scaling a model using the scale tool

In this exercise, you will scale **model_generic** to match the anthropometric dimensions of a child with cerebral palsy. In OpenSim, generating a subject-specific model is typically the first step in either an inverse dynamics or forward dynamics analysis. This step is very important since the results of subsequent analyses are typically sensitive to the accuracy of the scaling step.

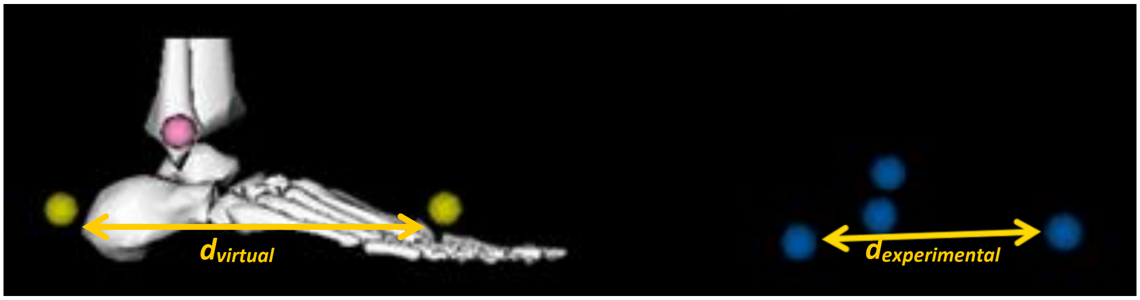
The Scale tool:

- scales the dimensions of bodies
- adjusts the mass and inertial properties of bodies
- scales muscle fiber lengths and tendon lengths, preserving their ratio

  (but does not change the muscles' maximum isometric forces)
- moves markers on the scaled model to match their measured locations, if desired

  (this facilitates tracking the markers if Inverse Kinematics is to be performed).

Scale factors are calculated independently for each body in the model and may be determined using a combination of two methods. In measurement-based scaling, scale factors are determined by comparing measured distances between specified pairs of markers on the model, known as *virtual markers,* and the corresponding distances between *experimental markers* placed on the subject (illustrated below). In manual scaling, scale factors are specified by the user based on some predetermined measure. Manual scaling is helpful when suitable marker data are unavailable or when the body's scale factors have already been determined using an alternative algorithm. In this exercise, you will scale the pelvis using manual scale factors derived from anthropometric measurements taken on the subject.



If you have information about the pose of the subject during the static trial (e.g., the experimental data were collected with the subject's knee flexed at a known angle), then you can specify the model's coordinates, in addition to the marker pairs and/or manual scale factors, during the scaling process. For all other coordinates that are not specified, the Scale tool will solve an optimization problem to determine the coordinates' values (i.e., the scaled model's pose) that best fit the virtual markers to the experimental markers.

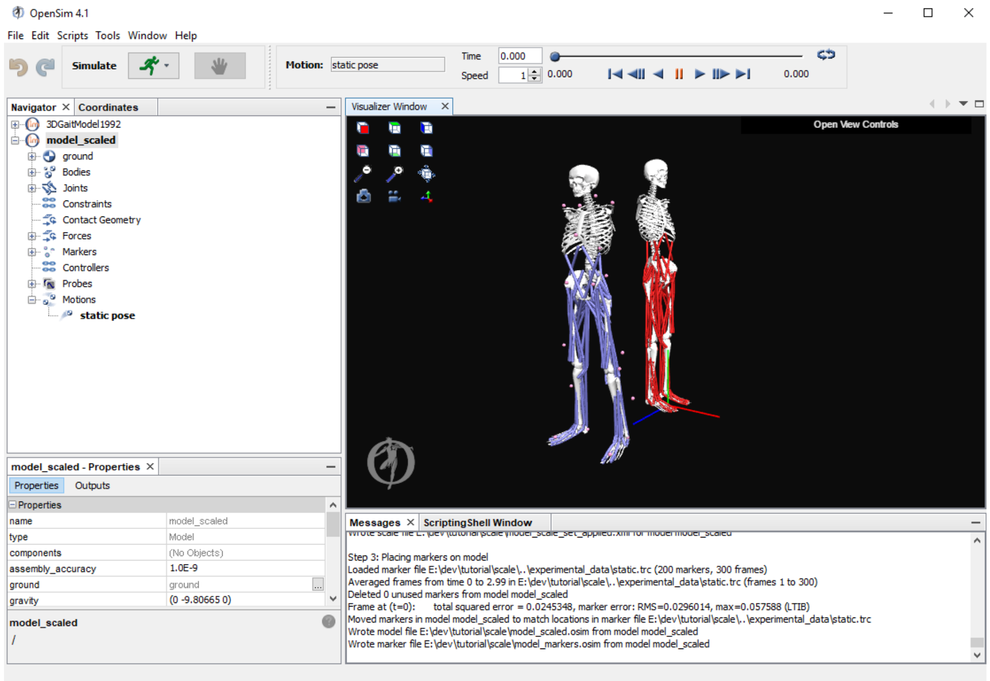
To see how the Scale tool works, let's use an existing scale settings file to scale our model, **model_generic**:

- Click on the **Tools** menu and select **Scale Model.**
- At the bottom of the *Scale Tool* dialog, click **Load**
- Browse and find the folder: **scale** within the **tutorial**
- Select and open the file: **setup_scale.xml**

The settings file that you just loaded, **setup_scale.xml**, is an *xml* file that contains pre-configured settings to scale our model to the dimensions of our subject. The settings file specifies, for example, the name of the file that includes the subject's experimental marker data. In this case, the subject's marker data are read from the file **static.trc**, which is located in the **experimental_data** folder within your **tutorial** folder. In general, settings for the Scale tool can either be specified within the OpenSim GUI or can be loaded and edited from an existing settings file (which is useful if you need to scale a model to different subjects in a consistent manner).

To complete the scaling process:

- Click on **Run** in the *Scale Tool* dialog, then click on **Close**.



Once scaling in completed, a new model named **3DGaitModel1992** will appear in the View window. Notice the difference in size between the generic model (adult-sized) and the scaled model (child-sized).

You won't use the generic model in the next exercise, so you may "hide" it as follows:

- In the **Navigator** window, right-click on the model's name, **model_generic**, and select **Display -> Hide.**

Questions

*11. What is the function of the **Adjust Model Markers** option in the **Scaling** tool?*

12. What is the scale factor for the left talus, calcaneus, and toes?

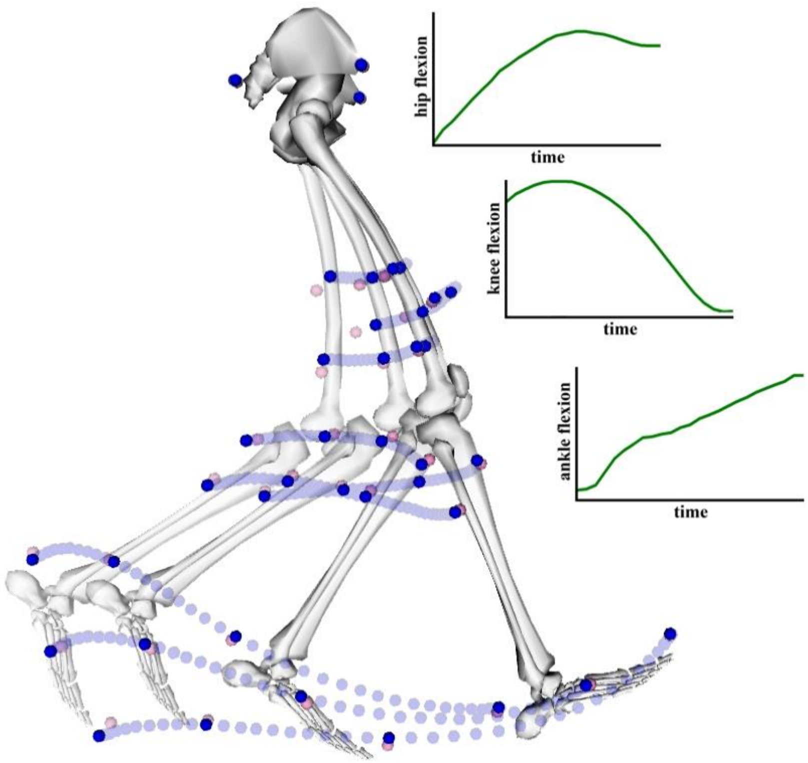
*13. In the metadata of the task.c3d file in Mokka (**View -> Metadata -> SUBJECTS)**, there exist direct segment length measurements. Compare the left and right foot length and the one reported by the **Scaling** tool (check the message window after running the tool, where the length is reported in meters).*

Additional resources

More information about how to use the Scale tool in OpenSim is available at:

http://simtk-confluence.stanford.edu:8080/display/OpenSim/Scaling

3.4 Solving for joint angles using the inverse kinematics tool



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

The model pose is the set of orientations and locations of body segments in the model at a given instant. A coordinate is a joint angle or distance that specifies the relative orientation or location of two body segments. Therefore, the set of all model coordinates completely defines the model pose.

The marker error is the distance between an experimental marker ● and the corresponding model marker ● at a given instant in time. During inverse kinematics, each marker has an associated weight specifying how strongly that marker's error should be minimized.

The coordinate error is the difference between an "experimental coordinate" and the corresponding model coordinate (e.g., joint angle or distance) value at a given instant in time. Experimental coordinate values can be joint angles obtained directly from a motion capture system (i.e., built-in mocap inverse kinematics capabilities) or techniques involving other measurement devices (e.g., a goniometer). The desired coordinate can also be specified as constant (e.g., if we know that a specific joint angle should stay at say 0˚). During inverse kinematics, each coordinate has an associated weight specifying how strongly that coordinate's error should be minimized.

The weighted least-squares minimization solved by the Inverse Kinematics Tool is

$$\underset{\boldsymbol{q}}{\text{minimize}}\left[\sum_{m=1}^{\#markers} w_m \left\| x_m^{exp} - x_m(\boldsymbol{q}) \right\|^2 + \sum_{c=1}^{\#coordinates} \omega_c \left(q_c^{exp} - q_c \right)^2 \right]$$

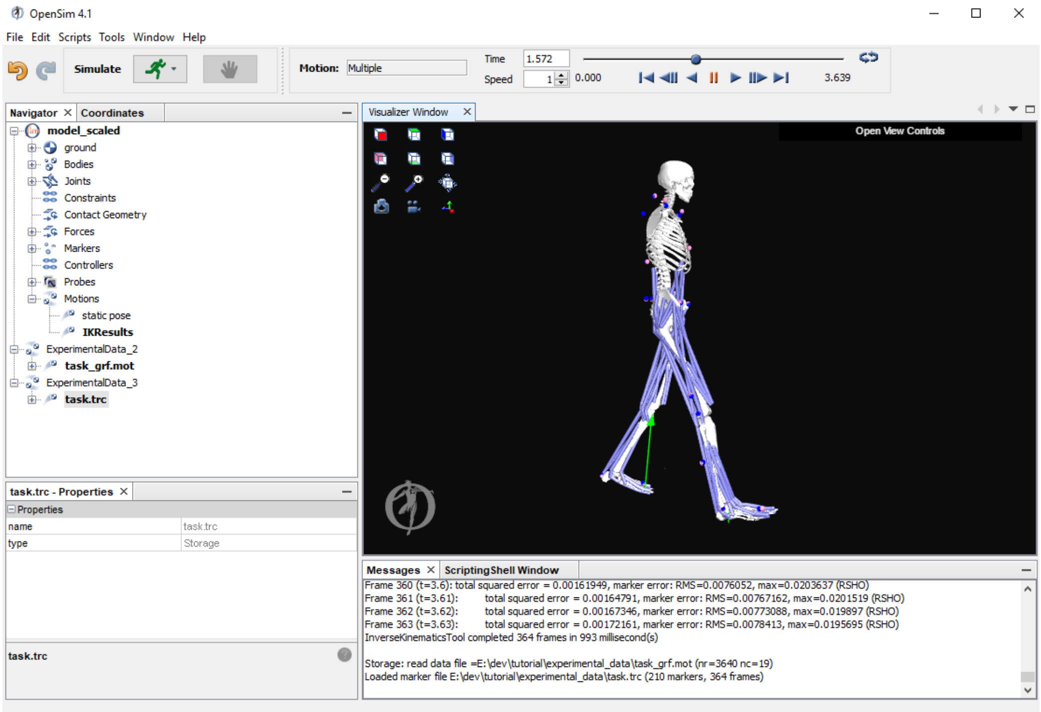
where $\boldsymbol{q}$ is the set of coordinates to be determined, $\boldsymbol{x}_m^{exp}$ is the experimental position of marker m, $\boldsymbol{x}_m(\boldsymbol{q})$ is the position of the corresponding model marker (as a function of the coordinates, $\boldsymbol{q}$), w_m is the weight associated with the mth marker, $\boldsymbol{q}_c^{exp}$ is the experimental value for coordinate c, q_c is the corresponding model coordinate, and ω_c is the weight associated with the cth coordinate.

OpenSim's IK algorithm differs from the direct kinematics methods used in many motion capture programs, such as Cortex and Plug-in-Gait. Direct kinematics methods rely on marker positions to directly define the joint axes of body segments and then use adjacent segments' axes to calculate joint angles directly. These methods are susceptible to errors in the marker positions, since noisy marker data (e.g., due to skin motion artifact) leads directly to errors in joint angles. OpenSim's IK tool, by contrast, solves for the joint angles that best align the model's virtual markers with the subject's experimental markers. In this approach, the model's virtual markers are fixed relative to the joint axes of the associated bodies, specified in the model's osim file independent of the experimental marker trajectories.

To see how the IK tool works, let's use the scaled model, **model_scaled,** together with an existing IK settings file, to solve for the joint angles:

- Click on the **Tools** menu and select **Inverse Kinematics**.
- At the bottom of the *IK Tool* dialog, click **Load.**
- Browse and find the folder: **inverse_kinematics** within the **tutorial**
- Select and open the file: **setup_ik.xml**

The settings file that you just loaded, **setup_ik.xml**, is an *xml* file that contains pre-configured settings to solve the IK problem for our subject. This file specifies weights for tracking each marker and the name of the file containing the subject's experimental marker data. In this case, the subject's marker data are read from the file **task.trc**, which is in the **experimental_data** folder within your **tutorial** folder. For the IK tool, ground contact forces are not needed, and the mass/inertia properties of the model are not used.



- Click on **Run** in the *IK Tool* dialog, then click on **Close**.

  Find the progress bar in the lower right corner of the OpenSim window. Even though you closed the *IK Tool* dialog, the tool is still running. **Wait** until the progress bar disappears before proceeding. In the **View** window, the model will begin to "walk" as the optimization problem is solved for each time step of the measured motion.

- If necessary, re-center the model in the **View** window using the View commands.

To compare your IK results to the subject's experimental marker data:

- In the **Navigator** window, under **Motions**, right-click on **Results –** this is the motion that the IK tool generated.
- From the drop-down menu, choose **Associated Motion Data.**
- Browse and find the folder: **experimental_data** within the **tutorial**
- Select and open the file: **task.trc**

  The model's virtual markers are shown in pink, and the subject's experimental markers are shown in blue. Hit **Play** in the **Motion Toolbar**. The virtual markers should closely track the experimental markers as the animation proceeds.

To examine the accuracy of the inverse kinematics solution:

- Click on the **Window** menu and select **Messages**.

  The *Messages* window displays details about all commands you have performed in OpenSim. Take a minute to explore the messages, then **scroll** to the bottom.

- The next-to-last line reports the marker errors (root-mean-square errors and maximum marker errors) associated with the last frame of the motion; the preceding lines report the marker errors associated with earlier frames of the motion. This information is often useful when identifying a set of marker weights that yields satisfactory IK results.

  Note: All marker errors are reported in units of meters, and all coordinate errors are reported in units of radians.

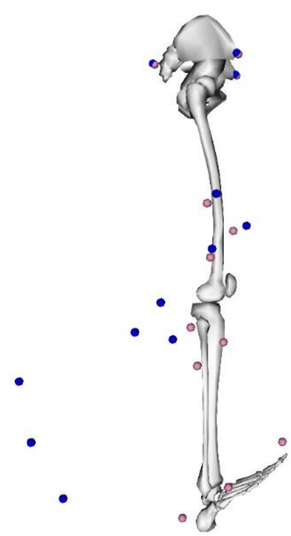
To plot the subject's calculated ankle joint angles vs. time:

- Click on the **Tools** menu and select **Plot**…
- In the plotter window, click on **Y-quantity** and select **Results (Deg.)**.
  This motion quantity provides your IK results for each of the model's coordinates.

  Alternatively, you could select **Load file…**, then browse to open the file named **task_InverseKinematics.mot**, located in the **inverse_kinematics** folder within your **tutorial 2015** folder. This file also provides your IK results. It was specified in the IK settings file and was created when you ran the IK tool.
- In the **Select Motion Quantity** window, select **ankle_angle_r**, and then click **OK**.
- Click on **X-Quantity**, select **time,** and click **OK**.
- Click on the **Add** button to generate the curve.

Questions

14. For the model shown on the right, which coordinate(s) need to be adjusted to create a model pose that "best matches" the experimental markers shown at the beginning of the swing phase?



A. Hip

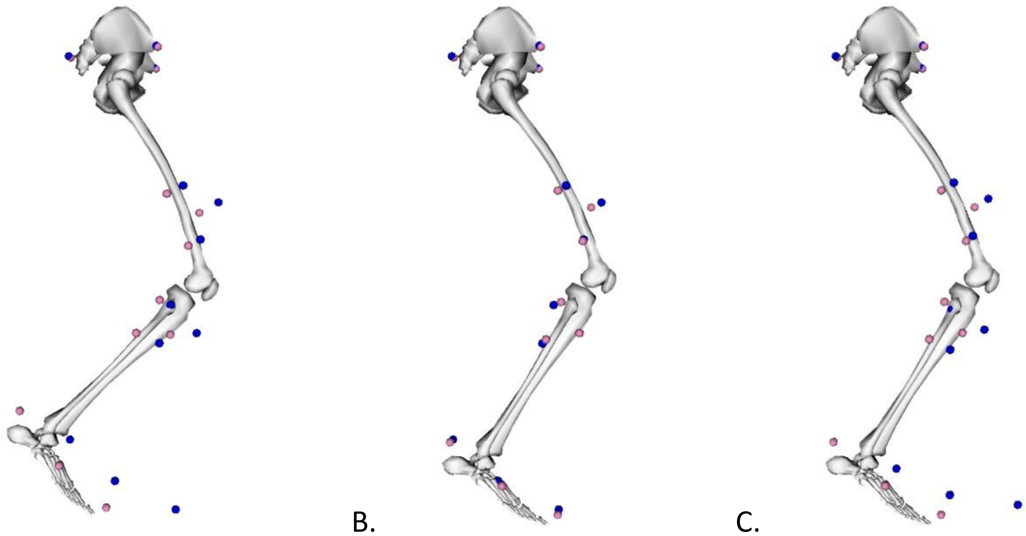
B. Knee

C. Ankle

D. Hip and ankle

E. Knee and ankle

15. For the model poses and experimental markers shown below, which combination of pose and markers has the minimum marker errors?



A. B. C.

16. *In theory, experimental markers on the thigh and shank could have more skin movement artifacts compared with the foot markers; which of the following scenarios would be most appropriate for the weighted least squares minimization solved by the Inverse Kinematics Tool?*

    A. Decrease tracking weights on thigh markers

    B. Decrease tracking weights on shank markers

    C. Increase tracking weights on foot markers

    D. All of the above

17. *What is the root-mean-squared (RMS) marker error from the last frame of the motion? Does this seem reasonable?*

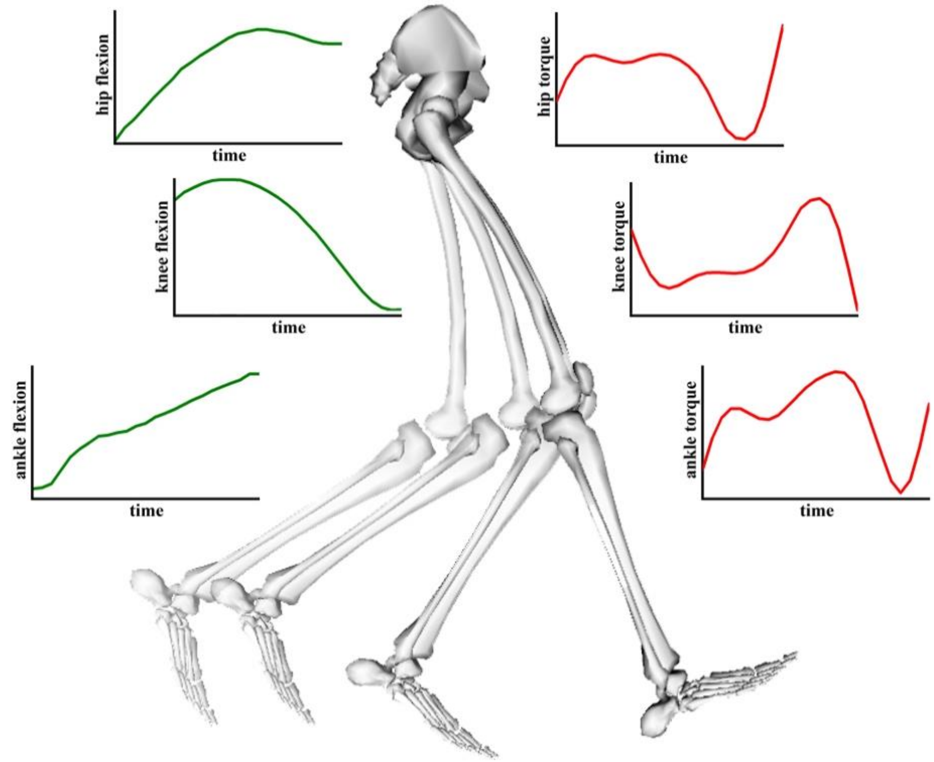
18. *Plot the hip flexion-extension, knee flexion-extension, and ankle plantar flexion-dorsiflexion angles of the left leg on a single graph. Search through the internet and compare the shape of these curves for a healthy gait. Do they seem normal?*

Additional resources

More information about the use of the Inverse Kinematics tool is available at:

http://simtk-confluence.stanford.edu:8080/display/OpenSim/Inverse+Kinematics

3.5 Solving for joint moments using the inverse dynamics tool



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 that generates the experimental kinematics. The equations of motion relate the model accelerations to the forces and/or joint torques applied to the model.

A *coordinate* is a joint angle or distance that specifies the relative orientation or location of two body segments in the model. The derivative (rate of change) of a coordinate with respect to time is the coordinate's *velocity*. In turn, the time derivative of its velocity is the coordinate's *acceleration*. The collection of coordinates and their velocities and accelerations describe the *kinematics* of the model.

Forces and *torques* cause the model to accelerate according to Newton's second law. A force can be applied to points (e.g., ground reactions) or between points (e.g., muscles) on the model. Torque can be applied to a coordinate (e.g., joint torque) to accelerate that joint angle directly.

From Newton's second law, we can determine the kinetics necessary to accelerate the model by treating the skeleton as a set of interconnected rigid - bodies with inertial properties such that:

$$\underbrace{\boldsymbol{\tau}}_{unknowns} = \underbrace{\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) - \boldsymbol{G}(\boldsymbol{q}) - \boldsymbol{F}}_{knowns}$$

where $\boldsymbol{\tau}$ is the set of joint torques, $\boldsymbol{q}, \dot{\boldsymbol{q}}, and\ \ddot{\boldsymbol{q}}$ are the coordinates and their velocities and accelerations, respectively; $\boldsymbol{M}(\boldsymbol{q})$ is the mass matrix, which depends on the coordinates and inertial properties of the model; $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is the combination of Coriolis and centrifugal forces, which depend on the coordinates and their velocities; $\boldsymbol{G}(\boldsymbol{q})$ is the gravitational force, which depends on the coordinates, and $\boldsymbol{F}$ is other forces applied to the model.

The motion of the model is entirely defined by the coordinates and their velocities and accelerations. Consequently, all the terms on the right-hand side of the equations of motion are known. The remaining set of joint torques on the left-hand side is unknown. The Inverse Dynamics Tool uses the known motion of the model to solve the equations of motion for the unknown joint torques.

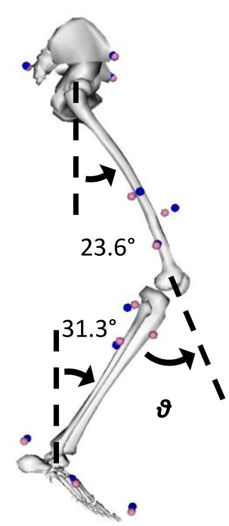
To see how the ID tool works, let's use the scaled model, **model_scaled**, together with an existing ID settings file, to solve for the generalized forces:

- Click on the **Tools** menu and select **Inverse Dynamics**.
- At the bottom of the *ID Tool* dialog, click **Load**.
- Browse and find the folder: **inverse_dynamics** within the **tutorial**
- Select and open the file: **setup_id.xml**

Questions

19. For the model shown on the right, what is the value (ϑ) of the knee coordinate (Note: extension is +)?

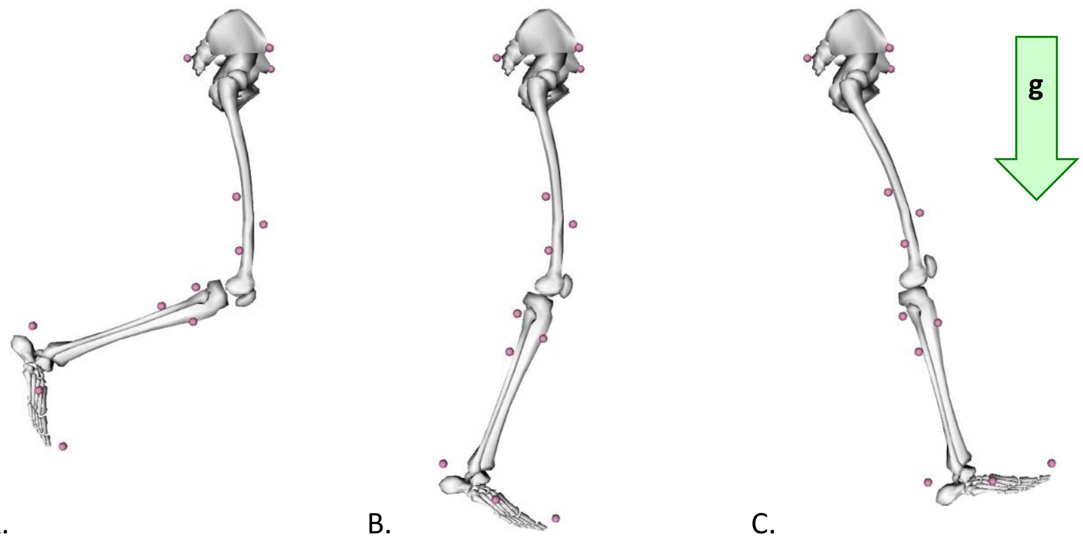
A. 23.6°   B. -54.9°   C. 31.3°   D. -125.1°



20. Given that the model shown on the right is at rest, what is the knee's velocity?

A. 23.6°/s   B. -54.9°/s   C. 3.89°/s   D. 0°/s

*21. For the model poses shown below at rest and with gravity (**g**) as the only force acting on the model, which pose requires the largest torque at the knee joint?*



A. B. C.

22. Plot the hip flexion-extension, knee flexion-extension, and ankle plantar flexion-dorsiflexion moments of the left leg on a single graph. Search through the internet and compare the shape of these curves for a healthy gait. Do they seem normal?

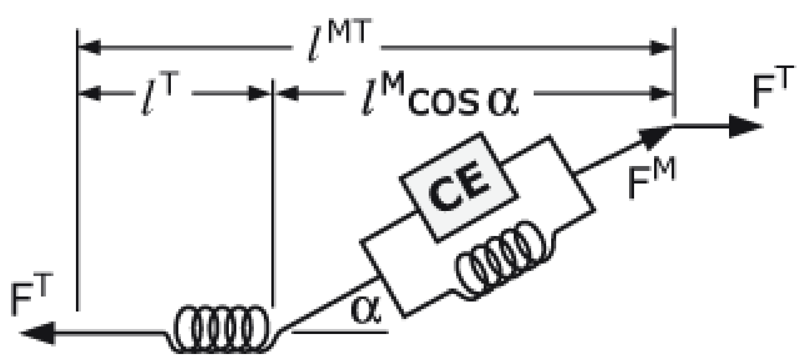
Additional resources

More information about using the Inverse Dynamics tool in OpenSim is available at:

http://simtk-confluence.stanford.edu:8080/display/OpenSim/Inverse+Dynamics

3.6 Muscle architecture

In OpenSim, a muscle's force is calculated as the sum of its active force, generated by a contractile element (CE), and its passive force, generated by a parallel elastic element. The muscle's force is transmitted via an elastic tendon to the skeleton, where the fraction of the force transmitted by the tendon depends on the pennation angle, α, between the muscle fibers and the tendon. A schematic of the muscle model is shown below. The active force produced by a muscle depends on its activation, length, and velocity.



$$f^M = f_o^M\left(af^L(\tilde{l}^M)f^V(\tilde{v}^M) + f^{PE}(\tilde{l}^M)\right)$$

$$f^T = f^M \cos \alpha$$

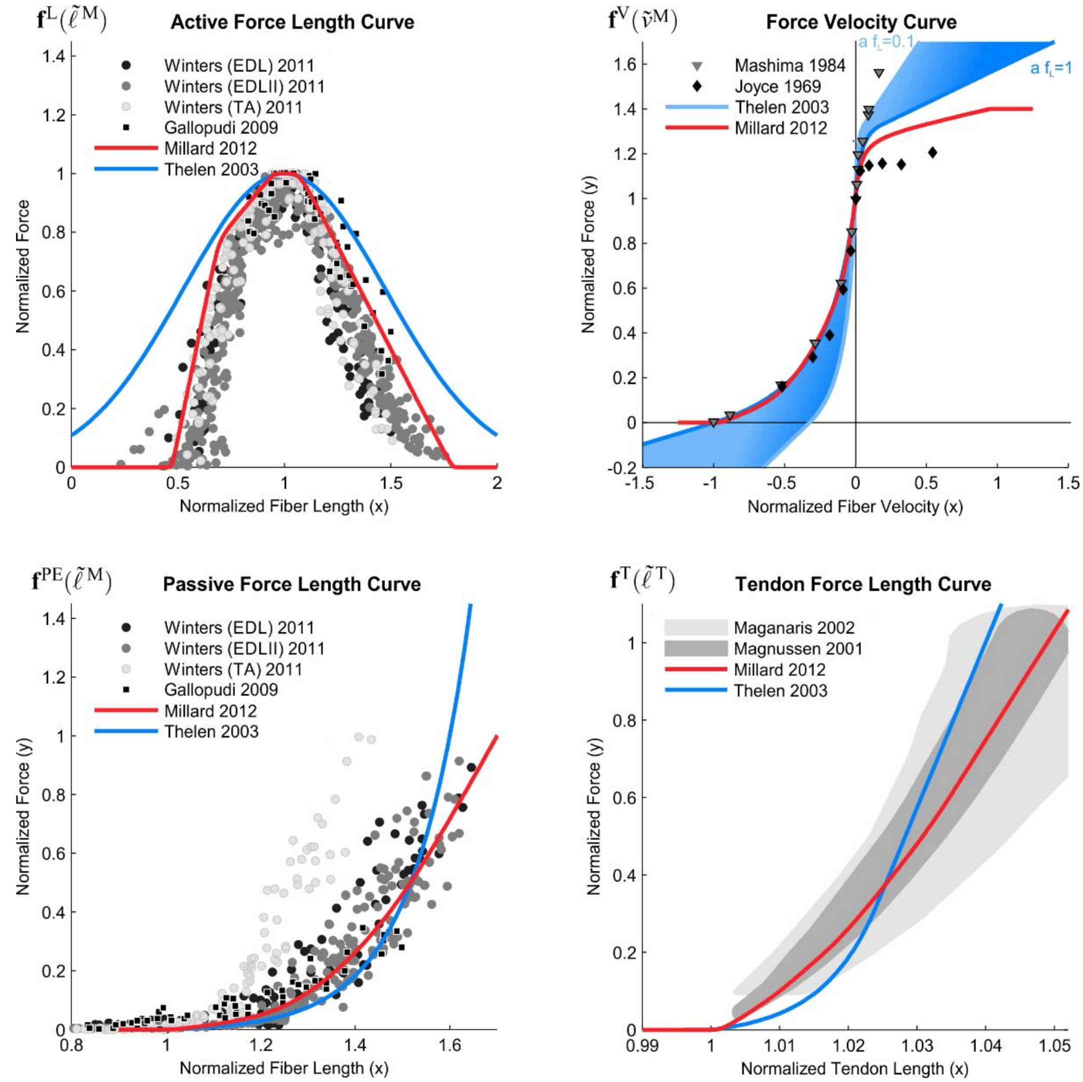
Five parameters (per muscle) are used to scale the properties of a generic Hill-type model to each muscle:

- Optimal fiber length of the muscle, l_o^M
- Maximum isometric force of the muscle, f_o^M
- Slack length of the tendon, l_s^T
- Pennation angle, α
- Maximum contraction velocity, v_{max}^m

A muscle generates its maximum active force (f_o^M) when the length of the muscle fiber (l^M) is at the optimal fiber length (l_o^M). Though the optimal fiber length differs for different muscles, the isometric force-generating capacity of *any* muscle can be characterized by its current fiber length relative to its optimal length.

During non-isometric contractions, the force developed by muscle varies nonlinearly with its rate of lengthening, which is represented by the force-velocity curve

A muscle's activation (defined in OpenSim as a number between 0 and 1) determines the percentage of the muscle's maximum active force (which, remember, is a function of the muscle fiber's length and velocity) that the muscle generates. For example, when activation=1 (100%), a muscle generates *all* of its maximum active force, and when activation = 0.5 (50%), a muscle generates *half* of its maximum active force.



When a muscle's fiber length is *longer* than its optimal fiber length, the parallel elastic element is stretched, generating passive force. This passive force depends on the muscle's length, independent of activation.

When a tendon is stretched beyond its slack length (l_s^T), it also generates passive force. Because muscle operates in series with tendon, the muscle's force and the tendon's force must be in equilibrium, accounting for pennation angle.

All muscle models in OpenSim include implementations of the F-L and F-V curves for muscle and the F-L curve for the tendon. Values for the muscle-tendon parameters are based on published measurements of the muscle architecture in cadaveric specimens and tested and refined based on strength measurements in healthy subjects.

Questions

23. *Why does the force-length curve of the contractile element of the muscle have a bell shape? Please elaborate. Hint: it has to do with the* overlap between actin and myosin filaments[5].

24. *What is the meaning of the force-velocity shape of the contractile element, and how does it relates to physiology? (You can refer to the footnote).*

Additional resources

More information about Hill-type muscle models in OpenSim is available at:

https://simtk-confluence.stanford.edu:8443/display/OpenSim/Muscle+Model+Theory+and+Publications

More information about muscle architecture measurements used to parameterize Hill-type models is available in the following papers:

Ward, Eng, Smallwood, and Lieber (2009)

Are current measurements of lower extremity muscle architecture accurate?

Clinical Orthopedics and Related Research, vol. 467, pp. 1074-1082.

Arnold, Ward, Lieber, and Delp (2010)

A model of the lower limb for analysis of human movement

Annals of Biomedical Engineering, vol. 38, pp. 269-279.

3.7 Solving for muscle forces using static optimization

The Static Optimization Tool steps through each time frame of a motion and computes the muscle activations in the model that generate the experimental kinematics. The equations of motion relate the model accelerations to the muscle moments applied to the model. The musculoskeletal geometry relates the muscle moments to the muscle forces generating those moments. The muscle contraction dynamics relate the muscle forces to the muscle activations.

As described earlier, the motion of the model is completely defined by the coordinates and their velocities and accelerations. The "distribution" problem arises from the fact there are more unknown muscle forces than the number of coordinates. 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 constraints:

[5] http://www.scholarpedia.org/article/Muscle_Physiology_and_Modeling

$$\underbrace{\sum_{m=1}^{nm}(a_m F_m^o) r_{m,j} = \tau_j}_{ideal\ force\ generators}$$

or

$$\underbrace{\sum_{m=1}^{nm}[a_m f(F_m^o, l_m, v_m)] r_{m,j} = \tau_j}_{constrained\ by\ force-lenght-velocity\ properties}$$

while minimizing the objective function:

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

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^o is its maximum isometric force; l_m is its length; v_m is its shortening velocity; $f(F_m^o, l_m, v_m)$ 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.

To see how the SO tool works, let's use the scaled model, **model_scaled**, together with an existing SO settings file, to solve for the generalized forces:

- Click on the **Tools** menu and select **Static Optimization**.
- At the bottom of the *SO Tool* dialog, click **Load**.
- Browse and find the folder: **static_optimization** within the **tutorial**
- Select and open the file: **setup_so.xml**

Questions

25. *After executing SO, the muscle activations* ***task_StaticOptimization_activation.sto*** *are exported in the* ***static_optimization*** *folder. The EMG envelop* ***emg_env.sto*** *was extracted and saved in the* ***experimental_data*** *folder. Compare the activations predicted by SO with the EMG for the following muscles: lat_gas_l, tib_ant_l, rect_fem_l. Please comment.*

26. *What is the difference between muscle excitation and activation? What does the EMG measure? How can we compare measured EMG and predicted muscle activity from OpenSim?*

27. *Is the cost function used in SO appropriate for studying pathological conditions such as Cerebral palsy or Parkinson's disease?*

Additional resources

More information about using the Inverse Dynamics tool in OpenSim is available at:
http://simtk-confluence.stanford.edu:8080/display/OpenSim/Static+Optimization