Modeling and Simulation Workflow Using Open Knee(s) Data

Model Development Specifications

Cleveland Clinic Approach

Document created on May 10, 2018; last updated on August 15, 2018.

Document prepared by

Ariel Schwartz, BSc – schwara2@ccf.org, +1 (216) 445 0373

Department of Biomedical Engineering and Computational Biomodeling (CoBi) Core, Lerner Research Institute, Cleveland Clinic, 9500 Euclid Avenue (ND20), Cleveland, OH 44195, USA

Snehal K. Chokhandre, MSc – chokhas@ccf.org, +1 (216) 445 3555

Department of Biomedical Engineering and Computational Biomodeling (CoBi) Core, Lerner Research Institute, Cleveland Clinic, 9500 Euclid Avenue (ND20), Cleveland, OH 44195, USA

***Ahmet Erdemir, PhD –** erdemira@ccf.org, +1 (216) 445 9523

Department of Biomedical Engineering and Computational Biomodeling (CoBi) Core, Lerner Research Institute, Cleveland Clinic, 9500 Euclid Avenue (ND20), Cleveland, OH 44195, USA

**For correspondence.*

Table of Contents

Cite this document as

Schwartz A, Chokhandre, SK, Erdemir, A, *Modeling and simulation workflow using Open Knee(s) data: model development specifications – Cleveland Clinic approach*, August 15, 2018, Cleveland Clinic, Cleveland, Ohio, USA.

Contributions to this document by Ariel Schwartz, Snehal Chokhandre, and Ahmet Erdemir were supported by the National Institute of Biomedical Imaging and Bioengineering, National Institutes of Health (R01EB024573; *Reproducibility in simulation-based prediction of natural knee mechanics*; Principal Investigator: Erdemir) and in part by the National Institute of General Medical Sciences, National Institutes of Health (R01GM104139; *Open Knee(s): virtual biomechanical representations of the knee joint*, Principal Investigator: Erdemir).

Synopsis

This document describes planned model development specifications that are aimed for generating an initial working model of the knee joint based on an existing data set from the Open Knee(s) project[1]. The proposed modeling activities are in response to the *Model Development* phase[2] of the project *Reproducibility in simulation-based prediction of natural knee mechanics,* a study funded by the National Institute of Biomedical Imaging and Bioengineering, National Institutes of Health (Grant No. R01EB024573)[3]. The outlined choices for modeling and simulation processes represent those of the Cleveland Clinic team, who launched and has been maintaining the Open Knee(s)[1]. These choices are primarily aimed for developing a comprehensive and extensible knee joint model incorporating anatomical and mechanical detail of its major structures.

Data Utilized

Described model development workflow utilizes Open Knee(s)[1] data, specifically those from specimen oks003. This specific data set was disseminated at the project site of *Reproducibility in simulation-based prediction of natural knee mechanics,* a study funded by the National Institute of Biomedical Imaging and Bioengineering, National Institutes of Health (Grant No. R01EB024573)[3].

Data set was part of the *Model Development* phase of the project[2] and can be accessed as the package *Data for MD – oks003*[4]. Donor specifics of specimen oks003 are:

- Left knee
- Age: 25 years
- Gender: Female
- Height: 1.73 m
- Weight: 68 kg
- BMI: 22.8

Described model development specifications utilize the following magnetic resonance imaging (MRI) datasets (all provided in NIfTI format in the same imaging coordinate system)[2]:

1. General Purpose MRI (3D T1-weighted without fat suppression, isotropic, 0.5 mm resolution)
   - 1.3.12.2.1107.5.2.19.45406.2014120210113013368841431.0.0.0.nii
2. Cartilage MRI (3D T1-weighted with fat suppression, 0.35 mm sagittal plane resolution, 0.7 mm out of plane resolution)
   - 1.3.12.2.1107.5.2.19.45406.2014120210325342222042395.0.0.0.nii
3. Connective tissue MRI (proton density, turbo spin echo in sagittal, axial, and coronal planes, 0.35 mm in plane and 2.8 mm out of plane resolution)
   - 1.3.12.2.1107.5.2.19.45406.2014120211450413949434840.0.0.nii
   - 1.3.12.2.1107.5.2.19.45406.2014120211045428131243076.0.0.0.nii

- 1.3.12.2.1107.5.2.19.45406.2014120211095362283243280.0.0.0.nii

Details of image acquisition specifics can be found at Open Knee(s) project site[5].

Overview of Modeling and Simulation Processes

A three-dimensional computational model of the knee, specifically finite element representation of the tibiofemoral and patellofemoral joints, will be developed. The model will be based on existing magnetic resonance images acquired from a cadaver specimen (left knee; 25 years old female donor – height: 1.73 m, weight: 68 kg, BMI: 22.8). Anatomy and mechanics (motion and deformation) of major tissue components of the knee will be represented: bones (femur, tibia, fibula, patella), cartilage (femoral, tibial (medial and lateral), patellar), menisci (medial and lateral) ligaments (anterior and posterior cruciate; medial and lateral collateral, patellar), and quadriceps tendon.

Using 3D Slicer[6,7], the tissue boundaries will be identified manually from multiple magnetic resonance image sets. When possible, semi-automated segmentation strategies will be used to facilitate image segmentation. Registration markers attached to femur, tibia, and patella will also be segmented. Segmented tissue volumes will be exported as triangulated surfaces[8]. Raw triangulated surfaces will be processed in MeshLab[9,10] in a series of smoothing, surface reconstruction, parametrization and resampling steps to obtain smoothed, watertight, and meshing ready representations of tissue boundaries as triangulated surfaces. Surface resampling will be at a discretization level such that edge length appears to accommodate more than three elements when thin regions of tissue are considered. Each tissue component will be meshed using SALOME[11]; bones as surface meshes composed of three node triangular shell elements and other tissues as volume meshes composed of four node tetrahedral solid elements.

Bones will be rigid. Cartilage will be represented as a nearly incompressible Neo-Hookean material[12] with coefficients adapted from literature[13]. Ligaments will be assigned nearly incompressible, transversely isotropic, hyperelastic materials with a Mooney-Rivlin ground substance[12]. Material coefficients will be identical to those used in literature[14]. In situ strain, i.e., strain at reference state, will be assigned to each ligament based on averages of values reported in literature[15]. The same constitutive model will be used for the quadriceps tendon with parameters same as that of the patellar ligament. Meniscus will also be modeled using the same constitutive model with coefficients obtained from a previous modeling study[16]. Medial and lateral patellofemoral ligaments will each be represented by two linear springs with anatomical insertions and stiffness adapted from literature[17–21].

Ligaments, tendons, and menisci will be tied to respective bones at their insertion sites. This will be accomplished by constraining nodes of the tissue at the insertion areas to move with rigid bones. Frictionless contact will be defined between articulating surfaces of cartilage (femoral and tibial; femoral and patellar) and between cartilage (femoral and tibial) and menisci. Potential wrapping of ligaments and tendons around bones and cartilage will be modeled as frictionless contact. In a similar fashion, frictionless contact between anterior and posterior cruciate ligaments will represent their potential mechanical interaction. Medial meniscus and medial collateral ligament will be tied to each other to represent their anatomical connectivity. Opposing regions of tissues to define tie constraints and contact interactions will be determined by an analysis leveraging mesh proximity and surface normals.

To facilitate prescription and interpretation of joint kinematics and kinetics, anatomically based coordinate systems will be defined on femur, tibia, and patella[22,23]. These coordinate systems will be used to define chains of cylindrical joints[22] that can be controlled in displacement or load mode. A rigid body at the proximal end of the quadriceps tendon will also be defined and connected to the femur with a sliding joint to permit prescription of quadriceps force or tendon excursion. Simulations with the model will include an initial step, to apply in situ ligament strains, followed by a loading step, to apply loading and boundary conditions representative of passive flexion[24]. Tibia and fibula will be fixed and femur and patella will be free throughout simulations. Passive flexion will be prescribed by setting angular displacement of the tibiofemoral cylindrical joint corresponding to flexion axis to change from 0º to 90º. The rest of the tibiofemoral and patellofemoral cylindrical joints will be unconstrained. Sliding joint attached to the quadriceps tendon will be free to move.

FEBio[25–27], along with FEBio PreStrain Plugin[28,29], will be used to conduct finite element analysis (solid mechanics, based on implicit static solver). Simulation results will be visualized using PostView[30] and kinematics-kinetics of the joints will be processed from raw simulation output to report joint movement during passive flexion. Python[31] and SciPy[32] will be used to automate pre-processing steps, e.g., template model generation and customization, and post-processing, i.e., to document joint movements. All modeling and simulation outputs, intermediate and final, will be publicly disseminated through an online repository[33].

Detailed Modeling and Simulation Outputs

Model Development Specifications will result in the following intermediate and final outputs. The Workflow section below provides detailed instructions on how to obtain these.

Output	**Description**	**File Format**
Segmentation	Volumetric reconstruction of tissues of interest, specifically the definition of tissue boundaries, as binary image volumes in the same coordinate system as imaging data. Tissues include bones – femur, tibia, fibula, patella; cartilage – femoral, tibial (medial & lateral), patellar; menisci – medial & lateral, ligaments – anterior/posterior cruciate, medial/lateral collateral, patellar; tendons – quadriceps.	.nii (NIfTI)[34]
Raw Geometry	Geometric reconstruction of tissues of interest as triangulated surface representations. Created from the segmentation volumes without any smoothing procedures.	.stl[8]
Smooth Geometry	Geometric reconstruction of tissues of interest as triangulated watertight surface representations. Created from raw geometry after applying volume-preserving smoothing procedures.	.stl[8]
Mesh	Finite element meshes of tissues of interest as triangulated surface (bones) or tetrahedral volume meshes (cartilage, menisci, ligaments, tendon) in binary format. Created from smooth geometry with node, face, element sets to facilitate model assembly, material property definitions, and assignment of tissue interactions.	.med[35]

Output	**Description**	**File Format**
Template Model (FEBio Input File)	XML[36] based text file (for finite element analysis with FEBio[25]) including mesh definitions, template constitutive models (rigid – bones; deformable – other tissue), template interactions between tissue, and template loading and boundary conditions (for rigid objects).	.feb[12]
Customized Model (FEBio Input File)	XML based text file (for finite element analysis with FEBio[25]) customized to include mesh definitions, tissue interactions, tissue-specific constitutive models, in situ ligament strains, representation of additional stabilizing structures, anatomical knee joint coordinate systems, specialized loading and boundary conditions to represent passive flexion, output requests relevant to knee mechanics; including numerical analysis settings.	.feb[12]
Raw Simulation Results	Binary (.xplt) and text files (.log)[12] obtained by simulation of passive flexion using customized model with FEBio[25].	.xplt[12] .log[12]
Processed Simulation Results	XML based text file storing extracted knee kinematics and kinetics during passive flexion simulations; processed using raw simulation results and supported by graphs as binary image files.	.xml[36] .png[37]

Workflow

Image Segmentation

Target Outcome

Volumetric reconstruction of tissues of interest, specifically the definition of tissue boundaries, as binary image volumes (.nii; NIfTI)[34] and raw (without smoothing) triangulated surface representations (.stl)[8] in the same coordinate system as imaging data. Tissues include bones – femur, tibia, fibula, patella; cartilage – femoral, tibial (medial & lateral), patellar; menisci – medial & lateral, ligaments – anterior/posterior cruciate, medial/lateral collateral, patellar; tendons – quadriceps.

Burden

Software requirements:

3D Slicer. 3D Slicer is a free, open source software package for visualization and image analysis (free and open source, BSD style, licensing, see http://www.slicer.org)[6]. The latest version (Slicer 4.8) will be used to access more tools and features.

Hardware requirements:

Any contemporary computer; desktop, workstation, or laptop. 3D Slicer is compatible with modern Windows, Mac OS X (10.7 and up), and various Linux distributions.

Anticipated Man Hours and Expertise Level:

1-2 weeks of full-time effort from a research engineer with bachelor’s degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to image segmentation.

Computational Cost:

Negligible compared to required manual effort.

Protocols

Input

Set(s) of MRI from Open Knee(s) – oks003 in NIfTI format (https://nifti.nimh.nih.gov/)[34] (all in the same coordinate system):

1. General Purpose MRI (3D T1-weighted without fat suppression, isotropic, 0.5 mm resolution)
2. Cartilage MRI (3D T1-weighted with fat suppression, 0.35 mm sagittal plane resolution, 0.7 mm out of plane resolution)
3. Connective tissue MRI (proton density, turbo spin echo in sagittal, axial, and coronal planes, 0.35 mm in plane and 2.8 mm out of plane resolution)

Segmentation Procedures

Segmentation Algorithms

The segmentation approach will be primarily manual, requiring the modeler to use common labeling tools, e.g. brush, pencil, etc. to paint or fill in the tissue region of interest in image volume. Depending on the tissue, multiple image volumes will be used to confirm tissue boundaries or assist tissue volume generation. Several 3D Slicer tools will also be used during the segmentation process to assist gross segmentation before manual touch up. These are described in following.

GrowCut Segmentation

From 3D Slicer documentation[38]:

> "GrowCut Segmentation is a competitive region growing algorithm using cellular automata. The algorithm works by using a set of user input scribbles for foreground and background. For N-class segmentation, the algorithm requires a set of scribbles corresponding the N classes and a scribble for a don't care class."

In this workflow, GrowCut will be used primarily for gross segmentation of large tissue volumes.

Label Map Smoothing

From 3D Slicer documentation[38]:

> "This filter smooths a binary label map. With a label map as input, this filter runs an anti-aliasing algorithm followed by a Gaussian smoothing algorithm. The output is a smoothed label map."

In the smoothing parameters outlined below, Sigma (Gaussian smoothing parameter) is chosen based on image resolution. For example, for cartilage MRI with resolution = 0.35 x 0.35 x 0.7 mm, sigma is set to 0.7. It should be noted that label map smoothing will be used to assist segmentation as an intermediate step and should not be confused with preparation and smoothing of raw geometry for meshing (see section on Geometry Generation).

Joint Smoothing

From 3D Slicer documentation[38]:

> “This will ensure that all resulting models fit together smoothly, like jigsaw puzzle pieces. Otherwise the models will be smoothed independently and may overlap.”

It should be noted that joint smoothing will be used to assist segmentation as an intermediate step and should not be confused with preparation and smoothing of raw geometry for meshing (see section on Geometry Generation).

Tissue-Specific Segmentation Procedures

Below are the guidelines for segmentation procedures based on tissue types. It is important to use the specified input MRI, and locate the boundary as described for each tissue. However, the segmentation procedures described are only suggestions, and one may choose to highlight the anatomy using whichever tool they find most effective and efficient for their manual workflow. Once all tissues are segmented, the label maps should be saved as NIfTI files (.nii)[34] and converted to raw triangulated surfaces (.stl, in units mm)[8] (without any further smoothing procedures). Slicer provides output options to accommodate these formats.

Tissue/Object	**Input MRI**	**Boundary Definition**[*]	**GrowCut Segmentation**[#]	**Manual Segmentation**	**Label Map Smoothing**[#]	**Joint Smoothing**[#]
Registration Markers[$] femoral (x3) tibial (x3) patellar (x3)	General purpose MRI	Markers will appear bright in the MRI – the outer edge of the bright region defines the boundary.	yes	yes (fill in screw holes/ bubbles)		
Bones femur tibia fibula patella	Cartilage MRI	Cortical bone will appear black in MRI – the outer edge of black region defines the bone surface.	yes		yes sigma=0.7	
Cartilage femoral tibial (medial) tibial (lateral) patellar	Cartilage MRI	Use bone boundary to help define cartilage boundary at bone interface. Cartilage will appear bright white – the outer edge defines articulating surface.	yes		yes sigma=0.7	yes (with bones)

Tissue/Object	**Input MRI**	**Boundary Definition**[*]	**GrowCut Segmentation**[#]	**Manual Segmentation**	**Label Map Smoothing**[#]	**Joint Smoothing**[#]
Menisci medial lateral	Cartilage MRI	Use cartilage boundary to help define meniscus boundary. Meniscus will appear darker.		yes	yes sigma=0.7	yes (with cartilage)
Ligaments **Tendons** ACL PCL patellar ligament quadriceps tendon	Connective tissue MRI (for boundary) General purpose MRI (for resolution)	Use bone boundary to help define connective tissue boundary at insertion and origin sites. Ligaments and tendons will appear dark.	yes Phase 1: using sagittal connective tissue MRI	yes Phase 2: using general purpose MRI, overlay labelmap from Phase 1 (as foreground layer) and trace boundary[&]		yes (with bones)
Ligaments LCL MCL	General purpose MRI (for boundary) Connective tissue MRI (for confirmation)	Use bone boundary to help define ligaments in coronal plane. Ligaments will appear dark.		yes	yes sigma=0.5	

ACL: anterior cruciate ligament. PCL: posterior cruciate ligament. LCL: lateral collateral ligament. MCL: medial collateral ligament.

[*]Refer to sample images below to help define boundaries.

[#]Always check the image slices manually after using GrowCut Segmentation or Label Map and Joint Smoothing, and perform touch ups as needed to better define the boundaries.

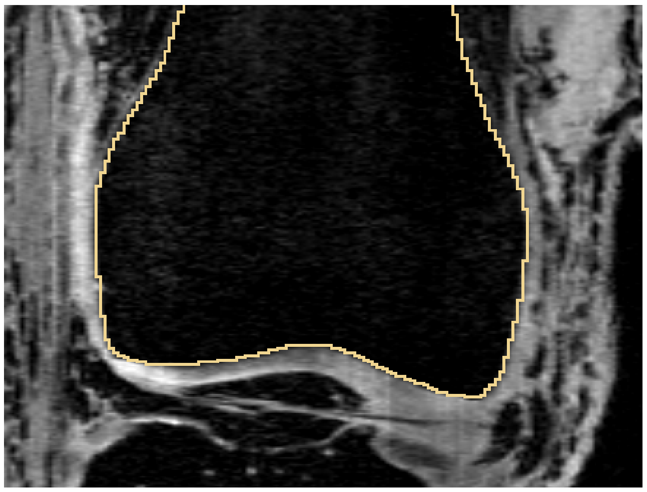
[$]While segmentation of reference markers are not needed to obtain an initial model, they are included in the workflow with the anticipation of their utility in upcoming modeling and simulation stages.

[&]Alternatively, one can display the connective tissue MRIs and the general purpose OR cartilage MRI in different windows. When linked, Slicer uses interpolation for coupled viewing of the image sets that are already spatially aligned. In return, one can do the segmentation on interpolated connective tissue MRIs using the general purpose OR cartilage MRI as the master volume for segmentation. This allows high resolution segmentation volume from images with lower resolution directly.

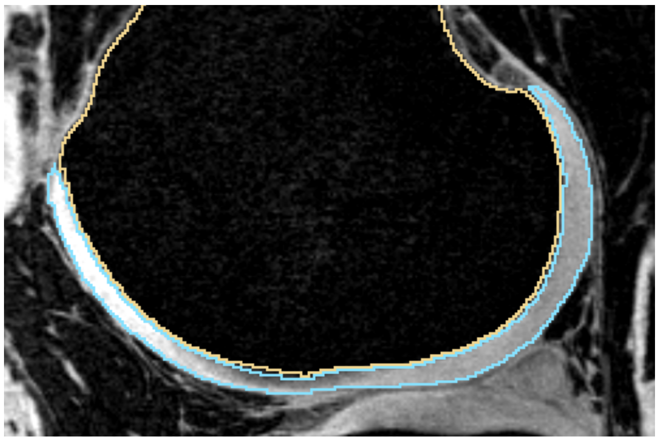
Sample Images

These images are provided in order to assist with boundary definitions of tissues.

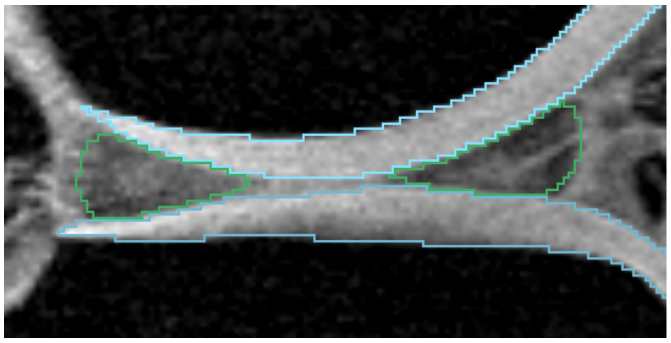
<u>Bone boundaries in cartilage MRI:</u>



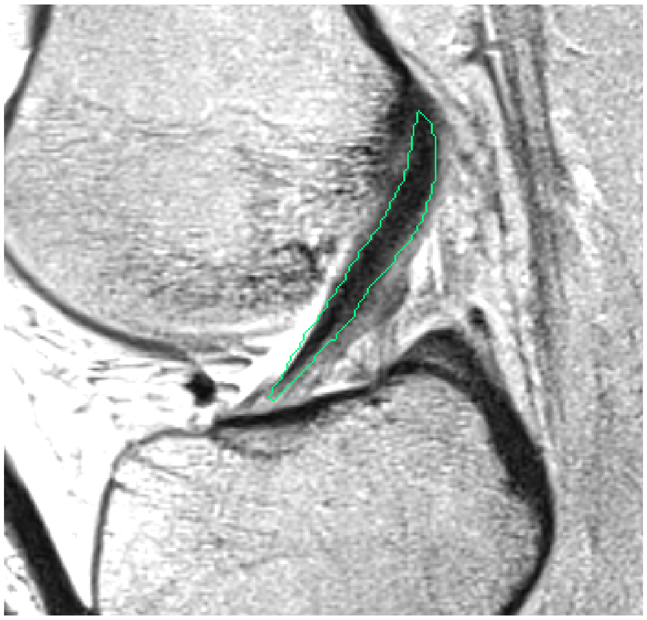
<u>Bone and cartilage boundaries in cartilage MRI:</u>



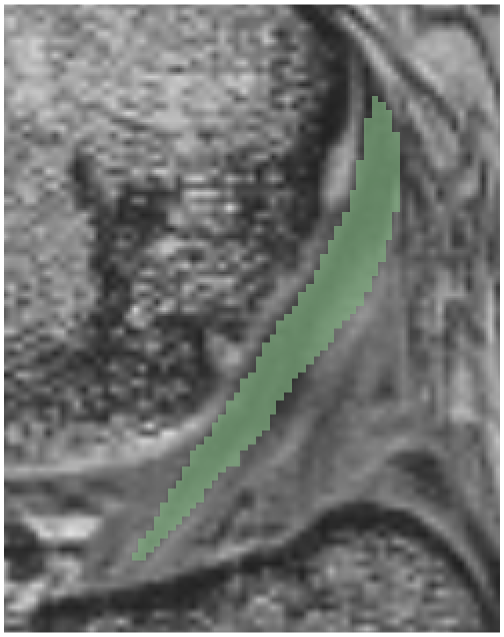
<u>Cartilage and meniscus boundaries in cartilage MRI:</u>



Ligament boundaries in connective tissue MRI:



Ligament boundaries in general purpose MRI:



Geometry Generation

Target Outcome

Geometric reconstruction of tissues of interest as smooth and watertight triangulated surface representations (.stl)[8] obtained from and in the same coordinate system as raw geometry; ready for volumetric meshing. Tissues include bones – femur, tibia, fibula, patella; cartilage – femoral, tibial (medial & lateral), patellar; menisci – medial & lateral, ligaments – anterior/posterior cruciate, medial/lateral collateral, patellar; tendons – quadriceps.

Burden

Software requirements:

MeshLab. MeshLab is an open source, portable, and extensible system for processing and editing of unstructured 3D triangular meshes (free and open source GPL license, see http://www.meshlab.net/)[9]. The latest version of MeshLab (MeshLab 2016.12) will be used.

Hardware requirements:

Any contemporary computer; desktop, workstation, or laptop. MeshLab is available for Windows, Mac OS X, and Linux.

Anticipated Man Hours and Expertise Level:

Up to a week of full-time effort from a research engineer with bachelor's degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to geometry processing.

Computational Cost:

Minimal compared to required interactions with software.

Protocols

Input

Raw triangulated surface representations of tissues of interest (without filtering and smoothing) in .stl format in image coordinate system.

Surface Processing Procedures

A 5-step procedure (LVTIT, see below for descriptions) will be used to process raw triangulated surface meshes. The process includes staged smoothing approaches interleaved with surface reconstruction and resampling. The procedures aim to generate uniform and watertight surface meshes that are smooth, volume-preserving and visually maintaining the geometrical shape of the tissues.

Smoothing Algorithms

This section provides brief descriptions of the MeshLab algorithms that will be used during processing of triangulated surface meshes.

Laplacian Smoothing [L]

For each vertex in the mesh, a new position is chosen based on average position with nearest vertex (as described in built-in documentation in MeshLab[9]). The "Smoothing steps" parameter is the number of times the process is repeated.

Surface Reconstruction: VCG [V]

From built-in documentation in MeshLab[9]:

> "Surface reconstruction algorithm that have been used for a long time inside the ISTI-Visual Computer Lab. It is mostly a variant of the Curless at al. e.g. a volumetric approach with some original weighting schemes, a different expansion rule, and another approach to hole filling through volume dilation/relaxations."

"Voxel Side" values should informed in relation to original image resolution from which raw surfaces are obtained; e.g., for cartilage images with a resolution of 0.35 x 0.35 x 0.7 mm, use a world unit of 0.35, 0.5, or

0.7 mm.

Taubin Smoothing [T]

From built-in documentation in MeshLab[9]:

> “The λ-μ Taubin smoothing, it makes two steps of smoothing, forth and back, for each iteration. Based on Gabriel Taubin, A signal processing approach to fair surface design, Siggraph 1995”

Iso Parameterization [I]

Stage 1: Iso Parameterization

From built-in documentation in MeshLab[9]:

> “The Filter builds the abstract Isoparameterization of a two-manifold triangular mesh. An adaptively chosen abstract domain of the parameterization is built. For more details see: Pietroni, Tarini, and Cignoni, 'Almost isometric mesh parameterization through abstract domains' IEEE Transaction of Visualization and Computer Graphics 2010”

Stage 2: Iso Parameterization Remeshing

From built-in documentation in MeshLab[9]:

> “Remeshing based on Abstract Isoparameterization, each triangle of the domain is recursively subdivided. For more details see Pietroni, Tarini, and Cignoni, 'Almost isometric mesh parameterization through abstract domains' IEEE Transaction of Visualization and Computer Graphics 2010”

“Sampling Rate” determines the number of subdivisions, aka. density of the surface mesh.

Tissue-Specific Processing Parameters

The table below specifies parameters to be used in the surface mesh processing for each tissue. These values are considered a starting point and the process should be performed in the order provided (if ‘-’, skip step). For parameters that were not specified MeshLab defaults should be used. The outcome of surface mesh processing may need to be checked (visually and when possible, quantitatively) to ensure that there are no significant loss of geometric features or volumes. If necessary, surface repairing should be performed, i.e. using other MeshLab tools, to ensure a manifold and watertight surface; faces may need to be re-oriented such that surface normals all point outward. Final surface representation of the tissue should be exported (.stl, in units mm).

	1. Laplacian Smoothing (Smoothing Steps)	**2. VCG Surface Reconstruction (Voxel Side)**	**3. Taubin Smoothing (Lambda, mu)**	**4. Iso Parameterization (Sampling Rate)***	**5. Taubin Smoothing (Lambda, mu)**
Registration Markers	-	-	-	-	-
Femur, Tibia	20	0.7	0.5,-0.53	10	0.5,-0.53
Patella	20	0.5	0.5,-0.53	7	0.5,-0.53
Fibula	20	0.5	0.5,-0.53	5	0.5,-0.53

	1. Laplacian Smoothing (Smoothing Steps)	**2. VCG Surface Reconstruction (Voxel Side)**	**3. Taubin Smoothing (Lambda, mu)**	**4. Iso Parameterization (Sampling Rate)***	**5. Taubin Smoothing (Lambda, mu)**
Femoral Cartilage	20	0.35	0.5,-0.53	[10,15,20]	0.5,-0.53
Tibial Cartilage	20	0.35	0.5,-0.53	[7,10,13]	0.5,-0.53
Patellar Cartilage	20	0.35	0.5,-0.53	[(6),8,10,(12)]	0.5,-0.53
Menisci	20	-	-	[6,8,10]	0.5,-0.53
ACL,PCL	20	0.35	0.5,-0.53	[3,4,5]	0.5,-0.53
Patellar Ligament	20	0.35	0.5,-0.53	[6,8,10]	0.5,-0.53
Quadriceps Tendon	20	0.35	0.5,-0.53	[4,6,8]	0.5,-0.53
MCL	20	0.35	-	[6,7,8]	0.5,-0.53
LCL	20	-	-	[3,4,5]	0.5,-0.53

ACL: anterior cruciate ligament. PCL: posterior cruciate ligament. LCL: lateral collateral ligament. MCL: medial collateral ligament.

*Within brackets multiple mesh densities are provided as a guide such that the user can determine density of the resampled surface mesh based on the the coarsest mesh that (1) maintains the geometry of the tissue, without visibly losing any features, and (2) edge length of the surface mesh appears to fits at least 3 triangular elements across the cross section of the thinnest part of the tissue.

Template Model Generation

Target Outcome

Template model for finite element analysis in FEBio[25] format (.feb, XML[36] based text file)[12] including mesh definitions (nodes, elements, surfaces; node, element, face sets), template constitutive models (rigid – bones; deformable – other tissue), template interactions between tissue (tie constraints, contact), and template loading and boundary conditions (for rigid objects); created from geometric reconstruction of tissues. Intermediate outputs will include finite element meshes of tissues of interest as triangulated surface (bones) or tetrahedral volume meshes (cartilage, menisci, ligaments, tendon) in binary format (.med)[35] with node, face, element sets to facilitate model assembly, material property definitions, and assignment of tissue interactions.

Burden

Software requirements:

SALOME. SALOME is an open-source software that provides a generic platform for pre- (cad, meshing) and post-processing for numerical simulation (free and open source LGPL license, see http://www.salome-platform.org/)[11]. SALOME includes built-in scripting functionality using Python[31], which will be required to utilize Python scripts for mesh generation and annotation, and model assembly. SALOME 7.8.0 will be used to support in-house Python scripts.

StlToMed.py. In-house script to generate meshes (surface or volume) and node, element, face sets using an XML based input file that points to .stl files, e.g., tissue geometries, and their connectivity. To be used with

SALOME's built-in Python installation; developed using SALOME 7.8.0, source code available at https://simtk.org/svn/multis/utl/ModelAssembly/[39].

MedToFebio.py. In-house script to generate a template FEBio model using XML based connectivity file and meshes, e.g. output of StlToMed.py. To be used with SALOME's built-in Python installation; developed using SALOME 7.8.0, supports FEBio file format version 2.5[12], source code available at https://simtk.org/svn/multis/utl/ModelAssembly/.

ConnectivityXML.py. Python utility script to read an XML file that points to .stl files and their connectivity for model assembly. Source code available at https://simtk.org/svn/multis/utl/ModelAssembly/[39].

Hardware requirements:

Any contemporary computer; desktop, workstation, or laptop. While Windows version of SALOME exists, Linux is preferable.

Anticipated Man Hours and Expertise Level:

1-2 days of full-time effort from a research engineer with bachelor's degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to model generation.

Computational Cost:

Minimal compared to required interactions with software.

Protocols

Input

Geometric reconstruction of tissues of interest – smooth and watertight triangulated surface representations (.stl)[8], all in the same coordinate system, e.g., image coordinate system.

Template Model Generation

Generation of the template model will be staged in three steps. The first step requires description of model components and their interactions with each other, essentially a model definition tree, as an input file to Python scripts. In following, Python scripts will be executed using SALOME in sequence, to first generate meshes and then to assemble the model and generate an FEBio model file.

Model Definition

A connectivity file to use with the aforementioned Python scripts (e.g., with ConnectivityXML.py) will be created. This file will include references to all tissue components that will be part of the model: bones – femur, tibia, fibula, patella; cartilage – femoral, tibial (medial & lateral), patellar; menisci – medial & lateral, ligaments – anterior/posterior cruciate, medial/lateral collateral, patellar; tendons – quadriceps. Each tissue entry will point to the file location of triangulated surface representation of the tissue boundary. Each tissue will be referred as "rigid" (bones) or "elastic" (other tissues) to generate relevant place holders for material definitions and loading and boundary conditions during model assembly. In addition, interactions between tissues will be defined as "Tie" or "Contact" to define kinematic constraints or contact, respectively, between opposing regions of tissues. Determination of surfaces, e.g. face sets, and/or node sets for these constraints will be based on geometric principles relating to the pair of meshes based on:

- Proximity – find all the nodes on part 1 that are within a prescribed distance (related to the element size) of any node on part 2.
- Normals – select faces on part 1 that have normal vectors that point toward the barycenter of part 2. This can also be limited by the proximity condition if desired.
- Contains – one mesh contains the other, select elements that have normal vectors that point inward or outward.
- All – all the surfaces.

This analysis will be automated using the Python scripts with SALOME during mesh generation and annotation. The following table specifies all the tie constraints and contacts surfaces between tissues, and which geometrical principles should be used to relate them.

	FM	**TB**	**FB**	**PT**	**FC**	**TLC**	**TMC**	**PC**	**MM**	**LM**	**ACL**	**PCL**	**MCL**	**LCL**	**PL**	**QT**
FM					T:p						T:p	T:p	C:a T:p	C:n T:p		C:n
TB						T:p	T:p		T:p	T:p	T:p	T:p	C:a T:p		T:p	
FB														T:p		
PT								T:p							T:p	T:p
FC	T:p					C:n	C:n	C:n	C:n	C:n						C:n
TLC		T:p			C:a					C:n						
TMC		T:p			C:a				C:n							
PC				T:p	C:n											
MM		T:p			C:n		C:n						T:p			
LM		T:p			C:n	C:n										
ACL	T:p	T:p										C:p				
PCL	T:p	T:p									C:p					
MCL	C:a T:p	C:a T:p							T:p							
LCL	C:n T:p		T:p													
PL	T:p			T:p												
QT	C:n			T:p	C:n											

FM: femur. TB: tibia. FB: fibula. PT: patella. FC: femoral cartilage. TLC: tibial lateral cartilage. TMC: tibial medial cartilage. PC: patellar cartilage. LM: lateral meniscus. MM: medical meniscus. ACL: anterior cruciate ligament. PCL: posterior cruciate ligament. MCL: medial collateral ligament. LCL: lateral collateral ligament. PL: patellar ligament. QT: quadriceps tendon. – C: Contact. T:Tie. p: proximity, n: normals. c: contains. a: all.

Mesh Generation and Annotation

Use of StlToMed.py with SALOME's built-in Python[11] and the previously described connectivity file will result in meshes of the tissue components along with node, element and surface definitions and node, element, face sets (for material property assignment, tie and contact definitions, and loading and boundary assignment). The script is automated, resulting in mesh files (in .med format[35] and in units of the input files) for each tissue component. Script starts with generating triangular surface meshes (for rigid objects – bones) and tetrahedral (four node) volume meshes (for elastic objects – other tissues) using geometric reconstructions of the tissue (.stl). Surface mesh discretization for rigid objects is identical to that of the input .stl file. Surface discretization of elastic objects is identical to that of the input .stl file. Volume meshing utilizes SALOME's interface to NETGEN 3D mesh generator[40] with the following settings (see SALOME NETGEN documentation for more details[41]):

Name	**Definition**	**Value**
Max Size	maximum linear dimensions for mesh cells.	Equal to the maximum linear dimension of the input STL
Min Size	minimum linear dimensions for mesh cells.	Equal to the minimum linear dimension of the input STL
Fineness	Ranging from *Very Coarse* to *Very Fine* allows to set the level of meshing detailization.	Level 4. (If this fails, use Level 3)
Optimize	the algorithm will modify initially created mesh in order to improve quality of elements. Optimization process is rather time consuming comparing to creation of initial mesh.	On

The script will automatically generate an element set including all elements within the tissue, which will later be used to assign material properties. Node and surface regions for tie and contact definitions will be determined based on previously noted "Proximity", "Normals", or "All" settings denoted in the connectivity file. For "Proximity" a multiplier value of 1.0 will be used to scale approximate characteristic length of the surface mesh to determine the proximity threshold. For "Normals" setting a multiplier will not be used.

Meshes of tissues will be visually inspected in SALOME[11]. This inspection will facilitate confirmation of appropriate definitions of node and surface regions to prescribe tissue connectivity, e.g., ligament insertion and origin sites, and contact, e.g. between cartilage. If necessary these sets will be interactively edited in SALOME by adding/removing nodes, faces, etc. from groups. Quantitative mesh analysis will also performed to assess mesh quality, e.g. element aspect ratios. Mesh convergence will not be conducted as part of the *Model Development* phase; it will likely be visited during the upcoming stages of modeling & simulation through the model's lifecycle.

Model Assembly

Use of MedToFebio.py with SALOME's built-in Python[11], the previously described connectivity file, and mesh files (outcome of mesh generation and annotation step) will result in a template FEBio model file (.feb, version 2.5)[12] for finite element analysis. The script is automated and it will result in the following contents written in the model file:

- Node definitions – list the node coordinates for each tissue
- Element definitions – list the node connections to define the elements for each tissue

- Node sets, face sets, element sets, surface definitions – groups of nodes, elements, faces labeled to be used in tie and contact pairs
- Material definitions
  - For rigid bodies (bones) only a place holder density is assigned (1e-9 tonnes/mm^3 – consistent with spatial units of mm)
  - For elastic bodies (other tissues) a place holder density is assigned as 1e-9 tonnes/mm^3 and a Neo-Hookean material is defined using FEBio material type Mooney-Rivlin (uncoupled)[12] and setting c1 = 0.1 MPa; c2 = 0 MPa; k = 100 MPa (consistent with spatial units of mm) as a place holder.
- Surface pairs – to define connections (ties) or interactions between the surfaces, specifies a "master" surface, and "slave" surface by using the following criteria:
  - The surface on a rigid body should be the master surface.
  - The larger of the two surfaces should act as the master surface.
  - If the surfaces are of comparable size, the surface on the stiffer body should act as the master surface.
  - If the surfaces are of comparable size and stiffness, the surface with the coarser mesh should act as the master surface.
- Tied interfaces – "tied" type contact between deformable surface pairs (penalty = 10)[12]
- Contact interactions – "facet-to-facet-sliding" type contact (called "sliding-facet-on-facet" in recent versions of FEBio) between surface pairs (frictionless, laugon = 1, penalty = 1, auto_penalty = 1, maxaug = 10)[12]
- Loading and boundary conditions
  - Assignment of nodes on deformable bodies, e.g. ligament insertion and origin sites, to relevant rigid body as boundary conditions to tie the nodes to rigid bodies.[12]
  - Setting of six degrees-of-freedom kinematics (translations and rotations) of all rigid bodies to fixed as a place holder. Note that FEBio assigns these boundary conditions at the center of mass of the rigid body.[12]
- Load Data – define the load curves that will be used; "linear", and "steps"[12]
- Output requests – "contact gap", "contact pressure", "contact traction", "displacement", "reaction forces", "stress"[12]
- Module – set to "solid" for solid mechanics analysis
- Step and control parameters – selected parameters to set simulation configuration for implicit static analysis using quasi-Newton incrementation with output at desired intervals using two simulation steps ("Initialize" and "LoadingStep") as place holders.[12]

- time_steps = steps (default for steps is 10)
- step_size = 1./steps
- dtmin = 1e-10
- dtmax = 1./steps
- max_retries = 20
- opt_iter = 10
- aggressiveness = 1
- optimize_bw = 1
- plot_level = PLOT_MUST_POINTS
- analysis = static
- min_residual = 1e-10

Many of the FEBio parameters set above are default values. For parameters that are not specified, FEBio defaults[12] will be used. Iterations of these parameters are possible depending on convergence of simulations.

Customization for Material Definitions

Target Outcome

Updated model for finite element analysis in FEBio[25] format (.feb, XML[36] based text file)[12] with modified tissue-specific constitutive models relevant to knee mechanics; generated from template model file (.feb).

Burden

Software requirements:

Python. Python is a high-level multi-platform programming language (free and open source GPL compatible Python Software Foundation license, see https://www.python.org)[31]. Any contemporary version available for the computing platform can be used; 3.7.0 is the latest version at the time of the preparation of this document.

SciPy. SciPy is a Python based open source software platform for mathematics, science and engineering (free and open source BSD-new license, see https://www.scipy.org)[32]. Any contemporary version available for the computing platform can be used; 1.1.0 is the latest version at the time of the preparation of this document.

Hardware requirements:

Any contemporary computer; desktop, workstation, or laptop. Python is a multi-platform programming language for Windows, Mac OS X, and Linux.

Anticipated Man Hours and Expertise Level:

Less than an hour of full-time effort from a research engineer with bachelor’s degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to Python scripting and model generation.

Computational Cost:

Minimal compared to required interactions with computer scripts.

Protocols

Input

Template (or customized) model in FEBio format (.feb) and tissue constitutive models and parameters relevant to knee mechanics (described below) as a text based input file.

Scripting

A Python script will be developed to modify an existing template model with desired tissue constitutive models.

Constitutive Models and Parameters

Material properties and the constitutive models for *Model Development* phase are adapted from literature as described below. It should be noted that these parameters and the constitutive representations may me modified in upcoming stages of modeling and simulation through the lifecycle of the model.

Bone

All bones (femur, tibia, fibula, patella) will be assumed to be rigid bodies. Bones have a much higher stiffness than the other knee tissues. Rigid body assumption simplifies computation, therefore decreasing the computational cost and facilitates definition of joint kinematics and/or kinetics as loading and boundary conditions. A density of 1e-9 tonnes/mm^3 (identical to water; consistent with spatial units of mm). It should be noted that density assignment will not have importance on static simulations without the action of gravity. It is provided as a place holder.

Cartilage

Cartilage will be modeled as a nearly incompressible Neo-Hookean material defined by FEBio setting C2 parameter of FEBio material type Mooney-Rivlin (uncoupled)[12]. This is a fairly simplified representation of cartilage's mechanical behavior[42]. We believe that it would be adequate and computationally less challenging for joint level simulations while providing an opportunity to understand local mechanical environment on and within the cartilage. Cartilage constitutive model and coefficients will be similar to previous modeling study[13], which reported an elastic modulus of 15 MPa and Poisson's ratio of 0.475. Corresponding Neo-Hookean material coefficients are noted below.

Density[*] (tonnes/mm^3)	**C1 (MPa)**	**C2 (MPa)**	**K (MPa)**
1e-9	2.54	0	100

All units are consistent with spatial units of mm.

[*]Density assignment is a place holder; it will not have importance on static simulations without the action of gravity.

Ligaments and Tendons

Ligaments and tendons will be modeled as nearly incompressible, transversely isotropic, hyperelastic material with a Mooney-Rivlin ground substance (Neo-Hookean by setting C2 = 0)[12]. This type of representation accommodates tensile dominant behavior of the ligaments dictated by their fiber alignment across their longitudinal axis[43]. The parameters will be identical to a previous modeling study[14], which fitted data from literature. These values are noted below.

Ligament	**Density*** **(tonnes/mm^3)**	**C1 (MPa)**	**C2 (MPa)**	**K# (MPa)**	**C3 (MPa)**	**C4**	**C5 (MPa)**	**λ_m**
ACL	1e-9	1.95	0	146.41	0.0139	116.22	535.039	1.046
PCL	1e-9	3.25	0	243.9	0.1196	87.178	431.063	1.035
MCL	1e-9	1.44	0	793.65	0.57	48.0	467.1	1.063
LCL$	1e-9	1.44	0	793.65	0.57	48.0	467.1	1.063
PL	1e-9	2.75	0	206.61	0.065	115.89	777.56	1.042
QT&	1e-9	2.75	0	206.61	0.065	115.89	777.56	1.042

All units are consistent with spatial units of mm.

ACL: anterior cruciate ligament. PCL: posterior cruciate ligament. LCL: lateral collateral ligament. MCL: medial collateral ligament. PL: patellar ligament. QT: quadriceps tendon.

*Density assignment is a place holder; it will not have importance on static simulations without the action of gravity.

#Bulk modulus (K) is calculated as (K = 1/D); D obtained from source literature.

$LCL properties were assumed to be identical to MCL.

&QT properties were assumed to be identical to PL. This assumption should not have significance as anticipated use of QT is to transfer loads to patella in a distributed manner.

Constitutive modeling of the ligaments requires specification of a fiber direction. In FEBio, under the material definition, fiber type can be specified as “vector” so that the initial direction of all fibers in the material point are along the direction of the vector[12]. For each ligament and tendon this direction will be defined as the direction of the longest edge of the oriented bounding box of the tissue to approximate the longitudinal alignment of the tissue.

Meniscus

Menisci will be modeled as nearly incompressible, transversely isotropic, hyperelastic material with a Mooney-Rivlin ground substance (Neo-Hookean by setting C2 = 0)[12]. This material model is seemingly more complicated than transversely orthothropic linear elastic models in literature. Yet, it will allow a convenient way to represent the behavior of meniscus which is largely dictated by its circumferential stiffness (based on fiber alignment) with the capacity to sustain compressive loading[44]. For convenience, the parameters will be identical to those used in a previous modeling study of meniscus[16] and they are noted below.

	Density* **(tonnes/mm^3)**	**C1 (MPa)**	**C2 (MPa)**	**K# (MPa)**	**C3 (MPa)**	**C4**	**C5 (MPa)**	**λ_m**
Meniscus	1e-9	4.61	0	92.16	0.1197	150.0	400.0	1.019

All units are consistent with spatial units of mm.

*Density assignment is a place holder; it will not have importance on static simulations without the action of gravity.

#Bulk modulus (K) is calculated as (K = 1/D); D obtained from source literature. It should be noted that due to differences in dilatational component of constitutive models (used in here and in source literature), equivalence of K is an approximation.

Constitutive modeling of the menisci requires specification of fiber direction. In FEBio, fiber direction can be specified for each element individually, in the ElementData section[12]. The meniscus fibers will be oriented in the circumferential direction. To accomplish this, a best-fit circle will be calculated to represent the shape of the meniscus in the transverse plane. The fibers in each element will be oriented so that their direction will be in parallel with the tangent of the closest point on the circle.

Customization for Other Stabilizing Components

Target Outcome

Updated model for finite element analysis in FEBio[25] format (.feb, XML[36] based text file)[12] with additional joint stabilizers and convenience structures relevant to simulation of knee mechanics; generated from template model file (.feb).

Burden

<u>Software requirements:</u>

Python. Python is a high-level multi-platform programming language (free and open source GPL compatible Python Software Foundation license, see https://www.python.org)[31]. Any contemporary version available for the computing platform can be used; 3.7.0 is the latest version at the time of the preparation of this document.

SciPy. SciPy is a Python based open source software platform for mathematics, science and engineering (free and open source BSD-new license, see https://www.scipy.org)[32]. Any contemporary version available for the computing platform can be used; 1.1.0 is the latest version at the time of the preparation of this document.

<u>Hardware requirements:</u>

Any contemporary computer; desktop, workstation, or laptop. Python is a multi-platform programming language for Windows, Mac OS X, and Linux.

<u>Anticipated Man Hours and Expertise Level:</u>

Less than an hour of full-time effort from a research engineer with bachelor's degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to Python scripting and model generation.

<u>Computational Cost:</u>

Minimal compared to required interactions with computer scripts.

Protocols

Input

Template (or customized) model in FEBio format (.feb) and parameters for stabilizing/convenience structures relevant to knee mechanics (described below) as a text based input file.

Scripting

A Python script will be developed to modify an existing template model with desired additional model components.

Components for Stabilization

The following components are not tissues of interest, however they will be incorporated in the model primarily to stabilize patella and represent the extensor mechanism. Mechanical springs and kinematic joints will be used to represent these anatomical structures.

Quadriceps Tendon Attachment

The quadriceps tendon will be attached to the femur proximally. Since the quadriceps muscles are not being modeled, this attachment will be represented by a slider joint. In order to define this slider joint, a rigid cylindrical joint (with prescribed rotation set to zero)[12] will be added to the FEBio model input file. This joint will connect two rigid bodies. Two imaginary rigid bodies will be defined for this purpose. The center of mass of both rigid bodies will be at the approximate center of the cross section of the quadriceps tendon at the most proximal end of the tendon. One rigid body will be fixed to the proximal end of the quadriceps tendon by including nodes at the proximal end of the tissue to the rigid body as a boundary condition. The other rigid body will be fixed to the femur. The axis of the slider joint will be defined as passing through the rigid bodies center of mass, and parallel to the anatomical axis of the femur. When needed, this joint can be used to prescribe quadriceps force.

Patellofemoral Joint Ligaments

The medial and lateral patellofemoral ligaments (MPFL, LPFL) will be modeled using linear springs. The insertion points of the patellofemoral ligaments will be located as follows:

- MPFL femoral insertion: 0.5xCD from the distal side of the medial condyle, and 0.4xCD from the posterior side or the medial condyle, where CD is the anterior-posterior size of the medial condyle[17]
- MPFL patellar insertion: superomedial aspect of patella (~ top 1/3)[18]
- LPFL femoral insertion: 10.6 mm anterior, and 2.6 mm distal to the lateral epicondyle, average width 11.7 mm[19]
- LPFL patellar insertion: 8 mm from superior pole to upper insertion, insertion width ~ 45% of articular surface.

Two springs will be used for each patellofemoral ligament. They will be placed such that the insertion points of the springs span the widths of the insertions described above. Coordinates of anatomical locations, e.g, femoral epicondyles and patellar regions, will be located in the MRI if their identification proves to be difficult to find on the meshes. Spring constants will be determined by equally dividing total stiffness of the ligament to individual springs. Based on previous reported values, total stiffness of MPFL will be set to 100 N/mm[20]; total stiffness of LPFL will be set to 16 N/mm[21]. The spring constants may be modified in upcoming stages of modeling and simulation depending on the desired fidelity and authenticity of patellofemoral joint response.

Customization for In Situ Ligament Strain

Target Outcome

Updated model for finite element analysis in FEBio[25] format (.feb, XML[36] based text file)[12] for implementation of in situ (reference) ligament strains relevant to knee mechanics; generated from template model file (.feb).

Burden

Software requirements:

Python. Python is a high-level multi-platform programming language (free and open source GPL compatible Python Software Foundation license, see https://www.python.org)[31]. Any contemporary version available for the computing platform can be used; 3.7.0 is the latest version at the time of the preparation of this document.

SciPy. SciPy is a Python based open source software platform for mathematics, science and engineering (free and open source BSD-new license, see https://www.scipy.org)[32]. Any contemporary version available for the computing platform can be used; 1.1.0 is the latest version at the time of the preparation of this document.

Hardware requirements:

Any contemporary computer; desktop, workstation, or laptop. Python is a multi-platform programming language for Windows, Mac OS X, and Linux.

Anticipated Man Hours and Expertise Level:

Less than an hour of full-time effort from a research engineer with bachelor’s degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to Python scripting and model generation.

Computational Cost:

Minimal compared to required interactions with computer scripts.

Protocols

Input

Template (or customized) model in FEBio format (.feb) and in situ (reference) ligament strains relevant to knee mechanics (described below) as a text based input file.

Scripting

A Python script will be developed to modify an existing template model to implement in situ strain feature with desired level of reference strains. Prescription of in situ strain will rely on FEBio PreStrain Plugin specifications[28,29].

Application of In Situ Strain

Prestrain formulation of FEBio PreStrain Plugin is described in detail in literature[29]; from this literature:

> “The gradient of the local mapping from the stress-free to the prestressed reference configuration is represented by F_p, which will be referred to as the *prestrain gradient*. The total elastic deformation gradient Fe is determined by the composited deformation gradient, $F_e = F F_p$ ”

Prestrain type will be defined as “in-situ stretch”, where the prestrain gradient is calculated based on the initial fiber stretch. The in situ stretch can be defined at the element level in the ElementData section of the FEBio input file, or the initial stretch can be defined for all fibers in one ligament. We will use the isochoric prestrain generator option, as the material will be assumed to be incompressible. The update rule type will be chosen as “prestrain”, meaning that F_p will be updated from the initial prestrain gradient given, to eliminate distortion due to incompatibility with the reference geometry (as opposed to enforcing the given in-situ stretch, resulting in possible distortion of the geometry). We will apply the prestrain using an elimination of distortion approach, where the goal is to eliminate distortion induced by the incompatibility of the initial prestrain gradient[29].

One average initial stretch will be defined for all fibers in each ligament as:

Ligament	**Initial Stretch**
ACL*	1.016
PCL#	1.0
MCL*	1.034
LCL*	1.027
PL#	1.0
QT#	1.0

1.0 indicates strain free initial state

*Initial strain was set to average of values reported in the modeling study of Dhaher et al.[15], who referred to previous knee models[14] and experimental data[45].

#Initial strain set to zero due to lack of data, similar to Dhaher et al.[15]

Since the prestrain update rule is chosen to eliminate distortion, the above values would only be used as an initial guess for fiber stretch. Due to changes in cross sectional area of the ligament, in order to maintain equilibrium, the solver will update the fiber stretches as needed. Due to this fact, it may be unnecessary for the modeler to define "anterior", "posterior" etc. regions (as done in previous studies[14,15]), as the values are likely to change. Thus, one average value will be given as the initial guess for fiber stretch for all regions in the ligament. After the solver determines the updated initial stretch values, they can be compared to the above literature values, and the average initial guess may be calibrated in upcoming modeling stages, if necessary.

Customization for Joint Coordinate System Definitions

Target Outcome

Updated model for finite element analysis in FEBio[25] format (.feb, XML[36] based text file)[12] for implementation of anatomically based coordinate systems and kinematic chains relevant to knee mechanics; generated from template model file (.feb).

Burden

Software requirements:

Python. Python is a high-level multi-platform programming language (free and open source GPL compatible Python Software Foundation license, see https://www.python.org)[31]. Any contemporary version available for the computing platform can be used; 3.7.0 is the latest version at the time of the preparation of this document.

SciPy. SciPy is a Python based open source software platform for mathematics, science and engineering (free and open source BSD-new license, see https://www.scipy.org)[32]. Any contemporary version available for the computing platform can be used; 1.1.0 is the latest version at the time of the preparation of this document.

Hardware requirements:

Any contemporary computer; desktop, workstation, or laptop. Python is a multi-platform programming language for Windows, Mac OS X, and Linux.

Anticipated Man Hours and Expertise Level:

Less than an hour of full-time effort from a research engineer with bachelor's degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to Python scripting and model generation.

Computational Cost:

Minimal compared to required interactions with computer scripts.

Protocols

Input

Template (or customized) model in FEBio format (.feb) and definition and parameters of anatomical coordinate systems and kinematic joints relevant to knee mechanics (described below) as a text based input file.

Scripting

A Python script will be developed to modify an existing template model with desired anatomical coordinate systems on the bones and kinematic chains in between.

Locating Joint Axes

Tibiofemoral Joint

Grood and Suntay Joint Coordinate System[22] (JCS) will be used to define this joint. This method requires a definition of a tibial coordinate system (x_T,y_T,z_T), a femoral coordinate system (x_F,y_F,z_F), and a floating axis (F_{TF}). The following anatomical landmarks will be located on the meshes of the femur and tibia:

- medial tibial spine (intercondylar eminence)
- lateral tibial spine (intercondylar eminence)
- approximate center of medial tibial plateau
- approximate center of lateral tibial plateau
- most distal point on the posterior surface of the femur, midway between the medial and lateral condyles
- medial femoral condyle (most distal point on posterior surface)
- lateral femoral condyle (most distal point on posterior surface)

Tibial Coordinate System:

Tibial Origin: Mid-point between tibial spines (medial and lateral intercondylar eminences).

Tibial Mechanical Axis (z_T-axis): JCS[22] defines this axis as passing through the midpoint between the tibial spines proximally, and and the center of the ankle distally. Due to a lack of MRI of the ankle, we will assume this axis to pass through the same point proximally (midpoint between spines), and extend parallel to the z-direction of the MRI coordinate system (approximately aligned with the longitudinal axis of the body). The z-axis is positive in the proximal direction.

Tibial Anterior Axis (y_T-axis): The cross product between the z_T-axis, and a line connecting the approximate center of each tibial plateau. The y-axis is positive in the anterior direction.

Tibial Mediolateral Axis (x_T-axis): The cross product between the y_T-axis and the z_T-axis.

Femoral Coordinate System:

Femoral Origin: Most distal point on the distal femur, midway between the medial and lateral condyles.

Femoral Mechanical Axis (z_F-axis): JCS[22] defines this axis as passing through the center of the femoral head proximally, and the most distal point on the posterior surface of the femur distally. Due to lack of MRI data of the femoral head, we will assume the axis to pass through the same point distally (distal point on posterior surface), and extend parallel to the z-direction of the MRI coordinate system (approximately aligned with the longitudinal axis of the body). The z_F-axis is positive in the proximal direction.

Femoral Anterior Axis (y_F-axis): The cross product between the femoral mechanical axis (z_F-axis) and a line connecting the femoral condyles. The y_F-axis is positive in the anterior direction.

Flexion Axis (x_F-axis): The cross product between the y_F-axis and the z_F-axis.

Tibiofemoral Floating Axis:

The tibiofemoral floating axis (F_{TF}-Axis) is defined as the cross product between the z_T-axis and the x_F-axis at any given joint position.

Patellofemoral Joint

This joint motion is described in a method similar to JCS[22], according to Bull et al.[23]. This method requires a definition of a patella coordinate system (x_P,y_P,z_P) , femoral coordinate system (same as that described above for the tibiofemoral joint), and a Floating axis (F_{PF}). The following anatomical landmarks will be located on the mesh of the patella:

- medial patella ridge
- lateral patella ridge
- midpoint of patella in coronal view

Patella Coordinate System:

Patella Origin: Mid-point of medial and lateral ridges of patella.

Medial-Lateral Axis (x_P-axis): The line connecting the medial and lateral ridges of the patella[46].

Superior-Inferior Axis (z_P-axis): Perpendicular to x_P and through the midpoint of the patella in coronal view[46].

Anterior-Posterior Axis (y_P-axis): The cross product between x_P-axis and z_P-axis.

Patellofemoral Floating axis:

The patellofemoral floating axis (F_{PF}-axis) is defined as the cross product between the z_P-axis and the x_F-axis.

Creating Kinematic Joints in FEBio

Recent versions of FEBio does not allow independent prescription of rotational degrees of freedom for a rigid body[12]. For this reason and to facilitate loading and boundary conditions relevant to anatomy of the knee, kinematic joint chains will be defined for tibiofemoral and patellofemoral joints.

Tibiofemoral Joint

Three rigid cylindrical joints[12] will be added to the constraints section to define tibiofemoral motion. For this reason two "imaginary" rigid bodies will be defined:

- Imaginary_Body_1: located at the origin of the tibial (x_T,y_T,z_T) coordinate system.
- Imaginary_Body_2: located at the origin of the femoral (x_F,y_F,z_F) coordinate system.

The cylindrical joints will be defined as follows:

Motion	**joint_origin**	**joint_axis**	**body_a**	**body_b**
Flexion-extension	Origin of (x_F,y_F,z_F)	x_F-axis	Femur	Imaginary_Body_1
External-internal rotation	Origin of (x_T,y_T,z_T)	z_T-axis	Imaginary_Body_2	Tibia
Abduction-adduction	Intersection of F_{TF}- and x_F-axes	F_{TF}-axis	Imaginary_Body_1	Imaginary_Body_2

All other parameters in the rigid cylindrical joints will be set to FEBio defaults[12]. Each cylindrical joint will also have a translational component to describe tibiofemoral joint translations[22].

Patellofemoral Joint

Three rigid cylindrical joints[12] will be added to the to constraints section to define patellofemoral motion. For this reason, two more "imaginary" rigid bodies will be defined:

- Imaginary_Body_3: located at the origin of the patella (x_P,y_P,z_P) coordinate system.
- Imaginary_Body_4: located at the origin of the femoral (x_F,y_F,z_F) coordinate system.

The cylindrical joints will be defined as follows:

Motion	**joint_origin**	**joint_axis**	**body_a**	**body_b**
Patellar flexion and shift	Origin of (x_F,y_F,z_F)	x_F-axis	Femur	Imaginary_Body_3
Patellar tilt	Origin of (x_P,y_P,z_P)	z_P-axis	Imaginary_Body_4	Patella
Patellar rotation	Intersection of F_{PF}- and x_F-axes	F_{PF}-axis	Imaginary_Body_3	Imaginary_Body_4

All other parameters in the rigid cylindrical joints will be set to FEBio defaults[12]. Each cylindrical joint will also have a translational component to describe patellofemoral joint translations[23].

Customization for Loading and Boundary Conditions

Target Outcome

Updated model for finite element analysis in FEBio[25] format (.feb, XML[36] based text file)[12] with modified loading and boundary conditions relevant to simulation of passive knee flexion; generated from template model file (.feb).

Burden

Software requirements:

Python. Python is a high-level multi-platform programming language (free and open source GPL compatible Python Software Foundation license, see https://www.python.org)[31]. Any contemporary version available for the computing platform can be used; 3.7.0 is the latest version at the time of the preparation of this document.

SciPy. SciPy is a Python based open source software platform for mathematics, science and engineering (free and open source BSD-new license, see https://www.scipy.org)[32]. Any contemporary version available for the computing platform can be used; 1.1.0 is the latest version at the time of the preparation of this document.

Hardware requirements:

Any contemporary computer; desktop, workstation, or laptop. Python is a multi-platform programming language for Windows, Mac OS X, and Linux.

Anticipated Man Hours and Expertise Level:

Less than an hour of full-time effort from a research engineer with bachelor's degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to Python scripting and model generation.

Computational Cost:

Minimal compared to required interactions with computer scripts.

Protocols

Input

Template (or customized) model in FEBio format (.feb) and loading and boundary conditions representative of passive knee flexion (described below) as a text based input file.

Scripting

A Python script will be developed to modify an existing template model with desired loading and boundary conditions to simulate passive knee flexion.

Simulation Steps

Loading and boundary conditions for the joint will be applied in two simulation steps (defined in template model as "Initialize" and "LoadingStep"); one for prescription of in situ strains, the other for representative of passive flexion[24]. In each of these steps, the kinematics or kinetics of the rigid bodies (femur, tibia, patella, fibula) and the cylindrical joints (for tibiofemoral and patellofemoral movements) can be prescribed.

In Situ Strain Application

In situ strains will be applied based on previously described specifications (see above) in the first simulation step ("Initialize"). Any loading and boundary conditions that are not specified below will be set to FEBio defaults[12].

- Prestrain: as described in in situ strain application section (see above)
- Femur: All degrees of freedom (3 translations, 3 rotations) free.

- Tibia: All degrees of freedom (3 translations, 3 rotations) fixed.
- Fibula: All degrees of freedom (3 translations, 3 rotations) fixed.
- Patella: All degrees of freedom (3 translations, 3 rotations) free.
- Tibiofemoral cylindrical joints: Flexion fixed (at 0º); remaining degrees of freedom (3 translations, 2 rotations) free.
- Patellofemoral cylindrical joints: All degrees of freedom (3 translations, 3 rotations) free.
- Quadriceps tendon slider joint: All degrees of freedom (3 translations, 3 rotations) free.

Passive Flexion

Loading and boundary conditions of passive knee flexion will be applied in the second simulation step ("LoadingStep"). At the start of this step, in situ strains should have already been applied as prescribed. Our interpretation of passive knee flexion[24] is that the motion of the knee is guided by joint contact and connective tissue recruitment. Therefore no external loading will be applied; the movement of the knee will be unconstrained other than prescription of the flexion angle up to 90º. Any loading and boundary conditions that are not specified below will be set to FEBio defaults[12].

- Femur: All degrees of freedom (3 translations, 3 rotations) free.
- Tibia: All degrees of freedom (3 translations, 3 rotations) fixed.
- Fibula: All degrees of freedom (3 translations, 3 rotations) fixed.
- Patella: All degrees of freedom (3 translations, 3 rotations) free.
- Tibiofemoral cylindrical joints: Flexion prescribed (0º to 90º during simulation step); remaining degrees of freedom (3 translations, 2 rotations) free.
- Patellofemoral cylindrical joints: All degrees of freedom (3 translations, 3 rotations) free.
- Quadriceps tendon slider joint: All degrees of freedom (1 translation) free.

Customization for Simulation Outputs

Target Outcome

Updated model for finite element analysis in FEBio[25] format (.feb, XML[36] based text file)[12] with modified output requests to quantify joint kinematics-kinetics and tissue mechanics relevant to simulation of passive knee flexion; generated from template model file (.feb).

Burden

<u>Software requirements:</u>

Python. Python is a high-level multi-platform programming language (free and open source GPL compatible Python Software Foundation license, see https://www.python.org)[31]. Any contemporary version available for the computing platform can be used; 3.7.0 is the latest version at the time of the preparation of this document.

SciPy. SciPy is a Python based open source software platform for mathematics, science and engineering (free and open source BSD-new license, see https://www.scipy.org)[32]. Any contemporary version available for the computing platform can be used; 1.1.0 is the latest version at the time of the preparation of this document.

Hardware requirements:

Any contemporary computer; desktop, workstation, or laptop. Python is a multi-platform programming language for Windows, Mac OS X, and Linux.

Anticipated Man Hours and Expertise Level:

Less than an hour of full-time effort from a research engineer with bachelor's degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to Python scripting and model generation.

Computational Cost:

Minimal compared to required interactions with computer scripts.

Protocols

Input

Template (or customized) model in FEBio format (.feb) and output requests relevant to interpretation of passive knee flexion simulations (described below) as a text based input file.

Scripting

A Python script will be developed to modify an existing template model with desired output requests relevant to interpretation of knee mechanics during passive flexion.

Output Requests

By default, FEBio provides two output files[12]: .xplt, plotfile, a binary file that includes all default and requested field variables compatible with PostView[30]; .log, logfile, a text file that reports convergence history of the simulation along with convergence metrics including any other variables requested for output. In accordance with the overall customization of the model and to provide output metrics relevant to simulation case, output of the following metrics will be requested:

- Prestrain stretch – to evaluate applied in situ strains at reference model configuration; compatible with FEBio PreStrain Plugin[28]; to be stored in plotfile.
- Fiber stretch – fiber stretch during simulation in tissues with compatible constitutive models (ligaments, tendon, menisci).
- Rigid body data – translation and rotation of rigid bodies (3 translation variables: positions of center of mass; 4 rotation variables: components of rotation quaternion); reaction loads on the rigid bodies (3 forces; 3 moments); for femur, tibia, fibula, and patella; to be stored in plotfile and logfile.
- Rigid joints data – translation and rotation of kinematic joints (3 translations; 3 rotations); reaction loads of kinematic joints (3 forces; 3 moments); for all cylindrical joints of tibiofemoral and patellofemoral joints and for slider joint of the quadriceps tendon; to be stored in plotfile and logfile.

See FEBio User's Manual[12] for more details. Any other outputs will be FEBio and template model file defaults[12].

Simulation

Target Outcome

Solution of fully customized model through finite element analysis using FEBio[25]; generating simulation results as binary and text output files (.xplt and .log, respectively)[12].

Burden

Software requirements:

FEBio. FEBio is a nonlinear implicit finite element analysis framework designed specifically for analysis in biomechanics and biophysics (custom open source license; free for academic research use, licensing for commercial use is available, see http://www.febio.org)[25]. The latest version of FEBio (version 2.8 at the time of preparation of this document) will be used.

FEBio PreStrain Plugin. PreStrain Plugin provides a general framework for representing prestrain in a finite element model using a prestrain gradient method. A version compatible with the latest version FEBio will be used (at the time of preparation of this document, version 1.0 supporting FEBio 2.5.1 has been available)[28].

Hardware requirements:

Any contemporary computer; desktop, workstation, or laptop. FEBio is supported on multiple platforms including Windows, Mac OS X, and Linux.

Anticipated Man Hours and Expertise Level:

Less than an hour of full-time effort from a research engineer with bachelor's degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to finite element analysis.

Computational Cost:

A few hours of simulation time (wall clock).

Protocols

Input

Fully customized model in FEBio format (.feb).

Simulation Process

Invoke FEBio with the model file as input and hope that, by some miracle, the simulation converges. If simulation does not convergence, relaxation of convergence tolerances and utilization of alternative solution algorithms, contact formulations, etc.[12] will likely be required.

Post-Processing

Target Outcome

Extraction and summary of knee kinematics and kinetics during passive flexion; processed using raw simulation results of fully customized model with FEBio (.log file)[12], supported by graphs as binary image files.

Burden

Software requirements:

Python. Python is a high-level multi-platform programming language (free and open source GPL compatible Python Software Foundation license, see https://www.python.org)[31]. Any contemporary version available for the computing platform can be used; 3.7.0 is the latest version at the time of the preparation of this document.

SciPy. SciPy is a Python based open source software platform for mathematics, science and engineering (free and open source BSD-new license, see https://www.scipy.org)[32]. Any contemporary version available for the computing platform can be used; 1.1.0 is the latest version at the time of the preparation of this document.

PostView. PostView is a post-processor to visualize and analyze results from FEBio, finite element analysis package for biomechanics (custom open source license; free for academic research use, licensing for commercial use is available, see https://febio.org/postview/)[30]. The latest version of PostView (version 2.3.0 at the time of preparation of this document) will be used.

Hardware requirements:

Any contemporary computer; desktop, workstation, or laptop. Python is a multi-platform programming language for Windows, Mac OS X, and Linux. PostView is supported on multiple platforms including Windows, Mac OS X, and Linux (note that latest Linux version at the moment is 1.10).

Anticipated Man Hours and Expertise Level:

Less than an hour of full-time effort from a research engineer with bachelor's degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to finite element analysis.

Computational Cost:

Minimal compared to required interactions with computer scripts.

Protocols

Input

Solution of fully customized model through finite element analysis using FEBio[25]; simulation results as binary and text output files (.xplt and .log, respectively)[12].

Scripting

A Python script will be developed to read the log file and extract, store (as .xml) and plot knee kinematics and kinetics during passive flexion (as .png):

- Kinematics of tibiofemoral cylindrical joints: 3 rotations, 3 translations total
- Kinematics of patellofemoral cylindrical joints: 3 rotations, 3 translations total
- Translation of tibia origin relative to femur origin in femoral coordinate system
- Translation of patella origin relative to femur origin in femoral coordinate system
- Constraint moment to maintain the knee joint at prescribed flexion

Visualization

PostView will be used to take snapshots of the model at different flexion angles, as obtained through simulation of passive flexion. PostView can also be used to inspect tissue stress-strain distributions, export data, images, and animations.

Dissemination

Target Outcome

Modeling and simulation outputs delivered to the public as a download package.

Burden

Infrastructure:

SimTK. SimTK is a free project-hosting platform for the biomedical computation community (see https://simtk.org/)[33]. Project sites at SimTK provide source code repositories, wikis to support development; and news, forums, downloads and documents sections to engage with user communities.

Anticipated Man Hours and Expertise Level:

Less than an hour of full-time effort from a research engineer with bachelor’s degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to public dissemination.

Protocols

All modeling and simulation outputs of the *Model Development* phase will be collated as a package and uploaded to the project site of *Reproducibility in simulation-based prediction of natural knee mechanics* located at SimTK[33] (https://simtk.org/projects/kneehub/)[3]. This download package will be accessible by the public licensed under Creative Commons Attribution 4.0 International License[47].

Protocol Deviations

Target Outcome

Protocol deviations to model development specifications documented and delivered to the public.

Burden

Infrastructure:

SimTK. SimTK is a free project-hosting platform for the biomedical computation community (see https://simtk.org/)[33]. Project sites at SimTK provide source code repositories, wikis to support development; and news, forums, downloads and documents sections to engage with user communities.

Anticipated Man Hours and Expertise Level:

For each protocol deviation, on the order of minutes of full-time effort from a research engineer with bachelor’s degree, mechanical/biomedical background, <2 years of research experience, and some familiarity to finite element analysis.

Protocols

It is anticipated that some deviations to modeling and simulation workflow, described in here as part of *Model Development* phase, will happen. There is also the possibility that some information on model development specifications may be missing. All these will be documented on an ongoing basis during the execution of the planned workflow[2]. Final document will be submitted to the project site of *Reproducibility in simulation-based prediction of natural knee mechanics* located at SimTK[33] (https://simtk.org/projects/kneehub/)[3] as a publicly accessible document under Creative Commons Attribution 4.0 International License[47].

Overall Burden

Overall burden of the modeling and simulation workflow described in here is determined by the requirements for data, labor, software and hardware, and other infrastructure. Use of existing, publicly available data, in this case Open Knee(s) data set, negates the burden for data acquisition. Software and hardware costs are associated with pre-/post-processing and simulation. It is anticipated that model development, simulation, and analysis of simulation results can be performed in any contemporary computer, minimizing hardware costs. All software packages used in the modeling and simulation workflow are freely available: 3D Slicer[6], MeshLab[9], SALOME[11] – for model development; Python[31] and SciPy[32] – to utilize Python scripts (existing and yet to be developed) for pre- and post-processing; FEBio[25] and FEBio PreStrain Plugin[28] – for finite element analysis; and PostView[30] – for visualization of simulation results. The activity will leverage SimTK for public dissemination. SimTK is a freely available project hosting site for biomedical computing[33]. Labor effort will be at a minimum of 4 weeks of full time effort from a research engineer with bachelor’s degree, mechanical/biomedical background, less than 2 years of research experience, and some familiarity to finite element analysis. This effort level includes all modeling activities, record keeping, and dissemination. It should be noted that this estimate relies on the assumption that modeling and simulation processes complete as planned, without any significant deviations and iterations. The overall burden of the model development specifications should be evaluated in concert with their desired final outcome – a comprehensive and extensible knee joint model incorporating anatomical and mechanical detail of its major structures.

References

1. SimTK: Open Knee(s): Virtual Biomechanical Representations of the Knee Joint: Project Home. Available at: https://simtk.org/projects/openknee. (Accessed: 13th August 2018)
2. ModelDevelopment - kneehub. Available at: https://simtk.org/plugins/moinmoin/kneehub/ModelDevelopment. (Accessed: 13th August 2018)
3. SimTK: Reproducibility in simulation-based prediction of natural knee mechanics: Project Home. Available at: https://simtk.org/projects/kneehub. (Accessed: 13th August 2018)
4. SimTK: Reproducibility in simulation-based prediction of natural knee mechanics: Downloads. Available at: https://simtk.org/frs/?group_id=1061. (Accessed: 13th August 2018)
5. Specifications/ExperimentationAnatomicalImaging - openknee. Available at: https://simtk.org/plugins/moinmoin/openknee/Specifications/ExperimentationAnatomicalImaging. (Accessed: 13th August 2018)
6. 3D Slicer. Available at: https://www.slicer.org/. (Accessed: 13th August 2018)
7. Kikinis, R., Pieper, S. D. & Vosburgh, K. G. 3D Slicer: A Platform for Subject-Specific Image Analysis, Visualization, and Clinical Support. in *Intraoperative Imaging and Image-Guided Therapy* 277–289 (Springer, New York, NY, 2014). doi:10.1007/978-1-4614-7657-3_19
8. Chua, C. K., Leong, K. F. & Lim, C. S. Chapter 6. Rapid Prototyping Data Formats. in *Rapid Prototyping: Principles and Applications* 301–356 (World Scientific Pub Co Inc, 2010).
9. MeshLab. Available at: http://www.meshlab.net/. (Accessed: 13th August 2018)
10. Cignoni, P. *et al.* MeshLab: an Open-Source Mesh Processing Tool. in *Sixth Eurographics Italian Chapter Conference* 129–136 (The Eurographics Association, 2008). doi:10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136
11. Welcome to the www.salome-platform.org — SALOME Platform. Available at: https://www.salome-platform.org/. (Accessed: 13th August 2018)
12. FEBio User's Manual Version 2.8. Available at: https://help.febio.org/FEBio/FEBio_um_2_8/index.html. (Accessed: 13th August 2018)
13. Donahue, T. L. H., Hull, M. L., Rashid, M. M. & Jacobs, C. R. A finite element model of the human knee joint for the study of tibio-femoral contact. *J Biomech Eng* **124,** 273–280 (2002).
14. Peña, E., Calvo, B., Martínez, M. A. & Doblaré, M. A three-dimensional finite element analysis of the combined behavior of ligaments and menisci in the healthy human knee joint. *J Biomech* **39,** 1686–1701 (2006).
15. Dhaher, Y. Y., Kwon, T.-H. & Barry, M. The effect of connective tissue material uncertainties on knee joint mechanics under isolated loading conditions. *J Biomech* **43,** 3118–3125 (2010).

16. Shriram, D., Praveen Kumar, G., Cui, F., Lee, Y. H. D. & Subburaj, K. Evaluating the effects of material properties of artificial meniscal implant in the human knee joint using finite element analysis. *Sci Rep* **7,** 6011 (2017).

17. Stephen, J. M., Lumpaopong, P., Deehan, D. J., Kader, D. & Amis, A. A. The medial patellofemoral ligament: location of femoral attachment and length change patterns resulting from anatomic and nonanatomic attachments. *Am J Sports Med* **40,** 1871–1879 (2012).

18. Aframian, A., Smith, T. O., Tennent, T. D., Cobb, J. P. & Hing, C. B. Origin and insertion of the medial patellofemoral ligament: a systematic review of anatomy. *Knee Surg Sports Traumatol Arthrosc* **25,** 3755–3772 (2017).

19. Shah, K. N., DeFroda, S. F., Ware, J. K., Koruprolu, S. C. & Owens, B. D. Lateral Patellofemoral Ligament: An Anatomic Study. *Orthop J Sports Med* **5,** 2325967117741439 (2017).

20. Elias, J. J. & Cosgarea, A. J. Technical errors during medial patellofemoral ligament reconstruction could overload medial patellofemoral cartilage: a computational analysis. *Am J Sports Med* **34,** 1478–1485 (2006).

21. Merican, A. M., Sanghavi, S., Iranpour, F. & Amis, A. A. The structural properties of the lateral retinaculum and capsular complex of the knee. *J Biomech* **42,** 2323–2329 (2009).

22. Grood, E. S. & Suntay, W. J. A joint coordinate system for the clinical description of three-dimensional motions: application to the knee. *J Biomech Eng* **105,** 136–144 (1983).

23. Bull, A. M. J., Katchburian, M. V., Shih, Y.-F. & Amis, A. A. Standardisation of the description of patellofemoral motion and comparison between different techniques. *Knee Surg Sports Traumatol Arthrosc* **10,** 184–193 (2002).

24. Wilson, D. R., Feikes, J. D., Zavatsky, A. B. & O'Connor, J. J. The components of passive knee movement are coupled to flexion angle. *J Biomech* **33,** 465–473 (2000).

25. FEBio Software Suite. Available at: https://febio.org/. (Accessed: 13th August 2018)

26. Maas, S. A., Ellis, B. J., Ateshian, G. A. & Weiss, J. A. FEBio: finite elements for biomechanics. *J Biomech Eng* **134,** 011005 (2012).

27. Maas, S. A., Ateshian, G. A. & Weiss, J. A. FEBio: History and Advances. *Annu Rev Biomed Eng* **19,** 279–299 (2017).

28. Plugins | FEBio Software Suite. Available at: https://febio.org/plugins/. (Accessed: 13th August 2018)

29. Maas, S. A., Erdemir, A., Halloran, J. P. & Weiss, J. A. A general framework for application of prestrain to computational models of biological materials. *J Mech Behav Biomed Mater* **61,** 499–510 (2016).

30. PostView | FEBio Software Suite. Available at: https://febio.org/postview/. (Accessed: 13th August 2018)

31. Welcome to Python.org. *Python.org* Available at: https://www.python.org/. (Accessed: 13th August 2018)

32. SciPy.org — SciPy.org. Available at: https://www.scipy.org/. (Accessed: 13th August 2018)

33. SimTK: Welcome. Available at: https://simtk.org/. (Accessed: 13th August 2018)

34. NIfTI: — Neuroimaging Informatics Technology Initiative. Available at: https://nifti.nimh.nih.gov/. (Accessed: 13th August 2018)
35. Med — SALOME Platform. Available at: http://www.salome-platform.org/user-section/about/med. (Accessed: 13th August 2018)
36. Extensible Markup Language (XML). Available at: https://www.w3.org/XML/. (Accessed: 13th August 2018)
37. Roelofs, G. *PNG: The Definitive Guide*. (O'Reilly Media, 1999).
38. Documentation - SlicerWiki. Available at: https://www.slicer.org/wiki/Documentation. (Accessed: 13th August 2018)
39. multis - Revision 612: /utl/ModelAssembly. Available at: https://simtk.org/svn/multis/utl/ModelAssembly/. (Accessed: 13th August 2018)
40. Schöberl, J. NETGEN An advancing front 2D/3D-mesh generator based on abstract rules. *Comput Visual Sci* **1,** 41–52 (1997).
41. SALOME NETGENPLUGIN User's Guide: Introduction to NETGENPLUGIN. Available at: http://docs.salome-platform.org/latest/gui/NETGENPLUGIN/index.html. (Accessed: 13th August 2018)
42. Henak, C. R., Anderson, A. E. & Weiss, J. A. Subject-specific analysis of joint contact mechanics: application to the study of osteoarthritis and surgical planning. *J Biomech Eng* **135,** 021003 (2013).
43. Weiss, J. A., Gardiner, J. C., Ellis, B. J., Lujan, T. J. & Phatak, N. S. Three-dimensional finite element modeling of ligaments: technical aspects. *Med Eng Phys* **27,** 845–861 (2005).
44. Fithian, D. C., Kelly, M. A. & Mow, V. C. Material properties and structure-function relationships in the menisci. *Clin. Orthop. Relat. Res.* **252,** 19–31 (1990).
45. Gardiner, J. C. & Weiss, J. A. Subject-specific finite element analysis of the human medial collateral ligament during valgus knee loading. *J. Orthop. Res.* **21,** 1098–1106 (2003).
46. Suzuki, T., Hosseini, A., Li, J.-S., Gill, T. J. & Li, G. In vivo patellar tracking and patellofemoral cartilage contacts during dynamic stair ascending. *J Biomech* **45,** 2432–2437 (2012).
47. Creative Commons — Attribution 4.0 International — CC BY 4.0. Available at: https://creativecommons.org/licenses/by/4.0/. (Accessed: 13th August 2018)