FreeBody 1.0

(Incorporating the Imperial College Lower Limb Model and the St Mary’s University Knee Model)

**Software and Model Description**

**And**

**User’s Guide**

Dr Dan Cleather

St Mary’s University

Contents

[Contents 2](#_Toc380745838)

[§ 1: Introduction 3](#_Toc380745839)

[§ 2: “Black Box” User’s Guide 4](#_Toc380745840)

[§3: FreeBody Overview 10](#_Toc380745841)

[§ 4: Detailed Software Description 11](#_Toc380745842)

[Software Main File (Model v2.cpp) 11](#_Toc380745843)

[Module 1: Data Input (m01\_00\_data\_input.cpp) 12](#_Toc380745844)

[Module 2: Model Construction (m02\_00\_create\_segments.cpp) 18](#_Toc380745845)

[Module 3: Calculate Model Kinematics (m03\_00\_calculate\_kinematics.cpp) 23](#_Toc380745846)

[Module 4: Calculate Segment Kinetics (m04\_00\_inverse\_dynamics.cpp) 27](#_Toc380745847)

[Module 5: Calculate Muscle Geometry (m05\_00\_muscle\_model.cpp) 29](#_Toc380745848)

[§ 5: Detailed Model Description 32](#_Toc380745849)

[References 36](#_Toc380745850)

[Appendices 37](#_Toc380745851)

[Appendix A: Class Descriptions 38](#_Toc380745852)

[Appendix B: Library Routines 39](#_Toc380745853)

[Appendix C: List of Variables/Arrays/Objects 40](#_Toc380745854)

§ 1: Introduction

This document describes the FreeBody modelling software. FreeBody consists of a framework for musculoskeletal modelling that incorporates a model of the lower limb (the Imperial College Lower Limb Model – ICLLM) and knee (St Mary’s University Knee Model – SMUKM). The purpose of the software is to allow the musculoskeletal geometry of a subject to be described, based upon motion capture data. This in turn then allows the calculation of muscle and joint forces, based upon an inverse optimization calculation method. 1–3

The philosophy behind the public release of FreeBody is to provide a musculoskeletal modelling tool with the greatest amount of utility for the largest range of users. To this end, the software is provided both as a pre-compiled application that is ready for immediate use and as the original source code. This should then allow a range of implementations, from the use of the model as a “black box” for calculating muscle and joint forces, through to using the code as a library of functions or a framework for the development of bespoke models.

The purpose of this manual is thus to provide the information that any user might need in order to apply FreeBody in the fashion desired. The manual is thus organised to make it as easy as possible for a user to find the required information.

Chapter 2 describes the version of FreeBody that is packaged as a MATLAB® application. This application allows the software to be used as a black box application for the calculation of muscle and joint forces in the lower limb using the ICLLM and SMUKM. The chapter describes the data that must be collected to facilitate this, how the data should be organised for use in the model (data pre-processing) and how to install and run the model.

Chapter 3 provides an overview of the function of the model.

Chapter 4 presents a detailed description of the source code.

Chapter 5

Finally, the Appendices provide some further detailed description of the source code which complements Chapter 4.

§ 2: The FreeBody MATLAB® App

*Installing FreeBody*

This article describes an implementation of FreeBody that is packaged as a MATLAB® app. The software is provided as a .ZIP file that can be downloaded from [www.drjump.co.uk/freebody](http://www.drjump.co.uk/freebody). Installation is straightforward – simply extract the contents of the .ZIP file to the desired directory, and then run the file “FreeBody Beta.mlappinstall”. This will launch MATLAB® and ask the user if they wish to install the application. Once the user has confirmed this, the app will be fully installed.

*Running the FreeBody Demo*

The software is provided including a pre-loaded example (that of an 86.4 kg male subject performing a vertical jump). This data can thus be used to try out the software immediately after installation. Clicking on the “FreeBody Beta” icon on the “Apps” pane of MATLAB® launches a graphical user interface (GUI) like the one depicted in Figure x. The first step is to enter the path name to the directory in which FreeBody was installed in the box at the bottom of the GUI (and to press return). The analysis pipeline of FreeBody is then run by clicking each button in turn and waiting for the given operation to complete.

Figure x. The FreeBody GUI.

freebody gui.tif

The “Musculoskeletal Geometry” button launches a Windows application that is external to MATLAB®. This application takes the raw data and performs all of the calculations of the model prior to the optimization. In particular, the application specifies the musculoskeletal geometry of the model for each frame. Once this process has been run, the user can plot the musculoskeletal geometry of the model. Firstly, the musculoskeletal geometry must be loaded into MATLAB® by clicking the “Load Geometry Plot Data” button – when this process is complete it will be indicated in the MATLAB® command window. Then the dynamic geometry of the model can be visualised by clicking the “Plot Geometry” button. Note that it is only necessary to load the geometry data once, then the plot can be run multiple times.

The next step is to run the optimization to calculate the muscle and joint forces. This is achieved by clicking the fourth button – “Run Optimization”. The optimization will take some time to complete and will be greatly dependent on the speed of the platform being used. The user can monitor the progress of the optimization in the MATLAB® command window.

Finally, once the optimization is complete, the user has two options to visualise the results. The “Analyse Results” button will produce a number of plots describing muscular and joint forces (Figure x). The “Plot Activations” button will launch a dynamic plot of the musculoskeletal geometry where the colour of the muscle elements will change reflecting the level of activity (more red indicates more activation; Figure x).

Figure x. Model analysis within FreeBody.

analysis.tif

Figure x. Visualising muscular activations with FreeBody.

jump.tif

For a given data set, it is only necessary to run the “Musculoskeletal Geometry” and “Run Optimization” routines once. These are then saved within the file structure of FreeBody. These results can then be visualised even if FreeBody has been closed and restarted. However, it will be necessary to load the geometry plot data before either of the geometry plots can be utilised.

*Using FreeBody with your own data*

The lower limb model within FreeBody is based upon the input of motion capture data (positions of anatomical landmarks) and the ground reaction force. Table x lists the standard marker positions necessary to use this model, and Figure x illustrates the placement of these markers on a subject.

Table x. Marker positions used for data capture.

|  |  |
| --- | --- |
| Marker | Location |
| FCC | Calcaneus |
| FMT | Tuberosity of the fifth metatarsal |
| FM2\* | Head of the second metatarsal |
| TF | Additional marker placed on the foot |
| FAM\* | Apex of the lateral malleolus |
| TAM\* | Apex of the medial malleolus |
| C1, C2, C3 | Additional markers placed on the calf segment |
| FLE\* | Lateral femoral epicondyle |
| FME\* | Medial femoral epicondyle |
| T1, T2, T3 | Additional markers placed on the thigh segment |
| RASIS\* | Right anterior superior iliac spine |
| LASIS\* | Left anterior superior iliac spine |
| RPSIS\* | Right posterior superior iliac spine |
| LPSIS\* | Left posterior superior iliac spine |

Notes: \* indicates those markers that must be placed upon the indicated anatomical landmark. The remaining markers could be placed in other positions (as long as they are still on the same segment).

Figure x. Position of markers on a subject.

**markers.tif**

This raw data then needs to be pre-processed into a standard format for input into FreeBody. Details as to the input files that need to be created from the raw data are provided in more detail in the Web Supplementary Material and at [www.drjump.co.uk/freebody](http://www.drjump.co.uk/freebody).

Finally, the input files need to be copied into the Inputs folder of FreeBody.

*Advanced options*

Model settings

Outputs

**Raw Data Pre Processing and Model Settings**

A number of input files are required in order to run FreeBody, which specify both the raw data, and the characteristics of the model.

**Raw Data**

If using FreeBody as a black box, then the contents of most of the model settings files will be

|  |  |
| --- | --- |
| **Input File** | **File Content** |

|  |  |
| --- | --- |
| basics.txt | 1 6  5 163 14 200 0.014 400 |

This file specifies the most fundamental characteristics of the model. The second row gives this information. The first three numbers on this row are the number of segments, muscles and ligaments in the model, and should not be changed if using the software as a black box. The fourth and fifth numbers will need to be changed however. The fourth number is the frequency of the raw data (in Hz), the fifth number is the size of the markers, and the sixth number is the number of frames to be analysed.

|  |  |
| --- | --- |
| force\_data.txt | 498 6  227.2 214.4 0 0.2 43.1 432.7  227.0 214.4 0 0.3 43.1 432.6  226.9 214.5 0 0.3 43.1 432.4  226.6 ......................... |

This gives the raw data pertaining to the external forces acting on the subject. If using FreeBody as a black box, the default setting is that there is one external force (the ground reaction force) acting on the foot. The first two numbers in the file describe the size of the following matrix. In the matrix, each row is one frame. Within a row, the first 3 numbers give the point at which the force can be considered to act (the centre of pressure from the force plate), and the second 3 numbers the force itself. These values are given in the laboratory coordinate frame.

|  |  |
| --- | --- |
| position\_data.txt | 490 54  107.0 237.8 39.1 212.0 178.3 .....  107.0 237.8 39.1 212.0 178.3 .....  107.1 ..... |

This gives the raw data pertaining to the positions of each marker. The first two numbers describe the size of the following matrix. In the matrix, each row is one frame. Each group of three numbers on a row, describes the position of one marker (given in the laboratory coordinate frame). The markers are arranged within each row as follows: first all of the markers on segment 0, next all of the markers on segment 1, etc. The order of the markers (which are described in Table 2.1) within the file should be as follows: FCC, FMT, FM2, FT, FAM, TAM, C1, C2, C3, FLE, FME, T1, T2, T3, RASIS, LASIS, RPSIS, LPSIS.

|  |  |
| --- | --- |
| anthropometry.txt | 5 6  1.2 0.442 0.558 0.006 0.0014 0.005  3.7 0.446 0.554 0.0525 0.0086 0.05  12.2 0.41 0.59 0.2703 0.0554 0.270  9.7 0.612 0.388 0.0887 0.0808 0.071  0 0 0 0 0 0 |

This file gives the anthropometry of each segment. The first two numbers describe the size of the matrix that follows. Each following row the represents a segment, starting with segment 0. Within a row, the first number gives the mass of the segment. The second number gives the location of the COM of the segment (assuming it lies on the longitudinal axis of the segment) expressed as the percentage of total length from proximal to distal. The third number gives the percentage length from COM to the distal end of the segment. The final three numbers describe the moment of inertia of the segment (given in the segment fixed coordinate system). The ICLLM uses the model of de Leva 4 to calculate the subject specific anthropometry, and an Excel file (anthropometry.xls) is provided to ease calculation of these parameters.

|  |  |
| --- | --- |
| calib\_data.txt | 1 54  107.0 237.8 39.1 212.0 178.3 ..... |

This gives the raw data pertaining to the positions of each marker when the subject is standing in the anatomical position (calibration position). The format of the file is the same as the position\_data.txt file.

|  |  |
| --- | --- |
| coord\_map.txt | 3  1 3 -2 |

This file specifies the mapping from the coordinate system of the laboratory where the data was collected to the model coordinate system. It is likely that this will need to be changed based on the particular configuration of the user’s laboratory. A description of this mapping is provided in [Chapter 4](#coord_map).

§3: FreeBody Overview

**Overview**

**Model Basics**

CS

**Classes and Objects**

FreeBody makes an extensive use of classes in order to store and manipulate data. Of most importance are the Segment and Muscle classes.

**Segment Class**

The Segment class is used to describe the characteristics of each body segment and to perform calculations pertaining to segments.

**Class Structure**

**Specific Objects**

|  |  |
| --- | --- |
| \*\*segment\_data[frames][segments] |  |
| \*segment\_chain |  |

§ 4: Detailed Software Description

The following pages present a detailed description of the software, arranged to mirror the order of the software.

## Software Main File (Model v2.cpp)

* The entry point for FreeBody is the file “Model v2.cpp”.
* The number of segments, muscles, and ligaments, the frequency of the data and the number of frames are input here:

|  |  |
| --- | --- |
| **Input File** | **File Content** |

|  |  |
| --- | --- |
| basics.txt | 1 6  5 163 14 200 0.014 400 |

This file specifies the most fundamental characteristics of the model. The first two numbers specify the size of the matrix that follows. The second row gives the basic information. The first three numbers on this row are the number of segments, muscles and ligaments in the model. The fourth number is the frequency of the raw data (in Hz), the fifth number is the size of the markers, and the sixth number is the number of frames to be analysed.

* This file first defines the fundamental variables and data structures used in the model. In particular, the objects Segment \*\*segment\_data[frames], Muscle \*\*muscle\_data[frames] and Muscle \*muscle\_model[muscles] are initialized here.
* Each module involved in the analysis pipeline is then called sequentially from this main file.

## Module 1: Data Input (m01\_00\_data\_input.cpp)

The purpose of this module is to input the raw data into the model and then to process it into the format that will be used in the rest of the model. The raw data includes a specification as to the structure of the model itself.

The model is provided with a routine to input and process raw data in a given format (described in this chapter). However, it is perfectly feasible to input data in other formats provided that new code is written to process this data into the format required by the model (also described in this chapter). If this code is written, it can be slotted into the model in this module.

**Inputs**

* Raw data (marker positions, external forces, calibration data)
* Mapping of laboratory to model
* Model structure

**Processes**

Data is input from text files and saved to relevant objects for later use.

**Outputs**

* All raw data saved to Segment \*\*segment\_data[]
* All calibration data saved to Structure \*calibrate\_pos[]

**Module Overview**

**Selected Detailed Software Description**

**m01\_01\_dans\_data\_input.cpp**

This routine takes raw data that is formatted as described below, and processes it for use by FreeBody. Raw data is provided in a number of text files that specify both the details of the model, and the kinematics and kinetics of the movement being analysed. The data for each segment and each frame is saved to Segment \*\*segment\_data[] (the structure of this class is described in [Appendix A](#_Appendix_A:_Class)).

|  |  |
| --- | --- |
| **Input File** | **Example File Content** |

*Model Settings*

|  |  |
| --- | --- |
| coord\_map.txt | 3  1 3 -2 |

This file specifies the mapping from the coordinate system of the laboratory where the data was collected to the model coordinate system. The model coordinate system is with the x axis running from posterior to anterior, the y axis from inferior to superior, and the z axis running medio-laterally (with the positive direction being towards the person’s right hand).

The first number in the file simply specifies the number of numbers to follow. The remaining 3 numbers specify the mapping based upon the following system. The first number given describes the laboratory axis that maps to the model x axis, the second number the laboratory axis that maps to the model y axis, etc. The numbers themselves represent the laboratory axes, with 1 being the x axis, 2 the y axis and 3 the z axis. Thus in the example the mapping described is as follows:

|  |  |  |
| --- | --- | --- |
| Laboratory  x'  z'  -y' | | Model  x  y  z |
|  | |  |
| markers.txt | 5  4 5 5 4 0 | |

This file specifies the number of markers per segment. The first number gives the remaining number of numbers in the file, then each following number gives the number of markers per segment starting from segment 0 (where the segment numbers are as defined in the model\_tree\_structure.txt).

|  |  |
| --- | --- |
| model\_tree\_structure.txt | 5 5  0 1 2 3 4  -1 0 1 2 2  1 1 2 1 1  -1 1 -1 1 -1 |

This file specifies the order in which segments link to one another in the model. The first two numbers represent the size of the matrix that follows. Within this matrix, each column represents a segment. The first row gives the number of each segment (numbered in order). The following row then gives the number of the segment that attaches distally to the segment in that column (-1 is the ground).

|  |  |
| --- | --- |
| external\_forces.txt | 5  1 0 0 0 0 |

This file describes how many external forces are acting on each segment. The first number gives the remaining number of numbers in the file. The remainder of the numbers indicate how many external forces act on each segment, starting from segment 0.

*Raw Data*

|  |  |
| --- | --- |
| force\_data.txt | 498 6  227.2 214.4 0 0.2 43.1 432.7  227.0 214.4 0 0.3 43.1 432.6  226.9 214.5 0 0.3 43.1 432.4  226.6 ......................... |

This gives the raw data pertaining to the external forces acting on the subject. The first two numbers describe the size of the following matrix. In the matrix, each row is one frame. Within a row, the first 3 numbers give the point at which the force can be considered to act, and the second 3 numbers the force itself. These values are given in the laboratory coordinate frame.

|  |  |
| --- | --- |
| position\_data.txt | 490 54  107.0 237.8 39.1 212.0 178.3 .....  107.0 237.8 39.1 212.0 178.3 .....  107.1 ..... |

This gives the raw data pertaining to the positions of each marker. The first two numbers describe the size of the following matrix. In the matrix, each row is one frame. Each group of three numbers on a row, describes the position of one marker (given in the laboratory coordinate frame). The markers are arranged within each row as follows: first all of the markers on segment 0, next all of the markers on segment 1, etc. The order of the markers must be the same as in the calibration file (calib\_data.txt).

**m01\_02\_dans\_calibration\_data.cpp**

This routine inputs the calibration data for the model. This is saved to Structure \*calibrate\_pos[].

*Calibration Data*

|  |  |
| --- | --- |
| anthropometry.txt | 5 6  1.2 0.442 0.558 0.006 0.0014 0.005  3.7 0.446 0.554 0.0525 0.0086 0.05  12.2 0.41 0.59 0.2703 0.0554 0.270  9.7 0.612 0.388 0.0887 0.0808 0.071  0 0 0 0 0 0 |

This file gives the anthropometry of each segment. The first two numbers describe the size of the matrix that follows. Each following row the represents a segment, starting with segment 0. Within a row, the first number gives the mass of the segment. The second number gives the location of the COM of the segment (assuming it lies on the longitudinal axis of the segment) expressed as the percentage of total length from proximal to distal. The third number gives the percentage length from COM to the distal end of the segment. The final three numbers describe the moment of inertia of the segment (given in the segment fixed coordinate system). The ICLLM uses the model of de Leva 4 to calculate the subject specific anthropometry, and an Excel file is provided to ease calculation of these parameters.

|  |  |
| --- | --- |
| calib\_data.txt | 1 54  107.0 237.8 39.1 212.0 178.3 ..... |

This gives the raw data pertaining to the positions of each marker when the subject is standing in the anatomical position (calibration position). The format of the file is the same as the position\_data.txt file. The position (within the matrix) of the anatomical landmarks to be used to create the LCSs is specified in the landmarks file (landmarks.txt). For the lower body model (ICLLM) provided as part of FreeBody, the order of the markers (which are described in Table 2.1) within the file should be as follows: FCC, FMT, FM2, TF, FAM, TAM, C1, C2, C3, FLE, FME, T1, T2, T3, RASIS, LASIS, RPSIS, LPSIS.

**m01\_01\_map\_coord.cpp**

This routine maps the input raw data from the laboratory coordinate system to the model coordinate system using the schema defined in the coord\_map.txt file (as described underneath the [description of the coord\_map.txt file](#coord_map)).

**m01\_03\_save\_raw\_positions.cpp**

This routine saves the input raw data and calibration data to the Segment \*\*segment\_data[]and Structure \*calibrate\_pos[] objects, respectively.

## Module 2: Model Construction (m02\_00\_create\_segments.cpp)

The purpose of this module is to create the system of rigid linked segments that describe the model. In particular, the calibration data is used to describe a local coordinate system (LCS) for each of the segments of the model. This is then used to find a transformation (rotation and translation) from the LCS of each segment to its position in each of the frames of the input data.

**Inputs**

* Positions of all markers saved in Segment \*\*segment\_data[]
* Positions of all markers with the subject standing in the anatomical position saved in Structure \*calibrate\_pos[]

**Processes**

Create rigid segments representing the model. Specify the position and orientation of each segment in each frame by finding the transformation from the LCS of the segment to its position in the laboratory fixed GCS.

In particular, firstly find the transformation of each segment from the calibration position to the position of the segment in the GCS for each frame (Segment::seg\_calculate\_transf\_cal\_to\_gcs.cpp). Then find the transformation of each segment from the LCS to the calibration position (m02\_02\_make\_lcs.cpp). Finally, combine these two to find the transformation from LCS to GCS (Segment::seg\_calculate\_transf\_lcs\_to\_gcs.cpp).

In addition, and as part of the above, this module specifies the position of the patellar segment, and the tibiofemoral joint reaction forces. That is, this module includes the detail of the SMUKM.

**Outputs**

* Positions and orientations of each segment saved in Segment \*\*segment\_data[]
  + segment\_data[][]->origin
  + segment\_data[][]->rot\_lcs\_to\_gcs

**Module Overview**

**Selected Detailed Software Description**

**m02\_04\_input\_landmarks.cpp**

This routine takes raw data that is formatted as described below, and processes it for use by FreeBody. Raw data is provided in a number of text files that specify both the details of the model, and the kinematics and kine

|  |  |
| --- | --- |
| **Input File** | **Example File Content** |

|  |  |
| --- | --- |
| landmarks.txt | 5 12  -1 -1 -1 -1 -1 -1 0 2 1 0 1 1  -1 -1 -1 -1 -1 -1 -1 -1 2 0 2 1  -1 -1 -1 -1 -1 -1 -1 -1 2 0 2 1  3 0 3 2 -1 -1 -1 -1 3 0 3 1  -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 |

This file describes the location of the landmarks for the Klein Horsman muscle model within the calibration raw data file. The first two figures specify the size of the matrix that follows. Following this, each row represents a segment starting from segment 0. Each row can be split into 3 sections containing four numbers. Each of these segments refers to one axis of the segment in the order x, y, z. Within each segment, the first two numbers refers to the most positive end of that axis, and the second two numbers refers to the most negative end of that axis.

The two numbers representing the markers are used in the following way. If the number is minus 1, then there is no anatomical marker that is used for the purpose of defining that end of the axis. Otherwise, the first number indicates the segment on which the marker lies. The second number gives the number of the marker on that segment (where the markers are numbered in the order they are given within the raw data file).

Thus for the example given here, for the foot segment, there are three anatomical landmarks that are used. That is, a marker defining the most negative end of the y axis (the third marker of the foot segment) and markers defining the z axis (the medial marker being the first marker of the calf segment and the lateral marker being the second marker of the calf segment – note that in the Klein Horsman model, the positive z axis points from medial to lateral).

**m02\_03\_horsman\_landmarks.cpp**

Calculate the longitudinal limits of each segment based upon the inputted landmarks, the Klein Horsman muscle model and the assumptions as to joint centres. Define the origins of each segment as follows: pelvis origin is the midpoint of ASISs; thigh origin is the hip joint centre; calf origin is the knee joint centre; and foot origin is the ankle centre. This is described in more detail in the description of the model in Chapter 5.

**m02\_02\_make\_lcs.cpp**

This routine finds the vectors that represent the LCS of each segment. Once these vectors have been found, they are then used to calculate the rotation from the LCS to the calibration position of the subject.

|  |  |
| --- | --- |
| make\_segments.txt | 5 2  0 2  0 2  0 2  1 0  1 2 |

This file specifies the instructions for creating each segment. The first row gives the size of the matrix that follows. Each following row represents one segment, starting from segment 0. The two numbers describe the order in which axes should be made. So in the example given, segment 0 is created by first finding the x axis, then the z axis and finally the y axis.

The landmarks to be used to create each axis have already been specified. These are then used to create the intermediate unit vectors representing each axis. The axes are then created as follows. The first axis is created by taking the cross product of the intermediate vectors representing the other two axes. Next, the second axis to be created is found by taking the cross product of the other two axes. The final axis is defined to be the intermediate vector that was used to represent it in the former calculations. The cross products of axes are calculated using the following conventions:

*x = y* x *z*

*y = z* x *x*

*z = x* x *y*

**m02\_03\_01\_calc\_seg\_pos\_vectors.cpp**

This routine finds the positions of the COM and distal point on each segment in the GCS for each frame. It also calculates the vector from the origin to the distal point for later use in the inverse dynamics calculations (also in the GCS).

**m02\_04\_01\_scaling.cpp**

This routine calculates the parameters that will be used to scale the Klein Horsman data set to the subject’s specific geometry. The first operation performed here is to create LCSs of the Klein Horsman segments, using the same methodology as for the subject. This requires the location of the same anatomical landmarks in the Klein Horsman data set. These are specified in the file horsman\_landmarks.txt:

|  |  |
| --- | --- |
| horsman\_landmarks.txt | 15 6  -1 -1 -1 -1 -1 -1  0.079 -0.804 0.028 0.198 -0.908 0.013  0.045 -0.816 0.046 0.112 -0.792 0.010  -1 -1 -1 -1 -1 -1  0.054 -0.402 0.011 0.079 -0.804 0.028  0.032 -0.400 0.055 0.077 -0.405 -0.032  -1 -1 -1 -1 -1 -1  0 0 0 0.054 -0.402 0.011  0.032 -0.400 0.055 0.077 -0.405 -0.032  0.038 0.088 0.042 -0.113 0.086 -0.045  -1 -1 -1 -1 -1 -1  0.038 0.088 0.042 0.038 0.088 -0.221  -1 -1 -1 -1 -1 -1  -1 -1 -1 -1 -1 -1  -1 -1 -1 -1 -1 -1 |

The first two numbers give the size of the matrix that follows. In the following matrix, each group of 3 rows represents a segment (starting from segment 0) and each row represents an axis (first the x axis, then the y axis, then the z axis). On each row there are six figures – representing two groups of three figures – that is the coordinates of up to two anatomical landmarks which are used to define that axis). On each row, the first set of coordinates represents the most positive landmark on that axis (so for instance, for the x axis, the most anterior landmark). Where there are groups of three -1s, these indicate that there is no landmark specified.

The instructions for creating each LCS are given in the file horsman\_instructions.txt:

|  |  |
| --- | --- |
| horsman\_instructions.txt | 5 2  0 2  0 2  0 2  1 0  1 2 |

The format of this file is exactly the same as the file make\_segments.txt. The Klein Horsman segments are created based upon these instructions in the same way as the LCSs of the subject, using the Klein Horsman landmarks.

This file also calculates the length scaling factors based on the comparison of the dimensions of the Klein Horsman segments to the subject’s segments.

**m02\_05\_01\_joint\_flexion.cpp**

This routine finds the joint angles between each segment. This is based upon the definition of all joint angles being zero when the subject is stood in the calibration position. A quaternion is then found which represents the relative rotation of the distal segment relative to the proximal segment from this calibration position. Finally, this quaternion is decomposed into three Euler angles representing the joint angles. The order of rotation (about the body fixed axes of the distal segment) is first about the x axis, then the y axis and then finally the z axis.

**m02\_05\_patella\_segment.cpp**

This routine finds the vectors that represent the LCS of each segment. Once these vectors

|  |  |
| --- | --- |
| patella\_parameters.txt | 9 6  0.06 0.02 0.001 0 0 0  0.049 -0.051 -0.0003 0 0 0  -0.001 0.05 0.0006 -0.001 0.047 0.0006  20.4 -0.26 0 0 0 0  10.88 -0.23 0.0019 -0.000006 0 0  0 0.15 -0.0019 0.000007 0 0  5.59 0.66 0 0 0 0  1.63 0.067 0.00014 -0.000005 0 0  1.43 0.106 -0.00345 0.00005 -0.0000002 0 |

This file contains the parameters used to create the patellofemoral joint model (described in detail in Chapter 5). The first line specifies the size of the matrix that follows. Following this, the numbers in the 9x6 matrix represent the following:

(1,1): length of the patellar tendon (in the Klein Horsman data)

(1,2): the thickness of the patella

(1,3): radius of quadriceps tendon wrapping cylinder

(1,4) – (1,6): 3 dummy zeros

(2,1) – (2,3): position of patellar tendon insertion on the tibia in the tibial local frame

(2,4) – (2,6): 3 dummy zeros

(3,1) – (3,3): position of quadriceps tendon insertion on the patella in the patellar local frame

(3,4) – (3,6): position of the most anterior part of the quadriceps tendon wrapping cylinder in the patellar local frame in the anatomical position

Rows 4-9: specify the regression relationships between the knee flexion angle and the following parameters (see Chapter 5 – description of patellofemoral joint model)

* + - Row 4: patellar tendon sagittal plane angle (*a0,* *a1,* then 4 dummy zeros)
    - Row 5: patellar tendon coronal plane angle (*b0* = 10.88, *b1* = -0.233, *b2* = 0.00189, *b3* = -0.00000569, then two dummy zeros)
    - Row 6: patellar tendon strain (*c0* = 0, *c1* = 0.153, *c2* = -0.00186, *c3* = 0.00000749, then two dummy zeros)
    - Row 7: patellar flexion (*d0* = 5.59, *d1* = 0.6601, then 4 dummy zeros)
    - Row 8: patellar tilt (*e0* = 1.628, *e1* = 0.0667, *e2* = 0.000144, *e3* = -0.00000537, then 2 dummy zeros)
    - Row 9: patellar rotation (*f0* = 1.427, *f1* = 0.1056, *f2* = -0.00345, *f3* = 0.000054656, *f4* = -0.00000023756, then 1 dummy zero)

cross product of the other two axes. The final axis is defined to be the intermediate vector that was used to represent it in the former calculations. The cross products of axes are calculated using the following conventions:

**m02\_05\_02\_patella\_perf\_parameters.cpp**

This routine finds the vectors that represent the LCS of each segment. Once these vectors

**m02\_05\_03\_patella\_contact.cpp**

This routine finds the vectors that represent the LCS of each segment. Once these vectors

## Module 3: Calculate Model Kinematics (m03\_00\_calculate\_kinematics.cpp)

The purpose of this module is to calculate the kinematics of each segment based upon the changes in the position and orientation data of each segment. These calculations rely heavily upon the kinematic formalism described by Dumas and colleagues5, and upon quaternion algebra. This can be found on pages 163-164 of the article:

Dumas, R., Aissaoui, R. & De Guise, J. A. (2004). A 3D generic inverse dynamics method using wrench notation and quaternion algebra. *Computer Methods in Biomechanics and Biomedical Engineering, 7,* 159-166.

**Inputs**

* Positions and orientations of each segment saved in Segment \*\*segment\_data[]
* Anthropometry of each segment saved in Segment \*\*segment\_data[]

**Processes**

Calculation of kinematics from position and orientation data (by consideration of the changes in position and orientation over time).

**Outputs**

* Kinematic variables for each segment saved in Segment \*\*segment\_data[]
  + segment\_data[][]->omega
  + segment\_data[][]->alpha
  + segment\_data[][]->joint\_angles
  + etc.

**Module Overview**

**Selected Detailed Software Description**

**m03\_00\_set\_anth.cpp**

Takes the anthropometry of each segment from the calibration Structure \*calibrate\_pos[] object and saves it in the Segment \*\*segment\_data[] object for each segment and each frame.

**m03\_01\_diff\_q.cpp**

Calculates the first and second derivatives of the quaternion representing the orientation of each segment, and saves it in Segment \*\*segment\_data[] for each segment and each frame.

**m03\_02\_calc\_angular\_quantities.cpp**

Calculates the angular velocity and acceleration of each segment and saves it in Segment \*\*segment\_data[] for each segment and each frame.

**m03\_05\_diff\_origin.cpp**

Calculates the first and second derivatives of the vector specifying the origin of each segment, and saves it in Segment \*\*segment\_data[] for each segment and each frame.

**m03\_03\_calc\_seg\_pos\_vectors.cpp**

Calculates the position of the COM and the distal end of each segment in the GCS.

**m03\_04\_calc\_linear\_quantities.cpp**

Calculates the linear acceleration of the COM of each segment for each frame and saves it in Segment \*\*segment\_data[].

**m03\_04\_calc\_linear\_quantities.cpp**

Calculates the joint angles between segments (based upon the Cardan sequence of rotation z-y’-x’’).

Note that the above routines make extensive use of the functions of the Segment class (that are described in more detail in the appendices).

## Module 4: Calculate Segment Kinetics (m04\_00\_inverse\_dynamics.cpp)

The purpose of this module is to perform an inverse dynamics analysis using a method based upon the work of Dumas and colleagues5:

Dumas, R., Aissaoui, R. & De Guise, J. A. (2004). A 3D generic inverse dynamics method using wrench notation and quaternion algebra. *Computer Methods in Biomechanics and Biomedical Engineering, 7,* 159-166.

In the first place, this analysis produces the inter-segmental forces and moments based entirely on the work of Dumas et al. The routine also calculates all known inter-segmental parameters for the equations of motion that need to be solved in an optimization based approach to the inverse dynamics1–3. These calculations are described in more detail in the publications below:

Cleather, D. J., & Bull, A. M. J. (2011). An optimization-based simultaneous approach to the determination of muscular, ligamentous, and joint contact forces provides insight into musculoligamentous interaction. *Annals of Biomedical Engineering, 39,* 1925-1934*.* doi: [10.1007/s10439-011-0303-8](http://dx.doi.org/10.1007/s10439-011-0303-8)

Cleather, D. J., Goodwin, J. E., & Bull, A. M. J. (2011). An optimization approach to inverse dynamics provides insight as to the function of the biarticular muscles during vertical jumping. *Annals of Biomedical Engineering, 39*, 147-160.doi: [10.1007/s10439-010-0161-9](http://dx.doi.org/10.1007/s10439-010-0161-9)

Cleather, D. J., Goodwin, J. E., & Bull, A. M. J. (2011). Erratum to: An optimization approach to inverse dynamics provides insight as to the function of the biarticular muscles during vertical jumping. *Annals of Biomedical Engineering, 39,* 2476-2478*.* doi: [10.1007/s10439-011-0340-3](http://dx.doi.org/10.1007/s10439-011-0340-3)

**Inputs**

* Kinematic data and external forces saved in Segment \*\*segment\_data[]
* Anthropometry of each segment saved in Segment \*\*segment\_data[]

**Processes**

Solution of wrench based equations of motion described in Dumas et al.5

**Outputs**

* Inter-segmental forces and moments saved in Segment \*\*segment\_data[]
  + segment\_data[][]->
  + segment\_data[][]->
* Parameters for optimization based inverse dynamics analysis saved in Segment \*\*segment\_data[]
  + segment\_data[][]->
  + segment\_data[][]->

**Selected Detailed Software Description**

**m04\_00\_inverse dynamics.cpp**

Simply loops sequentially through each segment and frame, calling the routine that solves the wrench based equations of motion. Iterates from the distal to proximal segment, establishing the force and moment acting on the distal end of the segment by using the proximal values of the previous (more distal) segment and a consideration of Newton’s third law.

**Segment::seg\_solve\_wrench\_eqns.cpp**

Calculates the wrench based equations of motion for the given segment.

## Module 5: Calculate Muscle Geometry (m05\_00\_muscle\_model.cpp)

The purpose of this module is to calculate the kinematics of each segment based upon the changes in the position and orientation data of each segment. These calculations rely heavily upon the kinematic formalism described by Dumas and colleagues5, and upon quaternion algebra. This can be found on pages 163-164 of the article:

Dumas, R., Aissaoui, R. & De Guise, J. A. (2004). A 3D generic inverse dynamics method using wrench notation and quaternion algebra. *Computer Methods in Biomechanics and Biomedical Engineering, 7,* 159-166.

**Inputs**

* Positions and orientations of each segment saved in Segment \*\*segment\_data[]
  + segment\_data[][]->origin
  + segment\_data[][]->rot\_lcs\_to\_gcs
* Anthropometry of each segment saved in Segment \*\*segment\_data[]

**Processes**

**Outputs**

**Module Overview**

**Selected Detailed Software Description**

**m05\_01\_pick\_muscle\_model.cpp**

Takes the a

|  |  |
| --- | --- |
| **Input File** | **Example File Content** |

*Model Details*

|  |  |
| --- | --- |
| horsman\_muscle\_model.txt | add\_b\_p1  2  0.0086 -0.1008 0.0205 3 0 0  -0.0185 -0.0834 0.0181 2 0 0  0.00019 0.054 0.078 0.13962634  add\_b\_p2  2  ... ... .... |

This file specifies the detail of each muscle element given in the model. For each muscle, firstly the name of the muscle is specified. Next the number of points specifying the path of that muscle is given (e.g. origin and insertion and any via points). Following this, each subsequent line gives the detail for one of the muscle points, starting at the most proximal end and working distally. On each of these lines, the first three numbers specifies the location of that point within the LCS of the segment upon which it is located. The next number indicates which segment the point is located on. The next two numbers. Finally, the last line gives some parameters relating to the architecture of the particular muscle.

**m05\_04\_wrap\_cylinder.cpp**

This routine calculates the wrapping of a muscle about its wrapping cylinder (if one is specified), using the method of Charlton and Johnson. 6 The parameters of the wrapping cylinders are defined in wrapping\_cylinders.txt (and have been previously input and stored as part of the routine m05\_02\_horsman\_muscle\_model.cpp):

|  |  |
| --- | --- |
| wrapping\_cylinders.txt | 3 8  0.061 -0.402 -0.018 -0.37 0.04 0.93 0.025 -1  -0.050 -0.078 0.072 -0.21 0.14 0.79 0.039 1  0 0 0 -0.46 0.055 0.886 0.001 1 |

The first two numbers specify the size of the matrix that follows. Then, each row of the following matrix describes a wrapping cylinder. The muscles that use a wrapping cylinder specify which one by giving the row number (as described in the description of the horsman\_muscle\_model.txt file). On each row, the first triple gives the location of a point on the central axis of the wrapping cylinder and the second triple gives a vector along this axis. This vector should be orientated laterally. The seventh number gives the radius of the wrapping cylinder. Finally, the eighth number specifies the direction the muscle should wrap about the cylinder. That is, if observing the model from the right hand side (lateral side) then -1 represents an anti-clockwise wrapping and 1 a clockwise wrapping.

Note that, for the quadriceps, the wrapping point is defined during the creation of the patellofemoral joint model.

§ 5: Detailed Model Description

This chapter provides a detailed description of the lower limb model provided with the FreeBody software.

**St Mary’s University Knee Model**

**Patellofemoral Joint**

The position and orientation of the patella is a function of the tibiofemoral joint flexion angle (*θ*) as follows. Firstly, the location of the patella is established:

1. The insertion of the patellar tendon on the tibia is specified in the tibial local frame using the data of Klein Horsman et al. (adjusted to the local tibial frame of the Klein Horsman model in the same way as the muscle parameters and scaled according to the subject’s dimensions).

PT insertion (tibial LCS): PT*insert*(x,y,z) = (0.0490,-0.0509,-0.0003)

1. pt sag ang.tifThe orientation of the patellar tendon in the sagittal plane of the tibial frame is calculated based upon the knee flexion angle.

PT sagittal plane angle (tibial LCS): PT*sagang =* *a0* + *a1*.*θ*

where *a0* = 20.4, *a1* = -0.26. These coefficients are derived from the data of Kobayashi et al. 7 The direction of a positive sagittal plane angle is indicated by the arrow on the figure.

[Scaling of a0]

1. The orientation of the patellar tendon in the coronal plane of the tibial frame is calculated based upon the knee flexion angle.

PT coronal plane angle (tibial LCS): PT*corang =* *b0* + *b1*.*θ* + *b2*.*θ2* + *b3*.*θ3*

where *b0* = 10.88, *b1* = -0.233, *b2* = 0.00189, *b3* = -0.00000569. These coefficients are also derived from the data of Kobayashi et al. 7 A positive coronal angle indicates that the proximal end of the patellar tendon is more medial than the distal end.

1. The length of the patellar tendon is determined. Firstly, the patellar tendon length from the Klein Horsman data set (0.0605 m) is scaled based upon the subject’s dimensions (using the tibial frame scaling factors). The patellar tendon strain is then calculated based upon the knee flexion angle.

PT strain: PT*strain* = *c0* + *c1*.*θ* + *c2*.*θ2* + *c3*.*θ3*

where *c0* = 0, *c1* = 0.153, *c2* = -0.00186, *c3* = 0.00000749. These coefficients are again based upon the data of Kobayashi et al. 7

1. The parameters calculated in 1-4 above, now allow the location of the patellar tendon origin (on the patella, and in the tibial frame) to be calculated by geometric considerations. Given the position of the patellar tendon’s origin and insertion in the tibial frame, their position in the global frame can then be calculated. The geometry of the patellar tendon has thus been established.
2. The origin of the patellar tendon is taken to be the origin of the patella segment. Thus the location of the segment has now been determined.

Next, the orientation of the patella is established, again by reference to the knee flexion angle:

1. The orientation of the patella is given based upon a set of three Euler angles. The sequence of rotation is as follows: 1) patellar flexion about the z axis (of the femoral LCS); 2) patellar tilt about the y axis; 3) patellar rotation about the x axis. These three angles are calculated using the following regression equations which were developed from the data of Nha et al. 8 (figure taken from Nha et al.).

Patellar flexion: PT*flex* = *d0* + *d1*.*θ*

where *d0* = 5.59, *d1* = 0.6601

Patellar tilt: PT*tilt* = *e0* + *e1*.*θ* + *e2*.*θ2* + *e3*.*θ3*

where *e0* = 1.628, *e1* = 0.0667, *e2* = 0.000144, *e3* = -0.00000537

Patellar rotation: PT*rot* = *f0* + *f1*.*θ* + *f2*.*θ2* + *f3*.*θ3* + *f4*.*θ4*

where *f0* = 1.427, *f1* = 0.1056, *f2* = -0.00345, *f3* = 0.000054656, *f4* = -0.00000023756



 (1)

 (2)

Where



 if the muscle does not attach to or cross the segment

 for uniarticular muscles

adapt for rotation

 for biarticular muscles [check]

(3)

*bji* = 1 for biarticular muscles that cross but do not attach to segment i;

*bji* = 0 for all other muscles

 for i>0

 (4)

 (5)

Definition of terms

|  |  |
| --- | --- |
|  | linear acceleration of the centre of mass |
|  | vector from the proximal joint to the segment COM |
|  | vector from the proximal to the distal joint |
|  | cost function |
|  | magnitude of force in muscle |
|  | acceleration due to gravity |
|  | segment/joint number (numbering from distal to proximal) |
|  | inertia tensor |
|  | muscle or ligament number |
|  | joint contact force at proximal end of segment |
|  | cost function coefficients |
|  | magnitude of force in ligament |
|  | mass of segment |
|  | inter-segmental moment at proximal end of segment |
|  | cost function exponents |
|  | line of action of biarticular muscle j about segment i |
|  | line of action of muscle j about joint i |
|  | line of action of ligament j about joint i |
|  | moment arm of muscle j about joint i |
|  | inter-segmental force at proximal end of segment |
|  | moment arm of ligament j about joint i |
|  | total number of muscles |
|  | total number of ligaments |
|  | total number of joints |
|  | angular velocity of segment |
|  | angular acceleration of segment |

Get up and running on optimization basis from v1

Add biarticular muscles opt bit

Add version based on joint forces moments

Add variable tib joint

Check and document

Notes

patella seg -> other points 1 is tibial tuberosity, other points 0 is the quad tendon insertion

References

1. Cleather, D. J., Goodwin, J. E. & Bull, A. M. J. An Optimization Approach to Inverse Dynamics Provides Insight as to the Function of the Biarticular Muscles During Vertical Jumping. *Ann. Biomed. Eng.* **39,** 147–160 (2011).

2. Cleather, D. J., Goodwin, J. E. & Bull, A. M. J. Erratum to: An Optimization Approach to Inverse Dynamics Provides Insight as to the Function of the Biarticular Muscles During Vertical Jumping. *Ann. Biomed. Eng.* **39,** 2476–2478 (2011).

3. Cleather, D. J. & Bull, A. M. J. An Optimization-Based Simultaneous Approach to the Determination of Muscular, Ligamentous, and Joint Contact Forces Provides Insight into Musculoligamentous Interaction. *Ann. Biomed. Eng.* **39,** 1925–1934 (2011).

4. De Leva, P. Adjustments to Zatsiorsky - Seluyanov’s segment inertia parameters. *J. Biomech.* **29,** 1223–1230 (1996).

5. Dumas, R., Aissaoui, R. & de Guise, J. A. A 3D generic inverse dynamic method using wrench notation and quaternion algebra. *Comput. Methods Biomech. Biomed. Engin.* **7,** 159–166 (2004).

6. Charlton, I. W. & Johnson, G. R. Application of spherical and cylindrical wrapping algorithms in a musculoskeletal model of the upper limb. *J. Biomech.* **34,** 1209–1216 (2001).

7. Kobayashi, K., Sakamoto, M., Hosseini, A., Rubash, H. E. & Li, G. In-vivo patellar tendon kinematics during weight-bearing deep knee flexion. *J. Orthop. Res.* **30,** 1596–1603 (2012).

8. Nha, K. W. *et al.* In vivo patellar tracking: Clinical motions and patellofemoral indices. *J. Orthop. Res.* **26,** 1067–1074 (2008).

Appendices

## Appendix A: Class Descriptions

## Appendix B: Library Routines

## Appendix C: List of Variables/Arrays/Objects

|  |  |
| --- | --- |
| Vector Objects | Description |
| Vec\_DP coord\_map |  |
| Vec\_DP g(3) | Gravity vector |
| Mat\_DP make\_segment\_info |  |
| Mat\_DP model\_structure |  |
|  |  |
|  |  |
|  |  |

|  |  |
| --- | --- |
| Objects | Description |
| Muscle \*\*muscle\_data[frames] |  |
| Muscle \*muscle\_model[muscles] |  |
| Segment \*\*segment\_data[frames] |  |
| Structure \*calibrate\_pos[segments] |  |
|  |  |
|  |  |
|  |  |