5-2014

Total Knee Arthroplasty Wear Is Caused by Malrotation and Excessive Laxity

Erik Woodard
University of Tennessee Health Science Center

Total Knee Arthroplasty Wear Is Caused by Malrotation and Excessive Laxity

DOI
10.21007/etd.cghs.2014.0357

Total Knee Arthroplasty Wear Is Caused by Malrotation and Excessive Laxity

A Thesis
Presented for
The Graduate Studies Council
The University of Tennessee
Health Science Center

In Partial Fulfillment
Of the Requirements for the Degree
Master of Science
In the Joint Graduate Program in Biomedical Engineering and Imaging
From The University of Tennessee
and
The University of Memphis

By
Erik Woodard
May 2014

DEDICATION

This thesis is dedicated to my parents for providing motivation and encouragement throughout my pursuit of an engineering curriculum, and to all the friends I've made along the way.

ACKNOWLEDGEMENTS

I would like to acknowledge my advisor, Dr. William Mihalko, for his guidance and instruction. My fellow graduate student, Casey Hebert, deserves significant recognition for his help with Matlab code, assisting during testing, and providing moral support. Many thanks to the graduate students who operated and collected data from the knee machine before I got here – their data were invaluable toward the completion of this thesis.

ABSTRACT

Total knee arthroplasty is a proven technique which combines specially designed components and surgical processes to treat cartilage degeneration and alleviate pain in arthritic knees. However, this technique is limited by component design and surgical precision. Due to these limitations, knee arthroplasty components will eventually wear out, causing rejection and necessitating the need for a replacement. For this reason, it would be beneficial to experts if the primary causes of this wear could be identified in order to minimize the number of replacements.

This study aims to determine if a correlation exists between instability of a knee joint and the amount of wear present in an implant, and also relate surgical alignment to this wear. To accomplish this aim, a custom laxity machine was used to assess joint stability in 20 knees of human bodies donated to science. This laxity data was compared to damage scores expressing the amount of wear on each implant specimen, and was used in conjuction with alignment data obtained from CT scans. Alignment data was expressed as the difference in component rotations, as well as a new method here named "congruency mismatch".

A significant correlation was found between wear and anterior and posterior laxity, indicating the need for additional constraint in implant design to minimize sliding which can lead to wear. No significant relationships were observed between either alignment analysis technique and wear scores. Results do show a positive postoperative relationship between external femoral rotation and increasing varus coronal angle, which is inversely related to previous studies which were undertaken preoperatively. Implant functionality and successful outcomes are directly related to design and proper surgical technique, which can be quantified and improved using new methods such as patient-specific design and robotic surgical systems.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ACL	Anterior cruciate ligament
AP	Anterior/Posterior
CR	Cruciate retaining
CT	Computed tomography
IE	Internal/external
LCL	Lateral collateral ligament
MCL	Medial collateral ligament
PCA	Posterior condylar axis
PCL	Posterior cruciate ligament
PMMA	Poly(methyl methacrylate)
PS	Posterior stabilized
STEA	Surgical transepicondylar axis
TKA	Total knee arthroplasty
TTA	Tibial tuberosity txis
UC	Uni-condylar
VV	Varus/valgus

CHAPTER 1. INTRODUCTION

Overview of Total Knee Arthroplasty

Total knee arthroplasty (TKA) is a technique where the cartilaginous articular surfaces of the distal end of the femur and proximal end of the tibia are replaced with metal and polyethylene materials in the attempt to allow the patient to have a pain free joint that functions to allow participation in activities of daily living. The procedure now is performed in over 600,000 patients in the United States and in over a million patients worldwide for the end stage treatment of rheumatoid or osteoarthritis [1]. Many different TKA designs exist, all of which attempt to improve the kinetic and kinematic function of the native knee in some way. Several components and materials, such as a metal and polyethylene bearing surface, are common to all TKA implants.

Soft tissue balancing (obtaining proper ligament tension) during total knee arthroplasty is a step that most surgeons agree is paramount to assuring longevity of the procedure. This is critical to maintaining a stable implant and knee joint after surgery, which is essential to successful patient outcomes. Important structures of the knee joint that must be considered for balancing are the medial collateral ligament (MCL), the lateral collateral ligament (LCL), the posterior capsule, and the posterior cruciate ligament (PCL), although many other structures play a role as well. The primary structures and their locations are shown in **Figure 1-1**.

Surgeons must align components and balance all soft tissue structures in both extension and flexion of the knee joint during TKA surgery. Because certain tissue structures tend to contract and loosen as the knee is flexed, a properly balanced knee in extension is not always an indicator of stability in flexion. Range of motion is also an important consideration; flexion of at least 115° is typically desired in the United States but can be much higher in some Asian countries [2].

Design of TKA Components

Many different TKA designs exist to address patient-specific deficiencies discovered before or during surgery. For example, the posterior cruciate ligament (PCL) is often found to be degenerated during surgery; thus, in this case it can be beneficial to utilize a TKA design which is manufactured with a post on the plastic tibial insert which interacts with a round cam on the metallic femoral component at approximately in knee flexion. This mechanism attempts to replicate the tightening of the PCL at flexion angles past 60 degrees. The flexion angle at which the post and cam interact is implant-specific and varies according to design [3]. The result of this mechanism is to translate the tibia anteriorly during activities requiring deep flexion. This design is known as a posterior stabilized (PS) knee replacement, shown in **Figure 1-2**. In contrast, a TKA implant that does not recommend removal of the PCL is known as a cruciate retaining (CR) design.

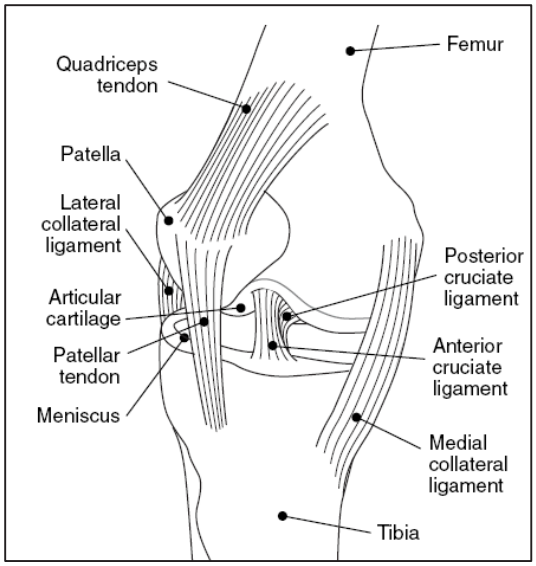

Figure 1-1: Major supporting soft tissue structures of the knee joint which constrain motion and affect laxity. These may need to be adjusted during TKA surgery if a deformity is present. Source**:** Reprinted with permission. **http://en.wikipedia.org/wiki/Knee#mediaviewer/File:Knee_medial_view.gif** Accessed 6/9/2014 [4].

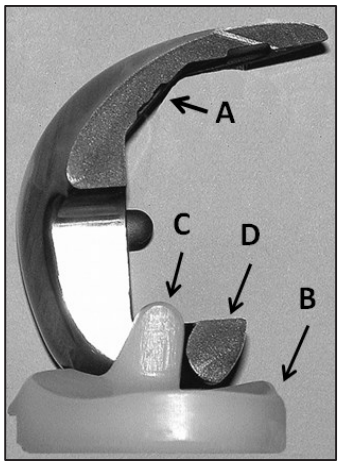

Figure 1-2: Typical design of a posterior stabilized (PS) implant. The metal femoral component (A) has been cut in half for visualization of the tibial post (C) and femoral cam (D). In flexion, the cam impinges on the spine of the post and prevents excessive posterior translation of the tibia. This simulates the PCL in a normally functioning knee. The femoral component rides on the articulating surface of the polyethylene tibial insert (B). Source: Modified with permission. Nakayama, K., et al., *Contact stress at the post-cam mechanism in posterior-stabilised total knee arthroplasty.* J Bone Joint Surg Br, 2005. 87(4): p. 483-8 [5].

Due to the nature of TKA surgery, most designs require the removal, or sacrifice, of the ACL. The ACL provides additional constraint in extension but does not contribute support during flexion. This creates an inherent deficiency in TKA design, although implants can be designed with increased constraint to compensate for the loss of the ACL. This can be accomplished by creating a more dished surface on the tibial insert, which can cause less translation and rotation of the femoral component.

Soft Tissue Balancing and Laxity

To better understand how and why a surgeon balances the soft tissues during TKA surgery, it is useful to know the motions of the knee joint as described by surgical and anatomical terms. Many soft tissues contribute to laxity of the knee joint, and each can tighten or loosen the knee in several different anatomical planes and in extension, flexion, or both. Excessive tension or laxity of these ligaments is often the cause of a varus or valgus deformity of the joint which can be a contributing factor to cartilage degeneration prior to correction with TKA, and failure to address this cause will likely lead to an unbalanced TKA, resulting in excessive long-term wear [6]. Anatomic planes are defined in **Figure 1-3**. Clinical motions of the knee joint are described in relation to these planes. The following clinical definitions are defined by movement of the tibia in relation to the femur.

Varus and valgus (VV) motion refers to the rotation of the tibia about the knee joint center in the coronal plane (as viewed from the front). A varus knee refers to a displacement of the distal tibia toward the midline of the body in the coronal plane, resulting in a "bow-legged" appearance with more space between the knee joints. Likewise, valgus knees occur when the tibia rotates away from the midline, resulting in a "knock-kneed" appearance. Both varus and valgus deformities can occur in the knee joint, and surgeons attempt to correct these deformities during TKA implantation primarily by relaxing the MCL or LCL.

Varus and valgus laxity are also affected by PCL tension, as this ligament has been shown to have a secondary effect on coronal laxity. The goal of correcting a varus or valgus deformity is to restore the mechanical axis of the joint. Prior to performing a TKA, the location of the femoral and tibial bone cuts is determined based on the geometry of the patient's joint and the severity of any deformities. This is typically accomplished using a radiograph of the entire leg. From this, the mechanical axis can be located and any necessary modifications to the VV angle of the knee will be apparent, as shown in **Figure 1-4**.

Internal and external (IE) rotation is defined as rotation of the tibia about its long axis with the femur held stationary. The tibial axis is defined as a line drawn from the intercondylar eminence between the medial and lateral condyles (superior) to the center of the medial and lateral malleoli (inferior). The location of the intercondylar eminence is clarified in **Figure 1-5**. Internal rotation is defined as a clockwise rotation of the left

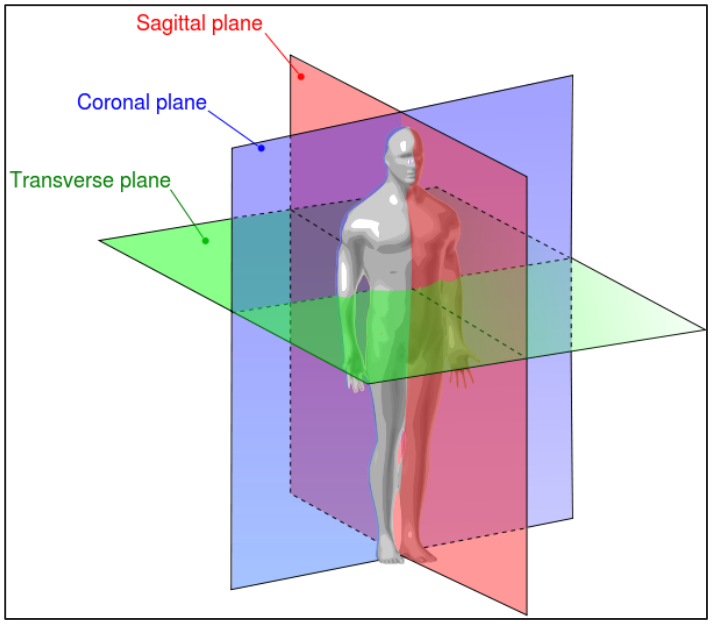

Figure 1-3: Anatomic planes of the human body. Motions of the limbs about joints are classified by movements in these three planes. Source: Reprinted with permission. **http://en.wikipedia.org/wiki/Anatomical_plane#mediaviewer/File:Human_anatomy_planes.svg.** Accessed 6/9/2014 [7].

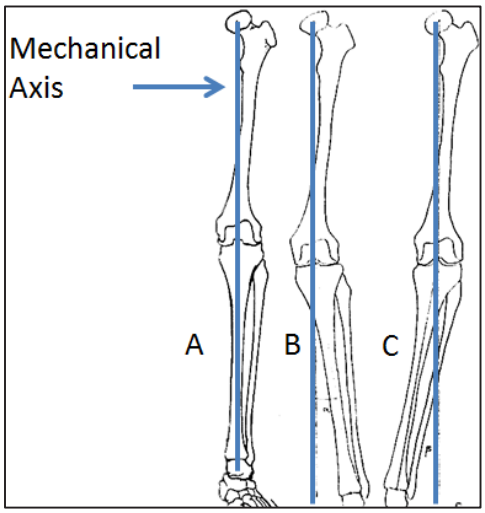

Figure 1-4: Example of the mechanical axis of the knee. Surgeons attempt to restore the knee joint to the center of this line during TKA surgery to correct a varus or valgus deformity. The mechanical axis extends from the center of the head of the femur to the distal center of the tibia. Proper alignment (A), valgus alignment (B), and varus alignment (C) are shown. **S**ource: Modified with permission. Winemaker, M.J., *Perfect balance in total knee arthroplasty: the elusive compromise.* J Arthroplasty, 2002. **17**(1): p. 2-10. [12].

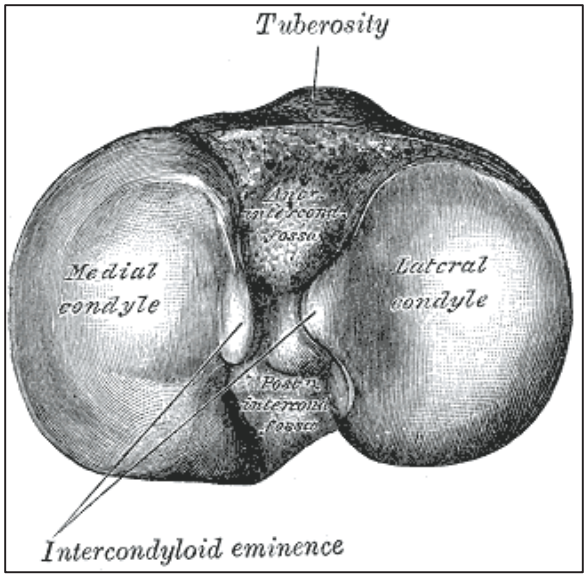

Figure 1-5: Location of the intercondylar eminence on the tibial plateau.
Source**:** Reprinted with permission. http://en.wikipedia.org/wiki/File:Gray257.png Accessed 4/4/2014 [8].

tibia and counterclockwise rotation of the right tibia when viewed from below; external rotation convention is opposite. Many structures contribute to internal and external rotational laxity, including the collateral ligaments and the internal geometry of the knee joint itself. Increasing conformity of the condyles on the articular surface will also produce more resistance to tibial rotation.

Anterior and posterior (AP) laxity is resistance to anterior and posterior (forward and backward) translation of the proximal end of the tibia in relation to a fixed femur. The cruciate ligaments within the knee joint primarily contribute to this laxity. The ACL is tight during extension of the knee and inhibits anterior motion of the tibia. Conversely, the PCL tightens in flexion and restricts posterior motion. Loss of the ACL or substitution of the PCL during TKA with a post and cam design can affect AP laxity. Specifically, a cam and post design will typically restrict posterior motion of the tibia in flexion, attempting to mimic the action of the PCL.

The typical surgical approach to properly balance a knee during TKA involves ensuring an equal gap distance of the medial and lateral compartments in both extension and flexion. This is accomplished by releasing ligaments on the medial or lateral side of the knee, depending on which side is tighter. To correct a valgus deformity, surgeons typically release soft tissues on the lateral side of the knee. These ligaments and tissues include the LCL, iliotibial band, popliteus tendon, and the tendon of the lateral head of the gastrocnemius. If a varus deformity exists, the MCL is typically released. The order in which these tissues are affected has been shown to produce changes in laxity of the joint [9].

As an alternative to releasing ligaments and soft tissue structures, a technique known as "pie crusting" has recently been introduced as a tool to correct varus or valgus deformities with a milder affect than ligament release [10]. This technique may provide advantages to a traditional release, such as increased post-operative stability and shorter recovery times. A recent technique introduced to aid in balancing the knee during TKA is computer-assisted surgery. This method uses guides on the femur and tibia which are tracked in real time and provide feedback to the surgeon regarding proper alignment. This may increase the alignment consistency during TKA, and has been shown to provide better restoration of the mechanical axis than with traditional manual techniques [11].

TKA Component Alignment

In addition to aligning TKA components in the coronal plane, transverse plane alignment is also essential to successful outcomes after surgery. To accomplish this, bony landmarks on the femur and tibia are used to align components. On the femoral side, the surgical transepicondylar axis (STEA) is used. This is defined as a line connecting the lateral epicondylar prominence and the medial sulcus. This line has a relationship with the posterior condylar axis (PCA) in the native knee. The PCA is defined as the line drawn between the two most posterior points on the medial and lateral condyles, and is also referred to as the posterior condylar line (PCL). Both axes are shown in **Figure 1-6**. The PCA is defined in a similar manner for both the femur and tibia; the femoral PCA is referred to here as the fPCA and the tibial PCA will be the tPCA. The average rotation between the STEA and fPCA is 0.3° of internal rotation in women and 3.5° of internal rotation in men [12, 13]. TKA femoral components are often designed with a built-in external rotation of approximately 3° to compensate for this fact and easily restore rotation of the femoral articulating surface to its position in the native knee.

Factors Related to TKA Wear

TKA surgery has been shown to be an effective treatment for the osteoarthritic knee with a high success rate; however, as many as 22,000 revision surgeries are performed annually to replace failed implants [6]. As previously stated, the primary cause of long-term failure in TKA is the loosening of the implant caused by the generation of wear particles. These particles are generated from the surface of the tibial insert as it articulates with the femoral component. Polyethylene does not degrade within the body like many other materials; rather, it accumulates around the implant site and is absorbed by macrophages, which are unable to degrade the particles as they would otherforeign material. This stimulates the macrophages to release inflammatory cytokines which cause biological rejection of the implant.

For the purposes of this study, only factors that can be controlled by surgical technique were assessed and related to wear after implantation. This is due to the large variety of implant designs that were received as part of the testing described here. Two

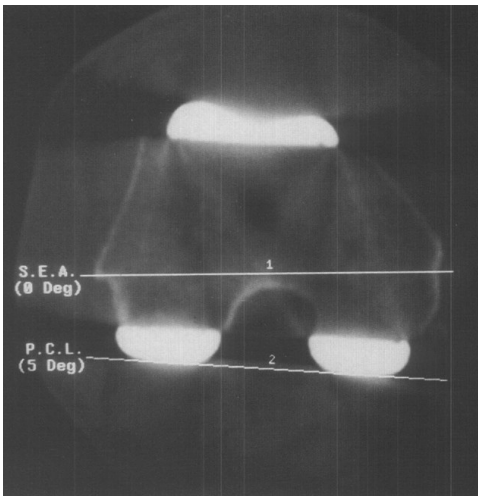

Figure 1-6: Image showing the surgical transepicondylar axis (STEA, 1) and the posterior condylar line (PCL, 2) of the femur after TKA surgery. Source: Modified with permission. Berger, R.A. and L.S. Crossett, *Determining the Rotation of the Femoral and Tibial Components in Total Knee Arthroplasty: a Computer Tomography Technique.* Operative Techniques in Orthopaedics, 1998. 8(3): p. 128-133.[12]

major surgical parameters were assessed in this study: component alignment and laxity due to soft tissue balancing.

Component Sub-optimal Alignment

Sub-optimal alignment of the femoral and tibial components in the transverse plane can lead to increased wear due to incongruity of the articulating surfaces, which causes impingement, especially in the posterior aspect of the condyles. Berger et al proposed a method to measure rotation of both the femoral and tibial TKA components using transverse slices from CT scans and known anatomical landmarks used during surgery to align components. Despite some subjectivity in this method, it has been shown to be accurate and repeatable [14, 15]. Studies have shown that malalignment of the tibial component in the transverse plane is related to chronic pain in patients. Specifically, a correlation was shown between internal tibial component rotation and chronic pain in patients [16]. Furthermore, poor tibial alignment may produce patellofemoral tracking problems, abnormal gait patterns, or increased polyethylene wear [17].

Surgeons typically place the tibial component to achieve maximum coverage of the tibial plateau with minimal overhang. One way to accomplish this is to use the tibial tubercle as a landmark, placing the center of the tibial component at the medial third of the tubercle. This is the traditional method; however, since the tubercle is rotated externally relative to the tibial plateau, it has been shown to create a tendency for external rotation [18]. An analysis of patients has shown that, on average, the tibial tubercle is 18

degrees external to an antero-posterior line drawn through the geometric center of the tibia [12].

Measuring the rotation of femoral and tibial components and determining the congruency of the articulating surfaces are separate issues when examining TKAs after implantation. Previously, rotational mismatch has been defined and reported as the difference in angles between the femoral and tibial components, based solely on visible bony landmarks [19]. This assumes that following bony landmarks during surgery will produce ideal congruency, according to the implant design. Using this method, a correlation was found between significant rotational mismatch and patients who reported experiencing pain after surgery [19].

A modeling study by Mihalko and Williams has also shown that deviations in transverse plane alignment of TKA components from manufacturer specifications results in significantly different kinematics[20]. By importing the specifications of components in to a virtual knee simulator called LifeMod and mimicking soft tissues, the transverse alignment of the femoral and tibial components were deviated by 5° internally and externally from neutral and the resulting kinematics were examined for a lunge simulation. Rotating the components resulted in an increase of anterior and posterior translation throughout flexion. Studies using a custom laxity testing machine have shown that increased external rotation of the femoral component increase internal rotational laxity and constrain external laxity in flexion [21, 22]. The results of these studies demonstrate the importance of using a consistent method to accurately align components during TKA.

Suboptimal Soft Tissue Balancing

During TKA, most surgeons attempt to balance the soft tissues such that the lateral and medial joint gaps are equal. If this is not accomplished through a range of flexion angles, it will lead to increased loading in one compartment, with the implant not functioning as intended. Improper varus and valgus balance also causes instability and incorrect positioning of the patella in the trochlear groove. Excessive laxity leads to force concentrations which are outside of the intended design parameters of the TKA components, and which exceed the material properties of the polyethylene tibial insert.

Currently, most surgeons rely on bony landmarks to perform bone cuts. However, these landmarks may not be accurate; especially in arthritic knees or knees with severe deformities [15]. For this reason, the landmarks used for bone cuts are often disputed or have been shown to be highly variable. Also, the order of ligament release can play an important role in final stability after correcting a varus or valgus deformity[23]. Surgeons must carefully consider the role each ligament plays in both extension and flexion.

To date, a correlation between laxity, component wear patterns, and alignment of the tibial and femoral implant components has not been established. A study by Kretzer

et al showed that increasing laxity using ligament models and a wear simulator leads to increased polyethylene wear [24], but it did not test the full effect of soft tissues, and wear simulation has been shown to underestimate polyethylene damage [25].

Theoretically, suboptimal alignment and poor soft tissue balancing should increase polyethylene wear and decrease implant survivorship, contributing to implant loosening and costly revision surgeries. This study utilizes a retrieval program of functioning TKAs obtained at the time of necropsy. By utilizing CT scans, mechanical laxity testing, and polyethylene damage scores, we aimed to determine if any correlation between proper alignment and ligament balancing to polyethylene damage scores exists.

CHAPTER 2. RETRIEVAL ANALYSIS

Wear in total knee arthroplasty (TKA) remains the single most common cause of revisions. Both polyethylene and metallic wear particles can cause adverse biological reactions which can lead to infection and loosing of components. In vivo polyethylene wear is a multifactorial process which can be caused by excessive or unexpected contact forces, or the method of sterilization used when manufacturing the implant. Gamma radiation has been shown to cause increased wear by introducing oxidation to the polyethylene when placed in the body, leading to delamination of the component surface as well as subsurface damage [26, 27]. Microfractures in the substructure decrease the structural integrity of the component and can cause entire sheets of polyethylene to loosen when shear forces are applied.

Analysis of retrieved implants is a multistep process that can be approached in a variety of ways. To researchers, this analysis is vital to assessing how the implant functioned after surgery. Traditionally, this involves examination of components by one or several experts who then give feedback on severity of observable wear. Engineering approaches attempt to minimize subjectivity by introducing standardized techniques that are able to quantify wear. These techniques include oxidation analysis, punch testing, and μCT analysis of retrieved polyethylene components. Methods such as these provide an objective way to measure wear, and can be used in addition to subjective techniques such as visual inspection.

Visual Inspection

Since the primary source of wear particles in TKA implants is the polyethylene tibial insert, performing a visual inspection of this component when retrieving the implant yields useful insight into the overall condition of the implant at the time of retrieval. A visual inspection is the least complex of all wear analysis methods. When performed by an expert observer, this method can identify the modes and location of wear on the implant surface. A summary of the degree and locations of wear on the implant surface can give an indication of the way the implant functioned when in the body. While visual inspection with the naked eye is sometimes sufficient to identify many types of gross damage, it is often useful to supplement this with inspection under a light microscope to identify damage covering smaller areas. Examples include pitting, which is typically 2-3mm across, and scratches, which can be very thin and difficult to identify without the aid of a microscope.

The gold standard of wear analysis in TKA remains the semi-quantitative wear score technique proposed by Hood et al in 1983 [29]. In this method, the surface of the insert is divided into 10 sectors, 4 on each condyle and 2 in the center region. Wear scar areas are identified in each sector and assigned a score ranging from 0 to 3 based on the surface area covered by wear. Thus, the maximum possible total score for any one

damage type is 30 (a score of 3 in each of the 10 sectors). The total maximum score for any one component is 210 (30 times 7 damage modes).

The Hood method recognizes 7 types of surface damage used when scoring components. These damage modes are based on past visual inspections and types of damage observed, which were compiled by Hood et al and summarized as follows:

- Surface deformation: any permanent deformation on the surface of the insert. This is typically caused by cold flow or creep of the polyethylene.
- Pitting: depressions in the articulating surface.
- Embedded PMMA debris: change in color or texture of the articulating surface due to the deposit of cement used to secure TKA components
- Scratching: long lines occurring in an anteroposterior direction.
- Burnishing: highly polished areas on the articulating surface.
- Abrasion: areas with a shredded or tufted appearance caused by contact with bone or cement
- Delamination: removal of sheets of polyethylene

Two approaches can be taken when reporting damage scores: use scores from a single expert observer for consistency or average scores from two or more observers. When averaging scores, it is essential to also report interobserver error. Hood et al compared wear scores from two observers and used a paired t-test to determine that there were no significant differences between observers. Experienced observers must be able to differentiate between wear which occurred in vivo and that which occurred during removal, due to gouging and scratching from removal tools.

This method is also useful for giving a wear score to retrieved patellar components; these components are divided into 4 sectors and scored using the same grading system. When retrieving patellar components, it is important to note the orientation as well. The orientation of tibial components is usually obvious, but this is not the case for patellar components. This can be accomplished by making a small notch with a scalpel blade on the medial or lateral side during retrieval.

Although wear scoring is traditionally used to analyze the articulating surface of tibial inserts, it can also be adapted to the backside of the inserts. Backside wear has become an increased issue due to micromotion associated with the introduction of modular tibial components. Modular components are advantageous in that they allow surgeons to easily and quickly replace worn polyethylene components during revision surgery; however, depending on the locking mechanism, they may lead to increased motion of the poly component on the baseplate compared to fully cemented designs. It is estimated that this backside wear accounts for 3-50% of total wear volume in modular TKA implants. Backside wear damage modes typically differ slightly from those of the articulating surface. Cold flow through screw holes in the baseplate is possible, and burnishing due to sliding of the insert relative to the baseplate is common [30]. A visual representation of these damage modes overlayed on a retrieved component is shown in **Figure 2-1**.

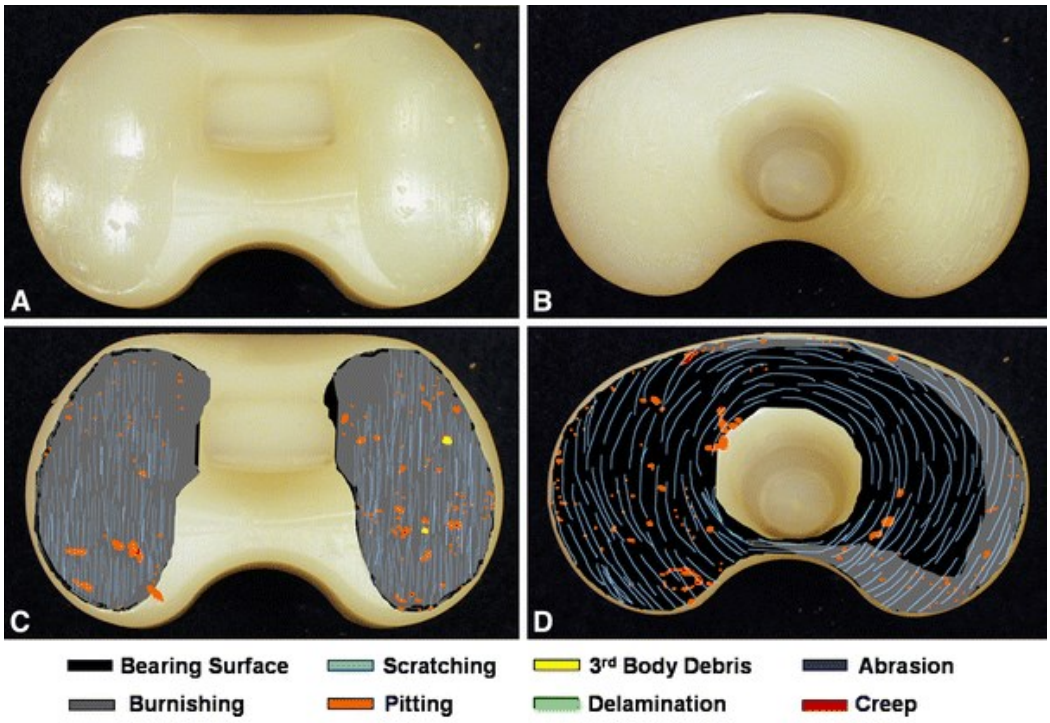

Figure 2-1: Hood grading method applied to a mobile-bearing insert.
Source: Modified with permission. Kelly NH et al. Wear Damage in Mobile-Bearing TKA is as Severe as that in Fixed-Bearing TKA. *Clin Orthop Relat Res* (2011) 469: 123-130. [33]

In 2012, Brandt et al [31]used a modified version of the Hood technique to assess the extent of damage on the backside of 52 retrieved polyethylene tibial components with 3 different locking mechanisms. The grading scale was modified to include both an area score and a severity score for each damage type. Area coverage ranged from 0 to 10, with 0 representing no area covered, 1 being less than 10% of the area covered, and 10 being above 90% coverage. This modification was combined with another refinement made by Wasielewski et al – a severity score. This refinement ranks damage using 1/3 increments on a scale from 0 to 1. These scores were multiplied by the area coverage scores to present the final wear scores as a combination of severity and area coverage by each damge mode.

A major limitation of visual inspection of the implant surface is that these methods only express damage in terms of the area covered by the defect. There is no way to determine the depth to which the damage extends, or the missing volume due to material loss. Thickness measurements can account for some loss of material, but often the original implant thickness must be known to express depth of penetration due to wear [32].

Photogrammetry

A more quantitative extension of the wear scoring technique is photogrammetry. In this method, wear scar areas on the insert are outlined using a marker or other tool and images of the insert are analyzed with image processing software such as ImageJ. This allows individuals to quantify the area covered by a specific type of damage. The surface area coverage can be expressed in mm^2, as opposed to a rough percentage given using the simple visual inspection methods described previously. Areas can be segmented using manual software methods or by using thresholding to detect and isolate regions of interest which change color of brightness. Although less subjective than simple visual inspection, this method still does not give any information on the depth of damage, and calculated surface areas may have significant error due to the concave shape of the tibial insert articulating surface. The equipment and software needed to accomplish this method are readily available and inexpensive. Usually a midrange camera and adequate lighting are sufficient, and open-source image processing software exists for segmentation and analysis.

Thickness Measurements

The thickness of the tibial insert can be measured at different locations using a micrometer, calipers, or other tools which are readily available in most labs. This measurement can then be compared to the thickness at the same location in the original implant, and wear penetration can be expressed as the difference between these two measurements. The major advantage to this technique is similar to that of visual inspection – it is simple and cheap to implement. A possible disadvantage is the lack of access to the original component specifications. Furthermore, current manufacturing tolerances may exceed the accuracy of thickness measurement devices. Thickness changes may be difficult to detect due to the resolution of these devices. Bartel et al demonstrated the importance of maintaining thickness in tibial components by using finite element analysis to measure contact stress while varying implant thickness. They concluded that a minimum thickness of 8-10mm was recommended to maintain contact stresses below the yield stress of tibial inserts.

μCT

Recently, a quantitative method using micro-computed tomography (μCT) has been developed to provide a volumetric measurement of wear in TKA tibial inserts [34]. In this method, a retrieved tibial insert is scanned using a microCT machine, and a 3D reconstruction is produced from the scan. This enables surface and subsurface analysis of the implant at a resolution of 50μm. If a scan of the original, undamaged insert is also available, the volumes of the two inserts can be compared and wear can be expressed as the volume loss after implantation (**Figure 2-2**).

Another advantage of using computed tomography is the ability to see possible

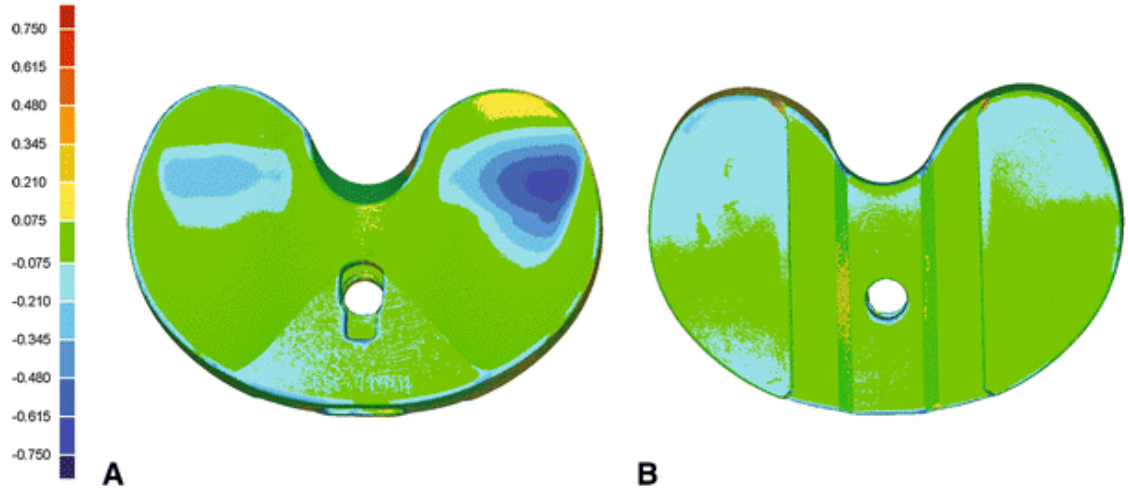

Figure 2-2: Surface deviation map for a worn insert compared to the unworn insert. Deviations of (A) bearing surfaces and (B) backside are in millimeters
Source: Modified with permission. Teeter, M.G., et al., *In vitro quantification of wear in tibial inserts using microcomputed tomography.* Clin Orthop Relat Res, 2011. **469**(1): p. 107-12. [34].

subsurface damage in the insert (**Figure 2-3**). Small subsurface cracks can be an indication of future surface damage and weakening of the overall structure of the implant. A major limitation of this technique is the requirement of the scan of the original component to determine volumetric differences. Bowden et al [35] measured wear of acetabular liners by fitting the worn inserts to an idealized hemisphere of the same dimensions. Tibial inserts present an additional challenge when determining volumetric loss; their complex geometry means that fitting techniques used with hip implants are not appropriate. Typical scans also take longer compared to other analysis techniques (several hours per scan). The expense of a μCT scanning machine can also be a limiting factor. However, this may be offset by the ability to almost completely automate retrieval analyses.

Femoral Component

Few methods exist to express the degree of wear present in metallic femoral components. Wear of these components can be just as problematic as that of tibial inserts, since metal ions have been shown to accumulate at the implant site and cause allergies over time. Since the volume of the femoral component is unlikely to decrease in any measurable way, it is not feasible to use quantitative techniques such as microCT imaging to determine the amount of wear. Therefore, the most effective method to express the degree of wear in these components remains visual inspection. One challenge with this method arises due to the very small nature of wear in metallic components.

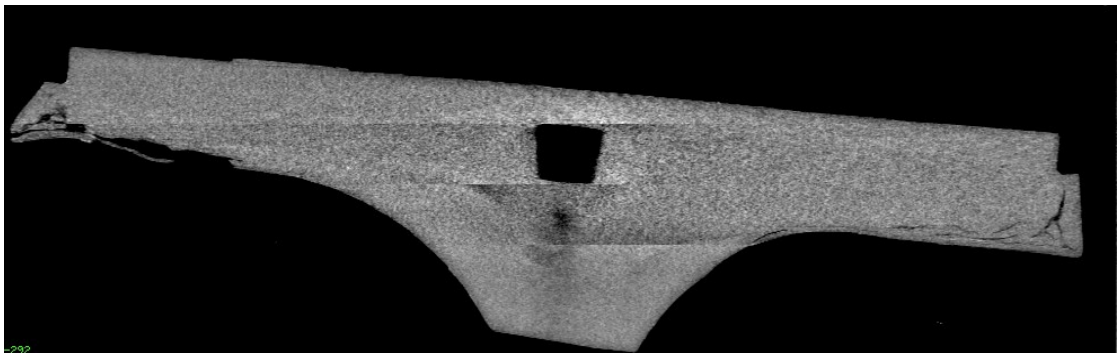

Figure 2-3: Example of subsurface cracking visible in a microCTimage slice of a tibial insert.

Types of wear are likely to be scratches and pitting which have features not visible to the naked eye. Additionally, conventional photographs taken of the femoral component often do not depict damage features due to the reflective surface.

Another complication of assigning wear scores to metallic femoral components is discerning which damage occurred in vivo and which occurred during explanting due to tool marks. Heyes et al performed wear scores on 15 retrieved oxidized zirconium femoral components using four wear types (scratching, delamination, pitting, and striation) and dividing the component surface into 5 distinct areas. They found removal damage on many of the components. This damage was typically more severe than in vivo wear and was discernable by identifying mediolateral damage not suggestive of in vivo wear.

One way to overcome the challenge of viewing the small wear features on the femoral component is to use scanning electron microscopy (SEM) to view the articulating surface. Using this method, the component is placed in the field of view of the microscope, grounded with carbon tape, and then an electron beam is aimed at the surface. This beam excites atoms at the surface of the specimen, and the resultant emitted secondary electrons are detected by the microscope and used to create an image of the surface features. Using this technique, it is possible to identify surface features not visible to the naked eye. High levels of magnification are possible (up to 6 orders of magnitude). SEM images can also be used to identify embedded debris and very small scratches on the implant surface (**Figure 2-4**), as well as provide a secondary way to distinguish between in vivo damage and damage occurring during explanting.

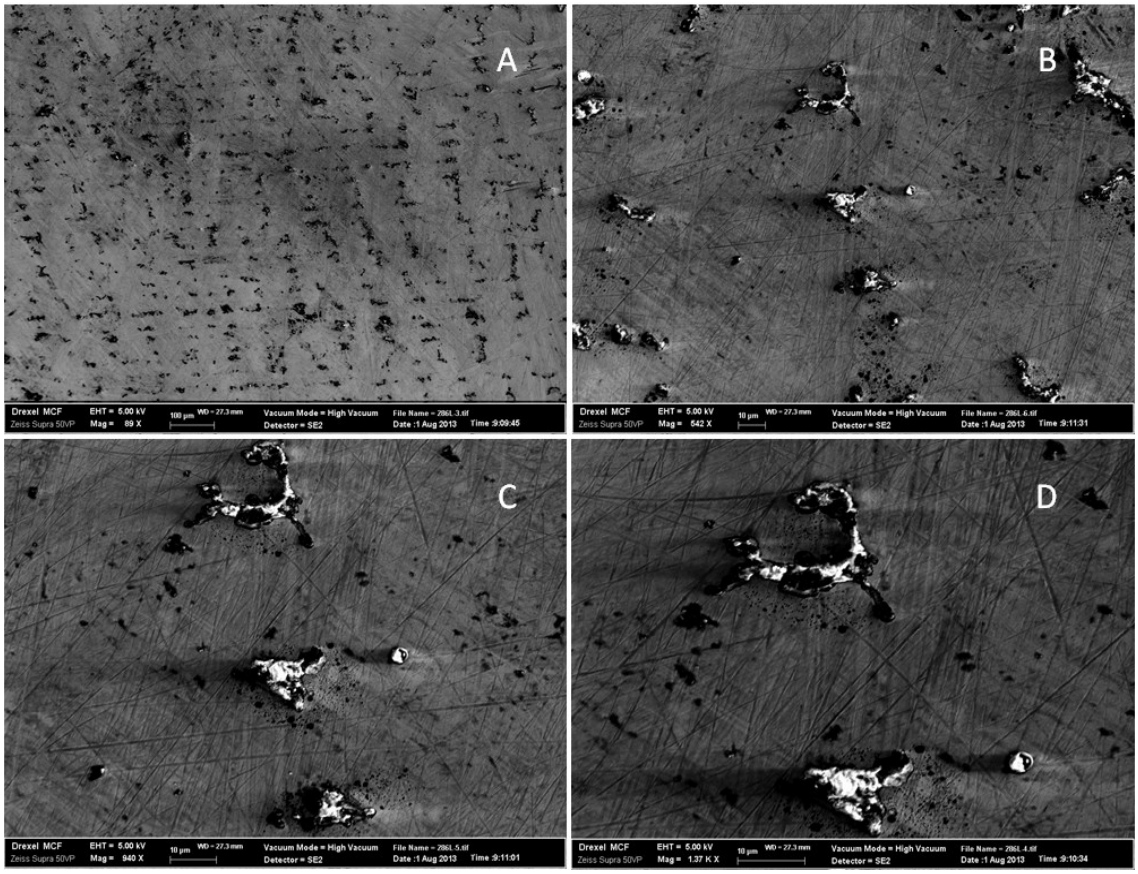

Figure 2-4: SEM images of the articulating surface of a femoral component. Images are magnified at 89X (A), 542X (B), 940X (C), 1.37K X (D). Scratches and debris can be seen.

CHAPTER 3. METHODOLOGY

Study Design and Overview

The purpose of the study was to attempt to determine if a correlation exists between transverse plane alignments of TKA components and wear of the polyethylene tibial insert, as well as correlate laxity in various planes to tibial insert wear in a series of TKA retrievals obtained at the time of necropsy.

Due to the many steps involved in this study required to obtain and process data, an overview of the process is provided that explains the individual steps in detail. The knees of donated human bodies to science which previously undergone TKA surgery were obtained through the Medical Education and Research Institute (MERI, Memphis TN) with approval through a previous IRB consent. Computed tomography (CT) scans were performed at the Semmes Murphy Neurologic and Spine Institute (Memphis TN) using a GE Brightspeed scanner with a resolution of 512x512 pixels and a transverse slice thickness of 1.25mm. All of the intact donated bodies were placed on the bed of the scanner and a scout radiograph was performed to locate the hip and ankle joint and set these landmarks as the upper and lower bounds of the CT scan, respectively.

The cadavers were then transported to the MERI for retrieval of the knee joint for further testing. Dissection and retrieval were performed by a fellowship-trained, board-certified orthopaedic surgeon (Dr. William Mihalko, Campbell Clinic Orthopaedics). All skin and muscle tissue was cut and removed from the area surrounding the knee joint capsule, while carefully preserving the ligaments and other structures which contribute to stability of the joint. The joint center was identified and the femur and tibia were cut transversely 180mm superior and inferior to the joint center, respectively with a sagittal saw (Stryker). The fibula was also transected 100mm from the joint center.

Retrieved specimens were labeled and placed in biohazard bags to be stored in the freezer until testing. Knee specimens were removed from the freezer and thawed in a refrigerator 36 hours prior to laxity testing. Laxity testing was performed on a custom knee testing platform designed and manufactured in 1991 by Paul McLeod. This testing apparatus was acquired by Dr. John William's laboratory at the University of Memphis in 2009, and restored to full working condition by October 2010. It was then transferred to Dr. William Mihalko's laboratory in the Coleman building at the University of Tennessee Health Science Center in April 2013. Laxity data for 8 of the 20 knees shown here was obtained by previous graduate students at the University of Memphis prior to moving the machine. For convenience, this testing platform will here be referred to as the Memphis Knee Simulator (MKS). This apparatus is designed to test the internal/external (IE), varus/valgus (VV), and anterior/posterior (AP) laxity of a cadaveric knee joint with the dimensions previously specified. The function and workings of this machine are further described in the next section. Deflection data corresponding to knee laxity for each of these testing conditions were recorded.

CT scans for each knee specimen were then analyzed to determine the rotation of both the femoral and tibial TKA components. Angle measurements were performed using an open-source image analysis program (ImageJ). Appropriate slices containing anatomic landmarks were identified using DICOM viewing software provided by Semmes Murphy, then converted to .jpeg files for further analysis in ImageJ. The complete rotation measurement procedure is described in a subsequent section.

After laxity testing, a parapatellar incision was made on the medial side of the joint capsule to gain access to the TKA components. Dr. Mihalko retrieved the polyethylene tibial insert from each specimen, and any abnormal conditions at the time of retrieval, such as excessive wear or oxidation, were noted. Tibial components were then soaked in a 10% bleach solution for 30 minutes, followed by a wash in Alconox for 2 minutes. The cleaned inserts were inspected under a light microscope at 10X magnification and assigned a numerical score to quantify wear based on the protocol proposed by Hood et al.

The steps used in this procedure can be summarized as follows for each TKA specimen:

1. Perform CT scan of specimen
2. Retrieve knee joint with intact TKA from cadaver
3. Load knee into MKS
4. Perform laxity testing
5. Retrieve tibial insert
6. Perform wear score on insert

The full procedure is explained in greater detail in the following sections, and summarized in **Figure 3-1**.

Laxity Testing Platform and Procedure

There are many contributing factors to the overall stiffness of the knee joint capsule. The primary factor is the ligamentous structures which form the capsule, as well as internal ligaments such as the ACL and PCL. Another important factor contributing to joint stiffness is the conformity of the TKA implant design. Many testing platforms have been developed to quantify the laxity of the knee joint in both cadaveric and living models. Typically, laxity is tested at flexion angles of 0, 30, 60, and 90 degrees. This provides a good summary of laxity through a range of flexion. Several standard testing procedures have been developed to report the laxity envelope in various planes of motion. The MKS is capable of performing these procedures and recording laxity data for each specimen.

All forces measured by the MKS are measured using a pair of strain gauges at the desired location. These include sensors in the ankle box for measuring IE torque and body weight and a sensor at the end of the VV adjustment crank to measure force which

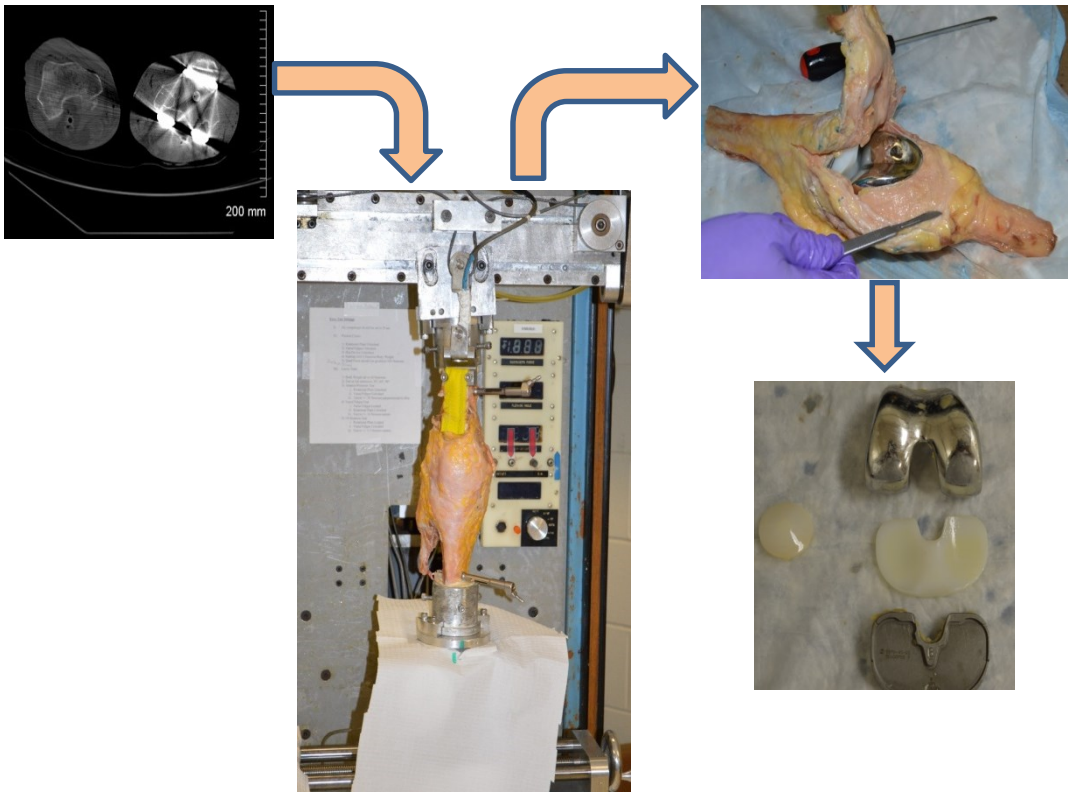

Figure 3-1: Overview of the CT scan, laxity testing, and retrieval process.

is then converted to VV torque using a set of equations. These strain gauges form two arms of a Wheatstone bridge, and function as separate resistors. A resistance difference between these gauges affects the voltage output of the Wheatstone bridge for increased sensitivity. This difference is then amplified using a National Instruments 2B31J instrumentation amplifier. The gain can be fine-tuned using set screws on the back of the amplifier cards. This feature is used to calibrate the machine prior to testing. The final output voltage readings are sent to a 12 bit data acquisition (DAQ) device which digitizes the signals then displays them in a custom Labview program.

The voltages are converted to useful force, angle, and displacement values using a calibration protocol. In this protocol, voltage readings are taken at known forces and angles and multiplied by a constant and adjusted with an offset using a linear calibration. Forces and angles are recorded at set intervals and a linear calibration is used to determine the slope and offset needed. These numbers are input to a separate Labview calibration program for each sensor, and the adjusted output is recorded in a main Labview program. The known forces and torques are acquired using force sensors and a torque wrench which are factory-calibrated. Angles are obtained using a Johnson magnetic angle locator placed parallel to the surface of the ankle box.

As previously described, the femur and tibia are cut 180mm from the joint center in preparation for laxity testing. This is useful because the moment arm is known when force is applied in the apparatus. The femur and tibia are then potted using a two part epoxy in metal couplings designed to be later fixed to the machine. The femur and tibia are first centered in the couplings using three pointed screws, and then the epoxy is poured into the coupling and allowed to set for 30 minutes until completely hardened. Specimens were then placed with the tibia mounted vertically in the machine, and the femoral coupling locked in a neutral position as defined by the vertically placed tibia. Neutral rotation was defined as placing the femoral epicondylar axis perpendicular to the centerline of the machine. Considering the possibility of flexion contracture due to bedridden specimens, 0 degrees of flexion was not always obtained. In these cases, extension was defined as the minimum flexion angle of the knee specimen after fixing it in the machine.

The femur is flexed by loosening the mechanism on the femoral crosshead (**Figure 3-2B**). As the femur is flexed, the tibia experiences an equal amount of flexion, which is output from the angle sensor mounted in the ankle flexion box (**Figure 3-2A**). Total flexion is the addition of the two flexion angles from the hip and ankle flexion sensors.

During all tests, a 30N upward vertical force is maintained at the distal end of the tibia. Force is applied using compressed air routed to the underside of the bottom stage of the testing platform. The air pressure is regulated to maintain a set upward vertical force. This force does not follow the tibial axis during knee flexion; rather, it is constantly vertical with respect to the machine axis. Three different laxity tests were performed with the knee at extension and at 30, 60, and 90 degrees of flexion:

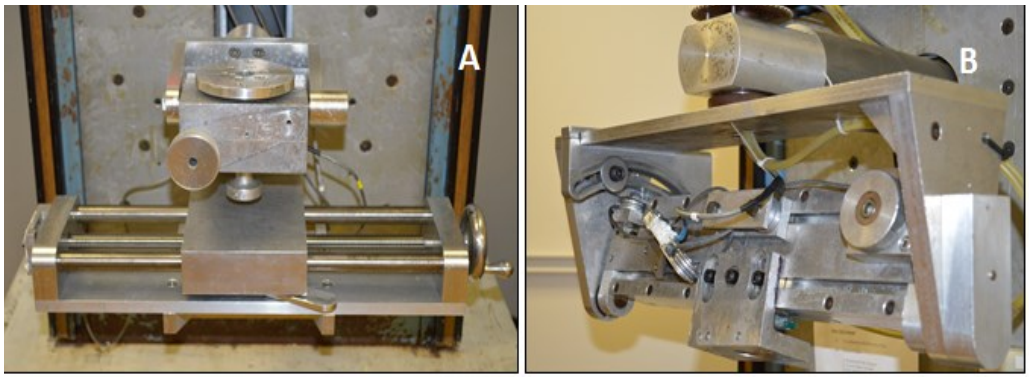

Figure 3-2: Ankle (A) and femoral (B) flexion simulators on the knee machine.

1. Varus/Valgus laxity test

For this test, a force was placed at the distal end of the tibia perpendicular to the tibial axis. During this test, the femur was fixed and the tibia was free to rotate about the joint center in the coronal plane, as well as rotate about its own axis. The tibia was displaced by applying a horizontal force to the ankle box using a hand crank connected to a screw mechanism. The moment was calculated as the applied force multiplied by the moment arm, which was defined as the length of the tibia from the joint center (180mm) plus the distance from the end of the tibia to the center of the ankle box (65mm) and the sensor to the ankle box (145mm). Thus, the moment arm was 390mm at 0° of VV rotation and flexion.

This distance was adjusted as the moment arm decreased due to flexion angle or varus/valgus angle. This was accomplished by multiplying the cosine of half the flexion angle by the sine of the varus/valgus angle and multiplying the result by 145mm (the distance from the ankle box to the sensor). This value was then subtracted from the total moment arm and multiplied by the calibrated force output from the strain gauge, resulting in the corrected torque output. The desired moment of the tibia about the joint center for this test was 10Nm. The VV angle of the tibia in the coronal plane was recorded by a sensor in the ankle box.

2. Internal/External rotation test

This test was performed by applying a rotational torque about the tibial axis in the transverse plane. The torque was applied using a handle on the front of the ankle box which is connected to the tibial coupling by a gear (**Figures 3-2** and **3-3**). The distal end of the tibia was free to move in the coronal plane while rotating about its long axis. The tibia was rotated clockwise until 1.5Nm of torque was recorded by the strain gauge at the distal end, and then rotated counterclockwise until the reading is -1.5Nm. Angular displacement of the distal tibia about its axis was recorded, as well.

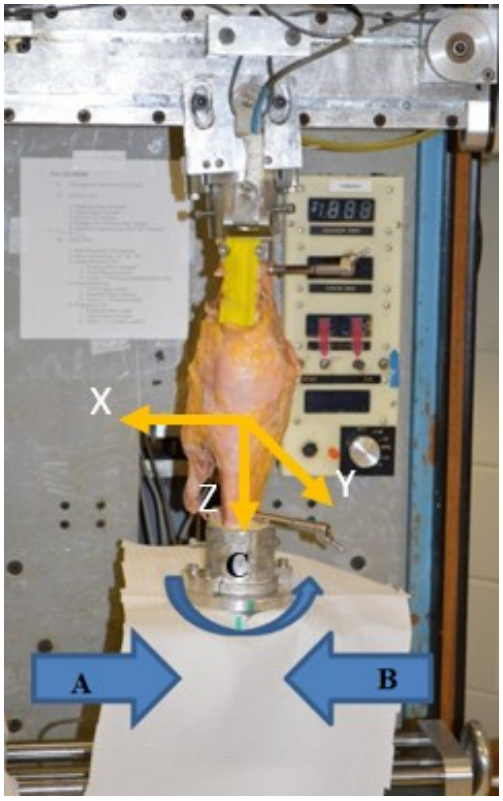

Figure 3-3: Image of a TKA specimen placed in the laxity testing machine in extension. The femur is superior to the tibia, and both bones are potted in custom fixtures which are then positioned and bolted to simulated hip and ankle joints. Arrows show varus motion (A), valgus motion (B) and internal rotation (C). The coordinate system originates at the tibial intercondylar eminence, and the z axis follows the long axis of the tibia. Movements are defined by tibial motion, since the femur is fixed for all tests.

3. Anterior/Posterior drawer test

An additional AP arm was used for this test. The device was plugged into the front of the machine. A strain gauge within the arm was connected to a spring; in this way, the displacement of the arm is related to the force applied by knowing the spring constant and using **Equation 3-1**:

$$F = kx \qquad \textbf{(Eq. 3-1)}$$

where k is the spring constant of the mechanism. The arm was attached to the tibia at the most prominent medial point and immediately anterior to the superior attachment of the fibula on the lateral side by using adjustable pointed screws (**Figure 3-4**). The arm was held perpendicular to the long axis of the tibia during all tests, so it was necessary to reposition it when adjusting the flexion angle. The operator slowly pulls the arm, resulting in a positive force and displacement output. Force was increased until the reading was at least 35N. The arm was then pushed, resulting in a negative force and displacement of the tibia. All recorded displacements measure tibial motion in relation to the femur. In both cases, the distal end of the tibia was free to move in the coronal plane and rotate about its long axis, while the proximal end of the femur remained fixed.

During each laxity test, data was saved using the Labview program to a .lvm file, which is a Labview specific format equivalent to a text file. Data was saved in labeled columns indicating the nature of the data source. Data columns were as follows:

Figure 3-4: The arm attachment for quantifying anterior/posterior translation in TKA specimens. The other end of the cable attaches to a port on the front of the machine which transmits voltage readings associated with force and displacement. Force is directed perpendicular to the tibial axis.

- Relative read time
- Flexion angle (volts)
- Flexion angle (degrees)
- Quad force (volts)
- Quad force (Newtons)
- Body weight (volts)
- Body weight (Newtons)
- Pressure (volts)
- Pressure (Pascals)
- Varus/Valgus angle (volts)
- Varus/Valgus angle (degrees)
- Varus/Valgus torque (volts)
- Varus/Valgus torque (N-m)
- Rotational angle (volts)
- Rotational angle (degrees)
- Rotational Torque (volts)
- Rotational torque (N-m)
- A/P distance (volts)
- A/P distance (mm)
- A/P force (volts)
- A/P force (N)

The .lvm file was first converted to a text file with a .txt extension, and then converted to a .xlsx Excel file using the Kneetxt2xls.m Matlab program. This is a simple program which recognizes any .txt file in the Matlab directory and changes the file extension to .xlsx while preserving all the data and formatting in the files.

Once all files from one knee test had been converted to Excel files, the Matlab program kneetest_info.m was run to extract data of interest and perform calculations to identify points of interest in the data from the files. The main program separated the files into AP, IE, and VV data based on a standard file naming convention. It then called several subprograms named kneetest_AP, kneetest_IE, and kneetest_VV to perform calculations on individual tests. For all tests, sign convention followed the Grood and Suntay coordinate system. This system specifies that anterior displacement of the tibia in relation to the femur is positive, posterior is negative, internal rotation is negative, external is positive, and varus motion is negative while valgus is positive. The tibia and femur each have a separate coordinate system; however, since the femur is fixed during all these tests, it is only necessary to define tibial motion in its own coordinate system.

Kneetest_AP found the anterior or posterior displacement of the tibia at 35N. This was accomplished by searching through values in the A/P force column, measured in Newtons, until two values bordering +35N were located. These two values, along with

the two associated values in the A/P displacement column, were used to linearly interpolate the displacement at +35N. If the force bordered +35N at more than one index in the column, the interpolated values at each location were averaged and the final displacement at +35N was stored as a variable denoted "anterior displacement". Likewise, the same process was used to calculate displacement at -35N and this value was stored as "posterior displacement".

Since Labview program has no sign correction performed on rotation data for right or left knees, data must be sorted by left or right conventions before any rotation laxity calculations are performed by the kneetest_IE program. This was accomplished by recognizing the "L" or "R" character in the file name and separating data accordingly, then assigning the appropriate sign convention specified by Grood and Suntay. The IE program then performed a similar calculation as the AP program, this time finding the angular rotation of the tibia when a rotational torque of 1.5Nm was applied.

Varus and valgus sign convention was corrected within the Labview program prior to recording data from the knee machine. The user selected a left or right leg designation using a toggle in the Labview program and the sign of VV torque was changed accordingly. This means that no correction was necessary when performing calculations within the kneetest_VV program. Varus angles were always negative, and valgus were always positive, so it was not necessary to sort files based on their naming convention for VV calculations.

Once all the IE and VV angles and AP positions were calculated, they were stored in cells then written to new Excel files. The convention for each file name consisted of the data type (AP, IE, VV) followed by "keyinfo.xlsx". This way, the data summaries could easily be located within the Matlab directory after the program was finished running. The Excel files also contained columns with labels for the original filenames, the standard deviation of the calculated angle or position, the body weight force applied at that angle or position, and the value of the maximum and minimum applied force or torque.

Measuring Component Rotations

All component rotation measurements were performed using the method proposed by Berger et al in 1998 [12]. CT scans were viewed slice by slice using a DICOM image viewing program provided with each scan by Semmes Murphy. For each rotation measurement procedure, slices were viewed to find the highest quality slice based on visibility of anatomic landmarks and overall contrast. To measure femoral component rotation, one CT image was found in which both epicondylar prominences were clearly visible (**Figure 3-5**), as well as the sulcus directly below the medial epicondyle. This appeared as a notch or indentation. A line drawn between the lateral epicondyle and the medial sulcus was defined as the surgical transepicondylar axis (TEA). In addition to these features, the image was also required to clearly display the posterior condyles of the femoral component. A line drawn connecting both posterior condyles is referred to as the

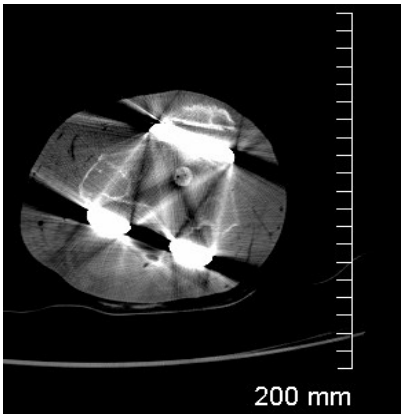

Figure 3-5: Example of a DICOM image of a right TKA used to measure femoral rotation. The lateral epicondylar prominence is visible on the right side, with the medial sulcus on the left side just below the medial prominence.

posterior condylar axis (PCA). Since the metallic component shows up as an artifact in the scan, it is impractical to select an image with the entire component in view, as this will obscure other features of interest. Therefore, it is convenient to use a slice further from the proximal end of the femur.

Femoral Rotation

Once proper images of the femoral components were selected, the images were saved as Jpegs and opened using open-source image processing software (ImageJ, NIH), where all further analysis was performed. The steps to determine femoral rotation were as follows:

1) Sharpen the image in ImageJ by clicking process > sharpen
2) Draw a line from the lateral epicondylar prominence to the medial sulcus. This is the TEA. Hold Control + D to save the line.
3) Draw a line between the posterior condyles of the femoral component. This is the PCL. Click on the center of this line and drag it upward till it intersects the TEA. Hold Control + D as before.
4) Use the angle tool to make an angle using a point on each line and the vertex formed by the intersection of the TEA and PCA. Click analyze > measure to display the angle formed, displayed in degrees (**Figure 3-6**).
5) Repeat the process 5 times and average the angle measurements. Ensure the standard deviation is no more than 0.5. If it is, repeat the measurements.
6) Internal rotation of the femoral component is defined as negative, and external rotation is assigned a positive value.

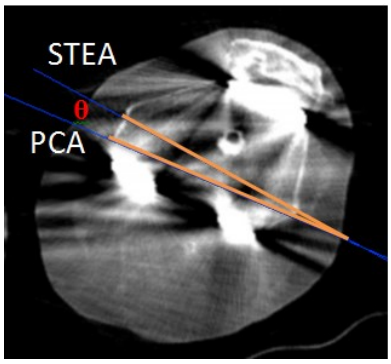

Figure 3-6: Image showing the surgical TEA and PCA and femoral component rotation, defined by the angle between these two lines (θ). The PCA line has been moved upward such that it intersects the TEA. If θ is 3°(male) or 0.5°(female) and the PCA is rotated externally to the STEA, the femoral component is considered to be in neutral alignment.

Tibial Rotation

A similar method was used to measure the rotations of the tibial inserts, with the major exception that this measurement required three separate images. The first image needed to clearly show the tibial plateau without artifacts from the metal tibial tray. This was used to find the geometric center of the tibia along the long axis. The geometric center was then transposed to a second image showing the tibial tubercle. A line drawn from the center of the tibia to the edge of the tubercle is known as the tibial tuberosity axis (TTA).

This line was then transposed to another image showing only the polyethylene insert. A line was drawn between the posterior condyles and this line was rotated 90 degrees clockwise. The angle between this line and the tubercle line was measured as the rotation of the tibial insert. The normal rotation of the tibial component is 18° (±2.6°) of internal rotation. Thus, an internal rotation of 18° is considered neutral. A summary of the steps to determine tibial component rotation are as follows:

1) Open 3 images in ImageJ clearly showing the tibial plateau, tibial tubercle, and outline of the tibial component.
2) Find the geometric center of the tibial plateau using two intersecting lines (**Figure 3-7**). Add these lines to the overlay manager by clicking Image > Overlay > Add Selection.
3) Click the image of the tibial tubercle to make it the primary image. Add the geometric center lines to this image by selecting them in the region of interest (ROI) manager and holding Ctrl +D.
4) Draw a line through the intersection of the geometric center lines and the tip of the tubercle, defined as the point on the tubercle farthest from the center. This is the TTA (**Figure 3-8**).

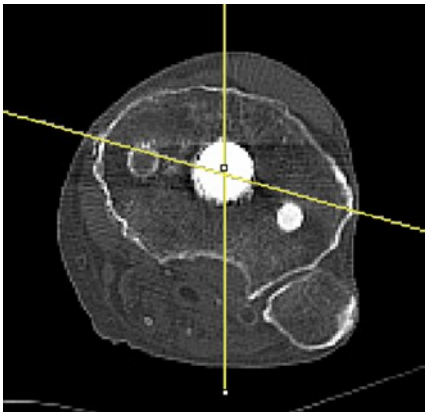

Figure 3-7: Location of the geometric center of the tibial plateau using the first image. The center is located at the intersection of the two lines.

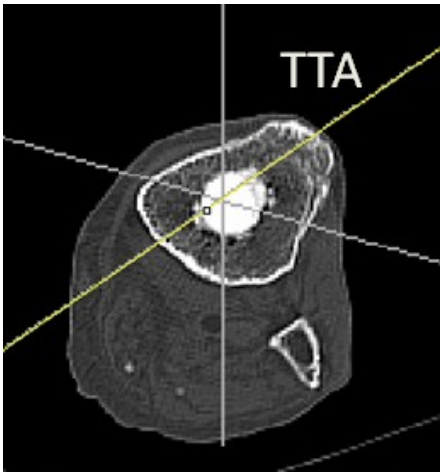

Figure 3-8: Location of the tibial tuberosity axis on the second image. This line intersects the geometric center and the center of the tibial tuberosity.

5) Add the TTA to the overlay as described in step 2 and click the third image showing the tibial insert.
6) Add the TTA to the insert image using the ROI manager and draw it there. Draw a line connecting the bottom edges of the posterior condyles on the tibial insert. This is the PCA of the insert (**Figure 3-9**).
7) Rotate this line by clicking Edit > Selection > Rotate, then type 90 degrees and click Ok. Draw the line in this location. Measure the angle between the rotated PCA and TTA (**Figure 3-10**).
8) Repeat the process 3 times, taking the difference between the measured angle and 18° of rotation. For instance, a measured angle of 23° internal rotation would a rotation of 5° from neutral position. Internal rotation is defined as negative and external is positive.
9) Ensure that the standard deviation of the 3 measurements is no more than 0.5.

For all rotation measurements, internal rotation was defined as negative and external rotation was designated positive. To calculate **component mismatch**, tibial rotation was subtracted from femoral rotation. Therefore, an externally rotated femoral component and an internally rotated tibial component led to an increase in mismatch, and vice versa.

In addition to calculating component mismatch defined by the difference in femoral and tibial rotations with respect to anatomical landmarks, a new definition of rotation without respect to landmarks was desired for this study. Therefore, **congruency mismatch** is here defined as the angular difference of the fPCA and the tPCA. This measurement should provide a representation of the mismatch of the femoral component and the polyethylene tibial insert. This was accomplished by drawing the fPCA line, then transposing this line to the image slice with the tibial insert, drawing the tPCA, and measuring the angle between the two lines (**Figure 3-11**). The process was repeated 3 times, and the angle measurements were averaged to determine average congruency mismatch.

Retrieval and Wear Analysis

After completion of laxity testing on each cadaveric specimen, retrieval was performed by a board-certified, fellowship-trained orthopedic surgeon (Dr. William Mihalko). Any abnormality in the components or surrounding tissue was noted, as well as the design and other component characteristics. First, the tibial insert was removed. The removal technique varied with the design of this component, but typically it involved depressing the plastic locking tab on the anterior side of the implant. Excessive amounts of visible wear on the tibial inserts were noted, the most common being delamination and oxidation. Since the implants were typically 10-15 years old, it is likely that they were sterilized using gamma radiation, which has been shown to cause oxidation over time.

As the femoral component and tibial tray were removed using a hammer and chisel, the level of fixation was observed and recorded. This level ranged from “not

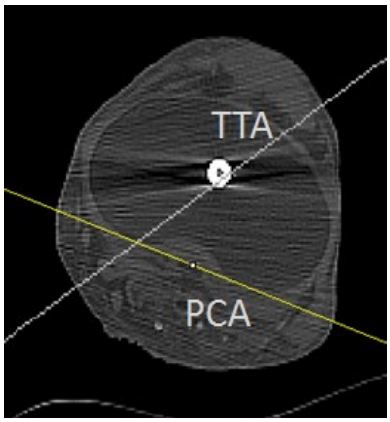

Figure 3-9: Location of the posterior condylar axis relative to the tibial tuberosity axis. The PCA is drawn between the two most posterior points on the condyles of the tibial insert.

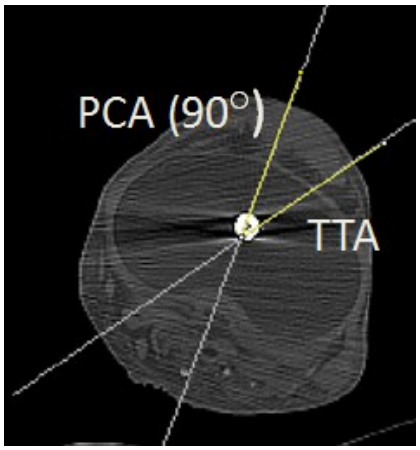

Figure 3-10: The PCA is rotated 90 degrees, and its intersection with the TTA determines the rotation angle of the tibial insert. This insert exhibits 23 degrees of internal rotation from the TTA, with 18 degrees considered neutral.

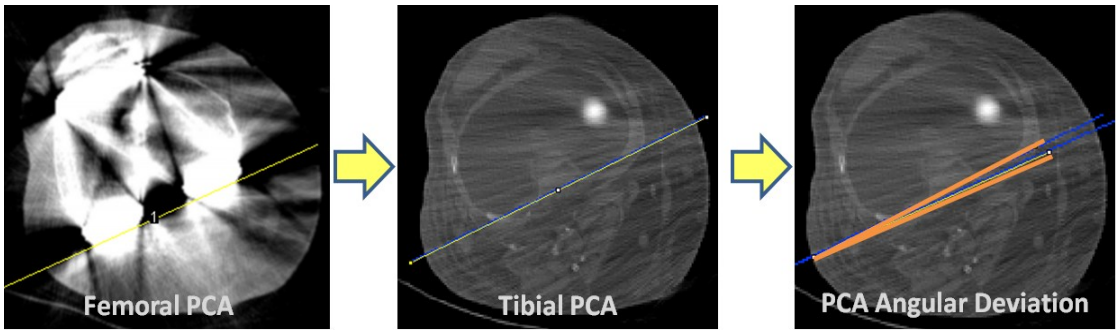

Figure 3-11: Process of measuring congruency mismatch using image slices of the femoral and tibial components.

fixed", meaning the component was very loose, to "extremely well fixed", meaning the component was difficult to remove. Two tibial trays were also fixed using two screws to secure the baseplate into the tibial bone. Three of the femoral components did not use PMMA as a fixation mechanism; these implants had a porous coating to encourage bone ingrowth.

The polyethylene patellar component was removed by inserting a flathead screwdriver between the bone and plastic and prying around the outside edge. Since this was usually the most difficult component to remove, it was often necessary to grip the patellar tendon, from which muscle tissue has already been removed prior to laxity testing, in a vice to better facilitate removal of the patella. The condition of the articular surface and number of pegs of each patella was noted, and a small notch was made with a scalpel on the lateral side. All retrieved components were then placed in 500mL of a 10% bleach solution for 30 minutes to sterilize them and remove surface debris. All components were then removed from solution and rinsed with deionized (DI) water.

The femoral component and tibial tray were then placed in 500mL of a 50% acetone solution and allowed to soak for at least 24 hours to soften the remaining PMMA. While these were soaking, the polyethylene components and any other components which did not have any remaining PMMA were washed in an Alconox solution to remove any bodily fluids which remained after the bleach soaking. This was useful because although the components were sterilized using bleach, a slimy layer often remained which could only be removed using a soap solution. Although the polyethylene components were lightly scrubbed, care was taken not to damage them or disrupt the wear areas which were already present on the surface. After washing in the soap solution, components were once again rinsed with deionized water and placed in labeled plastic bags. They were considered completely clean at this point.

After soaking for at least 24 hours in acetone solution, metal components were removed using tongs, since the standard nitrile gloves used in the lab are not safe to use with this chemical. They were not immediately rinsed, since the acetone softens the PMMA but this effect does not remain long once the components begin to dry. A combination of several sizes of uncoated kitchen knives and flathead screwdrivers were

used to pry the PMMA from the metal surfaces, taking care not to scratch the surfaces. It was occasionally necessary to place the metallic components back in the acetone solution if part of the PMMA remained hard or difficult to remove. The parts were allowed to soak for another 24 hours in this case. Once all PMMA was removed, components were again soaked in 10% bleach solution for 30 minutes and rinsed with DI water. At this point, all components were considered completely clean and ready for inspection.

As previously described, wear was quantified using the method proposed by Hood et al in 1983. After the cleaning process, the polyethylene tibial component was placed beneath transparent slide sheet and the outline was traced with a permanent marker. The primary observer (Erik Woodard) then drew lines inside the outline to divide the surface of the implant into 10 distinct regions (**Figure 3-12**). This was done to establish consistency between multiple observers, and the template was used for all future wear scores. The template followed the ten section division used by Hood et al for tibial inserts.

The implant was then placed beneath a light microscope with a 10X magnification and moved such that one sector of interest was clearly visible. The microscope had a 10 megapixel camera with adjustable focus to display images of the implant on a computer screen for ease of visualization. It was occasionally necessary to tilt the implant so the plane of the implant was perpendicular to the line of sight of the microscope. This corrected for glare and obstruction of surface features caused by the concave shape of the implants.

Each section was graded on a scale of 0 to 3 for seven different modes of surface degradation as explained by Hood:

- Surface deformation: any permanent deformation on the surface of the insert.

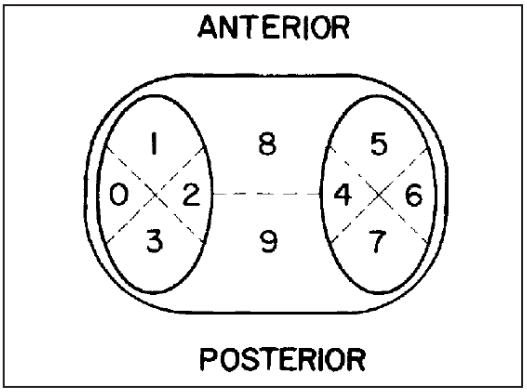

Figure 3-12: Technique used to divide the surface of a tibial insert into 10 sections to facilitate damage scoring. Source: Modified with permission. Hood, R.W., T.M. Wright, and A.H. Burstein, *Retrieval analysis of total knee prostheses: a method and its application to 48 total condylar prostheses.* J Biomed Mater Res, 1983. 17(5): p. 829-42 [29].

This is typically caused by cold flow or creep of the polyethylene.

- Pitting: depressions in the articulating surface. Embedded PMMA debris: change in color or texture of the articulating surface due to the deposit of cement used to secure TKA components
- Scratching: long lines occurring in an anteroposterior direction.
- Burnishing: highly polished areas on the articulating surface.
- Abrasion: areas with a shredded or tufted appearance caused by contact with bone or cement
- Delamination: removal of sheets of polyethylene

These modes were based on collections of observations made previously by various researchers and compiled by Hood and his team. A score of zero was used to indicate that the damage mode was not present on that section of the insert, while 1 denoted damage covering less than 10% of the section, 2 denoted damage covering 10-50%, and a score of 3 meant that damage was present on over 50% of the section, as expressed in **Table 3-1**.

These scores were recorded on an Excel spreadsheet for each specimen and all scores were added together to obtain a total wear score for the specimen. Scores were also separated into medial and lateral designations: sections 0-3 were lateral and 4-7 were medial for an implant taken from a right knee, and sections 0-3 were medial and 4-7 were lateral for a left knee implant. Medial and lateral scores were totaled separately and used to compare with VV laxity data. All other laxity data was compared to total average wear scores for each implant.

Significant correlations were determined using a linear least-squares best fit method, also known as a linear regression. This is a built-in function in Excel which uses a linear equation which minimizes the error of the line from all plotted points. The fit of this line, which describes how well the two data sets are correlated, is expressed as a coefficient of determination, displayed as an r squared value in Excel. This value indicates how well data points fit a statistical model. It is useful to compare this value to a critical r squared value at a given level of confidence (90%, 95%, etc.). To do this, the critical t value with a given number of samples must first be known. To determine

Table 3-1: Summary of the wear score quantification proposed by Hood et al.

Score	Area of sector affected by damage
0	No portion of the area affected
1	Damage present on less than 10% of the surface
2	Damage present on 10-50% of the surface
3	Damage present on more than 50% of the surface

Source: Hood, R.W., T.M. Wright, and A.H. Burstein, *Retrieval analysis of total knee prostheses: a method and its application to 48 total condylar prostheses.* J Biomed Mater Res, 1983. **17**(5): p. 829-42. [29]

significance with a given number of degrees of freedom, or data points, **Equation 3-2** can be used:

$$t = r\sqrt{\frac{n-2}{1-r^2}} \qquad \textbf{(Eq. 3-2)}$$

where n is the number of samples and t is the critical t value at a selected confidence interval with this many samples. Using these values, a critical value for r squared can be determined. If the coefficient of determination computed using linear regression is greater than this critical value, then the null hyposthesis that there is no correlation between the two data sets can be rejected.

Further statistical analysis was undertaken to compare laxity and wear data using the Spearman rank correlation coefficient. First, the ranks of each data point in the two data sets are calculated. The difference in the ranks of data points between data sets is then used to compute the correlation coefficient in **Equation 3-3**:

$$r_s = 1 - \frac{6\Sigma d^2}{n^3 - n} \qquad \textbf{(Eq. 3-3)}$$

where d is the rank differences, which are squared then summed, and n is the number of data points in each set. A coefficient of +1 indicates a perfect positive correlation, and -1 indicates a perfect negative correlation. A critical value is determined based on the desired confidence level and the number of data points.

CHAPTER 4. RESULTS

The total average wear score for all 20 retrieved implants was 19.7±5.6. The highest wear score for any single component was 33. Spearman's correlation coefficient between the two observers was 0.487, with a critical value of 0.45 for a two-tailed test with 18 degrees of freedom at the 95% confidence level. The most obvious modes of surface degradation were delamination and oxidation. Oxidation caused a yellow coloring on the surface of the insert that was always accompanied by some amount of delamination, as seen in the retrieved specimen in **Figure 4-1**. Thirteen of the retrieved implants were cruciate retaining designs, and 7 implants were posterior stabilized designs with a post on the tibial insert. Average wear scores were not significantly different between groups (PS = 19.6±5.0 and CR = 19.7±6.3).

Specimens exhibited large variations in laxity. Average posterior laxity was 1.0° less than anterior laxity in extension, but was 0.98, 4.4, and 1.4 degrees greater than anterior laxity at 30, 60, and 90 degrees of flexion, respectively. The largest deviation from the mean was seen in posterior laxity at 60 degrees of flexion, with a standard error of the mean (SEM) of 0.94°. External rotational laxity was 1.2, 2.8, and 2.2 degrees greater than internal laxity at extension and 30 and 60 degrees of flexion, but decreased by 2.9 degrees compared to internal laxity at 90 degrees of flexion. Varus laxity was greater than valgus for all flexion angles. A summary of this average laxity data is shown in **Table 4-1**, **Table 4-2**, and **Table 4-3**, and additional average laxity graphs can be seen in Appendix A.

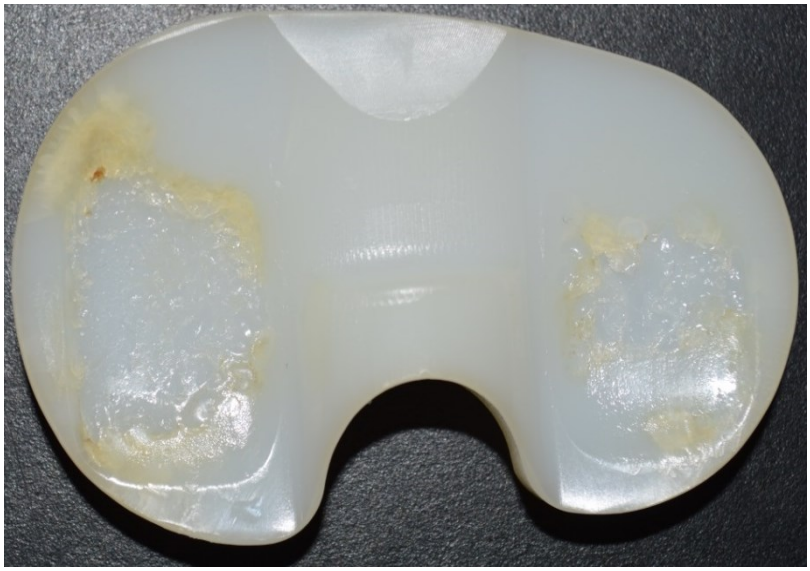

Figure 4-1: Image of an assymetric tibial insert exhibiting oxidation and delamination. This implant had the second highest wear score, which in this case was 30.

Table 4-1: Combined total average AP laxity (degrees).

Data Type	Anterior				Posterior			
Flexion Angle	0	30	60	90	0	30	60	90
Average	2.90	6.04	3.56	2.47	-1.89	-7.02	-7.97	-3.83
SEM	0.41	0.68	0.46	0.43	0.25	0.88	0.94	0.85

Average AP laxity for 16 tested specimens. SEM = standard error of the mean.

Table 4-2: Combined total average IE laxity (degrees).

Data Type	Internal				External			
Flexion Angle	0	30	60	90	0	30	60	90
Average	-5.48	-13.09	-14.28	-14.57	6.66	15.91	16.49	11.67
SEM	0.83	1.88	1.68	1.57	0.98	2.00	1.98	1.63

Average IE laxity for 20 tested specimens. SEM = standard error of the mean.

Table 4-3: Combined total average VV laxity (degrees).

Data Type	Varus				Valgus			
Flexion Angle	0	30	60	90	0	30	60	90
Average	-3.07	-6.40	-8.03	-8.48	2.80	5.07	5.78	5.90
SEM	0.37	0.86	1.10	1.65	0.51	0.85	1.19	1.27

Average VV laxity for 20 tested specimens. SEM = standard error of the mean.

Laxity and Wear Correlation

The following series of charts show the correlations between three different laxity tests and wear scores. In all cases, laxity was considered the independent variable and wear was the dependent variable. All laxity data was calculated using the Matlab code in Appendix B.

Anterior/Posterior Laxity

The following two graphs depict the relationship between anterior and posterior laxity of the tibia and total wear scores for each tested specimen. Both graphs contain laxity and wear data for 16 tested specimens. **Figure 4-2** shows anterior laxity, measured in millimeters of displacement, compared to wear scores. Posterior laxity vs wear scores are shown in **Figure 4-3**. Anterior laxity shows a significant linear correlation to wear scores in extension at the 90% confidence level, with an r-squared value of 0.124. Posterior laxity shows a linear relationship to wear scores at 30° flexion at the 90% confidence level with an r-squared of 0.14. An r-squared value of 0.12 was needed for significance at the 90% confidence level with this sample size. Significant linear correlations were not observed for any other flexion angles when comparing AP laxity data.

Internal/External Laxity

The next two graphs depict the relationship between rotational laxity and total wear scores for each TKA specimen. Equations represent a best fit linear line to the data. Internal laxity is compared to wear scores in **Figure 4-4**, and external laxity and wear scores are shown in **Figure 4-5**. None of the r-squared values indicate a significant linear correlation at the 95 or 90% confidence levels, with an r-squared value of 0.16 considered significant at the 95% confidence level.

Varus/Valgus Laxity

The next four graphs show the relationship between varus and valgus laxity and wear in both the medial and lateral compartments of the retrieved tibial inserts. Each compartment consisted of four sections using the Hood method of dividing each condyle into sections. Varus laxity is compared independently to the medial compartment at each tested flexion angle (**Figure 4-6**), and then to the lateral compartment (**Figure 4-7**). Likewise, valgus laxity is compared to the lateral compartment (**Figure 4-8**), and to the medial side (**Figure 4-9**). None of the r-squared values indicate a significant correlation at the 95 or 90% confidence intervals, with an r-squared value of 0.16 considered significant at the 95% confidence level..

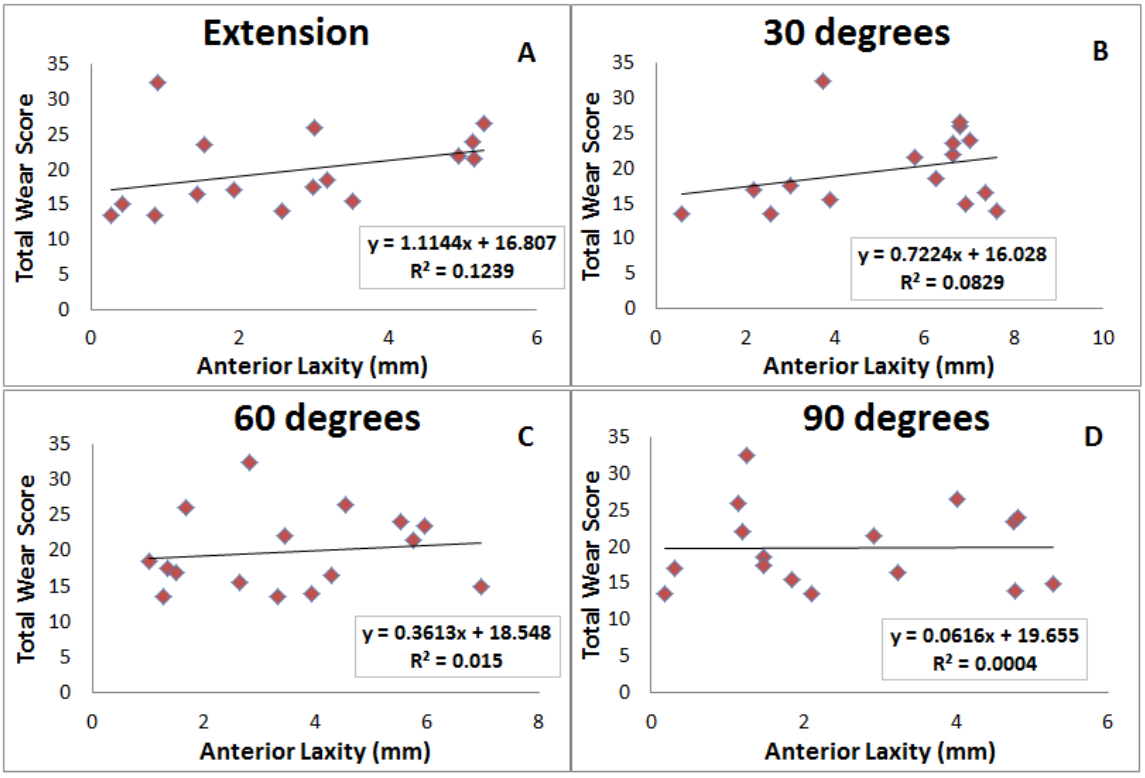

Figure 4-2: Anterior displacement of the tibia related to the femur when subjected to a 35N force compared to total wear score for 16 specimens. Graphs show laxity in full extension (A), 30 degrees of flexion (B), 60 degrees of flexion (C), and 90 degrees of flexion (D).

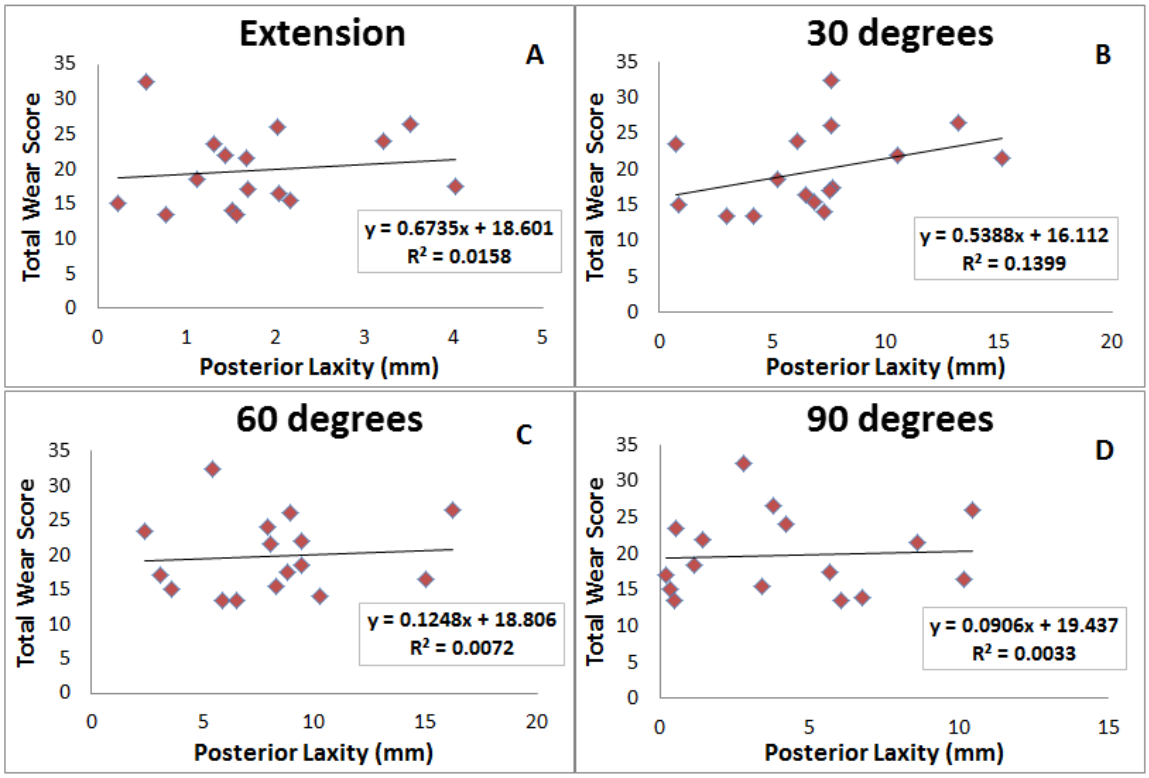

Figure 4-3: Posterior displacement of the tibia related to the femur when subjected to a 35N force compared to total wear score for 16 specimens. Graphs show laxity in full extension (A), 30 degrees of flexion (B), 60 degrees of flexion (C), and 90 degrees of flexion.

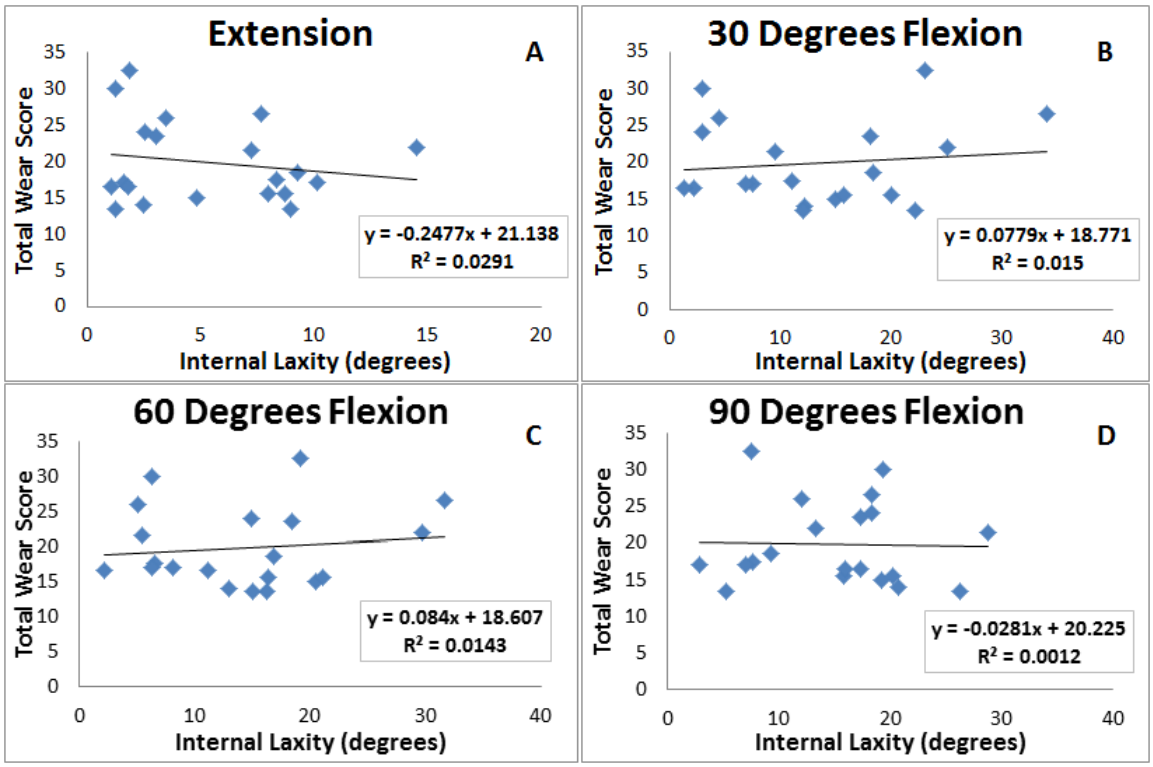

Figure 4-4: Internal rotation of the tibia related to the femur when subjected to a 1.5Nm torque compared to total wear score for 20 specimens. Graphs show laxity in full extension (A), 30 degrees of flexion (B), 60 degrees of flexion (C), and 90 degrees of flexion.

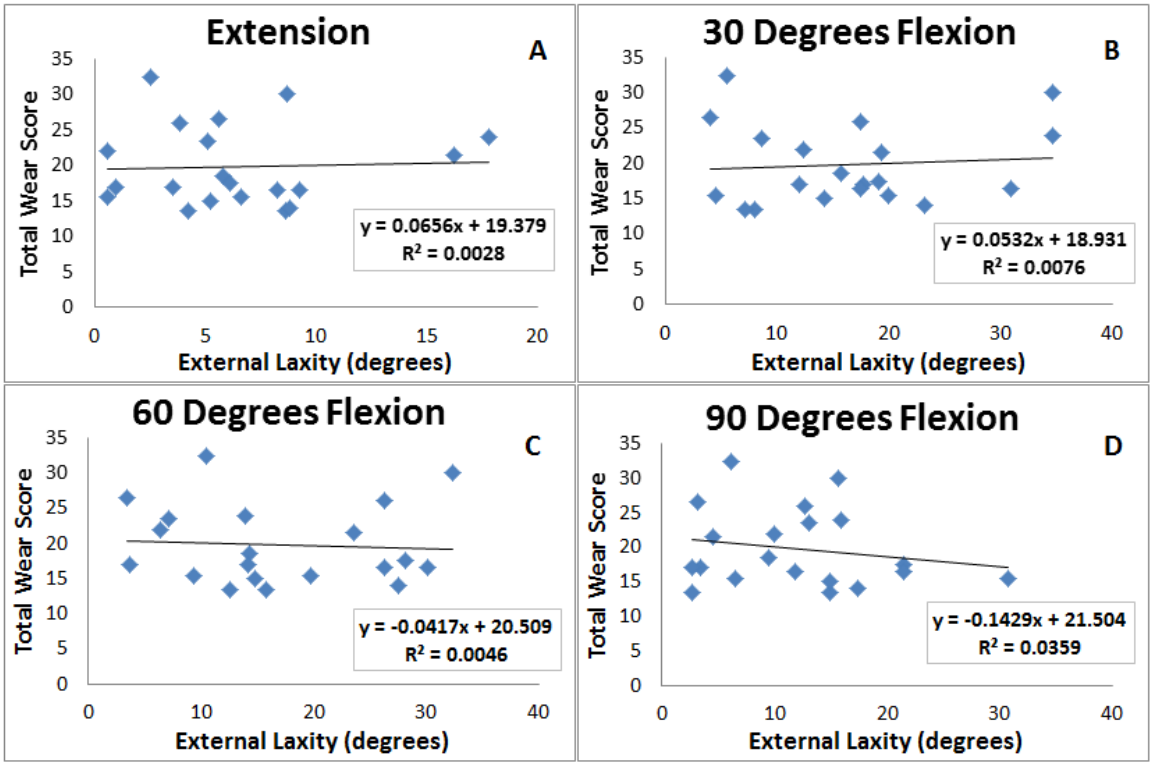

Figure 4-5: External rotation of the tibia related to the femur when subjected to a 1.5Nm torque compared to total wear score for 20 specimens. Graphs show laxity in full extension (A), 30 degrees of flexion (B), 60 degrees of flexion (C), and 90 degrees of flexion.

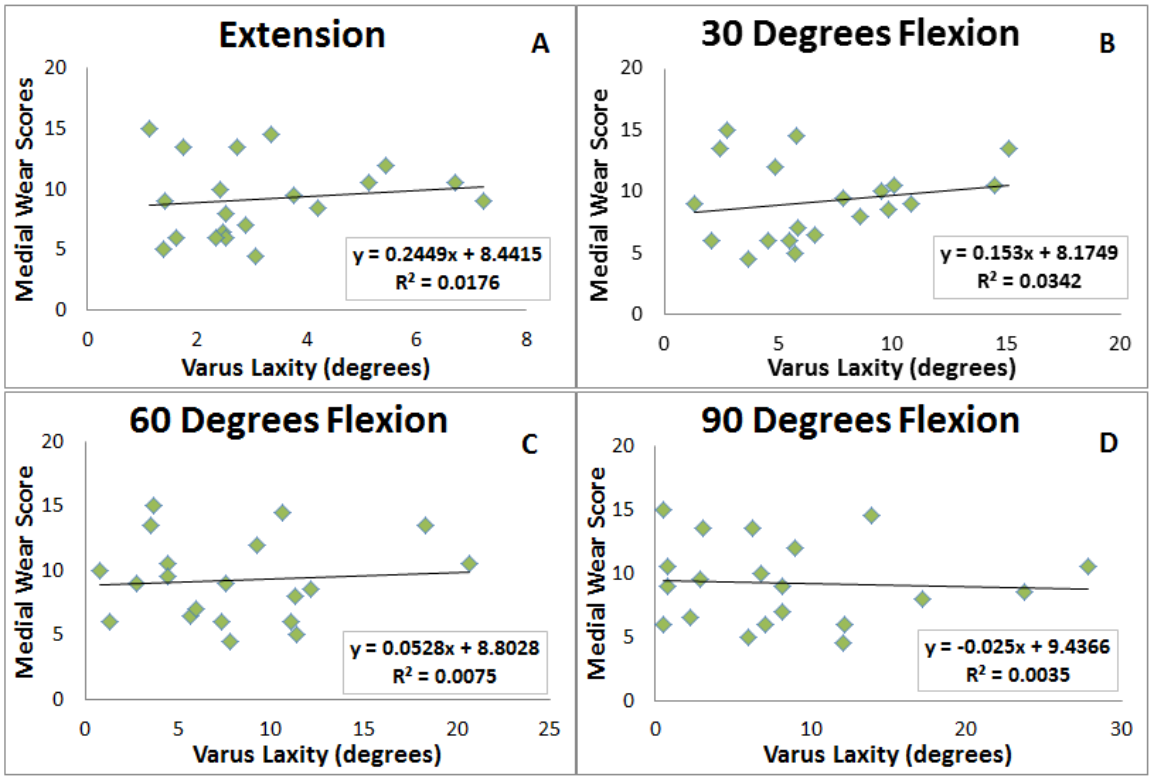

Figure 4-6: Varus rotation of the tibia in the coronal plane related to the femur when subjected to a 10Nm torque compared to medial compartment wear score for 20 specimens. Graphs show laxity in full extension (A), 30 degrees of flexion (B), 60 degrees of flexion (C), and 90 degrees of flexion.

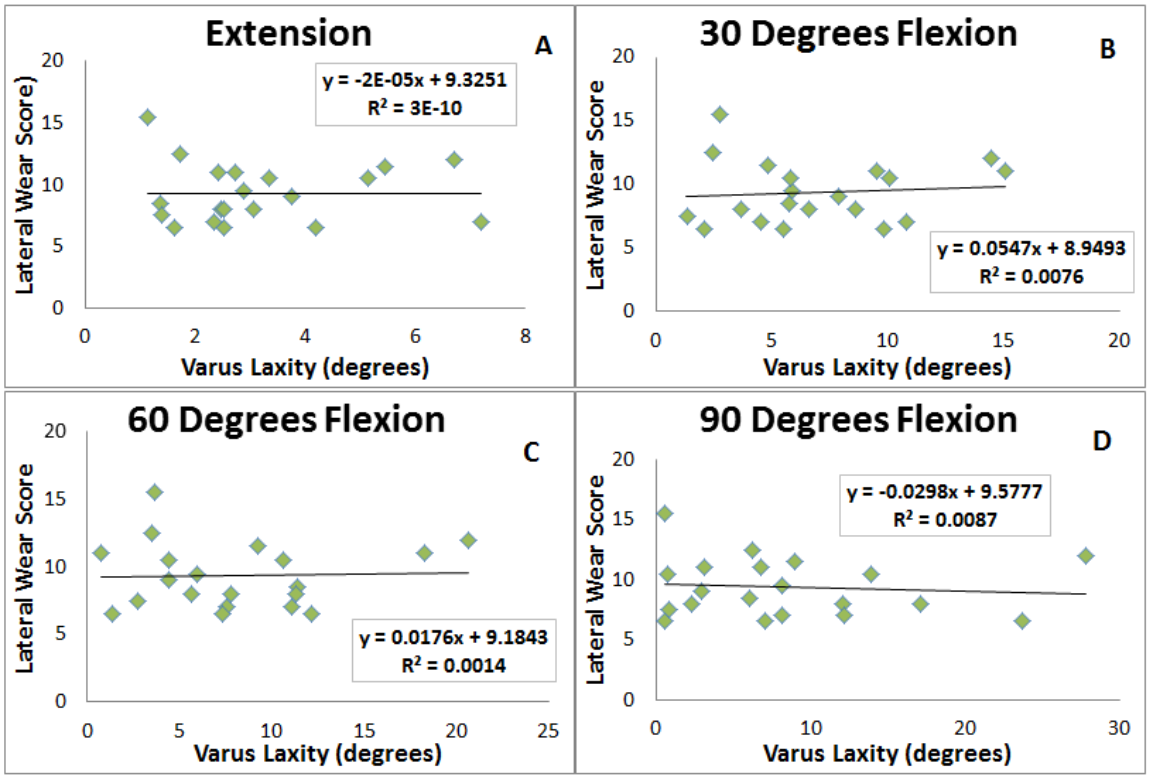

Figure 4-7: Varus rotation of the tibia in the coronal plane related to the femur when subjected to a 10Nm torque compared to lateral compartment wear score for 20 specimens. Graphs show laxity in full extension (A), 30 degrees of flexion (B), 60 degrees of flexion.

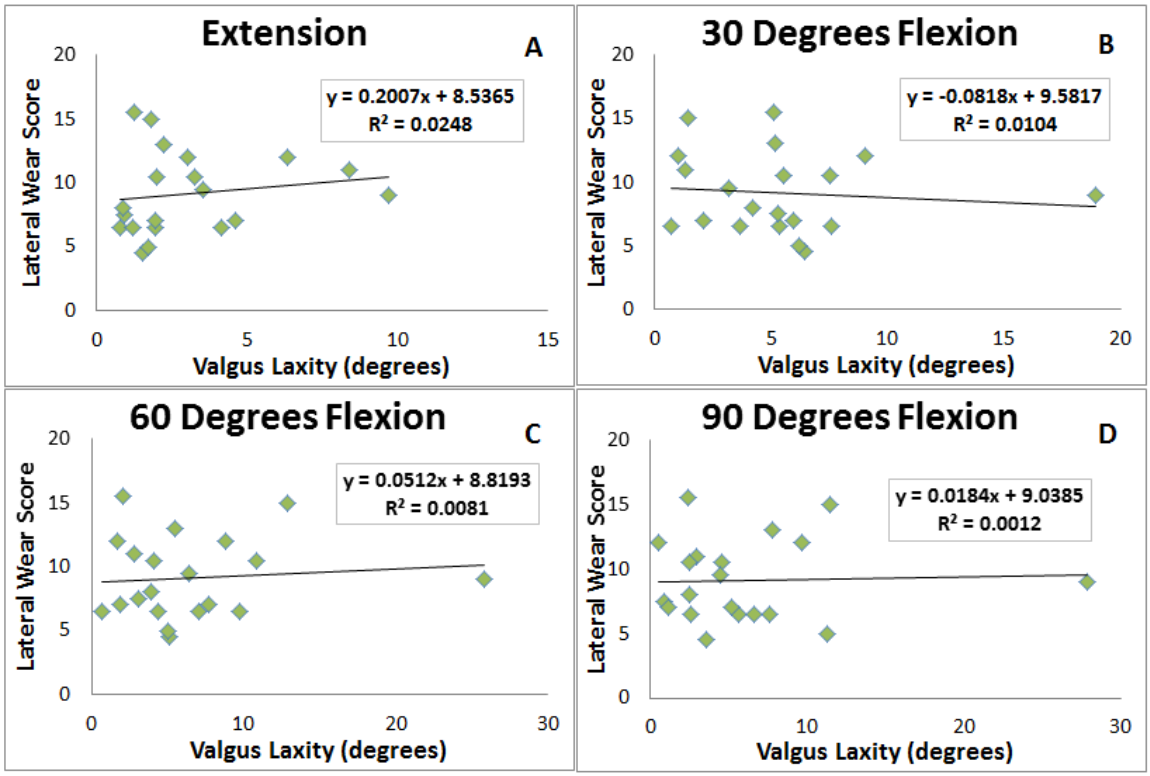

Figure 4-8: Valgus rotation of the tibia in the coronal plane related to the femur when subjected to a 10Nm torque compared to lateral compartment wear score for 20 specimens. Graphs show laxity in full extension (A), 30 degrees of flexion (B), 60 degrees of flexion.

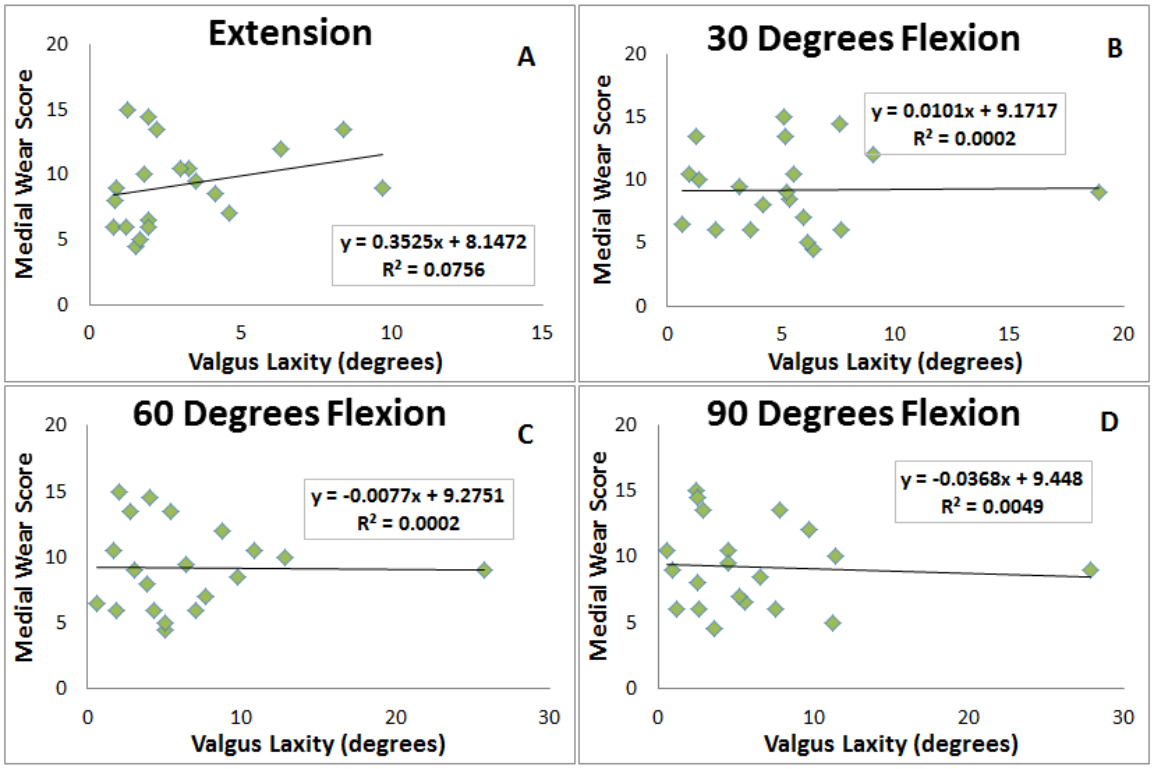

Figure 4-9: Valgus rotation of the tibia in the coronal plane related to the femur when subjected to a 10Nm torque compared to medial compartment wears score for 20 specimens. Graphs show laxity in full extension (A), 30 degrees of flexion (B), 60 degrees of flexion.

Alignment and Wear Correlation

The next two figures show the relationship between rotational alignment of the femoral and tibial components and total wear scores for each retrieved insert. The first graph (**Figure 4-10**) shows values of rotational mismatch calculated by measuring femoral and tibial component rotation separately using anatomic landmarks, then using the difference of these measurements as mismatch. The second graph (**Figure 4-11**) depicts the relationship between congruency mismatch and total wear scores. Congruency mismatch is the angle between the posterior condylar axis of each component, measured without anatomic landmarks. No significant correlation was found for either relationship.

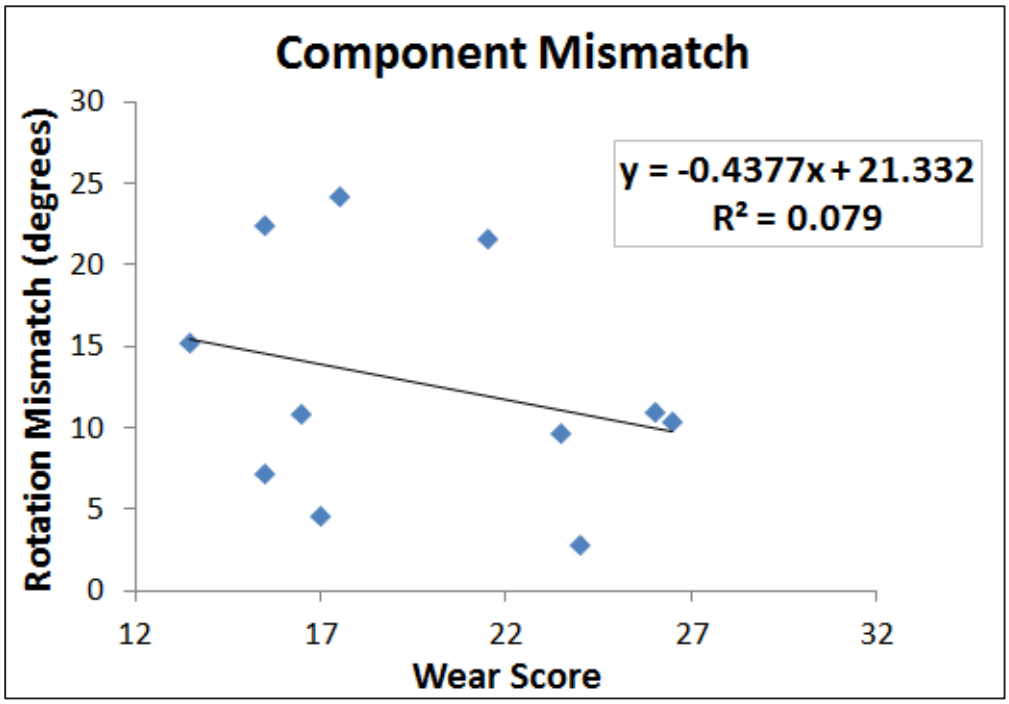

Figure 4-10: Relationship between component rotation mismatch and total wear scores.

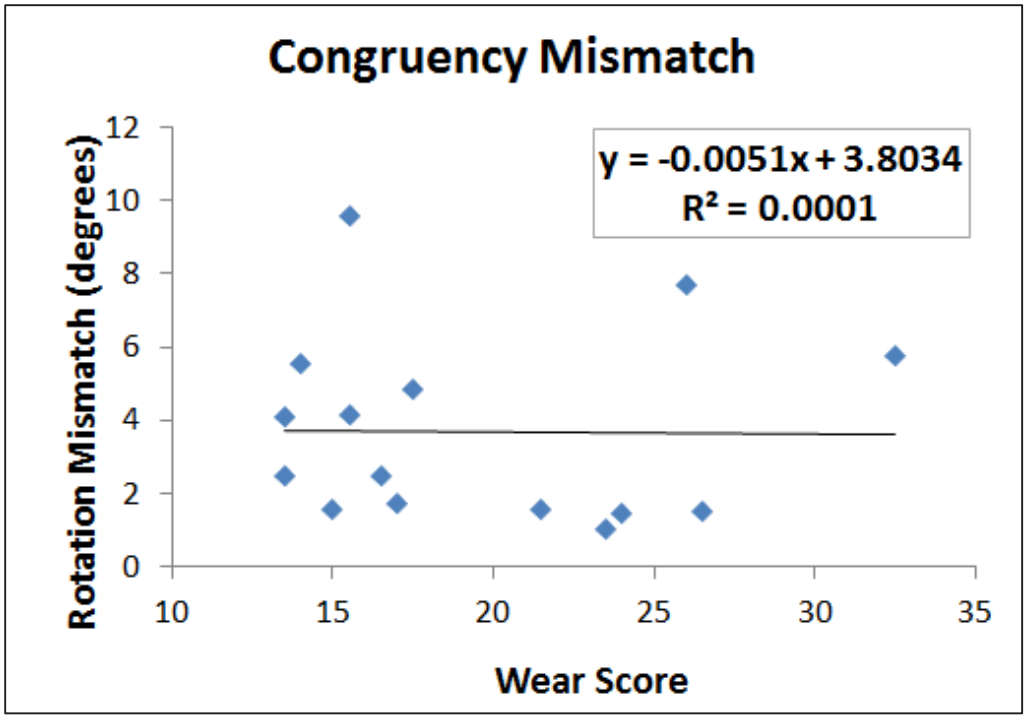

Figure 4-11: Relationship between congruency mismatch and total wear scores.

CHAPTER 5. DISCUSSION

Many failure modes of TKA have previously been reported, including polyethylene wear and instability [6, 36, 37]. Berend et al reported that the most common failure mode in a sample of 41 patients with revision TKA was due to medial bone collapse. Within this group, it was determined that a varus alignment of the tibial component of more than 3.0° greatly increased the chances of failure. The second most common cause of revision was ligament imbalance. Several studies have concluded that varus alignment fo the tibial component, particularly varus angles greater than 3°, greatly increased the chances of long-term TKA failure [36, 38, 39]. This may cause to increased tibial edge loading and osteolytic lesions in the medial compartment, which can lead to medial collapse and loosening of the tibial baseplate.

Flat-on-flat cruciate retaining designs may provide an increased range of motion postoperatively, but they have also been shown to have higher wear rates compared to highly conforming articular designs [40]. This is likely due to excessive loading of the polyethylene, since decreasing conformity leads to less stability. Contact stress also increases in implants with a varus bias. A combination of instability and varus alignment is therefore a good indication of failure in TKA.

TKA wear rates are commonly estimated using in vitro simulations which utilize a testing apparatus to apply approximated physiologic loads over many cycles. Bovine serum is typically used to mimic synovial fluid action. However, these simulations have been shown to consistently undershoot wear rates because they are forced to reproduce only one activity – typically a normal gait pattern, and do not replicate other activities such as deep flexion [41]. An advantage of this study is the inclusion of patients with successful implants, as opposed to studies which examine revision components that have failed in vivo.

In contrast to previous studies, varus and valgus alignment was not examined here. Since there is already an extablished relationship between excessive varus alignment and TKA revision due to early failure, rotational alignment was substituted for coronal alignment. This coronal malignment was expressed two ways: using component mismatch and congruency mismatch. No correlation was found between wear and component or congruency mismatch. Transverse component alignment is only one of many factors which may lead to wear, so these results do not necessarily exclude alignment as a contributing factor to wear. The results simply indicate that transverse alignment is not *directly* correlated to wear of the articulating surface.

The angle formed by the transepicondylar axis and the femoral posterior condylar axis has been shown to relate to the degree of pre-operative coronal deformity [17] (**Figure 5-1**). In contrast, external rotation of the PCA from the STEA showed a positive correlation to increased varus neutral angle at extension in this study (**Figure 5-2**). However, no significant correlation was observed between coronal angle and PCA rotation at 90° of flexion. This could be due to corrections of varus or valgus deformities

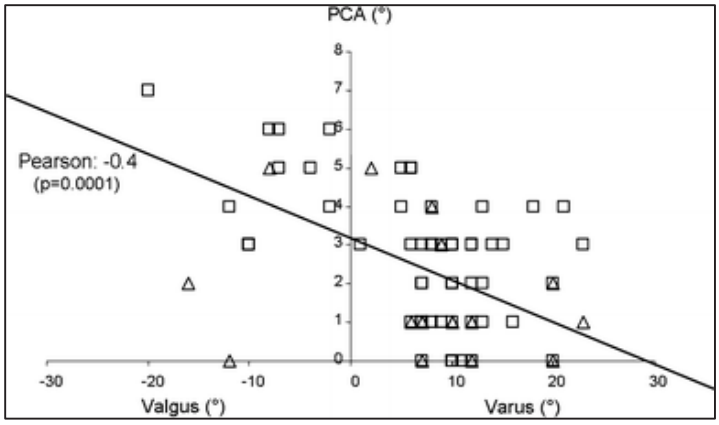

Figure 5-1: Relationship between the posterior condylar angle (PCA) and the degree of pre-operative coronal deformity, expressed as varus or valgus angle.
Source: Reprinted with permission. Aglietti, P., et al., *Rotational position of femoral and tibial components in TKA using the femoral transepicondylar axis.* Clin Orthop Relat Res, 2008. **466**(11): p. 2751-5 [17].

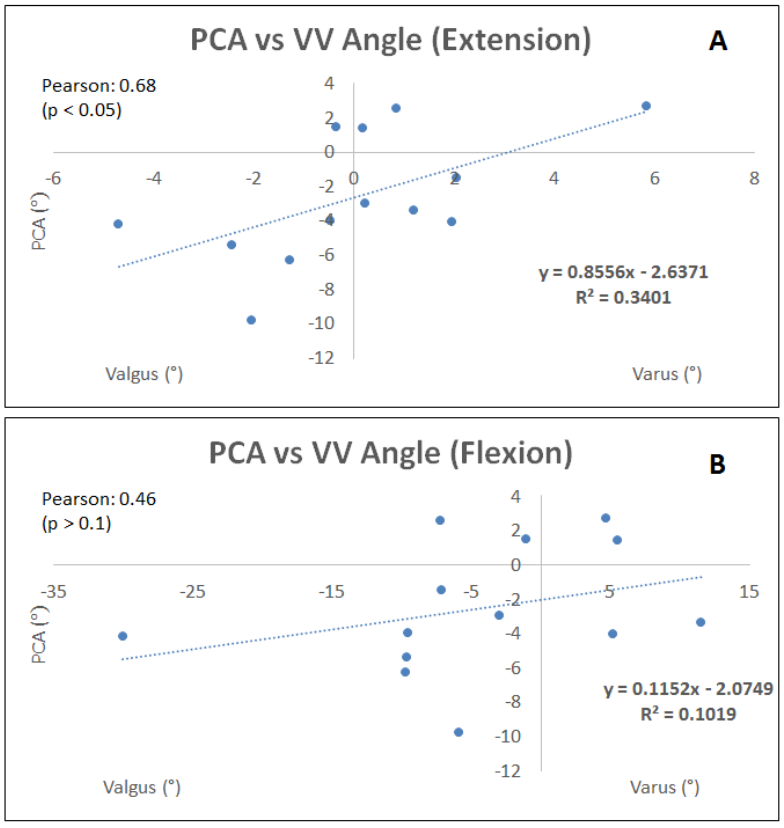

Figure 5-2: Neutral coronal angle at (A) extension and (B) 90 degrees flexion compared to rotation of the femoral PCA from the STEA. Sign conventions are the same as those used by Berger et al [12].

which were corrected at the time of surgery, and which influenced this data postoperatively. Rotational alignment of the tibial component would also affect these findings, as well as the joint contact force of 30N used in this study. However, this discrepancy between preoperative and postoperative coronal rotation should still be noted.

The results indicate a positive correlation between wear scores and increased anterior laxity in extension, as well as posterior laxity at 30 degrees of flexion. This indicates that an excessive anterior or posterior sliding may be a significant cause of wear in implants. The flexion angles at which this wear is significant (extension and 30°) correspond to flexion angles during common gait. If the most time is spent at these flexion angles, it makes sense that this is the primary location where wear occurs, and increased laxity at these flexion angles could cause more wear during common activities.

Tibial insert wear does not exhibit a correlation to rotational or varus and valgus laxity in these specimens. However, it is commonly known that stability in the transverse and coronal planes is essential to successful patient outcomes. The effect of surgical techniques on laxity in cadaveric specimens after TKA surgery has been assessed previously using the same custom machine [42-44]; however, the laxity conditions of the normal specimens have never been compared to alignment or wear data. Using a linear correlation to determine the relationship between wear scores and laxity can potentially be problematic since wear scores are not linear in nature – a score of 2 does not mean twice as much damage as a score of 1. These scores correspond to 10% and 10-50% damage over the surface area, respectively. For this reason, wear was also related to laxity using Spearman's rank correlation coefficient, which uses a rank-based comparison and is more appropriate for discrete, nonlinear data (**Table 5-1)**. These correlations correspond well to the best-fit linear regression values.

Typically, the post and cam mechanism on a posterior-stabilized TKA design only engages past 60 degrees of flexion. Thus, this mechanism does not play a role in the significant differences observed here. If the post and cam mechanism is not able to mimic the function of the PCL in flexion, this could contribute to variations in laxity and wear compared to cruciate retaining designs at flexion angles greater than 60°. For this reason, the laxity data at 90° of flexion was split into posterior stabilized and cruciate retaining groups and correlated with total wear scores using Spearman's rank correlation coefficient (**Table 5-2**). These coefficient values do not indicate any significant correlations between laxity and wear, yielding similar results to a linear best fit test. It can be concluded that PS and CR designs do not exhibit large differences in laxity in flexion which lead to differences in wear over time.

The average calculated component mismatch was significantly different from congruency mismatch (17.9±11.8° vs 3.7±2.6°). However, component mismatch was likely skewed by the tibial rotation measurement of specimens 414L and 414R, both of which were mobile bearing TKA designs. This means that the tibial insert is free to rotate with respect to the cemented tibial baseplate in these components. For this reason, these two measurements were not included when determining the correlation between

Table 5-1: Spearman's rank correlation for each testing group described previously.

Flexion Angle	Anterior	Posterior	Internal	External	Varus/Medial	Varus/Lateral	Valgus/Medial	Valgus/Lateral
0	0.54	0.19	-0.01	-0.05	0.20	0.04	0.40	0.30
30	0.21	0.49	0.03	0.08	0.22	0.12	-0.09	-0.21
60	0.18	0.10	0.02	-0.13	-0.08	-0.04	0.01	0.04
90	-0.02	0.11	-0.11	-0.12	-0.11	-0.09	-0.11	-0.06

Significant wear and laxity correlations are the same as those calculated using a linear best-fit approach. Anterior and posterior laxity are positively correlated to wear scores at extension and 30 degrees of flexion, respectively.

Table 5-2: Spearman's rank correlation coefficient at 90° flexion.

Implant Design	Anterior	Posterior	Internal	External	Varus	Valgus
PS	0.66	-0.77	-0.43	-0.57	-0.32	-0.29
CR	-0.18	0.39	-0.01	0.06	-0.00	-0.14

Coefficient values correlating laxity to total wear scores for all specimens. No correlation values are significant at the 90 or 95% confidence level.

component mismatch and wear. In these models, the component may be in a significantly different orientation compared to the tibial tuberosity axis. Furthermore, although these mobile bearing designs were introduced to reduce loading and wear on the polyethylene, they have not been shown to offer any advantages over fixed bearing designs with respect to clinical outcomes or decreased wear [33].

CHAPTER 6. CLINICAL SIGNIFICANCE

The results of this study could be used foremost to assist engineers when designing new TKA components before they are placed. It appears that anterior and posterior stability at extension through 30 degrees of flexion are crucial factors to consider when manufacturing and implant. This is not to downplay the importance of rotational or coronal stability; these factors simply may have as great a contribution to wear over time.

Additionally, surgeons must also bear in mind the same factors as the engineers who design the implants. They also have other considerations such as alignment. This data suggests that transverse plane alignments of the femoral and tibial components are not always in agreement. Robotic assisted surgery has been shown to provide more consistent alignment than manual technique. This may be able to reduce the large variability in component alignment seen here if more surgeons adopt this during TKA. Surgeons should strive to work with engineers to develop consistent methods for component alignment.

CHAPTER 7. LIMITATIONS AND CONCLUSIONS

Limitations

Numerous limitations exist for this study. Possibly the most impactful is the subjectivity of the wear score technique. Although wear scoring has been shown to be highly repeatable, it is still based on the expertise of each individual observer. Discrepancies between observes can lead to error in reported wear values. In this study, we attempted to minimize error by averaging wear scores of two observers. Some argue that only scores from one observer should be used; however, this data uses average scores of two expert observers, which should provide a good representation of wear. If full volumetric data from the original implants and μCT scans were available, expressing wear as volumetric loss and replacing wear scores with this more objective method might yield different results.

Another limitation is the variability in "normal" laxity of specimens due to the many different types of implants. Among the 20 implants we encountered posterior stabilized and cruciate retaining designs from a variety of manufacturers. However, wear in each implant should still be affected by the two surgical parameters mentioned here, regardless of implant type. A larger sample size would be useful to correct for this limitation. Surgical technique is another factor which could cause variations in normal laxity.

Since no TKA specimen had previously required revision, each specimen was assumed to be correctly balanced at the time of surgery. However, it is not known how bone cuts and ligament balancing were performed at the time of surgery. Bone cuts may have been performed manually or with robotic assisted surgical tools to attempt to increase their accuracy. This introduces a further variation into the study. Coronal deformities are corrected at the time of surgery; however, it is unknown how ligaments adapted after loosening due to release to correct imbalances.

The full medical history of these donated bodies is not known, so detailed individual parameters are not taken into account. Since patient-specific outcomes are not reported, it is necessary to assume that these specimens were properly balanced and had minimal postoperative deformities.

Conclusions

Traditionally component alignment has been a fairly subjective technique which relied on the expertise of the surgeon. Recently, the introduction of computerized surgical assistance and precision engineering cutting blocks has reduced the error associated with component placement. Determining component alignment using CT scans is a technique which has been utilized many times over the last 2 decades, but a

consistent method of expressing rotation of the tibial component relative to the femoral component is still in question.

This study is unprecedented in that it combines several techniques which have previously only been utilized separately. It also presents a novel way to express transverse alignment of the tibial component relative to the femoral component, referred to here as congruency mismatch. As shown, traditional measurements of alignment may not be appropriate to express component rotations. These techniques may need to be modified with the advent of computer-assisted surgery and mobile bearing implants.

Future work would involve replacing the subjective wear scores with volumetric loss of the tibial inserts calculated using microCT scans of the components. For this study, this was not possible due to a lack of access to the original inserts. By examining the results of this study, engineers and surgeons can pinpoint which aspects of TKA design and surgery to target in order to minimize wear, and therefore reduce costly revision surgeries.

LIST OF REFERENCES

1. Kurtz, S., et al., *Projections of primary and revision hip and knee arthroplasty in the United States from 2005 to 2030.* J Bone Joint Surg Am, 2007. **89**(4): p. 780-5.
2. Kurosaka, M., et al., *Maximizing flexion after total knee arthroplasty: the need and the pitfalls.* J Arthroplasty, 2002. **17**(4 Suppl 1): p. 59-62.
3. Argenson, J.N., et al., *In vivo kinematic evaluation and design considerations related to high flexion in total knee arthroplasty.* J Biomech, 2005. **38**(2): p. 277-84.
4. *http://en.wikipedia.org/wiki/Knee#mediaviewer/File:Knee_medial_view.gif*. May 1, 2001. [cited 2014 February 19].
5. Nakayama, K., et al., *Contact stress at the post-cam mechanism in posterior-stabilised total knee arthroplasty.* J Bone Joint Surg Br, 2005. **87**(4): p. 483-8.
6. Sharkey, P.F., et al., *Insall Award paper. Why are total knee arthroplasties failing today?* Clin Orthop Relat Res, 2002(404): p. 7-13.
7. *http://en.wikipedia.org/wiki/Anatomical_plane#mediaviewer/File:Human_anatomy_planes.svg*. June 7, 2008. [cited 2014 June 9].
8. *http://en.wikipedia.org/wiki/File:Gray257.png*. 2003 [cited 2014 April 4].
9. Krackow, K.A. and W.M. Mihalko, *Flexion-extension joint gap changes after lateral structure release for valgus deformity correction in total knee arthroplasty: a cadaveric study.* J Arthroplasty, 1999. **14**(8): p. 994-1004.
10. Meftah, M., et al., *Correcting fixed varus deformity with flexion contracture during total knee arthroplasty: the "inside-out" technique: AAOS exhibit selection.* J Bone Joint Surg Am, 2012. **94**(10): p. e66.
11. Bathis, H., et al., *Alignment in total knee arthroplasty. A comparison of computer-assisted surgery with the conventional technique.* J Bone Joint Surg Br, 2004. **86**(5): p. 682-7.
12. Berger, R.A. and L.S. Crossett, *Determining the Rotation of the Femoral and Tibial Components in Total Knee Arthroplasty: a Computer Tomography Technique.* Operative Techniques in Orthopaedics, 1998. **8**(3): p. 128-133.
13. Berger, R.A., et al., *Determining the rotational alignment of the femoral component in total knee arthroplasty using the epicondylar axis.* Clin Orthop Relat Res, 1993(286): p. 40-7.
14. Jazrawi, L.M., et al., *The accuracy of computed tomography for determining femoral and tibial total knee arthroplasty component rotation.* J Arthroplasty, 2000. **15**(6): p. 761-6.
15. Winemaker, M.J., *Perfect balance in total knee arthroplasty: the elusive compromise.* J Arthroplasty, 2002. **17**(1): p. 2-10.
16. Nicoll, D. and D.I. Rowley, *Internal rotational error of the tibial component is a major cause of pain after total knee replacement.* J Bone Joint Surg Br, 2010. **92**(9): p. 1238-44.
17. Aglietti, P., et al., *Rotational position of femoral and tibial components in TKA using the femoral transepicondylar axis.* Clin Orthop Relat Res, 2008. **466**(11): p. 2751-5.

18. Eckhoff, D.G., R.G. Metzger, and M.V. Vandewalle, *Malrotation associated with implant alignment technique in total knee arthroplasty.* Clin Orthop Relat Res, 1995(321): p. 28-31.
19. Bell, S.W., et al., *Component rotational alignment in unexplained painful primary total knee arthroplasty.* Knee, 2014. **21**(1): p. 272-7.
20. Mihalko, W.M. and J.L. Williams, *Computer modeling to predict effects of implant malpositioning during TKA.* Orthopedics, 2010. **33**(10 Suppl): p. 71-5.
21. Nagamine, R., et al., *Effect of rotational malposition of the femoral component on knee stability kinematics after total knee arthroplasty.* J Arthroplasty, 1995. **10**(3): p. 265-70.
22. Nagamine, R., et al., *Effect of medial displacement of the tibial tubercle on patellar position after rotational malposition of the femoral component in total knee arthroplasty.* J Arthroplasty, 1996. **11**(1): p. 104-10.
23. Mihalko, W.M., L.A. Whiteside, and K.A. Krackow, *Comparison of ligament-balancing techniques during total knee arthroplasty.* J Bone Joint Surg Am, 2003. **85-A Suppl 4**: p. 132-5.
24. Kretzer, J.P., et al., *Effect of joint laxity on polyethylene wear in total knee replacement.* J Biomech, 2010. **43**(6): p. 1092-6.
25. Harman, M.K., et al., *Comparison of polyethylene tibial insert damage from in vivo function and in vitro wear simulation.* J Orthop Res, 2009. **27**(4): p. 540-8.
26. Bell, C.J., et al., *Effect of oxidation on delamination of ultrahigh-molecular-weight polyethylene tibial components.* J Arthroplasty, 1998. **13**(3): p. 280-90.
27. White, S.E., et al., *Effects of sterilization on wear in total knee arthroplasty.* Clin Orthop Relat Res, 1996(331): p. 164-71.
28. Woodard, E., *Total Knee Arthroplasty Retrieval and Analysis Methods*, in *Retrieval Analysis of Orthopedic Devices*, W.M.M.a.S.S. Kurtz, Editor. 2014, Springer.
29. Hood, R.W., T.M. Wright, and A.H. Burstein, *Retrieval analysis of total knee prostheses: a method and its application to 48 total condylar prostheses.* J Biomed Mater Res, 1983. **17**(5): p. 829-42.
30. Wasielewski, R.C., et al., *Tibial insert undersurface as a contributing source of polyethylene wear debris.* Clin Orthop Relat Res, 1997(345): p. 53-9.
31. Brandt, J.M., et al., *Retrieval analysis of modular total knee replacements: factors influencing backside surface damage.* Knee, 2012. **19**(4): p. 306-15.
32. Engh, G.A., et al., *Analysis of wear in retrieved mobile and fixed bearing knee inserts.* J Arthroplasty, 2009. **24**(6 Suppl): p. 28-32.
33. Kelly, N.H., et al., *Wear damage in mobile-bearing TKA is as severe as that in fixed-bearing TKA.* Clin Orthop Relat Res, 2011. **469**(1): p. 123-30.
34. Teeter, M.G., et al., *In vitro quantification of wear in tibial inserts using microcomputed tomography.* Clin Orthop Relat Res, 2011. **469**(1): p. 107-12.
35. Bowden, A.E., S.M. Kurtz, and A.A. Edidin, *Validation of a micro-CT technique for measuring volumetric wear in retrieved acetabular liners.* J Biomed Mater Res B Appl Biomater, 2005. **75**(1): p. 205-9.
36. Berend, M.E., et al., *Tibial component failure mechanisms in total knee arthroplasty.* Clin Orthop Relat Res, 2004(428): p. 26-34.

37. Furnes, O., et al., *Early failures among 7,174 primary total knee replacements: a follow-up study from the Norwegian Arthroplasty Register 1994-2000.* Acta Orthop Scand, 2002. **73**(2): p. 117-29.
38. Ritter, M.A., et al., *Postoperative alignment of total knee replacement. Its effect on survival.* Clin Orthop Relat Res, 1994(299): p. 153-6.
39. Aglietti, P. and R. Buzzi, *Posteriorly stabilised total-condylar knee replacement. Three to eight years' follow-up of 85 knees.* J Bone Joint Surg Br, 1988. **70**(2): p. 211-6.
40. Feng, E.L., S.D. Stulberg, and R.L. Wixson, *Progressive subluxation and polyethylene wear in total knee replacements with flat articular surfaces.* Clin Orthop Relat Res, 1994(299): p. 60-71.
41. Knight, L.A., et al., *Comparison of long-term numerical and experimental total knee replacement wear during simulated gait loading.* J Biomech, 2007. **40**(7): p. 1550-8.
42. Saeki, K., et al., *Stability after medial collateral ligament release in total knee arthroplasty.* Clin Orthop Relat Res, 2001(392): p. 184-9.
43. Mihalko, W.M., C. Miller, and K.A. Krackow, *Total knee arthroplasty ligament balancing and gap kinematics with posterior cruciate ligament retention and sacrifice.* Am J Orthop (Belle Mead NJ), 2000. **29**(8): p. 610-6.
44. Krackow, K.A. and W.M. Mihalko, *The effect of medial release on flexion and extension gaps in cadaveric knees: implications for soft-tissue balancing in total knee arthroplasty.* Am J Knee Surg, 1999. **12**(4): p. 222-8.

APPENDIX A. ADDITIONAL GRAPHS AND TABLES

Additional laxity and rotation measurements not included in the main text are listed here.

Table A-1: Summary of rotation measurements of femoral and tibial components and wear scores of tibial inserts.

Specimen	Femoral Rotation (degrees)	Tibial Rotation (degrees)	Component Mismatch (degrees)	Congruency Mismatch (degrees)	Average Wear Score
286R	-9.78	-20.63	10.85	2.48	16.5
286L	-4.18	-14.57	10.39	1.51	26.5
383R	-4.02	-19.23	15.21	4.09	13.5
558R	-5.39	-0.87	4.52	1.73	17
726R	2.70	-18.88	21.58	1.58	21.5
531R	-1.50	-12.48	10.98	7.71	26
126L	-4.07	5.55	9.62	1.04	23.5
414L	1.48	-38.46	39.94	1.60	15
414R	2.56	-34.99	37.55	2.51	13.5
721R	1.39	-22.79	24.18	4.84	17.5
721L	-6.30	0.88	7.19	4.15	15.5
586L	-2.97	19.45	22.42	9.57	15.5
586R	-3.37	-0.62	2.75	1.45	24
553L	*	*	*	5.76	32.5
553R	*	*	*	5.55	14
Average	**-2.57**	**-17.17**	**17.85**	**3.70**	**18.9**

The medial sulcus and tibial tuberosity was not clearly visible on specimens 553L and 553R, indicated by the *.

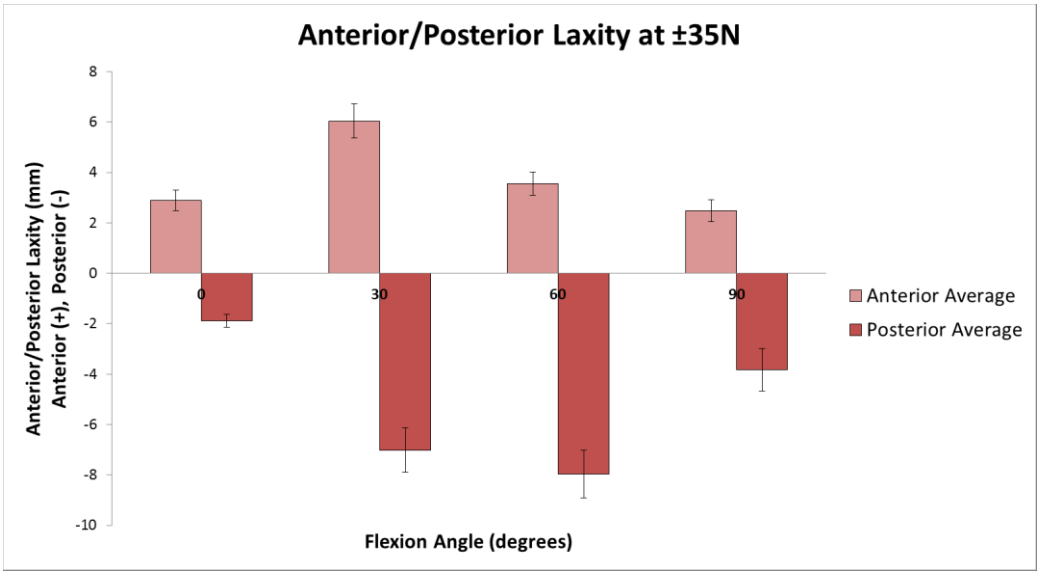

Figure A-1: Average AP laxity of 20 tested TKA knees. Error bars are standard error of the mean (SEM). Positive laxity values are anterior displacement of the tibia in mm, and negative values are posterior displacement.

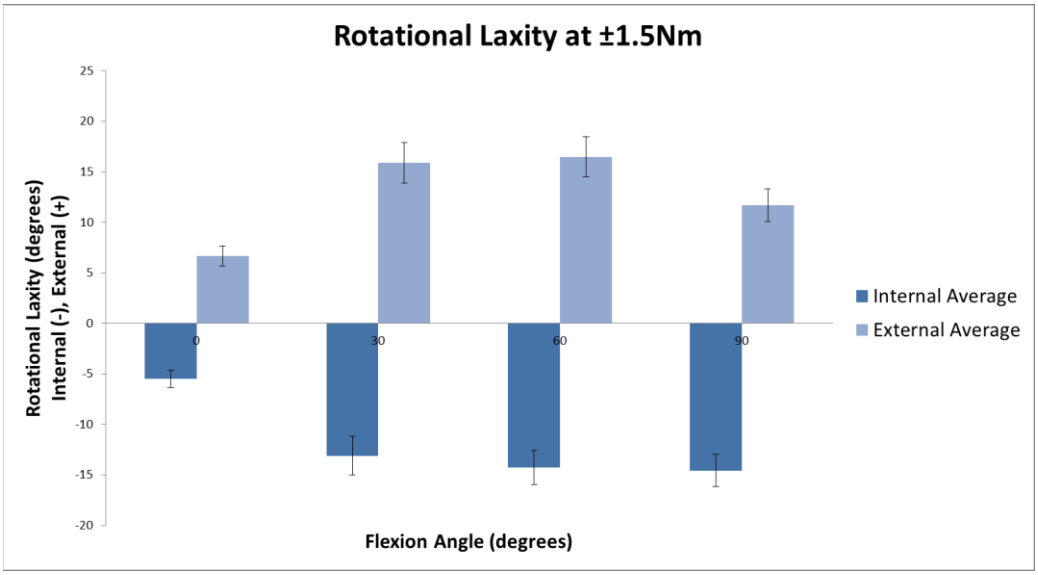

Figure A-2: Average IE laxity of 20 tested TKA knees. Error bars are standard error of the mean (SEM). Positive laxity values are external rotation of the tibia in degrees, and negative values are internal rotation.

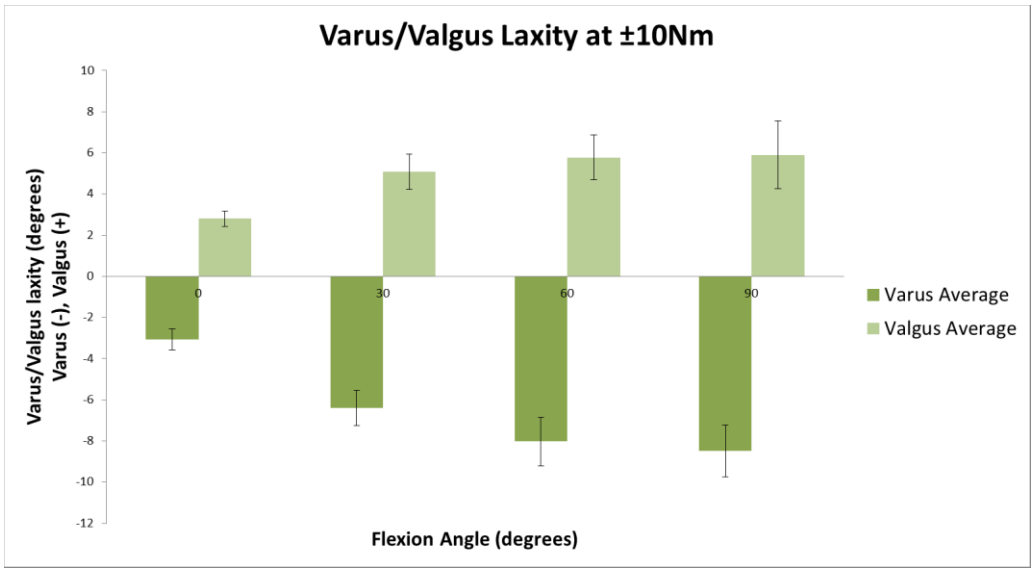

Figure A-3: Average VV laxity of 20 tested TKA knees. Error bars are standard error of the mean (SEM). Positive laxity values indicate valgus laxity and negative values are varus laxity.

APPENDIX B. MATLAB CODE

File conversion

The first matlab program converts text files to Excel files:

```
%% Kneetxt2xls
% This program converts text files produced from the Labview knee machine
% program to Excel files which can easily be used to calculate laxity data
% for tested specimens.

% Labview outputs files in .tsv format, so these must first be converted to
% text files as follows:
%
%       a. Place all .tsv files in one folder, hold shift + right-click on folder, and
%          select 'open command prompt here'.
%       b. Type 'ren *.tsv *.txt' into the command prompt.  Ensure that all
%          the file extensions have changed. Open at least one text file to
%          check that all data is present and correct.

% Next, place these text files in the Matlab directory and run this
% program.

clc, clear
QTM = dir(fullfile(pwd, '*.txt'));
 lQTM= size(QTM,1);
 for run = 1:lQTM

 % Rewrites header for Excel files
 header = {'Relative read time';'Flexion Angle (volts)'; 'Flexion Angle (degrees)';'Quad Force
(volts)';'Quad Force (Newtons)';...
    'Body Weight (Volts)';'Body Weight (Newtons)';'Pressure (volts)';'Pressure
(Pascals)';'Varus/Valgus Angle (volts)';...
    'Varus/Valgus Angle (degrees)';'Varus/Valgus Torque (Volts)';'Varus/Valgus Torque (N-m)';...
    'Rotational Angle (volts)';'Rotational Angle (degrees)';'Rotational Torque (Volts)';'Rotational
Torque (N-m)';
    'A/P Distance (Volts)';'A/P Distance (mm)';'A/P Force (Volts)';'A/P Force (Newtons)'}';
 fid = fopen(QTM(run).name);
 f= '%f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f %f';
 c = textscan(fid,f, 'Headerlines',10);

 matrix =cell2mat(c);

 % Separates loaded file name from extension
  [filename, ext] = strtok(QTM(run).name, '.');
   newext = '.xlsx';
[token, remain]=strtok(QTM(run).name, '.');
 combStr = strcat(token, newext);

 xlswrite(combStr,header);
```

```
xlswrite(combStr,matrix,'sheet1','A2');
end
disp('Text files converted to excel data')
```

Main Program

The main matlab program used to calculate laxity data for this study. Instructions for use are included in the comments.

```
%% Kneetest Main Program - Instructions for use

% Written by Erik Woodard on 1/9/2013

% This program reads in all knee test excel files in the Matlab directory and
% calculates the angle at which V/V Torque = +/-10Nm,IE Torque =
% +/-1.5Nm, and the distance at which A/P force = +/-35N.

% Operation instructions:
%
% 1. Convert all .txt or .tsv knee testing data to excel spreadsheets
%       a. Place all .tsv files in one folder, hold shift + right-click on folder, and
%          select 'open command prompt here'.
%       b. Type 'ren *.tsv *.txt' into the command prompt.  Ensure that all
%          the file extensions have changed. Open at least one text file to
%          check that all data is present and correct.
%
% 2. Convert text files to excel files.
%       a. Place text files in the 'File conversion' matlab folder.
%       b. Run the Kneetxt2xls.m program and ensure that all files have
%          been converted.

% 3. Calculate relevant data from excel files.
%       a. Place all excel files in the 'Knee test files' folder.
%       b. Run the kneetest_info program
%
% 'plotAP.xlsx', 'plotIE.xlsx' and 'plotVV.xlsx' contain angle and torque data useful
% for plotting graphs. Each sheet in the excel workbook is labeled with the
% name of the knee test. All data can be graphed at once using an
% excel macro. Ensure that the developer tab is checked in the Excel
% ribbon, and click the Macro button. Copy and paste the following code,
% and click 'run'.

% Code for creating Excel Macro to plot all charts at once:

% Sub plotcharts()
%     Dim chtTemp As Chart
%     Dim shtTemp As Worksheet
%     Dim rngData As Range
%     Dim sngLeft As Single, sngTop As Single, sngWidth As Single, sngHeight As Single
%     Dim lngCol As Long
%
```

```
%    sngLeft = 600
%    sngWidth = 400
%    sngHeight = 300
%
%    For Each shtTemp In ActiveWorkbook.Worksheets
%       sngTop = 50
%       lngCol = 2
%       Do While shtTemp.Cells(1, lngCol) <> ""
%          Set rngData = shtTemp.Cells(2, lngCol)
%          Set rngData = shtTemp.Range(rngData, rngData.End(xlDown)).Resize(, 2)
%          Set chtTemp = shtTemp.Shapes.AddChart(xlXYScatter, sngLeft, sngTop, sngWidth,
sngHeight).Chart
%          With chtTemp
%          .HasTitle = True
%          .ChartTitle.Text = "VVtorque vs. VVangle"
%          .Axes(xlCategory, xlPrimary).HasTitle = True
%          .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "VVtorque"
%          .Axes(xlCategory).MinimumScale = -20
%          .Axes(xlCategory).MaximumScale = 20
%          .Axes(xlValue, xlPrimary).HasTitle = True
%          .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "VVangle"
%          End With
%          With chtTemp.SeriesCollection.NewSeries
%             .Values = rngData.Columns(2)
%             .XValues = rngData.Columns(1)
%
%          End With
%          lngCol = lngCol + 2
%          sngTop = sngTop + sngHeight
%       Loop
%    Next
% End Sub

clc, clear

%List columns to call data in excel files.  These numbers may need to be
%changed depending on the way the data is stored and saved.

vvangle_col = 11;
vvtorque_col = 13;
bodyvv_col = 7;

ieangle_col = 15;
ietorque_col = 17;
bodyie_col = 7;

appos_col = 19;
apforce_col = 21;
bodyap_col = 7;
```

```
limit = 0.9; % set limit of data spread for interpolation
stdlim = 2; % set limit for elimination of data based on standard deviations from mean


%% Get directory and file names

warning('off','MATLAB:xlswrite:AddSheet') % Turn off warnings produced by erasing and
selecting Excel sheets when writing data
warning('off','MATLAB:NonIntegerInput')
dirName = pwd;              %# Matlab directory
files = dir( fullfile(dirName,'*.xlsx') );   % list all *.xlsx files
% Note: xlsx can be changed to xls if using an older version of Excel
files = {files.name}';   %'# file names
data = cell(numel(files),1); %# store file contents

% get number of each filename and preallocate variables in memory for speed
numIE = strfind(files,'IE');
numIE = numel(numIE(~cellfun('isempty',numIE)));
dataIE = cell(1,numIE);
filesIE = cell(1,numIE);

numVV = strfind(files,'VV');
numVV = numel(numVV(~cellfun('isempty',numVV)));
dataVV = cell(1,numVV);
filesVV = cell(1,numVV);

numAP = strfind(files,'AP');
numAP = numel(numAP(~cellfun('isempty',numAP)));
dataAP = cell(1,numAP);
filesAP = cell(1,numAP);

for i=1:numel(files)
   fname = fullfile(dirName,files{i});     %# full path to each file
   data{i} = xlsread(fname);  %# load excel file

   %separate data and files based on key words within filenames, store
   %data as cell arrays
   if isempty(cell2mat(strfind(files(i),'VV'))) == 0 || isempty(cell2mat(strfind(files(i),'vv'))) == 0
      dataVV{i} = data{i};
      filesVV{i} = files{i};
      if isempty(cell2mat(strfind(filesVV(i),'keyinfo'))) == 0 ||
isempty(cell2mat(strfind(filesVV(i),'plotVV'))) == 0 ||
isempty(cell2mat(strfind(filesVV(i),'BAD'))) == 0 || isempty(cell2mat(strfind(filesVV(i),'bad')))
== 0
         dataVV{i} = [];
         filesVV{i} = [];
      end

   elseif isempty(cell2mat(strfind(files(i),'IE'))) == 0 || isempty(cell2mat(strfind(files(i),'ie'))) == 0
      dataIE{i} = data{i};
      filesIE{i} = files{i};
```

```
        if isempty(cell2mat(strfind(filesIE(i),'keyinfo'))) == 0 ||
isempty(cell2mat(strfind(filesIE(i),'plotIE'))) == 0 || isempty(cell2mat(strfind(filesIE(i),'BAD')))
== 0 || isempty(cell2mat(strfind(filesIE(i),'bad'))) == 0
            dataIE{i} = [];
            filesIE{i} = [];
        end

    elseif isempty(cell2mat(strfind(files(i),'AP'))) == 0 || isempty(cell2mat(strfind(files(i),'ap'))) ==
0
        dataAP{i} = data{i};
        filesAP{i} = files{i};
        if isempty(cell2mat(strfind(filesAP(i),'keyinfo'))) == 0 ||
isempty(cell2mat(strfind(filesAP(i),'plotAP'))) == 0 ||
isempty(cell2mat(strfind(filesAP(i),'BAD'))) == 0 || isempty(cell2mat(strfind(filesAP(i),'bad')))
== 0
            dataAP{i} = [];
            filesAP{i} = [];
        end
    end
end

%remove empty cell arrays
dataIE = dataIE(~cellfun('isempty',dataIE));
dataVV = dataVV(~cellfun('isempty',dataVV));
filesIE = filesIE(~cellfun('isempty',filesIE));
filesVV = filesVV(~cellfun('isempty',filesVV));
dataAP = dataAP(~cellfun('isempty',dataAP));
filesAP = filesAP(~cellfun('isempty',filesAP));
clear fname

for indexIE = 1:length(filesIE)
    IE = cell2mat(dataIE(indexIE));
    IEangle = IE(:,ieangle_col); % Isolate angle and torque data from each file
    IEangle = removerows(IEangle,isnan(IEangle)); % Remove text, which appears as NaNs, from
data

   if isempty(cell2mat(strfind(filesIE(indexIE),'_EXT_'))) == 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'VC'))) ~= 0 && ...
            isempty(cell2mat(strfind(filesIE(indexIE),'NEUTRAL'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'UP'))) ~= 0 &&...
            isempty(cell2mat(strfind(filesIE(indexIE),'DOWN'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'ANTPIE'))) ~= 0 &&...
            isempty(cell2mat(strfind(filesIE(indexIE),'POSTANTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'STDREL'))) ~= 0 && ...
            isempty(cell2mat(strfind(filesIE(indexIE),'POSTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'ANTPOSTPIE'))) ~= 0

            start_angextIE = IEangle(1);

   elseif isempty(cell2mat(strfind(filesIE(indexIE),'_30_'))) == 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'VC'))) ~= 0 && ...
```

```
        isempty(cell2mat(strfind(filesIE(indexIE),'NEUTRAL'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'UP'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesIE(indexIE),'DOWN'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'ANTPIE'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesIE(indexIE),'POSTANTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'STDREL'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesIE(indexIE),'POSTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'ANTPOSTPIE'))) ~= 0

        start_ang30IE = IEangle(1);

  elseif isempty(cell2mat(strfind(filesIE(indexIE),'_60_'))) == 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'VC'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesIE(indexIE),'NEUTRAL'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'UP'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesIE(indexIE),'DOWN'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'ANTPIE'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesIE(indexIE),'POSTANTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'STDREL'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesIE(indexIE),'POSTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'ANTPOSTPIE'))) ~= 0

        start_ang60IE = IEangle(1);

  elseif isempty(cell2mat(strfind(filesIE(indexIE),'_90_'))) == 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'VC'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesIE(indexIE),'NEUTRAL'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'UP'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesIE(indexIE),'DOWN'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'ANTPIE'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesIE(indexIE),'POSTANTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'STDREL'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesIE(indexIE),'POSTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesIE(indexIE),'ANTPOSTPIE'))) ~= 0

        start_ang90IE = IEangle(1);

  end
end

for indexVV = 1:length(filesVV)
   VV = cell2mat(dataVV(indexVV));
   VVangle = VV(:,vvangle_col); % Isolate angle and torque data from each file
   VVangle = removerows(VVangle,isnan(VVangle)); % Remove text, which appears as NaNs,
from data

  if isempty(cell2mat(strfind(filesVV(indexVV),'_EXT_'))) == 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'VC'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesVV(indexVV),'NEUTRAL'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'UP'))) ~= 0 &&...
```

```
        isempty(cell2mat(strfind(filesVV(indexVV),'DOWN'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'ANTPIE'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesVV(indexVV),'POSTANTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'STDREL'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesVV(indexVV),'POSTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'ANTPOSTPIE'))) ~= 0

        start_angextVV = VVangle(1);

  elseif isempty(cell2mat(strfind(filesVV(indexVV),'_30_'))) == 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'VC'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesVV(indexVV),'NEUTRAL'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'UP'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesVV(indexVV),'DOWN'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'ANTPIE'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesVV(indexVV),'POSTANTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'STDREL'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesVV(indexVV),'POSTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'ANTPOSTPIE'))) ~= 0

        start_ang30VV = VVangle(1);

  elseif isempty(cell2mat(strfind(filesVV(indexVV),'_60_'))) == 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'VC'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesVV(indexVV),'NEUTRAL'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'UP'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesVV(indexVV),'DOWN'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'ANTPIE'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesVV(indexVV),'POSTANTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'STDREL'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesVV(indexVV),'POSTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'ANTPOSTPIE'))) ~= 0

        start_ang60VV = VVangle(1);

  elseif isempty(cell2mat(strfind(filesVV(indexVV),'_90_'))) == 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'VC'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesVV(indexVV),'NEUTRAL'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'UP'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesVV(indexVV),'DOWN'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'ANTPIE'))) ~= 0 &&...
        isempty(cell2mat(strfind(filesVV(indexVV),'POSTANTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'STDREL'))) ~= 0 && ...
        isempty(cell2mat(strfind(filesVV(indexVV),'POSTPIE'))) ~= 0 &&
isempty(cell2mat(strfind(filesVV(indexVV),'ANTPOSTPIE'))) ~= 0

        start_ang90VV = VVangle(1);

  end
end
```

```
for indexAP = 1:length(filesAP)
    AP = cell2mat(dataAP(indexAP));
    APpos = AP(:,appos_col); % Isolate angle and torque data from each file
    APpos = removerows(APpos,isnan(APpos)); % Remove text, which appears as NaNs, from
data

   if isempty(cell2mat(strfind(filesAP(indexAP),'EXT'))) == 0 &&
isempty(cell2mat(strfind(filesAP(indexAP),'VC'))) ~= 0

           start_posextAP = APpos(1);

   elseif isempty(cell2mat(strfind(filesAP(indexAP),'30'))) == 0 &&
isempty(cell2mat(strfind(filesAP(indexAP),'VC'))) ~= 0

           start_pos30AP = APpos(1);

   elseif isempty(cell2mat(strfind(filesAP(indexAP),'60'))) == 0 &&
isempty(cell2mat(strfind(filesAP(indexAP),'VC'))) ~= 0

           start_pos60AP = APpos(1);

   elseif isempty(cell2mat(strfind(filesAP(indexAP),'90'))) == 0 &&
isempty(cell2mat(strfind(filesAP(indexAP),'VC'))) ~= 0

           start_pos90AP = APpos(1);

   end
end

%% Find Varus/Valgus data, graph and calculate, write to files

MinVV = cell(1,length(filesVV)); % preallocate variables to increase speed
MaxVV = cell(1,length(filesVV));
vvlimpos = zeros(1,length(filesVV));
vvlimneg = zeros(1,length(filesVV));

kneetest_VV  %Run the matlab program to calculate varus/valgus angle data

%% Find Internal/External rotation data, graph and calculate

MinIE = cell(1,length(filesIE));
MaxIE = cell(1,length(filesIE));
ielimpos = zeros(1,length(filesIE));
ielimneg = zeros(1,length(filesIE));

kneetest_IE  %Run the matlab program to calculate internal/external angle data

%% Find Anterior/Posterior position data

MinAP = cell(1,length(filesAP)); % preallocate variables to increase speed
MaxAP = cell(1,length(filesAP));
```

```
kneetest_AP  %Run the matlab program to calculate anterior/posterior position data

 %% Average linear interpolated Varus/Valgus angles

angle_pos2VV = cell(1,length(filesVV)); % Preallocate variables
angle_neg2VV = cell(1,length(filesVV));
std_posVV = cell(1,length(filesVV));
std_negVV = cell(1,length(filesVV));
bodypos2VV = cell(1,length(filesVV));
bodyneg2VV = cell(1,length(filesVV));
angle_valgusVV = cell(1,length(filesVV));
angle_varusVV = cell(1,length(filesVV));

if ~isempty(filesVV)
for indexVV = 1:length(filesVV)

   mean_posVV = mean(cell2mat(angle_posVV(:,indexVV)));
   stdposVV = std(cell2mat(angle_posVV(:,indexVV)));

   % Remove outliers from data.  If data points are outside limit of
   % standard deviation defined earlier, that data is changed to an empty
   % cell.
   if mean_posVV > 0
      outlier_posVV = mean_posVV + stdlim*stdposVV;
      for j = 1:length(angle_posVV(:,indexVV))
         if angle_posVV{j,indexVV} > outlier_posVV
            angle_posVV{j,indexVV} = [];
         elseif angle_posVV{j,indexVV} < 0
            angle_posVV{j,indexVV} = [];
         end
      end
   end

   if mean_posVV < 0
      outlier_posVV = mean_posVV - stdlim*stdposVV;
       for j = 1:length(angle_posVV(:,indexVV))
         if angle_posVV{j,indexVV} < outlier_posVV
            angle_posVV{j,indexVV} = [];
         elseif angle_posVV{j,indexVV} > 0
            angle_posVV{j,indexVV} = [];
         end
      end
   end

   mean_negVV = mean(cell2mat(angle_negVV(:,indexVV)));
   stdnegVV = std(cell2mat(angle_negVV(:,indexVV)));

   if mean_negVV > 0
      outlier_negVV = mean_negVV + stdlim*stdnegVV;
      for j = 1:length(angle_negVV(:,indexVV))
```

```
        if angle_negVV{j,indexVV} > outlier_negVV
            angle_negVV{j,indexVV} = [];
        elseif angle_negVV{j,indexVV} < 0
            angle_negVV{j,indexVV} = [];
        end
    end
end

if mean_negVV < 0
    outlier_negVV = mean_negVV - stdlim*stdnegVV;
    for j = 1:length(angle_negVV(:,indexVV))
        if angle_negVV{j,indexVV} < outlier_negVV
            angle_negVV{j,indexVV} = [];
        elseif angle_negVV{j,indexVV} > 0
            angle_negVV{j,indexVV} = [];
        end
    end
end

angle_pos2VV(indexVV) = {round(mean(cell2mat(angle_posVV(:,indexVV)))*10^3)/10^3};
angle_neg2VV(indexVV) = {round(mean(cell2mat(angle_negVV(:,indexVV)))*10^3)/10^3};

angle_valgusVV(indexVV) = {abs(angle_neg2VV{indexVV})};
angle_varusVV(indexVV) = {-1*abs(angle_pos2VV{indexVV})};

std_posVV(indexVV) = {(std(cell2mat(angle_posVV(:,indexVV))))};
std_negVV(indexVV) = {(std(cell2mat(angle_negVV(:,indexVV))))};

%    toterr_posVV(indexVV) = {(sqrt(std_posVV{indexVV}^2) + (errVV^2))}; %total system
error
%    toterr_negVV(indexVV) = {(sqrt(std_negVV{indexVV}^2) + (errVV^2))};

bodypos2VV(indexVV) = {(mean(cell2mat(bodyposVV(:,indexVV))))};
bodyneg2VV(indexVV) = {(mean(cell2mat(bodynegVV(:,indexVV))))};

 end
end
 %% Average linear interpolated Internal/External rotation angles

angle_ext2IE = cell(1,length(filesIE)); % Preallocate variables
angle_int2IE = cell(1,length(filesIE));
std_posIE = cell(1,length(filesIE));
std_negIE = cell(1,length(filesIE));
bodypos2IE = cell(1,length(filesIE));
bodyneg2IE = cell(1,length(filesIE));
angle_internalIE = cell(1,length(filesIE));
angle_externalIE = cell(1,length(filesIE));

if ~isempty(filesIE)

for indexIE = 1:length(filesIE)
```

```
mean_posIE = mean(cell2mat(angle_extIE(:,indexIE)));
stdposIE = std(cell2mat(angle_extIE(:,indexIE)));

if mean_posIE > 0
   outlier_posIE = mean_posIE + stdlim*stdposIE;
   for j = 1:length(angle_extIE(:,indexIE))
      if angle_extIE{j,indexIE} > outlier_posIE
         angle_extIE{j,indexIE} = [];
      elseif angle_extIE{j,indexIE} < 0
         angle_extIE{j,indexIE} = [];
      end
   end
end

if mean_posIE < 0
   outlier_posIE = mean_posIE - stdlim*stdposIE;
   for j = 1:length(angle_extIE(:,indexIE))
      if angle_extIE{j,indexIE} < outlier_posIE
         angle_extIE{j,indexIE} = [];
      elseif angle_extIE{j,indexIE} > 0
         angle_extIE{j,indexIE} = [];
      end
   end
end

mean_negIE = mean(cell2mat(angle_intIE(:,indexIE)));
stdnegIE = std(cell2mat(angle_intIE(:,indexIE)));

if mean_negIE > 0
   outlier_negIE = mean_negIE + stdlim*stdnegIE;
   for j = 1:length(angle_intIE(:,indexIE))
      if angle_intIE{j,indexIE} > outlier_negIE
         angle_intIE{j,indexIE} = [];
      elseif angle_intIE{j,indexIE} < 0
         angle_intIE{j,indexIE} = [];
      end
   end
end

if mean_negIE < 0
   outlier_negIE = mean_negIE - stdlim*stdnegIE;
   for j = 1:length(angle_intIE(:,indexIE))
      if angle_intIE{j,indexIE} < outlier_negIE
         angle_intIE{j,indexIE} = [];
      elseif angle_intIE{j,indexIE} > 0
         angle_intIE{j,indexIE} = [];
      end
   end
end
```

```
    angle_ext2IE(indexIE) = {round(mean(cell2mat(angle_extIE(:,indexIE)))*10^3)/10^3};
    angle_int2IE(indexIE) = {round(mean(cell2mat(angle_intIE(:,indexIE)))*10^3)/10^3};

    %Grood and Suntay propose a specific sign convention - internal
    %rotation is negative and external is positive.

    angle_internalIE(indexIE) = {-1*abs(angle_int2IE{indexIE})};
    angle_externalIE(indexIE) = {abs(angle_ext2IE{indexIE})};

    std_negIE(indexIE) = {(std(cell2mat(angle_extIE(:,indexIE))))};
    std_posIE(indexIE) = {(std(cell2mat(angle_intIE(:,indexIE))))};

%     toterr_posIE(indexIE) = {sqrt((std_posIE{indexIE}^2) + (errIE^2))};
%     toterr_negIE(indexIE) = {sqrt((std_negIE{indexIE}^2) + (errIE^2))};

    bodypos2IE(indexIE) = {(mean(cell2mat(bodyposIE(:,indexIE))))};
    bodyneg2IE(indexIE) = {(mean(cell2mat(bodynegIE(:,indexIE))))};
end
end

%% Average linear interpolated Anterior/Posterior position data

angle_pos2AP = cell(1,length(filesAP)); % Preallocate variables
angle_neg2AP = cell(1,length(filesAP));
std_posAP = cell(1,length(filesAP));
std_negAP = cell(1,length(filesAP));
bodypos2AP = cell(1,length(filesAP));
bodyneg2AP = cell(1,length(filesAP));
angle_zero2AP = cell(1,length(filesAP));
angle_anteriorAP = cell(1,length(filesAP));
angle_posteriorAP = cell(1,length(filesAP));

if ~isempty(filesAP)
for indexAP = 1:length(filesAP)

    mean_posAP = mean(cell2mat(angle_posAP(:,indexAP)));
    stdposAP = std(cell2mat(angle_posAP(:,indexAP)));

    if mean_posAP > 0
        outlier_posAP = mean_posAP + stdlim*stdposAP;
        for j = 1:length(angle_posAP(:,indexAP))
            if angle_posAP{j,indexAP} > outlier_posAP
                angle_posAP{j,indexAP} = [];
            elseif angle_posAP{j,indexAP} < 0
                angle_posAP{j,indexAP} = [];
            end
        end
    end

    if mean_posAP < 0
        outlier_posAP = mean_posAP - stdlim*stdposAP;
```

```
        for j = 1:length(angle_posAP(:,indexAP))
            if angle_posAP{j,indexAP} < outlier_posAP
                angle_posAP{j,indexAP} = [];
            elseif angle_posAP{j,indexAP} > 0
                angle_posAP{j,indexAP} = [];
            end
        end
    end

    mean_negAP = mean(cell2mat(angle_negAP(:,indexAP)));
    stdnegAP = std(cell2mat(angle_negAP(:,indexAP)));

    if mean_negAP > 0
        outlier_negAP = mean_negAP + stdlim*stdnegAP;
        for j = 1:length(angle_negAP(:,indexAP))
            if angle_negAP{j,indexAP} > outlier_negAP
                angle_negAP{j,indexAP} = [];
            elseif angle_negAP{j,indexAP} < 0
                angle_negAP{j,indexAP} = [];
            end
        end
    end

    if mean_negAP < 0
        outlier_negAP = mean_negAP - stdlim*stdnegAP;
        for j = 1:length(angle_negAP(:,indexAP))
            if angle_negAP{j,indexAP} < outlier_negAP
                angle_negAP{j,indexAP} = [];
            elseif angle_negAP{j,indexAP} > 0
                angle_negAP{j,indexAP} = [];
            end
        end
    end

    angle_pos2AP(indexAP) = {round(mean(cell2mat(angle_posAP(:,indexAP)))*10^3)/10^3};
    angle_neg2AP(indexAP) = {round(mean(cell2mat(angle_negAP(:,indexAP)))*10^3)/10^3};

    angle_anteriorAP(indexAP) = {abs(angle_pos2AP{indexAP})};
    angle_posteriorAP(indexAP) = {-1*abs(angle_neg2AP{indexAP})};

    std_posAP(indexAP) = {(std(cell2mat(angle_posAP(:,indexAP))))};
    std_negAP(indexAP) = {(std(cell2mat(angle_negAP(:,indexAP))))};

    bodypos2AP(indexAP) = {(mean(cell2mat(bodyposAP(:,indexAP))))};
    bodyneg2AP(indexAP) = {(mean(cell2mat(bodynegAP(:,indexAP))))};

 end
end

clear  data dirname i angle_posIE angle_posAP angle_negIE angle_negAP
clear mean_posAP mean_posIE mean_posVV mean_negAP mean_negIE mean_negVV
```

```
clear stdposAP stdposIE stdposVV stdnegAP stdnegIE stdnegVV stdlim limit j
clear outlier_posAP outlier_posIE outlier_posVV outlier_negAP outlier_negIE outlier_negVV
%% Save file names and angle data to array

if ~isempty(filesVV)
[nameVV,~] = strtok(filesVV,'.'); % VV file names
arrayVV =
[nameVV;angle_varusVV;std_posVV;bodypos2VV;MaxVV;angle_valgusVV;std_negVV;bodyn
eg2VV;MinVV]';  % concatenate data into a single array
end

if ~isempty(filesIE)
[nameIE,~] = strtok(filesIE,'.'); % IE file names
arrayIE =
[nameIE;angle_internalIE;std_posIE;bodypos2IE;MaxIE;angle_externalIE;std_negIE;bodyneg2I
E;MinIE]';
end

if ~isempty(filesAP)
[nameAP,~] = strtok(filesAP,'.'); % AP file names
arrayAP =
[nameAP;angle_anteriorAP;std_posAP;bodypos2AP;MaxAP;angle_posteriorAP;std_negAP;bod
yneg2AP;MinAP]';
end

clear indexVV indexIE indexAP nameVV nameIE nameAP
%% Save array data to excel and text files

if ~isempty(filesVV)
 HeaderVV = {'Filename'; 'Varus Angle (degrees)';'Standard Deviation of +10Nm Angle';'Body
Weight at Positive Angle'; 'Value of Maxtorque (Nm)'; 'Valgus Angle (degrees)';'Standard
Deviation of -10Nm Angle';'Body Weight at Negative Angle';'Value of Mintorque (Nm)'}';
 arrayVV = [HeaderVV;arrayVV];
 xlswrite([specVV '_VV_keyinfo.xlsx'],arrayVV)
disp('Varus/Valgus data written to VV_keyinfo excel file')
end

if ~isempty(filesIE)
 HeaderIE = {'Filename'; 'Internal Rotation Angle (degrees)';'Standard Deviation of Internal
Angle'; 'BodyWeight at Positive Angle'; 'Value of Maxtorque (Nm)'; 'External Rotation Angle
(degrees)';'Standard Deviation of External Angle';'Body Weight at Negative Angle'; 'Value of
Mintorque (Nm)'}';
 arrayIE = [HeaderIE;arrayIE];
 xlswrite([specIE '_IE_keyinfo.xlsx'],arrayIE)
disp('Internal/External Rotation data written to IE_keyinfo excel file')
end

if ~isempty(filesAP)
 HeaderAP = {'Filename'; 'Anterior Displacement at +35N (mm)';'Standard Deviation of +35Nm
position'; 'Body Weight at +35N'; 'Value of Max Force (N)'; 'Posterior Displacement at -35N
```

```
(mm)';'Standard Deviation of -35Nm position'; 'Body Weight at -35N'; 'Value of Min Force
(N)'}';
 arrayAP = [HeaderAP;arrayAP];
 xlswrite([specAP '_AP_keyinfo.xlsx'],arrayAP)
disp('Anterior/Posterior data written to AP_keyinfo excel file')
end
 %% Remove unnecessary sheets

if ~isempty(filesVV)
excelFileName = [specVV '_'   'plotVV.xlsx'];
excelFilePath = pwd; % Current working directory.
sheetName = 'Sheet'; % EN: Sheet, DE: Tabelle, etc. (Lang. dependent)

% Open Excel file.
objExcel = actxserver('Excel.Application');
objExcel.Workbooks.Open(fullfile(excelFilePath, excelFileName)); % Full path is necessary!

% Delete first 3 blank sheets.
try
% Throws an error if the sheets do not exist.
objExcel.ActiveWorkbook.Worksheets.Item([sheetName '1']).Delete;
objExcel.ActiveWorkbook.Worksheets.Item([sheetName '2']).Delete;
objExcel.ActiveWorkbook.Worksheets.Item([sheetName '3']).Delete;
catch
 % Do nothing.
end

% Save, close and clean up.
objExcel.ActiveWorkbook.Save;
objExcel.ActiveWorkbook.Close;
objExcel.Quit;
objExcel.delete;
 end

 if ~isempty(filesIE)
excelFileName = [specIE '_'   'plotIE.xlsx'];
excelFilePath = pwd; % Current working directory.
sheetName = 'Sheet'; % EN: Sheet, DE: Tabelle, etc. (Lang. dependent)

% Open Excel file.
objExcel = actxserver('Excel.Application');
objExcel.Workbooks.Open(fullfile(excelFilePath, excelFileName)); % Full path is necessary!

% Delete sheets.
try
% Throws an error if the sheets do not exist.
objExcel.ActiveWorkbook.Worksheets.Item([sheetName '1']).Delete;
objExcel.ActiveWorkbook.Worksheets.Item([sheetName '2']).Delete;
objExcel.ActiveWorkbook.Worksheets.Item([sheetName '3']).Delete;
catch
 % Do nothing.
```

```
end

% Save, close and clean up.
objExcel.ActiveWorkbook.Save;
objExcel.ActiveWorkbook.Close;
objExcel.Quit;
objExcel.delete;
 end

 if ~isempty(filesAP)
excelFileName = [specAP '_'   'plotAP.xlsx'];
excelFilePath = pwd; % Current working directory.
sheetName = 'Sheet'; % EN: Sheet, DE: Tabelle, etc. (Lang. dependent)

% Open Excel file.
objExcel = actxserver('Excel.Application');
objExcel.Workbooks.Open(fullfile(excelFilePath, excelFileName)); % Full path is necessary!

% Delete sheets.
try
% Throws an error if the sheets do not exist.
objExcel.ActiveWorkbook.Worksheets.Item([sheetName '1']).Delete;
objExcel.ActiveWorkbook.Worksheets.Item([sheetName '2']).Delete;
objExcel.ActiveWorkbook.Worksheets.Item([sheetName '3']).Delete;
catch
 % Do nothing.
end

% Save, close and clean up.
objExcel.ActiveWorkbook.Save;
objExcel.ActiveWorkbook.Close;
objExcel.Quit;
objExcel.delete;
 end

clear  numIE numVV numAP dataIE dataVV dataAP output_file ans titleVV titleIE titleAP %
clear remaining unused variables

clear objExcel HeaderVV HeaderIE HeaderAP sheetName excelFileName excelFilePath %clear
remaining unnecessary variables
clear specVV partVV specIE partIE specAP partAP indmaxAP indminAP indmaxIE indminIE ...
     IEangle APpos
```

Anterior/posterior calculations

```
%% Kneetest_AP
% This subprogram calculates laxity data for all anterior/posterior drawer
% tests.  It is called by the main program kneetest_info.

if ~isempty(filesAP)
for indexAP = 1:length(filesAP) % this loop only applies to files containing AP in the filename

   AP = cell2mat(dataAP(indexAP));

   APpos = (AP(:,appos_col)); % Isolate angle and torque data from each file
   APpos = removerows(APpos,isnan(APpos)); % Remove text, which appears as NaNs, from
data
   APforce = (AP(:,apforce_col));
   APforce = removerows(APforce,isnan(APforce));
   bodyAP = AP(:,bodyap_col);
   bodyAP = removerows(bodyAP,isnan(bodyAP));

   APpos = removerows(APpos,APforce == -35);
   APforce = removerows(APforce,APforce == -35);
   APpos = removerows(APpos,APforce == 35);
   APforce = removerows(APforce,APforce == 35);

   [xAP,indmaxAP] = max(APforce); % Find minimum and maximum of all force data for each
file
   [yAP,indminAP] = min(APforce);
   MaxAP(indexAP) = {round(xAP*10^3)/10^3}; % Store min and max values in a cell array
   MinAP(indexAP) = {round(yAP*10^3)/10^3};


   [specAP, partAP, c, d, ~, ~] = strread(filesAP{indexAP}, '%s %s %s %s %s %s', 'delimiter',
'_');
   specAP = cell2mat(specAP);
   partAP = cell2mat(partAP);
   c = cell2mat(c);
   d = cell2mat(d);
   sheet = [partAP '_' c '_' d]; %only use desired parts of filename for sheet titles
   excelarrayAP = horzcat(APforce, APpos);
   headerAP = {'APforce (N)' 'AP position (mm)'};
   xlswrite([specAP '_'  'plotAP.xlsx'],headerAP,sheet)
   xlswrite([specAP '_'  'plotAP.xlsx'],excelarrayAP,sheet,'A2') %save plot data to excel
spreadsheet, with each file on a diferent sheet
   clear c d sheet

for i = 2:length(APforce)

   % START FLEXION ANGLE = 90
   if isempty(cell2mat(strfind(filesAP(indexAP),'_90_'))) == 0
      APposflex = APpos - start_pos90AP;
```

```
  if xAP > 35
     if APforce(i) > 35 && APforce(i-1) < 35 % only performs calculation on specific data
        angle_posAP(i,indexAP) = {((35 - APforce(i-1))*(APposflex(i)-APposflex(i-
1))/(APforce(i)-APforce(i-1)) + APposflex(i-1))};
        bodyposAP(i,indexAP) = {bodyAP(i)};
     elseif APforce(i) < 35 && APpos(i-1) > 35
        angle_posAP(i,indexAP) = {((35 - APforce(i))*(APposflex(i-1)-APposflex(i))/(APforce(i-
1)-APforce(i)) + APposflex(i))};
        bodyposAP(i,indexAP) = {bodyAP(i)};
     %perform linear interpolation on all points which meet the criteria
     end
  else
     if APforce(i) > xAP*limit && APforce(i-1) < xAP*limit
        angle_posAP(i,indexAP) = {((35 - APforce(i-1))*(APposflex(i)-APpos(i-1))/(APforce(i)-
APforce(i-1)) + APposflex(i-1))};
        bodyposAP(i,indexAP) = {bodyAP(i)};
     elseif APforce(i) < xAP*limit && APforce(i-1) > xAP*limit
        angle_posAP(i,indexAP) = {((35 - APforce(i))*(APposflex(i-1)-APposflex(i))/(APforce(i-
1)-APforce(i)) + APposflex(i))};
        bodyposAP(i,indexAP) = {bodyAP(i)};
     end
  end


  if yAP < -35
     if APforce(i) > -35 && APforce(i-1) < -35
        angle_negAP(i,indexAP) = {((-35 - APforce(i-1))*(APposflex(i)-APposflex(i-
1))/(APforce(i)-APforce(i-1)) + APposflex(i-1))};
        bodynegAP(i,indexAP) = {bodyAP(i)};
     elseif APforce(i) < -35 && APforce(i-1) > -35
        angle_negAP(i,indexAP) = {((-35 - APforce(i))*(APposflex(i-1)-
APposflex(i))/(APforce(i-1)-APforce(i)) + APposflex(i))};
        bodynegAP(i,indexAP) = {bodyAP(i)};
     end

  else
     if APforce(i) > yAP*limit && APforce(i-1) < yAP*limit
        angle_negAP(i,indexAP) = {((-35 - APforce(i-1))*(APposflex(i)-APposflex(i-
1))/(APforce(i)-APforce(i-1)) + APposflex(i-1))};
        bodynegAP(i,indexAP) = {bodyAP(i)};
     elseif APforce(i) < yAP*limit && APforce(i-1) > yAP*limit
        angle_negAP(i,indexAP) = {((-35 - APforce(i))*(APposflex(i-1)-
APposflex(i))/(APforce(i-1)-APforce(i)) + APposflex(i))};
        bodynegAP(i,indexAP) = {bodyAP(i)};
     end
  end


  %START FLEXION ANGLE = 60
  elseif isempty(cell2mat(strfind(filesAP(indexAP),'_60_'))) == 0
     APpos60 = APpos - start_pos60AP;
```

```
  if xAP > 35
     if APforce(i) > 35 && APforce(i-1) < 35 % only performs calculation on specific data
        angle_posAP(i,indexAP) = {((35 - APforce(i-1))*(APpos60(i)-APpos60(i-
1))/(APforce(i)-APforce(i-1)) + APpos60(i-1))};
        bodyposAP(i,indexAP) = {bodyAP(i)};
     elseif APforce(i) < 35 && APpos(i-1) > 35
        angle_posAP(i,indexAP) = {((35 - APforce(i))*(APpos60(i-1)-APpos60(i))/(APforce(i-
1)-APforce(i)) + APpos60(i))};
        bodyposAP(i,indexAP) = {bodyAP(i)};
     %perform linear interpolation on all points which meet the criteria
     end
  else
     if APforce(i) > xAP*limit && APforce(i-1) < xAP*limit
        angle_posAP(i,indexAP) = {((35 - APforce(i-1))*(APpos60(i)-APpos60(i-
1))/(APforce(i)-APforce(i-1)) + APpos60(i-1))};
        bodyposAP(i,indexAP) = {bodyAP(i)};
     elseif APforce(i) < xAP*limit && APforce(i-1) > xAP*limit
        angle_posAP(i,indexAP) = {((35 - APforce(i))*(APpos60(i-1)-APpos60(i))/(APforce(i-
1)-APforce(i)) + APpos60(i))};
        bodyposAP(i,indexAP) = {bodyAP(i)};
     end
  end


  if yAP < -35
     if APforce(i) > -35 && APforce(i-1) < -35
        angle_negAP(i,indexAP) = {((-35 - APforce(i-1))*(APpos60(i)-APpos60(i-
1))/(APforce(i)-APforce(i-1)) + APpos60(i-1))};
        bodynegAP(i,indexAP) = {bodyAP(i)};
     elseif APforce(i) < -35 && APforce(i-1) > -35
        angle_negAP(i,indexAP) = {((-35 - APforce(i))*(APpos60(i-1)-APpos60(i))/(APforce(i-
1)-APforce(i)) + APpos60(i))};
        bodynegAP(i,indexAP) = {bodyAP(i)};
     end

  else
     if APforce(i) > yAP*limit && APforce(i-1) < yAP*limit
        angle_negAP(i,indexAP) = {((-35 - APforce(i-1))*(APpos60(i)-APpos60(i-
1))/(APforce(i)-APforce(i-1)) + APpos60(i-1))};
        bodynegAP(i,indexAP) = {bodyAP(i)};
     elseif APforce(i) < yAP*limit && APforce(i-1) > yAP*limit
        angle_negAP(i,indexAP) = {((-35 - APforce(i))*(APpos60(i-1)-APpos60(i))/(APforce(i-
1)-APforce(i)) + APpos60(i))};
        bodynegAP(i,indexAP) = {bodyAP(i)};
     end
  end


  % START FLEXION ANGLE = 30
  elseif isempty(cell2mat(strfind(filesAP(indexAP),'_30_'))) == 0
```

```
    APpos30 = APpos - start_pos30AP;

  if xAP > 35
    if APforce(i) > 35 && APforce(i-1) < 35 % only performs calculation on specific data
      angle_posAP(i,indexAP) = {((35 - APforce(i-1))*(APpos30(i)-APpos30(i-
1))/(APforce(i)-APforce(i-1)) + APpos30(i-1))};
      bodyposAP(i,indexAP) = {bodyAP(i)};
    elseif APforce(i) < 35 && APpos(i-1) > 35
      angle_posAP(i,indexAP) = {((35 - APforce(i))*(APpos30(i-1)-APpos30(i))/(APforce(i-
1)-APforce(i)) + APpos30(i))};
      bodyposAP(i,indexAP) = {bodyAP(i)};
    %perform linear interpolation on all points which meet the criteria
    end
  else
    if APforce(i) > xAP*limit && APforce(i-1) < xAP*limit
      angle_posAP(i,indexAP) = {((35 - APforce(i-1))*(APpos30(i)-APpos30(i-
1))/(APforce(i)-APforce(i-1)) + APpos30(i-1))};
      bodyposAP(i,indexAP) = {bodyAP(i)};
    elseif APforce(i) < xAP*limit && APforce(i-1) > xAP*limit
      angle_posAP(i,indexAP) = {((35 - APforce(i))*(APpos30(i-1)-APpos30(i))/(APforce(i-
1)-APforce(i)) + APpos30(i))};
      bodyposAP(i,indexAP) = {bodyAP(i)};
    end
  end


  if yAP < -35
    if APforce(i) > -35 && APforce(i-1) < -35
      angle_negAP(i,indexAP) = {((-35 - APforce(i-1))*(APpos30(i)-APpos30(i-
1))/(APforce(i)-APforce(i-1)) + APpos30(i-1))};
      bodynegAP(i,indexAP) = {bodyAP(i)};
    elseif APforce(i) < -35 && APforce(i-1) > -35
      angle_negAP(i,indexAP) = {((-35 - APforce(i))*(APpos30(i-1)-APpos30(i))/(APforce(i-
1)-APforce(i)) + APpos30(i))};
      bodynegAP(i,indexAP) = {bodyAP(i)};
    end

  else
    if APforce(i) > yAP*limit && APforce(i-1) < yAP*limit
      angle_negAP(i,indexAP) = {((-35 - APforce(i-1))*(APpos30(i)-APpos30(i-
1))/(APforce(i)-APforce(i-1)) + APpos30(i-1))};
      bodynegAP(i,indexAP) = {bodyAP(i)};
    elseif APforce(i) < yAP*limit && APforce(i-1) > yAP*limit
      angle_negAP(i,indexAP) = {((-35 - APforce(i))*(APpos30(i-1)-APpos30(i))/(APforce(i-
1)-APforce(i)) + APpos30(i))};
      bodynegAP(i,indexAP) = {bodyAP(i)};
    end
  end


  % START FLEXION ANGLE = EXT
```

```
   elseif isempty(cell2mat(strfind(filesAP(indexAP),'_EXT_'))) == 0
      APposext = APpos - start_posextAP;

   if xAP > 35
      if APforce(i) > 35 && APforce(i-1) < 35 % only performs calculation on specific data
         angle_posAP(i,indexAP) = {((35 - APforce(i-1))*(APposext(i)-APposext(i-
1))/(APforce(i)-APforce(i-1)) + APposext(i-1))};
         bodyposAP(i,indexAP) = {bodyAP(i)};
      elseif APforce(i) < 35 && APpos(i-1) > 35
         angle_posAP(i,indexAP) = {((35 - APforce(i))*(APposext(i-1)-APposext(i))/(APforce(i-
1)-APforce(i)) + APposext(i))};
         bodyposAP(i,indexAP) = {bodyAP(i)};
      %perform linear interpolation on all points which meet the criteria
      end
   else
      if APforce(i) > xAP*limit && APforce(i-1) < xAP*limit
         angle_posAP(i,indexAP) = {((35 - APforce(i-1))*(APposext(i)-APposext(i-
1))/(APforce(i)-APforce(i-1)) + APposext(i-1))};
         bodyposAP(i,indexAP) = {bodyAP(i)};
      elseif APforce(i) < xAP*limit && APforce(i-1) > xAP*limit
         angle_posAP(i,indexAP) = {((35 - APforce(i))*(APposext(i-1)-APposext(i))/(APforce(i-
1)-APforce(i)) + APposext(i))};
         bodyposAP(i,indexAP) = {bodyAP(i)};
      end
   end


   if yAP < -35
      if APforce(i) > -35 && APforce(i-1) < -35
         angle_negAP(i,indexAP) = {((-35 - APforce(i-1))*(APposext(i)-APposext(i-
1))/(APforce(i)-APforce(i-1)) + APposext(i-1))};
         bodynegAP(i,indexAP) = {bodyAP(i)};
      elseif APforce(i) < -35 && APforce(i-1) > -35
         angle_negAP(i,indexAP) = {((-35 - APforce(i))*(APposext(i-1)-APposext(i))/(APforce(i-
1)-APforce(i)) + APposext(i))};
         bodynegAP(i,indexAP) = {bodyAP(i)};
      end

   else
      if APforce(i) > yAP*limit && APforce(i-1) < yAP*limit
         angle_negAP(i,indexAP) = {((-35 - APforce(i-1))*(APposext(i)-APposext(i-
1))/(APforce(i)-APforce(i-1)) + APposext(i-1))};
         bodynegAP(i,indexAP) = {bodyAP(i)};
      elseif APforce(i) < yAP*limit && APforce(i-1) > yAP*limit
         angle_negAP(i,indexAP) = {((-35 - APforce(i))*(APposext(i-1)-APposext(i))/(APforce(i-
1)-APforce(i)) + APposext(i))};
         bodynegAP(i,indexAP) = {bodyAP(i)};
      end
   end

   end
```

```
end

end
else disp('No AP files to process') %display error if there are no files containing A/P data
end

clear xAP yAP excelarrayAP AP appos_col apforce_col bodyap_col
```

Internal/External Calculations

```
%% Kneetest_IE
% This subprogram calculates laxity data for all internal/external rotation
% tests.  It is called by the main program kneetest_info.  For these
% calculations, it is necessary to files based on left or right
% designation, since the Labview program does not perform sign corrections.

if ~isempty(filesIE)
for indexIE = 1:length(filesIE)  % this loop only applies to files containing IE in the filename.

   IE = cell2mat(dataIE(indexIE));
   IEangle = IE(:,ieangle_col); % Isolate angle and torque data from each file
   IEangle = removerows(IEangle,isnan(IEangle)); % Remove text, which appears as NaNs, from
data
   IEtorque = IE(:,ietorque_col);
   IEtorque = removerows(IEtorque,isnan(IEtorque));
   bodyIE = IE(:,bodyie_col);
   bodyIE = removerows(bodyIE,isnan(bodyIE));

   IEangle = removerows(IEangle,IEtorque == -1.5);
   IEtorque = removerows(IEtorque,IEtorque == -1.5);
   IEangle = removerows(IEangle,IEtorque == 1.5);
   IEtorque = removerows(IEtorque,IEtorque == 1.5);

   IEtorque = removerows(IEtorque,'ind',IEtorque == 0);
   IEangle = removerows(IEangle,'ind',IEtorque == 0);



   [xIE,indmaxIE] = max(IEtorque); % Find minimum and maximum of all torque data for each
file
   [yIE,indminIE] = min(IEtorque);
   MaxIE(indexIE) = {round(xIE*10^3)/10^3}; % Store min and max values in a cell array
   MinIE(indexIE) = {round(yIE*10^3)/10^3};

   IEtorquezero = zeros(indmaxIE,1);
   IEtorquezero2 = zeros(indminIE,1);
   IEanglezero = zeros(indmaxIE,1);
   IEanglezero2 = zeros(indminIE,1);
```

```
bodyIE_zero = zeros(indmaxIE,1);
bodyIE_zero2 = zeros(indminIE,1);

for k = 1:length(IEtorque)
   if indmaxIE < indminIE
      if k <= indmaxIE

         IEtorquezero(k) = IEtorque(k);
         IEanglezero(k) = IEangle(k);
         bodyIE_zero(k) = bodyIE(k);

      elseif k > indmaxIE && k <= indminIE
         IEtorquezero2(k) = IEtorque(k);
         IEanglezero2(k) = IEangle(k);
         bodyIE_zero2(k) = bodyIE(k);
      end

   elseif indmaxIE > indminIE
      if k <= indminIE
         IEtorquezero(k) = IEtorque(k);
         IEanglezero(k) = IEangle(k);
         bodyIE_zero(k) = bodyIE(k);

      elseif k > indminIE && k <= indmaxIE
         IEtorquezero2(k) = IEtorque(k);
         IEanglezero2(k) = IEangle(k);
         bodyIE_zero2(k) = bodyIE(k);

      end
   end
end

%    IEtorquezero2(IEtorquezero2 == 0) = [];
%    IEanglezero2(IEanglezero2 == 0) = [];
%    bodyIE_zero2(bodyIE_zero2 == 0) = [];

[~,startIE] = min(abs(IEanglezero(1)-IEanglezero2));
IEtorquezero3 = zeros(length(IEtorquezero2)-startIE,1);
IEanglezero3 = zeros(length(IEanglezero2)-startIE,1);
bodyIE_zero3 = zeros(length(bodyIE_zero2)-startIE,1);

for j = 1:length(IEtorquezero2)
   if j > startIE
      IEtorquezero3(j) = IEtorquezero2(j);
      IEanglezero3(j) = IEanglezero2(j);
      bodyIE_zero3(j) = bodyIE_zero2(j);
   end
end

IEtorquezero3(IEtorquezero3 == 0) = [];
IEanglezero3(IEanglezero3 == 0) = [];
```

```
    bodyIE_zero3(bodyIE_zero3 == 0) = [];

    IEtorque = [flipud(IEtorquezero3); IEtorquezero];
    IEangle = [flipud(IEanglezero3); IEanglezero];
    bodyIE = [flipud(bodyIE_zero3); bodyIE_zero];

    [specIE, partIE, c, d, ~, ~, ~, ~] = strread(filesIE{indexIE}, '%s %s %s %s %s %s %s %s',
'delimiter', '_');
    specIE = cell2mat(specIE);
    partIE = cell2mat(partIE);
    c = cell2mat(c);
    d = cell2mat(d);
    sheet = [partIE '_' c '_' d]; % only use desired parts of filename for sheet titles
    excelarrayIE = horzcat(IEtorque, IEangle);
    headerIE = {'IEtorque (N-m)' 'Rotational Angle (degrees)'};
    xlswrite([specIE '_'  'plotIE.xlsx'],headerIE,sheet)
    xlswrite([specIE '_'  'plotIE.xlsx'],excelarrayIE,sheet,'A2')
    clear c d sheet

for i = 2:length(IEtorque)

    % The KM Labview program does not account for sign convention regarding
    % internal/external rotation, so files must be separated based on left
    % and right designation.  For a left leg, external rotation is a positive
    % angle and internal rotation is negative. External rotation is
    % negative and internal is positive for a right leg.

    % START FLEXION ANGLE = 90
    if isempty(cell2mat(strfind(filesIE(indexIE),'_90_'))) == 0
        IEangleflex = IEangle - start_ang90IE;

    if isempty(strfind(specIE,'L')) == 0 || isempty(strfind(specIE,'l')) == 0 % Performs calculations
only on a speciment designated "left"
    if xIE > 1.5
        if IEtorque(i) > 1.5 && IEtorque(i-1) < 1.5 % select data points, ensure slope ~= 0
            angle_intIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangleflex(i)-IEangleflex(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleflex(i-1))};
            bodyposIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < 1.5 && IEtorque(i-1) > 1.5
            angle_intIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangleflex(i-1)-
IEangleflex(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleflex(i))};
            bodyposIE(i,indexIE) = {bodyIE(i)};
        % perform linear interpolation on all points which meet the criteria
        end
    else
        if IEtorque(i) > xIE*limit && IEtorque(i-1) < xIE*limit
            angle_intIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangleflex(i)-IEangleflex(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleflex(i-1))};
            bodyposIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < xIE*limit && IEtorque(i-1) > xIE*limit
```

```
        angle_intIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangleflex(i-1)-
IEangleflex(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleflex(i))};
        bodyposIE(i,indexIE) = {bodyIE(i)};
      end
   end

   if yIE < -1.5
      if IEtorque(i) > -1.5 && IEtorque(i-1) < -1.5
        angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangleflex(i)-IEangleflex(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleflex(i-1))};
        bodynegIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < -1.5 && IEtorque(i-1) > -1.5
        angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangleflex(i-1)-IEangle(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangleflex(i))};
        bodynegIE(i,indexIE) = {bodyIE(i)};
      end
   else
      if IEtorque(i) > yIE*limit && IEtorque(i-1) < yIE*limit
        angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangleflex(i)-IEangleflex(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleflex(i-1))};
        bodynegIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < yIE*limit && IEtorque(i-1) > yIE*limit
        angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangleflex(i-1)-
IEangleflex(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleflex(i))};
        bodynegIE(i,indexIE) = {bodyIE(i)};
      end
   end

   elseif isempty(strfind(specIE,'R')) == 0 || isempty(strfind(specIE,'r')) == 0 % Only perform
calculations on specimens with "right" designation
   if xIE > 1.5
      if IEtorque(i) > 1.5 && IEtorque(i-1) < 1.5 % select data points, ensure slope ~= 0
        angle_extIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangleflex(i)-IEangleflex(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleflex(i-1))};
        bodyposIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < 1.5 && IEtorque(i-1) > 1.5
        angle_extIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangleflex(i-1)-
IEangleflex(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleflex(i))};
        bodyposIE(i,indexIE) = {bodyIE(i)};
      % perform linear interpolation on all points which meet the criteria
      end
   else
      if IEtorque(i) > xIE*limit && IEtorque(i-1) < xIE*limit
        angle_extIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangleflex(i)-IEangleflex(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleflex(i-1))};
        bodyposIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < xIE*limit && IEtorque(i-1) > xIE*limit
        angle_extIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangleflex(i-1)-
IEangleflex(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleflex(i))};
        bodyposIE(i,indexIE) = {bodyIE(i)};
      end
```

```
    end


    if yIE < -1.5
        if IEtorque(i) > -1.5 && IEtorque(i-1) < -1.5
            angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangleflex(i)-IEangleflex(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleflex(i-1))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < -1.5 && IEtorque(i-1) > -1.5
            angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangleflex(i-1)-
IEangleflex(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleflex(i))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        end
    else
        if IEtorque(i) > yIE*limit && IEtorque(i-1) < yIE*limit
            angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangleflex(i)-IEangleflex(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleflex(i-1))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < yIE*limit && IEtorque(i-1) > yIE*limit
            angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangleflex(i-1)-
IEangleflex(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleflex(i))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        end
    end
    end

    % START FLEXION ANGLE = 60
    elseif isempty(cell2mat(strfind(filesIE(indexIE),'_60_'))) == 0
        IEangle60 = IEangle - start_ang60IE;

    if isempty(strfind(specIE,'L')) == 0 || isempty(strfind(specIE,'l')) == 0 % Performs calculations
only on a speciment designated "left"
    if xIE > 1.5
        if IEtorque(i) > 1.5 && IEtorque(i-1) < 1.5 % select data points, ensure slope ~= 0
            angle_intIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangle60(i)-IEangle60(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle60(i-1))};
            bodyposIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < 1.5 && IEtorque(i-1) > 1.5
            angle_intIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangle60(i-1)-IEangle60(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle60(i))};
            bodyposIE(i,indexIE) = {bodyIE(i)};
        % perform linear interpolation on all points which meet the criteria
        end
    else
        if IEtorque(i) > xIE*limit && IEtorque(i-1) < xIE*limit
            angle_intIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangle60(i)-IEangle60(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle60(i-1))};
            bodyposIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < xIE*limit && IEtorque(i-1) > xIE*limit
            angle_intIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangle60(i-1)-IEangle60(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle60(i))};
```

```
            bodyposIE(i,indexIE) = {bodyIE(i)};
        end
    end

    if yIE < -1.5
        if IEtorque(i) > -1.5 && IEtorque(i-1) < -1.5
            angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangle60(i)-IEangle60(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle60(i-1))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < -1.5 && IEtorque(i-1) > -1.5
            angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangle60(i-1)-IEangle60(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle60(i))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        end
    else
        if IEtorque(i) > yIE*limit && IEtorque(i-1) < yIE*limit
            angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangle60(i)-IEangle60(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle60(i-1))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < yIE*limit && IEtorque(i-1) > yIE*limit
            angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangle60(i-1)-IEangle60(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle60(i))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        end
    end

    elseif isempty(strfind(specIE,'R')) == 0 || isempty(strfind(specIE,'r')) == 0 % Only perform
calculations on specimens with "right" designation
    if xIE > 1.5
        if IEtorque(i) > 1.5 && IEtorque(i-1) < 1.5 % select data points, ensure slope ~= 0
            angle_extIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangle60(i)-IEangle60(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle60(i-1))};
            bodyposIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < 1.5 && IEtorque(i-1) > 1.5
            angle_extIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangle60(i-1)-IEangle60(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle60(i))};
            bodyposIE(i,indexIE) = {bodyIE(i)};
        % perform linear interpolation on all points which meet the criteria
        end
    else
        if IEtorque(i) > xIE*limit && IEtorque(i-1) < xIE*limit
            angle_extIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangle60(i)-IEangle60(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle60(i-1))};
            bodyposIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < xIE*limit && IEtorque(i-1) > xIE*limit
            angle_extIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangle60(i-1)-IEangle60(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle60(i))};
            bodyposIE(i,indexIE) = {bodyIE(i)};
        end
    end
```

```
   if yIE < -1.5
      if IEtorque(i) > -1.5 && IEtorque(i-1) < -1.5
         angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangle60(i)-IEangle60(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle60(i-1))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < -1.5 && IEtorque(i-1) > -1.5
         angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangle60(i-1)-IEangle60(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle60(i))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      end
   else
      if IEtorque(i) > yIE*limit && IEtorque(i-1) < yIE*limit
         angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangle60(i)-IEangle60(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle60(i-1))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < yIE*limit && IEtorque(i-1) > yIE*limit
         angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangle60(i-1)-IEangle60(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle60(i))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      end
   end
   end

   %START FLEXION ANGLE = 30
   elseif isempty(cell2mat(strfind(filesIE(indexIE),'_30_'))) == 0
      IEangle30 = IEangle - start_ang30IE;

   if isempty(strfind(specIE,'L')) == 0 || isempty(strfind(specIE,'l')) == 0 % Performs calculations
only on a speciment designated "left"
   if xIE > 1.5
      if IEtorque(i) > 1.5 && IEtorque(i-1) < 1.5 % select data points, ensure slope ~= 0
         angle_intIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangle30(i)-IEangle30(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle30(i-1))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < 1.5 && IEtorque(i-1) > 1.5
         angle_intIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangle30(i-1)-IEangle30(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle30(i))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      % perform linear interpolation on all points which meet the criteria
      end
   else
      if IEtorque(i) > xIE*limit && IEtorque(i-1) < xIE*limit
         angle_intIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangle30(i)-IEangle30(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle30(i-1))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < xIE*limit && IEtorque(i-1) > xIE*limit
         angle_intIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangle30(i-1)-IEangle30(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle30(i))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      end
   end
```

```
   if yIE < -1.5
      if IEtorque(i) > -1.5 && IEtorque(i-1) < -1.5
         angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangle30(i)-IEangle30(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle30(i-1))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < -1.5 && IEtorque(i-1) > -1.5
         angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangle30(i-1)-IEangle30(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle30(i))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      end
   else
      if IEtorque(i) > yIE*limit && IEtorque(i-1) < yIE*limit
         angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangle30(i)-IEangle30(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle30(i-1))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < yIE*limit && IEtorque(i-1) > yIE*limit
         angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangle30(i-1)-IEangle30(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle30(i))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      end
   end

   elseif isempty(strfind(specIE,'R')) == 0 || isempty(strfind(specIE,'r')) == 0 % Only perform
calculations on specimens with "right" designation
   if xIE > 1.5
      if IEtorque(i) > 1.5 && IEtorque(i-1) < 1.5 % select data points, ensure slope ~= 0
         angle_extIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangle30(i)-IEangle30(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle30(i-1))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < 1.5 && IEtorque(i-1) > 1.5
         angle_extIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangle30(i-1)-IEangle30(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle30(i))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      % perform linear interpolation on all points which meet the criteria
      end
   else
      if IEtorque(i) > xIE*limit && IEtorque(i-1) < xIE*limit
         angle_extIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangle30(i)-IEangle30(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle30(i-1))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < xIE*limit && IEtorque(i-1) > xIE*limit
         angle_extIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangle30(i-1)-IEangle30(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle30(i))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      end
   end

   if yIE < -1.5
      if IEtorque(i) > -1.5 && IEtorque(i-1) < -1.5
```

```
        angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangle30(i)-IEangle30(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle30(i-1))};
        bodynegIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < -1.5 && IEtorque(i-1) > -1.5
        angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangle30(i-1)-IEangle30(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle30(i))};
        bodynegIE(i,indexIE) = {bodyIE(i)};
      end
   else
      if IEtorque(i) > yIE*limit && IEtorque(i-1) < yIE*limit
        angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangle30(i)-IEangle30(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangle30(i-1))};
        bodynegIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < yIE*limit && IEtorque(i-1) > yIE*limit
        angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangle30(i-1)-IEangle30(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangle30(i))};
        bodynegIE(i,indexIE) = {bodyIE(i)};
      end
   end
   end

   % START FLEXION ANGLE = EXT
   elseif isempty(cell2mat(strfind(filesIE(indexIE),'_EXT_'))) == 0
        IEangleext = IEangle - start_angextIE;

   if isempty(strfind(specIE,'L')) == 0 || isempty(strfind(specIE,'l')) == 0 % Performs calculations
only on a speciment designated "left"
   if xIE > 1.5
      if IEtorque(i) > 1.5 && IEtorque(i-1) < 1.5 % select data points, ensure slope ~= 0
        angle_intIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangleext(i)-IEangleext(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleext(i-1))};
        bodyposIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < 1.5 && IEtorque(i-1) > 1.5
        angle_intIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangleext(i-1)-IEangleext(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangleext(i))};
        bodyposIE(i,indexIE) = {bodyIE(i)};
      % perform linear interpolation on all points which meet the criteria
      end
   else
      if IEtorque(i) > xIE*limit && IEtorque(i-1) < xIE*limit
        angle_intIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangleext(i)-IEangleext(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleext(i-1))};
        bodyposIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < xIE*limit && IEtorque(i-1) > xIE*limit
        angle_intIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangleext(i-1)-IEangleext(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangleext(i))};
        bodyposIE(i,indexIE) = {bodyIE(i)};
      end
   end

   if yIE < -1.5
```

```
      if IEtorque(i) > -1.5 && IEtorque(i-1) < -1.5
         angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangleext(i)-IEangleext(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleext(i-1))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < -1.5 && IEtorque(i-1) > -1.5
         angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangleext(i-1)-
IEangleext(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleext(i))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      end
   else
      if IEtorque(i) > yIE*limit && IEtorque(i-1) < yIE*limit
         angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangleext(i)-IEangleext(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleext(i-1))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < yIE*limit && IEtorque(i-1) > yIE*limit
         angle_extIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangleext(i-1)-
IEangleext(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleext(i))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
      end
   end

   elseif isempty(strfind(specIE,'R')) == 0 || isempty(strfind(specIE,'r')) == 0 % Only perform
calculations on specimens with "right" designation
   if xIE > 1.5
      if IEtorque(i) > 1.5 && IEtorque(i-1) < 1.5 % select data points, ensure slope ~= 0
         angle_extIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangleext(i)-IEangleext(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleext(i-1))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < 1.5 && IEtorque(i-1) > 1.5
         angle_extIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangleext(i-1)-IEangleext(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangleext(i))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      % perform linear interpolation on all points which meet the criteria
      end
   else
      if IEtorque(i) > xIE*limit && IEtorque(i-1) < xIE*limit
         angle_extIE(i,indexIE) = {((1.5 - IEtorque(i-1))*(IEangleext(i)-IEangleext(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleext(i-1))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      elseif IEtorque(i) < xIE*limit && IEtorque(i-1) > xIE*limit
         angle_extIE(i,indexIE) = {((1.5 - IEtorque(i))*(IEangleext(i-1)-IEangleext(i))/(IEtorque(i-
1)-IEtorque(i)) + IEangleext(i))};
         bodyposIE(i,indexIE) = {bodyIE(i)};
      end
   end

   if yIE < -1.5
      if IEtorque(i) > -1.5 && IEtorque(i-1) < -1.5
         angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangleext(i)-IEangleext(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleext(i-1))};
         bodynegIE(i,indexIE) = {bodyIE(i)};
```

```
        elseif IEtorque(i) < -1.5 && IEtorque(i-1) > -1.5
            angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangleext(i-1)-
IEangleext(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleext(i))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        end
    else
        if IEtorque(i) > yIE*limit && IEtorque(i-1) < yIE*limit
            angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i-1))*(IEangleext(i)-IEangleext(i-
1))/(IEtorque(i)-IEtorque(i-1)) + IEangleext(i-1))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        elseif IEtorque(i) < yIE*limit && IEtorque(i-1) > yIE*limit
            angle_intIE(i,indexIE) = {((-1.5 - IEtorque(i))*(IEangleext(i-1)-
IEangleext(i))/(IEtorque(i-1)-IEtorque(i)) + IEangleext(i))};
            bodynegIE(i,indexIE) = {bodyIE(i)};
        end
    end

    end

    else disp('Please specify left or right designation in filename')
    end

end

end

%     if isempty(find(IEangle == saturationIE_pos, 1)) == 0 || isempty(find(IEangle ==
saturationIE_neg, 1)) == 0 || isempty(find(cell2mat(angle_zeroIE(:,indexIE)) ==
saturationIE_pos, 1))== 0 || isempty(find(cell2mat(angle_zeroIE(:,indexIE)) == saturationIE_neg,
1)) == 0
%         warningIE(indexIE) = {1};
%     else
%         warningIE(indexIE) = {0};
%     end

else disp('No IE files to process')
end
clear xIE yIE  IE ieangle_col ietorque_col bodyie_col
```

Varus/Valgus Calculations

```
% kneetest_VV: subprogram called by the main kneetest_info program to
% calculate varus/valgus angles at 10Nm from excel files loaded by the main
% program.

if ~isempty(filesVV)
for indexVV = 1:length(filesVV) % this loop only applies to files containing VV in the filename

   VV = cell2mat(dataVV(indexVV));

   VVangle = (VV(:,vvangle_col)); % Isolate angle and torque data from each file
   VVangle = removerows(VVangle,isnan(VVangle)); % Remove text, which appears as NaNs,
from data
   vvtorque = (VV(:,vvtorque_col));
   vvtorque = removerows(vvtorque,isnan(vvtorque));
   bodyVV = VV(:,bodyvv_col); % do the same for body weight data
   bodyVV = removerows(bodyVV,isnan(bodyVV));

   %Remove specific values from data - these cause INF to appear in
   %calculations
    VVangle = removerows(VVangle,'ind',vvtorque == 10);
    vvtorque = removerows(vvtorque,'ind',vvtorque == 10);
    VVangle = removerows(VVangle,'ind',vvtorque == -10);
    vvtorque = removerows(vvtorque,'ind',vvtorque == -10);

%      vvtorque = removerows(vvtorque,'ind',vvangle == 0);
%      vvangle = removerows(vvangle,'ind',vvangle == 0);

    vvtorque = removerows(vvtorque,'ind',vvtorque == 0);
    VVangle = removerows(VVangle,'ind',vvtorque == 0);


   [xVV,indmaxVV] = max(vvtorque); % Find minimum and maximum of all torque data for
each file
   [yVV,indminVV] = min(vvtorque);
   MaxVV(indexVV) = {round(xVV*10^3)/10^3}; % Store min and max values in a cell array
   MinVV(indexVV) = {round(yVV*10^3)/10^3};

   vvtorquezero = zeros(indmaxVV,1);
   vvtorquezero2 = zeros(indminVV,1);
   vvanglezero = zeros(indmaxVV,1);
   vvanglezero2 = zeros(indminVV,1);
   bodyVV_zero = zeros(indmaxVV,1);
   bodyVV_zero2 = zeros(indminVV,1);

   for k = 1:length(vvtorque)
      if indmaxVV < indminVV
         if k <= indmaxVV
```

```
            vvtorquezero(k) = vvtorque(k);
            vvanglezero(k) = VVangle(k);
            bodyVV_zero(k) = bodyVV(k);

        elseif k > indmaxVV && k <= indminVV
            vvtorquezero2(k) = vvtorque(k);
            vvanglezero2(k) = VVangle(k);
            bodyVV_zero2(k) = bodyVV(k);
        end

    elseif indmaxVV > indminVV
        if k <= indminVV
            vvtorquezero(k) = vvtorque(k);
            vvanglezero(k) = VVangle(k);
            bodyVV_zero(k) = bodyVV(k);

        elseif k > indminVV && k <= indmaxVV
            vvtorquezero2(k) = vvtorque(k);
            vvanglezero2(k) = VVangle(k);
            bodyVV_zero2(k) = bodyVV(k);

        end
    end
end

%    vvtorquezero2(vvanglezero2 == 0) = [];
%    vvanglezero2(vvanglezero2 == 0) = [];
%    bodyVV_zero2(vvanglezero2 == 0) = [];

[~,startVV] = min(abs(vvanglezero(1)-vvanglezero2));
vvtorquezero3 = zeros(length(vvtorquezero2)-startVV,1);
vvanglezero3 = zeros(length(vvanglezero2)-startVV,1);
bodyVV_zero3 = zeros(length(bodyVV_zero2)-startVV,1);

for j = 1:length(vvtorquezero2)
    if j > startVV
        vvtorquezero3(j) = vvtorquezero2(j);
        vvanglezero3(j) = vvanglezero2(j);
        bodyVV_zero3(j) = bodyVV_zero2(j);
    end
end

vvanglezero3(vvtorquezero3 == 0) = [];
vvtorquezero3(vvtorquezero3 == 0) = [];
bodyVV_zero3(bodyVV_zero3 == 0) = [];

vvtorque = [flipud(vvtorquezero3); vvtorquezero];
VVangle = [flipud(vvanglezero3); vvanglezero];
bodyVV = [flipud(bodyVV_zero3); bodyVV_zero];
```

```
% Save angle vs torque data as excel files with sheets written as filenames
   [specVV, partVV, c, d, ~, ~] = strread(filesVV{indexVV}, '%s %s %s %s %s %s', 'delimiter',
'_');
   specVV = cell2mat(specVV);
   partVV = cell2mat(partVV);
   c = cell2mat(c);
   d = cell2mat(d);
   sheet = [partVV '_' c '_' d]; %only use desired parts of filename for sheet titles
   excelarrayVV = horzcat(vvtorque, VVangle);
   headerVV = {'VVtorque (N-m)' 'VVangle (degrees)'};
   xlswrite([specVV '_'  'plotVV.xlsx'],headerVV,sheet)
   xlswrite([specVV '_'  'plotVV.xlsx'],excelarrayVV,sheet,'A2') %save plot data to excel
spreadsheet, with each file on a diferent sheet
   clear c d sheet

% Interpolate to find angle values near positive and negative 10Nm of
% torque.  Save these values as an array with columns designating different
% files.

for i = 2:length(vvtorque)

   % START FLEXION ANGLE = 90
   if isempty(cell2mat(strfind(filesVV(indexVV),'_90_'))) == 0
      vvangleflex = VVangle - start_ang90VV;

   if xVV > 10
      if vvtorque(i) > 10 && vvtorque(i-1) < 10   % only performs calculation on specific data
         angle_posVV(i,indexVV) = {((10 - vvtorque(i-1))*(vvangleflex(i)-vvangleflex(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangleflex(i-1))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < 10 && vvtorque(i-1) > 10
         angle_posVV(i,indexVV) = {((10 - vvtorque(i))*(vvangleflex(i-1)-
vvangleflex(i))/(vvtorque(i-1)-vvtorque(i)) + vvangleflex(i))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      %perform linear interpolation on all points which meet the criteria
      end
   else
      if vvtorque(i) > xVV*limit && vvtorque(i-1) < xVV*limit
         angle_posVV(i,indexVV) = {((10 - vvtorque(i-1))*(vvangleflex(i)-vvangleflex(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangleflex(i-1))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < xVV*limit && vvtorque(i-1) > xVV*limit
         angle_posVV(i,indexVV) = {((10 - vvtorque(i))*(vvangleflex(i-1)-
vvangleflex(i))/(vvtorque(i-1)-vvtorque(i)) + vvangleflex(i))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      end
   end


   if yVV < -10
      if vvtorque(i) > -10 && vvtorque(i-1) < -10
```

```
        angle_negVV(i,indexVV) = {((-10 - vvtorque(i-1))*(vvangleflex(i)-vvangleflex(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangleflex(i-1))};
        bodynegVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < -10 && vvtorque(i-1) > -10
        angle_negVV(i,indexVV) = {((-10 - vvtorque(i))*(vvangleflex(i-1)-
vvangleflex(i))/(vvtorque(i-1)-vvtorque(i)) + vvangleflex(i))};
        bodynegVV(i,indexVV) = {bodyVV(i)};
      end

   else
      if vvtorque(i) > yVV*limit && vvtorque(i-1) < yVV*limit
        angle_negVV(i,indexVV) = {((-10 - vvtorque(i-1))*(vvangleflex(i)-vvangleflex(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangleflex(i-1))};
        bodynegVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < yVV*limit && vvtorque(i-1) > yVV*limit
        angle_negVV(i,indexVV) = {((-10 - vvtorque(i))*(vvangleflex(i-1)-
vvangleflex(i))/(vvtorque(i-1)-vvtorque(i)) + vvangleflex(i))};
        bodynegVV(i,indexVV) = {bodyVV(i)};
      end
   end



   % START FLEXION ANGLE = 60
   elseif isempty(cell2mat(strfind(filesVV(indexVV),'_60_'))) == 0
      vvangle60 = VVangle - start_ang60VV;

   if xVV > 10
      if vvtorque(i) > 10 && vvtorque(i-1) < 10   % only performs calculation on specific data
        angle_posVV(i,indexVV) = {((10 - vvtorque(i-1))*(vvangle60(i)-vvangle60(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangle60(i-1))};
        bodyposVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < 10 && vvtorque(i-1) > 10
        angle_posVV(i,indexVV) = {((10 - vvtorque(i))*(vvangle60(i-1)-
vvangle60(i))/(vvtorque(i-1)-vvtorque(i)) + vvangle60(i))};
        bodyposVV(i,indexVV) = {bodyVV(i)};
      %perform linear interpolation on all points which meet the criteria
      end
   else
      if vvtorque(i) > xVV*limit && vvtorque(i-1) < xVV*limit
        angle_posVV(i,indexVV) = {((10 - vvtorque(i-1))*(vvangle60(i)-vvangle60(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangle60(i-1))};
        bodyposVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < xVV*limit && vvtorque(i-1) > xVV*limit
        angle_posVV(i,indexVV) = {((10 - vvtorque(i))*(vvangle60(i-1)-
vvangle60(i))/(vvtorque(i-1)-vvtorque(i)) + vvangle60(i))};
        bodyposVV(i,indexVV) = {bodyVV(i)};
      end
   end
```

```
   if yVV < -10
      if vvtorque(i) > -10 && vvtorque(i-1) < -10
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i-1))*(vvangle60(i)-vvangle60(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangle60(i-1))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < -10 && vvtorque(i-1) > -10
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i))*(vvangle60(i-1)-
vvangle60(i))/(vvtorque(i-1)-vvtorque(i)) + vvangle60(i))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      end

   else
      if vvtorque(i) > yVV*limit && vvtorque(i-1) < yVV*limit
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i-1))*(vvangle60(i)-vvangle60(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangle60(i-1))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < yVV*limit && vvtorque(i-1) > yVV*limit
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i))*(vvangle60(i-1)-
vvangle60(i))/(vvtorque(i-1)-vvtorque(i)) + vvangle60(i))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      end
   end


   % START FLEXION ANGLE = 30
   elseif isempty(cell2mat(strfind(filesVV(indexVV),'_30_'))) == 0
      vvangle30 = VVangle - start_ang30VV;

   if xVV > 10
      if vvtorque(i) > 10 && vvtorque(i-1) < 10   % only performs calculation on specific data
         angle_posVV(i,indexVV) = {((10 - vvtorque(i-1))*(vvangle30(i)-vvangle30(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangle30(i-1))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < 10 && vvtorque(i-1) > 10
         angle_posVV(i,indexVV) = {((10 - vvtorque(i))*(vvangle30(i-1)-
vvangle30(i))/(vvtorque(i-1)-vvtorque(i)) + vvangle30(i))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      %perform linear interpolation on all points which meet the criteria
      end
   else
      if vvtorque(i) > xVV*limit && vvtorque(i-1) < xVV*limit
         angle_posVV(i,indexVV) = {((10 - vvtorque(i-1))*(vvangle30(i)-vvangle30(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangle30(i-1))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < xVV*limit && vvtorque(i-1) > xVV*limit
         angle_posVV(i,indexVV) = {((10 - vvtorque(i))*(vvangle30(i-1)-
vvangle30(i))/(vvtorque(i-1)-vvtorque(i)) + vvangle30(i))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      end
   end
```

```
   if yVV < -10
      if vvtorque(i) > -10 && vvtorque(i-1) < -10
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i-1))*(vvangle30(i)-vvangle30(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangle30(i-1))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < -10 && vvtorque(i-1) > -10
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i))*(vvangle30(i-1)-
vvangle30(i))/(vvtorque(i-1)-vvtorque(i)) + vvangle30(i))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      end

   else
      if vvtorque(i) > yVV*limit && vvtorque(i-1) < yVV*limit
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i-1))*(vvangle30(i)-vvangle30(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangle30(i-1))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < yVV*limit && vvtorque(i-1) > yVV*limit
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i))*(vvangle30(i-1)-
vvangle30(i))/(vvtorque(i-1)-vvtorque(i)) + vvangle30(i))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      end
   end


   % START FLEXION ANGLE = EXT
   elseif isempty(cell2mat(strfind(filesVV(indexVV),'_EXT_'))) == 0
      vvangleext = VVangle - start_angextVV;

   if xVV > 10
      if vvtorque(i) > 10 && vvtorque(i-1) < 10   % only performs calculation on specific data
         angle_posVV(i,indexVV) = {((10 - vvtorque(i-1))*(vvangleext(i)-vvangleext(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangleext(i-1))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < 10 && vvtorque(i-1) > 10
         angle_posVV(i,indexVV) = {((10 - vvtorque(i))*(vvangleext(i-1)-
vvangleext(i))/(vvtorque(i-1)-vvtorque(i)) + vvangleext(i))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      %perform linear interpolation on all points which meet the criteria
      end
   else
      if vvtorque(i) > xVV*limit && vvtorque(i-1) < xVV*limit
         angle_posVV(i,indexVV) = {((10 - vvtorque(i-1))*(vvangleext(i)-vvangleext(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangleext(i-1))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < xVV*limit && vvtorque(i-1) > xVV*limit
         angle_posVV(i,indexVV) = {((10 - vvtorque(i))*(vvangleext(i-1)-
vvangleext(i))/(vvtorque(i-1)-vvtorque(i)) + vvangleext(i))};
         bodyposVV(i,indexVV) = {bodyVV(i)};
      end
   end
```

```
   if yVV < -10
      if vvtorque(i) > -10 && vvtorque(i-1) < -10
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i-1))*(vvangleext(i)-vvangleext(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangleext(i-1))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < -10 && vvtorque(i-1) > -10
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i))*(vvangleext(i-1)-
vvangleext(i))/(vvtorque(i-1)-vvtorque(i)) + vvangleext(i))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      end

   else
      if vvtorque(i) > yVV*limit && vvtorque(i-1) < yVV*limit
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i-1))*(vvangleext(i)-vvangleext(i-
1))/(vvtorque(i)-vvtorque(i-1)) + vvangleext(i-1))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      elseif vvtorque(i) < yVV*limit && vvtorque(i-1) > yVV*limit
         angle_negVV(i,indexVV) = {((-10 - vvtorque(i))*(vvangleext(i-1)-
vvangleext(i))/(vvtorque(i-1)-vvtorque(i)) + vvangleext(i))};
         bodynegVV(i,indexVV) = {bodyVV(i)};
      end
   end

   end

end


end
else disp('No VV files to process') % Display a message if no files labeled 'VV' are in the current
directory
end

clear  xVV yVV excelarrayVV VV vvangle_col vvtorque_col bodyvv_col
```

VITA

Erik Lane Woodard was born in Lafayette, Louisiana in 1990. After completing high school, he went on the receive a B.S. in Biomedical Engineering with a concentration in Mechanical Engineering from Louisiana Tech University in Ruston, Louisiana in May 2012. After graduation, he was accepted into the Joint Program in Biomedical Engineering at the University of Tennessee Health Science Center and the University of Memphis as an M.S. candidate. During this time, Erik was a student of Dr. William Mihalko and researched joint mechanics and implants, with a focus on knee mechanics. In May 2014, Erik earned a Master of Science in Biomedical Engineering for investigating the causes of wear in total knee arthroplasty.