Evaluating an Aggregate Method for Estimating Intra-Limb Proportions for Commingled Remains

Carol Doudou Cao

2023-06-08

# Install and load necessary packages

install.packages("tidyverse", repos = "https://cran.r-project.org/")

## Installing package into 'C:/Users/carol/Documents/R/win-library/4.1'  
## (as 'lib' is unspecified)

## package 'tidyverse' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\carol\AppData\Local\Temp\RtmpUXyJjV\downloaded\_packages

install.packages("viridis", repos = "https://cran.r-project.org/")

## Installing package into 'C:/Users/carol/Documents/R/win-library/4.1'  
## (as 'lib' is unspecified)

##   
## There is a binary version available but the source version is later:  
## binary source needs\_compilation  
## viridis 0.6.2 0.6.3 FALSE

## installing the source package 'viridis'

install.packages("terra", repos = "https://cran.r-project.org/")

## Installing package into 'C:/Users/carol/Documents/R/win-library/4.1'  
## (as 'lib' is unspecified)

##   
## There is a binary version available but the source version is later:  
## binary source needs\_compilation  
## terra 1.7-23 1.7-29 TRUE  
##   
## Binaries will be installed  
## package 'terra' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'terra'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying  
## C:\Users\carol\Documents\R\win-library\4.1\00LOCK\terra\libs\x64\terra.dll to  
## C:\Users\carol\Documents\R\win-library\4.1\terra\libs\x64\terra.dll: Permission  
## denied

## Warning: restored 'terra'

##   
## The downloaded binary packages are in  
## C:\Users\carol\AppData\Local\Temp\RtmpUXyJjV\downloaded\_packages

install.packages("gridExtra", repos = "https://cran.r-project.org/")

## Installing package into 'C:/Users/carol/Documents/R/win-library/4.1'  
## (as 'lib' is unspecified)

## package 'gridExtra' successfully unpacked and MD5 sums checked  
##   
## The downloaded binary packages are in  
## C:\Users\carol\AppData\Local\Temp\RtmpUXyJjV\downloaded\_packages

library(viridis)

## Loading required package: viridisLite

library(tidyverse)

## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --  
## v dplyr 1.1.2 v readr 2.1.4  
## v forcats 1.0.0 v stringr 1.5.0  
## v ggplot2 3.4.1 v tibble 3.2.1  
## v lubridate 1.9.2 v tidyr 1.3.0  
## v purrr 1.0.1

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()  
## i Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force all conflicts to become errors

library(terra)

## terra 1.7.23  
##   
## Attaching package: 'terra'  
##   
## The following object is masked from 'package:tidyr':  
##   
## extract

library(gridExtra)

##   
## Attaching package: 'gridExtra'  
##   
## The following object is masked from 'package:dplyr':  
##   
## combine

library(scales)

##   
## Attaching package: 'scales'  
##   
## The following object is masked from 'package:terra':  
##   
## rescale  
##   
## The following object is masked from 'package:purrr':  
##   
## discard  
##   
## The following object is masked from 'package:readr':  
##   
## col\_factor  
##   
## The following object is masked from 'package:viridis':  
##   
## viridis\_pal

# 1. Prepare the data

## 1.1. Read the Goldman and EU datasets

# Goldman dataset  
Goldman <- read.csv("D:/DATA SET/Data processing/AggregatedCI&BI/simulation\_dataset/Goldman.csv")  
  
# EU\_Ruff  
EU <- read.csv("D:/DATA SET/Data processing/AggregatedCI&BI/simulation\_dataset/Ruff-European.csv")

## 1.2. Select the columns of interest (Assuming datasets have been loaded)

# Select limb lengths in Goldman dataset  
Goldman\_selected\_columns <- Goldman[, c("ID", "Sex", "NOTE", "LHML", "RHML", "LRML", "RRML", "LFBL", "RFBL", "LTML", "RTML")]  
  
# Select limb lengths in EU\_Ruff dataset  
EU\_selected\_columns <- EU[, c("SITE", "SPECIMEN\_No", "SEX", "DATE.RANGE", "YEARS.BP", "FBICLN", "TMAXLN", "HRMAXLN", "HLMAXLN", "RRMAXLN", "RLMAXLN")]

## 1.3. Calculate final length of the humerus (HML), radius (RML),femur (FBL) and tibia (TML)

—Because in some limbs, the lengths of both left and right sides are reported, the final length of a limb is the averge of two sides-

# --- Arm ---  
## Goldman  
### Average the maximum length of the left and right humerus   
Goldman\_selected\_columns$finalHML <- rowMeans(Goldman\_selected\_columns[, c("LHML", "RHML")], na.rm = TRUE)  
  
### Average the maximum length of the left and right radius   
Goldman\_selected\_columns$finalRML <- rowMeans(Goldman\_selected\_columns[, c("LRML", "RRML")], na.rm = TRUE)  
  
## EU\_Ruff  
### Average the maximum length of the left and right humerus   
EU\_selected\_columns$finalHML <- rowMeans(EU\_selected\_columns[, c("HRMAXLN", "HLMAXLN")], na.rm = TRUE)  
  
### Average the maximum length of the left and right radius   
EU\_selected\_columns$finalRML <- rowMeans(EU\_selected\_columns[, c("RRMAXLN", "RLMAXLN")], na.rm = TRUE)  
  
# --- Leg ---  
  
## Goldman  
### Average the bicondylar length of the left and right femur  
Goldman\_selected\_columns$finalFBL <- rowMeans(Goldman\_selected\_columns[, c("LFBL", "RFBL")], na.rm = TRUE)  
  
### Average the maximum length of the left and right tibia   
Goldman\_selected\_columns$finalTML <- rowMeans(Goldman\_selected\_columns[, c("LTML", "RTML")], na.rm = TRUE)

## 1.4. Drop rows with missing values

# --- Arm ---  
Goldman\_filtered\_arm <- Goldman\_selected\_columns[complete.cases(Goldman\_selected\_columns$finalHML, Goldman\_selected\_columns$finalRML), ]  
  
## EU\_Ruff  
EU\_filtered\_arm <- EU\_selected\_columns[complete.cases(EU\_selected\_columns$finalHML, EU\_selected\_columns$finalRML), ]  
  
# --- Leg ---  
## Goldman   
Goldman\_filtered\_leg <- Goldman\_selected\_columns[complete.cases(Goldman\_selected\_columns$finalFBL, Goldman\_selected\_columns$finalTML), ]  
## EU\_Ruff  
EU\_filtered\_leg <- EU\_selected\_columns[complete.cases(EU\_selected\_columns$TMAXLN, EU\_selected\_columns$FBICLN), ]

## 1.5. Filter out small-size groups (no.individuals < 5)

# Count number of individuals per NOTE (in Goldman dataset) and SITE group (in EU\_Ruff set)  
# Arm  
no\_group\_GM\_arm <- table(Goldman\_filtered\_arm$NOTE)  
no\_group\_EU\_arm <- table(EU\_filtered\_arm$SITE)  
  
# Leg  
no\_group\_GM\_leg <- table(Goldman\_filtered\_leg$NOTE)  
no\_group\_EU\_leg <- table(EU\_filtered\_leg$SITE)  
  
# Filter out small-size groups (no.individuals < 5)  
## Arm  
### Goldman  
Goldman\_filtered\_final\_arm <- Goldman\_filtered\_arm[Goldman\_filtered\_arm$NOTE %in% names(no\_group\_GM\_arm[no\_group\_GM\_arm >= 5]), ]  
  
### EU\_Ruff  
EU\_filtered\_final\_arm <- EU\_filtered\_arm[EU\_filtered\_arm$SITE %in% names(no\_group\_EU\_arm[no\_group\_EU\_arm >= 5]), ]  
  
## Leg  
### Goldman  
Goldman\_filtered\_final\_leg <- Goldman\_filtered\_leg[Goldman\_filtered\_leg$NOTE %in% names(no\_group\_GM\_leg[no\_group\_GM\_leg >= 5]), ]  
  
EU\_filtered\_final\_leg <- EU\_filtered\_leg[EU\_filtered\_leg$SITE %in% names(no\_group\_EU\_leg[no\_group\_EU\_leg >= 5]), ]

## 1.6. Calculate the Conventional Brachial and Crural indices

# Brachial index  
## Goldman  
Goldman\_filtered\_final\_arm$ConventionalBI <- Goldman\_filtered\_final\_arm$finalRML/   
Goldman\_filtered\_final\_arm$finalHML  
  
## EU\_Ruff  
EU\_filtered\_final\_arm$ConventionalBI <- EU\_filtered\_final\_arm$finalRML/   
EU\_filtered\_final\_arm$finalHML  
  
# Crural index  
## Goldman  
Goldman\_filtered\_final\_leg$ConventionalCI <- Goldman\_filtered\_final\_leg$finalTML /   
Goldman\_filtered\_final\_leg$finalFBL  
  
# Rename columns  
names(EU\_filtered\_final\_leg)[names(EU\_filtered\_final\_leg) == "FBICLN"] <- "finalFBL"  
names(EU\_filtered\_final\_leg)[names(EU\_filtered\_final\_leg) == "TMAXLN"] <- "finalTML"  
  
# Calculate TrueCI  
EU\_filtered\_final\_leg$ConventionalCI <- EU\_filtered\_final\_leg$finalTML / EU\_filtered\_final\_leg$finalFBL

# 2. Create the subset of the arm and leg

## 2.1. Data cleaning and transformation operations on the Sex column of two datasets

# Convert missing data or uncertain sex values to NA in arm dataset  
Goldman\_filtered\_final\_arm$Sex <- ifelse(Goldman\_filtered\_final\_arm$Sex %in% c("0?", "1?"), NA, Goldman\_filtered\_final\_arm$Sex)  
  
# Convert 0 and 1 values to 1 and 2, respectively, in arm dataset  
Goldman\_filtered\_final\_arm$Sex <- ifelse(Goldman\_filtered\_final\_arm$Sex == 1, 2, Goldman\_filtered\_final\_arm$Sex)  
Goldman\_filtered\_final\_arm$Sex <- ifelse(Goldman\_filtered\_final\_arm$Sex == 0, 1, Goldman\_filtered\_final\_arm$Sex)  
  
# Convert missing data or uncertain sex values to NA in leg dataset  
Goldman\_filtered\_final\_leg$Sex <- ifelse(Goldman\_filtered\_final\_leg$Sex %in% c("0?", "1?"), NA, Goldman\_filtered\_final\_leg$Sex)  
  
# Convert 0 and 1 values to 1 and 2, respectively, in leg dataset  
Goldman\_filtered\_final\_leg$Sex <- ifelse(Goldman\_filtered\_final\_leg$Sex == 1, 2, Goldman\_filtered\_final\_leg$Sex)  
Goldman\_filtered\_final\_leg$Sex <- ifelse(Goldman\_filtered\_final\_leg$Sex == 0, 1, Goldman\_filtered\_final\_leg$Sex)

## 2.2. Create the subset of the arm and leg

Armset <- rbind(transform(  
 EU\_filtered\_final\_arm[, c("SITE", "SEX", "SPECIMEN\_No", "finalHML", "finalRML", "ConventionalBI")],  
 ID = SPECIMEN\_No,  
 Sex = SEX,  
 source = "EU"  
 )[, c("SITE", "ID", "Sex", "finalHML", "finalRML", "ConventionalBI")]  
,  
transform(  
 Goldman\_filtered\_final\_arm[, c("NOTE", "ID", "Sex", "finalHML", "finalRML", "ConventionalBI")],  
 SITE = NOTE,  
 source = "Goldman"  
 )[, c("SITE", "ID", "Sex", "finalHML", "finalRML", "ConventionalBI")]  
   
)  
  
# Legset  
Legset <- rbind(  
 transform(  
 EU\_filtered\_final\_leg[, c("SITE", "SEX", "SPECIMEN\_No", "finalFBL", "finalTML", "ConventionalCI")],  
 ID = SPECIMEN\_No,  
 Sex = SEX,  
 source = "EU"  
 )[, c("SITE", "Sex", "ID", "finalFBL", "finalTML", "ConventionalCI")],  
 transform(  
 Goldman\_filtered\_final\_leg[, c("NOTE", "ID", "Sex", "finalTML", "finalFBL", "ConventionalCI")],  
 SITE = NOTE,  
 source = "Goldman",  
 Sex = as.numeric(Sex)  
 )[, c("SITE", "Sex", "ID", "finalFBL", "finalTML", "ConventionalCI")]  
)

# 3. Visualise the limb lengths and indices between males and females

## 3.1. Counting males and females

# Counting males and females in Armset dataset  
Armset\_count\_male <- sum(Armset$Sex == 1, na.rm = TRUE)  
Armset\_count\_female <- sum(Armset$Sex == 2, na.rm = TRUE)  
  
# Counting males and females in Legset dataset  
Legset\_count\_male <- sum(Legset$Sex == 1, na.rm = TRUE)  
Legset\_count\_female <- sum(Legset$Sex == 2, na.rm = TRUE)  
  
# Print the counts  
cat("Count of males in Armset:", Armset\_count\_male, "\n")

## Count of males in Armset: 1513

cat("Count of females in Armset:", Armset\_count\_female, "\n")

## Count of females in Armset: 1019

cat("Count of males in Legset:", Legset\_count\_male, "\n")

## Count of males in Legset: 1667

cat("Count of females in Legset:", Legset\_count\_female, "\n")

## Count of females in Legset: 1143

## 3.2. Plot the lengths and indices of males and females

### 3.2.1 Combine the data

# Combine the data of limb lengths  
HML\_male <- Armset$finalHML[Armset$Sex == 1]  
HML\_female <- Armset$finalHML[Armset$Sex == 2]  
HML\_both <- Armset$finalHML  
RML\_male <- Armset$finalRML[Armset$Sex == 1]  
RML\_female <- Armset$finalRML[Armset$Sex == 2]  
RML\_both <- Armset$finalRML  
FBL\_male <- Legset$finalFBL[Legset$Sex == 1]  
FBL\_female <- Legset$finalFBL[Legset$Sex == 2]  
FBL\_both <- Legset$finalFBL  
TML\_male <- Legset$finalTML[Legset$Sex == 1]  
TML\_female <- Legset$finalTML[Legset$Sex == 2]  
TML\_both <- Legset$finalTML  
  
# Combine the data of index  
BI\_male <- Armset$ConventionalBI[Armset$Sex == 1]  
BI\_female <- Armset$ConventionalBI[Armset$Sex == 2]  
BI\_both <- Armset$ConventionalBI  
CI\_male <- Legset$ConventionalCI[Legset$Sex == 1]  
CI\_female <- Legset$ConventionalCI[Legset$Sex == 2]  
CI\_both <- Legset$ConventionalCI

# Create a data frame for HML  
df\_HML <- data.frame(values = c(HML\_male, HML\_female),  
 group = rep(c("HML - Male", "HML - Female"), c(length(HML\_male), length(HML\_female))))  
  
# Create the histogram for HML   
p1 <- ggplot(df\_HML, aes(x = values, fill = group)) +   
 geom\_histogram(binwidth = 10, alpha = 0.6, position = "identity") +  
 scale\_fill\_manual(values = c("#d45087","skyblue" )) +  
 labs(title = "", x = "HML", y = "Frequency", fill = "Sex") +  
 theme\_classic()+  
 theme(legend.title = element\_text(face = "bold", size = 12),  
 legend.text = element\_text(face = "bold", size = 12),  
 axis.title = element\_text(face = "bold", size = 14),  
 axis.text = element\_text(face = "bold", size = 12))  
  
  
# Create a data frame for RML  
df\_RML <- data.frame(values = c(RML\_male, RML\_female),  
 group = rep(c("RML - Male", "RML - Female"), c(length(RML\_male), length(RML\_female))))  
  
# Create the histogram for RML  
p2 <- ggplot(df\_RML, aes(x = values, fill = group)) +   
 geom\_histogram(binwidth = 10, alpha = 0.6, position = "identity") +  
 scale\_fill\_manual(values = c( "#d45087","skyblue" )) +  
 labs(title = "", x = "RML", y = "Frequency", fill = "Sex") +  
 theme\_classic()+  
 theme(legend.title = element\_text(face = "bold", size = 12),  
 legend.text = element\_text(face = "bold", size = 12),  
 axis.title = element\_text(face = "bold", size = 14),  
 axis.text = element\_text(face = "bold", size = 12))  
  
  
# Create a data frame for FBL  
df\_FBL <- data.frame(values = c(FBL\_male, FBL\_female),  
 group = rep(c("FBL - Male", "FBL - Female"), c(length(FBL\_male), length(FBL\_female))))  
  
# Create the histogram for FBL  
p3 <- ggplot(df\_FBL, aes(x = values, fill = group)) +   
 geom\_histogram(binwidth = 10, alpha = 0.6, position = "identity") +  
 scale\_fill\_manual(values = c("#a05195", "blue")) +  
 labs(title = "", x = "FBL", y = "Frequency", fill = "Sex") +  
 theme\_classic()+  
 theme(legend.title = element\_text(face = "bold", size = 12),  
 legend.text = element\_text(face = "bold", size = 12),  
 axis.title = element\_text(face = "bold", size = 14),  
 axis.text = element\_text(face = "bold", size = 12))  
  
# Create a data frame for TML  
df\_TML <- data.frame(values = c(TML\_male, TML\_female),  
 group = rep(c("TML - Male", "TML - Female"), c(length(TML\_male), length(TML\_female))))  
  
# Create a data frame for TML  
df\_TML <- data.frame(values = c(TML\_male, TML\_female),  
 group = rep(c("TML - Male", "TML - Female"), c(length(TML\_male), length(TML\_female))))  
  
# Create the histogram for RML  
p4 <- ggplot(df\_TML, aes(x = values, fill = group)) +   
 geom\_histogram(binwidth = 10, alpha = 0.6, position = "identity") +  
 scale\_fill\_manual(values = c( "#a05195", "blue")) +  
 labs(title = "", x = "TML", y = "Frequency", fill = "Sex") +  
 theme\_classic()+  
 theme(legend.title = element\_text(face = "bold", size = 12),  
 legend.text = element\_text(face = "bold", size = 12),  
 axis.title = element\_text(face = "bold", size = 14),  
 axis.text = element\_text(face = "bold", size = 12))  
  
# Create a data frame for BI  
df\_BI <- subset(data.frame(values = c(BI\_male, BI\_female),  
 group = rep(c("BIt - Male", "BIt - Female"), c(length(BI\_male), length(BI\_female)))),  
 values > 0.6 & values < 0.99)  
  
# Create the histogram for BIt   
  
p5 <- ggplot(df\_BI, aes(x = values, fill = group)) +   
 geom\_histogram(binwidth = 0.01, alpha = 0.6, position = "identity") +  
 scale\_fill\_manual(values = c("#d45087", "skyblue")) +  
 labs(title = "", x = "BIc", y = "Frequency", fill = "Sex") +  
 theme\_classic() +  
 theme(  
 legend.title = element\_text(face = "bold", size = 14),  
 legend.text = element\_text(face = "bold", size = 14),  
 axis.title = element\_text(face = "bold", size = 18),  
 axis.text = element\_text(face = "bold", size = 14)) +  
 scale\_x\_continuous(breaks = seq(0, 1, by = 0.05))  
  
# Create a data frame for CI  
df\_CI <- subset(data.frame(values = c(CI\_male, CI\_female),  
 group = rep(c("CIt - Male", "CIt - Female"), c(length(CI\_male), length(CI\_female)))), values < 1)  
  
# Create the histogram for CIt  
p6 <- ggplot(df\_CI, aes(x = values, fill = group)) +   
 geom\_histogram(binwidth = 0.01, alpha = 0.6, position = "identity") +  
 scale\_fill\_manual(values = c( "#a05195", "blue")) +  
 labs(title = "", x = "CIc", y = "Frequency", fill = "Sex") +  
 theme\_classic()+  
 theme(legend.title = element\_text(face = "bold", size = 14),  
 legend.text = element\_text(face = "bold", size = 14),  
 axis.title = element\_text(face = "bold", size = 18),  
 axis.text = element\_text(face = "bold", size = 14))  
  
# Create the combined plot  
combined\_histogram\_limblength <- grid.arrange(p1, p2, p3, p4, ncol = 2,nrow = 2)

## Warning: Removed 18 rows containing non-finite values (`stat\_bin()`).  
## Removed 18 rows containing non-finite values (`stat\_bin()`).

## Warning: Removed 20 rows containing non-finite values (`stat\_bin()`).  
## Removed 20 rows containing non-finite values (`stat\_bin()`).

A picture containing diagram

Description automatically generated

combined\_histogram\_indices <- grid.arrange(p5, p6, ncol = 2)

A picture containing diagram, screenshot, plot, line

Description automatically generated

# 4. Summarise the limb lengths and indices

## 4.1. Summarise the limb lengths

### 4.1.1. Summarise the limb lengths (combined)

# Humerus  
Mean\_HML <- aggregate(Armset$finalHML, by = list(Armset$SITE), FUN = mean)  
colnames(Mean\_HML) <- c("SITE", "mean\_finalHML")  
  
summary\_HML <- data.frame(  
 mean\_mean\_finalHML = mean(Mean\_HML$mean\_finalHML),  
 max\_mean\_finalHML = max(Mean\_HML$mean\_finalHML),  
 min\_mean\_finalHML = min(Mean\_HML$mean\_finalHML),  
 sd\_mean\_finalHML = sd(Mean\_HML$mean\_finalHML)  
)  
  
# Radius  
Mean\_RML <- aggregate(Armset$finalRML, by = list(Armset$SITE), FUN = mean)  
colnames(Mean\_RML) <- c("SITE", "mean\_finalRML")  
  
summary\_RML <- data.frame(  
 mean\_mean\_finalRML = mean(Mean\_RML$mean\_finalRML),  
 max\_mean\_finalRML = max(Mean\_RML$mean\_finalRML),  
 min\_mean\_finalRML = min(Mean\_RML$mean\_finalRML),  
 sd\_mean\_finalRML = sd(Mean\_RML$mean\_finalRML)  
)  
  
# Femur  
Mean\_FBL <- aggregate(Legset$finalFBL, by = list(Legset$SITE), FUN = mean)  
colnames(Mean\_FBL) <- c("SITE", "mean\_finalFBL")  
  
summary\_FBL <- data.frame(  
 mean\_mean\_finalFBL = mean(Mean\_FBL$mean\_finalFBL),  
 max\_mean\_finalFBL = max(Mean\_FBL$mean\_finalFBL),  
 min\_mean\_finalFBL = min(Mean\_FBL$mean\_finalFBL),  
 sd\_mean\_finalFBL = sd(Mean\_FBL$mean\_finalFBL)  
)  
  
# Tibia  
Mean\_TML <- aggregate(Legset$finalTML, by = list(Legset$SITE), FUN = mean)  
colnames(Mean\_TML) <- c("SITE", "mean\_finalTML")  
  
summary\_TML <- data.frame(  
 mean\_mean\_finalTML = mean(Mean\_TML$mean\_finalTML),  
 max\_mean\_finalTML = max(Mean\_TML$mean\_finalTML),  
 min\_mean\_finalTML = min(Mean\_TML$mean\_finalTML),  
 sd\_mean\_finalTML = sd(Mean\_TML$mean\_finalTML)  
)  
  
# Print summary statistics  
print(summary\_HML)

## mean\_mean\_finalHML max\_mean\_finalHML min\_mean\_finalHML sd\_mean\_finalHML  
## 1 308.7287 344.6 271.2679 13.23827

print(summary\_RML)

## mean\_mean\_finalRML max\_mean\_finalRML min\_mean\_finalRML sd\_mean\_finalRML  
## 1 234.4692 259.9 209.2976 10.10622

print(summary\_FBL)

## mean\_mean\_finalFBL max\_mean\_finalFBL min\_mean\_finalFBL sd\_mean\_finalFBL  
## 1 426.6817 467.7273 383.7941 16.51014

print(summary\_TML)

## mean\_mean\_finalTML max\_mean\_finalTML min\_mean\_finalTML sd\_mean\_finalTML  
## 1 356.1171 399.7273 314.1429 14.91811

### 4.1.2. Summarise the limb lengths (by sex)

# Humerus  
## Male  
Mean\_HML\_male <- aggregate(finalHML ~ SITE, data = Armset[Armset$Sex == 1, ], FUN = mean)  
colnames(Mean\_HML\_male) <- c("SITE", "mean\_finalHML\_male")  
  
summary\_HML\_male <- data.frame(  
 mean\_mean\_finalHML\_male = mean(Mean\_HML\_male$mean\_finalHML\_male),  
 max\_mean\_finalHML\_male = max(Mean\_HML\_male$mean\_finalHML\_male),  
 min\_mean\_finalHML\_male = min(Mean\_HML\_male$mean\_finalHML\_male),  
 sd\_mean\_finalHML\_male = sd(Mean\_HML\_male$mean\_finalHML\_male)  
)  
  
## Female  
Mean\_HML\_female <- aggregate(finalHML ~ SITE, data = Armset[Armset$Sex == 2, ], FUN = mean)  
colnames(Mean\_HML\_female) <- c("SITE", "mean\_finalHML\_female")  
  
summary\_HML\_female <- data.frame(  
 mean\_mean\_finalHML\_female = mean(Mean\_HML\_female$mean\_finalHML\_female),  
 max\_mean\_finalHML\_female = max(Mean\_HML\_female$mean\_finalHML\_female),  
 min\_mean\_finalHML\_female = min(Mean\_HML\_female$mean\_finalHML\_female),  
 sd\_mean\_finalHML\_female = sd(Mean\_HML\_female$mean\_finalHML\_female)  
)  
  
# Radius  
## Male  
Mean\_RML\_male <- aggregate(finalRML ~ SITE, data = Armset[Armset$Sex == 1, ], FUN = mean)  
colnames(Mean\_RML\_male) <- c("SITE", "mean\_finalRML\_male")  
  
summary\_RML\_male <- data.frame(  
 mean\_mean\_finalRML\_male = mean(Mean\_RML\_male$mean\_finalRML\_male),  
 max\_mean\_finalRML\_male = max(Mean\_RML\_male$mean\_finalRML\_male),  
 min\_mean\_finalRML\_male = min(Mean\_RML\_male$mean\_finalRML\_male),  
 sd\_mean\_finalRML\_male = sd(Mean\_RML\_male$mean\_finalRML\_male)  
)  
  
## Female  
Mean\_RML\_female <- aggregate(finalRML ~ SITE, data = Armset[Armset$Sex == 2, ], FUN = mean)  
colnames(Mean\_RML\_female) <- c("SITE", "mean\_finalRML\_female")  
  
summary\_RML\_female <- data.frame(  
 mean\_mean\_finalRML\_female = mean(Mean\_RML\_female$mean\_finalRML\_female),  
 max\_mean\_finalRML\_female = max(Mean\_RML\_female$mean\_finalRML\_female),  
 min\_mean\_finalRML\_female = min(Mean\_RML\_female$mean\_finalRML\_female),  
 sd\_mean\_finalRML\_female = sd(Mean\_RML\_female$mean\_finalRML\_female)  
)  
  
# Femur  
## Male  
Mean\_FBL\_male <- aggregate(finalFBL ~ SITE, data = Legset[Legset$Sex == 1, ], FUN = mean)  
colnames(Mean\_FBL\_male) <- c("SITE", "mean\_finalFBL\_male")  
  
summary\_FBL\_male <- data.frame(  
 mean\_mean\_finalFBL\_male = mean(Mean\_FBL\_male$mean\_finalFBL\_male),  
 max\_mean\_finalFBL\_male = max(Mean\_FBL\_male$mean\_finalFBL\_male),  
 min\_mean\_finalFBL\_male = min(Mean\_FBL\_male$mean\_finalFBL\_male),  
 sd\_mean\_finalFBL\_male = sd(Mean\_FBL\_male$mean\_finalFBL\_male)  
)  
  
## Female  
Mean\_FBL\_female <- aggregate(finalFBL ~ SITE, data = Legset[Legset$Sex == 2, ], FUN = mean)  
colnames(Mean\_FBL\_female) <- c("SITE", "mean\_finalFBL\_female")  
  
summary\_FBL\_female <- data.frame(  
 mean\_mean\_finalFBL\_female = mean(Mean\_FBL\_female$mean\_finalFBL\_female),  
 max\_mean\_finalFBL\_female = max(Mean\_FBL\_female$mean\_finalFBL\_female),  
 min\_mean\_finalFBL\_female = min(Mean\_FBL\_female$mean\_finalFBL\_female),  
 sd\_mean\_finalFBL\_female = sd(Mean\_FBL\_female$mean\_finalFBL\_female)  
)  
  
# Tibia  
## Male  
Mean\_TML\_male <- aggregate(finalTML ~ SITE, data = Legset[Legset$Sex == 1, ], FUN = mean)  
colnames(Mean\_TML\_male) <- c("SITE", "mean\_finalTML\_male")  
  
summary\_TML\_male <- data.frame(  
 mean\_mean\_finalTML\_male = mean(Mean\_TML\_male$mean\_finalTML\_male),  
 max\_mean\_finalTML\_male = max(Mean\_TML\_male$mean\_finalTML\_male),  
 min\_mean\_finalTML\_male = min(Mean\_TML\_male$mean\_finalTML\_male),  
 sd\_mean\_finalTML\_male = sd(Mean\_TML\_male$mean\_finalTML\_male)  
)  
  
## Female  
Mean\_TML\_female <- aggregate(finalTML ~ SITE, data = Legset[Legset$Sex == 2, ], FUN = mean)  
colnames(Mean\_TML\_female) <- c("SITE", "mean\_finalTML\_female")  
  
summary\_TML\_female <- data.frame(  
 mean\_mean\_finalTML\_female = mean(Mean\_TML\_female$mean\_finalTML\_female),  
 max\_mean\_finalTML\_female = max(Mean\_TML\_female$mean\_finalTML\_female),  
 min\_mean\_finalTML\_female = min(Mean\_TML\_female$mean\_finalTML\_female),  
 sd\_mean\_finalTML\_female = sd(Mean\_TML\_female$mean\_finalTML\_female)  
)  
  
# Print summary statistics  
summary\_HML\_male

## mean\_mean\_finalHML\_male max\_mean\_finalHML\_male min\_mean\_finalHML\_male  
## 1 317.5135 351.1562 277.3571  
## sd\_mean\_finalHML\_male  
## 1 13.47029

summary\_HML\_female

## mean\_mean\_finalHML\_female max\_mean\_finalHML\_female min\_mean\_finalHML\_female  
## 1 294.4087 319.3333 259.5  
## sd\_mean\_finalHML\_female  
## 1 11.94954

summary\_RML\_male

## mean\_mean\_finalRML\_male max\_mean\_finalRML\_male min\_mean\_finalRML\_male  
## 1 242.8363 270.0909 216.1818  
## sd\_mean\_finalRML\_male  
## 1 9.812945

summary\_RML\_female

## mean\_mean\_finalRML\_female max\_mean\_finalRML\_female min\_mean\_finalRML\_female  
## 1 220.3566 248.75 195.25  
## sd\_mean\_finalRML\_female  
## 1 9.689279

summary\_FBL\_male

## mean\_mean\_finalFBL\_male max\_mean\_finalFBL\_male min\_mean\_finalFBL\_male  
## 1 439.514 487 387.6786  
## sd\_mean\_finalFBL\_male  
## 1 17.99187

summary\_FBL\_female

## mean\_mean\_finalFBL\_female max\_mean\_finalFBL\_female min\_mean\_finalFBL\_female  
## 1 406.637 439.1667 352.5833  
## sd\_mean\_finalFBL\_female  
## 1 15.8518

summary\_TML\_male

## mean\_mean\_finalTML\_male max\_mean\_finalTML\_male min\_mean\_finalTML\_male  
## 1 367.0466 412.75 327.4643  
## sd\_mean\_finalTML\_male  
## 1 15.84586

summary\_TML\_female

## mean\_mean\_finalTML\_female max\_mean\_finalTML\_female min\_mean\_finalTML\_female  
## 1 338.9564 371 299.5  
## sd\_mean\_finalTML\_female  
## 1 15.21897

## 4.2 Summarise the indices

### 4.2.1. Summarise the indices (combined)

# Brachial index  
## Mean\_BIc  
Mean\_BIc <- aggregate(Armset$ConventionalBI, by = list(Armset$SITE), FUN = mean)  
colnames(Mean\_BIc) <- c("SITE", "mean\_BIc")  
  
summary\_mean\_BIc <- data.frame(  
 mean\_mean\_BIc = mean(Mean\_BIc$mean\_BIc),  
 max\_mean\_BIc = max(Mean\_BIc$mean\_BIc),  
 min\_mean\_BIc = min(Mean\_BIc$mean\_BIc),  
 sd\_mean\_BIc = sd(Mean\_BIc$mean\_BIc)  
)  
  
## SD\_BIc  
SD\_BIc <- aggregate(Armset$ConventionalBI, by = list(Armset$SITE), FUN = sd)  
colnames(SD\_BIc) <- c("SITE", "sd\_BIc")  
  
summary\_sd\_BIc <- data.frame(  
 mean\_sd\_BIc = mean(SD\_BIc$sd\_BIc),  
 max\_sd\_BIc = max(SD\_BIc$sd\_BIc),  
 min\_sd\_BIc = min(SD\_BIc$sd\_BIc),  
 sd\_sd\_BIc = sd(SD\_BIc$sd\_BIc)  
)  
  
  
# Crural index  
## Mean\_CIt  
Mean\_CIc <- aggregate(Legset$ConventionalCI, by = list(Legset$SITE), FUN = mean)  
colnames(Mean\_CIc) <- c("SITE", "mean\_CIc")  
  
summary\_mean\_CIc <- data.frame(  
 mean\_mean\_CIc = mean(Mean\_CIc$mean\_CIc),  
 max\_mean\_CIc = max(Mean\_CIc$mean\_CIc),  
 min\_mean\_CIc = min(Mean\_CIc$mean\_CIc),  
 sd\_mean\_CIc = sd(Mean\_CIc$mean\_CIc)  
)  
  
## SD\_CIt  
SD\_CIc <- aggregate(Legset$ConventionalCI, by = list(Legset$SITE), FUN = sd)  
colnames(SD\_CIc) <- c("SITE", "sd\_CIc")  
  
summary\_sd\_CIc <- data.frame(  
 mean\_sd\_CIc = mean(SD\_CIc$sd\_CIc),  
 max\_sd\_CIc = max(SD\_CIc$sd\_CIc),  
 min\_sd\_CIc = min(SD\_CIc$sd\_CIc),  
 sd\_sd\_CIc = sd(SD\_CIc$sd\_CIc)  
)  
  
# Print summary statistics  
summary\_mean\_BIc

## mean\_mean\_BIc max\_mean\_BIc min\_mean\_BIc sd\_mean\_BIc  
## 1 0.7597169 0.8083747 0.7189173 0.01946723

summary\_mean\_CIc

## mean\_mean\_CIc max\_mean\_CIc min\_mean\_CIc sd\_mean\_CIc  
## 1 0.8347534 0.8693167 0.7930667 0.01539735

summary\_sd\_BIc

## mean\_sd\_BIc max\_sd\_BIc min\_sd\_BIc sd\_sd\_BIc  
## 1 0.02433381 0.07224909 0.002343426 0.008424029

summary\_sd\_CIc

## mean\_sd\_CIc max\_sd\_CIc min\_sd\_CIc sd\_sd\_CIc  
## 1 0.02297197 0.04407612 0.008654487 0.005982578

### 4.2.2. Summarise the indices (by sex)

# Brachial index  
## Mean\_BIc   
### Male  
Mean\_BIc\_male <- aggregate(ConventionalBI ~ SITE, data = Armset[Armset$Sex == 1, ], FUN = mean)  
colnames(Mean\_BIc\_male) <- c("SITE", "mean\_BIc\_male")  
  
summary\_mean\_BIc\_male <- data.frame(  
 mean\_mean\_BIc\_male = mean(Mean\_BIc\_male$mean\_BIc\_male),  
 max\_mean\_BIc\_male = max(Mean\_BIc\_male$mean\_BIc\_male),  
 min\_mean\_BIc\_male = min(Mean\_BIc\_male$mean\_BIc\_male),  
 sd\_mean\_BIc\_male = sd(Mean\_BIc\_male$mean\_BIc\_male)  
)  
  
## SD\_BIc   
### Male  
SD\_BIc\_male <- aggregate(ConventionalBI ~ SITE, data = Armset[Armset$Sex == 1, ], FUN = sd)  
colnames(SD\_BIc\_male) <- c("SITE", "sd\_BIc\_male")  
  
summary\_sd\_BIc\_male <- data.frame(  
 mean\_sd\_BIc\_male = mean(SD\_BIc\_male$sd\_BIc\_male, na.rm = TRUE),  
 max\_sd\_BIc\_male = max(SD\_BIc\_male$sd\_BIc\_male, na.rm = TRUE),  
 min\_sd\_BIc\_male = min(SD\_BIc\_male$sd\_BIc\_male, na.rm = TRUE),  
 sd\_sd\_BIc\_male = sd(SD\_BIc\_male$sd\_BIc\_male, na.rm = TRUE)  
)  
  
#### Female  
Mean\_BIc\_female <- aggregate(ConventionalBI ~ SITE, data = Armset[Armset$Sex == 2, ], FUN = mean)  
colnames(Mean\_BIc\_female) <- c("SITE", "mean\_BIc\_female")  
  
summary\_mean\_BIc\_female <- data.frame(  
 mean\_mean\_BIc\_female = mean(Mean\_BIc\_female$mean\_BIc\_female),  
 max\_mean\_BIc\_female = max(Mean\_BIc\_female$mean\_BIc\_female),  
 min\_mean\_BIc\_female = min(Mean\_BIc\_female$mean\_BIc\_female),  
 sd\_mean\_BIc\_female = sd(Mean\_BIc\_female$mean\_BIc\_female)  
)  
  
### SD\_BIt   
SD\_BIc\_female <- aggregate(ConventionalBI ~ SITE, data = Armset[Armset$Sex == 2, ], FUN = sd)  
colnames(SD\_BIc\_female) <- c("SITE", "sd\_BIc\_female")  
  
summary\_sd\_BIc\_female <- data.frame(  
 mean\_sd\_BIc\_female = mean(SD\_BIc\_female$sd\_BIc\_female, na.rm = TRUE),  
 max\_sd\_BIc\_female = max(SD\_BIc\_female$sd\_BIc\_female, na.rm = TRUE),  
 min\_sd\_BIc\_female = min(SD\_BIc\_female$sd\_BIc\_female, na.rm = TRUE),  
 sd\_sd\_BIc\_female = sd(SD\_BIc\_female$sd\_BIc\_female, na.rm = TRUE)  
)  
  
## Crural index  
  
### Male  
#### Mean\_CIc  
Mean\_CIc\_male <- aggregate(Legset$ConventionalCI[Legset$Sex == 1], by = list(Legset$SITE[Legset$Sex == 1]), FUN = mean)  
colnames(Mean\_CIc\_male) <- c("SITE", "mean\_CIc\_male")  
  
summary\_mean\_CIc\_male <- data.frame(  
 mean\_mean\_CIc\_male = mean(Mean\_CIc\_male$mean\_CIc\_male),  
 max\_mean\_CIc\_male = max(Mean\_CIc\_male$mean\_CIc\_male),  
 min\_mean\_CIc\_male = min(Mean\_CIc\_male$mean\_CIc\_male),  
 sd\_mean\_CIc\_male = sd(Mean\_CIc\_male$mean\_CIc\_male)  
)  
  
#### SD\_CIc   
SD\_CIc\_male <- aggregate(Legset$ConventionalCI[Legset$Sex == 1], by = list(Legset$SITE[Legset$Sex == 1]), FUN = sd)  
colnames(SD\_CIc\_male) <- c("SITE", "sd\_CIc\_male")  
  
summary\_sd\_CIc\_male <- data.frame(  
 mean\_sd\_CIc\_male = mean(SD\_CIc\_male$sd\_CIc\_male, na.rm = TRUE),  
 max\_sd\_CIc\_male = max(SD\_CIc\_male$sd\_CIc\_male, na.rm = TRUE),  
 min\_sd\_CIc\_male = min(SD\_CIc\_male$sd\_CIc\_male, na.rm = TRUE),  
 sd\_sd\_CIc\_male = sd(SD\_CIc\_male$sd\_CIc\_male, na.rm = TRUE)  
)  
  
### Female  
#### Mean\_CIc  
Mean\_CIc\_female <- aggregate(Legset$ConventionalCI[Legset$Sex == 2], by = list(Legset$SITE[Legset$Sex == 2]), FUN = mean)  
colnames(Mean\_CIc\_female) <- c("SITE", "mean\_CIc\_female")  
  
summary\_mean\_CIc\_female <- data.frame(  
 mean\_mean\_CIc\_female = mean(Mean\_CIc\_female$mean\_CIc\_female),  
 max\_mean\_CIc\_female = max(Mean\_CIc\_female$mean\_CIc\_female),  
 min\_mean\_CIc\_female = min(Mean\_CIc\_female$mean\_CIc\_female),  
 sd\_mean\_CIc\_female = sd(Mean\_CIc\_female$mean\_CIc\_female)  
)  
  
#### SD\_CIc   
SD\_CIc\_female <- aggregate(Legset$ConventionalCI[Legset$Sex == 2], by = list(Legset$SITE[Legset$Sex == 2]), FUN = sd)  
colnames(SD\_CIc\_female) <- c("SITE", "sd\_CIc\_female")  
  
summary\_sd\_CIc\_female <- data.frame(  
 mean\_sd\_CIc\_female = mean(SD\_CIc\_female$sd\_CIc\_female, na.rm = TRUE),  
 max\_sd\_CIc\_female = max(SD\_CIc\_female$sd\_CIc\_female, na.rm = TRUE),  
 min\_sd\_CIc\_female = min(SD\_CIc\_female$sd\_CIc\_female, na.rm = TRUE),  
 sd\_sd\_CIc\_female = sd(SD\_CIc\_female$sd\_CIc\_female, na.rm = TRUE)  
)  
  
  
# Print summary statistics  
summary\_mean\_BIc\_male

## mean\_mean\_BIc\_male max\_mean\_BIc\_male min\_mean\_BIc\_male sd\_mean\_BIc\_male  
## 1 0.765382 0.8177492 0.7202265 0.02026929

summary\_mean\_CIc\_male

## mean\_mean\_CIc\_male max\_mean\_CIc\_male min\_mean\_CIc\_male sd\_mean\_CIc\_male  
## 1 0.8353172 0.8707922 0.774247 0.01662843

summary\_sd\_BIc\_male

## mean\_sd\_BIc\_male max\_sd\_BIc\_male min\_sd\_BIc\_male sd\_sd\_BIc\_male  
## 1 0.02231857 0.08363879 0.002185631 0.01040683

summary\_sd\_CIc\_male

## mean\_sd\_CIc\_male max\_sd\_CIc\_male min\_sd\_CIc\_male sd\_sd\_CIc\_male  
## 1 0.02309152 0.07623647 0.002767523 0.008869297

summary\_mean\_BIc\_female

## mean\_mean\_BIc\_female max\_mean\_BIc\_female min\_mean\_BIc\_female  
## 1 0.7489142 0.803763 0.6893204  
## sd\_mean\_BIc\_female  
## 1 0.02204575

summary\_mean\_CIc\_female

## mean\_mean\_CIc\_female max\_mean\_CIc\_female min\_mean\_CIc\_female  
## 1 0.833674 0.8829517 0.7516861  
## sd\_mean\_CIc\_female  
## 1 0.01885759

summary\_sd\_BIc\_female

## mean\_sd\_BIc\_female max\_sd\_BIc\_female min\_sd\_BIc\_female sd\_sd\_BIc\_female  
## 1 0.02219891 0.08592764 0.0003745373 0.01269634

summary\_sd\_CIc\_female

## mean\_sd\_CIc\_female max\_sd\_CIc\_female min\_sd\_CIc\_female sd\_sd\_CIc\_female  
## 1 0.02064353 0.05740544 0.0001869027 0.008734694

# 5. Compare the limb lengths and indices between males and females

## 5.1. Conduct t-tests on the limb lengths

results\_ttest\_limblength <- data.frame()  
  
# Conduct t-tests for limb lengths between males and females at each site  
for (site in unique(Armset$SITE)) {  
 male\_hml <- subset(Armset, Sex == 1 & SITE == site)$finalHML  
 female\_hml <- subset(Armset, Sex == 2 & SITE == site)$finalHML  
 male\_rml <- subset(Armset, Sex == 1 & SITE == site)$finalRML  
 female\_rml <- subset(Armset, Sex == 2 & SITE == site)$finalRML  
  
 male\_fbl <- subset(Legset, Sex == 1 & SITE == site)$finalFBL  
 female\_fbl <- subset(Legset, Sex == 2 & SITE == site)$finalFBL  
 male\_tml <- subset(Legset, Sex == 1 & SITE == site)$finalTML  
 female\_tml <- subset(Legset, Sex == 2 & SITE == site)$finalTML  
  
 if (length(male\_hml) >= 3 & length(female\_hml) >= 3 & length(male\_rml) >= 3 & length(female\_rml) >= 3 & length(male\_fbl) >= 3 & length(female\_fbl) >= 3 & length(male\_tml) >= 3 & length(female\_tml) >= 3) {  
 ttest\_hml <- t.test(male\_hml, female\_hml)  
 ttest\_rml <- t.test(male\_rml, female\_rml)  
 ttest\_fbl <- t.test(male\_fbl, female\_fbl)  
 ttest\_tml <- t.test(male\_tml, female\_tml)  
  
 results\_ttest\_limblength <- rbind(results\_ttest\_limblength,  
 data.frame(Site = site,  
 Variable = c("HML", "RML", "FBL", "TML"),  
 Group = rep("Male vs. Female", 4),  
 t\_statistic = c(ttest\_hml$statistic, ttest\_rml$statistic, ttest\_fbl$statistic, ttest\_tml$statistic),  
 p\_value = c(ttest\_hml$p.value, ttest\_rml$p.value, ttest\_fbl$p.value, ttest\_tml$p.value))  
 )  
 }  
}  
  
# HML  
num\_sig\_ttests\_HML <- sum(results\_ttest\_limblength$Variable == 'HML' & results\_ttest\_limblength$p\_value < 0.05)  
cat("Number of significant T-tests for HML:", num\_sig\_ttests\_HML, "\n")

## Number of significant T-tests for HML: 78

num\_nonsig\_ttests\_HML <- sum(results\_ttest\_limblength$Variable == 'HML' & results\_ttest\_limblength$p\_value >= 0.05)  
cat("Number of non-significant T-tests for HML:", num\_nonsig\_ttests\_HML, "\n")

## Number of non-significant T-tests for HML: 21

# RML  
num\_sig\_ttests\_RML <- sum(results\_ttest\_limblength$Variable == 'RML' & results\_ttest\_limblength$p\_value < 0.05)  
cat("Number of significant T-tests for RML:", num\_sig\_ttests\_RML, "\n")

## Number of significant T-tests for RML: 81

num\_nonsig\_ttests\_RML <- sum(results\_ttest\_limblength$Variable == 'RML' & results\_ttest\_limblength$p\_value >= 0.05)  
cat("Number of non-significant T-tests for RML:", num\_nonsig\_ttests\_RML, "\n")

## Number of non-significant T-tests for RML: 18

# FBL  
num\_sig\_ttests\_FBL <- sum(results\_ttest\_limblength$Variable == 'FBL' & results\_ttest\_limblength$p\_value < 0.05)  
cat("Number of significant T-tests for FBL:", num\_sig\_ttests\_FBL, "\n")

## Number of significant T-tests for FBL: 81

num\_nonsig\_ttests\_FBL <- sum(results\_ttest\_limblength$Variable == 'FBL' & results\_ttest\_limblength$p\_value >= 0.05)  
cat("Number of non-significant T-tests for FBL:", num\_nonsig\_ttests\_FBL, "\n")

## Number of non-significant T-tests for FBL: 18

# TML  
num\_sig\_ttests\_TML <- sum(results\_ttest\_limblength$Variable == 'TML' & results\_ttest\_limblength$p\_value < 0.05)  
cat("Number of significant T-tests for TML:", num\_sig\_ttests\_TML, "\n")

## Number of significant T-tests for TML: 73

num\_nonsig\_ttests\_TML <- sum(results\_ttest\_limblength$Variable == 'TML' & results\_ttest\_limblength$p\_value >= 0.05)  
cat("Number of non-significant T-tests for TML:", num\_nonsig\_ttests\_TML, "\n")

## Number of non-significant T-tests for TML: 26

## 5.2. Conduct t-tests on the indices

results\_ttest\_BC <- data.frame()  
  
# Conduct t-tests for BI and CI between males and females at each site  
for (site in unique(Armset$SITE)) {  
 male\_bic <- subset(Armset, Sex == 1 & SITE == site)$ConventionalBI  
 female\_bic <- subset(Armset, Sex == 2 & SITE == site)$ConventionalBI  
 male\_cic <- subset(Legset, Sex == 1 & SITE == site)$ConventionalCI  
 female\_cic <- subset(Legset, Sex == 2 & SITE == site)$ConventionalCI  
  
 if (length(male\_bic) >= 3 & length(female\_bic) >= 3 & length(male\_cic) >= 3 & length(female\_cic) >= 3) {  
 ttest\_bic <- t.test(male\_bic, female\_bic)  
 ttest\_cic <- t.test(male\_cic, female\_cic)  
  
 results\_ttest\_BC <- rbind(results\_ttest\_BC,  
 data.frame(Site = site,  
 Variable = c("BIc", "CIc"),  
 Group = rep("Male vs. Female", 2),  
 t\_statistic = c(ttest\_bic$statistic, ttest\_cic$statistic),  
 p\_value = c(ttest\_bic$p.value, ttest\_cic$p.value))  
 )  
 }  
}  
  
# Count the number of significant and non-significant t-tests  
num\_sig\_ttests\_BIc <- sum(results\_ttest\_BC$Variable == 'BIc' & results\_ttest\_BC$p\_value < 0.05)  
cat("Number of significant T-tests for BIc:", num\_sig\_ttests\_BIc, "\n")

## Number of significant T-tests for BIc: 26

num\_nonsig\_ttests\_BIc <- sum(results\_ttest\_BC$Variable == 'BIc' & results\_ttest\_BC$p\_value >= 0.05)  
cat("Number of non-significant T-tests for BIc:", num\_nonsig\_ttests\_BIc, "\n")

## Number of non-significant T-tests for BIc: 73

num\_sig\_ttests\_CIc <- sum(results\_ttest\_BC$Variable == 'CIc' & results\_ttest\_BC$p\_value < 0.05)  
cat("Number of significant T-tests for CIc:", num\_sig\_ttests\_CIc, "\n")

## Number of significant T-tests for CIc: 4

num\_nonsig\_ttests\_CIc <- sum(results\_ttest\_BC$Variable == 'CIc' & results\_ttest\_BC$p\_value >= 0.05)  
cat("Number of non-significant T-tests for CIc:", num\_nonsig\_ttests\_CIc, "\n")

## Number of non-significant T-tests for CIc: 95

# 6. Compare the conventional and aggregated indices without the simulation

## 6.1. Compute averages of conventinal values and mean limb lengths for each site

res\_arm <- aggregate(cbind(finalHML,finalRML,ConventionalBI)~SITE,data=Armset,mean)  
res\_leg <- aggregate(cbind(finalFBL,finalTML,ConventionalCI)~SITE,data=Legset,mean)

## 6.2. compute conventional indices obtained from the mean limb lengths

res\_arm$BIa <- res\_arm$finalRML/res\_arm$finalHML  
res\_leg$CIa <- res\_leg$finalTML/res\_leg$finalFBL

## 6.3. Linear regression

# Fit the linear regression model  
lm\_model\_bi <- lm(BIa ~ ConventionalBI, data = res\_arm)  
lm\_model\_ci <- lm(CIa ~ ConventionalCI, data = res\_leg)  
  
# Print the model summary  
summary(lm\_model\_bi)

##   
## Call:  
## lm(formula = BIa ~ ConventionalBI, data = res\_arm)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.0037638 -0.0002091 0.0000340 0.0002725 0.0017243   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.003114 0.001952 1.595 0.113   
## ConventionalBI 0.995895 0.002569 387.678 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.0005658 on 127 degrees of freedom  
## Multiple R-squared: 0.9992, Adjusted R-squared: 0.9991   
## F-statistic: 1.503e+05 on 1 and 127 DF, p-value: < 2.2e-16

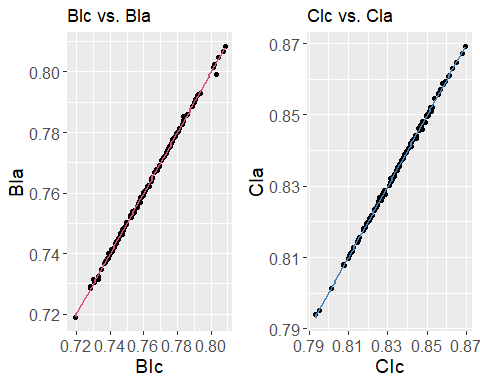
summary(lm\_model\_ci)

##   
## Call:  
## lm(formula = CIa ~ ConventionalCI, data = res\_leg)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -1.375e-03 -1.768e-04 2.800e-07 1.887e-04 1.184e-03   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.002570 0.001923 1.337 0.184   
## ConventionalCI 0.996803 0.002303 432.781 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.0004121 on 134 degrees of freedom  
## Multiple R-squared: 0.9993, Adjusted R-squared: 0.9993   
## F-statistic: 1.873e+05 on 1 and 134 DF, p-value: < 2.2e-16

## 6.4. Plot

# Scatter plot with linear regression line for BIa vs. BIc  
plot\_bi <- ggplot(data = res\_arm, aes(x = ConventionalBI, y = BIa)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", se = FALSE, color = "#d45087") +  
 labs(title = "BIc vs. BIa", x = "BIc", y = "BIa")+  
 theme(axis.text = element\_text(size = 12),  
 axis.title = element\_text(size = 14))  
  
# Scatter plot with linear regression line for CIa vs. CIc  
plot\_ci <- ggplot(data = res\_leg, aes(x = ConventionalCI, y = CIa)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", se = FALSE, color = "steelblue") +  
 labs(title = "CIc vs. CIa", x = "CIc", y = "CIa")+  
 theme(axis.text = element\_text(size = 12),  
 axis.title = element\_text(size = 14))  
  
# Combine the plots  
combined\_indices <- grid.arrange(plot\_bi, plot\_ci, ncol = 2,nrow = 1)

## `geom\_smooth()` using formula = 'y ~ x'  
## `geom\_smooth()` using formula = 'y ~ x'



# 7. Simulating Three Commingling Conditions

## Description of Arguments in the Simulation

### Input Arguments

# - \*\*N\*\*: The total number of sample limbs, which can be the combination of the humerus and radius for the upper limb, or the combination of the femur and tibiae for the lower limb  
# - \*\*n1\*\*: The number of matched bones, i.e., humeri and radius (or the femur and tibia) from the same individual  
# - \*\*n2\*\*: The number of matched bones, i.e., the humerus and radius (or the femur and tibia) from different individuals  
# - \*\*c\*\*: The commingling degree of sample limbs, which quantifies the proportion of unmatched bones within the assemblage. For example, the n2 can be calculated as N \* c, while n1 can be calculated as N \* (1 - c)  
# - \*\*mean.hml\*\*: The mean of the maximum humeral length  
# - \*\*mean.fbl\*\*: The mean of the bicondylar femoral length  
# - \*\*mean.bic\*\*: The mean of the conventional Brachial Index  
# - \*\*mean.cic\*\*: The mean of the conventional Crural Index  
# - \*\*sd.hml\*\*: The standard deviation of the maximum humeral length  
# - \*\*sd.fbl\*\*: The standard deviation of the bicondylar femoral length  
# - \*\*sd.bic\*\*: The standard deviation of the conventionalBrachial Index, which also represents the within-group variation in the intra-upper-limb proportion.  
# - \*\*sd.cic\*\*: The standard deviation of the conventional Crural Index, which also represents the within-group variation in the intra-lower-limb proportion.  
# - \*\*nsim\*\*: The number of iterations to be performed in the simulation loop

### Output Arguments

# - \*\*est.bi\*\*: Estimated Brachial Index using the aggregate method  
# - \*\*est.ci\*\*: Estimated Crural Index using the aggregate method  
# - \*\*error.bi\*\*: The difference between the estimated Brachial Index and the input Brachial Index, i.e., error.bi = est.bi – mean.bi  
# - \*\*error.ci\*\*: The difference between the estimated Crural Index and the input Crural Index, i.e., error.ci = est.ci – mean.ci

## 7.1. Function for calculating and comparing the aggregated and conventional indices

### 7.1.1 BIc vs. BIa

simFun\_bi <- function(n1,n2,mean.hml,sd.hml,sd.bi,mean.bi,nsim)  
{  
 est.bi <- numeric(length=nsim) #just a placeholder vector for storing estimates  
 for (i in 1:nsim) #main simulation loop  
 {  
 #Simulate paired individuals  
 hml.paired <- rnorm(n1/2,mean=mean.hml,sd=sd.hml)  
 bi.paired <- rnorm(n1/2,mean=mean.bi,sd=sd.bi)  
 rml.paired <- hml.paired \* bi.paired  
 #Simulate unpaired individuals  
 hml.unpaired <- rnorm(n2/2,mean=mean.hml,sd=sd.hml)   
 #Simulate half the femur as normal  
 rml.unpaired <- rnorm(n2/2,mean=mean.hml,sd=sd.hml) \* rnorm(n2/2,mean=mean.bi,sd=sd.bi) #Simulate another set of femurs but store only their corresponding Tibia after drawing other crural index values  
 #Combine  
 hml <- c(hml.paired,hml.unpaired)  
 rml <- c(rml.paired,rml.unpaired)  
 #Estimate  
 est.bi[i] <- mean(rml)/mean(hml)  
 }  
 return(list(true.bi=mean.bi,est.bi=est.bi,error=est.bi-mean.bi))  
}

### 7.1.2 CIc vs. CIa

simFun\_ci <- function(n1,n2,mean.fbl,sd.fbl,sd.ci,mean.ci,nsim)  
{  
 est.ci <- numeric(length=nsim) #just a placeholder vector for storing estimates  
 for (i in 1:nsim) #main simulaion loop  
 {  
 #Simulate paired individuals  
 fbl.paired <- rnorm(n1/2,mean=mean.fbl,sd=sd.fbl)  
 ci.paired <- rnorm(n1/2,mean=mean.ci,sd=sd.ci)  
 tml.paired <- fbl.paired \* ci.paired  
 #Simulate unpaired individuals  
 fbl.unpaired <- rnorm(n2/2,mean=mean.fbl,sd=sd.fbl)   
 #Simulate half the femur as normal  
 tml.unpaired <- rnorm(n2/2,mean=mean.fbl,sd=sd.fbl) \* rnorm(n2/2,mean=mean.ci,sd=sd.ci) #Simulate another set of femurs but store only their corresponding Tibia after drawing other crural index values  
 #Combine  
 fbl <- c(fbl.paired,fbl.unpaired)  
 tml <- c(tml.paired,tml.unpaired)  
 #Estimate  
 est.ci[i] <- mean(tml)/mean(fbl)  
 }  
 return(list(true.ci=mean.ci,est.ci=est.ci,error=est.ci-mean.ci)) #   
}

## 7.2. Function for simulating the sample size (n), commingling (c) and variation (v)

### 7.2.1. Simulate the BI

# Create grid of parameter combinations to simulate  
n <- seq(10, 300, length.out = 30) # Sample size  
c <- seq(0, 1, length.out = 30) # Commingling Degree  
v <- c(0.01, 0.02, 0.08) # Within-group Variation in the Brachial index  
parm <- expand.grid(n, c, v)  
  
# Modify simulate function to run simulation 10000 times  
simulate <- function(parms, n\_sims = 10000) {  
 # Extract parameters from parms  
 n <- parms[1]  
 c <- parms[2]  
 v <- parms[3]  
   
 # Round n to the nearest even number  
 n <- round(n/2) \* 2  
   
 # Initialize variables to store results  
 error\_set <- numeric(n\_sims)  
   
 # Run simulations  
 for (i in 1:n\_sims) {  
 # Call the simFun\_ci function with the specified parameters  
 sim\_result <- simFun\_bi(n1 = round(n\*(1-c)), n2 = round(n\*c), mean.hml=308.73,sd.hml=13.24, sd.bi = v, mean.bi = 0.7597, nsim = 1)  
   
 # Compute error  
 error <- abs(sim\_result$error)  
   
 # Store results  
 error\_set[i] <- sim\_result$error  
 }  
   
 # Compute summary statistics of the errors  
 mean\_abs\_error <- abs(mean(error\_set)) # the absolute value of the mean  
 median\_abs\_error <- abs(median(error\_set)) # absolute value of the median\_error  
 percentile\_75 <- abs(quantile(error\_set, probs = 0.75, na.rm = TRUE))# Compute the 75th percentile of the errors  
   
 # Compute values that are greater than 0.02  
 abs\_error\_count <- sum(abs(error\_set) > 0.02, na.rm = TRUE) # the number of errors in the error\_set that have an absolute value greater than 0.02, excluding any NA values.  
 # the proportion of simulations where the absolute value of the median\_error is greater than 0.02  
 prop\_large\_error <- abs\_error\_count / sum(!is.na(error\_set))  
  
 # Create data frame with results and parameters  
 results\_df\_bi <- data.frame(n = n, c = c, v = v,   
 mean\_abs\_error = mean\_abs\_error,median\_abs\_error = median\_abs\_error, percentile\_75=percentile\_75,   
 abs\_error\_count = abs\_error\_count, prop\_large\_error = prop\_large\_error)  
  
 return(results\_df\_bi)  
}  
  
# Run simulation and store results in a data frame  
set.seed(123) # set seed for reproducibility  
results\_bi <- replicate(1, apply(parm, 1, simulate), simplify = FALSE)  
results\_df\_bi <- bind\_rows(results\_bi)

### 7.2.1. Simulate the CI

# Create grid of parameter combinations to simulate  
n <- seq(10, 300, length.out = 30) # Sample size  
c <- seq(0, 1, length.out = 30) # Commingling Degree  
v <- c(0.01, 0.02, 0.05) # Within-group Variation in the Crural index  
parm <- expand.grid(n, c, v)  
  
# Modify simulate function to run simulation 10000 times  
simulate <- function(parms, n\_sims = 10000) {  
 # Extract parameters from parms  
 n <- parms[1]  
 c <- parms[2]  
 v <- parms[3]  
   
 # Round n to the nearest even number  
 n <- round(n/2) \* 2  
   
 # Initialize variables to store results  
 error\_set <- numeric(n\_sims)  
   
 # Run simulations  
 for (i in 1:n\_sims) {  
 # Call the simFun\_ci function with the specified parameters  
 sim\_result <- simFun\_ci(n1 = round(n\*(1-c)), n2 = round(n\*c), mean.fbl=426.64,sd.fbl=26.07, sd.ci = v, mean.ci = 0.8348, nsim = 1)  
   
 # Compute error  
 error <- abs(sim\_result$error)  
   
 # Store results  
 error\_set[i] <- sim\_result$error  
 }  
   
 # Compute summary statistics of the errors  
 mean\_abs\_error <- abs(mean(error\_set)) # the absolute value of the mean  
 median\_abs\_error <- abs(median(error\_set)) # absolute value of the median\_error  
 percentile\_75 <- abs(quantile(error\_set, probs = 0.75, na.rm = TRUE))# Compute the 75th percentile of the errors  
   
 # Compute values that are greater than 0.02  
 abs\_error\_count <- sum(abs(error\_set) > 0.02, na.rm = TRUE) # the number of errors in the error\_set that have an absolute value greater than 0.02, excluding any NA values.  
 # the proportion of simulations where the absolute value of the median\_error is greater than 0.02  
 prop\_large\_error <- abs\_error\_count / sum(!is.na(error\_set))  
   
  
 # Create data frame with results and parameters  
 results\_df\_ci <- data.frame(n = n, c = c, v = v,   
 mean\_abs\_error = mean\_abs\_error,median\_abs\_error = median\_abs\_error, percentile\_75=percentile\_75,   
 abs\_error\_count = abs\_error\_count, prop\_large\_error = prop\_large\_error)  
  
 return(results\_df\_ci)  
}  
  
# Run simulation and store results in a data frame  
set.seed(123) # set seed for reproducibility  
results\_ci <- replicate(1, apply(parm, 1, simulate), simplify = FALSE)  
results\_df\_ci <- bind\_rows(results\_ci)

# 8. Plot

## 8.1. Set the parameter for indicative lines and common colour scale for each graph

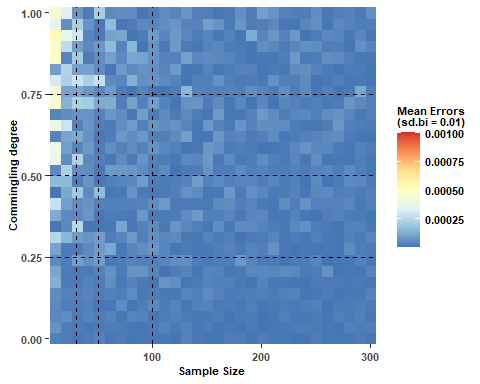
# parameter for indicative lines  
simulated\_n <- c (30, 50, 100)  
simulated\_c <- c (0.25, 0.5, 0.75)  
  
# Find the common color scale limits for different sets of plots (mean, median, 75 percentile, and the number as well as proportion of large errors)  
## BI  
color\_scale\_mean\_bi <- scale\_fill\_distiller(palette = "RdYlBu", limits = c(min(results\_df\_bi$mean\_abs\_error), max(results\_df\_bi$mean\_abs\_error)))  
color\_scale\_median\_bi <- scale\_fill\_distiller(palette = "RdYlBu", limits = c(min(results\_df\_bi$median\_abs\_error), max(results\_df\_bi$median\_abs\_error)))  
color\_scale\_percentile\_75\_bi <- scale\_fill\_distiller(palette = "RdYlBu", limits = c(min(results\_df\_bi$percentile\_75), max(results\_df\_bi$percentile\_75)))  
color\_scale\_no\_large\_error\_bi <- scale\_fill\_distiller(palette = "RdYlBu", limits = c(min(results\_df\_bi$abs\_error\_count), max(results\_df\_bi$abs\_error\_count)))  
color\_scale\_prop\_large\_error\_bi <- scale\_fill\_distiller(palette = "RdYlBu", limits = c(min(results\_df\_bi$prop\_large\_error), max(results\_df\_bi$prop\_large\_error)))  
  
## CI  
color\_scale\_mean\_ci <- scale\_fill\_distiller(palette = "RdYlBu", limits = c(min(results\_df\_ci$mean\_abs\_error), max(results\_df\_ci$mean\_abs\_error)))  
color\_scale\_median\_ci <- scale\_fill\_distiller(palette = "RdYlBu", limits = c(min(results\_df\_ci$median\_abs\_error), max(results\_df\_ci$median\_abs\_error)))  
color\_scale\_percentile\_75\_ci <- scale\_fill\_distiller(palette = "RdYlBu", limits = c(min(results\_df\_ci$percentile\_75), max(results\_df\_ci$percentile\_75)))  
color\_scale\_no\_large\_error\_ci <- scale\_fill\_distiller(palette = "RdYlBu", limits = c(min(results\_df\_ci$abs\_error\_count), max(results\_df\_ci$abs\_error\_count)))  
color\_scale\_prop\_large\_error\_ci <- scale\_fill\_distiller(palette = "RdYlBu", limits = c(min(results\_df\_ci$prop\_large\_error), max(results\_df\_ci$prop\_large\_error)))

## 8.2. Plot-BI

### 8.2.1.Variation = 0.01

#### 8.2.1.1. Mean error ~ sample size + commingling

plot01 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.01, ], aes(x = n, y = c, fill = mean\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 color\_scale\_mean\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling degree') +  
 labs(fill = "Mean Errors\  
(sd.bi = 0.01)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot01

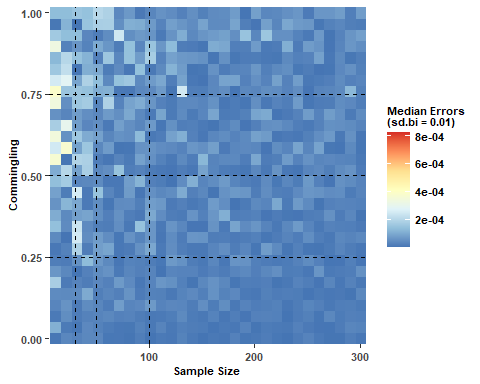


#### 8.2.1.2. Median absolute error ~ sample size + commingling

plot02 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.01, ], aes(x = n, y = c, fill = median\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_median\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Median Errors\  
(sd.bi = 0.01)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))

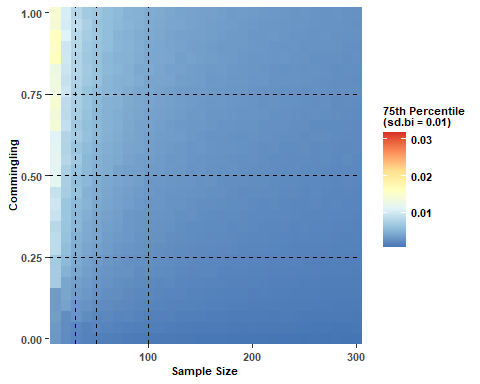
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
## i Please use `linewidth` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.

# Print  
plot02



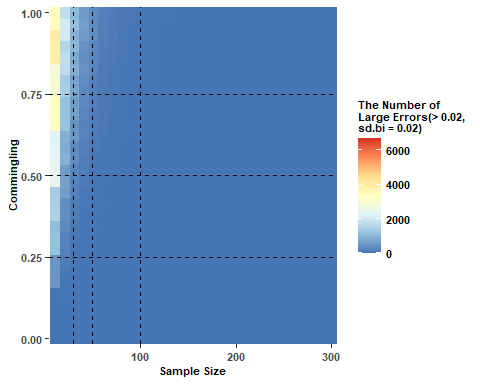
#### 8.2.1.3. Error 75% ~ sample size + commingling

plot03 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.01, ], aes(x = n, y = c, fill = percentile\_75)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_percentile\_75\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "75th Percentile\  
(sd.bi = 0.01)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot03



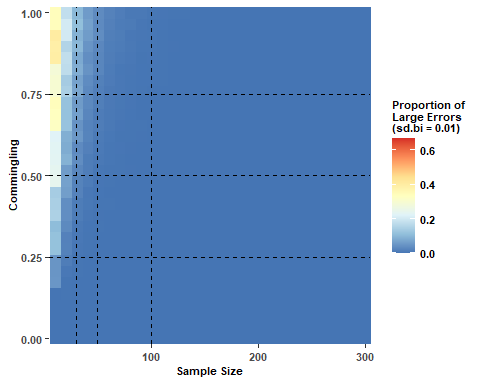
#### 8.2.1.4. Large error count ~ sample size + commingling

plot04 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.01, ], aes(x = n, y = c, fill = abs\_error\_count)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_no\_large\_error\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "The Number of\  
Large Errors(> 0.02,\  
sd.bi = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot04



#### 8.2.1.5. Proportion of large error ~ sample size + commingling

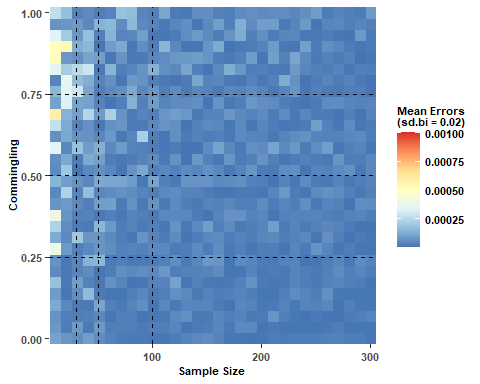
plot05 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.01, ], aes(x = n, y = c, fill = prop\_large\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_prop\_large\_error\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Proportion of\  
Large Errors\  
(sd.bi = 0.01)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
# Print  
plot05



### 8.2.2. Variation = 0.02

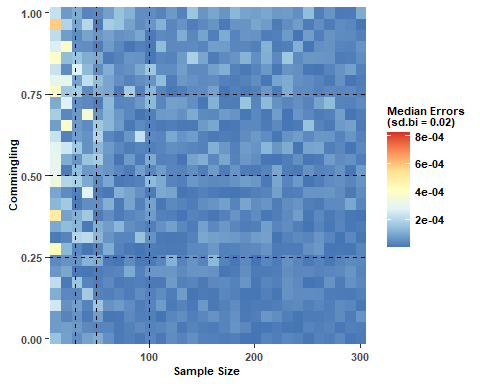
#### 8.2.2.1. Mean error ~ sample size + commingling

plot06 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.02, ], aes(x = n, y = c, fill = mean\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 color\_scale\_mean\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Mean Errors\  
(sd.bi = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot06



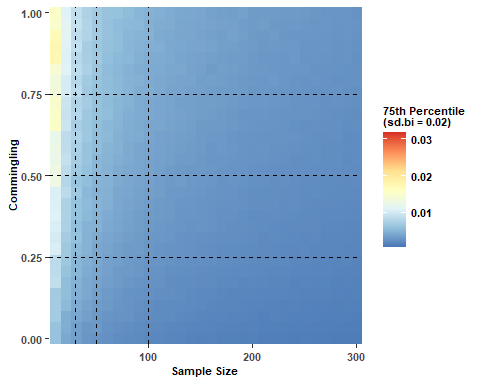
#### 8.2.2.2. Median absolute error ~ sample size + commingling

plot07 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.02, ], aes(x = n, y = c, fill = median\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_median\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Median Errors\  
(sd.bi = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot07



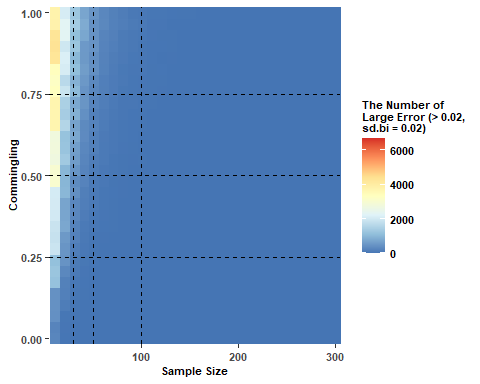
#### 8.2.2.3. Error 75% ~ sample size + commingling

plot08 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.02, ], aes(x = n, y = c, fill = percentile\_75)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_percentile\_75\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "75th Percentile\  
(sd.bi = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot08



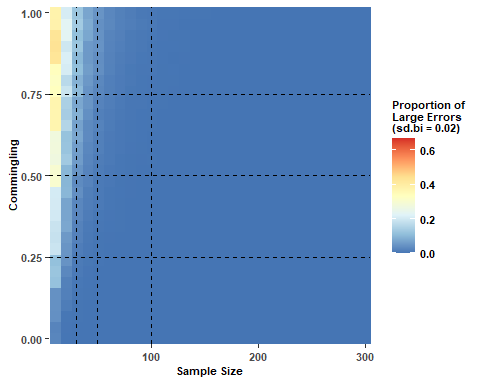
#### 8.2.2.4. Large error count ~ sample size + commingling

plot09 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.02, ], aes(x = n, y = c, fill = abs\_error\_count)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_no\_large\_error\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "The Number of\  
Large Error (> 0.02,\  
sd.bi = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot09



#### 8.2.2.5. Proportion of large error ~ sample size + commingling

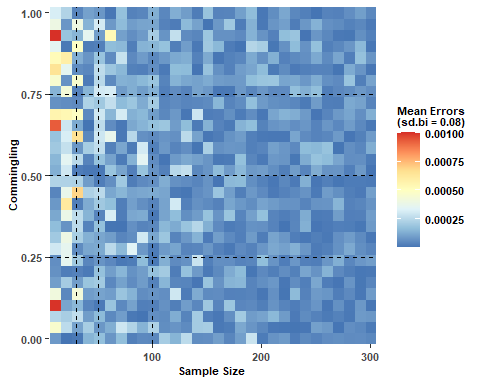
plot010 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.02, ], aes(x = n, y = c, fill = prop\_large\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_prop\_large\_error\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Proportion of\  
Large Errors\  
(sd.bi = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot010



### 8.2.1. Variation = 0.08

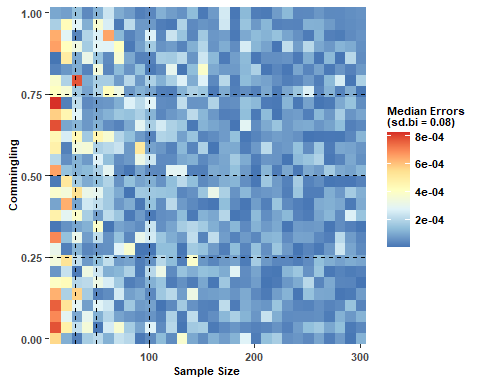
#### 8.2.3.1. Mean error ~ sample size + commingling

plot011 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.08, ], aes(x = n, y = c, fill = mean\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 color\_scale\_mean\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Mean Errors\  
(sd.bi = 0.08)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot011



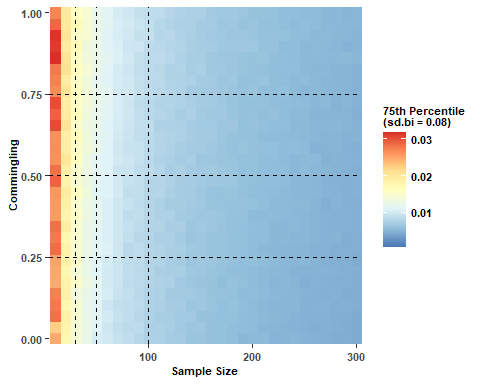
#### 8.2.3.2. Median absolute error ~ sample size + commingling

plot012 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.08, ], aes(x = n, y = c, fill = median\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_median\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Median Errors\  
(sd.bi = 0.08)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot012



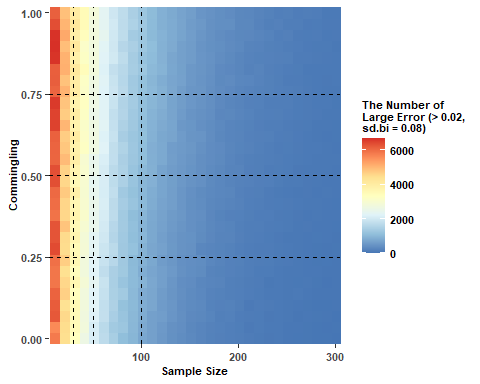
#### 8.2.3.3. Error 75% ~ sample size + commingling

plot013 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.08, ], aes(x = n, y = c, fill = percentile\_75)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_percentile\_75\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "75th Percentile\  
(sd.bi = 0.08)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot013



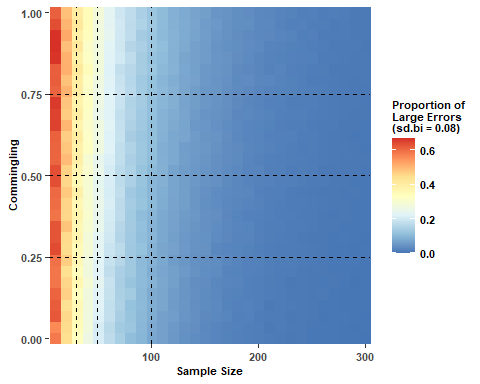
#### 8.2.3.4. Large error count ~ sample size + commingling

plot014 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.08, ], aes(x = n, y = c, fill = abs\_error\_count)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_no\_large\_error\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "The Number of\  
Large Error (> 0.02,\  
sd.bi = 0.08)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot014



#### 8.2.3.5. Proportion of large error ~ sample size + commingling

plot015 <- ggplot(results\_df\_bi[results\_df\_bi$v == 0.08, ], aes(x = n, y = c, fill = prop\_large\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_prop\_large\_error\_bi +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Proportion of\  
Large Errors\  
(sd.bi = 0.08)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot015

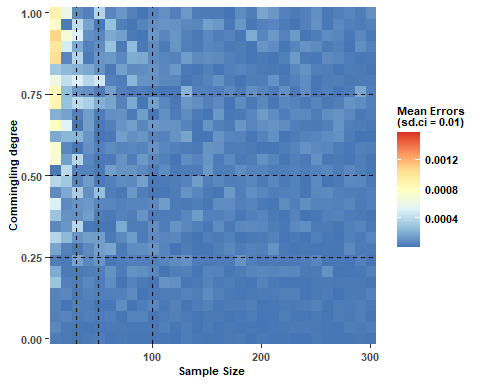


## 8.3. Plot CI

### 8.3.1. Variation = 0.01

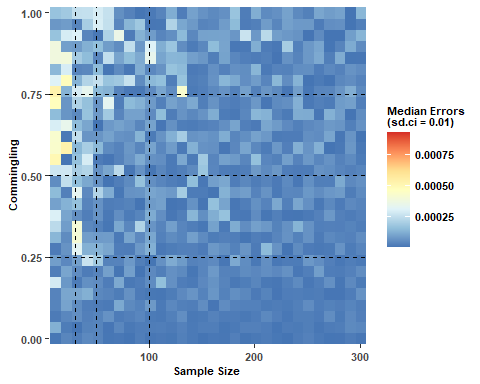
#### 8.3.1.1. Mean error ~ sample size + commingling

plot1 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.01, ], aes(x = n, y = c, fill = mean\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 color\_scale\_mean\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling degree') +  
 labs(fill = "Mean Errors\  
(sd.ci = 0.01)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot1



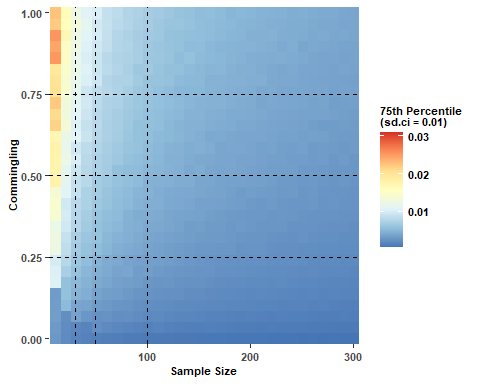
#### 8.3.1.2. Median absolute error ~ sample size + commingling

plot2 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.01, ], aes(x = n, y = c, fill = median\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_median\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Median Errors\  
(sd.ci = 0.01)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot2



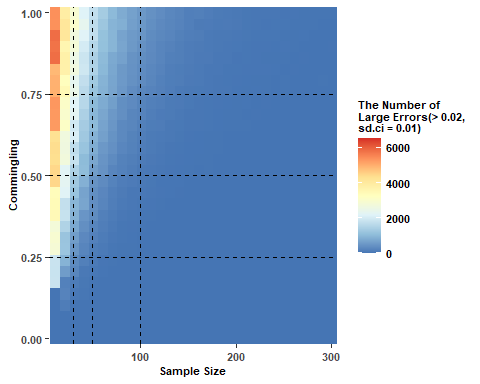
#### 8.3.1.3. Error 75% ~ sample size + commingling

plot3 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.01, ], aes(x = n, y = c, fill = percentile\_75)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_percentile\_75\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "75th Percentile   
(sd.ci = 0.01)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot3



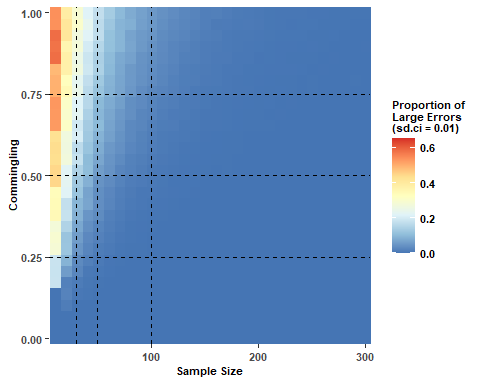
#### 8.3.1.4. Large error count ~ sample size + commingling

plot4 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.01, ], aes(x = n, y = c, fill = abs\_error\_count)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_no\_large\_error\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "The Number of\  
Large Errors(> 0.02,\  
sd.ci = 0.01)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot4



#### 8.3.1.5. Proportion of large error ~ sample size + commingling

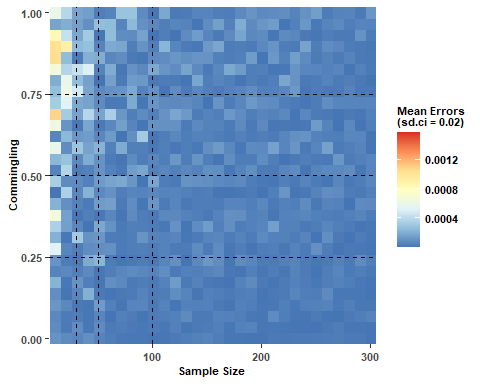
plot5 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.01, ], aes(x = n, y = c, fill = prop\_large\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_prop\_large\_error\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Proportion of\  
Large Errors\  
(sd.ci = 0.01)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot5



### 8.3.2. Variation = 0.02

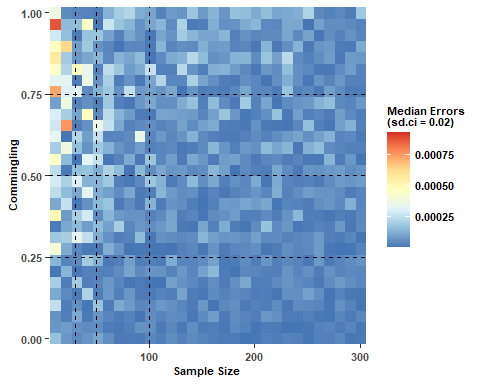
#### 8.3.2.1. Mean error ~ sample size + commingling

plot6 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.02, ], aes(x = n, y = c, fill = mean\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 color\_scale\_mean\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Mean Errors\  
(sd.ci = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot6



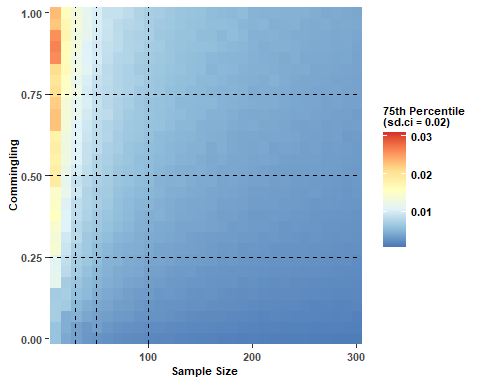
#### 8.3.2.2. Median absolute error ~ sample size + commingling

plot7 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.02, ], aes(x = n, y = c, fill = median\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_median\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Median Errors\  
(sd.ci = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot7



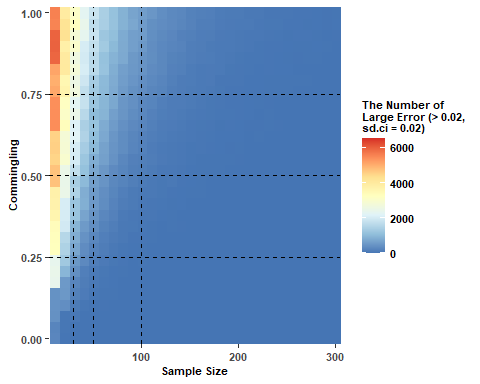
#### 8.3.2.3. Error 75% ~ sample size + commingling

plot8 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.02, ], aes(x = n, y = c, fill = percentile\_75)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_percentile\_75\_ci+  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "75th Percentile\  
(sd.ci = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot8



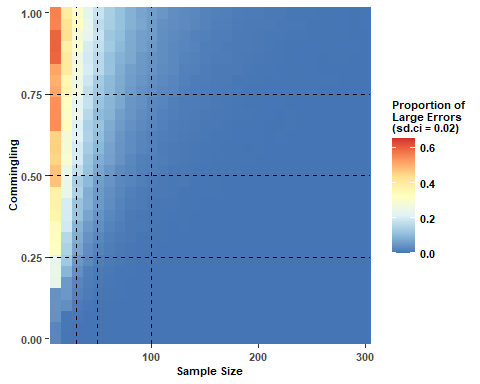
#### 8.3.2.4. Large error count ~ sample size + commingling

plot9 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.02, ], aes(x = n, y = c, fill = abs\_error\_count)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_no\_large\_error\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "The Number of\  
Large Error (> 0.02,\  
sd.ci = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot9



#### 8.3.2.5. Proportion of large error ~ sample size + commingling

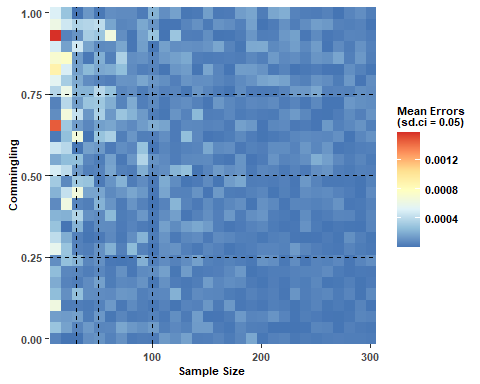
plot10 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.02, ], aes(x = n, y = c, fill = prop\_large\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_prop\_large\_error\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Proportion of\  
Large Errors\  
(sd.ci = 0.02)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot10



### 8.3.3. Variation = 0.05

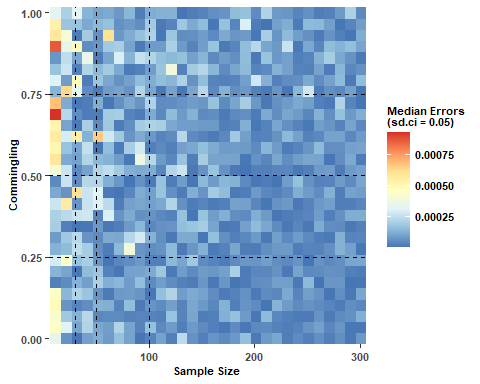
#### 8.3.3.1. Mean error ~ sample size + commingling

plot11 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.05, ], aes(x = n, y = c, fill = mean\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', linewidth = 0.5, linetype = 'dashed') +  
 color\_scale\_mean\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Mean Errors\  
(sd.ci = 0.05)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot11



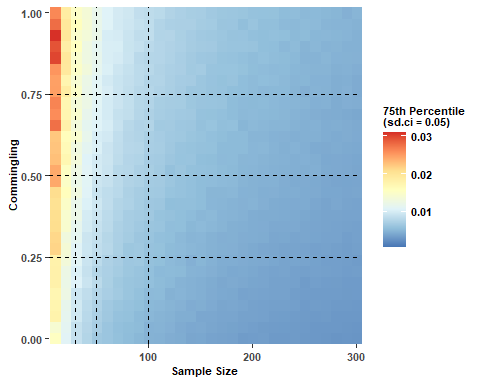
#### 8.3.3.2. Median absolute error ~ sample size + commingling

plot12 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.05, ], aes(x = n, y = c, fill = median\_abs\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_median\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Median Errors\  
(sd.ci = 0.05)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot12



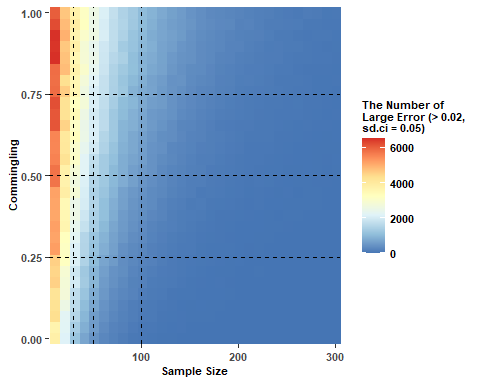
#### 8.3.3.3. Error 75% ~ sample size + commingling

plot13 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.05, ], aes(x = n, y = c, fill = percentile\_75)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_percentile\_75\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "75th Percentile\  
(sd.ci = 0.05)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot13



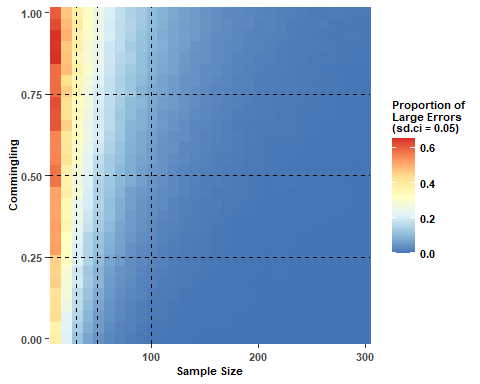
#### 8.3.3.4. Large error count ~ sample size + commingling

plot14 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.05, ], aes(x = n, y = c, fill = abs\_error\_count)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_no\_large\_error\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "The Number of\  
Large Error (> 0.02,\  
sd.ci = 0.05)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot14



#### 8.3.3.5. Proportion of large error ~ sample size + commingling

plot15 <- ggplot(results\_df\_ci[results\_df\_ci$v == 0.05, ], aes(x = n, y = c, fill = prop\_large\_error)) +  
 geom\_raster() +  
 geom\_vline(xintercept = simulated\_n, color = 'black', size = 0.5, linetype = 'dashed') +  
 geom\_hline(yintercept = simulated\_c, color = 'black', size = 0.5, linetype = 'dashed') +  
 color\_scale\_prop\_large\_error\_ci +  
 xlab('Sample Size') +  
 ylab('Commingling') +  
 labs(fill = "Proportion of\  
Large Errors\  
(sd.ci = 0.05)") +  
 coord\_cartesian(expand = FALSE) +  
 theme\_bw() +  
 theme(panel.grid = element\_blank(), panel.border = element\_blank(),  
 axis.text = element\_text(size = 8, face = 'bold'), # Adjust axis text size  
 axis.title = element\_text(size = 8, face = 'bold'), # Adjust axis title size  
 legend.title = element\_text(size = 8, face = 'bold'),  
 legend.text = element\_text(size = 8, face = 'bold'))  
  
# Print  
plot15



## 8.4. Combine the plots

### 8.4.1. Combine the BI plots

# Combine the plots using grid.arrange  
comb\_bi <- grid.arrange(plot01, plot06, plot011,  
 plot02, plot07, plot012,  
 plot03, plot08, plot013, ncol = 3)

A picture containing text, screenshot

Description automatically generated

### 8.4.2. Combine the CI plots

# Combine the plots using grid.arrange  
comb\_ci <- grid.arrange(plot1, plot6, plot11,  
 plot2, plot7, plot12,  
 plot3, plot8, plot13, ncol = 3)

A picture containing text, screenshot

Description automatically generated

### 8.4.2. Combine the BI and CI plots

# Combine the plots using grid.arrange  
comb\_bc <- grid.arrange(plot05, plot5,  
 plot010,plot10,  
 plot015,plot15, ncol = 2)

A picture containing text, screenshot, line, parallel

Description automatically generated