Plots - Lithic analysis Balver Höhle

Lisa Schunk

2021-02-04

# Goal of the script

This script reads the xlsx file (derived data) containing all the information gained through a lithic analysis.  
The script will:

1. Read the xlsx file
2. Plot all relevant variables in various combinations
3. Save the plot as PDFs

dir\_in <- "analysis/Balve/derived\_data/"  
dir\_out <- "analysis/Balve/plots/"

Raw data must be located in “analysis/Balve/derived\_data/”.  
Formatted data will be saved in “analysis/Balve/plots/”. The knit directory for this script is the project directory.

# Load packages

library(openxlsx)

Warning: package 'openxlsx' was built under R version 4.0.3

library(readxl)  
library(R.utils)  
library(tools)  
library(chron)  
library(ggplot2)

Warning: package 'ggplot2' was built under R version 4.0.3

library(wesanderson)

Warning: package 'wesanderson' was built under R version 4.0.3

library(dplyr)

Warning: package 'dplyr' was built under R version 4.0.3

library(ggsci)

# Get name, path and information of the file

data\_file <- list.files(dir\_in, pattern = "\\.xlsx$", full.names = TRUE)  
md5\_in <- md5sum(data\_file)  
info\_in <- data.frame(files = basename(names(md5\_in)), checksum = md5\_in,   
 row.names = NULL)

The checksum (MD5 hashes) of the imported files are:

files checksum  
1 Balve\_lithic\_analysis.xlsx a4d7b432556150ee5eb0e3ea35ac69d1

# Load data into R object

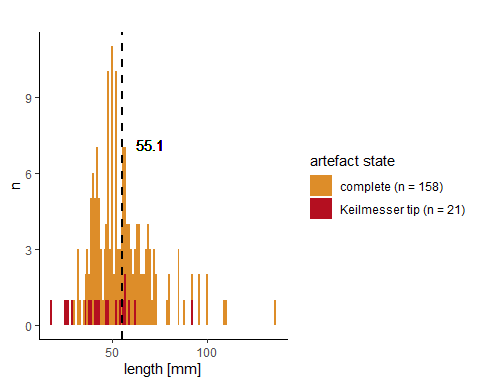
imp\_data <- read.xlsx(xlsxFile = data\_file, sheet = 1, startRow = 1,   
 colNames = TRUE,  
 rowNames = FALSE, skipEmptyRows = FALSE)

# Data analsysis - plots

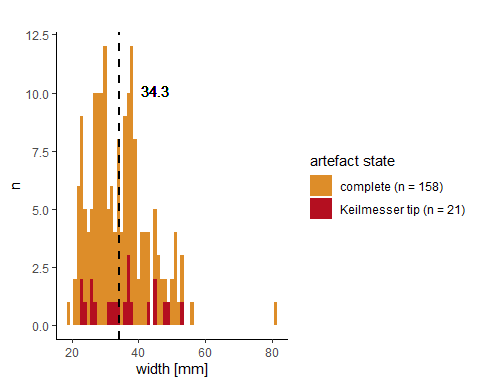
## Histogram

### Histogram dimensions - Keilmesser

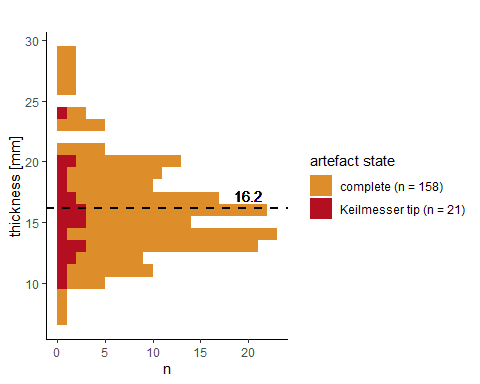
# Load data sheet Keilmesser   
KM\_dim <- read.xlsx(xlsxFile = data\_file, sheet = 3)   
KM\_dim <- KM\_dim [ , ] %>% arrange(artefact.state)   
KM.tip\_dim <- KM\_dim[-c(180:191), ]  
  
# Keilmesser length   
# Calculates the mean value for the plot and ascribes the N value   
mean\_length <- mean(KM.tip\_dim$length, na.rm = TRUE)  
n <- doBy::summaryBy(length ~ artefact.state, data = KM.tip\_dim, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Histogram Keilmesser length  
KM.length <- ggplot(KM.tip\_dim, aes(x = length, fill = artefact.state)) +   
 geom\_histogram(binwidth = 1) +  
 labs(x = "length [mm]", y = "n", title = "",   
 fill = "artefact state", size = 12) +  
 theme\_classic() +  
 geom\_vline(aes(xintercept = mean\_length), linetype="dashed",   
 size = 1) +  
 geom\_text(aes(y = mean\_length, x = 70, label = round(mean\_length,   
 1)), nudge\_y = -48) +  
 scale\_fill\_manual(values = wes\_palette(n = 2, name =  
 "FantasticFox1", type = "continuous"), labels = tag)   
  
print(KM.length)



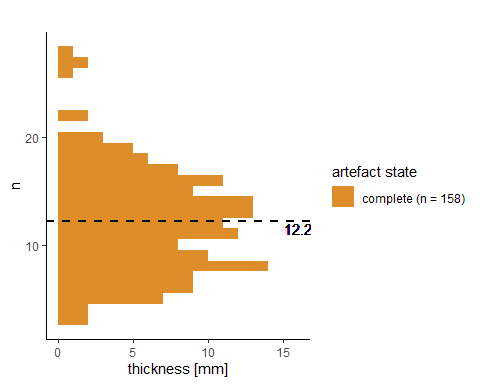
file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.length", ".pdf")  
ggsave(filename = file\_out, plot = KM.length, path = dir\_out, device = "pdf",  
 width = 170, height = 250, units = "mm")  
  
  
# Keilmesser width  
# Calculates the mean value for the plot and ascribes the N value   
mean\_width <- mean(KM.tip\_dim$width, na.rm = TRUE)  
n <- doBy::summaryBy(width ~ artefact.state, data = KM.tip\_dim, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Histogram Keilmesser width  
KM.width <- ggplot(KM.tip\_dim, aes(x = width, fill = artefact.state)) +   
 geom\_histogram(binwidth = 1) +  
 labs(x = "width [mm]", y = "n", title = "", fill = "artefact state",  
 size = 12) +  
 theme\_classic() +  
 geom\_vline(aes(xintercept = mean\_width), linetype = "dashed",   
 size = 1) +  
 geom\_text(aes(y = mean\_length, x = 45, label = round(mean\_width, 1)),  
 nudge\_y = -45) +  
 scale\_fill\_manual(values = wes\_palette(n = 2, name = "FantasticFox1",  
 type = "continuous"), labels = tag)   
  
print(KM.width)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.width", ".pdf")  
ggsave(filename = file\_out, plot = KM.width, path = dir\_out, device = "pdf",  
 width = 170, height = 250, units = "mm")  
  
  
# Keilmesser thickness  
# Calculates the mean value for the plot and ascribes the N value   
mean\_thickness <- mean(KM.tip\_dim$thickness, na.rm = TRUE)  
n <- doBy::summaryBy(thickness ~ artefact.state, data = KM.tip\_dim, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Histogram Keilmesser thickness   
KM.thickness <- ggplot(KM.tip\_dim, aes(y = thickness, fill = artefact.state)) +   
 geom\_histogram(binwidth = 1) +  
 labs(y = "thickness [mm]", x = "n", title = "",   
 fill = "artefact state", size = 12) +  
 theme\_classic() +  
 geom\_hline(aes(yintercept = mean\_thickness),   
 linetype = "dashed", size = 1) +  
 geom\_text(aes(y = mean\_thickness, x = 20,   
 label = round(mean\_thickness, 1)), nudge\_y = 1) +  
 scale\_fill\_manual(values = wes\_palette(n = 2,   
 name = "FantasticFox1", type = "continuous"), labels = tag)   
  
print(KM.thickness)



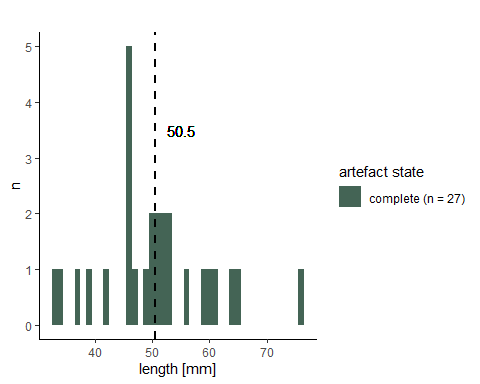
file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.thickness", ".pdf")  
ggsave(filename = file\_out, plot = KM.thickness, path = dir\_out, device = "pdf",  
 width = 250, height = 170, units = "mm")  
  
  
# Keilmesser Back   
# Load data sheet Keilmesser thickness back   
KM\_back <- read.xlsx(xlsxFile = data\_file, sheet = 22)   
KM\_back <- KM\_back[-c(159:191), ]  
  
# Calculates the mean value for the plot and ascribes the N value   
mean\_KM\_back <- mean(KM\_back$thickness.back, na.rm = TRUE)  
n <- doBy::summaryBy(thickness.back ~ artefact.state, data = KM\_back,   
 FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]],   
 " (n = ", n[[2]], ")"))  
  
# Histogram Keilmesser thickness back   
KM.back <- ggplot(KM\_back, aes(y = thickness.back, fill = artefact.state)) +   
 geom\_histogram(binwidth = 1) +  
 labs(x = "thickness [mm]", y = "n", title = "",   
 fill = "artefact state", size = 12) +  
 theme\_classic() +  
 geom\_hline(aes(yintercept = mean\_KM\_back), linetype = "dashed",  
 size = 1) +  
 geom\_text(aes(y = mean\_KM\_back, x = 16,   
 label = round(mean\_KM\_back, 1)), nudge\_y = -0.7) +  
 scale\_fill\_manual(values = wes\_palette(n = 2,   
 name = "FantasticFox1", type = "continuous"), labels = tag)   
  
print(KM.back)



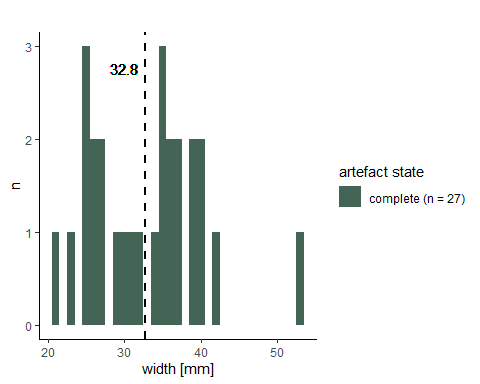
file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.back", ".pdf")  
ggsave(filename = file\_out, plot = KM.back, path = dir\_out, device = "pdf",   
 width = 250, height = 170, units = "mm")

### Histogram dimensions - Pradnik scraper

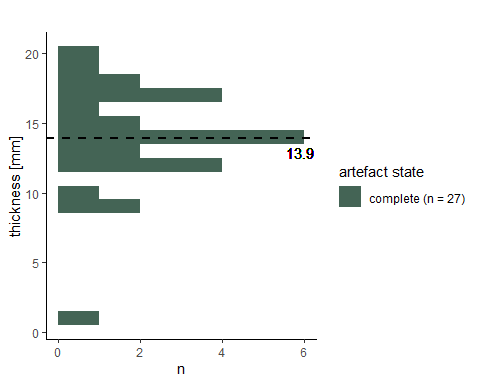
# Load data sheet Pradnik scraper   
PS\_dim <- read.xlsx(xlsxFile = data\_file, sheet = 7)   
  
# Pradnik scraper length   
# Calculates the mean value for the plot and ascribes the N value   
mean\_PS\_length <- mean(PS\_dim$length, na.rm = TRUE)  
n <- doBy::summaryBy(length ~ artefact.state, data = PS\_dim, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]],  
 " (n = ", n[[2]], ")"))  
  
# Histogram Pradnik scraper length  
PS.length <- ggplot(PS\_dim, aes(x = length, fill = artefact.state)) +   
 geom\_histogram(binwidth = 1) +  
 labs(x = "length [mm]", y = "n", title = "",   
 fill = "artefact state", size = 12) +  
 theme\_classic() +  
 geom\_vline(aes(xintercept = mean\_PS\_length), linetype = "dashed",  
 size = 1) +  
 geom\_text(aes(y = mean\_PS\_length, x = 55,   
 label = round(mean\_PS\_length, 1)), nudge\_y = -47) +  
 scale\_fill\_manual(values = wes\_palette(n = 1, name = "Chevalier1"),  
 labels = tag)   
  
print(PS.length)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "PS.length", ".pdf")  
ggsave(filename = file\_out, plot = PS.length, path = dir\_out, device = "pdf",  
 width = 170, height = 250, units = "mm")  
  
  
# Pradnik scraper width   
# Calculates the mean value for the plot and ascribes the N value   
mean\_PS\_width <- mean(PS\_dim$width, na.rm = TRUE)  
n <- doBy::summaryBy(width ~ artefact.state, data = PS\_dim, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]],  
 " (n = ", n[[2]], ")"))  
  
# Histogram Pradnik scraper width  
PS.width <- ggplot(PS\_dim, aes(x = width, fill = artefact.state)) +   
 geom\_histogram(binwidth = 1) +  
 labs(x = "width [mm]", y = "n", title = "", fill = "artefact state",  
 size = 12) +  
 theme\_classic() +  
 geom\_vline(aes(xintercept = mean\_PS\_width), linetype="dashed",  
 size = 1) +  
 geom\_text(aes(y = mean\_PS\_width, x = 30, label = round(mean\_PS\_width,  
 1)), nudge\_y = -30) +  
 scale\_fill\_manual(values = wes\_palette(n = 1, name = "Chevalier1"), labels = tag)   
   
print(PS.width)

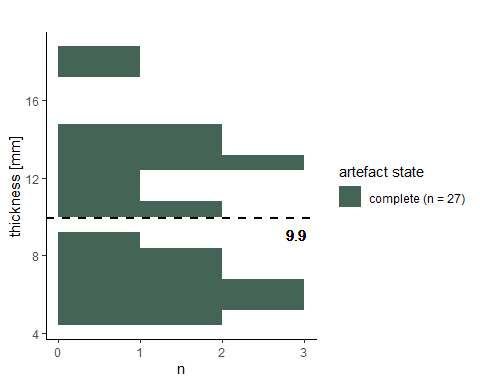


file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "PS.width", ".pdf")  
ggsave(filename = file\_out, plot = PS.width, path = dir\_out, device = "pdf",  
 width = 170, height = 250, units = "mm")  
  
  
# Pradnik scraper thickness  
# Calculates the mean value for the plot and ascribes the N value   
mean\_PS\_thickness <- mean(PS\_dim$thickness, na.rm = TRUE)  
n <- doBy::summaryBy(thickness ~ artefact.state, data = PS\_dim, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]],   
 " (n = ", n[[2]], ")"))  
  
# Histogram Pradnik scraper thickness   
PS.thickness <- ggplot(PS\_dim, aes(y = thickness, fill = artefact.state)) +   
 geom\_histogram(binwidth = 1) +  
 labs(y = "thickness [mm]", x = "n", title = "",   
 fill = "artefact state", size = 12) +  
 theme\_classic() +  
 geom\_hline(aes(yintercept = mean\_PS\_thickness),   
 linetype = "dashed", size = 1) +  
 geom\_text(aes(y = mean\_PS\_thickness, x = 5.9,   
 label = round(mean\_PS\_thickness, 1)), nudge\_y = -1) +  
 scale\_fill\_manual(values = wes\_palette(n = 1, name =  
 "Chevalier1"), labels = tag)   
  
print(PS.thickness)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "PS.thickness", ".pdf")  
ggsave(filename = file\_out, plot = PS.thickness, path = dir\_out, device = "pdf",  
 width = 250, height = 170, units = "mm")  
  
  
  
# Back Pradnik scraper thickness  
# Load data sheet Pradnik scraper thickness back   
PS\_back <- read.xlsx(xlsxFile = data\_file, sheet = 23)   
  
# Calculates the mean value for the plot and ascribes the N value   
mean\_PS\_back <- mean(PS\_back$thickness.back, na.rm = TRUE)  
n <- doBy::summaryBy(thickness.back ~ artefact.state, data = PS\_back,   
 FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Histogram Pradnik scraper thickness back   
PS.back <- ggplot(PS\_back, aes(y = thickness.back, fill = artefact.state)) +   
 geom\_histogram(binwidth = 0.8) +  
 labs(y = "thickness [mm]", x = "n", title = "",   
 fill = "artefact state", size = 12) +  
 theme\_classic() +  
 geom\_hline(aes(yintercept = mean\_PS\_back), linetype = "dashed",  
 size = 1) +  
 geom\_text(aes(y = mean\_PS\_back, x = 2.9,   
 label = round(mean\_PS\_back, 1)), nudge\_y = -0.8) +  
 scale\_fill\_manual(values = wes\_palette(n = 1, name =  
 "Chevalier1"), labels = tag)   
  
print(PS.back)

Warning: Removed 1 rows containing non-finite values (stat\_bin).

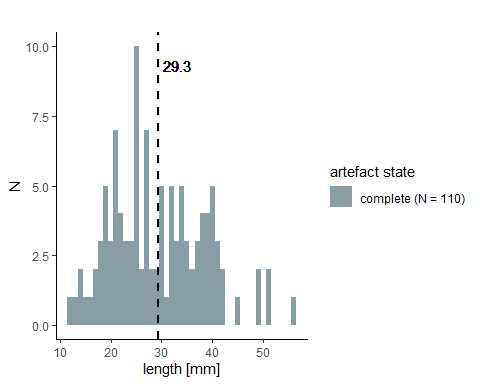


file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "PS.back", ".pdf")  
ggsave(filename = file\_out, plot = PS.back, path = dir\_out, device = "pdf",   
 width = 250, height = 170, units = "mm")

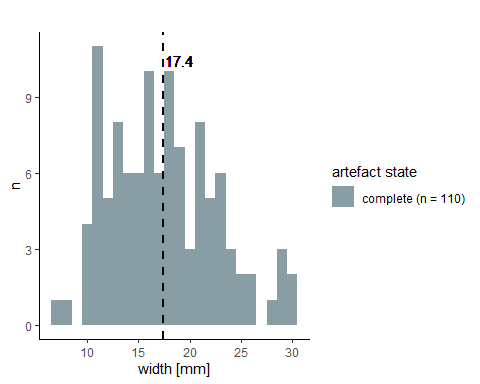
Warning: Removed 1 rows containing non-finite values (stat\_bin).

### Histogram dimension - Lateral sharpening spall

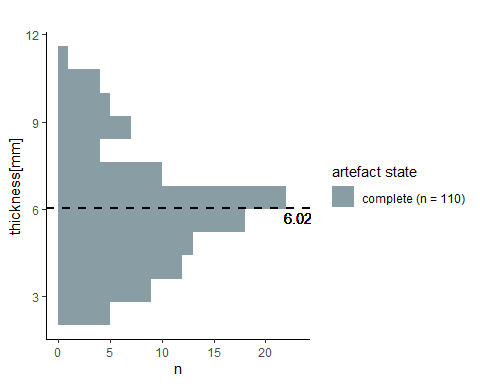
# Load data sheet lateral sharpening spall   
LSS\_dim <- read.xlsx(xlsxFile = data\_file, sheet = 8)   
LSS\_dim <- LSS\_dim [-c(111:117), ]  
  
# Lateral sharpening spall length   
# Calculates the mean value for the plot and ascribes the N value   
mean\_LSS\_length <- mean(LSS\_dim$length, na.rm = TRUE)  
n <- doBy::summaryBy(length ~ artefact.state, data = LSS\_dim, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (N = ", n[[2]],  
 ")"))  
   
# Histogram lateral sharpening spall length  
LSS.length <- ggplot(LSS\_dim, aes(x = length, fill = artefact.state)) +   
 geom\_histogram(binwidth = 1) +  
 labs(x = "length [mm]", y = "N", title = "",   
 fill = "artefact state", size = 12) +  
 theme\_classic() +  
 geom\_vline(aes(xintercept = mean\_LSS\_length), linetype = "dashed",  
 size = 1) +  
 geom\_text(aes(y = mean\_LSS\_length, x = 33,   
 label = round(mean\_LSS\_length, 1)), nudge\_y = -20) +  
 scale\_fill\_manual(values = wes\_palette(n = 1, name = "Royal1"),  
 labels = tag)   
  
print(LSS.length)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "LSS.length", ".pdf")  
ggsave(filename = file\_out, plot = LSS.length, path = dir\_out, device = "pdf",  
 width = 170, height = 250, units = "mm")  
  
  
# Lateral sharpening spall width   
# Calculates the mean value for the plot and ascribes the N value   
mean\_LSS\_width <- mean(LSS\_dim$width, na.rm = TRUE)  
n <- doBy::summaryBy(width ~ artefact.state, data = LSS\_dim, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Histogram lateral sharpening spall width  
LSS.width <- ggplot(LSS\_dim, aes(x = width, fill = artefact.state)) +   
 geom\_histogram(binwidth = 1) +  
 labs(x = "width [mm]", y = "n", title = "", fill = "artefact state",  
 size = 12) +  
 theme\_classic() +  
 geom\_vline(aes(xintercept = mean\_LSS\_width), linetype = "dashed",  
 size = 1) +  
 geom\_text(aes(y = mean\_LSS\_width, x = 19,   
 label = round(mean\_LSS\_width, 1)), nudge\_y = -7) +  
 scale\_fill\_manual(values = wes\_palette(n = 1, name = "Royal1"),  
 labels = tag)   
  
print(LSS.width)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "LSS.width", ".pdf")  
ggsave(filename = file\_out, plot = LSS.width, path = dir\_out, device = "pdf",  
 width = 170, height = 250, units = "mm")  
  
  
# Lateral sharpening spall thickness  
# Calculates the mean value for the plot and ascribes the N value   
mean\_LSS\_thickness <- mean(LSS\_dim$thickness, na.rm = TRUE)  
n <- doBy::summaryBy(thickness ~ artefact.state, data = LSS\_dim, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Histogram lateral sharpening spall thickness   
LSS.thickness <- ggplot(LSS\_dim, aes(y = thickness, fill = artefact.state)) +   
 geom\_histogram(binwidth = 0.8) +  
 labs(y = "thickness[mm]", x = "n", title = "",   
 fill = "artefact state", size = 12) +  
 theme\_classic() +   
 geom\_hline(aes(yintercept = mean\_LSS\_thickness),   
 linetype = "dashed", size = 1) +  
 geom\_text(aes(y = mean\_LSS\_thickness, x = 23.1,   
 label = round(mean\_LSS\_thickness, 2)), nudge\_y = -0.3) +  
 scale\_fill\_manual(values = wes\_palette(n = 1, name = "Royal1"),  
 labels = tag)   
  
print(LSS.thickness)

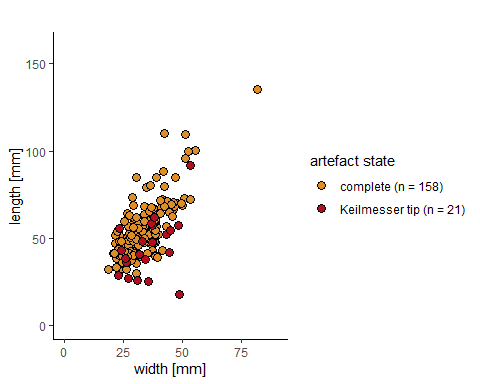


file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "LSS.thickness",  
 ".pdf")  
ggsave(filename = file\_out, plot = LSS.thickness, path = dir\_out,   
 device = "pdf", width = 250, height = 170, units = "mm")

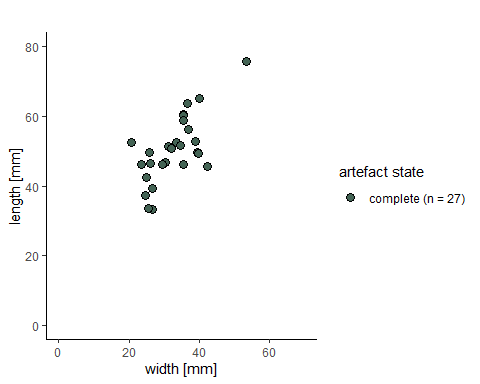
## Scatterplot

### Length-width ratio

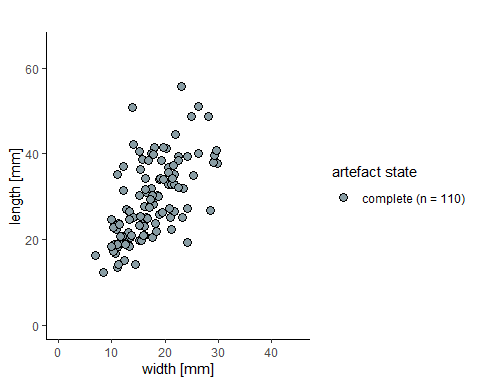
# Load data sheet Keilmesser   
KM\_comp\_dim <- read.xlsx(xlsxFile = data\_file, sheet = 6)   
  
# Keilmesser length VS width   
# Ascribes the N value   
n <- doBy::summaryBy(length + width ~ artefact.state, data = KM\_comp\_dim,   
 FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Scatterplot Keilmesser (complete + point) length VS width   
KM.length\_width <- ggplot(KM\_comp\_dim, aes(y = length, x = width,   
 fill = artefact.state)) +  
 geom\_point(size = 3, shape = 21) +  
 labs(y = "length [mm]", x = "width [mm]", title = "",   
 fill = "artefact state", size = 12) +  
 xlim(0, 90) + ylim(0, 160) +  
 theme\_classic() +  
 scale\_fill\_manual(values = wes\_palette(n = 2,   
 name = "FantasticFox1", type = "continuous"), labels = tag)   
  
print(KM.length\_width)



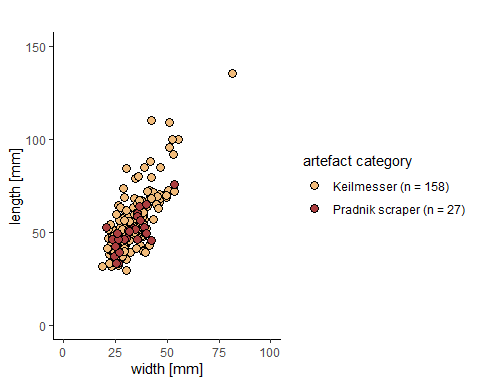
file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.length\_width",  
 ".pdf")  
ggsave(filename = file\_out, plot = KM.length\_width, path = dir\_out,   
 device = "pdf", width = 170, height = 250, units = "mm")  
  
  
# Pradnik scraper length VS width   
# Ascribes the N value   
n <- doBy::summaryBy(length + width ~ artefact.state, data = PS\_dim, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Scatterplot Pradnik scraper length VS width   
PS.length\_width <- ggplot(PS\_dim, aes(y = length, x = width,   
 fill = artefact.state)) +  
 geom\_point(size = 3, shape = 21) +  
 labs(y = "length [mm]", x = "width [mm]", title = "",   
 fill = "artefact state", size = 12) +  
 xlim(0, 70) + ylim(0, 80) +  
 theme\_classic() +  
 scale\_fill\_manual(values = wes\_palette(n = 6,   
 name = "Chevalier1", type = "continuous"), labels = tag)   
  
print(PS.length\_width)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "PS.length\_width",  
 ".pdf")  
ggsave(filename = file\_out, plot = PS.length\_width, path = dir\_out,   
 device = "pdf", width = 170, height = 250, units = "mm")  
  
  
# Lateral sharpening spall length VS width   
# Defines only the rows with complete LSS   
LSS.comp\_dim <- LSS\_dim[1:110,]   
  
# Lateral sharpening spall length VS width   
# Ascribes the N value   
n <- doBy::summaryBy(length + width ~ artefact.state, data = LSS.comp\_dim,   
 FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Scatterplot lateral sharpening spall length VS width   
LSS.length\_width <- ggplot(LSS.comp\_dim, aes(y = length, x = width,   
 fill = artefact.state)) +  
 geom\_point(size = 3, shape = 21) +  
 labs(y = "length [mm]", x = "width [mm]", title = "",   
 fill = "artefact state", size = 12) +  
 xlim(0, 45) + ylim(0, 65) +  
 theme\_classic() +  
 scale\_fill\_manual(values = wes\_palette(n = 2,   
 name = "Royal1", type = "continuous"), labels = tag)   
  
print(LSS.length\_width)

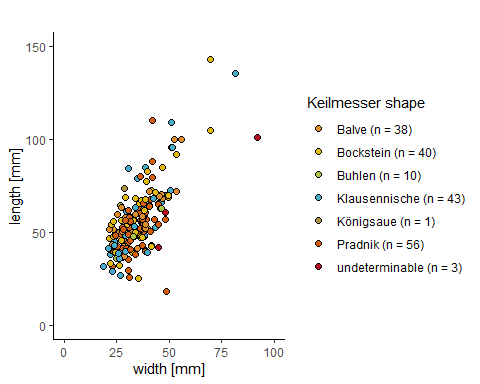


file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "LSS.length\_width",  
 ".pdf")  
ggsave(filename = file\_out, plot = LSS.length\_width, path = dir\_out,   
 device = "pdf", width = 170, height = 250, units = "mm")  
  
  
# Keilmesser (complete) + Pradnik scraper length VS width  
# Load data sheet dimensions   
dim <- read.xlsx(xlsxFile = data\_file, sheet = 2)   
# Defines only the relevant rows   
KM.PS\_dim <- dim[c(1:191, 309:335), ] %>% arrange(artefact.state)  
KM.PS\_dim <- KM.PS\_dim[1:185,]  
  
# Keilmesser (complete) + Pradnik scraper length VS width   
# Ascribes the N value   
n <- doBy::summaryBy(length + width ~ technological.class, data = KM.PS\_dim,   
 FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Scatterplot Keilmesser (complete) + Pradnik scraper length VS width  
KM.PS.length\_width <- ggplot(KM.PS\_dim, aes(y = length, x = width,   
 fill = technological.class)) +  
 geom\_point(size = 3, shape = 21) +  
 labs(y = "length [mm]", x = "width [mm]", title = "",  
 fill = "artefact category", size = 12) +  
 xlim(0, 100) + ylim(0, 150) +  
 theme\_classic() +  
 scale\_fill\_manual(values = wes\_palette(n = 3, name =  
 "GrandBudapest1", type = "continuous"), labels = tag)   
  
print(KM.PS.length\_width)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.PS.length\_width",  
 ".pdf")  
ggsave(filename = file\_out, plot = KM.PS.length\_width, path = dir\_out,   
 device = "pdf", width = 170, height = 250, units = "mm")  
   
# Keilmesser (complete): length-width combined with morpho type  
# Load data sheet Keilmesser morpho type   
KM\_morpho.type <- read.xlsx(xlsxFile = data\_file, sheet = 28)   
  
# Ascribes the N value   
n <- doBy::summaryBy(length + width ~ morpho.type, data = KM\_morpho.type,   
 FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
  
# Scatterplot Keilmesser (complete): length-width combined with morpho type   
KM.width\_length\_morpho <- ggplot(KM\_morpho.type, aes(y = length, x = width,   
 fill = morpho.type)) +  
 geom\_point(size = 2, shape = 21) +  
 labs(y = "length [mm]", x = "width [mm]", title = "",   
 fill = "Keilmesser shape", size = 12) +  
 xlim(0, 100) + ylim(0, 150) +  
 theme\_classic() +  
 scale\_fill\_manual(values = wes\_palette(n = 7,name =  
 "FantasticFox1", type = "continuous"), labels = tag)   
   
print(KM.width\_length\_morpho)

Warning: Removed 1 rows containing missing values (geom\_point).



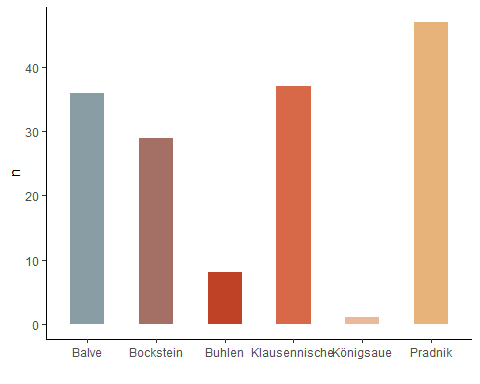
file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.width\_length\_morpho", ".pdf")  
ggsave(filename = file\_out, plot = KM.width\_length\_morpho, path = dir\_out,   
 device = "pdf", width = 170, height = 250, units = "mm")

Warning: Removed 1 rows containing missing values (geom\_point).

## Barplot

### Morphotype

# Keilmesser morpho type   
# Load data sheet Keilmesser morpho type   
KM\_morpho.type <- read.xlsx(xlsxFile = data\_file, sheet = 28)   
# Defines only the rows with complete Keilmesser   
KM\_morpho.type <- KM\_morpho.type[1:158,]  
  
# Barplot Keilmesser morpho type   
KM.morpho.type <- ggplot(data = KM\_morpho.type) + aes(x = morpho.type,   
 fill = morpho.type) +   
 geom\_bar(stat = "count", width = 0.5, fill = c("#899DA4", "#A46F65",  
 "#BF4226", "#D76848", "#EBB99A", "#E8B37B")) +  
 theme\_classic() +  
 theme(legend.position = "none") +   
 labs(x = " ", y = "n")   
   
print(KM.morpho.type)

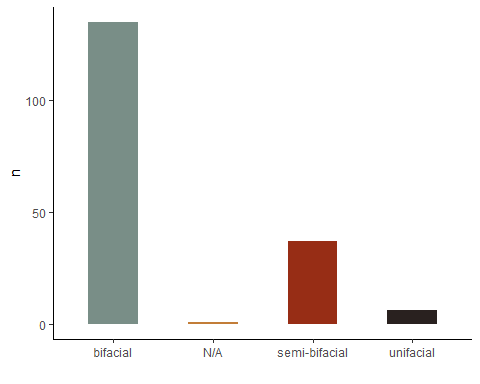


file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.morpho.type",  
 ".pdf")  
ggsave(filename = file\_out, plot = KM.morpho.type, path = dir\_out,   
 device = "pdf", width = 170, height = 200, units = "mm")

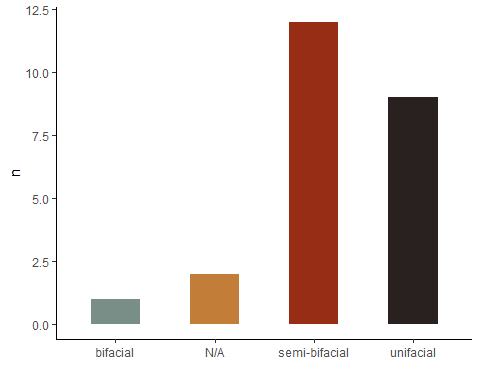
## Barplot

### Edge retouch

# Keilmesser edge retouch   
# Load data sheet Keilmesser edge retouch   
KM\_edge <- read.xlsx(xlsxFile = data\_file, sheet = 24)   
# Defines only the rows with complete Keilmesser and Keilmesser tips   
KM\_edge <- KM\_edge[-c(180:191),]  
  
# Barplot Keilmesser edge retouch   
KM.edge <- ggplot(data = KM\_edge) + aes(x = retouch.type.edge,   
 fill = retouch.type.edge) +   
 geom\_bar(stat = "count", width = 0.5, fill = c ("#798E87", "#C27D38",  
 "#972D15", "#29211F")) +  
 theme\_classic() +  
 theme(legend.position = "none") +   
 labs(x = " ", y = "n")   
   
print(KM.edge)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.edge", ".pdf")  
ggsave(filename = file\_out, plot = KM.edge, path = dir\_out, device = "pdf",   
 width = 170, height = 200, units = "mm")  
  
  
  
# Pradnik scraper edge retouch   
# Load data sheet Pradnik scraper edge retouch   
PS\_edge <- read.xlsx(xlsxFile = data\_file, sheet = 25)   
# Defines only the rows with complete Keilmesser and Keilmesser tips   
PS\_edge <- PS\_edge[-c(3:5),]  
  
# Barplot Pradnik scraper edge retouch   
PS.edge <- ggplot(data = PS\_edge) + aes(x = retouch.type.edge,   
 fill = retouch.type.edge) +   
 geom\_bar(stat = "count", width = 0.5, fill = c ("#798E87", "#C27D38",  
 "#972D15", "#29211F")) +  
 theme\_classic() +  
 theme(legend.position = "none") +   
 labs(x = " ", y = "n")   
   
print(PS.edge)

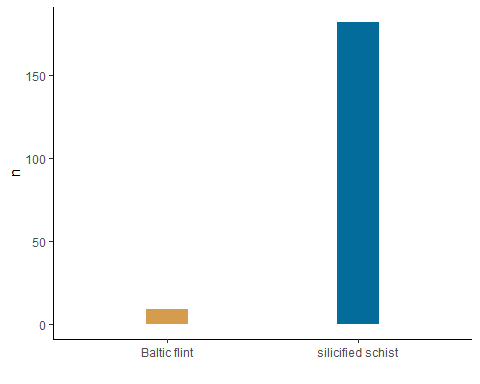


file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "PS.edge", ".pdf")  
ggsave(filename = file\_out, plot = PS.edge, path = dir\_out, device = "pdf",   
 width = 170, height = 200, units = "mm")

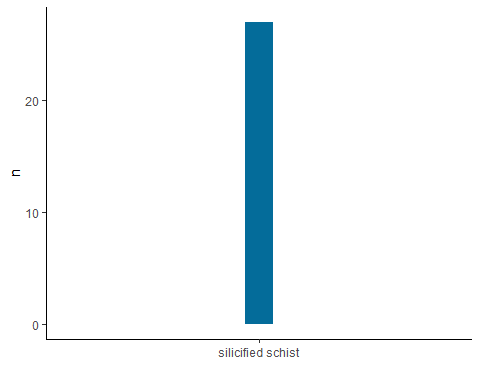
## Barplot

### Raw material

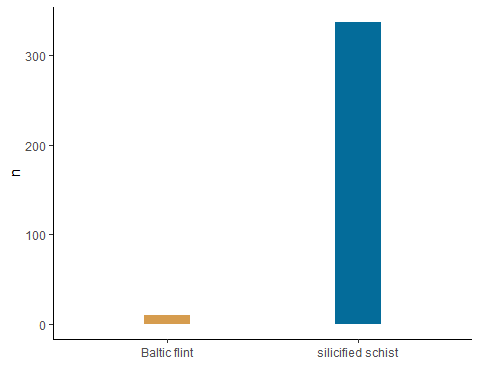
# Keilmesser raw material  
# Load data sheet Keilmesser raw material  
KM\_raw\_material <- read.xlsx(xlsxFile = data\_file, sheet = 16)   
  
# Barplot Keilmesser raw material   
KM.raw\_material <- ggplot(data = KM\_raw\_material) + aes(x = raw.material,   
 fill = raw.material) +   
 geom\_bar(stat = "count", width = 0.22, fill = c("#D69C4E", "#046C9A")) +  
 theme\_classic() +  
 theme(legend.position = "none") +   
 labs(x = " ", y = "n") +   
 scale\_x\_discrete(labels=c("Baltic flint", "silicified schist"))   
   
print(KM.raw\_material)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.raw\_material",  
 ".pdf")  
ggsave(filename = file\_out, plot = KM.raw\_material, path = dir\_out,   
 device = "pdf")  
  
  
# Pradnik scraper raw material  
# Load data sheet Pradnik scraper raw material  
PS\_raw\_material <- read.xlsx(xlsxFile = data\_file, sheet = 17)   
  
# Barplot Pradnik scraper raw material   
PS.raw\_material <- ggplot(data = PS\_raw\_material) + aes(x = raw.material,   
 fill = raw.material) +   
 geom\_bar(stat = "count", width = 0.08, fill = c("#046C9A")) +  
 theme\_classic() +  
 theme(legend.position = "none") +   
 labs(x = " ", y = "n") +   
 scale\_x\_discrete(labels=c("silicified schist"))   
   
print(PS.raw\_material)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "PS.raw\_material",  
 ".pdf")  
ggsave(filename = file\_out, plot = PS.raw\_material, path = dir\_out,   
 device = "pdf")  
  
  
# All tool types raw material  
# Load data sheet all tool types raw material  
all\_raw\_material <- read.xlsx(xlsxFile = data\_file, sheet = 1)   
  
# Barplot Pradnik scraper raw material   
all.raw\_material <- ggplot(data = all\_raw\_material) + aes(x = raw.material,   
 fill = raw.material) +   
 geom\_bar(stat = "count", width = 0.24, fill = c("#D69C4E", "#046C9A")) +  
 theme\_classic() +  
 theme(legend.position = "none") +   
 labs(x = " ", y = "n") +   
 scale\_x\_discrete(labels=c("Baltic flint", "silicified schist"))   
   
print(all.raw\_material)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "all.raw\_material",  
 ".pdf")  
ggsave(filename = file\_out, plot = all.raw\_material, path = dir\_out,   
 device = "pdf")

## Barplot

### Morphology back

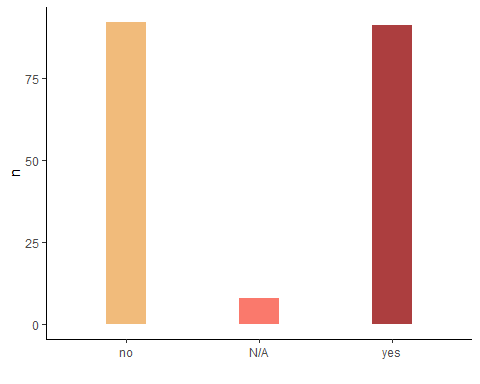
# Keilmesser morphology back   
# Load data sheet Keilmesser morphology back  
KM\_back <- read.xlsx(xlsxFile = data\_file, sheet = 22)   
  
# Barplot Keilmesser morphology back   
KM.back\_morpho <- ggplot(data = KM\_back) + aes(x = morphology.back,   
 fill = morphology.back) +   
 geom\_bar(stat = "count", width = 0.5, fill = c("#518BA0", "#497C80",  
 "#D69C4E", "#729394","#B9C7AD")) +  
 theme\_classic() +  
 theme(legend.position = "none") +   
 labs(x = " ", y = "n") +   
 scale\_x\_discrete(labels=c("cortex + partly retouched", "cortex/unworked",  
 "N/A", "partly retouched", "retouched"))   
   
   
print(KM.back\_morpho)  
file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.back\_morpho",  
 ".pdf")  
ggsave(filename = file\_out, plot = KM.back\_morpho, path = dir\_out,   
 device = "pdf",width = 250, height = 170, units = "mm")  
  
  
# Pradnik scraper morphology back   
# Load data sheet Pradnik scraper morphology back  
PS\_back <- read.xlsx(xlsxFile = data\_file, sheet = 23)   
  
# Barplot Pradnik scraper morphology back   
PS.back\_morpho <- ggplot(data = PS\_back) + aes(x = morphology.back,   
 fill = morphology.back) +   
 geom\_bar(stat = "count", width = 0.4, fill = c("#518BA0", "#497C80",  
 "#D69C4E", "#729394")) +  
 theme\_classic() +  
 theme(legend.position = "none") +   
 labs(x = " ", y = "n") +   
 scale\_x\_discrete(labels=c("cortex + partly retouched", "cortex/unworked",  
 "N/A", "partly retouched", "retouched")) +  
 scale\_fill\_manual(values = wes\_palette(n = 7, name = "Darjeeling2",   
 type = "continuous"))   
   
print(PS.back\_morpho)  
file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "PS.back\_morpho",  
 ".pdf")  
ggsave(filename = file\_out, plot = PS.back\_morpho, path = dir\_out,   
 device = "pdf",width = 250, height = 170, units = "mm")  
  
  
# Keilmesser blanks  
# Load data sheet Keilmesser blanks   
  
KM\_cortex\_blanks <- read.xlsx(xlsxFile = data\_file, sheet = 20)   
# Ascribes the N value   
n <- doBy::summaryBy(blank ~ cortex, data = KM\_cortex\_blanks, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]], " (n = ", n[[2]],  
 ")"))  
  
# Barplot Keilmesser blanks   
KM.cortex\_blanks <- ggplot(data = KM\_cortex\_blanks) + aes(x = blank,   
 fill = cortex) +   
 geom\_bar(stat = "count", width = 0.4) +  
 theme\_classic() +  
 labs(x = " ", y = "n") +   
 scale\_x\_discrete(labels=c("core", "flake", "N/A")) +  
 scale\_fill\_manual(values = wes\_palette(n = 7, name = "Darjeeling2",   
 type = "continuous"), labels = tag)   
   
print(KM.cortex\_blanks)  
file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]),   
 "KM.cortex\_blanks", ".pdf")  
ggsave(filename = file\_out, plot = KM.cortex\_blanks, path = dir\_out,   
 device = "pdf")  
  
  
# Pradnik scraper blanks  
# Load data sheet Pradnik scraper blanks   
PS\_cortex\_blanks <- read.xlsx(xlsxFile = data\_file, sheet = 21)   
  
# Ascribes the N value   
n <- doBy::summaryBy(blank ~ cortex, data = PS\_cortex\_blanks, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]],   
 " (n = ", n[[2]], ")"))  
  
# Barplot Pradnik scraper blanks   
PS.cortex\_blanks <- ggplot(data = PS\_cortex\_blanks) + aes(x = blank,   
 fill = cortex) +   
 geom\_bar(stat = "count", width = 0.3) +  
 theme\_classic() +  
 labs(x = " ", y = "n") +   
 scale\_x\_discrete(labels=c("core", "flake", "N/A")) +  
 scale\_fill\_manual(values = wes\_palette(n = 7, name = "Darjeeling2",   
 type = "continuous"), labels = tag)   
   
print(PS.cortex\_blanks)  
file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "PS.cortex\_blanks",  
 ".pdf")  
ggsave(filename = file\_out, plot = PS.cortex\_blanks, path = dir\_out,   
 = "pdf")

Error: <text>:97:17: unexpected '='  
96: ggsave(filename = file\_out, plot = PS.cortex\_blanks, path = dir\_out,   
97: =  
 ^

## Barplot

### Pradnik method

# Keilmesser application Pradnik method  
# Load data sheet Keilmesser Pradnik method  
KM\_Pradnik.method <- read.xlsx(xlsxFile = data\_file, sheet = 30)   
  
# Barplot Keilmesser Pradnik method   
KM.PM <- ggplot(data = KM\_Pradnik.method) + aes(x = application.Pradnik.method,  
 fill = application.Pradnik.method) +   
 geom\_bar(stat = "count", width = 0.3) +   
 theme\_classic() +  
 labs(x = " ", y = "n") +   
 theme(legend.position = "none") +   
 scale\_x\_discrete(labels=c("no", "N/A", "yes")) +  
 scale\_fill\_manual(values = wes\_palette(n = 5, name = "GrandBudapest1",   
 type = "continuous"))   
   
print(KM.PM)

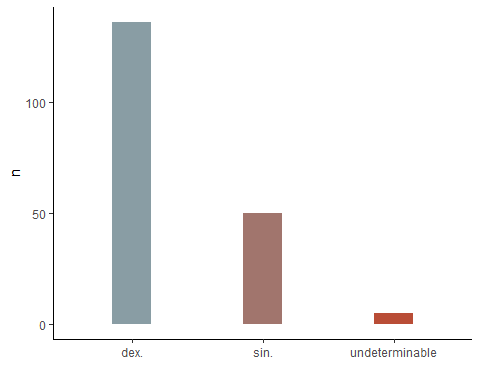


file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.PM", ".pdf")  
ggsave(filename = file\_out, plot = KM.PM, path = dir\_out, device = "pdf")

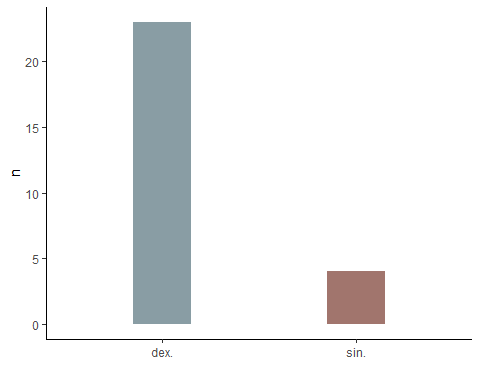
## Barplot

### Lateralisation

# Keilmesser lateralisation  
# Load data sheet Keilmesser lateralisation  
KM\_lateralisation <- read.xlsx(xlsxFile = data\_file, sheet = 32)   
 # Barplot Keilmesser lateralisation  
KM.lat <- ggplot(data = KM\_lateralisation) + aes(x = tool.lateralisation,   
 fill = tool.lateralisation) +   
 geom\_bar(stat = "count", width = 0.3) +   
 theme\_classic() +  
 labs(x = " ", y = "n") +   
 theme(legend.position = "none") +  
 scale\_fill\_manual(values = wes\_palette(n = 9, name = "Royal1",   
 type = "continuous"))   
   
print(KM.lat)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.lat", ".pdf")  
ggsave(filename = file\_out, plot = KM.lat, path = dir\_out, device = "pdf")  
  
  
# Pradnik scraper lateralisation  
# Load data sheet Pradnik scraper lateralisation  
PS\_lateralisation <- read.xlsx(xlsxFile = data\_file, sheet = 33)   
  
# Pradnik scraper lateralisation  
# Load data sheet Pradnik scraper lateralisation  
PS\_lateralisation <- read.xlsx(xlsxFile = data\_file, sheet = 33)  
# Barplot Keilmesser lateralisation  
PS.lat <- ggplot(data = PS\_lateralisation) + aes(x = tool.lateralisation,   
 fill = tool.lateralisation) +   
 geom\_bar(stat = "count", width = 0.3) +   
 theme\_classic() +  
 labs(x = " ", y = "n") +   
 theme(legend.position = "none") +  
 scale\_fill\_manual(values = wes\_palette(n = 9, name = "Royal1",   
 type = "continuous"))   
  
print(PS.lat)

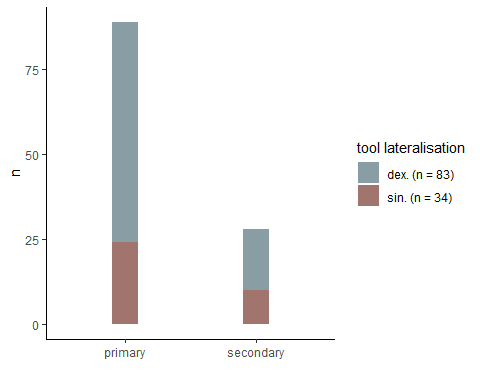


file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "PS.lat", ".pdf")  
ggsave(filename = file\_out, plot = PS.lat, path = dir\_out, device = "pdf")

## Barplot

### Barplot lateral resharpening spall type

# Lateral resharpening spall type  
# Load data sheet lateral resharpening spall type  
LSS\_type <- read.xlsx(xlsxFile = data\_file, sheet = 34)   
  
# Ascribes the N value   
n <- doBy::summaryBy(type.lateral.sharpening.spall ~ tool.lateralisation,   
 data = LSS\_type, FUN = length)  
tag <- gsub(pattern = "\_", replacement = " ", paste0(n[[1]],   
 " (n = ", n[[2]], ")"))  
  
# Barplot lateral resharpening spall type  
LSS.type <- ggplot(data = LSS\_type) + aes(x = type.lateral.sharpening.spall,   
 fill = tool.lateralisation) +   
 geom\_bar(stat = "count", width = 0.2) +   
 theme\_classic() +  
 labs(x = " ", y = "n") +   
 labs(fill = "tool lateralisation") +  
 scale\_x\_discrete() +  
 scale\_fill\_manual(values = wes\_palette(n = 9, name = "Royal1",   
 type = "continuous"), labels = tag)  
   
print(LSS.type)

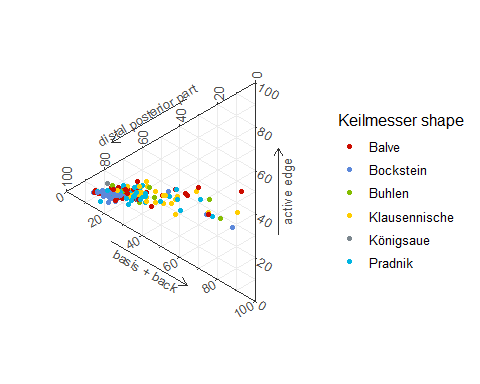


file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "LSS.type", ".pdf")  
ggsave(filename = file\_out, plot = LSS.type, path = dir\_out, device = "pdf")

## Ternary plot

### Perimeter

library(ggtern)  
# Perimeter Keilmesser   
# Load data sheet Keilmesser perimeter  
KM\_perimeter <- read.xlsx(xlsxFile = data\_file, sheet = 10)   
# Defines only the rows with complete Keilmesser   
KM\_perimeter <- KM\_perimeter[1:158,]  
  
# Ternary diagram Keilmesser perimeter  
KM.perimeter <- ggtern(data = KM\_perimeter, aes(x = perimeter.distal.posterior.part, y = perimeter.active.edge, z = perimeter.basis.back)) +  
 geom\_point(aes(colour = morpho.type)) +  
 theme\_bw() +  
 scale\_colour\_startrek() +  
 theme\_hidetitles() +  
 theme\_showarrows() +  
 xlab("distal posterior part")+   
 ylab("active edge")+  
 zlab("basis + back")+   
 labs(colour = "Keilmesser shape") +  
 tern\_limits(labels=c(0, 20, 40, 60, 80, 100)) +  
 theme\_rotate(degrees = 330)  
   
  
print(KM.perimeter)



file\_out <- paste0(file\_path\_sans\_ext(info\_in[["file"]]), "KM.perimeter", ".pdf")  
ggsave(filename = file\_out, plot = KM.perimeter, path = dir\_out, device = "pdf")

# sessionInfo() and RStudio version

sessionInfo()

R version 4.0.2 (2020-06-22)  
Platform: x86\_64-w64-mingw32/x64 (64-bit)  
Running under: Windows 10 x64 (build 19041)  
  
Matrix products: default  
  
locale:  
[1] LC\_COLLATE=German\_Germany.1252 LC\_CTYPE=German\_Germany.1252   
[3] LC\_MONETARY=German\_Germany.1252 LC\_NUMERIC=C   
[5] LC\_TIME=German\_Germany.1252   
  
attached base packages:  
[1] tools stats graphics grDevices utils datasets methods   
[8] base   
  
other attached packages:  
 [1] ggtern\_3.3.0 ggsci\_2.9 dplyr\_1.0.3 wesanderson\_0.3.6  
 [5] ggplot2\_3.3.3 chron\_2.3-56 R.utils\_2.10.1 R.oo\_1.24.0   
 [9] R.methodsS3\_1.8.1 readxl\_1.3.1 openxlsx\_4.2.3   
  
loaded via a namespace (and not attached):  
 [1] tidyselect\_1.1.0 xfun\_0.20 purrr\_0.3.4 lattice\_0.20-41   
 [5] latex2exp\_0.4.0 colorspace\_2.0-0 vctrs\_0.3.6 generics\_0.1.0   
 [9] doBy\_4.6.8 htmltools\_0.5.1.1 compositions\_2.0-1 yaml\_2.2.1   
[13] rlang\_0.4.10 pillar\_1.4.7 glue\_1.4.2 withr\_2.4.1   
[17] DBI\_1.1.1 plyr\_1.8.6 lifecycle\_0.2.0 robustbase\_0.93-7   
[21] stringr\_1.4.0 munsell\_0.5.0 gtable\_0.3.0 cellranger\_1.1.0   
[25] zip\_2.1.1 evaluate\_0.14 labeling\_0.4.2 knitr\_1.31   
[29] DEoptimR\_1.0-8 highr\_0.8 proto\_1.0.0 broom\_0.7.4   
[33] Rcpp\_1.0.6 scales\_1.1.1 backports\_1.2.1 farver\_2.0.3   
[37] gridExtra\_2.3 Deriv\_4.1.2 tensorA\_0.36.2 digest\_0.6.27   
[41] stringi\_1.5.3 grid\_4.0.2 magrittr\_2.0.1 tibble\_3.0.6   
[45] crayon\_1.4.0 tidyr\_1.1.2 pkgconfig\_2.0.3 bayesm\_3.1-4   
[49] ellipsis\_0.3.1 MASS\_7.3-51.6 Matrix\_1.2-18 assertthat\_0.2.1   
[53] rmarkdown\_2.6 R6\_2.5.0 compiler\_4.0.2

RStudio version 1.3.1056.

END OF SCRIPT