Filtering VCF files

Hazel Daniels (adapted from Javier Tabima)

30 March 2021

# Reading in data

vcfNA <- read.vcfR("ramorumNA1.vcf.gz")

## Scanning file to determine attributes.  
## File attributes:  
## meta lines: 66  
## header\_line: 67  
## variant count: 484461  
## column count: 144  
## Meta line 66 read in.  
## All meta lines processed.  
## gt matrix initialized.  
## Character matrix gt created.  
## Character matrix gt rows: 484461  
## Character matrix gt cols: 144  
## skip: 0  
## nrows: 484461  
## row\_num: 0  
## Processed variant 1000 … Processed variant: 484461  
## All variants processed

raw.vcfNA <- nrow(vcfNA)  
  
vcfEU <- read.vcfR("ramorumEU1.vcf.gz")

## Scanning file to determine attributes.  
## File attributes:  
## meta lines: 65  
## header\_line: 66  
## variant count: 499848  
## column count: 169  
## Meta line 65 read in.  
## All meta lines processed.  
## gt matrix initialized.  
## Character matrix gt created.  
## Character matrix gt rows: 499848  
## Character matrix gt cols: 169  
## skip: 0  
## nrows: 499848  
## row\_num: 0  
## Processed variant 1000 … Processed variant: 499848  
## All variants processed

raw.vcfEU <- nrow(vcfEU)

# Calculating the number of polymorphic sites and removing all non-biallelic sites

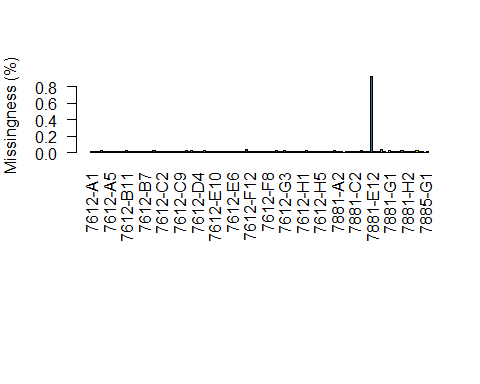
vcfNA <- vcfNA[is.polymorphic(vcfNA, na.omit = T)]  
poly.vcfNA <- nrow(vcfNA)  
vcfNA <- vcfNA[is.biallelic(vcfNA)]  
biallelic.vcfNA <- nrow(vcfNA)  
  
vcfEU <- vcfEU[is.polymorphic(vcfEU, na.omit = T)]  
poly.vcfEU <- nrow(vcfEU)  
vcfEU <- vcfEU[is.biallelic(vcfEU)]  
biallelic.vcfEU <- nrow(vcfEU)

# Backing up files

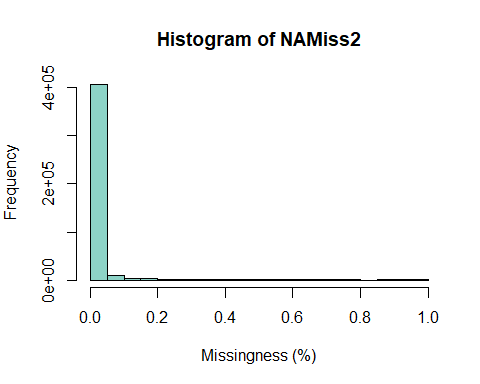
dp.vcNA <- extract.gt(vcfNA, element = "DP", as.numeric = T)  
gt.vcNA <- extract.gt(vcfNA, element = "GT", as.numeric = T)  
  
# Backup  
dp.vc.bk.na <- dp.vcNA  
gt.vc.bk.na <- gt.vcNA  
  
dp.vcEU <- extract.gt(vcfEU, element = "DP", as.numeric = T)  
gt.vcEU <- extract.gt(vcfEU, element = "GT", as.numeric = T)  
  
# Backup  
dp.vc.bk.eu <- dp.vcEU  
gt.vc.bk.eu <- gt.vcEU

# Visualizing missing data

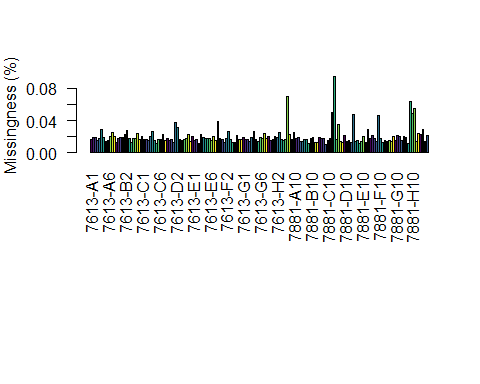
palette(viridis(n=12))  
  
NAMiss <- apply(gt.vcNA, MARGIN = 2, function(x){ sum(is.na(x)) })  
NAMiss <- NAMiss/nrow(vcfNA)  
  
par(mar = c(12,4,4,2))  
barplot(NAMiss, las = 2, col = 1:12)  
title(ylab = "Missingness (%)")



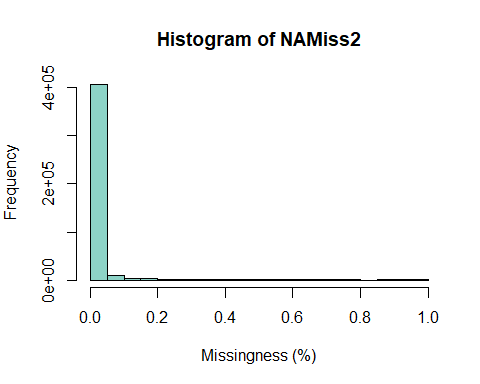
par(mar = c(5,4,4,2))  
NAMiss2 <- apply(gt.vcNA, MARGIN = 1, function(x){ sum(is.na(x)) })  
NAMiss2 <- NAMiss2/ncol(vcfNA@gt[,-1])  
  
hist(NAMiss2, col = "#8DD3C7", xlab = "Missingness (%)")



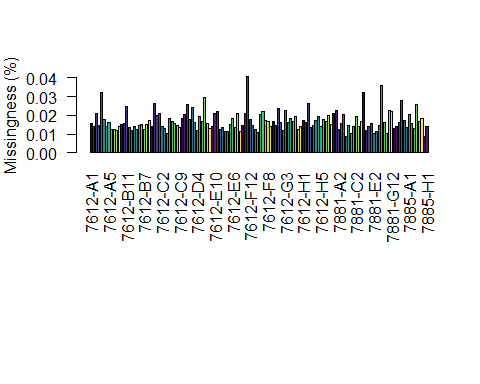
EUMiss <- apply(gt.vcEU, MARGIN = 2, function(x){ sum(is.na(x)) })  
EUMiss <- EUMiss/nrow(vcfEU)  
  
par(mar = c(12,4,4,2))  
barplot(EUMiss, las = 2, col = 1:12)  
title(ylab = "Missingness (%)")



par(mar = c(5,4,4,2))  
EUMiss2 <- apply(gt.vcEU, MARGIN = 1, function(x){ sum(is.na(x)) })  
EUMiss2 <- EUMiss2/ncol(vcfEU@gt[,-1])  
  
hist(NAMiss2, col = "#8DD3C7", xlab = "Missingness (%)")



#Removing an extreme outlier (7881-E12 has 92% missingness)  
gt.vcNA <- gt.vcNA[,-113]  
  
NAMiss <- apply(gt.vcNA, MARGIN = 2, function(x){ sum(is.na(x)) })  
NAMiss <- NAMiss/nrow(vcfNA)  
  
par(mar = c(12,4,4,2))  
barplot(NAMiss, las = 2, col = 1:12)  
title(ylab = "Missingness (%)")



## Filtering by DP (DP (1?x < x > 95%))

# Creating quantiles based on the lower 5% and the upper 95%  
sumsNA <- apply(dp.vc.bk.na, 2,   
 function (x) quantile(x, probs=c(0.05,   
 0.50,   
 0.95),   
 na.rm = T))  
  
# Add the minimum depth (14x)  
sumsNA[1,][sumsNA[1,] < 14] <- 14  
dp.all.na <- sweep(dp.vc.bk.na, MARGIN=2, FUN="-", sumsNA[1,])  
dp.vcNA[dp.all.na <= 0] <- NA  
dp.all.na <- sweep(dp.vc.bk.na, MARGIN=2, FUN="-", sumsNA[3,])  
dp.vcNA[dp.all.na > 0] <- NA  
  
# Changing GT  
vcfNA@gt[,-1][is.na(dp.vcNA)] <- NA  
  
# Creating quantiles based on the lower 5% and the upper 95%  
sumsEU <- apply(dp.vc.bk.eu, 2,   
 function (x) quantile(x, probs=c(0.05,   
 0.50,   
 0.95),   
 na.rm = T))  
  
# Add the minimum depth (12x)  
sumsEU[1,][sumsEU[1,] < 12] <- 12  
dp.all.eu <- sweep(dp.vc.bk.eu, MARGIN=2, FUN="-", sumsEU[1,])  
dp.vcEU[dp.all.eu <= 0] <- NA  
dp.all.eu <- sweep(dp.vc.bk.eu, MARGIN=2, FUN="-", sumsEU[3,])  
dp.vcEU[dp.all.eu > 0] <- NA  
  
# Changing GT  
vcfEU@gt[,-1][is.na(dp.vcEU)] <- NA

# Filtering by maximum MQ (MQ == 50)

mq.na <- extract.info(vcfNA, element = "MQ", as.numeric = T)  
  
# Creating mask  
mask.mq.na <- rep(T, nrow(vcfNA))  
  
# Filtering in the mask  
mask.mq.na[mq.na < 50] <- F  
  
# Regions removed by mask  
biallelic.vcfNA - sum(mask.mq.na)

## [1] 103597

mq.eu <- extract.info(vcfEU, element = "MQ", as.numeric = T)  
  
# Creating mask  
mask.mq.eu <- rep(T, nrow(vcfEU))  
  
# Filtering in the mask  
mask.mq.eu[mq.eu < 50] <- F  
  
# Regions removed by mask  
biallelic.vcfEU - sum(mask.mq.eu)

## [1] 100496

# Filtering on MAF (5% minor allele freq)

maf.tresh.na <- 7/(ncol(vcfNA@gt[,-1]))  
cat("MAF threshold:", maf.tresh.na, "\n")

## MAF threshold: 0.05185185

# Creating mask  
mask.maf.na <- rep(T, nrow(vcfNA))  
  
# Extracting GT and calculating MAF  
class(gt.vcNA) <- 'numeric'  
mask.maf.na <- apply(gt.vcNA, 1,   
 function (x) min(table(x)))/ncol(vcfNA@gt) >= maf.tresh.na  
  
# Regions removed by mask  
biallelic.vcfNA - sum(mask.maf.na)

## [1] 143779

maf.tresh.eu <- 8/(ncol(vcfEU@gt[,-1]))  
cat("MAF threshold:", maf.tresh.eu, "\n")

## MAF threshold: 0.05

# Creating mask  
mask.maf.eu <- rep(T, nrow(vcfEU))  
  
# Extracting GT and calculating MAF  
class(gt.vcEU) <- 'numeric'  
mask.maf.eu <- apply(gt.vcEU, 1,   
 function (x) min(table(x)))/ncol(vcfEU@gt) >= maf.tresh.eu  
  
# Regions removed by mask  
biallelic.vcfEU - sum(mask.maf.eu)

## [1] 53456

# Filtering by missing data:

# Creating mask  
mask.miss.na <- rep(T, nrow(vcfNA))  
  
# Extracting GT and calculating missingness  
gt.vcNA <- extract.gt(vcfNA, element = "GT", as.numeric = T)  
mask.miss.na <- apply(gt.vcNA, 1,   
 function (x) sum(is.na(x),  
 na.rm = TRUE)/ncol(gt.vcNA)) <= 0.10  
  
# Regions removed by mask  
biallelic.vcfNA - sum(mask.miss.na)

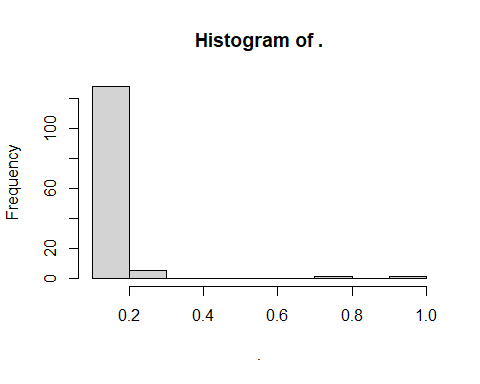
## [1] 132845

# Creating mask  
mask.miss.eu <- rep(T, nrow(vcfEU))  
  
# Extracting GT and calculating missingness  
gt.vcEU <- extract.gt(vcfEU, element = "GT", as.numeric = T)  
mask.miss.eu <- apply(gt.vcEU, 1,   
 function (x) sum(is.na(x),  
 na.rm = TRUE)/ncol(gt.vcEU)) <= 0.10  
  
# Regions removed by mask  
biallelic.vcfEU - sum(mask.miss.eu)

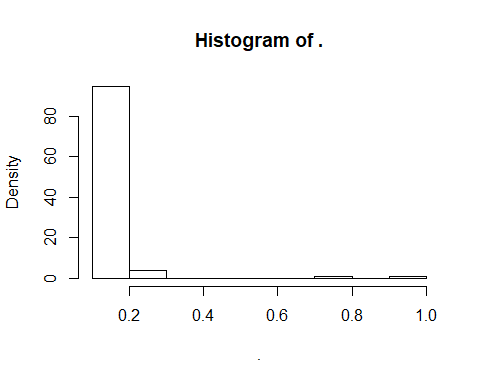
## [1] 98119

## Missing DP per sample (Missing more than ??% data)

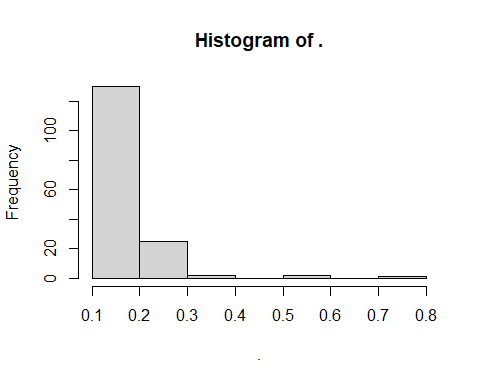
h.na <- apply(dp.vcNA, 2,   
 function (x) is.na(x) %>% sum/nrow(dp.vcNA)) %>% hist()



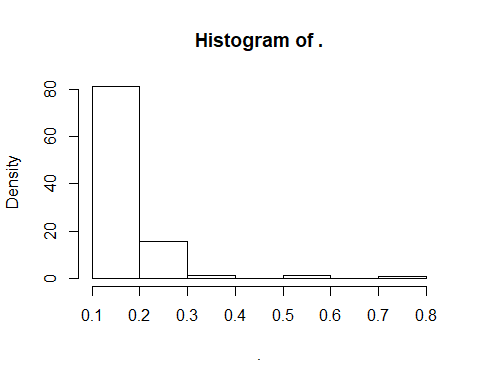
h.na$density = h.na$counts/sum(h.na$counts)\*100  
plot(h.na,freq=FALSE)



h.eu <- apply(dp.vcEU, 2,   
 function (x) is.na(x) %>% sum/nrow(dp.vcEU)) %>% hist()



h.eu$density = h.eu$counts/sum(h.eu$counts)\*100  
plot(h.eu,freq=FALSE)



missing.na <- colnames(dp.vcNA)[apply(dp.vcNA, 2,   
 function (x) is.na(x)  
 %>%  
 sum/nrow(dp.vcNA)) > 0.2]  
vcfNA@gt <- vcfNA@gt[,!colnames(vcfNA@gt) %in% missing.na]  
mask.miss.na <- rep(T, nrow(vcfNA))  
gt.vcNA <- extract.gt(vcfNA, element = "GT", as.numeric = T)  
mask.miss.na <- apply(gt.vcNA, 1,   
 function (x) sum(is.na(x))/ncol(gt.vcNA)) <= 0.00  
  
missing.eu <- colnames(dp.vcEU)[apply(dp.vcEU, 2,   
 function (x) is.na(x)  
 %>%  
 sum/nrow(dp.vcEU)) > 0.2]  
vcfEU@gt <- vcfEU@gt[,!colnames(vcfEU@gt) %in% missing.eu]  
mask.miss.eu <- rep(T, nrow(vcfEU))  
gt.vcEU <- extract.gt(vcfEU, element = "GT", as.numeric = T)  
mask.miss.eu <- apply(gt.vcEU, 1,   
 function (x) sum(is.na(x))/ncol(gt.vcEU)) <= 0.00

# Processing masks

mask.vcf.na <- cbind(mask.mq.na, mask.maf.na, mask.miss.na)  
filtered.vcf.na <- vcfNA[apply(mask.vcf.na, 1, sum) == 3,]  
filtered.vcf.na

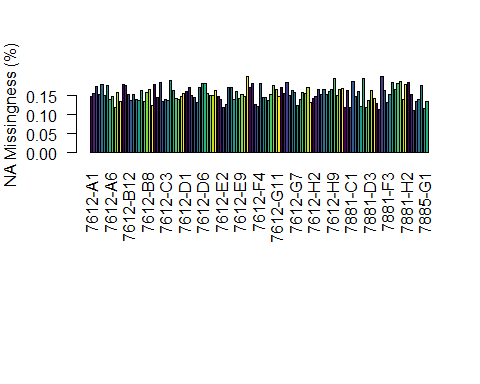
## \*\*\*\*\* Object of Class vcfR \*\*\*\*\*  
## 128 samples  
## 28 CHROMs  
## 50,693 variants  
## Object size: 479.3 Mb  
## 0 percent missing data  
## \*\*\*\*\* \*\*\*\*\* \*\*\*\*\*

mask.vcf.eu <- cbind(mask.mq.eu, mask.maf.eu, mask.miss.eu)  
filtered.vcf.eu <- vcfEU[apply(mask.vcf.eu, 1, sum) == 3,]  
filtered.vcf.eu

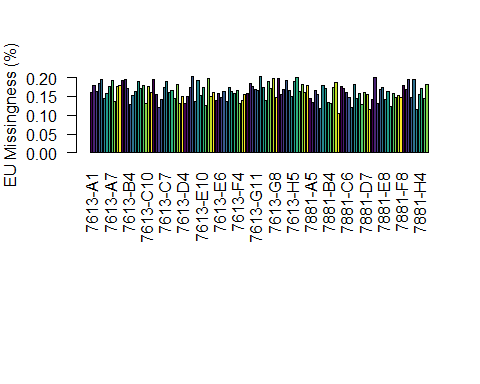
## \*\*\*\*\* Object of Class vcfR \*\*\*\*\*  
## 130 samples  
## 27 CHROMs  
## 63,736 variants  
## Object size: 522.2 Mb  
## 0 percent missing data  
## \*\*\*\*\* \*\*\*\*\* \*\*\*\*\*

# Revisualizing missingness

gt.vcNA <- extract.gt(vcfNA, element = "GT", as.numeric = T)  
NAMiss <- apply(gt.vcNA, MARGIN = 2, function(x){ sum(is.na(x)) })  
NAMiss <- NAMiss/nrow(vcfNA)  
  
par(mar = c(12,4,4,2))  
barplot(NAMiss, las = 2, col = 1:12)  
title(ylab = "NA Missingness (%)")



gt.vcEU <- extract.gt(vcfEU, element = "GT", as.numeric = T)  
EUMiss <- apply(gt.vcEU, MARGIN = 2, function(x){ sum(is.na(x)) })  
EUMiss <- EUMiss/nrow(vcfEU)  
  
par(mar = c(12,4,4,2))  
barplot(EUMiss, las = 2, col = 1:12)  
title(ylab = "EU Missingness (%)")

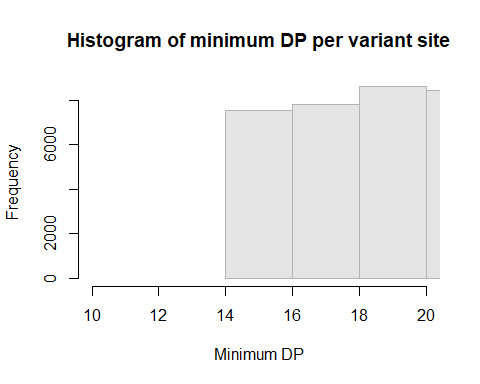


# Recalculating the number of polymorphic sites and removing all non-biallelic sites

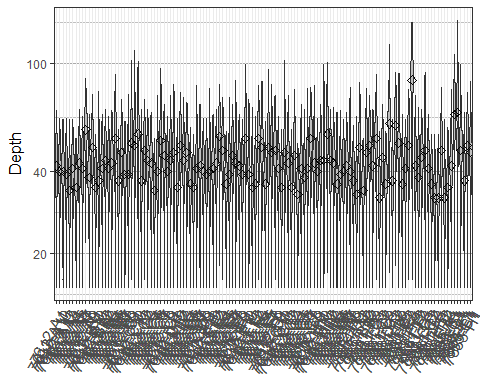
filtered.vcf.na <- filtered.vcf.na[is.polymorphic(filtered.vcf.na, na.omit = T)]  
poly.vcfNA <- nrow(filtered.vcf.na)  
vcfNA <- filtered.vcf.na[is.biallelic(filtered.vcf.na)]  
biallelic.vcfNA <- nrow(filtered.vcf.na)  
write.vcf(filtered.vcf.na, file = "FinalNA.vcf.gz", mask = F)  
  
filtered.vcf.eu <- filtered.vcf.eu[is.polymorphic(filtered.vcf.eu, na.omit = T)]  
poly.vcfEU <- nrow(filtered.vcf.eu)  
vcfEU <- filtered.vcf.eu[is.biallelic(filtered.vcf.eu)]  
biallelic.vcfEU <- nrow(filtered.vcf.eu)  
write.vcf(filtered.vcf.eu, file = "FinalEU.vcf.gz", mask = F)

# Did the new filtering by DP work?

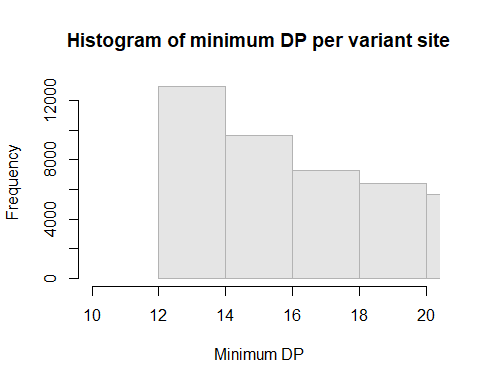
# DP from the new filtered data:  
dp.filtered.na <- extract.gt(filtered.vcf.na, element = "DP", as.numeric = T)  
apply(dp.filtered.na, 1, min) %>%   
 hist(xlab="Minimum DP",   
 main="Histogram of minimum DP per variant site",   
 border="grey70", col="grey90", breaks=10, xlim=c(10,20))



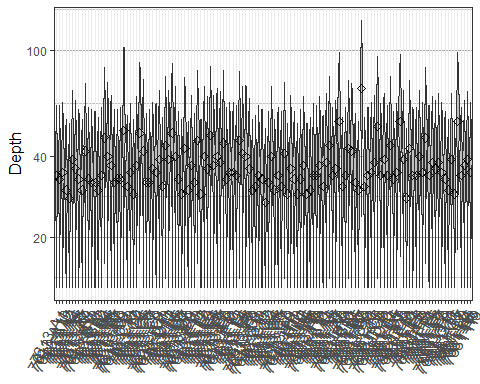
dpfNA <- melt(dp.filtered.na, varnames=c('Index', 'Sample'),   
 value.name = 'Depth', na.rm=TRUE)  
  
p <- ggplot(dpfNA, aes(x=Sample, y=Depth))+   
 geom\_violin(fill="#C0C0C0", adjust=1.0,  
 scale = "count", trim=TRUE)+   
 theme\_bw()+  
 theme(axis.title.x = element\_blank(),   
 axis.text.x = element\_text(angle = 60,   
 hjust = 1,   
 size=12),  
 axis.title.y = element\_text(size=12),  
 panel.grid.major.y=element\_line(color = "#A9A9A9",   
 size=0.6),  
 panel.grid.minor.y=element\_line(color = "#C0C0C0",   
 size=0.2))+   
 scale\_y\_continuous(trans=scales::log2\_trans(),   
 breaks=c(1, 10, 20, 40, 100))+  
 stat\_summary(fun=median, geom="point", shape=23, size=2)  
p



# DP from the new filtered data:  
dp.filtered.eu <- extract.gt(filtered.vcf.eu, element = "DP", as.numeric = T)  
apply(dp.filtered.eu, 1, min) %>%   
 hist(xlab="Minimum DP",   
 main="Histogram of minimum DP per variant site",   
 border="grey70", col="grey90", breaks=10, xlim=c(10,20))



dpfEU <- melt(dp.filtered.eu, varnames=c('Index', 'Sample'),   
 value.name = 'Depth', na.rm=TRUE)  
  
q <- ggplot(dpfEU, aes(x=Sample, y=Depth))+   
 geom\_violin(fill="#C0C0C0", adjust=1.0,  
 scale = "count", trim=TRUE)+   
 theme\_bw()+  
 theme(axis.title.x = element\_blank(),   
 axis.text.x = element\_text(angle = 60,   
 hjust = 1,   
 size=12),  
 axis.title.y = element\_text(size=12),  
 panel.grid.major.y=element\_line(color = "#A9A9A9",  
 size=0.6),  
 panel.grid.minor.y=element\_line(color = "#C0C0C0",  
 size=0.2))+   
 scale\_y\_continuous(trans=scales::log2\_trans(),   
 breaks=c(1, 10, 20, 40, 100))+  
 stat\_summary(fun=median, geom="point", shape=23, size=2)  
q



# Testing for heterozygosity

gt.vc.het.na <- extract.gt(vcfNA, element = "GT", as.numeric = F)  
het.na <- is.het(gt.vc.het.na)  
het.sums.na <- rowSums(het.na)  
#view(het.sums.na)  
  
gt.vc.het.eu <- extract.gt(vcfEU, element = "GT", as.numeric = F)  
het.eu <- is.het(gt.vc.het.eu)  
het.sums.eu <- rowSums(het.eu)  
view(het.sums.eu)