

MAGERI benchmark using reference standard DNA library

Mikhail Shugay

December 1, 2016

Load metadata

```
df.vmeta <- read.table("hd734_variant_metadata.txt", sep="\t", header=T) # variants observed in HD734 a
df.smeta <- read.table("sample_metadata.txt", sep="\t", header=T) # metadata for amplicon sequencing sa
```

VCF parsing function

```
library(stringr)
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
```

```
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
##      cov, smooth, var
```

```
library(ggplot2)
```

```
library(ggbeeswarm)
```

```
library(dplyr)
```

```
##
```

```
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
##      filter, lag
```

```
## The following objects are masked from 'package:base':
```

```
##
```

```
##      intersect, setdiff, setequal, union
```

```
read_vcf <- function(file_name) {
  .vcf <- read.table(file_name, header = F, sep = "\t", stringsAsFactors = F)
  colnames(.vcf) <- c("chromosome", "position", "skip1", "from", "to", "qual", "skip2", "info", "skip3")
  .vcf$skip1 <- NULL
  .vcf$skip2 <- NULL
  .vcf$skip3 <- NULL
  .vcf$skip4 <- NULL

  .vcf$qual <- as.integer(.vcf$qual)
  .vcf <- subset(.vcf, nchar(from) == 1 &
                 nchar(to) == 1 & !is.na(qual)) # no indels

  .infosplit <- str_split_fixed(.vcf$info, regex("[=;]"), 14)[,c(2, 4, 12, 14)]

  .vcf$coverage <- as.numeric(.infosplit[,1])
}
```

```

.vcf$frequency <- as.numeric(.infosplit[,2])
.vcf$bb.a <- as.numeric(.infosplit[,3])
.vcf$bb.b <- as.numeric(.infosplit[,4])
.vcf$info <- NULL

.vcf$count <- as.integer(.vcf$coverage * .vcf$frequency)
.vcf$qual <- as.integer(.vcf$qual)

subset(.vcf, nchar(from) == 1 & nchar(to) == 1 & !is.na(qual))
}

head(read_vcf("p126.h4_2_ballast_m1.vcf"))

```

```
## Warning in read_vcf("p126.h4_2_ballast_m1.vcf"): NAs introduced by coercion
```

```
## chromosome position from to qual coverage frequency bb.a
## 1 chr2 212295704 C A 8 2265 0.0004415011 0.9957058
## 2 chr2 212295705 C A 8 2265 0.0004415011 0.9957058
## 3 chr2 212295713 A G 7 2265 0.0004415011 1.0295689
## 4 chr2 212295718 G A 6 2265 0.0004415011 2.1314828
## 5 chr2 212295725 C A 8 2264 0.0004416961 0.9957058
## 6 chr2 212295732 C T 24 2264 0.0013250883 2.1314828
## bb.b count
## 1 21862.23 0
## 2 21862.23 0
## 3 16394.12 0
## 4 24344.82 0
## 5 21862.23 1
## 6 24344.82 2
```

Read samples with HD734 standard DNA and control human DNA, append metadata

```
library(TailRank) # For betabinom.
```

```
## Loading required package: oompaBase
```

```

# installing:
# source("https://bioconductor.org/biocLite.R")
# biocLite("Biobase")
# install.packages("TailRank", repos="http://R-Forge.R-project.org")

```

```
df <- data.frame()
```

```

read_vcf_with_metadata <- function(file_name, primer_set, replica, ratio, type) {
  .vcf <- read_vcf(file_name)
  .vcf <- merge(.vcf, df.vmeta, all.x = type != "standard", all.y = F)
  .vcf$known.frequency <- .vcf$known.frequency * ratio
  .vcf$known.frequency[is.na(.vcf$known.frequency)] <- 0
  .vcf$primer_set <- primer_set
  .vcf$replica <- primer_set
  .vcf$type <- type

  .vcf <- subset(.vcf, frequency < 0.4 & count > 0) # remove alleles in control

  .vcf
}

```

```

}

for (i in 1:nrow(df.smeta)) {
  df <- with(df.smeta, rbind(df,
                             read_vcf_with_metadata(paste(prefix[i], "vcf", sep="."),
                                                       primer_set[i],
                                                       replica[i],
                                                       ratio[i],
                                                       type[i])))
}

```

```

## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion
## Warning in read_vcf(file_name): NAs introduced by coercion

```

```

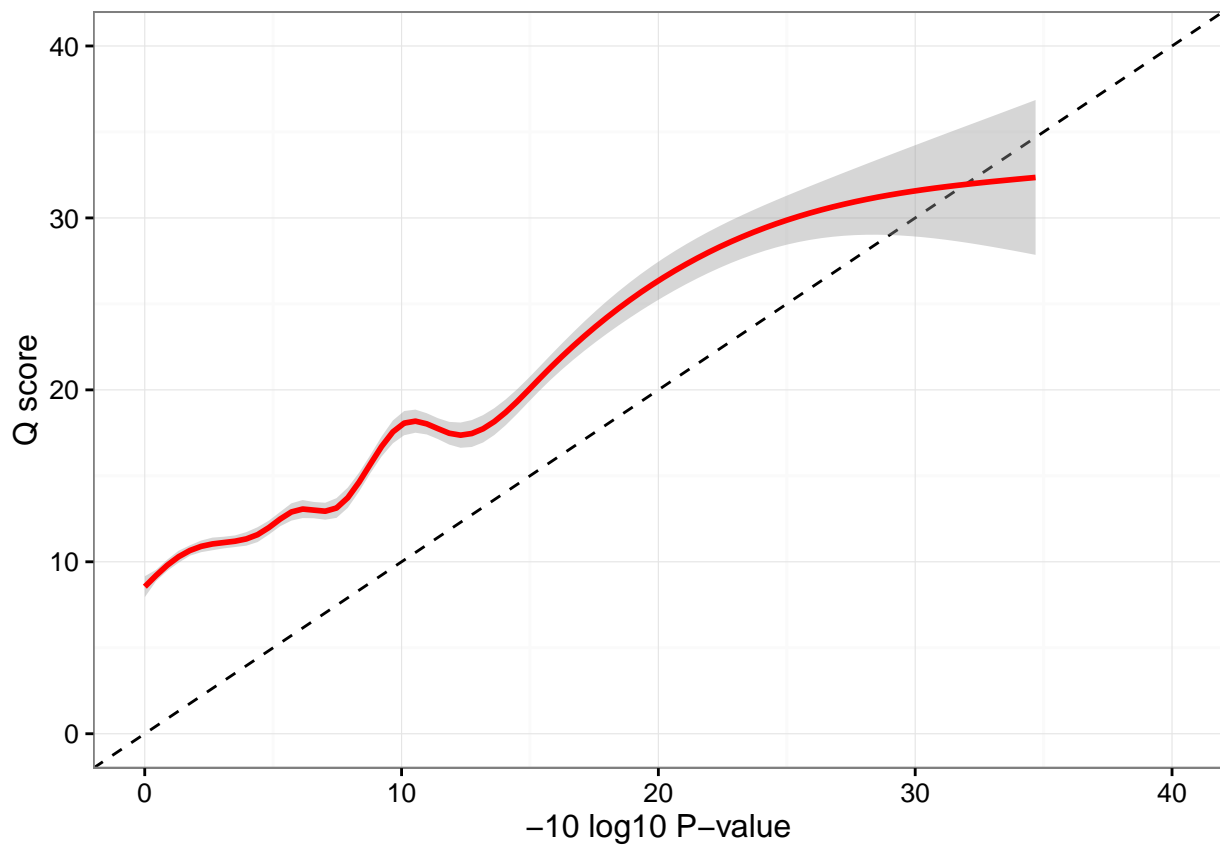
# Compute scores

```

```

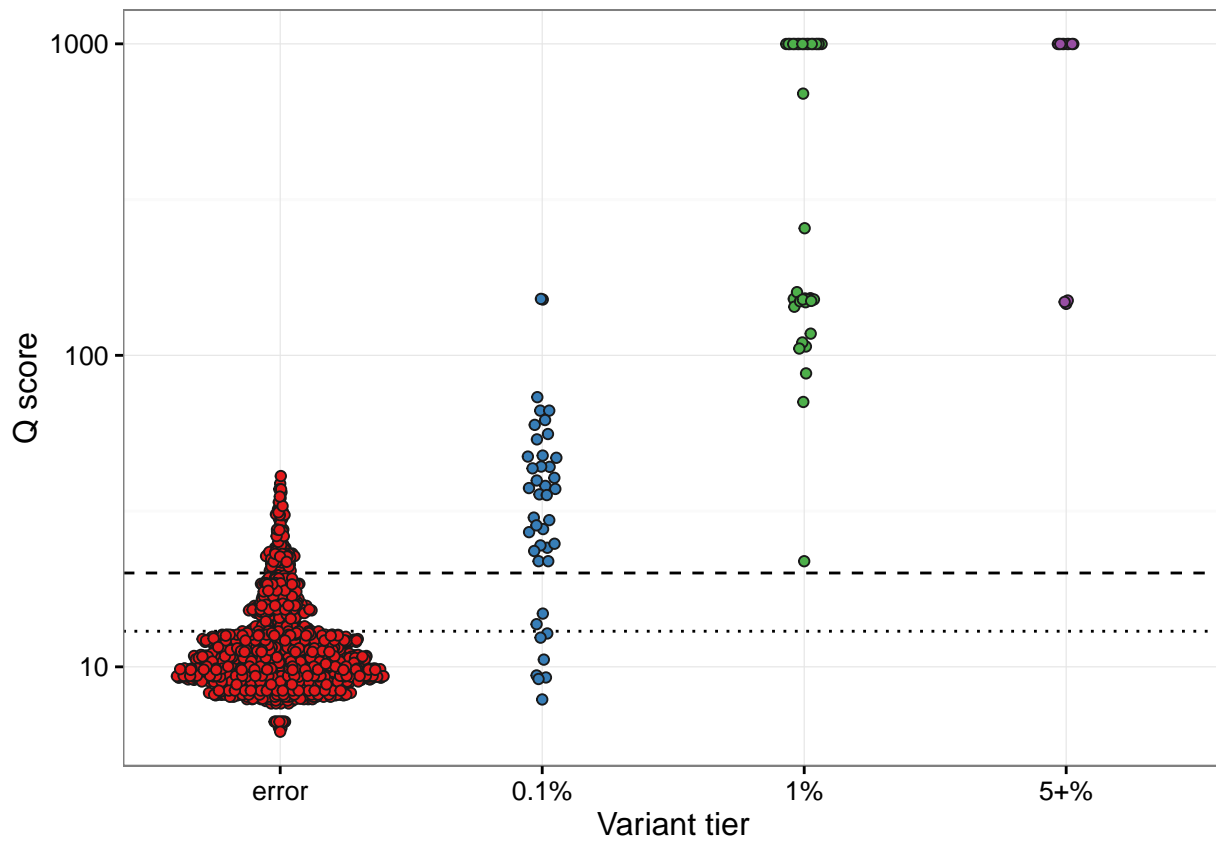
df$qual <- with(df,
  mapply(function(x,y,a,b) 1 - pbb(x,y,a,b) + 0.5 * dbb(x,y,a,b),
          count, coverage, bb.a, bb.b)

```

Variant quality scores

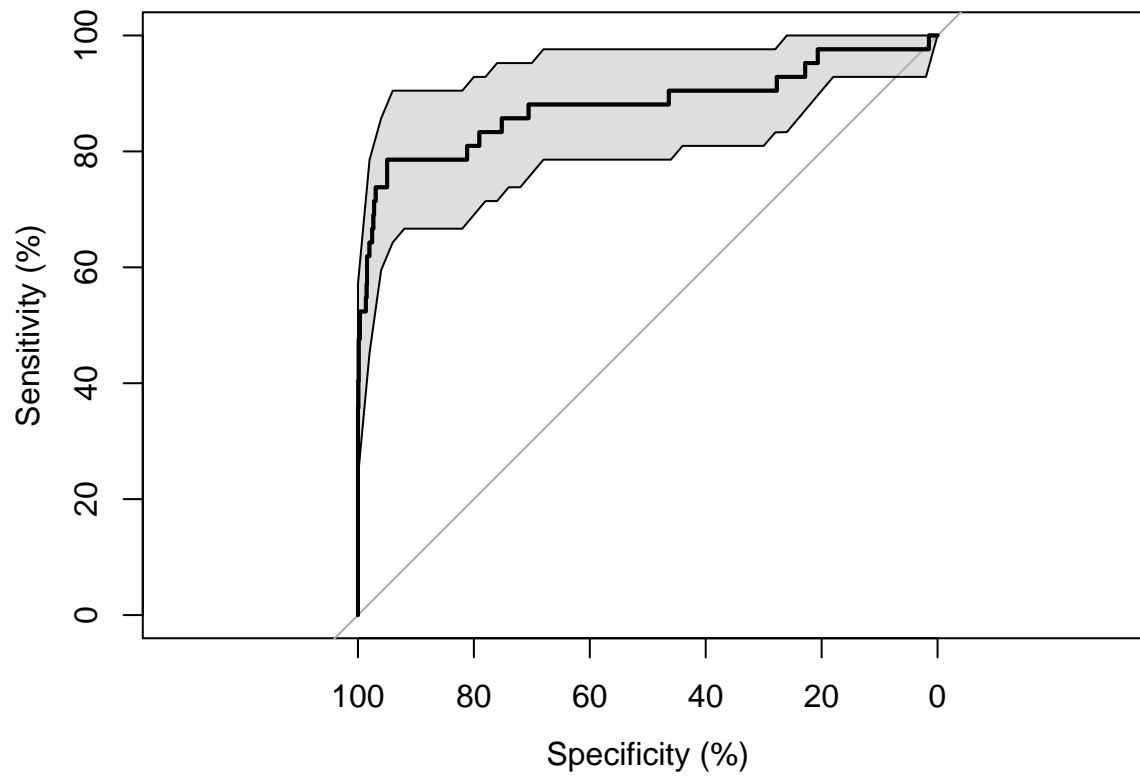
```
ggplot(df,aes(tier, qual, fill=tier)) +
  geom_hline(yintercept = 13.0103, linetype ="dotted") +
  geom_hline(yintercept = 20.0103, linetype ="dashed") +
  geom_quasirandom(varwidth = T, shape=21, color="grey10") +
  scale_fill_brewer("", palette = "Set1", guide=F) +
  xlab("Variant tier") + scale_y_log10("Q score") +
  theme_bw()
```



ROC curve for rare variants

```
df.1 <- subset(df, tier %in% c("error", "0.1%"))
df.1$type <- ifelse(df.1$tier == "error", 0, 1)
rocobj1 <- plot.roc(df.1$type, df.1$qual, percent=T, col="black", ci=T)

sens.ci <- ci.se(rocobj1, specificities=seq(0, 100, 2))
plot(sens.ci, type="shape", col = alpha("grey", 0.5))
```



```
rocobj1
```

```
##
## Call:
## plot.roc.default(x = df.1$type, predictor = df.1$qual, percent = T,      col = "black", ci = T)
##
## Data: df.1$qual in 1471 controls (df.1$type 0) < 42 cases (df.1$type 1).
## Area under the curve: 87.97%
## 95% CI: 80.3%-95.65% (DeLong)
```