data_simulation_analysis

Daniel

June 8, 2020

```
#libraries
library(tidyr)
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------- t
```

```
## v ggplot2 3.2.1     v purrr   0.3.2
## v tibble  2.1.3     v dplyr   0.8.3
## v readr   1.3.1     v stringr 1.4.0
## v ggplot2 3.2.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------------------------------------- tidyvers
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(reshape2)
```

```
##
## Attaching package: 'reshape2'
```

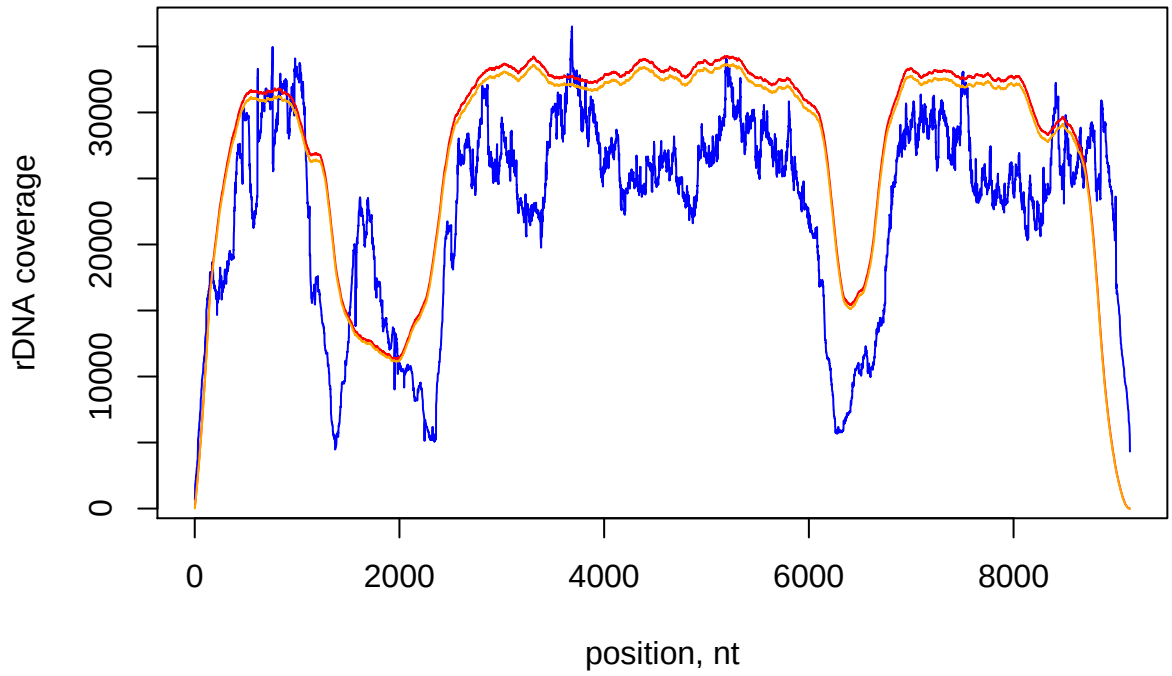
```
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(wesanderson) #grand budapest color pallettes
```

first, plot coverage

```
#these are coverage files generated using bedtool genomecov -d -ibam
#real dataset (S288C from Liti's paper)
#experiment - coverage~230X
#simulation7 - 230X genome cov, 30000X rDNA coverage
experiment <- read.table("~/Desktop/rDNA_project/benchmarking/data_simulation/SRR4074255_coverage.txt",
                    "\t",
                    header = F)
experiment <- as.data.frame(experiment)
colnames(experiment) <- c("genome", "position", "coverage")
#simulated dataset
simulation7 <- read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_7/sim7_gold_
                    "\t",
                    header = F)
simulation7 <- as.data.frame(simulation7)
colnames(simulation7) <- c("genome", "position", "coverage")
#pipeline run on simulation7
pipeline7 <- read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_7/sim7_cov.tx
pipeline7 <- as.data.frame(pipeline7)
colnames(pipeline7) <- c("genome", "position", "coverage")
```

```r
#make the graph in ggplot
plot(experiment$position, experiment$coverage,
     type="l",
     col="blue",
     xlab="position, nt",
     ylab="rDNA coverage"
     )
lines(simulation7$position, simulation7$coverage,
      col="red")
lines(pipeline7$position, pipeline7$coverage,
      col="orange")
```



##analyze .vcf files from simulations #vcf produced by pipeline using simulated data; .gold vcf was produces by the simulator (true priors) #all simulation were done with –rng that produced identical results; the difference is in the coverage #I spiked datasets with resultant AF=0.005

```r
#golden vcf with true priors; same for all simulation bc the same --rng seed was used;
gold.vcf <- read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_7/simulation7_g
                       "\t",
                       header = F,
                       comment.char = "#")
gold.vcf <-as.data.frame(gold.vcf)
#assign colnamnes
vcf_col_names <- c("CHROM", "POS", "ID", "REF", "ALT", "QUAL", "FILTER", "INFO")
colnames(gold.vcf) <- vcf_col_names
#simulation8 - 20X genome coverage, 3000X rDNA coverage; pipeline results
#NB: i wrote a function downstream to process the vcf files automatically
sim8.vcf <- as.data.frame(read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_8
                          "\t",
                          header = F,
                          comment.char = "#"
                          ))
colnames(sim8.vcf) <- vcf_col_names
```

```
#split INFO column into several; HRUN is only for INDEL
#NB: if some future column that will not shared between different rows, try to explore the 'fill' argum
sim8.vcf <- sim8.vcf %>% separate("INFO",
                    c("DP","AF","SB","DP4","INDEL","HRUN"),
                    sep=";"
                    )
```

```
## Warning: Expected 6 pieces. Missing pieces filled with `NA` in 7 rows [3,
## 4, 5, 6, 7, 8, 9].
```

```
#make REF as char
sim8.vcf$REF<-as.character(sim8.vcf$REF)
#remove characters from the new columnus and make them appropriate class
#[A-Z] - substring starts with a letter followed by everything else * and then the = sign. replace with
sim8.vcf$DP<-as.integer(gsub("[A-Z]*=","",sim8.vcf$DP))
sim8.vcf$AF<-as.numeric(gsub("[A-Z]*=","",sim8.vcf$AF)) #set it here for numeric bc these are NOT integ
sim8.vcf$SB<-as.integer(gsub("[A-Z]*=","",sim8.vcf$SB))
sim8.vcf$DP4<-as.factor(gsub("[A-Z]*4=","",sim8.vcf$DP4)) #note the '4' in the first argument bc it sta
sim8.vcf$HRUN<-as.integer(gsub("[A-Z]*=","",sim8.vcf$HRUN))
#sim8.vcf

#write a function for processing vcf files
vcf_processing <- function(name) {
  colnames(name) <- c("CHROM", "POS", "ID", "REF", "ALT", "QUAL", "FILTER", "INFO")
  name <- name %>% separate("INFO",
                    c("DP","AF","SB","DP4","INDEL","HRUN"),
                    sep=";"
                    )
  #make REF as char
  name$REF<-as.character(name$REF)
  #remove characters from the new columnus and make them appropriate class
  #[A-Z] - substring starts with a letter followed by everything else * and then the = sign. replace wi
  name$DP<-as.integer(gsub("[A-Z]*=","",name$DP))
  name$AF<-as.numeric(gsub("[A-Z]*=","",name$AF)) #set it here for numeric bc these are NOT integers
  name$SB<-as.integer(gsub("[A-Z]*=","",name$SB))
  name$DP4<-as.factor(gsub("[A-Z]*4=","",name$DP4)) #note the '4' in the first argument bc it starts as
  name$HRUN<-as.integer(gsub("[A-Z]*=","",name$HRUN))
  return(name)
}

#simulation9 - 50X genome coverage, 7500X rDNA coverage
sim9.vcf <- as.data.frame(read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_
                                     ,
                                     "\t",
                                     header = F,
                                     comment.char = "#"
                                     )
)
sim9.vcf <- vcf_processing(sim9.vcf)
```

```
## Warning: Expected 6 pieces. Missing pieces filled with `NA` in 24 rows [2,
## 3, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, ...].
```

```
#simulation10 - 100X genome coverage, 15000X rDNA coverage
sim10.vcf <- as.data.frame(read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_
```

```
                                    "\t",
                                    header = F,
                                    comment.char = "#"
                                    )
)
sim10.vcf <- vcf_processing(sim10.vcf)
```

```
## Warning: Expected 6 pieces. Missing pieces filled with `NA` in 44 rows [1,
## 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 18, 19, 20, 21, 22, ...].
```

```
#simulation7 (nb: yes, it goes a bit out of order here, but it is next in coverage); 200X genome covera
sim7.vcf <- as.data.frame(read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_
                                    "\t",
                                    header = F,
                                    comment.char = "#"
                                    )
)
sim7.vcf <- vcf_processing(sim7.vcf)
```

```
## Warning: Expected 6 pieces. Missing pieces filled with `NA` in 54 rows [1,
## 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, ...].
```

```
#simulation11 - 400X genome coverage, 60000X rDNA coverage
sim11.vcf <- as.data.frame(read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation
                                    "\t",
                                    header = F,
                                    comment.char = "#"
                                    )
)
sim11.vcf <- vcf_processing(sim11.vcf)
```

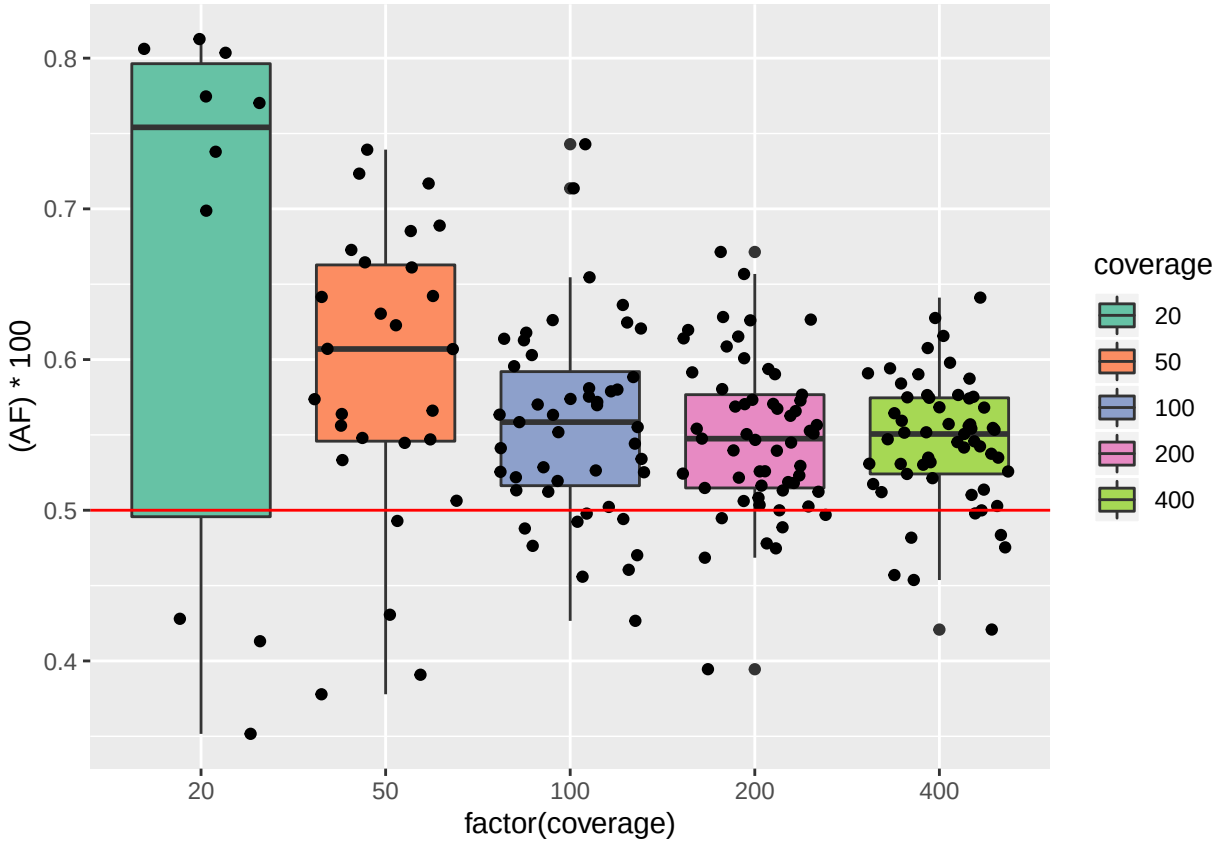
```
## Warning: Expected 6 pieces. Missing pieces filled with `NA` in 54 rows [1,
## 2, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, ...].
```

```
#sim11.vcf
```

creating boxplots

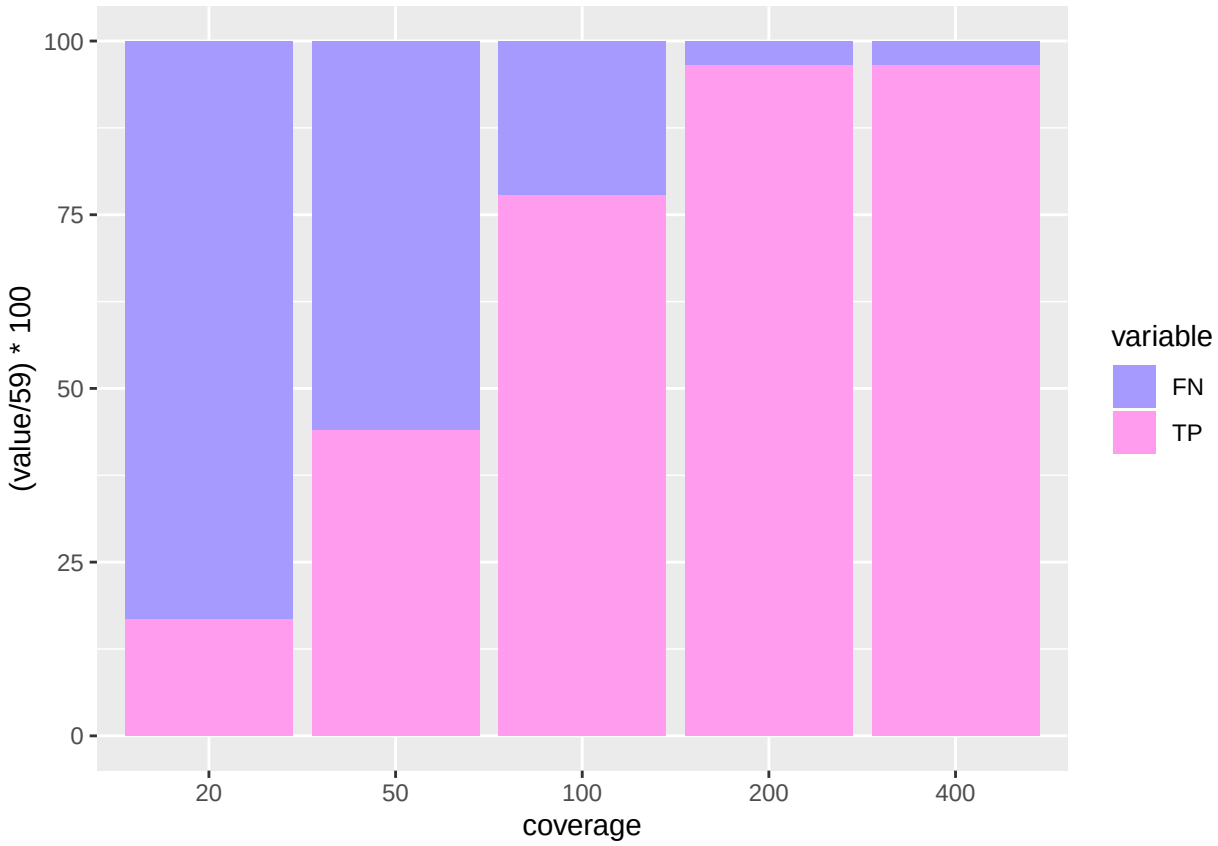
```
#plotted are all called by lofreq (there were no FP though only FN; hence, here are plotted only TP(tru
#"melted data"
#first, create a new dataframe
names_for_col <- c("coverage", "AF")
coverage_20X <-  data.frame(as.factor(rep(c(20))), sim8.vcf$AF) #first column must be factor
coverage_50X <- data.frame(as.factor(rep(c(50))), sim9.vcf$AF)
coverage_100X <- data.frame(as.factor(rep(c(100))), sim10.vcf$AF)
coverage_200X <- data.frame(as.factor(rep(c(200))), sim7.vcf$AF)
coverage_400X <- data.frame(as.factor(rep(c(400))), sim11.vcf$AF)
colnames(coverage_20X) = names_for_col #all tables should have the same names to be rbind
colnames(coverage_50X) = names_for_col
colnames(coverage_100X) = names_for_col
colnames(coverage_200X) = names_for_col
colnames(coverage_400X) = names_for_col
coverage_all <- rbind(coverage_20X, coverage_50X, coverage_100X, coverage_200X, coverage_400X)
```

```
#coverage_all

#plot boxplots; specify factor to plot multiple boxplots
#NB the x argument must be a factor; otherwise when coloring it will treat is as a continuous rather th
ggplot(data=coverage_all, aes(x = factor(coverage), y=(AF)*100))+
  geom_boxplot(aes(fill=coverage))+
  geom_jitter()+
  scale_fill_brewer(palette = "Set2")+
  geom_hline(yintercept = 0.5, col="red")
```



#plot FN (false neg) and TP (true pos) ##NB: no FPs were detected by the pipeline so I am not plotting it here

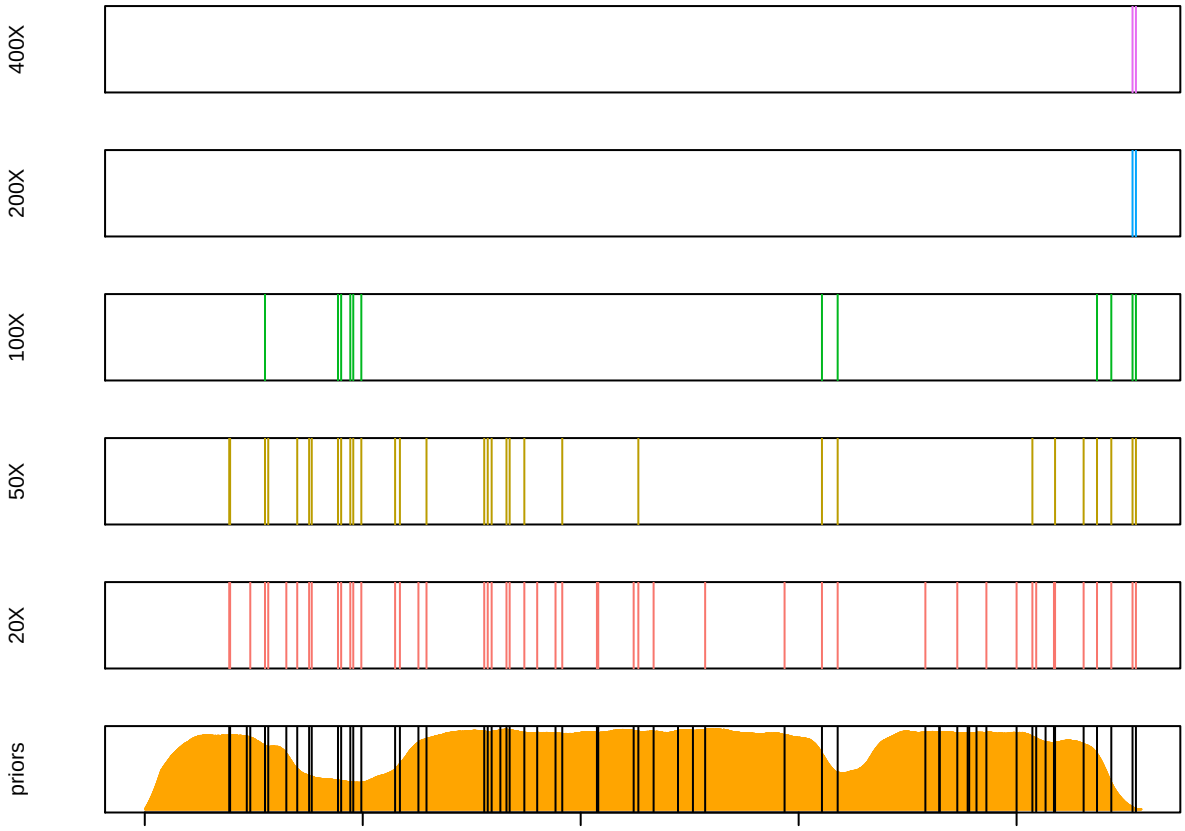
```
#NB: TP = same position AND variant
#make a table
TP_FN <- data.frame(coverage = as.factor(c(20, 50, 100, 200, 400)),
                    FN = c(49, 33, 13, 2, 2),
                    TP = c(10, 26, 46, 57, 57))
TP_FN<-melt(TP_FN, id=c("coverage"))
#TP_FN; plot values in percentage
ggplot(data=TP_FN, aes(x=coverage, y=(value/59)*100))+
  geom_col(aes(fill=variable))+
  scale_fill_manual(values = c("TP" = "#FF9CEE",
                               "FN" = "#A79AFF")
  )
```



#position of FN

```
#load FN files
#note: no need to convert additionally into .df because it is already imported as a dataframe using thi
FN_20X <- read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_8/simulation8_FN
                     "\t",
                     header = F)
colnames(FN_20X) <- vcf_col_names

FN_50X <- read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_9/simulation9_FN
                    "\t",
                     header = F)
colnames(FN_50X) <- vcf_col_names

FN_100X <- read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_10/simulation10_
                     "\t",
                     header = F)
colnames(FN_100X) <- vcf_col_names

FN_200X <- read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_7/vcfcomparison_
                     "\t",
                     header = F)
colnames(FN_200X) <- vcf_col_names

FN_400X <- read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_11/simulation11_
                     "\t",
                     header = F)
colnames(FN_400X) <- vcf_col_names
```

```
#true priors
priors <- read.table("~/Desktop/rDNA_project/benchmarking/data_simulation_2/simulation_7/simulation7_gol
                     "\t",
                     header = F)
colnames(priors) <- vcf_col_names


#plot
#this f(x) doesn't work - will work on it later
#plot_FN <- function(data, position, col_scheme) {
#  plot(c(1,9137), c(1,2), type = "n",
#     xlab = "",
#     ylab ="",
#     xaxt="n",
#     yaxt="n")
#  a<-recordPlot(abline(v = data$position, col=col_scheme))
#return(a)
#}


#FN distribution
par(mfrow=c(6,1), mai=c(0.1,0.7,0.2,0.2))
#initiate empty plot first

#400X
plot(c(1,9137), c(1,2), type = "n", xlab = "", ylab ="400X", xaxt="n", yaxt="n")
abline(v = FN_400X$POS, col="#E76BF3")

#200X
plot(c(1,9137), c(1,2), type = "n", xlab = "", ylab ="200X", xaxt="n", yaxt="n")
abline(v = FN_200X$POS, col="#00A5FF")

#100X
plot(c(1,9137), c(1,2), type = "n", xlab = "", ylab ="100X", xaxt="n", yaxt="n")
abline(v = FN_100X$POS, col="#00B81F")

#50X
plot(c(1,9137), c(1,2), type = "n", xlab = "", ylab ="50X", xaxt="n", yaxt="n")
abline(v = FN_50X$POS, col="#BB9D00")

#20X
plot(c(1,9137), c(1,2), type = "n", xlab = "", ylab ="20X", xaxt="n", yaxt="n")
abline(v = FN_20X$POS, col="#F8766D")

#true priors
#plot(c(1,9137), c(1,2), type = "n", xlab = "", ylab ="priors", xaxt="n", yaxt="n")

#coverage (simulated pipeline) with true priors
plot(pipeline7$position, pipeline7$coverage, type = "h", col="orange", xlab="position", ylab="priors",
abline(v = priors$POS, col="black")
```


400X
200X
100X
50X
20X
priors