TM032

Rachael Valdez

November 10, 2017

if (Sys.getenv("JAVA\_HOME")!="")  
 Sys.setenv(JAVA\_HOME="")

### Test Method

This test method analysis will be used to determine if fast protein liquid chromatography(FPLC) and tangential flow filtration(TFF) meet the set specifications for each run to be built into production.Several samples throughout the downstream processing will be tested to make sure the protein is still functional and loss during processing is minimized. The protein is tested for activity at 8% against 4 positive and 1 negative bacterial strains to make sure the protein is targeting set organisms. Using a Spectramax-L, luminescence is measured over 12.5 seconds. The data generated is from a Spectramax-L which outputs a .txt file which will be imported.

### Data Analysis

1: Import data and libraries

2: Calculated averages

3: Calculated standard deviation

4: Calculated CV

5: Generate strain analysis chart with pass or fail

6: Generate dot plot

7: Generate bar graph

8: Data export

Note: Before running script go to section 8 and input a unique name identifier so files do not save over each other.

### 1: Data Import

Data file, .txt, will be imported. The columns will be named and the peak of each well will be calculated.

##Set your directory to folder with your data  
  
spec <- file.choose()  
data = read.table(spec, header=FALSE, sep='\t', skip = 3,nrows = 50)  
colNames = c('Time', 'TemperatureC')  
colLetters = LETTERS[1:8]  
for(i in 1:length(colLetters)) {  
 for(j in 1:12) {  
 temp = paste(colLetters[i], j, sep = "")  
 colNames = c(colNames, temp)  
 }  
}  
colnames(data) = colNames  
  
Max = apply(data,2,max)  
#print(Max)  
df = rbind(data, Max) #add max of each column to last row  
myData = round(df[-c(1:50), -c(1,2) ]) #gathered headers and max values  
MaxValues = t(myData) #Transpose data frame  
colnames(MaxValues)[1] <- "Max\_RLU" #name column  
PeakValues = data.frame(na.omit(MaxValues)) #remove values with NA  
  
  
###Libraries  
library(xlsx)

## Loading required package: rJava

## Loading required package: xlsxjars

library(rJava)  
library(xlsxjars)  
library(ggplot2)  
library(reshape2)

### Strain Analysis

Each bacterial strain tested will have the average, standard deviation and CV calculated. Based off of these calulations a chart will be output with the set specifications and a pass or fail output.

### 2: Peak Value

###AVC Means  
Strain1 <- rowMeans(myData[,1:2])  
Strain2 <- rowMeans(myData[,3:4])  
Strain3 <- rowMeans(myData[,5:6])  
Strain4 <- rowMeans(myData[,7:8])  
Strain5 <- (myData[,9])  
Strain6 <- (myData[,10])  
AVCMean = rbind(Strain1, Strain2, Strain3, Strain4, Strain5, Strain6)  
colnames(AVCMean)[1] <- "AVC" #name column  
AVCMean = t(AVCMean)  
  
###Retentate Means  
Strain1 <- rowMeans(myData[,13:14])  
Strain2 <- rowMeans(myData[,15:16])  
Strain3 <- rowMeans(myData[,17:18])  
Strain4 <- rowMeans(myData[,19:20])  
Strain5 <- (myData[,21])  
Strain6 <- (myData[,22])  
RetMean = rbind(Strain1, Strain2, Strain3, Strain4, Strain5, Strain6)  
colnames(RetMean)[1] <- "Retentate" #name column  
RetMean = t(RetMean)  
  
###Eluate Means  
Strain1 <- rowMeans(myData[,25:26])  
Strain2 <- rowMeans(myData[,27:28])  
Strain3 <- rowMeans(myData[,29:30])  
Strain4 <- rowMeans(myData[,31:32])  
Strain5 <- (myData[,33])  
Strain6 <- (myData[,34])  
EluMean = rbind(Strain1, Strain2, Strain3, Strain4, Strain5, Strain6)  
colnames(EluMean)[1] <- "Eluate" #name column  
EluMean = t(EluMean)  
  
###Lys Means  
Strain1 <- rowMeans(myData[,37:38])  
Strain2 <- rowMeans(myData[,39:40])  
Strain3 <- rowMeans(myData[,41:42])  
Strain4 <- rowMeans(myData[,43:44])  
Strain5 <- (myData[,45])  
Strain6 <- (myData[,46])  
LysMean = rbind(Strain1, Strain2, Strain3, Strain4, Strain5, Strain6)  
colnames(LysMean)[1] <- "Lysate" #name column  
LysMean = t(LysMean)  
  
###LM Means  
Strain1 <- rowMeans(myData[,49:50])  
Strain2 <- rowMeans(myData[,51:52])  
Strain3 <- rowMeans(myData[,53:54])  
Strain4 <- rowMeans(myData[,55:56])  
Strain5 <- (myData[,57])  
Strain6 <- (myData[,58])  
LMMean = rbind(Strain1, Strain2, Strain3, Strain4, Strain5, Strain6)  
colnames(LMMean)[1] <- "LM" #name column  
LMMean = t(LMMean)  
  
###FT Means  
Strain1 <- rowMeans(myData[,61:62])  
Strain2 <- rowMeans(myData[,63:64])  
Strain3 <- rowMeans(myData[,65:66])  
Strain4 <- rowMeans(myData[,67:68])  
Strain5 <- (myData[,69])  
Strain6 <- (myData[,70])  
FTMean = rbind(Strain1, Strain2, Strain3, Strain4, Strain5, Strain6)  
colnames(FTMean)[1] <- "FT" #name column  
FTMean = t(FTMean)  
  
###Wash1 Means  
Strain1 <- rowMeans(myData[,73:74])  
Strain2 <- rowMeans(myData[,75:76])  
Strain3 <- rowMeans(myData[,77:78])  
Strain4 <- rowMeans(myData[,79:80])  
Strain5 <- (myData[,81])  
Strain6 <- (myData[,82])  
W1Mean = rbind(Strain1, Strain2, Strain3, Strain4, Strain5, Strain6)  
colnames(W1Mean)[1] <- "Wash1" #name column  
W1Mean = t(W1Mean)  
  
###Wash2 Means  
Strain1 <- rowMeans(myData[,85:86])  
Strain2 <- rowMeans(myData[,87:88])  
Strain3 <- rowMeans(myData[,89:90])  
Strain4 <- rowMeans(myData[,91:92])  
Strain5 <- (myData[,93])  
Strain6 <- (myData[,94])  
W2Mean = rbind(Strain1, Strain2, Strain3, Strain4, Strain5, Strain6)  
colnames(W2Mean)[1] <- "Wash2" #name column  
W2Mean = t(W2Mean)  
  
Mean = rbind(AVCMean, RetMean, EluMean, LysMean, LMMean, FTMean, W1Mean, W2Mean)  
Mean = as.data.frame(round(Mean))  
print(Mean)

## Strain1 Strain2 Strain3 Strain4 Strain5 Strain6  
## AVC 37694 9836 23280 2822 241 250  
## Retentate 9132 9046 7554 966 318 326  
## Eluate 26169 8564 20938 2967 267 275  
## Lysate 46908 7511 4912 1692 267 224  
## LM 1007 1587 432 227 191 109  
## FT 26812 9301 4447 1138 191 173  
## Wash1 2412 2915 1578 272 227 109  
## Wash2 9900 7185 4076 522 191 191

### 3: Standard Deviation

###AVC SD  
Strain1 <- sd(myData[,1:2])  
Strain2 <- sd(myData[,3:4])  
Strain3 <- sd(myData[,5:6])  
Strain4 <- sd(myData[,7:8])  
AVCSD = rbind(Strain1, Strain2, Strain3, Strain4)  
colnames(AVCSD)[1] <- "AVC" #name column  
AVCSD = t(AVCSD)  
  
###Retentate SD  
Strain1 <- sd(myData[,13:14])  
Strain2 <- sd(myData[,15:16])  
Strain3 <- sd(myData[,17:18])  
Strain4 <- sd(myData[,19:20])  
RetSD = rbind(Strain1, Strain2, Strain3, Strain4)  
colnames(RetSD)[1] <- "Retentate" #name column  
RetSD = t(RetSD)  
  
###Eluate SD  
Strain1 <- sd(myData[,25:26])  
Strain2 <- sd(myData[,27:28])  
Strain3 <- sd(myData[,29:30])  
Strain4 <- sd(myData[,31:32])  
EluSD = rbind(Strain1, Strain2, Strain3, Strain4)  
colnames(EluSD)[1] <- "Eluate" #name column  
EluSD = t(EluSD)  
  
###Lys SD  
Strain1 <- sd(myData[,37:38])  
Strain2 <- sd(myData[,39:40])  
Strain3 <- sd(myData[,41:42])  
Strain4 <- sd(myData[,43:44])  
LysSD = rbind(Strain1, Strain2, Strain3, Strain4)  
colnames(LysSD)[1] <- "Lysate" #name column  
LysSD = t(LysSD)  
  
###LM SD  
Strain1 <- sd(myData[,49:50])  
Strain2 <- sd(myData[,51:52])  
Strain3 <- sd(myData[,53:54])  
Strain4 <- sd(myData[,55:56])  
LMSD = rbind(Strain1, Strain2, Strain3, Strain4)  
colnames(LMSD)[1] <- "LM" #name column  
LMSD = t(LMSD)  
  
###FT SD  
Strain1 <- sd(myData[,61:62])  
Strain2 <- sd(myData[,63:64])  
Strain3 <- sd(myData[,65:66])  
Strain4 <- sd(myData[,67:68])  
FTSD = rbind(Strain1, Strain2, Strain3, Strain4)  
colnames(FTSD)[1] <- "FT" #name column  
FTSD = t(FTSD)  
  
###Wash1 SD  
Strain1 <- sd(myData[,73:74])  
Strain2 <- sd(myData[,75:76])  
Strain3 <- sd(myData[,77:78])  
Strain4 <- sd(myData[,79:80])  
W1SD = rbind(Strain1, Strain2, Strain3, Strain4)  
colnames(W1SD)[1] <- "Wash1" #name column  
W1SD = t(W1SD)  
  
###Wash2 SD  
Strain1 <- sd(myData[,85:86])  
Strain2 <- sd(myData[,87:88])  
Strain3 <- sd(myData[,89:90])  
Strain4 <- sd(myData[,91:92])  
W2SD = rbind(Strain1, Strain2, Strain3, Strain4)  
colnames(W2SD)[1] <- "Wash2" #name column  
W2SD = t(W2SD)  
  
SD = rbind(AVCSD, RetSD, EluSD, LysSD, LMSD, FTSD, W1SD, W2SD)  
SD = as.data.frame(round(SD))  
  
print(SD)

## Strain1 Strain2 Strain3 Strain4  
## AVC 10464 2963 4284 1068  
## Retentate 2425 350 567 60  
## Eluate 5510 368 284 199  
## Lysate 7212 181 730 36  
## LM 218 243 19 0  
## FT 6225 564 365 96  
## Wash1 38 1020 77 52  
## Wash2 2154 301 211 134

### 4: Coefficient of Variation

CV = ((SD/Mean[,1:4])\*100)  
CV = round(CV,2)  
print(CV)

## Strain1 Strain2 Strain3 Strain4  
## AVC 27.76 30.12 18.40 37.85  
## Retentate 26.55 3.87 7.51 6.21  
## Eluate 21.06 4.30 1.36 6.71  
## Lysate 15.37 2.41 14.86 2.13  
## LM 21.65 15.31 4.40 0.00  
## FT 23.22 6.06 8.21 8.44  
## Wash1 1.58 34.99 4.88 19.12  
## Wash2 21.76 4.19 5.18 25.67

### 5: Strain Specification

Each positive control will have a pass or fail based on set specification output. If one strain does not pass, the test method failed.

TMsignal = log(Mean[2,])/log(Mean[4,])  
  
TMsignal1 = TMsignal[,1:4]  
  
Strain1 = ifelse(TMsignal$Strain1 > 0.9 & TMsignal$Strain1 < 1.25, "Pass", "Fail")  
Strain2 = ifelse(TMsignal$Strain2 > 0.95 & TMsignal$Strain2 < 1.25, "Pass", "Fail")  
Strain3 = ifelse(TMsignal$Strain3 > 0.95 & TMsignal$Strain3 < 1.5, "Pass", "Fail")  
Strain4 = ifelse(TMsignal$Strain4 > 0.95 & TMsignal$Strain4 < 1.5, "Pass", "Fail")  
  
TMResult = rbind(Strain1, Strain2, Strain3, Strain4)  
TMResult = t(TMResult)  
TMResults = rbind(TMsignal1, TMResult)  
TMResults = t(TMResults)  
colnames(TMResults)[2] <- "TMSpec"   
TMResults = as.data.frame(TMResults)  
print(TMResults)

## Retentate TMSpec  
## Strain1 0.847860533521695 Fail  
## Strain2 1.02083723406451 Pass  
## Strain3 1.05063818465769 Pass  
## Strain4 0.924599428434797 Fail

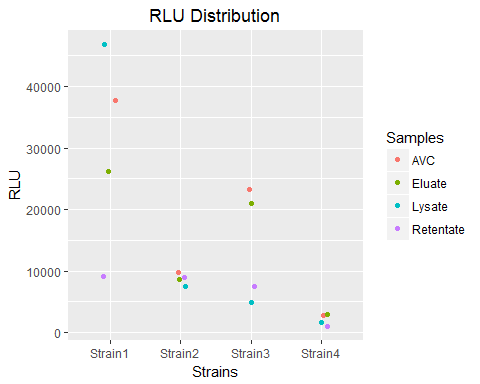
### 6:Dot plot

Plot showing RLU distribution for each strain

## Looking at 4 positive strains  
  
M1 = Mean[1:4,1:4]  
df <- melt(M1)

## No id variables; using all as measure variables

df$rowid <- c('AVC', 'Retentate', 'Eluate', 'Lysate')  
P <- ggplot(df, aes(x = factor(variable), y = value)) + geom\_jitter(width = 0.1, height = 0,   
 aes(color=factor(rowid))) + xlab("Strains") + ylab("RLU") + ggtitle("RLU Distribution") + theme(plot.title = element\_text(hjust = 0.5)) +  
 labs(color='Samples')   
  
print(P)



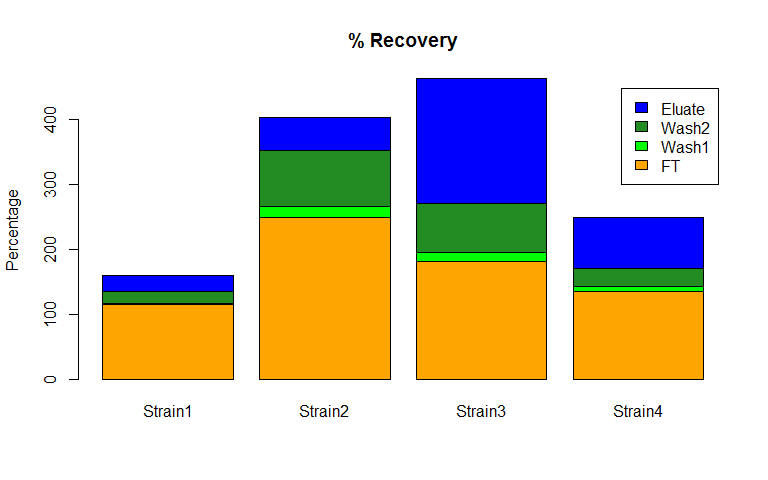
### 7: % Recovery graph

Graph showing distribution of % recovery for each strain and sample calculation output for each sample.

#Elution Volumes  
EluVol = 0.045  
LysVol = 0.1  
LMVol = 0.2  
FTVol = 0.2  
W1Vol = 0.045  
W2Vol = 0.09  
  
# % Recovery Calculations   
M2 = Mean[3:8,1:4]  
L1 = M2[2,]\*LysVol  
Lys = M2[2,]\*LysVol/L1  
LM = M2[3,]\*LMVol/L1  
FT = M2[4,]\*FTVol/L1  
Wash1 = M2[5,]\*W1Vol/L1  
Wash2 = M2[6,]\*W2Vol/L1  
Eluate = M2[1,]\*EluVol/L1  
  
R1 = rbind(Lys, LM, FT, Wash1, Wash2, Eluate)  
R1 = round(R1,2)  
R1 = R1\*100  
R1 = round(R1, 0)  
  
Recovery = R1[-1,]  
Recovery = Recovery[-1,]  
  
Strain1 = sum(Recovery[1:4, 1])  
Strain2 = sum(Recovery[1:4, 2])  
Strain3 = sum(Recovery[1:4, 3])  
Strain4 = sum(Recovery[1:4, 4])  
  
Recovery1 = rbind(Strain1, Strain2, Strain3, Strain4)  
colnames(Recovery1)[1] <- "Sum"  
Recovery1 = t(Recovery1)  
  
Recovery2 = rbind(Recovery, Recovery1)  
print(Recovery2)

## Strain1 Strain2 Strain3 Strain4  
## FT 114 248 181 135  
## Wash1 2 17 14 7  
## Wash2 19 86 75 28  
## Eluate 25 51 192 79  
## Sum 160 402 462 249

Recovery = as.matrix(Recovery)  
  
##Barplot  
barplot(Recovery, main = "% Recovery",   
 xlab = "",  
 ylab = "Percentage",  
 legend.text = TRUE,  
 col = c("Orange", "Green", "Forestgreen", "Blue"))



### 8: Data Export

Create a unique name identifier for saving files each time script is run. Example:"../Data/Plate1\_PeakValues.xlsx"

write.xlsx(PeakValues, "../data/Plate1\_PeakValues.xlsx") #Export max values of plate  
write.xlsx(Mean, "../data/Plate1\_Mean.xlsx") #Export Mean values of plate  
write.xlsx(SD, "../data/Plate1\_SD.xlsx") #Export StDev values of plate  
write.xlsx(CV, "../data/Plate1\_CV.xlsx") #Export CV values of plate  
write.xlsx(TMResults, "../data/Plate1\_TMResults.xlsx") #Export TMResults values of plate