# Independence of V and J Primers

*Wes Horton, Burcu Gurun-Demir*

*May 4, 2016*

## Summary of Dataset and Purpose

We have approximately 170 samples per sequencing batch, and for each sample we have 260 counts, one for each of the unique combinations of V and J primers. The primers have different amplification rates, which we need to characterize. In order to do this most accurately, we need to determine if the forward (V) primer and the reverse (J) primer act independently to influence spike amplification, or if their interaction is important as well.

Two different PCR batches comprise our sequencing batch. There may be a difference in amplification biases based on these batches as well. In addition, samples were diluted by varying degrees prior to sequencing, and we need to determine if that had an influence.

### Variables

1. Independent variables
   - Forward (V) primer identity - 20 total
   - Reverse (J) primer identity - 13 total
   - PCR batch identity - 2 total
   - Tape Station dilution factor
2. Dependent variable
   - Spike Count

Each sample has an individual file containing the 260 counts. These need to be combined into a single data frame prior to the analysis.

```
read.data <- function(count.dir){

  # Read in files and sort by sample number
  all.counts <- list.files(count.dir)
  all.counts <- all.counts[order(as.numeric(gsub(".*_S|.assembled.*", '',
                                                 all.counts)))]

  # Read in first file to start aggregate data frame
  count.df <- read.table(file.path(count.dir, all.counts[1]), sep =',', header = T)
  count.df <- count.df[,3:5]

  # Comine spike counts for all files into 1 data frame
    # Columns are samples
    # Rows are spikes
  for (i in 2:length(all.counts)){
    curr.df <- read.table(file.path(count.dir, all.counts[i]), sep = ',',
                          header = T)
    count.df <- cbind(count.df, curr.df$spike.count)
  }   #   for i in 2:length(new.counts)
  colnames(count.df) <- c("V", "J", seq(1:length(all.counts)))
```

```r
#head(count.df[,1:10], n = 20)

  # Collapse data frame to 1 count column
  melt.count.df <- melt(count.df, id.vars = c("V", "J"))

  # Add pseudo-variable of V/J combos
  melt.count.df$combos <- paste(melt.count.df$V, melt.count.df$J, sep = '')

  # Take log2 of count values due to geometric distribution
  log2.melt.count.df <- melt.count.df
  log2.melt.count.df$value <- log2(melt.count.df$value + 1)

  # Divide all values by upper quartile as another normalization method
  norm.melt.count.df <- melt.count.df
  norm.melt.count.df$value <- norm.melt.count.df$value /
                              summary(norm.melt.count.df$value)[5]

  # V is all V segments, repeated for each J for each sample
  # J is all J segments, repeated same as V
  # Variable corresponds to sample number
  # Value is log2 of count

  return(list("original" = count.df, "melt" = melt.count.df,
              "log2" = log2.melt.count.df, "third.q" = norm.melt.count.df))
} # read.data(count.df)

DNA150826 <- read.data("~/Desktop/OHSU/tcr_spike/data/vj_counts/DNA150826/")
DNA160107 <- read.data("~/Desktop/OHSU/tcr_spike/data/equiv_DNA160107LC/counts/")
```

## Linear Regression Model

We want to create a linear regression model in order to test the interaction effect between V and J primers on spike counts. First we'll show an additive model, then a multiplicative model.

```r
make.models <- function(batch){
  ### Using Log2 Data
  # Additive
  log2.add.lm <- lm(value ~ V + J, batch$log2)
  log2.add.adj.r2 <- round(summary(log2.add.lm)$adj.r.squared, digits = 4)
  print(paste("Log2 Additive R^2:", log2.add.adj.r2, sep = ' '))
  # Multiplicative
  log2.mult.lm <- lm(value ~ V * J, batch$log2)
  log2.mult.adj.r2 <- round(summary(log2.mult.lm)$adj.r.squared, digits = 4)
  print(paste("Log2 Multiplicative R^2:", log2.mult.adj.r2, sep = ' '))


  ### Using Third Quartile data
  # Additive
  third.q.add.lm <- lm(value ~ V + J, batch$third.q)
  third.q.add.adj.r2 <- round(summary(third.q.add.lm)$adj.r.squared, digits = 4)
  print(paste("Third Quartile Additive R^2:", third.q.add.adj.r2, sep = ' '))
```

```
  # Multiplicative
  third.q.mult.lm <- lm(value ~ V * J, batch$third.q)
  third.q.mult.adj.r2 <- round(summary(third.q.mult.lm)$adj.r.squared, digits = 4)
  print(paste("Third Quartile Multiplicative R^2:", third.q.mult.adj.r2, sep = ' '))


  return(list("log2.add" = log2.add.lm, "log2.add.r2" = log2.add.adj.r2,
         "log2.mult" = log2.mult.lm, "log2.mult.r2" = log2.mult.adj.r2,
         "third.q.add" = third.q.add.lm, "third.q.add.r2" = third.q.add.adj.r2,
         "third.q.mult" = third.q.add.lm, "third.q.mult.r2" = third.q.mult.adj.r2))
} # make.models(batch)

DNA150826.models <- make.models(DNA150826)
```

```
## [1] "Log2 Additive R^2: 0.3177"
## [1] "Log2 Multiplicative R^2: 0.4877"
## [1] "Third Quartile Additive R^2: 0.0653"
## [1] "Third Quartile Multiplicative R^2: 0.1221"
```

```
DNA160107.models <- make.models(DNA160107)
```

```
## [1] "Log2 Additive R^2: 0.2517"
## [1] "Log2 Multiplicative R^2: 0.3986"
## [1] "Third Quartile Additive R^2: 7e-04"
## [1] "Third Quartile Multiplicative R^2: 0.0051"
```

We can see that the multiplicative model explains more variation than the additive. Now we should look at specific V/J pairs to see if there are any specific combinations that are contributing to this increase.

```
steps <- function(batch){
  per.norm.method <- function(batch.method){
    print(deparse(substitute(batch.method)))
    # Null
    null <- lm(value ~ 1, data = batch.method)
    # Full
    full <- lm(value ~ ., data = batch.method)

    # Step forward
    print("Forward: ")
    forward <- step(null, scope = list(lower = null, upper = full),
                    direction = "forward")
    forward.summ <- summary(forward)
    print("Backward: ")
    backward <- step(full, scope = list(lower = null, upper = full),
                     direction = "backward")
    backward.summ <- summary(backward)
    print("Both: ")
    both <- step(full, scope = list(lower = null, upper = full),
                 direction = "both")
    both.summ <- summary(both)

    return(list("forward" = forward, "for.summary" = forward.summ,
```

```r
           "backward" = backward, "back.summary" = backward.summ,
           "both" = both, "both.summary" = both.summ))
  } # per.norm.method(batch.method)

  batch.log2 <- per.norm.method(batch$log2)
  batch.third.q <- per.norm.method(batch$third.q)

  return(list("log2" = batch.log2,
              "third.quartile" = batch.third.q))
} # steps(batch)

DNA150826.steps <- steps(DNA150826)
```

```
## [1] "batch$log2"
## [1] "Forward: "
## Start:  AIC=85392.46
## value ~ 1
##
##             Df Sum of Sq    RSS   AIC
## + combos   259    152291 158008 56607
## + variable 166    148118 162181 57552
## + V         19     74326 235973 73541
## + J         12     24406 285893 81860
## <none>                   310299 85392
##
## Step:  AIC=56606.74
## value ~ combos
##
##             Df Sum of Sq    RSS    AIC
## + variable 166    148118   9890 -63384
## <none>                   158008  56607
##
## Step:  AIC=-63383.82
## value ~ combos + variable
##
##        Df Sum of Sq    RSS    AIC
## <none>              9889.9 -63384
## [1] "Backward: "
## Start:  AIC=-63383.82
## value ~ V + J + variable + combos
##
##
## Step:  AIC=-63383.82
## value ~ V + variable + combos
##
##
## Step:  AIC=-63383.82
## value ~ variable + combos
##
##             Df Sum of Sq    RSS    AIC
## <none>                     9890 -63384
## - variable 166    148118 158008  56607
## - combos   259    152291 162181  57552
```

```
## [1] "Both: "
## Start:  AIC=-63383.82
## value ~ V + J + variable + combos
##
##
## Step:  AIC=-63383.82
## value ~ V + variable + combos
##
##
## Step:  AIC=-63383.82
## value ~ variable + combos
##
##             Df Sum of Sq    RSS    AIC
## <none>                     9890 -63384
## - variable 166    148118 158008  56607
## - combos   259    152291 162181  57552
## [1] "batch$third.q"
## [1] "Forward: "
## Start:  AIC=112599
## value ~ 1
##
##             Df Sum of Sq    RSS    AIC
## + variable 166    185348 395289  96236
## + combos   259     73957 506680 107201
## + V         19     22440 558197 110926
## + J         12     15851 564786 111421
## <none>                   580637 112599
##
## Step:  AIC=96235.61
## value ~ variable
##
##           Df Sum of Sq    RSS   AIC
## + combos 259     73957 321332 87759
## + V       19     22440 372849 93736
## + J       12     15851 379438 94483
## <none>                 395289 96236
##
## Step:  AIC=87759.48
## value ~ variable + combos
##
##        Df Sum of Sq    RSS   AIC
## <none>              321332 87759
## [1] "Backward: "
## Start:  AIC=87759.48
## value ~ V + J + variable + combos
##
##
## Step:  AIC=87759.48
## value ~ V + variable + combos
##
##
## Step:  AIC=87759.48
## value ~ variable + combos
##
```

```
##             Df Sum of Sq    RSS    AIC
## <none>                   321332  87759
## - combos   259     73957 395289  96236
## - variable 166    185348 506680 107201
## [1] "Both: "
## Start:  AIC=87759.48
## value ~ V + J + variable + combos
##
##
## Step:  AIC=87759.48
## value ~ V + variable + combos
##
##
## Step:  AIC=87759.48
## value ~ variable + combos
##
##             Df Sum of Sq    RSS    AIC
## <none>                   321332  87759
## - combos   259     73957 395289  96236
## - variable 166    185348 506680 107201
```

```r
DNA160107.steps <- steps(DNA160107)
```

```
## [1] "batch$log2"
## [1] "Forward: "
## Start:  AIC=73490.8
## value ~ 1
##
##             Df Sum of Sq    RSS   AIC
## + combos   259     94089 139907 51409
## + variable 168     77749 156247 56081
## + V         19     45727 188269 63975
## + J         12     13292 220705 70945
## <none>                   233996 73491
##
## Step:  AIC=51409.3
## value ~ combos
##
##             Df Sum of Sq    RSS   AIC
## + variable 168     77749  62158 16097
## <none>                   139907 51409
##
## Step:  AIC=16096.8
## value ~ combos + variable
##
##        Df Sum of Sq   RSS   AIC
## <none>              62158 16097
## [1] "Backward: "
## Start:  AIC=16096.8
## value ~ V + J + variable + combos
##
##
## Step:  AIC=16096.8
## value ~ V + variable + combos
```

```
##
##
## Step:  AIC=16096.8
## value ~ variable + combos
##
##            Df Sum of Sq    RSS   AIC
## <none>                   62158 16097
## - variable 168     77749 139907 51409
## - combos   259     94089 156247 56081
## [1] "Both: "
## Start:  AIC=16096.8
## value ~ V + J + variable + combos
##
##
## Step:  AIC=16096.8
## value ~ V + variable + combos
##
##
## Step:  AIC=16096.8
## value ~ variable + combos
##
##            Df Sum of Sq    RSS   AIC
## <none>                   62158 16097
## - variable 168     77749 139907 51409
## - combos   259     94089 156247 56081
## [1] "batch$third.q"
## [1] "Forward: "
## Start:  AIC=412393
## value ~ 1
##
##            Df Sum of Sq       RSS    AIC
## + V        19    490990 522932549 412390
## <none>                  523423539 412393
## + J        12    237809 523185730 412397
## + combos   259   5747601 517675938 412426
## + variable 168   3440813 519982725 412439
##
## Step:  AIC=412389.8
## value ~ V
##
##            Df Sum of Sq       RSS    AIC
## <none>                  522932549 412390
## + J        12    237809 522694740 412394
## + combos   240   5256611 517675938 412426
## + variable 168   3440813 519491735 412436
## [1] "Backward: "
## Start:  AIC=412468.8
## value ~ V + J + variable + combos
##
##
## Step:  AIC=412468.8
## value ~ V + variable + combos
##
##
```

```
## Step:  AIC=412468.8
## value ~ variable + combos
##
##            Df Sum of Sq       RSS    AIC
## - variable 168   3440813 517675938 412426
## - combos   259   5747601 519982725 412439
## <none>                   514235125 412469
##
## Step:  AIC=412425.8
## value ~ combos
##
##          Df Sum of Sq       RSS    AIC
## - combos 259   5747601 523423539 412393
## <none>                 517675938 412426
##
## Step:  AIC=412393
## value ~ 1
##
## [1] "Both: "
## Start:  AIC=412468.8
## value ~ V + J + variable + combos
##
##
## Step:  AIC=412468.8
## value ~ V + variable + combos
##
##
## Step:  AIC=412468.8
## value ~ variable + combos
##
##            Df Sum of Sq       RSS    AIC
## - variable 168   3440813 517675938 412426
## - combos   259   5747601 519982725 412439
## <none>                   514235125 412469
##
## Step:  AIC=412425.8
## value ~ combos
##
##            Df Sum of Sq       RSS    AIC
## - combos   259   5747601 523423539 412393
## <none>                   517675938 412426
## + variable 168   3440813 514235125 412469
##
## Step:  AIC=412393
## value ~ 1
##
##            Df Sum of Sq       RSS    AIC
## + V         19    490990 522932549 412390
## <none>                   523423539 412393
## + J         12    237809 523185730 412397
## + combos   259   5747601 517675938 412426
## + variable 168   3440813 519982725 412439
##
## Step:  AIC=412389.8
```

```
## value ~ V
##
##              Df Sum of Sq       RSS    AIC
## <none>                   522932549 412390
## - V          19    490990 523423539 412393
## + J          12    237809 522694740 412394
## + combos    240   5256611 517675938 412426
## + variable 168   3440813 519491735 412436
```

Here is a different step function attempt:

```r
# Need to rearrange the data frame
# First transform, then create new column with rowsums
#primer.names <- paste(count.df$V, count.df$J, sep = '')

#new.count <- cbind(primer.names,
#                   count.df[,3:169], stringsAsFactors = F)
#new.count <- as.data.frame(t(new.count), stringsAsFactors = F)
#new.count <- new.count[c(2:168),]
#new.count <- apply(new.count, c(1,2), function(x) as.numeric(x))
#count.total <- apply(new.count, 1, sum)
#colnames(new.count) <- primer.names
#new.count <- cbind("total" = count.total, new.count)
#new.count <- data.frame(new.count)

#test.lm <- lm(total ~ ., new.count)

# This doesn't work either...
```

Also going to do a chi squared test. Need to format the data so that we have V's as rows and J's as columns

```r
# Need to create a 20 x 13 matrix of V and J counts for chi-squared test
# This empty matrix will be populated by counts
vs <- unique(DNA150826$original$V)
js <- unique(DNA150826$original$J)
v.j.df <- data.frame(matrix(nrow = length(vs), ncol = length(js)))
rownames(v.j.df) <- vs
colnames(v.j.df) <- js

# We have two options for running the chi-squared. We can sum the counts of all of the samples and do o

# Variance and standard deviation of each of the 260 spikes
#spike.var <- apply(subset.count, 1, var)
#spike.sd <- apply(subset.count, 1, sd)

# This is for the second option - 170 individual chi-squared tests:
# Run Chi-squared on each individually and extract p value
chi.sq.ps <- NULL
log.chi.sq.ps <- NULL
for (i in 3:length(names(DNA150826$original))){
  curr.df <- DNA150826$original[,c(1:2,i)]
  curr.vj <- populate.vj.df(curr.df, v.j.df)
  log.curr.vj <- log2(curr.vj + 1)
  curr.chi <- chisq.test(curr.vj, correct = F)
```

```r
  curr.log.chi <- chisq.test(log.curr.vj, correct = F)
  curr.log.p <- curr.log.chi$p.value
  curr.p <- curr.chi$p.value
  chi.sq.ps <- c(chi.sq.ps, curr.p)
  log.chi.sq.ps <- c(log.chi.sq.ps, curr.log.p)
}  # for
```

```
## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect
```

```
## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect
```

```
## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect
```

```
## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect
```

```
## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect
```

```
## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect
```

```
## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect
```

```
## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect
```

```
## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect
```

```
## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect
```

```
## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect
```

```
## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect
```

```
## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(curr.vj, correct = F): Chi-squared approximation may
## be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect

## Warning in chisq.test(log.curr.vj, correct = F): Chi-squared approximation
## may be incorrect
```

```r
# Returns a vector of p-values for the chi-squared tests. They're all zero, which means that the null hy


# Second option: Run chi-squared for the sum of all of the counts.
# We want to scale the counts within samples due to the high variation between samples.
# Extract counts only
# Rows are spikes, columns are samples
subset.count <- DNA150826$original[,3:169]


# Extract column totals - this is a sum of all 260 spike counts for a given sample
sum.of.samples <- apply(subset.count, 2, sum)

# Divide each cell in a sample by its sum
# Since the sum is a vector of length(samples) and our data frame is structured as columns = length(sam
scaled.counts <- apply(subset.count, 1, function(x) x / sum.of.samples)
# Rows are now samples and columns are spikes...
rownames(scaled.counts) <- seq(1:167)
colnames(scaled.counts) <- paste(DNA150826$original$V, DNA150826$original$J, sep = '')

# Variance and Standard Deviation of scaled counts
scaled.spike.var <- apply(scaled.counts, 2, var)
scaled.spike.sd <- apply(scaled.counts, 2, sd)

# Now that we've scaled each of the counts, we want to sum each row so that we get a total count for ea
# Sum each row
count.sums.by.sample <- apply(DNA150826$original[,3:169], 1, sum)

# Recombine with names for population
count.sums.by.sample <- cbind(DNA150826$original[,1:2], count.sums.by.sample)


# Call function
v.j.df <- populate.vj.df(count.sums.by.sample, v.j.df)



# Now use the 20 x 13 matrix to run a chi-squared
scaled.chi <- chisq.test(v.j.df, correct = F)
scaled.chi
```

```
##
##  Pearson's Chi-squared test
##
## data:  v.j.df
## X-squared = 9121700, df = 228, p-value < 2.2e-16
```

```r
# We get a p-value of 2.2 e -16, which tells us we should reject the null hypothesis that primers are i
scaled.chi$observed
```

```
##            J1-1   J1-2   J1-3   J1-4   J1-5   J1-6   J1-7   J2-1   J2-2
## V1-      164461 659342  15796 142615 313415 187385 319250 334708  51358
## V2-       55946 186943   3314  53977  46862  95053  67568  98240  31172
```

```
## V3-        71794 488137  49384  13762  17377  90268 129114  22406 107036
## V4-        52569  96077  18796   8168  12492  38417  61163   6793  66920
## V5-        63701  59448  44839   7347   4748  49485 147094  15443  74988
## V12-1-2- 145812 736609  17540  14258 136605  54735  81418 247712  16037
## V13-1-   125859 168332 169932 174064  88547 173671 110632 313296 252701
## V13-2-   104517 173779  82898 176859  68721 188346  41709 221675 327661
## V13-3-   119624 102795  46313  58918  71071  70382  87971 122657 142007
## V14-     194592 418060  95118  12853  11372 102925 180394  57530 246824
## V15-      43910  88244  54962   8358   8721  60526 116577  36652 291774
## V16-     141360 180774  24337  68843 111342 188691 153056 170211   8040
## V17-      59708 113969   3818  51108  86707  41872  71193  93711  13969
## V19-     106128 529990  65760   5912   9089 148622 203774  41815 264311
## V20-      14965  12646   6987  14353   9608   5508  13624   8118    939
## V23-      98588 345523  10781  49374 118488 155877 105136 302539 149727
## V24-      86406 296022  72737  79286  68629 206493 313531 108101 175240
## V26-      42147  59942   9671   5382   6782  33664  40377   3223  26137
## V29-      88498 161844  42365 106329 119265 129692  89441 204606 162990
## V30-     162016 271125   3735  22536  61251  21498  44417  37084   8972
##            J2-3   J2-4   J2-5   J2-7
## V1-      357564 330825 102979  20114
## V2-       71325  73800  80342  53733
## V3-       15224   9780  13507  82253
## V4-        8116   4553   4743  12360
## V5-       15044   2256   2198  49303
## V12-1-2- 356988 160890 265551  40684
## V13-1-   300314 330440 240190 221975
## V13-2-   193745 146059 124648 215355
## V13-3-   241953  64909 173363 111625
## V14-      29048  37549  61547 142273
## V15-      36672  15640  10988 123902
## V16-     246242 254105 217397  30233
## V17-      92319  59686  62884   6691
## V19-      27619  40720  35233 155014
## V20-       7282  10090  12433  13693
## V23-     227680 192667 186522  43394
## V24-     206054 211519 101392 134527
## V26-       3189   3725   2653  24979
## V29-     286216 102235 134135  99101
## V30-      92646  27834  76090  17633
```

```
scaled.chi$expected
```

```
##                J1-1      J1-2      J1-3       J1-4       J1-5       J1-6
## V1-      207642.901 550436.29 89688.839 114831.189 146554.810 218386.219
## V2-       63561.745 168494.52 27454.727  35151.073  44862.017  66850.391
## V3-       76835.595 203681.90 33188.206  42491.811  54230.730  80811.022
## V4-       27076.047  71775.34 11695.171  14973.662  19110.333  28476.945
## V5-       37093.853  98331.33 16022.241  20513.734  26180.922  39013.066
## V12-1-2- 157461.257 417410.80 68013.485  87079.613 111136.497 165608.207
## V13-1-   184810.510 489910.38 79826.664 102204.364 130439.659 194372.494
## V13-2-   143003.768 379085.74 61768.747  79084.297 100932.370 150402.696
## V13-3-    97846.636 259379.63 42263.671  54111.388  69060.368 102909.162
## V14-     110063.518 291765.11 47540.605  60867.598  77683.070 115758.138
## V15-      62083.996 164577.19 26816.431  34333.845  43819.019  65296.184
```

```
## V16-      124221.913 329297.31 53656.153  68697.509  87676.096 130649.080
## V17-       52442.463 139018.64 22651.887  29001.860  37014.004  55155.804
## V19-      113102.355 299820.70 48853.194  62548.143  79827.887 118954.202
## V20-        9015.451  23898.87  3894.115   4985.747   6363.125   9481.905
## V23-      137488.704 364465.97 59386.582  76034.342  97039.825 144602.286
## V24-      142586.033 377978.38 61588.312  78853.280 100637.532 149963.348
## V26-       18126.354  48050.78  7829.459  10024.281  12793.620  19064.201
## V29-      119520.999 316835.76 51625.649  66097.798  84358.180 125704.943
## V30-       58616.904 155386.34 25318.862  32416.465  41371.938  61649.707
##                J1-7      J2-1      J2-2      J2-3       J2-4      J2-5
## V1-       254122.35 261506.36 258543.71 300918.51 222252.612 204029.41
## V2-        77789.61  80049.93  79143.03  92114.42  68033.936  62455.62
## V3-        94034.72  96767.08  95670.79 111351.04  82241.732  75498.47
## V4-        33136.84  34099.69  33713.37  39238.92  28981.112  26604.86
## V5-        45397.06  46716.16  46186.90  53756.84  39703.768  36448.33
## V12-1-2- 192707.89 198307.38 196060.73 228194.69 168540.198 154721.05
## V13-1-   226179.08 232751.15 230114.27 267829.54 197813.739 181594.36
## V13-2-   175014.19 180099.56 178059.18 207242.73 153065.483 140515.15
## V13-3-   119748.94 123228.47 121832.40 141800.48 104731.105  96143.87
## V14-     134700.49 138614.46 137044.08 159505.33 117807.564 108148.14
## V15-      75981.08  78188.85  77303.04  89972.85  66452.213  61003.58
## V16-     152028.14 156445.61 154673.21 180023.84 132962.141 122060.15
## V17-      64181.35  66046.26  65298.01  76000.23  56132.303  51529.84
## V19-     138419.55 142441.59 140827.85 163909.25 121060.213 111134.10
## V20-      11033.50  11354.10  11225.46  13065.30   9649.776   8858.56
## V23-     168264.61 173153.86 171192.17 199250.23 147162.380 135096.06
## V24-     174502.95 179573.46 177539.04 206637.34 152618.357 140104.69
## V26-      22183.81  22828.41  22569.78  26268.92  19401.720  17810.91
## V29-     146274.96 150525.26 148819.93 173211.22 127930.471 117441.04
## V30-      71737.90  73822.38  72986.03  84948.30  62741.177  57596.83
##                J2-7
## V1-      170898.806
## V2-       52313.979
## V3-       63238.914
## V4-       22284.721
## V5-       30529.795
## V12-1-2- 129597.211
## V13-1-   152106.792
## V13-2-   117698.091
## V13-3-    80531.880
## V14-      90586.886
## V15-      51097.729
## V16-     102239.838
## V17-      43162.344
## V19-      93087.976
## V20-       7420.094
## V23-     113158.963
## V24-     117354.278
## V26-      14918.749
## V29-      98370.789
## V30-      48244.167
```

```
# We can take a look at the residuals to try and determine which ones are causing the dependence
# Calculate standardized residuals
```

```r
scaled.std.resid <- (scaled.chi$observed - scaled.chi$expected) / (sqrt(scaled.chi$expected))

# Calulate adjusted standardized residuals
adj.std.resid <- round(((scaled.chi$observed - scaled.chi$expected) /
                        sqrt(scaled.chi$expected * ((1 - rowSums(scaled.chi$observed) /
                                                         sum(scaled.chi$observed))
                                                      %*% t(1 - colSums(scaled.chi$observed) /
                                                              sum(scaled.chi$observed))))),
                       digits = 1)
write.csv(adj.std.resid, file = "~/Desktop/chi.sq.resids.csv", quote = F)

# Chi squared with log2 data
# Take the log2 of the counts
log2.v.j.df <- log2(v.j.df)

# Chi squared
log2.chi <- chisq.test(log2.v.j.df, correct = F)
log2.chi
```

```
##
##  Pearson's Chi-squared test
##
## data:  log2.v.j.df
## X-squared = 24.058, df = 228, p-value = 1
```

```r
# Calculate standardized residuals
log2.std.resid <- (log2.chi$observed - log2.chi$expected) / (sqrt(log2.chi$expected))

# Calulate adjusted standardized residuals
log2.adj.std.resid <- round(((log2.chi$observed - log2.chi$expected) /
                        sqrt(log2.chi$expected * ((1 - rowSums(log2.chi$observed) /
                                                         sum(log2.chi$observed))
                                                      %*% t(1 - colSums(log2.chi$observed) /
                                                              sum(log2.chi$observed))))),
                       digits = 1)

#######
#######
####### Old Stuff
# Sum rows - this is a sum of counts across all samples, 1 for each spike
#sum.of.rows <- apply(subset.count, 1, sum)

# Take mean of each spike count across the samples
#count.mean <- sum.of.rows / length(names(subset.count))

# Create scaling factor by dividing count mean by 1
#scaling.factor <- 1 / count.mean

# Multiply each column (sample) by scaling factor
# This should cause each row to be multiplied by the same scaling factor
#scaled.counts <- apply(subset.count, 2, function(x) (x * scaling.factor))

# Sum scaled counts
```

```r
#sum.scaled.rows <- apply(scaled.counts, 1, sum)
```