VariablesEffectingPerformance

David A Hughes

27/09/2019

What are the variables that are influencing the performance of our experiments??

1. First how do we define and/or measure performance ? + the number of reliable reads: reads retained after QC and mapping filtering (secondary alignments and mapping quality score < 30) + enrichment factor (EF): (reliable-on-target reads / production reads) / (target space/genomic space) + library complexity (LC): # reliable reads / total number of mapped reads, including duplicates + capture sensitivity (CS): # of target regions covered by 1 read / total number of target regions + capture specificity (CSp): reliable-on-target / reliable reads

2. What are some of the variable that we think may influence performance? + geogrpahic sampling site + % endogenous DNA + DNA fragment size + the sample poolit belongs to + the hybridization + production or sequencing reads acquired for a sample/hybridization + pipeting volume used to make a library + pipeting volume used to make the pool

Read in the data and report the variables available in each table

```
########################
## Read in the data
########################
n = excel_sheets("data/SupplementaryMaterial2.xlsx")
##
mydata = sapply(1:length(n), function(x){
  read_excel("data/SupplementaryMaterial2.xlsx", sheet = x)
  })
```

```
## New names:
## * `` -> ...12
```

```
names(mydata) = n

########################
## What data is available
## in each sheet ?
########################
lapply(mydata, names)
```

```
## $`Table S1`
## [1] "Sample"                           "Site"
## [3] "Subspecies"                       "Common name"
## [5] "Total DNA Concentration (ng/ul)" "Endogenous DNA (qPCR - ng/ul)"
## [7] "% Endogenous DNA"                 "Average Fragment Size"
##
## $`Table S2`
## [1] "Sample"                           "Site"
## [3] "Subspecies"                       "Common name"
## [5] "Total DNA Concentration (ng/ul)" "Endogenous DNA (qPCR - pg/ul)"
## [7] "% Endogenous DNA"
```

```
##
## $`Table S3`
## [1] "Sites"              "Median Endogenous"  "Average Endogenous"
## [4] "Min"                "Max"
##
## $`Table S4`
##  [1] "Extract ID"
##  [2] "Sequencing Batch"
##  [3] "Capture Pool"
##  [4] "Starting DNA  (ug)"
##  [5] "Production Reads"
##  [6] "Production Bases"
##  [7] "Mapped Reads"
##  [8] "Percentage Mapped Reads"
##  [9] "Unique Reads"
## [10] "Percentage Unique Reads"
## [11] "Reliable Reads"
## [12] "Percentage Reliable Reads"
## [13] "OnTarget Reliable Reads"
## [14] "OnTarget Reliable Bases"
## [15] "Percentage OnTarget Reliable Reads"
## [16] "Percentage OnTarget Reliable Bases"
## [17] "Coverage OnTarget"
## [18] "Enrichment Factor (EF)"
## [19] "Capture Specificity (CSp)"
## [20] "Library Complexity (LC)"
## [21] "Capture Sensitivity (CS) DP1"
## [22] "Capture Sensitivity (CS) DP4"
## [23] "Capture Sensitivity (CS) DP10"
## [24] "Capture Sensitivity (CS) DP50"
##
## $`Table S5`
##  [1] "Capture Pool"
##  [2] "Production Reads"
##  [3] "Mapped Reads"
##  [4] "Percentage Mapped Reads"
##  [5] "Unique Reads"
##  [6] "Percentage Unique Reads"
##  [7] "Uniq HQ Reads"
##  [8] "Percentage Unique HQ Reads"
##  [9] "OnTarget Uniq HQ Reads"
## [10] "Percentage OnTargetUnique HQ Reads"
## [11] "Average Coverage OnTarget"
## [12] "...12"
##
## $Downsampled
##  [1] "Extract ID"
##  [2] "Sequencing Batch"
##  [3] "Capture Pool"
##  [4] "Starting DNA (ug)"
##  [5] "Production Reads"
##  [6] "Production Bases"
##  [7] "Mapped Reads"
##  [8] "Percentage Mapped Reads"
```

```
##  [9] "Unique Reads"
## [10] "Percentage Unique Reads"
## [11] "Reliable Reads"
## [12] "Percentage Reliable Reads"
## [13] "OnTarget Reliable Reads"
## [14] "OnTarget Reliable Bases"
## [15] "Percentage OnTarget Reliable Reads"
## [16] "Percentage OnTarget Reliable Bases"
## [17] "Coverage OnTarget"
## [18] "Enrichment"
## [19] "Specificity"
## [20] "LC"
## [21] "DP1"
## [22] "DP4"
## [23] "DP10"
## [24] "DP20"
## [25] "DP50"
```

Add the data from Table S2 to Table S4

- include a new variable accounting for the total amount of DNA used in a hybridization

```
m = match( unlist( mydata[[4]][,1] ), unlist( mydata[[2]][,1]  ) )

mydata[[4]] =  as_tibble( cbind( mydata[[4]][,1], mydata[[2]][m, -1], mydata[[4]][, -1] ) )

## convert characters to factors
# mydata[[4]] %>% mutate_if(is.character, as.factor) %>% str()

mydata[[4]] = mydata[[4]] %>% mutate_if(is.character, as.factor)

## Set the working data frame to wdata
wdata = mydata[[4]]

wdata = wdata %>% mutate(TotalDNA_inHyb = 1)
w = which(wdata$`Sequencing Batch` == 'SeqBatch 3')
wdata$TotalDNA_inHyb[w] = 2
```

Add the data from Table S2 to Table Downsampled

- include a new variable accounting for the total amount of DNA used in a hybridization

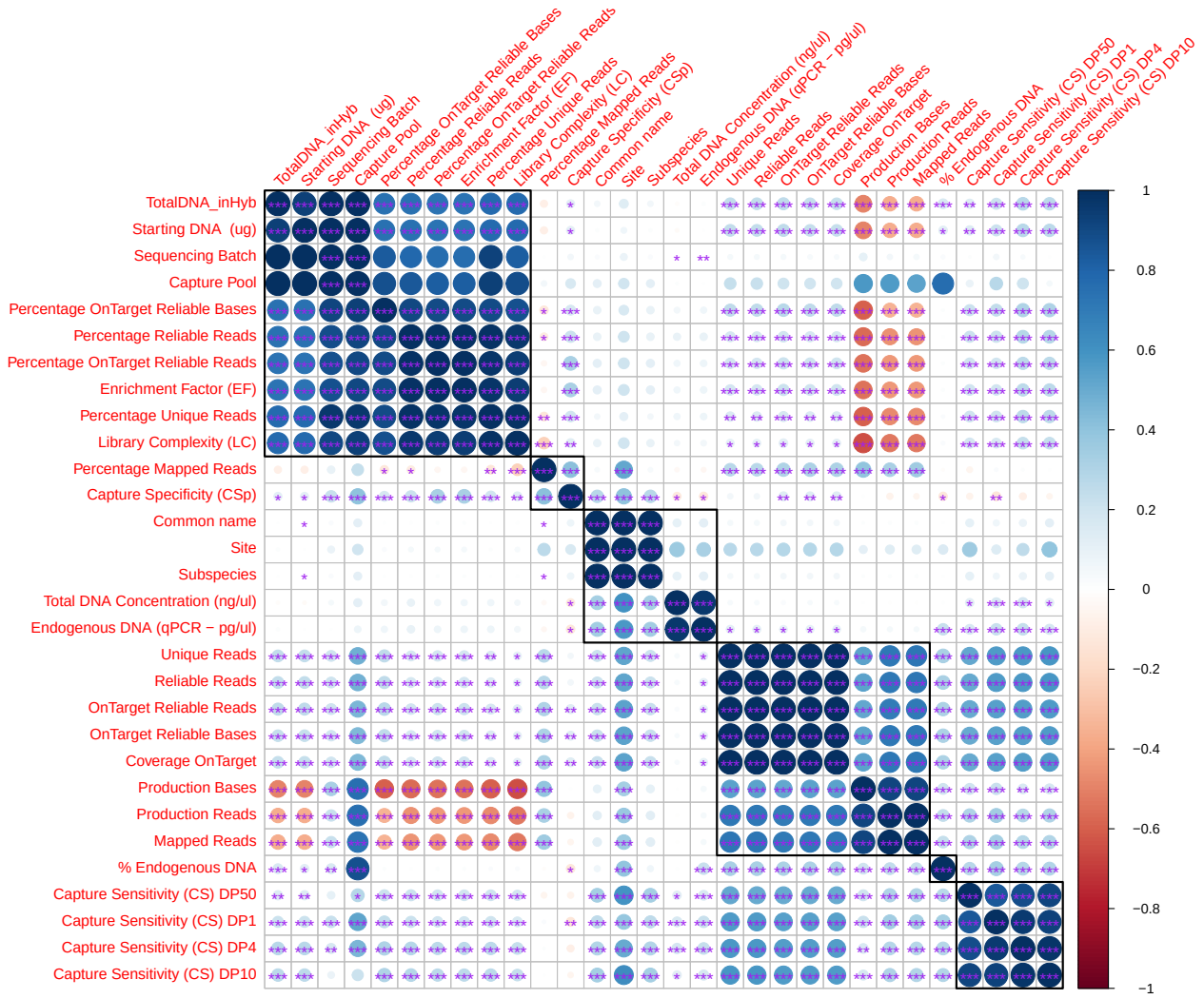
```
m = match( unlist( mydata[[6]][,1] ), unlist( mydata[[6]][,1]  ) )

mydata[[6]] =  as_tibble( cbind( mydata[[6]][,1], mydata[[2]][m, -1], mydata[[6]][, -1] ) )

## convert characters to factors
# mydata[[4]] %>% mutate_if(is.character, as.factor) %>% str()

mydata[[6]] = mydata[[6]] %>% mutate_if(is.character, as.factor)

## Set the working data frame to wdata
downdata = mydata[[6]]
```

```
downdata = downdata %>% mutate(TotalDNA_inHyb = 1)
w = which(downdata$`Sequencing Batch` == 'SeqBatch 3')
downdata$TotalDNA_inHyb[w] = 2
```

Using the funtions in the NILC R package, estimate a correlation matrix among all variables in the study

```
CorMat = Test_DF_Correlations( wdata )
rownames(CorMat[[1]]) = colnames(CorMat[[1]]) = names( wdata )
rownames(CorMat[[2]]) = colnames(CorMat[[2]]) = names( wdata )
```

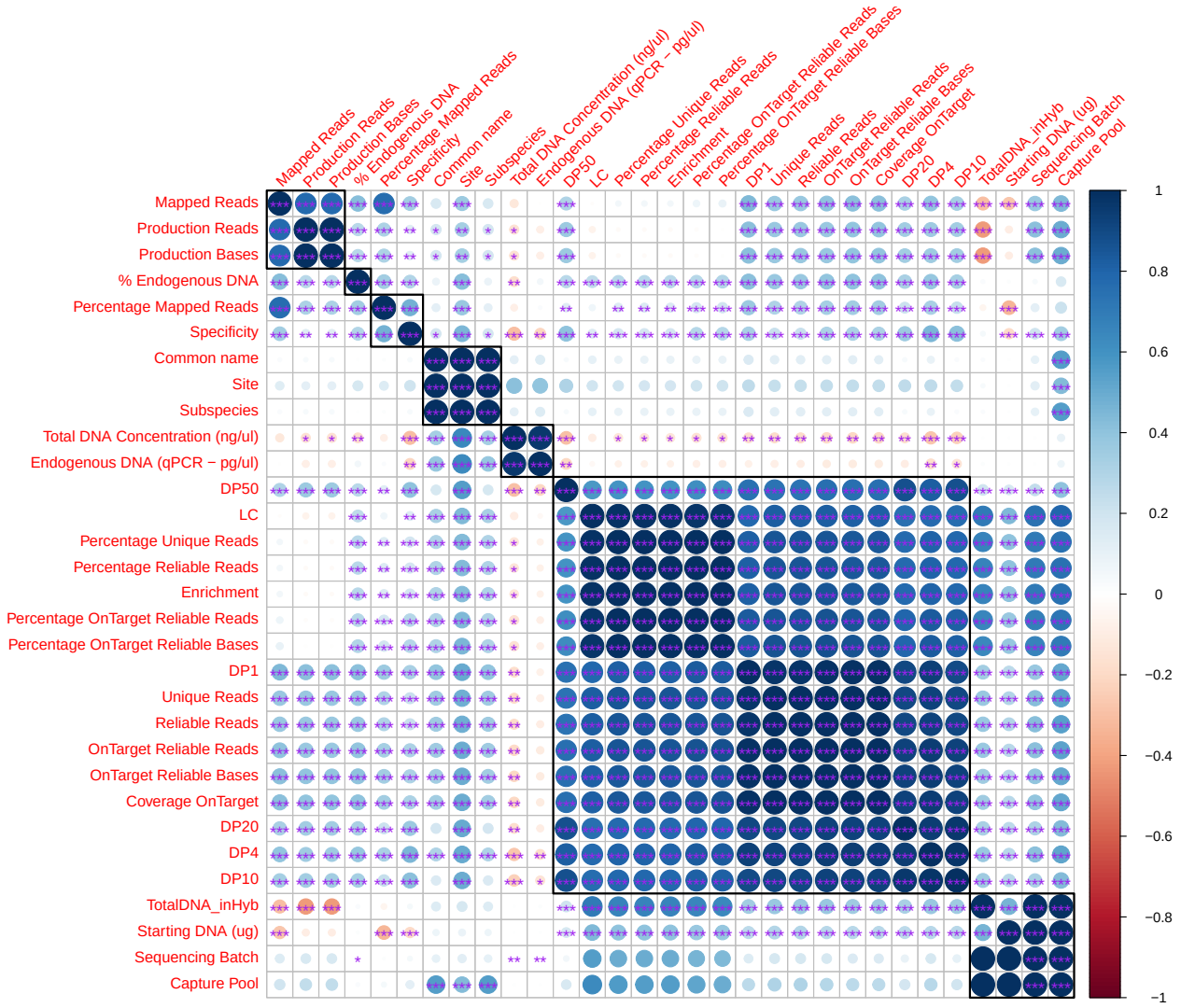
```
corrplot(CorMat[[1]][-1,-1],
         order = "hclust",
         addrect = 6,
         tl.col = "red",
         tl.cex = 0.95,
         tl.srt = 45, method = "cir",
         p.mat = CorMat[[2]][-1,-1],
         insig = "label_sig",
         sig.level = c(.001, .01, .05),
         #sig.level = 0.05,
         pch.cex = 1.0,
         pch.col = "purple")
```



Correlation matrix for the downsampled data

```
DownCorMat = Test_DF_Correlations( downdata )
rownames(DownCorMat[[1]]) = colnames(DownCorMat[[1]]) = names( downdata )
rownames(DownCorMat[[2]]) = colnames(DownCorMat[[2]]) = names( downdata )
```

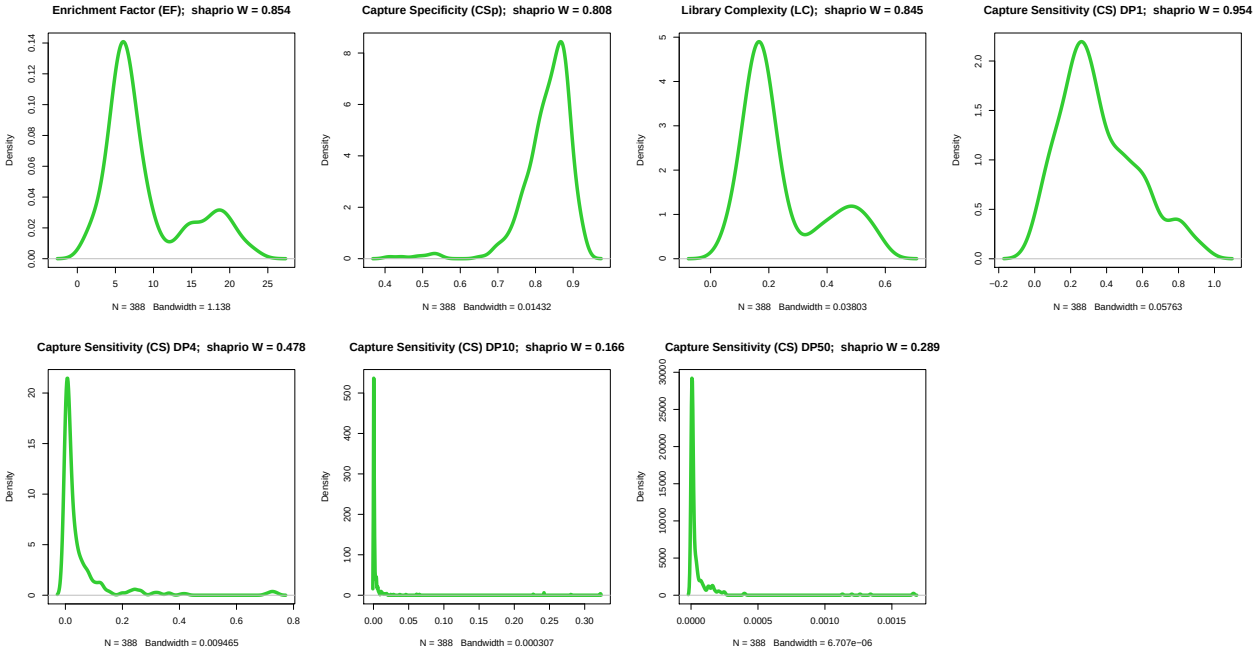
```
corrplot(DownCorMat[[1]][-1,-1],
         order = "hclust",
         addrect = 7,
         tl.col = "red",
         tl.cex = 0.95,
         tl.srt = 45, method = "cir",
         p.mat = DownCorMat[[2]][-1,-1],
         insig = "label_sig",
         sig.level = c(.001, .01, .05),
         #sig.level = 0.05,
         pch.cex = 1.0,
         pch.col = "purple")
```



Distribution of summary statistics

Are the distributions normal??

```
par(mfrow = c(2,4))
invisible(
  sapply(24:30, function(i){
  d = na.omit( unlist( wdata[, i] ) )
  Wstat = signif( shapiro.test( d )$statistic , d = 3)
  plot( density( d) , lwd = 4, col = "limegreen",
        main = paste0( colnames(wdata)[i], ";  shaprio W = ", Wstat)  )
  })
)
```



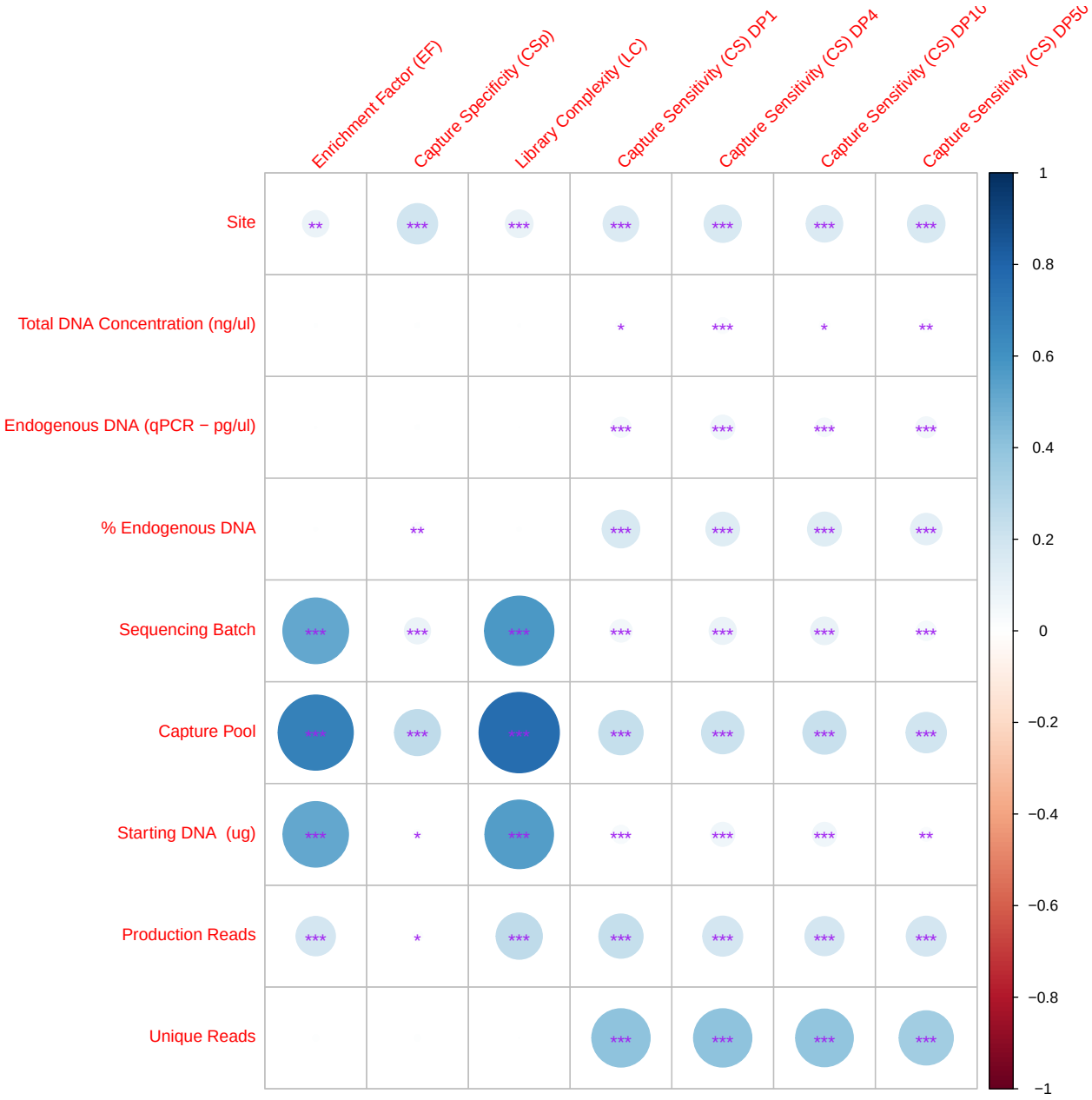
Explicity Univariate Analysis:

EF, LC, CS, CSp: as influenced by site, DNA [concentration], %eDNA, fragment size, pool, amount of DNA in hybridaztion, hybridization, Sequencing run, production reads

```
## using wdata data frame as input
dep_cols = 24:30
ind_cols = c(2, 5:11,15)

UnivarMat = lapply(dep_cols, function(dep){
  out = sapply(ind_cols, function(ind){
    df = data.frame( dep = unlist(wdata[, dep]), ind = unlist(wdata[, ind]) )
    df$dep = rntransform( df$dep )
    fit = lm( dep ~ ind, data = df)
    s = summary(fit)
    #####
    o = c(s$r.squared, s$adj.r.squared, s$fstatistic[1])
    names(o) = c("Rsq","Adj_Rsq","Fstat")
    #####
    a = anova(fit)
    eta = a[1,2]/sum(a[,2])
    Fstat = a[1, 4]
    pval = a[1, 5]
    o = c(o, eta, Fstat, pval)
    names(o) = c("Rsq","Adj_Rsq","Fstat", "EtaSq", "Fstat_","pval")
    #####
    return(o)
    })
  out = t(out)
  rownames(out) = colnames(wdata)[ind_cols]
  ##
```

```
  return(out)
  })

names(UnivarMat) = colnames(wdata)[dep_cols]
```

```
testmat = sapply(1:length(UnivarMat), function(x){ return( UnivarMat[[x]][,1] ) })
pmat = sapply(1:length(UnivarMat), function(x){ return( UnivarMat[[x]][,6] ) })
####
colnames(testmat) = names(UnivarMat)
corrplot(testmat,
         tl.col = "red",
         tl.cex = 0.95,
         tl.srt = 45, method = "cir",
         p.mat = pmat,
         insig = "label_sig",
         sig.level = c(.001, .01, .05),
         #sig.level = 0.05,
         pch.cex = 1.0,
         pch.col = "purple")
```



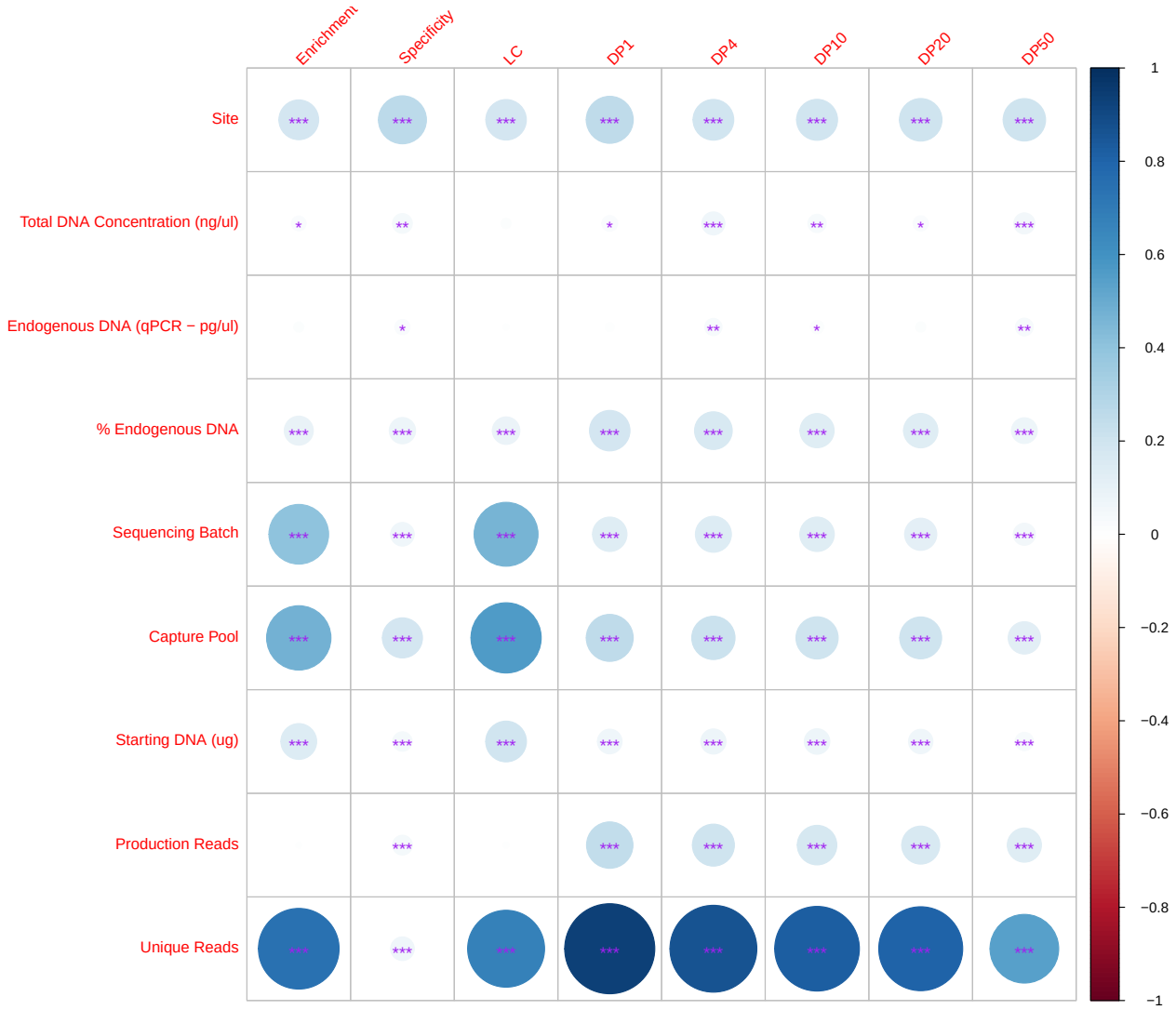
Explicity Univariate Analysis on downsampled data:

EF, LC, CS, CSp: as influenced by site, DNA [concentration], %eDNA, fragment size, pool, amount of DNA in hybridaztion, hybridization, Sequencing run, production reads

```
## using wdata data frame as input
dep_cols = 24:31
ind_cols = c(2, 5:11, 15)
#ind_cols = c(2, 5:23)

UnivarMat = lapply(dep_cols, function(dep){
  out = sapply(ind_cols, function(ind){
```

```
    df = data.frame( dep = unlist(downdata[, dep]), ind = unlist(downdata[, ind]) )
    df$dep = rntransform( df$dep )
    fit = lm( dep ~ ind, data = df)
    s = summary(fit)
    #####
    o = c(s$r.squared, s$adj.r.squared, s$fstatistic[1])
    names(o) = c("Rsq","Adj_Rsq","Fstat")
    #####
    a = anova(fit)
    eta = a[1,2]/sum(a[,2])
    Fstat = a[1, 4]
    pval = a[1, 5]
    o = c(o, eta, Fstat, pval)
    names(o) = c("Rsq","Adj_Rsq","Fstat", "EtaSq", "Fstat_","pval")
    #####
    return(o)
    })
  out = t(out)
  rownames(out) = colnames(downdata)[ind_cols]
  ##
  return(out)
  })

names(UnivarMat) = colnames(downdata)[dep_cols]
```

```
testmat = sapply(1:length(UnivarMat), function(x){ return( UnivarMat[[x]][,1] ) })
pmat = sapply(1:length(UnivarMat), function(x){ return( UnivarMat[[x]][,6] ) })
####
colnames(testmat) = names(UnivarMat)
corrplot(testmat,
         tl.col = "red",
         tl.cex = 0.95,
         tl.srt = 45, method = "cir",
         p.mat = pmat,
         insig = "label_sig",
         sig.level = c(.001, .01, .05),
         #sig.level = 0.05,
         pch.cex = 1.0,
         pch.col = "purple")
```



Multivariate model

```
## using wdata data frame as input
dep_cols = 24:30

MultivarMat = t( sapply(dep_cols, function(i){
    dep = rntransform( unlist( wdata[ ,i] ) )
    ############
    fit0 = lm( dep ~ `Site`+
                `Total DNA Concentration (ng/ul)` +
                `% Endogenous DNA` +
                #`Starting DNA  (ug)` +
                `Capture Pool` +
                `Sequencing Batch` +
                `Production Reads`
                , data = wdata)

    ################
```

```
    fit = lm( dep ~ `Production Reads` +
                `Sequencing Batch` +
                `Capture Pool` +
                #`Starting DNA  (ug)` +
                `% Endogenous DNA` +
                `Total DNA Concentration (ng/ul)` +
                `Site`, data = wdata)

    s = summary(fit)
    #####
    o = c(s$r.squared, s$adj.r.squared, s$fstatistic[1])
    names(o) = c("Rsq","Adj_Rsq","Fstat")
    #####
    a = anova(fit)
    n = gsub(" ","",rownames(a)); n = gsub("`","", n )
    eta = a[,2]/sum(a[,2]); names(eta) = paste0("etasq_", n )
    pval = a[, 5]; names(pval) = paste0("pval_", n )
    #####
    a = anova(fit0)
    n = gsub(" ","",rownames(a)); n = gsub("`","", n )
    eta0 = a[,2]/sum(a[,2]); names(eta0) = paste0("etasq_0_", n )
    pval0 = a[,5 ]; names(pval0) = paste0("pval_0_", n )
    ####
    o = c(o, eta0, pval0, eta, pval)
    ######################
    ## TYPE II ANOVA
    ####################
    a = Anova(fit, type = "II")
    eta_type2 = a[,1]/sum(a[,1], na.rm = TRUE)
    n = rownames(a); n = gsub(" ","", n); n = gsub("`","", n)
    names(eta_type2) = paste0("eta_type2_", n)

    pval_type2 = a[,4];
    names(pval_type2) = paste0("pval_type2", n)

    #####
    out = c(o, eta_type2, pval_type2)
    #####
    return(out)
    })
)
rownames(MultivarMat) = colnames(wdata)[dep_cols]
```

```
## using wdata data frame as input
dep_cols = 24:31

MultivarMat_DOWNSAM = t( sapply(dep_cols, function(i){
    dep = rntransform( unlist( downdata[ ,i] ) )
    ############
    fit0 = lm( dep ~ `Site`+
                `Total DNA Concentration (ng/ul)` +
                `% Endogenous DNA` +
                #`Starting DNA  (ug)` +
```

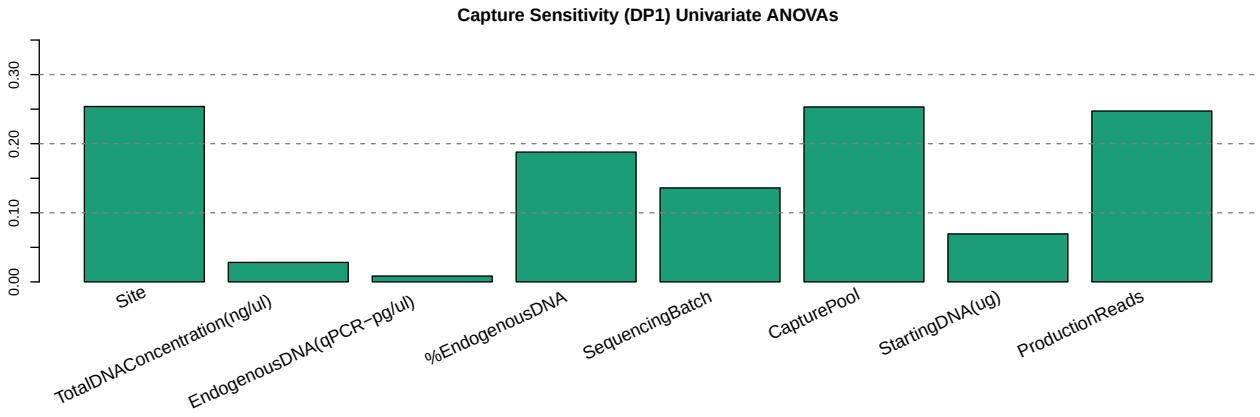
```
                `Capture Pool` +
                `Sequencing Batch` +
                `Production Reads`
                , data = downdata)

    ################

    fit = lm( dep ~ `Production Reads` +
                `Sequencing Batch` +
                `Capture Pool` +
                #`Starting DNA  (ug)` +
                `% Endogenous DNA` +
                `Total DNA Concentration (ng/ul)` +
                `Site`, data = downdata)

    s = summary(fit)
    #####
    o = c(s$r.squared, s$adj.r.squared, s$fstatistic[1])
    names(o) = c("Rsq","Adj_Rsq","Fstat")
    #####
    a = anova(fit)
    n = gsub(" ","",rownames(a)); n = gsub("`","", n )
    eta = a[,2]/sum(a[,2]); names(eta) = paste0("etasq_", n )
    pval = a[, 5]; names(pval) = paste0("pval_", n )
    #####
    a = anova(fit0)
    n = gsub(" ","",rownames(a)); n = gsub("`","", n )
    eta0 = a[,2]/sum(a[,2]); names(eta0) = paste0("etasq_0_", n )
    pval0 = a[,5 ]; names(pval0) = paste0("pval_0_", n )
    ####
    o = c(o, eta0, pval0, eta, pval)
    ######################
    ## TYPE II ANOVA
    ####################
    a = Anova(fit, type = "II")
    eta_type2 = a[,1]/sum(a[,1], na.rm = TRUE)
    n = rownames(a); n = gsub(" ","", n); n = gsub("`","", n)
    names(eta_type2) = paste0("eta_type2_", n)

    pval_type2 = a[,4];
    names(pval_type2) = paste0("pval_type2", n)

    #####
    out = c(o, eta_type2, pval_type2)
    #####
    return(out)
    })
)
rownames(MultivarMat_DOWNSAM) = colnames(downdata)[dep_cols]
```

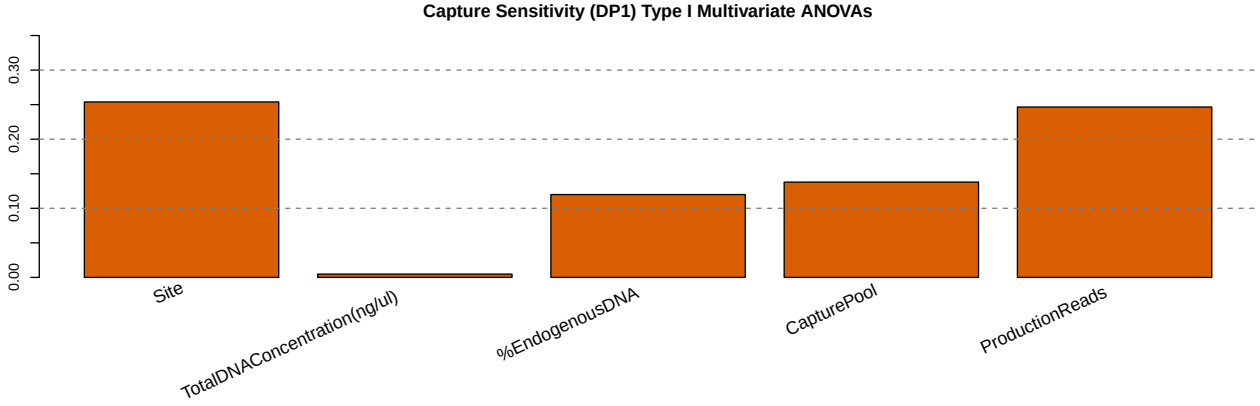
```
rotate_x <- function(data, column_to_plot, labels_vec, rot_angle, pcol = "steelblue", pmain = "") {
    plt <- barplot(data[[column_to_plot]], col=pcol, xaxt="n", ylim = c(0, 0.35), main = pmain)
    abline(h = seq(0.1, 0.3, by = 0.1), lty = 2, col = "grey50" )
    text(plt, par("usr")[3], labels = labels_vec, srt = rot_angle, adj = c(1.1,1.1), xpd = TRUE, cex=1.
```

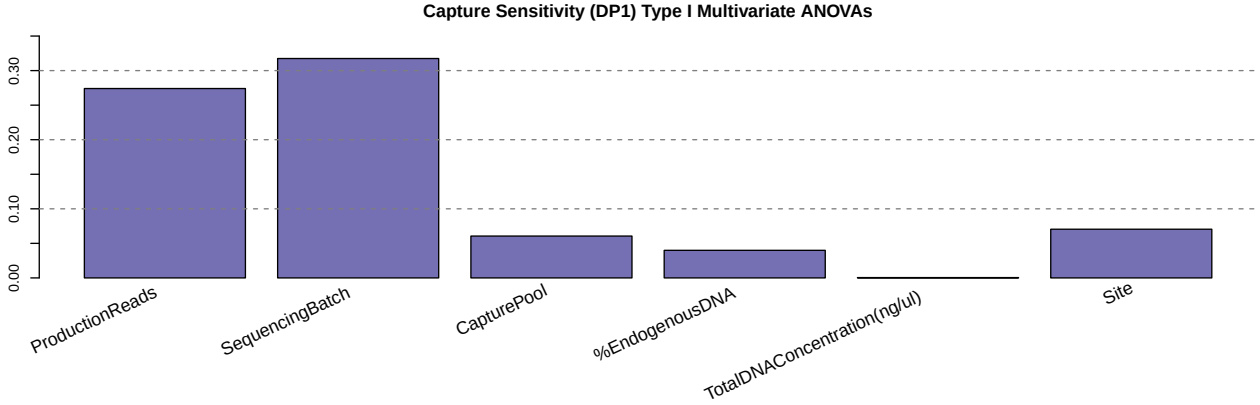
```
}
```

DownSampled Variance Explained

```
pcols = RColorBrewer::brewer.pal(3, "Dark2")

###
par(mfrow = c(3, 1), mar = c(8,3,3,1))
####
d = as.data.frame(UnivarMat$DP1[1:8, ])
rownames(d) = gsub(" ","",rownames(d))
rotate_x( d, 'Rsq', row.names(d), 25, pcol = pcols[1],
          pmain = "Capture Sensitivity (DP1) Univariate ANOVAs")

#####
d = data.frame( d = MultivarMat_DOWNSAM["DP1", c(4:8) ] )
rownames(d) = gsub("etasq_0_","",rownames(d))
rotate_x( d , 'd', row.names(d) , 25, pcol =  pcols[2],
          pmain = "Capture Sensitivity (DP1) Type I Multivariate ANOVAs")

#####
d = data.frame( d = MultivarMat_DOWNSAM["DP1", c(16:21) ] )
rownames(d) = gsub("etasq_","",rownames(d))
rotate_x( d , 'd', row.names(d) , 25, pcol =  pcols[3],
          pmain = "Capture Sensitivity (DP1) Type I Multivariate ANOVAs")
```



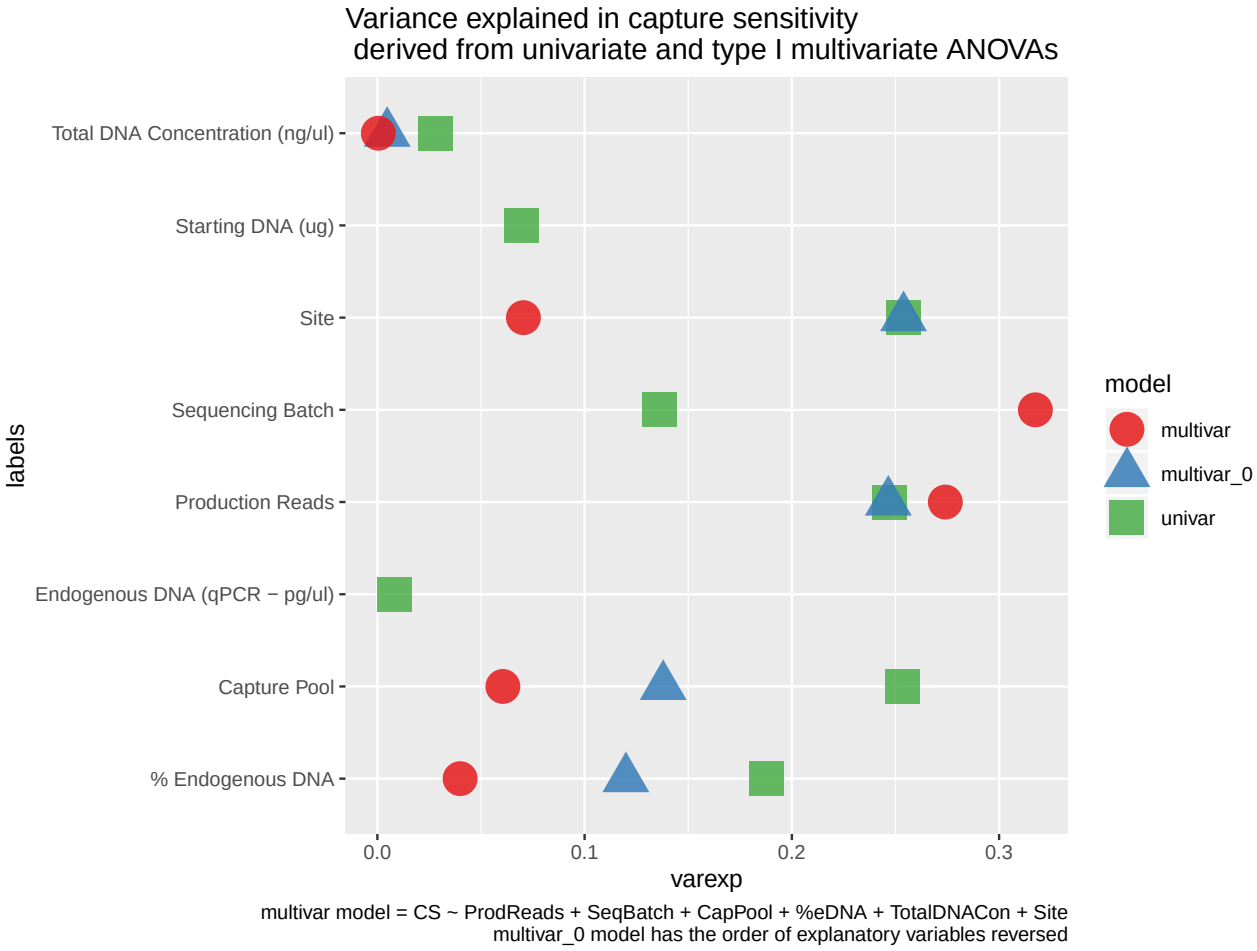





```
u = UnivarMat$DP1[1:8, 1]
m0 = MultivarMat_DOWNSAM["DP1", c(4:8) ]
m0 = c(m0[1:2], NA, m0[3], NA, m0[4], NA, m0[5])
m =  MultivarMat_DOWNSAM["DP1", c(16:21) ]
m = c(m[6:5], NA, m[4], m[2], m[3], NA, m[1])

varexp = c(u, m0, m)
n = names(u)
l = c(n , n, n)
mod = c( rep("univar", length(u))  , rep("multivar_0", length(u)) , rep("multivar", length(u)) )
###
d = data.frame( label = names( UnivarMat$DP1[1:8, 1]) ,
                  univar = UnivarMat$DP1[1:8, 1] ,
                  multi0 = c(m0[1:2], NA, m0[3], NA, m0[4], NA, m0[5]) ,
```

```
                    multi = c(m[6:5], NA, m[4], m[2], m[3], NA, m[1])
                    )

d = data.frame(labels = l, varexp = varexp, model = mod)

### turn data frame into tibble
d = as_tibble(d)

## plot
d %>% ggplot(aes(x = varexp, y = labels)) +
  geom_point( aes(color = model, shape = model), size = 7, alpha = 0.85) +
  scale_color_manual(values = RColorBrewer::brewer.pal(3, "Set1") ) +
  labs(title = paste0("Variance explained in capture sensitivity \n derived from univariate and type I m
       caption = paste0( "multivar model = CS ~ ProdReads + SeqBatch + CapPool + %eDNA + TotalDNACon + S
```

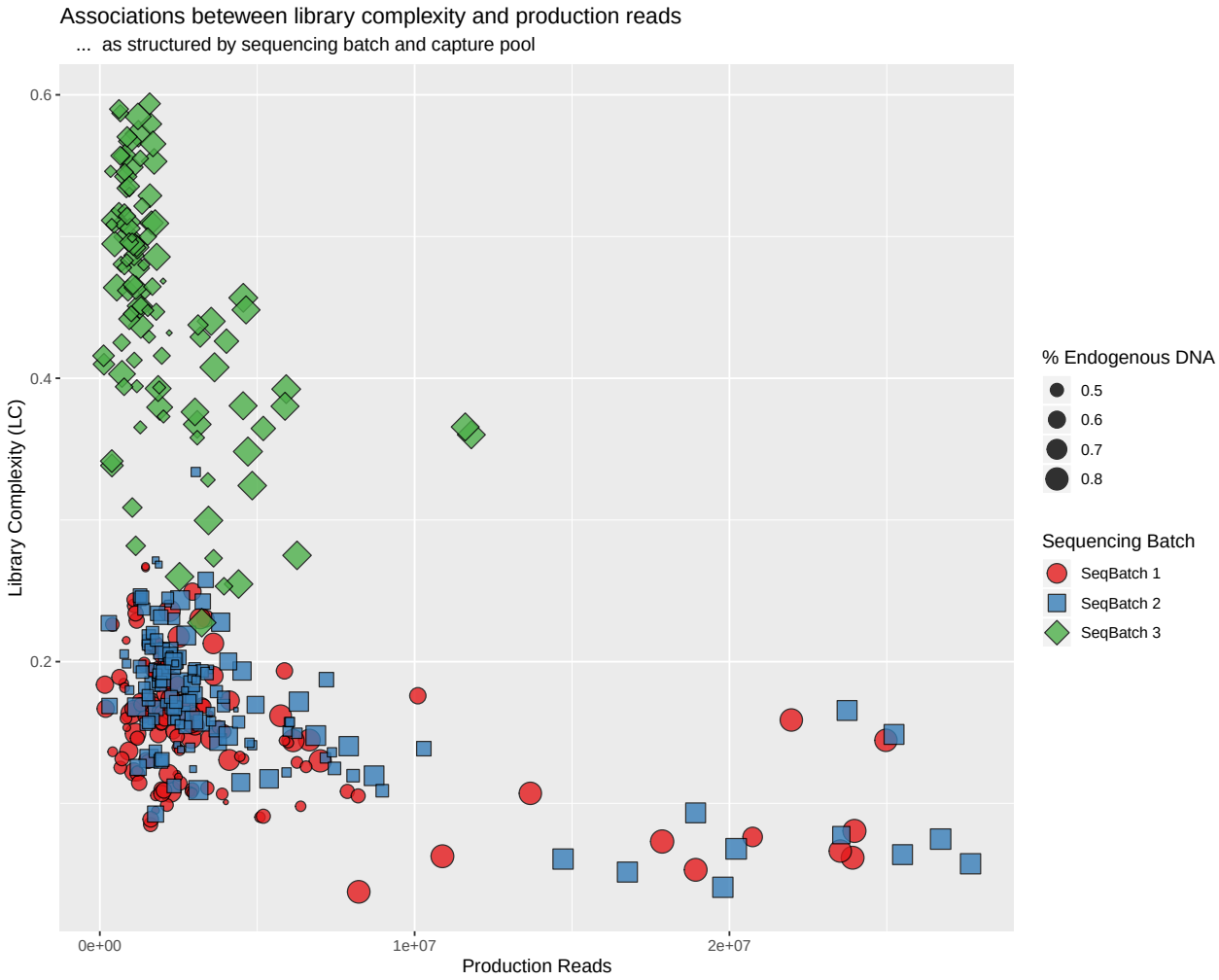
```
## Warning: Removed 5 rows containing missing values (geom_point).
```



How are production reads influencing library complexity ?

```
w = which(is.na(wdata$`Production Reads`))
pcol = brewer.pal(9, "Blues")[-1]
## a ramp of colors
```

```
pcol = colorRampPalette( pcol )(19)

#wdata %>% ggplot( aes(x = `Production Reads`, y = `Enrichment Factor (EF)`)) +
wdata[-w,] %>% ggplot( aes(x = `Production Reads`, y = `Library Complexity (LC)`)) +
  #geom_point(aes(fill = `Capture Pool`, shape = `Sequencing Batch`,  size = `% Endogenous DNA`), alpha
  geom_point(aes(fill = `Sequencing Batch`, shape = `Sequencing Batch`,  size = `% Endogenous DNA`), al
  scale_shape_manual(values=c(21, 22, 23)) +
  scale_fill_brewer(palette = "Set1") +
  #scale_fill_manual(values = pcol) +
  guides(fill = guide_legend(override.aes = list(size = 5) ) ) +
  labs(title = "Associations beteween library complexity and production reads",
       subtitle ="   ...  as structured by sequencing batch and capture pool")
```



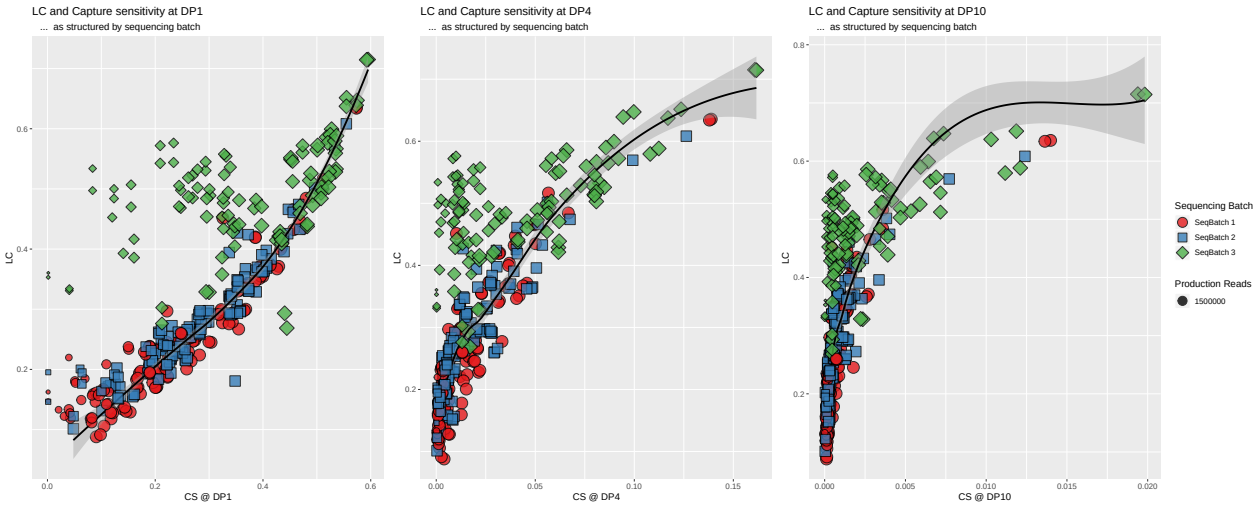
1. Samples in SeqBatch 3 all had 2ug of in the hybridization, hence the large bump in LC.
2. Drowning the data in useless sequencing also appears to have had a negative effect on LC.

What can we learn from these observations? 1. Increase the total DNA concentration in hybridization reactions (Perry paper did this) 2. Do not sequence to deeply.

How are production reads influencing library complexity in the down sampled data ?

```
w = which(downdata$`Production Reads` < 1500000)

pcol = brewer.pal(9, "Blues")[-1]
## a ramp of colors
pcol = colorRampPalette( pcol )(19)

################################
##  DP1
################################

p = downdata[-w,] %>% ggplot( aes(x = `DP1`, y = `LC`)) +
  geom_point(aes(fill = `Sequencing Batch`, shape = `Sequencing Batch`,  size = `Production Reads`), al
  geom_smooth(method = "loess", color = "black") +
  scale_shape_manual(values=c(21, 22, 23)) +
  scale_fill_brewer(palette = "Set1") +
  #scale_fill_manual(values = pcol) +
  guides(fill = guide_legend(override.aes = list(size = 5) ) ) +
  labs(title = "LC and Capture sensitivity at DP1",
       subtitle ="   ...  as structured by sequencing batch",
       x = "CS @ DP1")

##################



p1 = p + downdata[w,] %>%
  geom_point(mapping = aes(x = `DP1`, y = `LC`,
                           fill = `Sequencing Batch`,
                           shape = `Sequencing Batch`,
                           size = `Production Reads`), alpha = 0.8 ) +
  theme(legend.position = "none")

################################
##  DP4
################################
p = downdata[-w,] %>% ggplot( aes(x = `DP4`, y = `LC`)) +
  geom_point(aes(fill = `Sequencing Batch`, shape = `Sequencing Batch`,  size = `Production Reads`), al
  geom_smooth(method = "loess", color = "black") +
  scale_shape_manual(values=c(21, 22, 23)) +
  scale_fill_brewer(palette = "Set1") +
  #scale_fill_manual(values = pcol) +
  guides(fill = guide_legend(override.aes = list(size = 5) ) ) +
  labs(title = "LC and Capture sensitivity at DP4",
       subtitle ="   ...  as structured by sequencing batch",
       x = "CS @ DP4")

##################

p4 = p + downdata[w,] %>%
  geom_point(mapping = aes(x = `DP4`, y = `LC`,
```

```
                          fill = `Sequencing Batch`,
                          shape = `Sequencing Batch`,
                          size = `Production Reads`), alpha = 0.8 ) +
  theme(legend.position = "none")

################################
##  DP10
################################
p = downdata[-w,] %>% ggplot( aes(x = `DP10`, y = `LC`)) +
  geom_point(aes(fill = `Sequencing Batch`, shape = `Sequencing Batch`,  size = `Production Reads`), al
  geom_smooth(method = "loess", color = "black") +
  scale_shape_manual(values=c(21, 22, 23)) +
  scale_fill_brewer(palette = "Set1") +
  #scale_fill_manual(values = pcol) +
  guides(fill = guide_legend(override.aes = list(size = 5) ) ) +
  labs(title = "LC and Capture sensitivity at DP10",
       subtitle ="   ...  as structured by sequencing batch",
       x = "CS @ DP10")

##################

p10 = p + downdata[w,] %>%
  geom_point(mapping = aes(x = `DP10`, y = `LC`,
                          fill = `Sequencing Batch`,
                          shape = `Sequencing Batch`,
                          size = `Production Reads`), alpha = 0.8 )

l = cowplot::get_legend(p)

p10 = p10 + theme(legend.position = "none")


grid.arrange(p1, p4, p10, l, nrow = 1, widths = c(4,4,4,1))
```



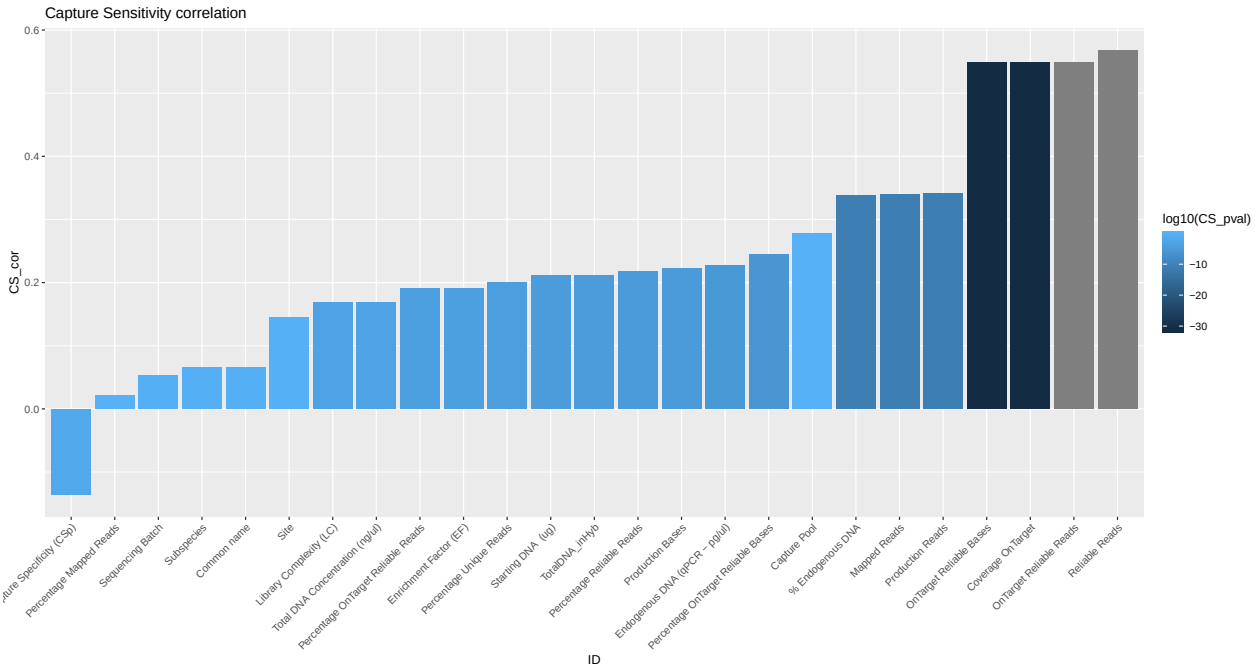
Capture Sensitivity

It would be useful to identify a single summary statistic that summarizes what a good "performing" target capture seqeuencing experiment is. I think what we be most useful is to count the number of (population-wide) variable position that we were able to genotype (at whatever criteria uniformly executed across all samples). In the absence of this data it would seem that the best summary statistic that would predict this number is Capture Sensitivity (# of target regions covered by 1 read / total number of target regions), as having a base covered by a read provides a chance for genotyping.

Now depth in coverage gives us accuracy in genotyping heterozygosity and there is most certainly going to be some bias in capturing specific alleles, but we have some good evidence that just making hemizygous calls is informative for the gross|macro level population genetics we would like to do. However, this should eventually be quantified.

Below we are asking in a univariate fashion how each of our variables, and sumstat, correlate with CAPTURE SENSITIVITY.

```
#x = sort( CorMat[[1]][-26,26] ); barplot(x)

df = tibble( ID = colnames( CorMat[[1]] ) ,
             CS_cor = unlist( CorMat[[1]][,"Capture Sensitivity (CS) DP1"]),
             CS_pval = unlist( CorMat[[2]][,"Capture Sensitivity (CS) DP1"]))

## change order to increasing values
o = order(df$CS_cor)
df = df[o,]
df = df[1:25, ]
### set ploting order with factor
df$ID <- factor(df$ID, levels = df$ID)

### plot
(
  p <- df %>% ggplot( aes( x = ID, y = CS_cor ) ) +
    geom_bar(stat="identity", aes(fill = log10(CS_pval) )) +
    labs(title = "Capture Sensitivity correlation") +
    theme(axis.text.x = element_text(angle = 45,  hjust = 1))
)
```

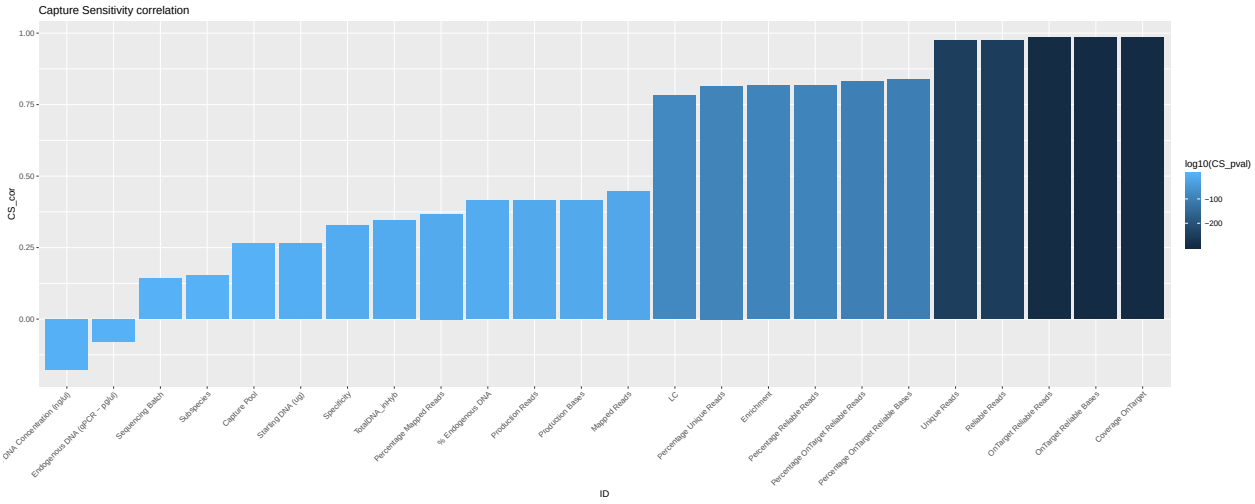


What can we learn from this analysis?

1. Want to increase capture sensitivity aquire more unique reads ! + kind of obvious but great to observe and demonstrate.
2. For technical or methodological choices it would appear that + samples with more Endogenous DNA, (note that this is NOT %DNA), it is higher DNA [concentrations] perform better + captures with more DNA in the hybridization perform better

Below we are asking in a univariate fashion how each of our variables, and sumstat, correlate with CAPTURE SENSITIVITY at a uniform production

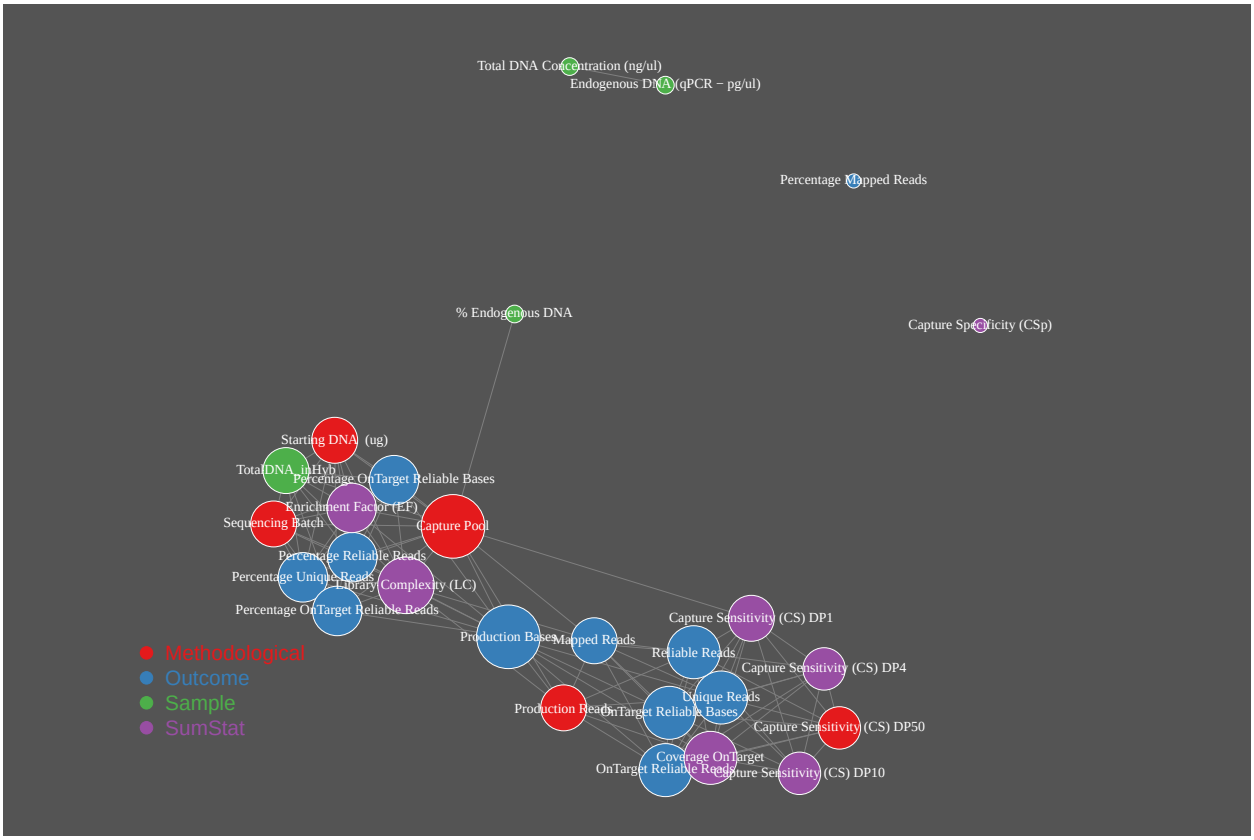
```
#x = sort( CorMat[[1]][-26,26] ); barplot(x)

df = tibble( ID = colnames( DownCorMat[[1]] ) ,
             CS_cor = unlist( DownCorMat[[1]][,"DP1"]),
             CS_pval = unlist( DownCorMat[[2]][,"DP1"]))
df = df[-c(1,2,4,27:31), ]

## change order to increasing values
o = order(df$CS_cor)
df = df[o,]
#df = df[1:25, ]
### set ploting order with factor
df$ID <- factor(df$ID, levels = df$ID)

### plot
(
  p <- df %>% ggplot( aes( x = ID, y = CS_cor ) ) +
    geom_bar(stat="identity", aes(fill = log10(CS_pval) )) +
    labs(title = "Capture Sensitivity correlation") +
    theme(axis.text.x = element_text(angle = 45,  hjust = 1))
)
```



Build a network of relationships based on correaltion estimates

```
#library(network)
library(igraph)

##############################
## Make Adjecency Matrix
##############################

# x = CorMat[[2]][-c(1:4), -c(1:4)]
# adjMat = x
# adjMat[ x > 0.00001] = 0
# adjMat[ x <= 0.00001] = 1
# diag(adjMat) = 0

x = CorMat[[1]][-c(1:4), -c(1:4)]
adjMat = abs( x )
adjMat[adjMat < 0.5] = 0

##############################
## Categorize the Nodes
##############################
n = colnames(adjMat)
nodecats = c( rep("Sample", 3), rep("Methodological", 4), rep("Outcome",11), rep("SumStat", 7 ), "Metho
pcol_o = brewer.pal(nlevels(as.factor(nodecats)), "Set1")
pcol <- pcol_o[as.numeric(as.factor(nodecats))]

##############################
## Generate network
##############################
network <- graph_from_adjacency_matrix(adjMat, weighted=T, mode="undirected", diag=F)

## estimate degree for each node
deg <- degree(network, mode="all")

##############################
```

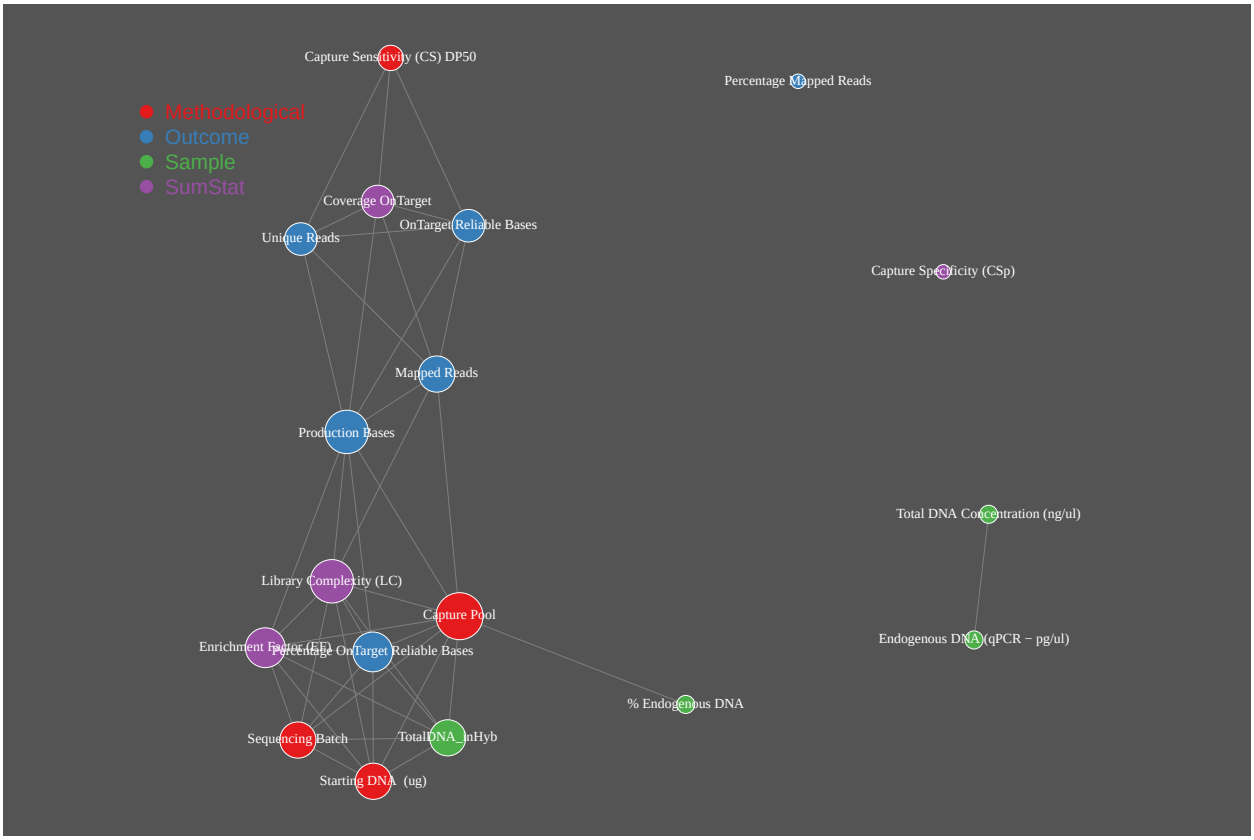
```
## Make Plot
#############################
par(bg="grey33", mar=c(0,0,0,0))
plot(network,
    #layout=layout.sphere,
    #layout=layout.circle,
    layout=layout.fruchterman.reingold,
    vertex.color = pcol,           # Node color
    vertex.label.color="white",
    vertex.frame.color = "white",    # Node border color
    vertex.shape= "circle",          # One of "none", "circle", "square", "csquare", "rectangle" "crecta
    vertex.size=deg+4,                   # Size of the node (default is 15)
    #vertex.size2=NA,
    edge.color = "grey50",
    #edge.arrow.size=0
    )
legend("bottomleft",
       legend=levels(as.factor(nodecats)),
       col = pcol_o,
       bty = "n",
       pch=20 ,
       pt.cex = 3,
       cex = 1.5,
       text.col=pcol_o,
       horiz = FALSE,
       inset = c(0.1, 0.1))
```



Remove redundancies in Network

```
##############################
## Data Reductions
##############################

### to do computationally -- LATER

##############################
## Make Adjecency Matrix
##############################

# x = CorMat[[2]][-c(1:4), -c(1:4)]
# adjMat = x
# adjMat[ x > 0.00001] = 0
# adjMat[ x <= 0.00001] = 1
# diag(adjMat) = 0

x = CorMat[[1]][-c(1:4,11, 16:19, 21, 27:29), -c(1:4,11, 16:19, 21, 27:29)]
adjMat = abs( x )
adjMat[adjMat < 0.5] = 0

##############################
## Categorize the Nodes
##############################
n = colnames(adjMat)
nodecats = c( rep("Sample", 3), rep("Methodological", 3), rep("Outcome",6), rep("SumStat", 4 ), "Methode
pcol_o = brewer.pal(nlevels(as.factor(nodecats)), "Set1")
pcol <- pcol_o[as.numeric(as.factor(nodecats))]

##############################
## Generate network
##############################
network <- graph_from_adjacency_matrix(adjMat, weighted=T, mode="undirected", diag=F)

## estimate degree for each node
deg <- degree(network, mode="all")

##############################
## Make Plot
##############################
par(bg="grey33", mar=c(0,0,0,0))
plot(network,
    #layout=layout.sphere,
    #layout=layout.circle,
    layout=layout.fruchterman.reingold,
    vertex.color = pcol,          # Node color
    vertex.label.color="white",
    vertex.frame.color = "white",   # Node border color
    vertex.shape= "circle",         # One of "none", "circle", "square", "csquare", "rectangle" "crecta
    vertex.size=deg+4,                  # Size of the node (default is 15)
    #vertex.size2=NA,
    edge.color = "grey50",
```

```
    #edge.arrow.size=0
    )
legend("topleft",
       legend=levels(as.factor(nodecats)),
       col = pcol_o,
       bty = "n",
       pch=20 ,
       pt.cex = 3,
       cex = 1.5,
       text.col=pcol_o,
       horiz = FALSE,
       inset = c(0.1, 0.1))
```



Remove redundancies in Network and construct with down sampled data

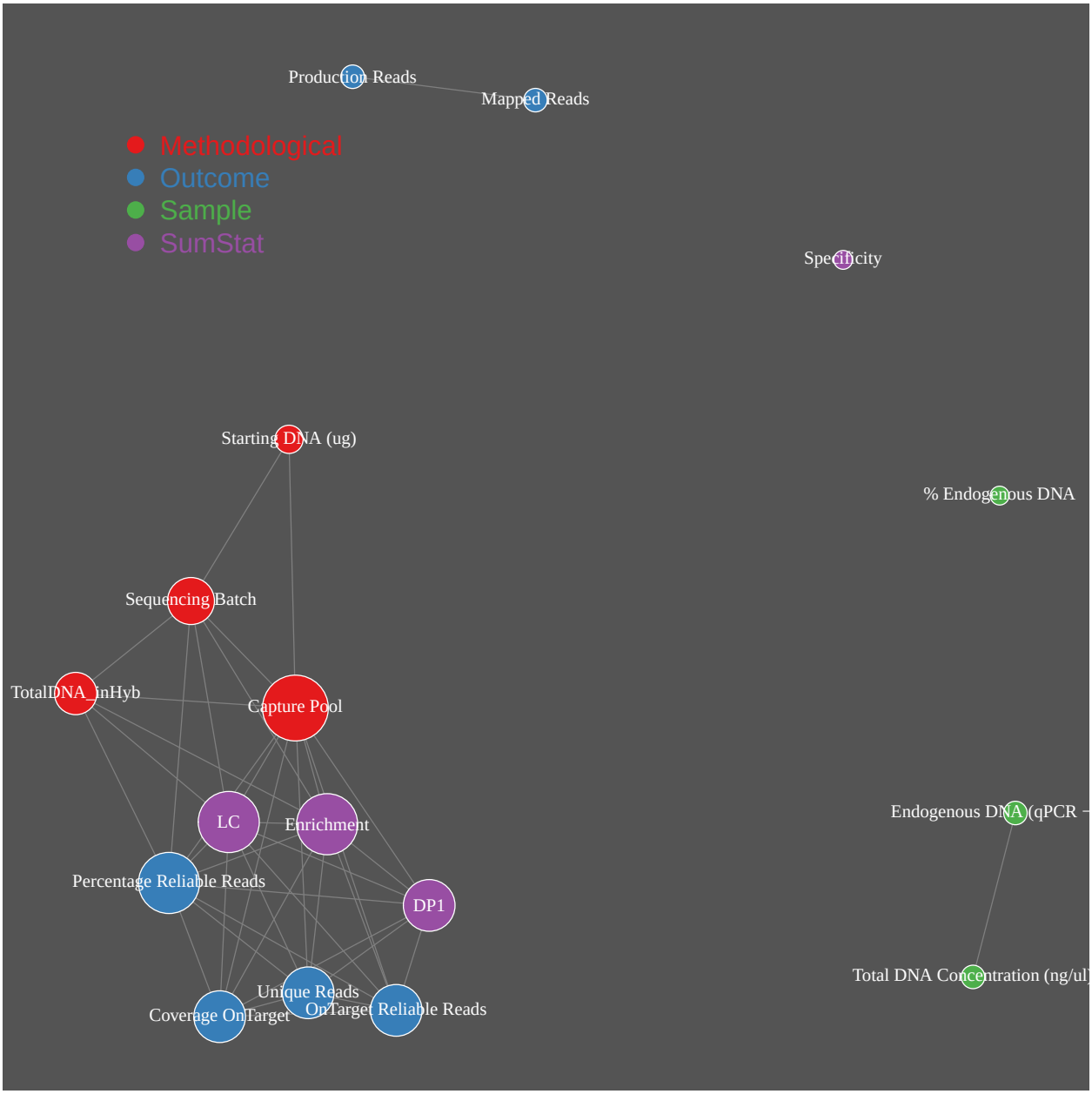
```
##############################
## Data Reductions
##############################

Dmat = as.dist( na.omit( 1 - abs(DownCorMat[[1]]) ) )
```

```
## Warning in as.dist.default(na.omit(1 - abs(DownCorMat[[1]]))): non-square
## matrix
```

```
tree = hclust(Dmat, method = "complete")
# plot(tree, hang = -1)
```

```
##############################
## Make Adjecency Matrix
##############################

# x = CorMat[[2]][-c(1:4), -c(1:4)]
# adjMat = x
# adjMat[ x > 0.00001] = 0
# adjMat[ x <= 0.00001] = 1
# diag(adjMat) = 0
r = c(1:4, 12, 14, 16, 17,20, 21, 22, 28:31)
x = DownCorMat[[1]][-r, -r]
adjMat = abs( x )
adjMat[adjMat < 0.5] = 0

##############################
## Categorize the Nodes
##############################
n = colnames(adjMat)
nodecats = c( rep("Sample", 3), rep("Methodological", 3), rep("Outcome",6), rep("SumStat", 4 ), "Methode
pcol_o = brewer.pal(nlevels(as.factor(nodecats)), "Set1")
pcol <- pcol_o[as.numeric(as.factor(nodecats))]

##############################
## Generate network
##############################
network <- graph_from_adjacency_matrix(adjMat, weighted=T, mode="undirected", diag=F)

## estimate degree for each node
deg <- degree(network, mode="all")

##############################
## Make Plot
##############################
par(bg="grey33", mar=c(0,0,0,0))
plot(network,
    #layout=layout.sphere,
    #layout=layout.circle,
    layout=layout.fruchterman.reingold,
    vertex.color = pcol,          # Node color
    vertex.label.color="white",
    vertex.frame.color = "white",   # Node border color
    vertex.shape= "circle",         # One of "none", "circle", "square", "csquare", "rectangle" "crecta
    vertex.size=deg+4,                  # Size of the node (default is 15)
    #vertex.size2=NA,
    edge.color = "grey50",
    #edge.arrow.size=0
    )
legend("topleft",
       legend=levels(as.factor(nodecats)),
       col = pcol_o,
       bty = "n",
       pch=20 ,
       pt.cex = 3,
```

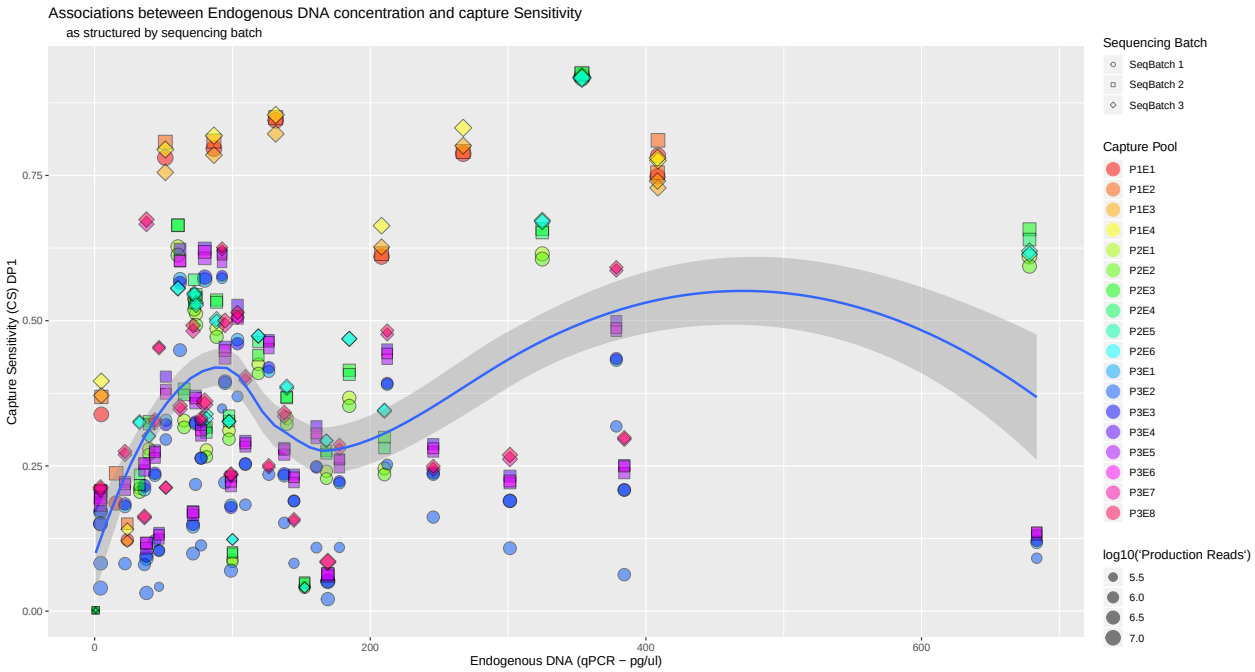
```
       cex = 1.5,
       text.col=pcol_o,
       horiz = FALSE,
       inset = c(0.1, 0.1))
```



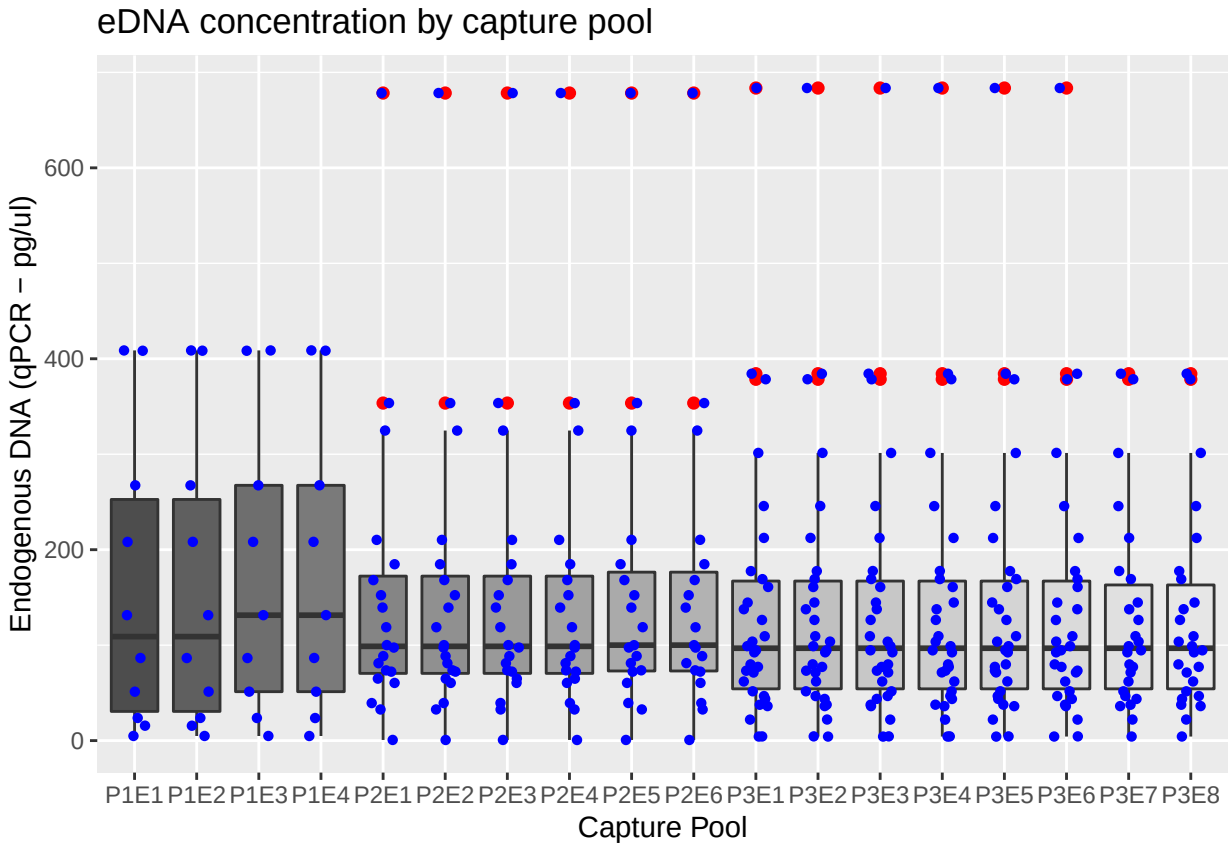
How is the concentration of a sample influencing capture sensitivity ?

```
w = which(is.na(wdata$`Production Reads`))
pcol = brewer.pal(9, "Blues")[-1]
## a ramp of colors
pcol = colorRampPalette( pcol )(19)
```

```
wdata[-w,] %>% mutate(NCS = `Capture Sensitivity (CS) DP1` / `Production Reads`) %>%  ggplot( aes(x = `
  #geom_point(aes(fill = `Sequencing Batch`, shape = `Sequencing Batch`,  size = `% Endogenous DNA`), a
  #geom_point(aes(fill = `Sequencing Batch`, shape = `Sequencing Batch`,  size = log10(`Production Read
  geom_point(aes(fill = `Capture Pool`, shape = `Sequencing Batch`,  size = log10(`Production Reads`)),
  geom_smooth(  method = "loess") +
  scale_shape_manual(values=c(21, 22, 23)) +
  #scale_fill_brewer(palette = "Set1") +
  scale_fill_manual(values = rainbow(nlevels(wdata$`Capture Pool`)) ) +
  guides(fill = guide_legend(override.aes = list(size = 5, color = rainbow(nlevels(wdata$`Capture Pool`
  labs(title = "Associations beteween Endogenous DNA concentration and capture Sensitivity",
       subtitle ="     as structured by sequencing batch")
```




```
wdata[-w,] %>% ggplot( aes(y = `Endogenous DNA (qPCR - pg/ul)`, x = `Capture Pool`)) +
  geom_boxplot(fill = gray.colors(nlevels(wdata$`Capture Pool`) ), outlier.colour="red", outlier.shape=
  geom_jitter(shape=16, position=position_jitter(0.2), color = "blue") +
  labs(title = "eDNA concentration by capture pool")
```

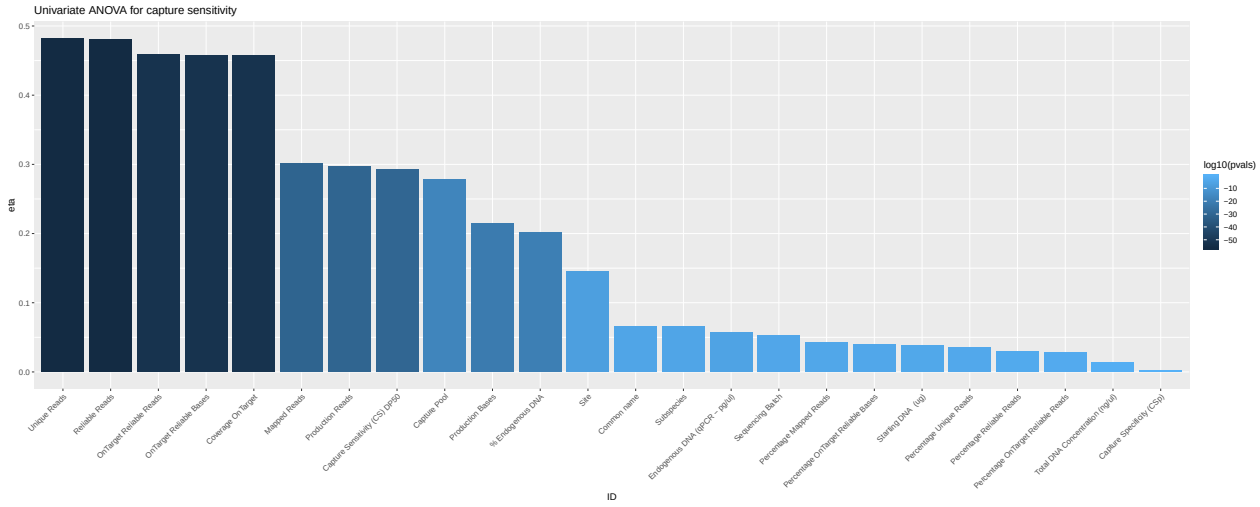


Univariate ANOVA on Summary Statistics

```
cols2test = c(2:3, 4:23, 25, 30)
UnivarateANOVA = matrix(NA, length(cols2test), 2)
for(i in 1:length(cols2test) ){
  x = unlist(wdata[,cols2test[i] ])
  test = class( x )
  ###

    fit = lm(wdata$`Capture Sensitivity (CS) DP1` ~  x)

  ###
  a = anova(fit)
  eta = a[1,2]/sum(a[,2])
  pval = a[1, 5]
  out = c(eta, pval)
  ##
  UnivarateANOVA[i, ] = out
  }

rownames(UnivarateANOVA) = colnames(wdata)[cols2test]
colnames(UnivarateANOVA) = c("eta","pval")

## order
o = order(UnivarateANOVA[,1], decreasing = TRUE)
```

```
UnivarateANOVA = UnivarateANOVA[o,]
```

```
df = tibble(ID = rownames(UnivarateANOVA), eta = UnivarateANOVA[,1], pvals = UnivarateANOVA[,2])
### maintain model order for the plot
df$ID <- factor(df$ID, levels = df$ID)

### plot
(
  p <- df %>% ggplot( aes( x = ID, y = eta ) ) +
    geom_bar(stat="identity", aes(fill = log10(pvals) )) +
    labs(title = "Univariate ANOVA for capture sensitivity")+
    theme(axis.text.x = element_text(angle = 45,  hjust = 1))
)
```



A multivariate model to explain how sample quality, and methodological choice influences Capture Sensitivity

```
library(car)
##############################
## fit a simple linear model
##############################
fit = lm( `Capture Sensitivity (CS) DP1` ~ `Total DNA Concentration (ng/ul)` +
            `Endogenous DNA (qPCR - pg/ul)` +
            `% Endogenous DNA` +
            `Capture Pool` +
            `TotalDNA_inHyb` +
            `Sequencing Batch` +
            `Production Reads` +
            `Unique Reads`
            , data = wdata )

fit = lm( `Capture Sensitivity (CS) DP1` ~ `Total DNA Concentration (ng/ul)` +
            `Endogenous DNA (qPCR - pg/ul)` +
            `% Endogenous DNA` +
            `TotalDNA_inHyb` +
            `Production Reads` +
```

```
            `Unique Reads`
            , data = wdata )


##############################
## are model residuals normal ?
##############################
W = shapiro.test(residuals(fit))

##############################
##  estiamte SS and VarExp
##  assuming an TypeI hierarchical
##  ANOVA
##############################

(a = anova(fit) )
```

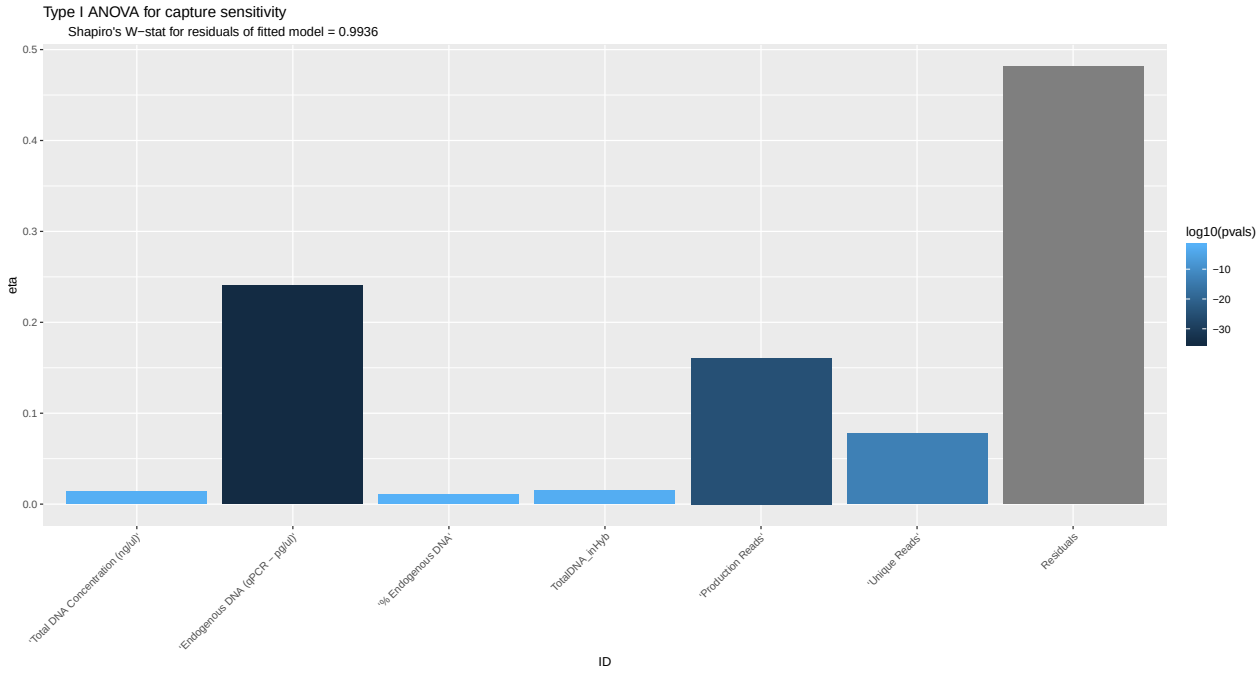
```
## Analysis of Variance Table
##
## Response: Capture Sensitivity (CS) DP1
##                                    Df Sum Sq Mean Sq  F value    Pr(>F)
## `Total DNA Concentration (ng/ul)`   1 0.2370  0.2370  10.7605 0.0011324
## `Endogenous DNA (qPCR - pg/ul)`     1 4.1996  4.1996 190.6785 < 2.2e-16
## `% Endogenous DNA`                  1 0.1849  0.1849   8.3952 0.0039795
## TotalDNA_inHyb                      1 0.2634  0.2634  11.9582 0.0006055
## `Production Reads`                  1 2.8046  2.8046 127.3397 < 2.2e-16
## `Unique Reads`                      1 1.3593  1.3593  61.7162 4.089e-14
## Residuals                         381 8.3914  0.0220
##
## `Total DNA Concentration (ng/ul)` **
## `Endogenous DNA (qPCR - pg/ul)`   ***
## `% Endogenous DNA`                **
## TotalDNA_inHyb                    ***
## `Production Reads`                ***
## `Unique Reads`                    ***
## Residuals
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
eta = a[, 2]/ sum(a[,2])
names(eta) = rownames(a)


summary(fit)
```

```
##
## Call:
## lm(formula = `Capture Sensitivity (CS) DP1` ~ `Total DNA Concentration (ng/ul)` +
##     `Endogenous DNA (qPCR - pg/ul)` + `% Endogenous DNA` + TotalDNA_inHyb +
##     `Production Reads` + `Unique Reads`, data = wdata)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.38579 -0.11332 -0.00564  0.11084  0.41269
##
```

```
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     9.648e-03  5.903e-02   0.163  0.87026
## `Total DNA Concentration (ng/ul)` 3.783e-04  2.125e-03   0.178  0.85878
## `Endogenous DNA (qPCR - pg/ul)`   3.103e-05  3.760e-04   0.083  0.93427
## `% Endogenous DNA`                2.003e-01  9.798e-02   2.045  0.04158
## TotalDNA_inHyb                    5.956e-02  2.021e-02   2.946  0.00341
## `Production Reads`                7.901e-09  2.693e-09   2.934  0.00355
## `Unique Reads`                    2.037e-07  2.593e-08   7.856 4.09e-14
##
## (Intercept)
## `Total DNA Concentration (ng/ul)`
## `Endogenous DNA (qPCR - pg/ul)`
## `% Endogenous DNA`                *
## TotalDNA_inHyb                    **
## `Production Reads`                **
## `Unique Reads`                    ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1484 on 381 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.5188, Adjusted R-squared:  0.5113
## F-statistic: 68.47 on 6 and 381 DF,  p-value: < 2.2e-16
```

```
df = tibble(ID = names(eta) , eta = eta, pvals = a[, 5])
### maintain model order for the plot
df$ID <- factor(df$ID, levels = df$ID)

### plot
(
  p <- df %>% ggplot( aes( x = ID, y = eta ) ) +
    geom_bar(stat="identity", aes(fill = log10(pvals) )) +
    labs(title = "Type I ANOVA for capture sensitivity",
         subtitle = paste0( "       Shapiro's W-stat for residuals of fitted model = ", signif(W$statis
    theme(axis.text.x = element_text(angle = 45,  hjust = 1))
)
```


Type I ANOVA for capture sensitivity
Shapiro's W−stat for residuals of fitted model = 0.9936
eta
0.5
0.4
0.3
0.2
0.1
0.0
log10(pvals)
−10
−20
−30
'Total DNA Concentration (ng/ul)'
'Endogenous DNA (qPCR − pg/ul)'
'% Endogenous DNA'
TotalDNA_inHyb
'Production Reads'
'Unique Reads'
Residuals
ID