Projekt Zaliczeniowy AT

Mateusz Janusz

13 02 2021

Bash

Pobranie niezbednych plikow dla BioProject 313294 za pomoca skryptu Entrez esearch, oraz stworzenie pliku run.txt z numerami SRR w kolejnych linijkach

```
#esearch -db sra -query PRJNA313294 | efetch -format runinfo -mode xml | xtract
#-pattern SraRunInfo -element Run > runinfo.txt
#cat runinfo.txt | tr '\t' '\n' >run.txt
```

Wzór kodu do sciagania odczytow, nie korzystano ze skryptu ze wzgledu na ograniczona pojemnosc dysku komputera, na ktorym zainstalowano wirtualna maszyne i przeprowadzano analize

```
#fastq-dump <numer SRR> --split-files
```

Skrypt do obrobki programem Trimmomatic zarowna dla Paired Ends jak i Single Ends

```
#java -jar /bioapp/Trimmomatic-0.39/trimmomatic-0.39.jar PE
#<2 inputs and 4 outputs> LEADING:6 TRAILING:30 SLIDINGWINDOW:4:30
```

```
#java -jar /bioapp/Trimmomatic-0.39/trimmomatic-0.39.jar SE <1 input 1 output> # LEADING:30 TRAILING:30
```

Stworzenie genomu referencyjnego- pojedyncze chromosomy pobrano ze strony http://hgdownload.soe.ucsc.edu/goldenPath/hg19/chromosomes/ , nastepnie rozpakowano je i sklejono w jeden plik za pomoca skryptu. Pobrano rowniez plik gtf niezbedny do featureCounts

```
#gunzip <file>.fa.gz
#head -q -n-0 *fa > Genoms.fa
```

Mapowanie przeprowadzono za pomoca programu Hisat2. Wzor uzytego skryptu:

```
#hisat2 [options]* -x <hisat2-idx> {-1 <m1> -2 <m2> | -U <r> | --sra-acc <SRA #accession number>} [-S <
```

Nastepnie przeksztalcono pliki SAM do BAM, a nastepnie posortowano pliki BAM

```
#samtools view -Sb -@ 2 *.sam > *.bam
#samtools sort *bam -o *sorted.bam
```

Na koniec stworzono pliki counts2Zika.txt za pomoca programu FeatureCounts. Plik ten uzyto w analizie danych w R

```
#featureCounts -a hg19.gtf -o counts2Zika.txt *.bam
```

R analiza

```
library(ggfortify)
```

```
## Loading required package: ggplot2
```

```
library(DESeq2)
```

```
## Loading required package: S4Vectors

## Loading required package: stats4

## Loading required package: BiocGenerics

## Loading required package: parallel

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:base':
##
##     expand.grid

## Loading required package: IRanges

##
## Attaching package: 'IRanges'

## The following object is masked from 'package:grDevices':
##
##     windows

## Loading required package: GenomicRanges

## Loading required package: GenomeInfoDb

## Loading required package: SummarizedExperiment

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

##
## Attaching package: 'MatrixGenerics'

## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars
```

```
## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

##
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians

## The following objects are masked from 'package:matrixStats':
##
##     anyMissing, rowMedians
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0
## v purrr   0.3.4

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::collapse()        masks IRanges::collapse()
## x dplyr::combine()         masks Biobase::combine(), BiocGenerics::combine()
## x dplyr::count()           masks matrixStats::count()
## x dplyr::desc()            masks IRanges::desc()
## x tidyr::expand()          masks S4Vectors::expand()
## x dplyr::filter()          masks stats::filter()
## x dplyr::first()           masks S4Vectors::first()
## x dplyr::lag()             masks stats::lag()
## x BiocGenerics::Position() masks ggplot2::Position(), base::Position()
## x purrr::reduce()          masks GenomicRanges::reduce(), IRanges::reduce()
## x dplyr::rename()          masks S4Vectors::rename()
## x dplyr::slice()           masks IRanges::slice()
```

```
library(dplyr)
library(RColorBrewer)
library(heatmap.plus)
library(ggplot2)
library(stringr)
```

Analiza DESeq

```
data = read.delim('counts2Zika.txt', comment= '#', stringsAsFactors = F)

countdata = data[,7:14]
rownames(countdata)=data$Geneid
colnames(countdata)=c('SRR3194428','SRR3194429','SRR3194430','SRR3194431','SRR3191542','SRR3191543','SRR
countdata=select(countdata,c(1,5,2,6,3,7,4,8))

print(dim(countdata))
```

```
## [1] 57820     8
```

```
keep <- rowSums(countdata) > 5
countdata <- countdata[keep,]
print(dim(countdata))
```

```
## [1] 31260     8
```

Pominiecie genow, dla ktorych ekspresja jest mniejsza niz 5

Porównanie wyników wzgledem uzytych urzadzen

```
samples=names(countdata)
cond_1=rep("Mock",4)
cond_2=rep("ZIKAV",4)
condition=factor(c(cond_1,cond_2))
colData=data.frame(samples=samples,
Instrument=factor(rep(c("Illumina MiSeq","NextSeq 500"),4)),
Condition=condition)
dds=DESeqDataSetFromMatrix(countData=countdata,
colData=colData,
design=~Instrument)
```

```
##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]
```

```
log_data <- rlog(dds)
```

```
##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]
```

```
norm_data<-assay(log_data)
norm_data <- as.data.frame(norm_data)
dds=DESeq(dds)
```

```
## estimating size factors
##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]

## estimating dispersions

## gene-wise dispersion estimates

## mean-dispersion relationship

##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]

## final dispersion estimates

##   Note: levels of factors in the design contain characters other than
##   letters, numbers, '_' and '.'. It is recommended (but not required) to use
##   only letters, numbers, and delimiters '_' or '.', as these are safe characters
##   for column names in R. [This is a message, not a warning or an error]

## fitting model and testing
```

```
res=results(dds)

head(na.omit(res[order(res$pvalue, decreasing = F),]))
```

```
## log2 fold change (MLE): Instrument NextSeq.500 vs Illumina.MiSeq
## Wald test p-value: Instrument NextSeq.500 vs Illumina.MiSeq
## DataFrame with 6 rows and 6 columns
##                     baseMean log2FoldChange     lfcSE      stat       pvalue
##                    <numeric>      <numeric> <numeric> <numeric>    <numeric>
## ENSG00000178464.6   620.786        3.02918 0.0988535   30.6431 3.26199e-206
## ENSG00000220842.5   288.206        4.57582 0.1639225   27.9145 1.77848e-171
## ENSG00000183298.4   254.620        4.78786 0.1992573   24.0285 1.39968e-127
## ENSG00000230291.4   307.960       -4.22470 0.2149832  -19.6513  5.63397e-86
## ENSG00000225093.1   304.726       -3.26870 0.1694946  -19.2850  7.18356e-83
## ENSG00000256148.1   242.033       -3.78354 0.2145670  -17.6334  1.36545e-69
##                           padj
##                      <numeric>
## ENSG00000178464.6 8.61329e-202
## ENSG00000220842.5 2.34804e-167
## ENSG00000183298.4 1.23195e-123
## ENSG00000230291.4  3.71912e-82
## ENSG00000225093.1  3.79364e-79
## ENSG00000256148.1  6.00911e-66
```

Obserwujac p-value testu Walda dla porownania sposobow sekwencjonowania widzimy, ze nie ma miedzy nimi znaczacej roznicy.

Korelacja Pearsona

```
wynik1 = cor.test(countdata[,1],countdata[,2])
wynik2 = cor.test(countdata[,3],countdata[,4])
wynik3 = cor.test(countdata[,5],countdata[,6])
wynik4 = cor.test(countdata[,7],countdata[,8])

print(wynik1)
```

```
##
##  Pearson's product-moment correlation
##
## data:  countdata[, 1] and countdata[, 2]
## t = 899.94, df = 31258, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9808273 0.9816513
## sample estimates:
##       cor
## 0.9812438
```

```
print(wynik2)
```

```
##
##  Pearson's product-moment correlation
##
## data:  countdata[, 3] and countdata[, 4]
## t = 946.85, df = 31258, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9826326 0.9833797
## sample estimates:
##       cor
## 0.9830102
```

```
print(wynik3)
```

```
##
##  Pearson's product-moment correlation
##
## data:  countdata[, 5] and countdata[, 6]
## t = 883.12, df = 31258, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9801115 0.9809660
## sample estimates:
##       cor
## 0.9805434
```

```
print(wynik4)
```

```
##
##  Pearson's product-moment correlation
##
## data:  countdata[, 7] and countdata[, 8]
## t = 680.21, df = 31258, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.9671321 0.9685354
## sample estimates:
##       cor
## 0.9678413
```

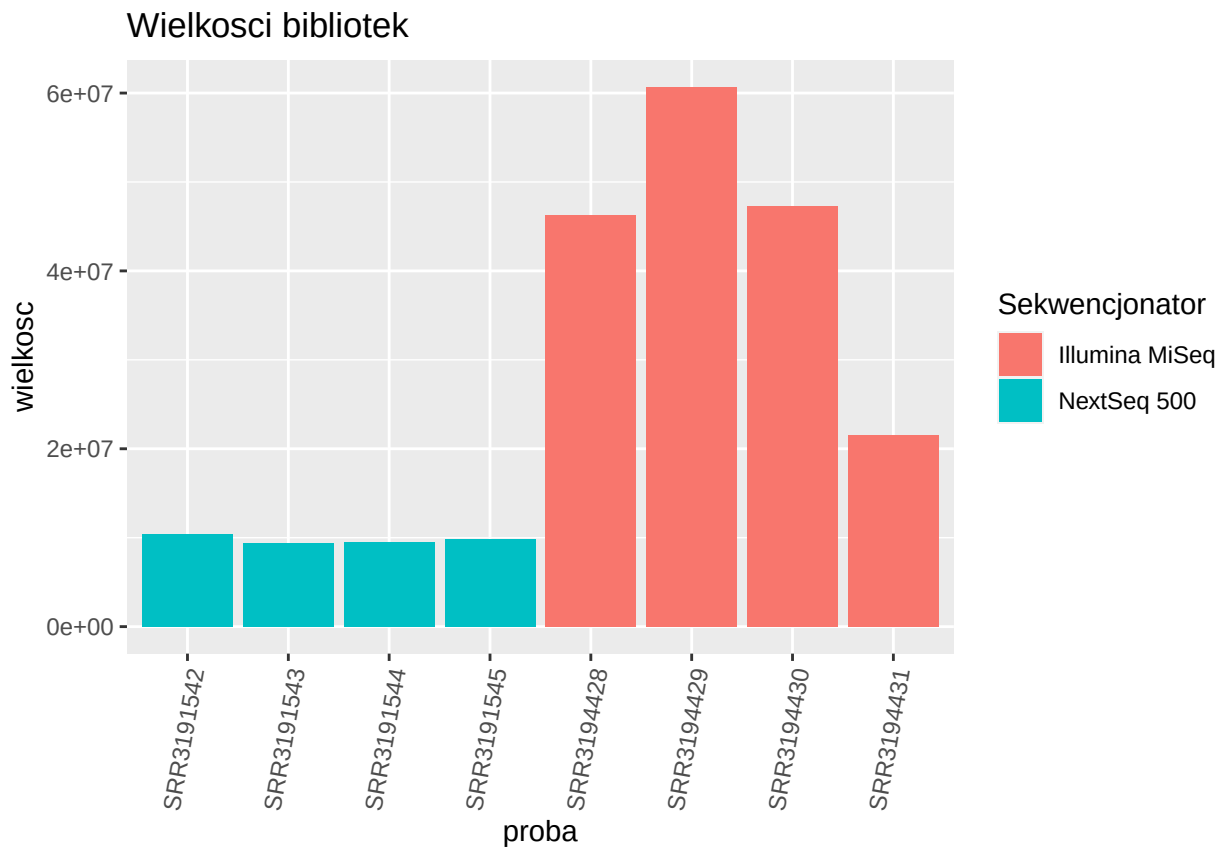
Jak widzimy, wszystkie korelacje sa zblizone do 1, co mowi nam, ze dane sa silnie dodatnio skorelowane.

Porównanie wielkosci bibliotek

```
librarySizes <- colSums(countdata)
bySize=data_frame(wielkosc=librarySizes,
proba=names(librarySizes))
```

```
## Warning: `data_frame()` is deprecated as of tibble 1.1.0.
## Please use `tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_warnings()` to see where this warning was generated.
```

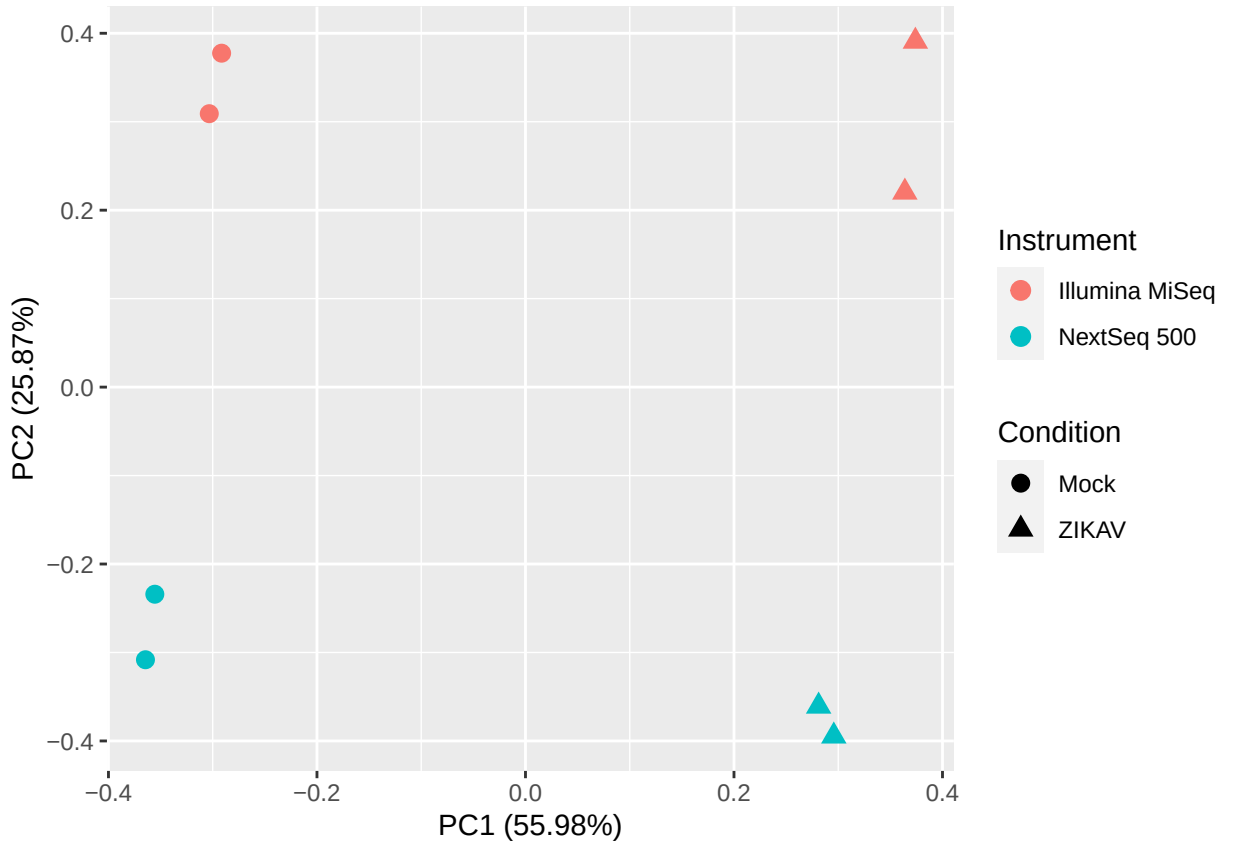
```
Sekwencjonator=colData$Instrument
ggplot(bySize,aes(x=proba,
y=wielkosc,
fill=Sekwencjonator))+
  geom_bar(stat="identity")+
theme(axis.text.x = element_text(angle = 80, hjust = 1))+
ggtitle(label = "Wielkosci bibliotek")
```



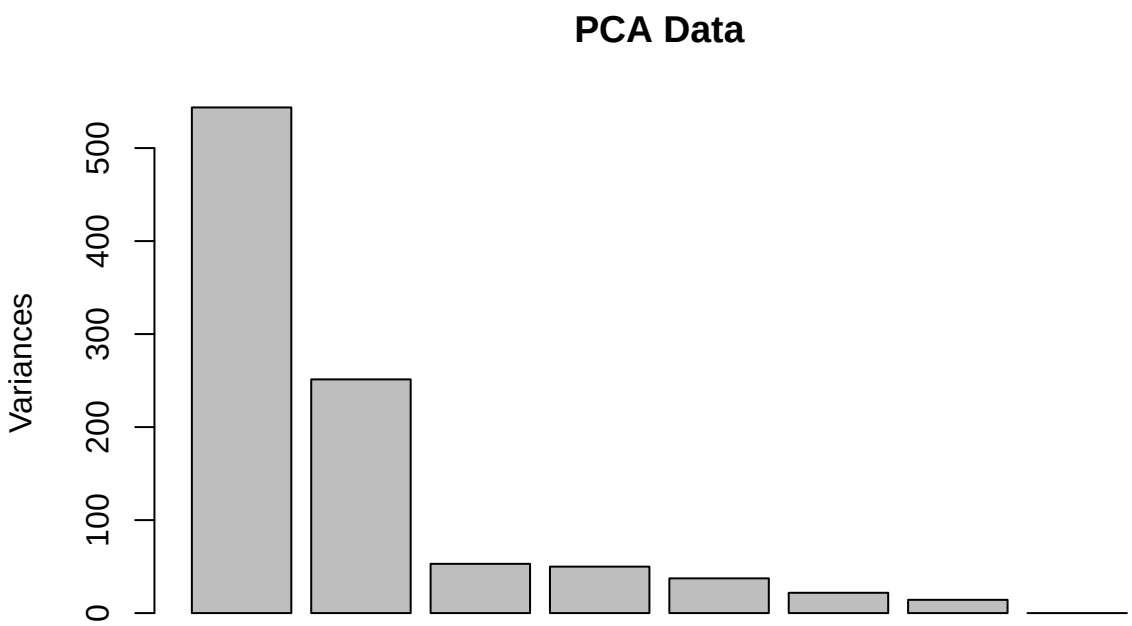
Analiza wielkosci bibliotek mowi o wiekszych bibliotekach uzywanych przez Illumina MiSeq, niz NextSeq500

Analiza PCA- analiza głównych skladowych

```
pcDat <- prcomp(t(norm_data))
autoplot(pcDat,
data = colData,
colour = "Instrument",
shape = "Condition",
size = 3)
```




```
screeplot(pcDat,main = "PCA Data")
```



Analiza PCA pokazuje, że najwieksza wariancja wystepuje miedzy rodzajami komorek (ZIKAV a komórki kontrolne). W dolnym wykresie Screen Plot widzimy wiekszosc wariancji wystepujaca w pierwszej kolumnie.

Stworzenie heatmapy 500 najmocniej ekspresjonowanych genów

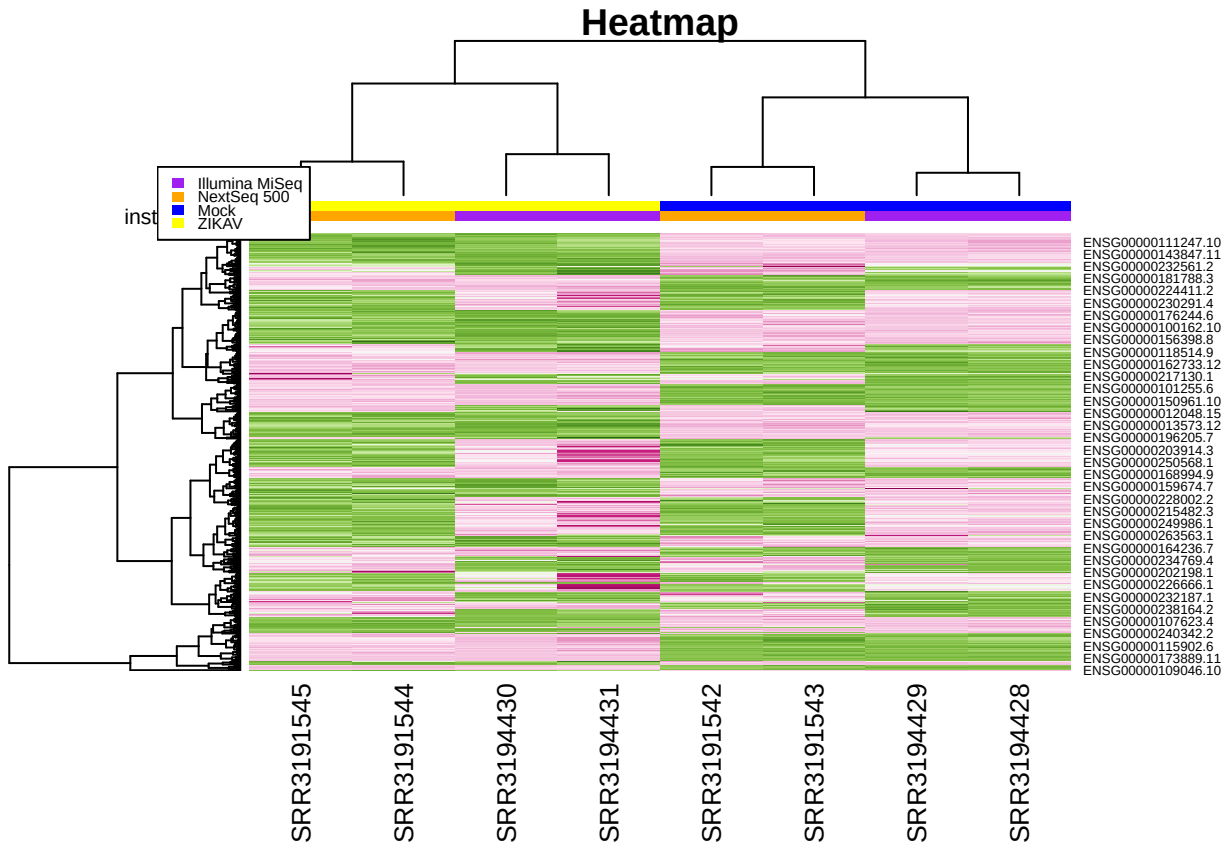
```
countVar <- apply(norm_data, 1, var)
highVar <- order(countVar, decreasing=TRUE)[1:500]
hmDat <- norm_data[highVar,]
mypalette <- brewer.pal(11, "PiYG")
morecols <- colorRampPalette(mypalette)
instrument <- c("purple","orange")[colData$Instrument]
treatment <- c("blue","yellow")[colData$Condition]
heatmap.plus(as.matrix(hmDat),
col=rev(morecols(50)),
trace="column",
main="Heatmap",
ColSideColors=cbind(instrument,treatment),
scale="row",
margins = c(8,7))
```

```
## Warning in plot.window(...): 'trace' nie jest parametrem graficznym

## Warning in plot.xy(xy, type, ...): 'trace' nie jest parametrem graficznym

## Warning in title(...): 'trace' nie jest parametrem graficznym
```

```
legend(0,1,
legend = c("Illumina MiSeq","NextSeq 500","Mock","ZIKAV"),
fill=c("purple","orange","blue","yellow"),
border=F, y.intersp = 0.7, cex=0.5,bg = "white")
```



Heatmapa wskazuje, iz odczyty sa zblizone, jak rowniez na istnienie niewielkich roznic miedzy intensywnoscia eksrepsji pojedynczych genow.