# R implementation

Sleiman Bassim, PhD

June 28, 2016

Loaded functions:

```
#source("/media/Data/Dropbox/humanR/01funcs.R")
rm(list=ls())
#setwd("/media/Data/Dropbox/humanR/PD/")
#setwd("~/Dropbox/humanR/PD/")
###load("PD.Rdata", .GlobalEnv)
#lsos(pat="")
```

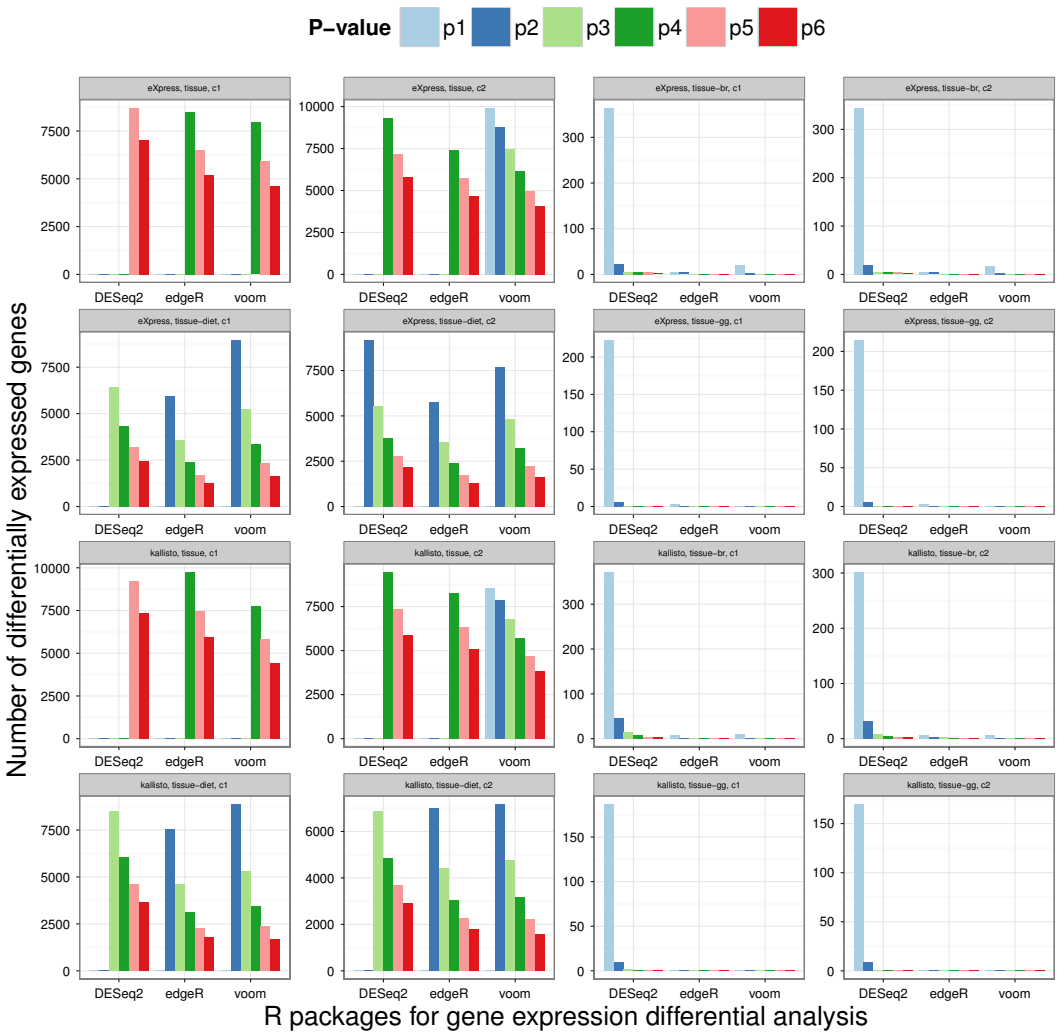
Load packages.

```
pkgs <- c('xlsx','caret','leaps','glmnet','lattice',
          'latticeExtra', 'dplyr', 'tidyr', 'grid')
lapply(pkgs, require, character.only = TRUE)
```

## 1 Differentially expressed genes

Differentially expressed genes are counted from mapping **both gills and ganglia** sequenced samples to reference transcriptome built from all samples.

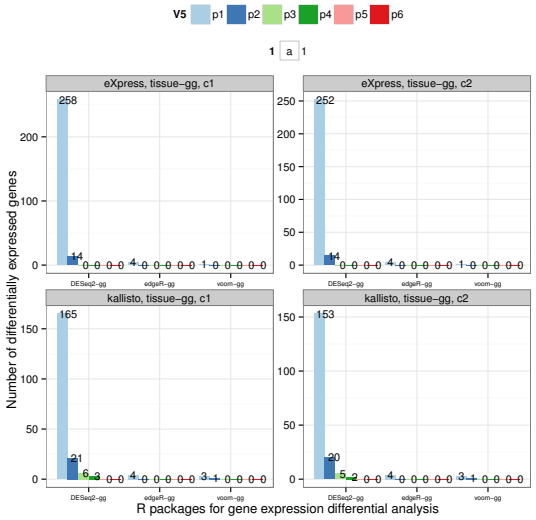
```
read.table("./data/summary.raw.all.txt") %>%
    ggplot(aes(
        x = V1,
        y = V8,
        fill = V6)) +
    theme_bw() +
    geom_bar(stat = "identity",
             position = "dodge") +
    facet_wrap(~ V4 + V5 + V7,
               ncol = 4,
               scales = "free") +
    scale_fill_brewer(type = "qual", palette = "Paired",
                      name = "P-value") +
    labs(x = "R packages for gene expression differential analysis",
         y = "Number of differentially expressed genes") +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5, size = 6),
          axis.text.y = element_text(vjust = .5, size = 6),
          strip.text = element_text(size = 4),
          axis.ticks = element_line(size = .2),
          axis.ticks.length = unit(.05, "cm"))
```




Differentially expressed genes are counted from mapping **ganglia** sequenced samples to reference transcriptome built from all samples.

```
read.table("./data/summary.gg.txt") %>%
```

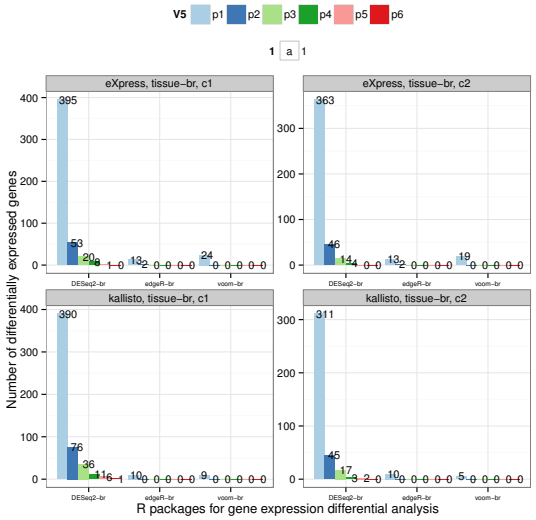
```
    ggplot(aes(
    x = V2,
    y = V8,
    fill = V5)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = V2,
                  y = V8,
                  ymax = V8,
                  label = V8,
                  size = 1,
                  hjust = 0),
              position = position_dodge(width = 1)) +
    facet_wrap(~ V3 + V4 + V6,
               ncol = 2,
               scales = "free") +
    scale_fill_brewer(type = "qual", palette = "Paired") +
    theme_bw() +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5,
                                     size = 6)) +
    labs(x = "R packages for gene expression differential analysis",
         y = "Number of differentially expressed genes")
```




Differentially expressed genes are counted from mapping **gills** sequenced samples to reference transcriptome built from all samples.

```
read.table("./data/summary.br.txt") %>%
```

```
    ggplot(aes(
    x = V2,
    y = V8,
    fill = V5)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = V2,
                  y = V8,
                  ymax = V8,
                  label = V8,
                  size = 1,
                  hjust = 0),
              position = position_dodge(width = 1)) +
    facet_wrap(~ V3 + V4 + V6,
               ncol = 2,
               scales = "free") +
    scale_fill_brewer(type = "qual", palette = "Paired") +
    theme_bw() +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5,
                                     size = 6)) +
    labs(x = "R packages for gene expression differential analysis",
         y = "Number of differentially expressed genes")
```



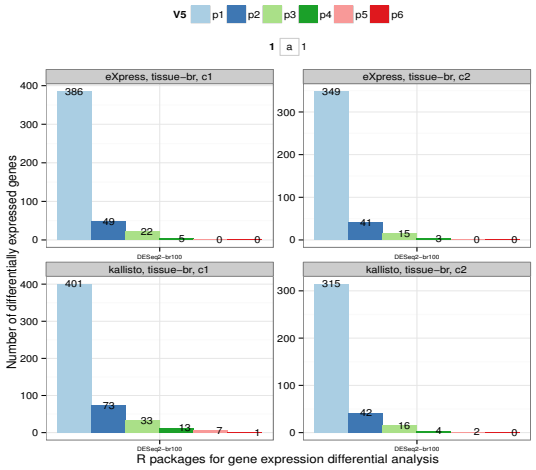

## 1.1 Increasing DEG by changing the trimming rates of raw reads

Getting gene expression by mapping the original raw reads **without trimming** to the **gills** de novo transcriptome.

[1] De novo assembly was carried out with trimmed reads though

```
read.table("./data/summary.br_35078.txt") %>%
```

```
    ggplot(aes(
    x = V2,
    y = V8,
    fill = V5)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = V2,
                  y = V8,
                  ymax = V8,
                  label = V8,
                  size = 1,
                  hjust = 0),
              position = position_dodge(width = 1)) +
    facet_wrap(~ V3 + V4 + V6,
               ncol = 2,
               scales = "free") +
    scale_fill_brewer(type = "qual", palette = "Paired") +
    theme_bw() +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5,
                                     size = 6)) +
    labs(x = "R packages for gene expression differential analysis",
         y = "Number of differentially expressed genes")
```



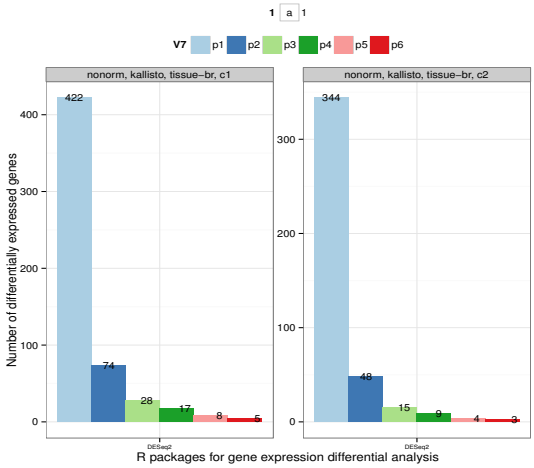

## 1.2 Increasing DEGs by changing the normalization strategy: Fast abundance quantification *kallisto*

The below graph shows the number of differentially expressed genes when raw reads were normalized **separately** for each biological sample.

↰ All the analyses before were done on normalized reads by grouping all biological samples together

```
read.table("./data/summary.br.nonorm.txt") %>%
```

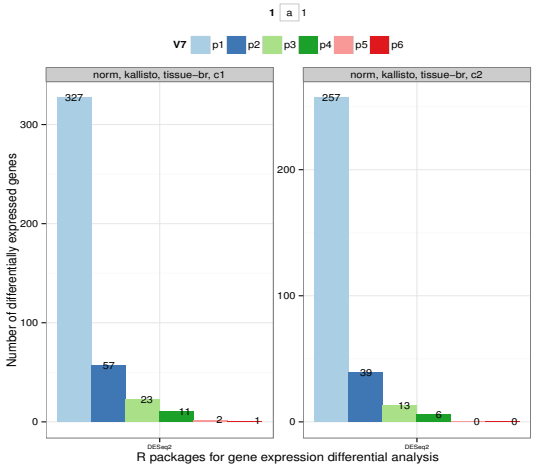
```
    ggplot(aes(
    x = V1,
    y = V10,
    fill = V7)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = V1,
                  y = V10,
                  ymax = V10,
                  label = V10,
                  size = 1,
                  hjust = 0),
              position = position_dodge(width = 1)) +
    facet_wrap(~ V4 + V5 + V6 + V8,
               ncol = 2,
               scales = "free") +
    scale_fill_brewer(type = "qual", palette = 'Paired') +
    theme_bw() +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5,
                                     size = 6)) +
    labs(x = "R packages for gene expression differential analysis",
         y = "Number of differentially expressed genes")
```




The below graph shows the number of differentially expressed genes when raw reads were **NOT** normalized.

```
read.table("./data/summary.br.norm.txt") %>%
```

```
    ggplot(aes(
    x = V1,
    y = V10,
    fill = V7)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = V1,
                  y = V10,
                  ymax = V10,
                  label = V10,
                  size = 1,
                  hjust = 0),
              position = position_dodge(width = 1)) +
    facet_wrap(~ V4 + V5 + V6 + V8,
               ncol = 2,
               scales = "free") +
    scale_fill_brewer(type = "qual", palette ="Paired") +
    theme_bw() +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5,
                                     size = 6)) +
    labs(x = "R packages for gene expression differential analysis",
         y = "Number of differentially expressed genes")
```

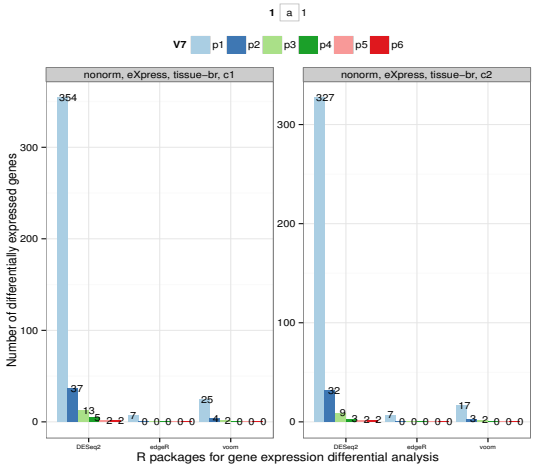



## 1.3 Increasing DEGs by changing the normalization strategy: abundance quantification with alignment *express*

The R package *eXpress* is solely used to quantify the abundance of differentially expressed genes from **NOT normalized** reads and aligned with **Bowtie 1**.

```
read.table("./data/summary.br.nonorm.44062.txt") %>%
```

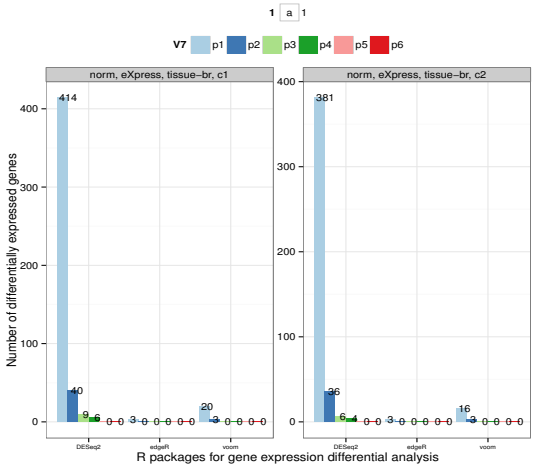
```
    ggplot(aes(
    x = V1,
    y = V10,
    fill = V7)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = V1,
                  y = V10,
                  ymax = V10,
                  label = V10,
                  size = 1,
                  hjust = 0),
              position = position_dodge(width = 1)) +
    facet_wrap(~ V4 + V5 + V6 + V8,
               ncol = 2,
               scales = "free") +
    scale_fill_brewer(type = "qual", palette = "Paired") +
    theme_bw() +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5,
                                     size = 6)) +
    labs(x = "R packages for gene expression differential analysis",
         y = "Number of differentially expressed genes")
```




The R package *eXpress* is solely used to quantify the abundance of differentially expressed genes from **normalized** reads and aligned with **Bowtie 1**.

```
read.table("./data/summary.br.norm.44060.txt") %>%
```

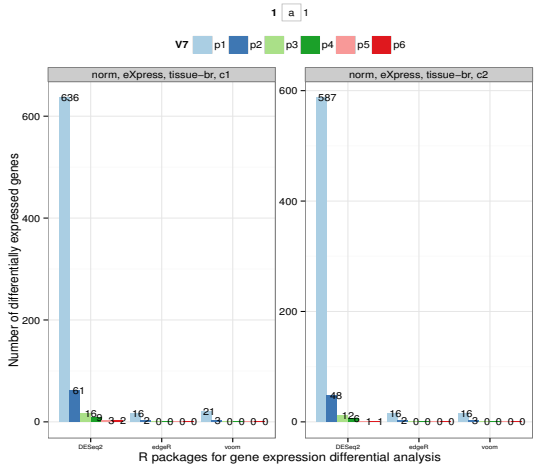
```
    ggplot(aes(
    x = V1,
    y = V10,
    fill = V7)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = V1,
                  y = V10,
                  ymax = V10,
                  label = V10,
                  size = 1,
                  hjust = 0),
              position = position_dodge(width = 1)) +
    facet_wrap(~ V4 + V5 + V6 + V8,
               ncol = 2,
               scales = "free") +
    scale_fill_brewer(type = "qual", palette = "Paired") +
    theme_bw() +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5,
                                     size = 6)) +
    labs(x = "R packages for gene expression differential analysis",
         y = "Number of differentially expressed genes")
```




The R package *eXpress* is solely used to quantify the abundance of differentially expressed genes from **normalized** reads and aligned with **Bowtie 2**.

```
read.table("./data/summary.br.norm.44061.txt") %>%
```

```
    ggplot(aes(
    x = V1,
    y = V10,
    fill = V7)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = V1,
                  y = V10,
                  ymax = V10,
                  label = V10,
                  size = 1,
                  hjust = 0),
              position = position_dodge(width = 1)) +
    facet_wrap(~ V4 + V5 + V6 + V8,
               ncol = 2,
               scales = "free") +
    scale_fill_brewer(type = "qual", palette = "Paired") +
    theme_bw() +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5,
                                     size = 6)) +
    labs(x = "R packages for gene expression differential analysis",
         y = "Number of differentially expressed genes")
```



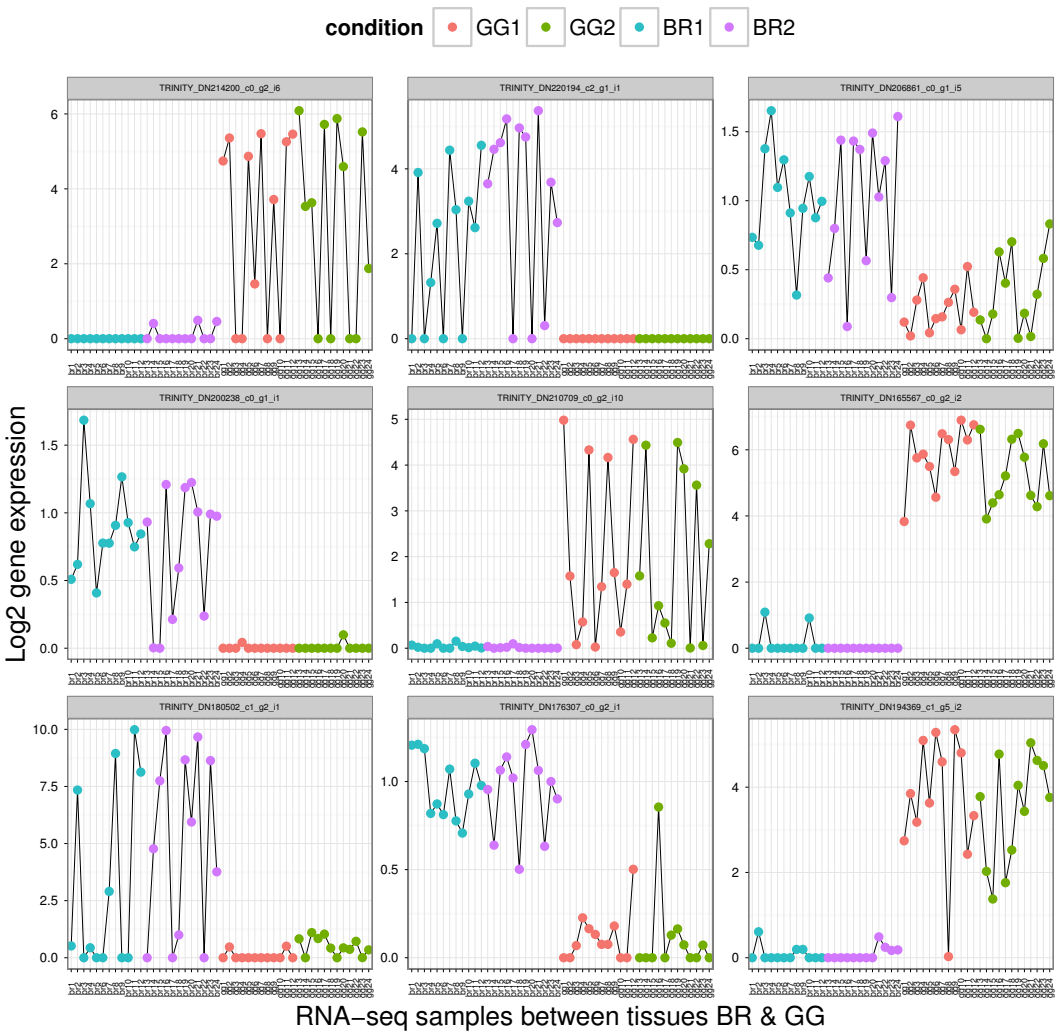

## 2 A linear representation of gene expression

Only selected genes can be represented as follow.

[1] Not more than 10 genes

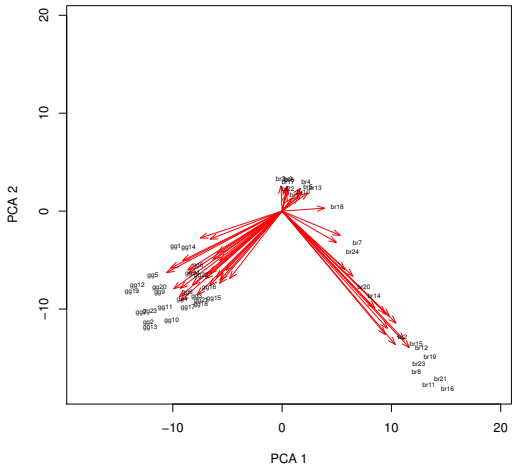
```
dat <- t(read.table("./data/test.txt"))
```

```r
dat <- data.frame(dat,
                  sample = rownames(dat),
                  condition = gl(4,12,48,
                    labels=c("GG1","GG2","BR1","BR2")))
dat %>%
    gather("genes","expression",1:(dim(dat)[2]-2)) %>%
    ggplot(aes(x = factor(sample,
                          c(paste("br",seq(1,24),sep=""),
                            paste("gg",seq(1,24),sep="")))),
               y = expression,
               group = condition)) +
    theme_bw() +
    geom_line(size = .2) +
    geom_point(aes(x = factor(sample),
                   y = expression,
                   colour = condition)) +
    facet_wrap(~ genes,
               ncol = 3,
               scales = "free") +
    labs(x = "RNA-seq samples between tissues BR & GG",
         y = "Log2 gene expression") +
    theme(legend.position = "top",
          axis.text.x = element_text(angle = 90,
                                     vjust = .5, size = 4),
          axis.text.y = element_text(vjust = .5, size = 6),
          strip.text = element_text(size = 4),
          axis.ticks = element_line(size = .2),
          axis.ticks.length = unit(.05, "cm")) +
    scale_fill_brewer(type = "qual", palette = "Paired",
                      name = "Oyster tissues and Diet conditions")
```




Principal component analysis on testing data.

```
dat <- read.table("./data/test.txt")
p = prcomp(dat, retx=T)
scores = p$x
loadings <- p$rotation
sd <- p$sdev
plot(scores[,1], scores[,2],
     xlab="PCA 1", ylab="PCA 2",
     type="n", xlim=c(min(scores[,1:2]),
                      max(scores[,1:2])),
     ylim=c(min(scores[,1:2]),
            max(scores[,1:2])))
arrows(0,0,loadings[,1]*50,loadings[,2]*50,
       length=0.1,angle=20, col="red")
text(loadings[,1]*50*1.3,loadings[,2]*50*1.3,
     rownames(loadings), col="black", cex=0.5)
```




# 3 Identifying foreign transcripts in oyster cells

Microscopy protocols identified the presence of a form of parasitic worm in healthy oyster tissues. As a result we might have a parasitic contamination of our RNA-seq reads. Fortunately, this can give a chance to extract the foreign RNA reads, which are already sequenced, assemble them so we identify later on the parasitic species.

We gathered 5 genomes of parasitic worms first link, second link. We used 2 separate strategies to assemble contigs specific for each genome, i- using genome-guided assembly by mapping reads to genome or ii- mapping contigs directly to genome.

## 3.1 Genome-guided assembly

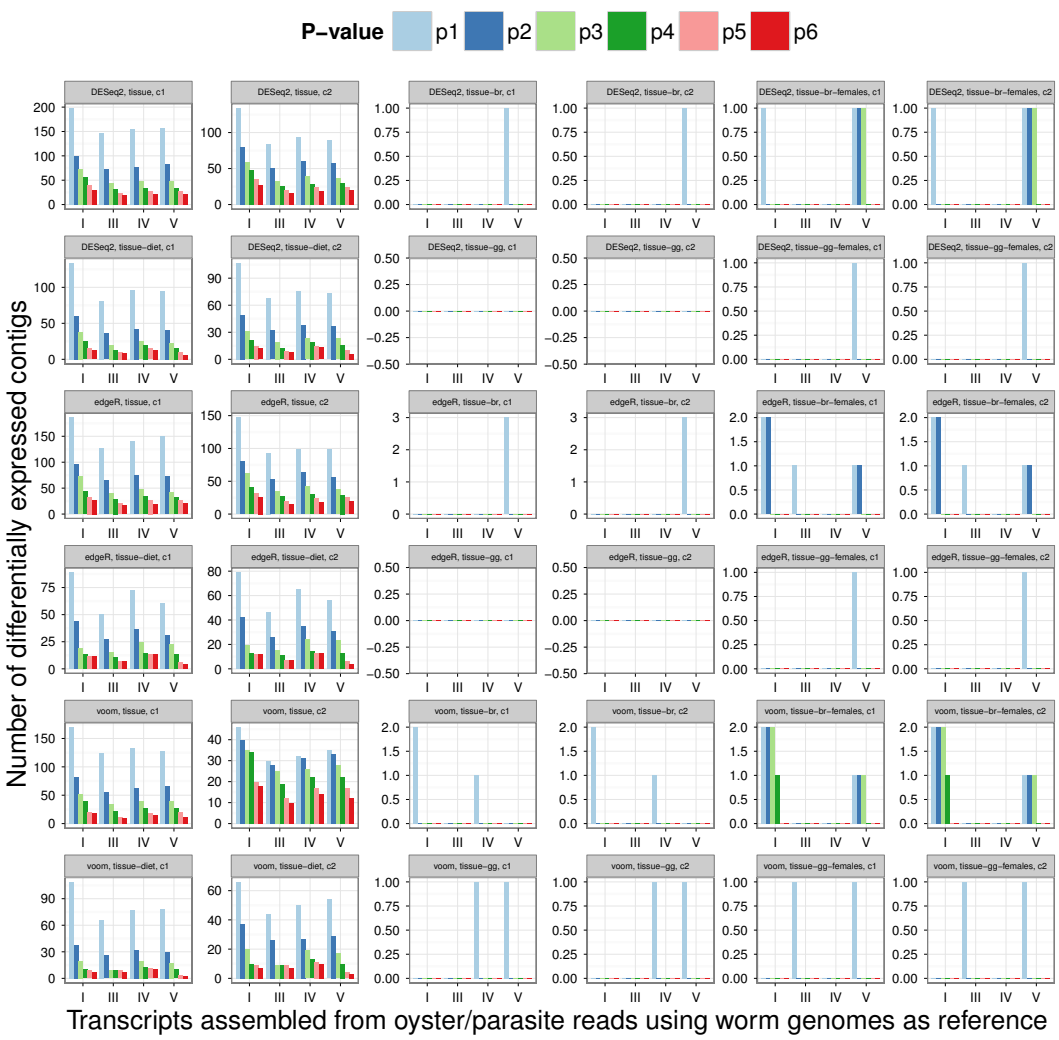
We used bwa to map reads to each genome separately, then use only the proper mapped mates to assemble a special transcriptome using trinity. Those selected reads were then aligned to each of the five new transcriptomes separately. Abundance of reads and differentially expressed transcripts were finally identified.

1. *Clonorchis sinensis*
2. *Opisthorchis viverrini*
3. *Schistosoma haematobium*
4. *Schistosoma japonicum*
5. *Schistosoma mansoni*

All previous worm genomes, fasta files, and some annotations can be found at NCBI.

```
p1 <- read.table("./data/summary.eXpress.134221.txt")
```

```
p3 <- read.table("./data/summary.eXpress.134239.txt")
p4 <- read.table("./data/summary.eXpress.134248.txt")
p5 <- read.table("./data/summary.eXpress.135450.txt")
p1$V9 <- c("I")
p3$V9 <- c("III")
p4$V9 <- c("IV")
p5$V9 <- c("V")
rbind(p1,p3,p4,p5) %>%
    ggplot(aes(x = V9,
               y = V7,
               fill = V4)) +
    theme_bw() +
    geom_bar(stat = "identity",
             position = "dodge") +
    facet_wrap(~ V1 + V3 + V5,
               ncol = 6,
               scales = "free") +
    labs(x = "Transcripts assembled from oyster/parasite reads using worm genomes as reference",
         y = "Number of differentially expressed contigs") +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5, size = 6),
          axis.text.y = element_text(vjust = .5, size = 6),
          strip.text = element_text(size = 4),
          axis.ticks = element_line(size = .2),
          axis.ticks.length = unit(.05, "cm")) +
    scale_fill_brewer(type = "qual", palette = "Paired",
                      name = "P-value")
```

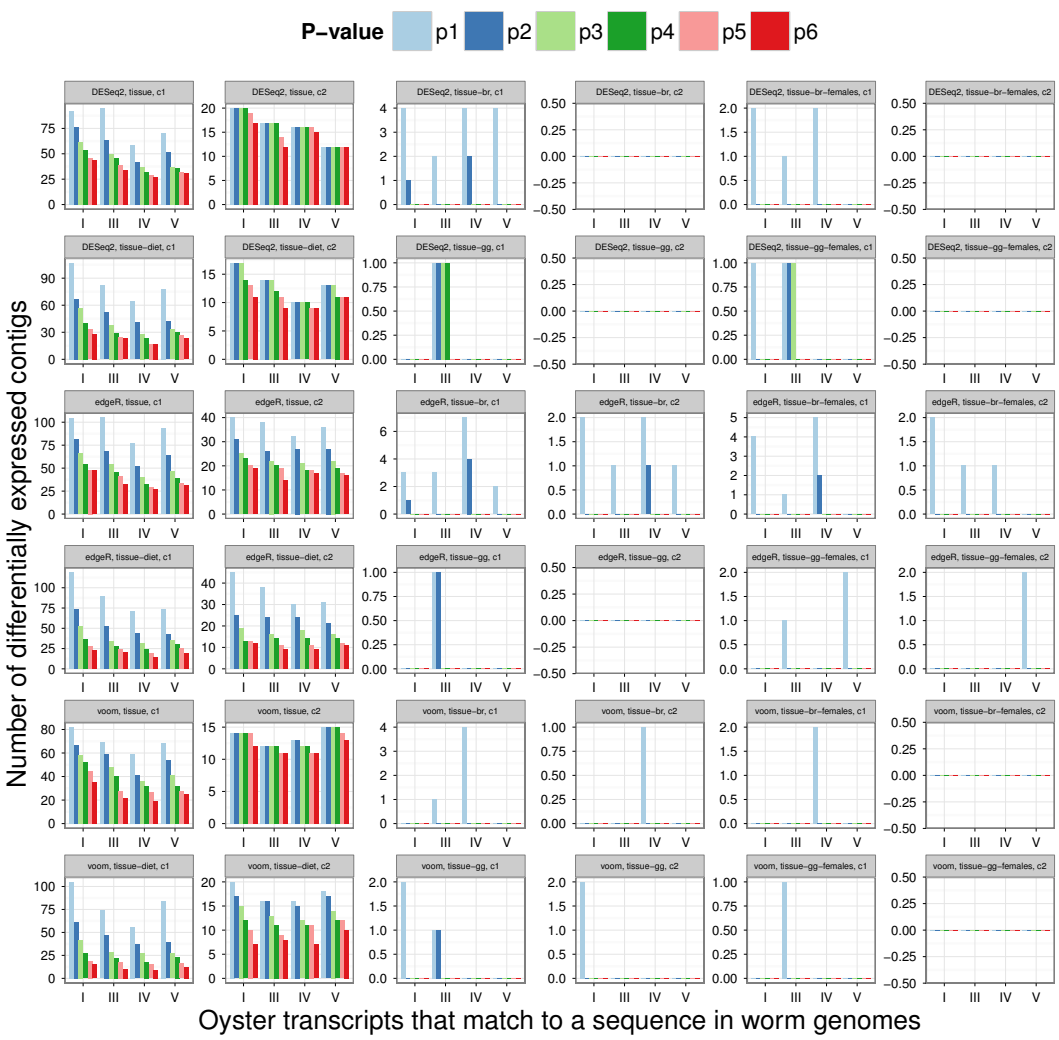



## 3.2 Aligning contigs to worm genomes

Contigs were generated by assembling raw sequencing reads from all tissues and dietary conditions. These transcripts were then aligned to each worm genome separately. We chose the alignments that match at least 500 sequence length.

```
p1 <- read.table("./data/summary.eXpress.134745.txt")
```

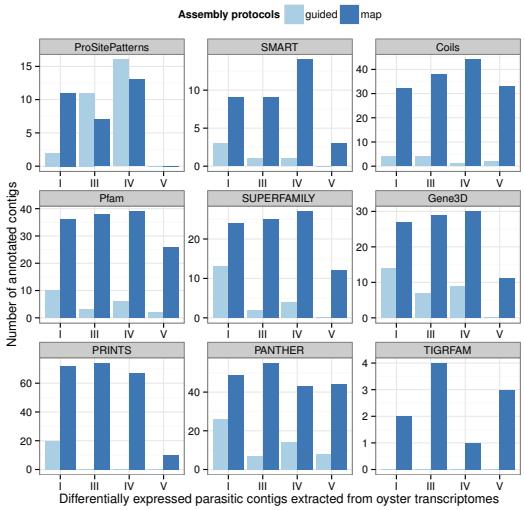
```
p3 <- read.table("./data/summary.eXpress.134744.txt")
p4 <- read.table("./data/summary.eXpress.135311.txt")
p5 <- read.table("./data/summary.eXpress.135424.txt")
p1$V9 <- c("I")
p3$V9 <- c("III")
p4$V9 <- c("IV")
p5$V9 <- c("V")
rbind(p1,p3,p4,p5) %>%
    ggplot(aes(x = V9,
               y = V7,
               fill = V4)) +
    theme_bw() +
    geom_bar(stat = "identity",
             position = "dodge") +
    facet_wrap(~ V1 + V3 + V5,
               ncol = 6,
               scales = "free") +
    labs(x = "Oyster transcripts that match to a sequence in worm genomes",
         y = "Number of differentially expressed contigs") +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5, size = 6),
          axis.text.y = element_text(vjust = .5, size = 6),
          strip.text = element_text(size = 4),
          axis.ticks = element_line(size = .2),
          axis.ticks.length = unit(.05, "cm")) +
    scale_fill_brewer(type = "qual", palette = "Paired",
                      name = "P-value")
```




### 3.3 Annotating genes found from mapping parasitic genomes to oyster reads

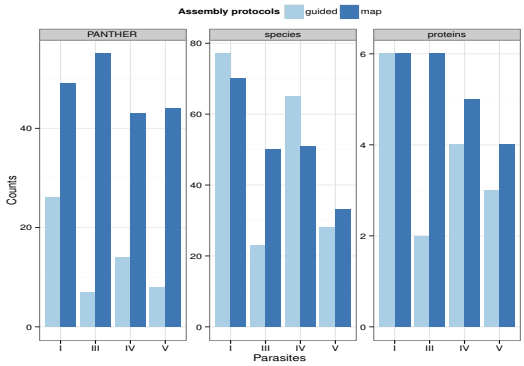
We used protein databases from ten different sources: **ProDom, PANTHER, TIGRFAM, SUPERFAMILY, PIRSF, Gene3D, Pfam, SMART, PROSITEPROFILES, PROSITEPATTERNS, COILS**. Hidden Markov models (HMM) were used to find annotations belonging to differentially expressed genes in section 3.1 and 3.2. We chose tissue specific differentially expressed genes at a p-value of 10e-3 and a c-fold of 2. DESeq2 R packages generated these gene selections between ganglia and gill tissues.

```
read.table("./data/parasite-ips.txt",header = T) %>%
    gather("database", "counts", 2:10) %>%
    ggplot(aes(x = parasite,
               y = counts,
               fill = assembly)) +
    theme_bw() +
    geom_bar(stat = "identity",
             position = "dodge") +
    facet_wrap(~ database,
               ncol = 3,
               scales = "free") +
    labs(x = "Differentially expressed parasitic contigs extracted from oyster transcriptomes",
         y = "Number of annotated contigs") +
    theme(legend.position = "top") +
    scale_fill_brewer(type = "qual", palette = "Paired",
                      name = "Assembly protocols")
```




The Panther database has an extended collection of species, functional domains, and GO-terms compared to other databases. Here is a short summary of what was found in Panther at e-value of 10e-10 and minimum amino acid alignment length of 20.

```
read.table("./data/parasite-panther.txt", header = T) %>%
    gather("category", "count",2:4) %>%
    ggplot(aes(x = parasite,
               y = count,
               fill = assembly)) +
    theme_bw() +
    geom_bar(stat = "identity",
             position = "dodge") +
    facet_wrap(~ category,
               ncol = 3,
               scales = "free") +
    labs(x = "Parasites",
         y = "Counts") +
    theme(legend.position = "top") +
    scale_fill_brewer(type = "qual", palette = "Paired",
                      name = "Assembly protocols")
```

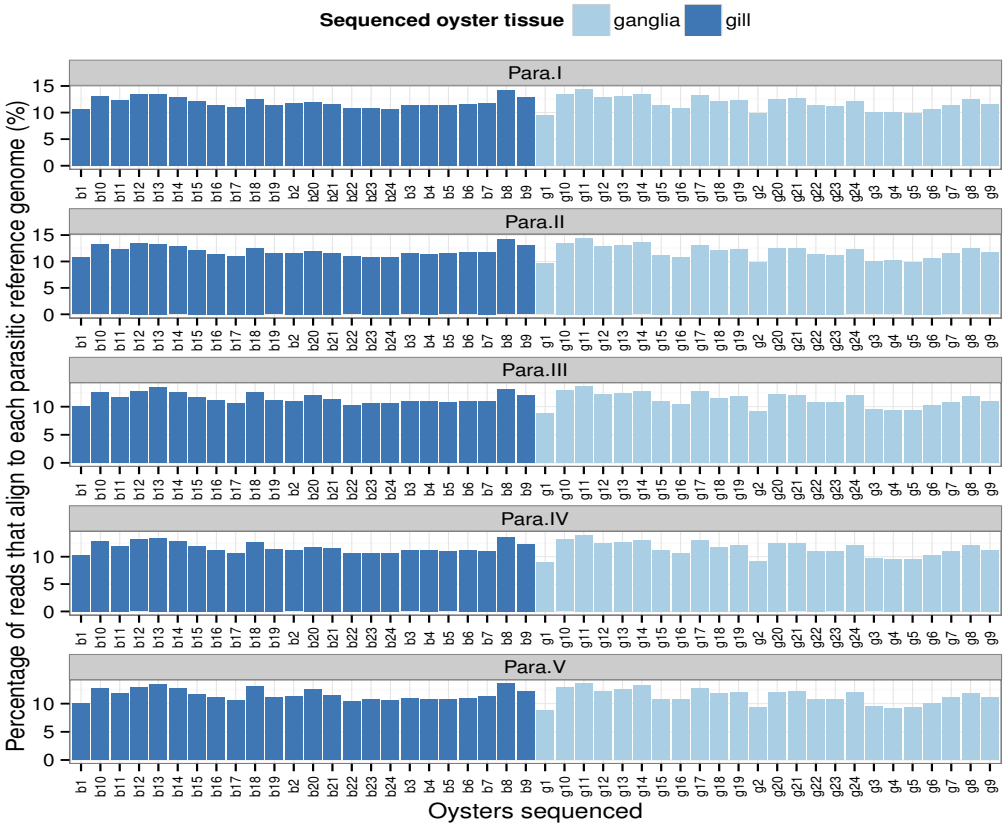



## 3.4 How many reads map by pair mates to each parasite genome?

Only pair reads where both mates map in a forward and reverse fashion are displayed.

```
df <- read.table("./data/parasite.mates.mapping.txt", header = T)
```

```
dfx <- sapply(df[,c(2:6)], function(x) {(x/df$Total)*100})
dfx <- data.frame(dfx, df$Tissue, df$Sample)
#require(scales)
#dfy[order(dfy$counts , decreasing=FALSE),] %>% head
gather(dfx, "parasite", "counts", 1:5) %>%
#arrange(desc(counts)) %>%
    ggplot(aes(x = df.Sample,
               y = counts,
               fill = df.Tissue)) +
    theme_bw() +
    geom_bar(stat = "identity",
             position = "dodge") +
    facet_wrap(~ parasite,
               ncol = 1,
               scales = "free") +
    labs(x = "Oysters sequenced",
         y = "Percentage of reads that align to each parasitic reference genome (%)") +
    theme(legend.position = "top",
          axis.text.x = element_text(angle = 90,
                                     vjust = .5, size = 7)) +
    scale_fill_brewer(type = "qual", palette = "Paired",
                      name = "Sequenced oyster tissue")
#    scale_y_continuous(labels = scales::percent)
```




## 4 System Information

The version number of R and packages loaded for generating the vignette were:

```
###save(list=ls(pattern=".*|.*"),file="PD.Rdata")
```

```
sessionInfo()

R version 3.2.1 (2015-06-18)
Platform: x86_64-unknown-linux-gnu (64-bit)
Running under: elementary OS Luna

locale:
 [1] LC_CTYPE=en_US.UTF-8          LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8           LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8       LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8          LC_NAME=en_US.UTF-8
 [9] LC_ADDRESS=en_US.UTF-8        LC_TELEPHONE=en_US.UTF-8
[11] LC_MEASUREMENT=en_US.UTF-8    LC_IDENTIFICATION=en_US.UTF-8

attached base packages:
[1] grid      stats     graphics  grDevices utils     datasets
[7] methods   base

other attached packages:
 [1] tidyr_0.2.0         dplyr_0.4.2         latticeExtra_0.6-26
 [4] RColorBrewer_1.1-2  glmnet_2.0-2        foreach_1.4.2
 [7] Matrix_1.2-1        leaps_2.9           caret_6.0-47
[10] ggplot2_1.0.1       lattice_0.20-31     xlsx_0.5.7
[13] xlsxjars_0.6.1      rJava_0.9-6         knitr_1.10.5
[16] RevoUtilsMath_3.2.1

loaded via a namespace (and not attached):
 [1] Rcpp_0.11.6          formatR_1.2          nloptr_1.0.4
 [4] plyr_1.8.3           highr_0.5            iterators_1.0.7
 [7] tools_3.2.1          digest_0.6.8         lme4_1.1-8
[10] evaluate_0.7         nlme_3.1-121         gtable_0.1.2
[13] mgcv_1.8-6           DBI_0.3.1            parallel_3.2.1
[16] brglm_0.5-9          SparseM_1.6          proto_0.3-10
[19] BradleyTerry2_1.0-6 stringr_1.0.0        gtools_3.5.0
[22] nnet_7.3-10          R6_2.0.1             minqa_1.2.4
[25] reshape2_1.4.1       car_2.0-25           magrittr_1.5
[28] scales_0.2.5         codetools_0.2-11     MASS_7.3-41
[31] splines_3.2.1        assertthat_0.1       pbkrtest_0.4-2
[34] colorspace_1.2-6     labeling_0.3         quantreg_5.11
[37] stringi_0.5-5        lazyeval_0.1.10      munsell_0.4.2
```