R implementation

Sleiman Bassim, PhD

December 14, 2016

Loaded functions:

```r
#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.

```r
pkgs <- c('caret','leaps','glmnet','lattice',
          'latticeExtra', 'dplyr', 'tidyr', 'grid',
          'gplots', 'WGCNA', 'reshape2','stringr')
lapply(pkgs, require, character.only = TRUE)
```

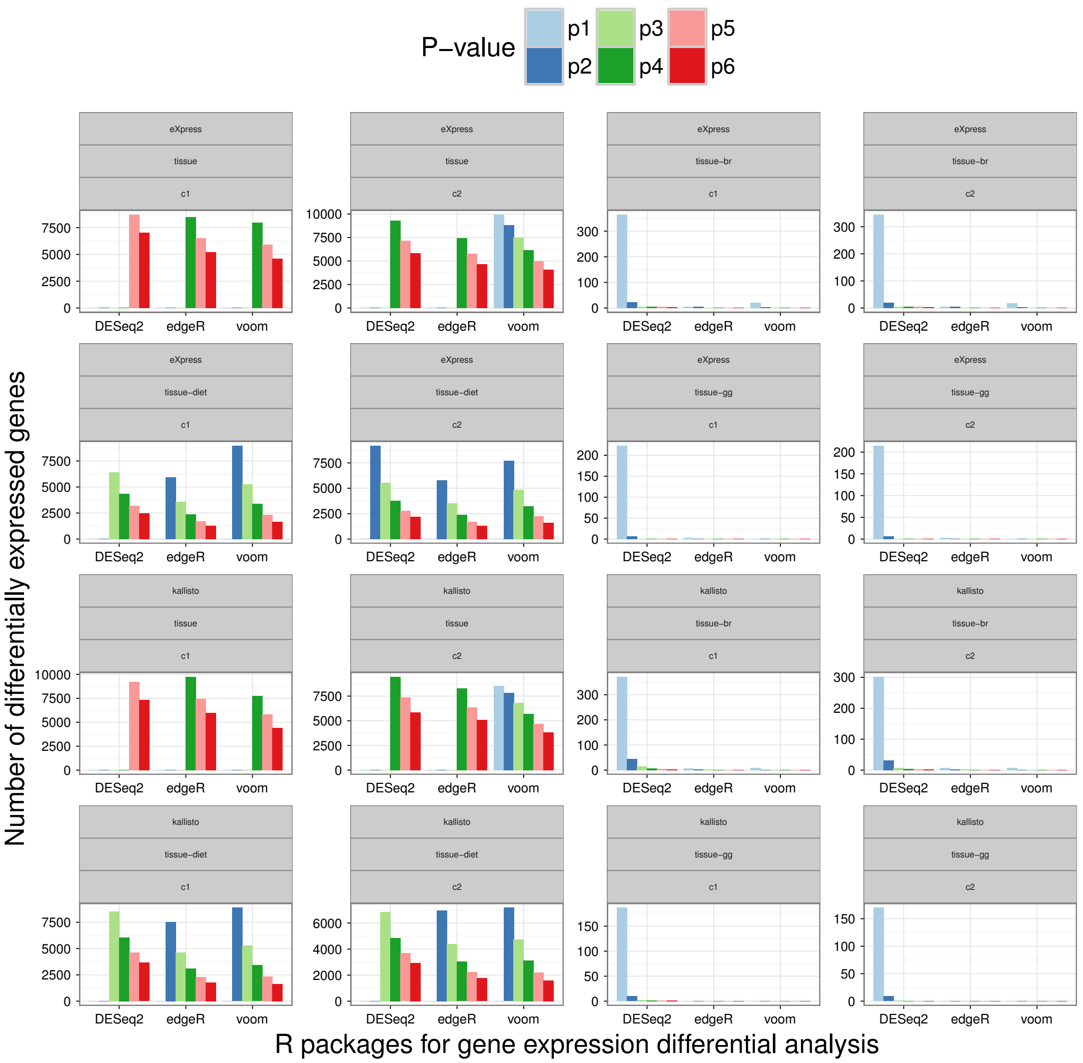
1 Summary of the workflow

For pathway construction we will be using a full featured pipeline that implements i- gene network inference from RNA-seq gene expression data, ii- gene discovery after protein domain alignments to ten well cited protein databases, iii- gene interaction validation from inferred networks published in the European private String database, and finally iv- regularization analysis to eliminate to least probabilistic networks identified, however are less likely to be accurately constructed. Briefly, R script is used to calculate a weighted matrix for every gene discovered from the RNA-seq data of ganglia samples. This correlated matrix will be used to select genes with similar patterns of expression that will constitute an ensemble of gene modules. Each module includes eigengenes, which through the guilt-by-association method are identified to be signatures of a cell response at a precise moment after detection of a stimuli. These group of genes will go through a second set of analysis to calculate a probability of interaction at different error thresholds. Next step is to identify the actual genes for their protein functions, hence create an image of the interactions between biological processes specific to these functions. Moreover, validation of the gene networks will depend on the nature of these processes. There is many online sequence databases that deliver this information after acquiring gene names and protein functions. However, we will be using an approach that contains both statistical inference of gene interactions and functional association between processes. Therefore, after constructing gene-gene interaction networks we will compare them to already established networks inferred from all published data over many clades and between several closely related species. Finally, we will use machine learning methods, mainly regularization methods which are known to reduce the hyperspace of the data (that is reduce the number of candidates, eg., genes, networks) and let us focus on the most probabilistically accurate results by keeping the least biased gene networks. Overall the pipeline has already been tested on different datasets, databases are already been acquired and deployed on our servers, and the R code and many of the packages to be used have been tested and some published.

2 Differentially expressed genes

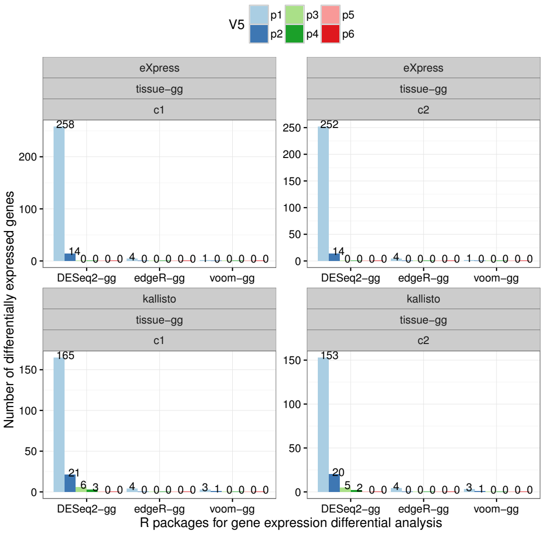
Differentially expressed genes are counted from mapping **both gills and ganglia** sequenced samples to reference transcriptome built from all samples.

```r
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"))
```



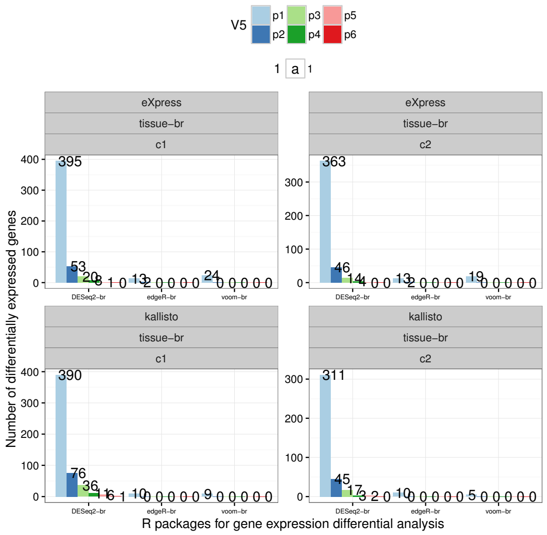
Differentially expressed genes are counted from mapping **ganglia** sequenced samples to reference transcriptome built from all samples.

```
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,
                  vjust = .2),
              position = position_dodge(width = 1),
              size = 3.5,
              hjust = 0) +
    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 = 10)) +
    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")
```

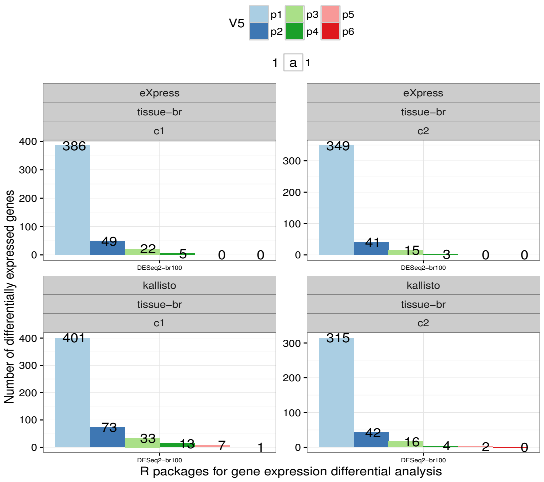


2.1 Increasing DEG by changing the trimming rates of raw reads

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

↰ 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")
```

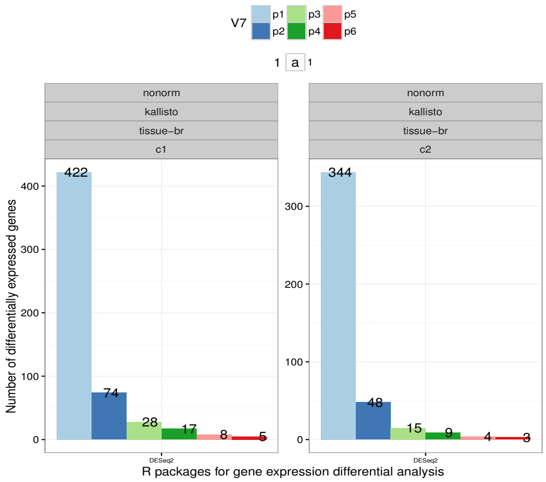


2.2 Increasing DEGs by changing the normalization strategy: Fast abundance quantification *kallisto*

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

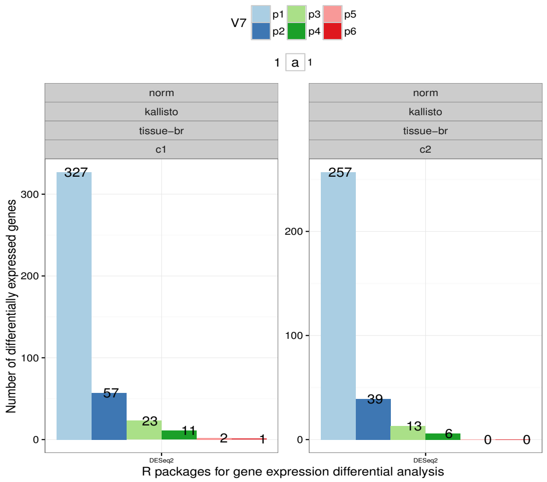
↰ All the analyses before were done on normalized reads by grouping all biological samples together

```
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.

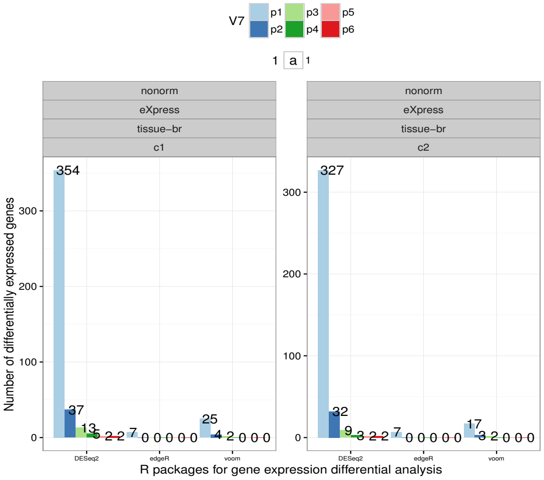
```
read.table("./data/summary.br.norm.txt") %>%
    ggplot(aes(
    x = V1,
    y = V10,
    fill = V7)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = V1,
                  y = V10,
                  ymax = V10,
                  label = V10,
                  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")
```



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

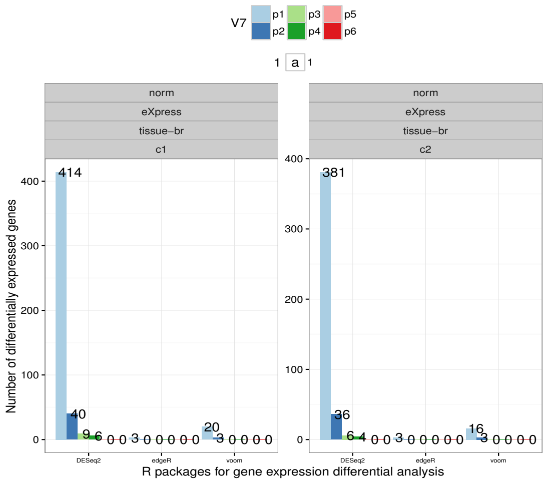
The R package *eXpress* is solely used to quantify the abundance of differentially expressed genes from **NOT normalized** reads and aligned with **Bowtie 1**.

```
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")
```



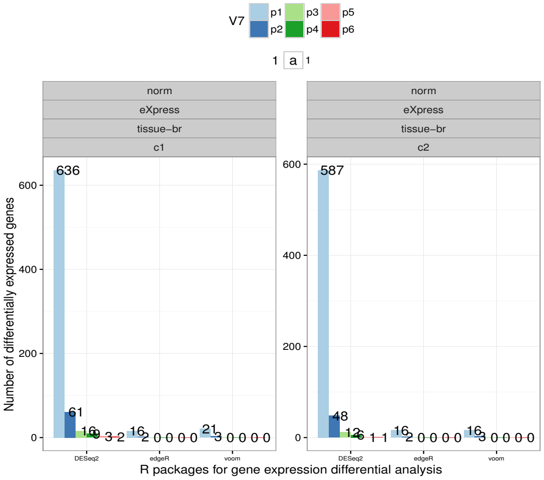
The R package *eXpress* is solely used to quantify the abundance of differentially expressed genes from **normalized** reads and aligned with **Bowtie 1**.

```
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")
```



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.

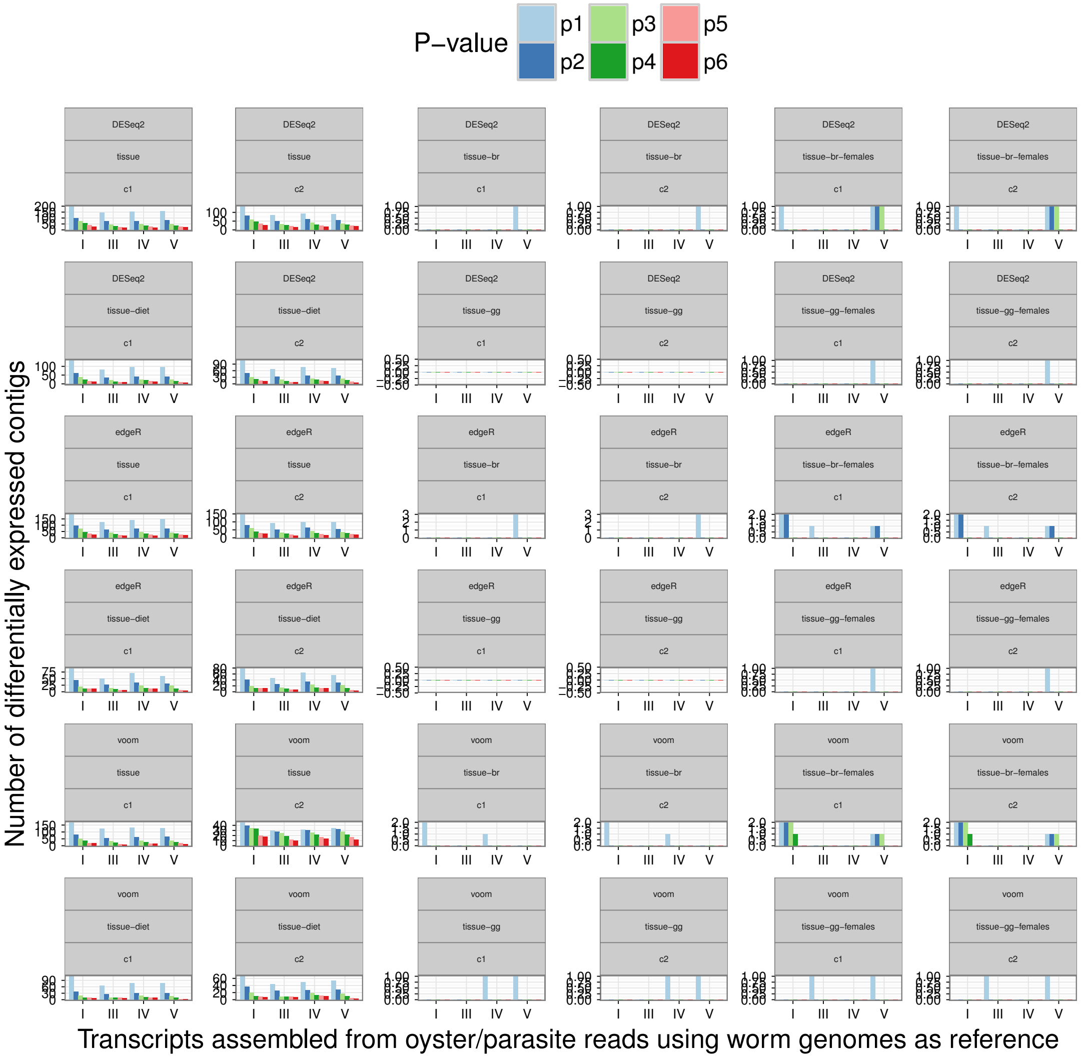
3.1 Genome-guided assembly

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")
```

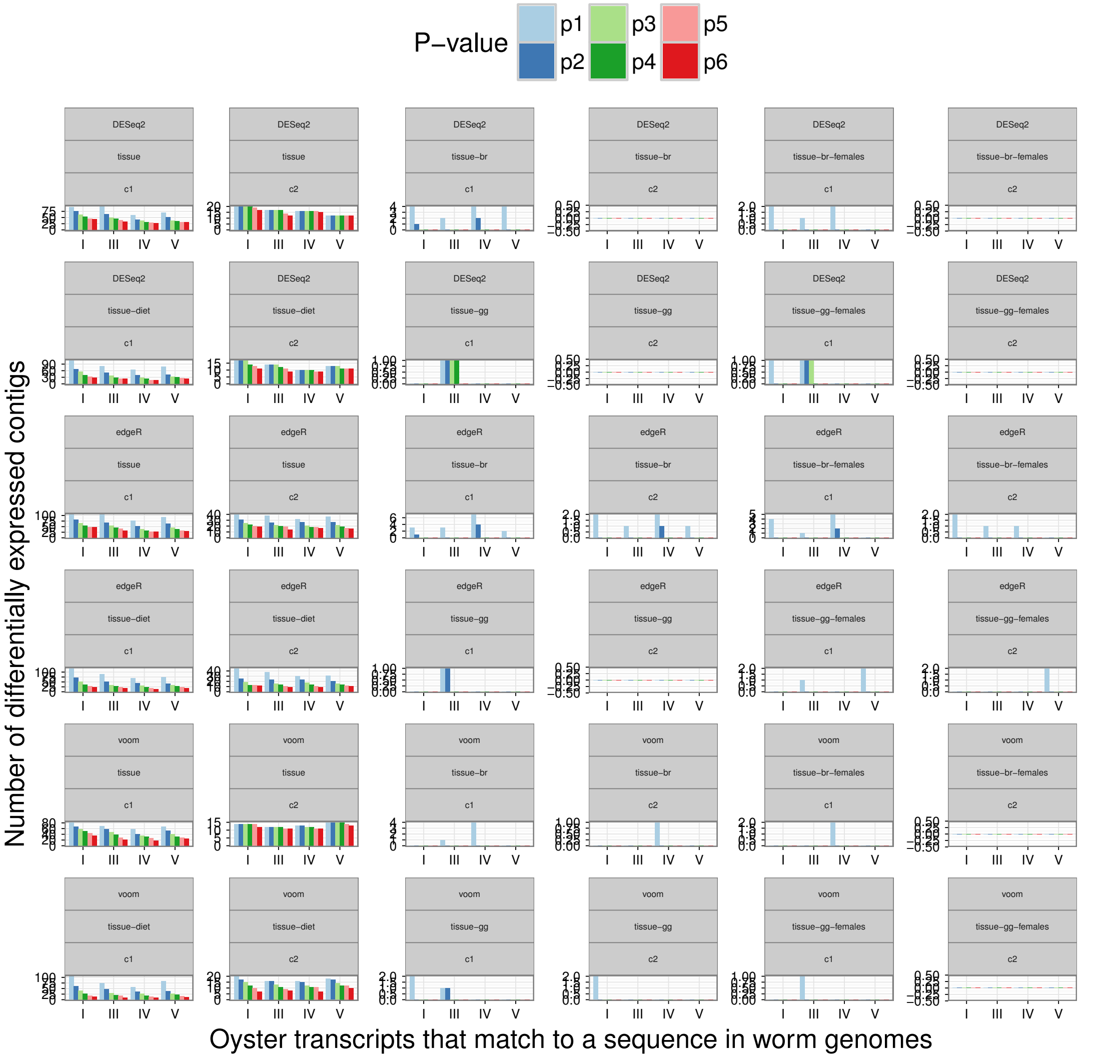


3.2 Aligning contigs to worm genomes

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")
```

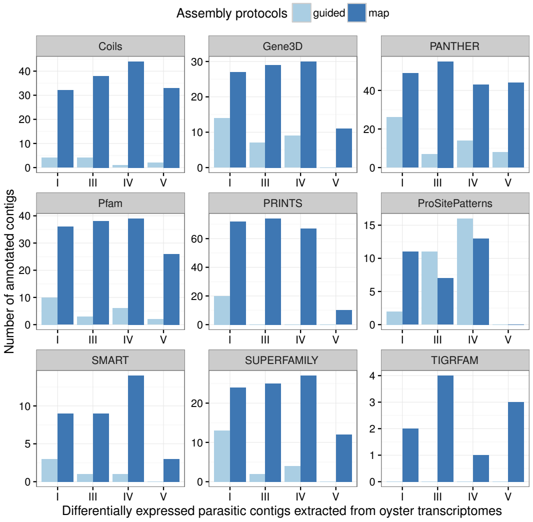
```r
p3 <- read.table("./data/summary.eXpress.134744.txt")
p4 <- read.table("./data/summary.eXpress.135311.txt")
p5 <- read.table("./data/summary.eXpress.135424.txt")
p1$V9 <- c("I")
p3$V9 <- c("III")
p4$V9 <- c("IV")
p5$V9 <- c("V")
rbind(p1,p3,p4,p5) %>%
    ggplot(aes(x = V9,
               y = V7,
               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

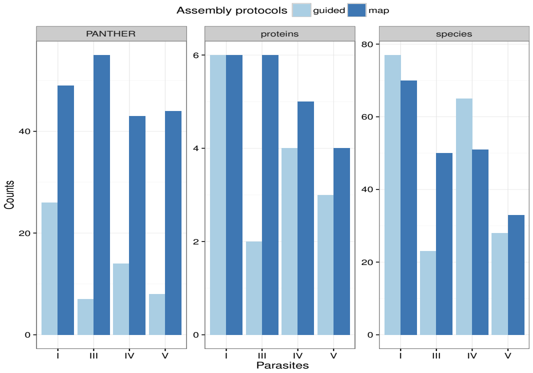
We used protein databases from ten different sources: **ProDom, PANTHER, TIGRFAM, SUPERFAMILY, PIRSF, Gene3D, Pfam, SMART, PROSITEPROFILES, PROSITEPATTERNS, COILS**. Hidden Markov models (HMM) were used to find annotations belonging to differentially expressed genes in section 3.1 and 3.2. We chose tissue 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.

```r
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")
```

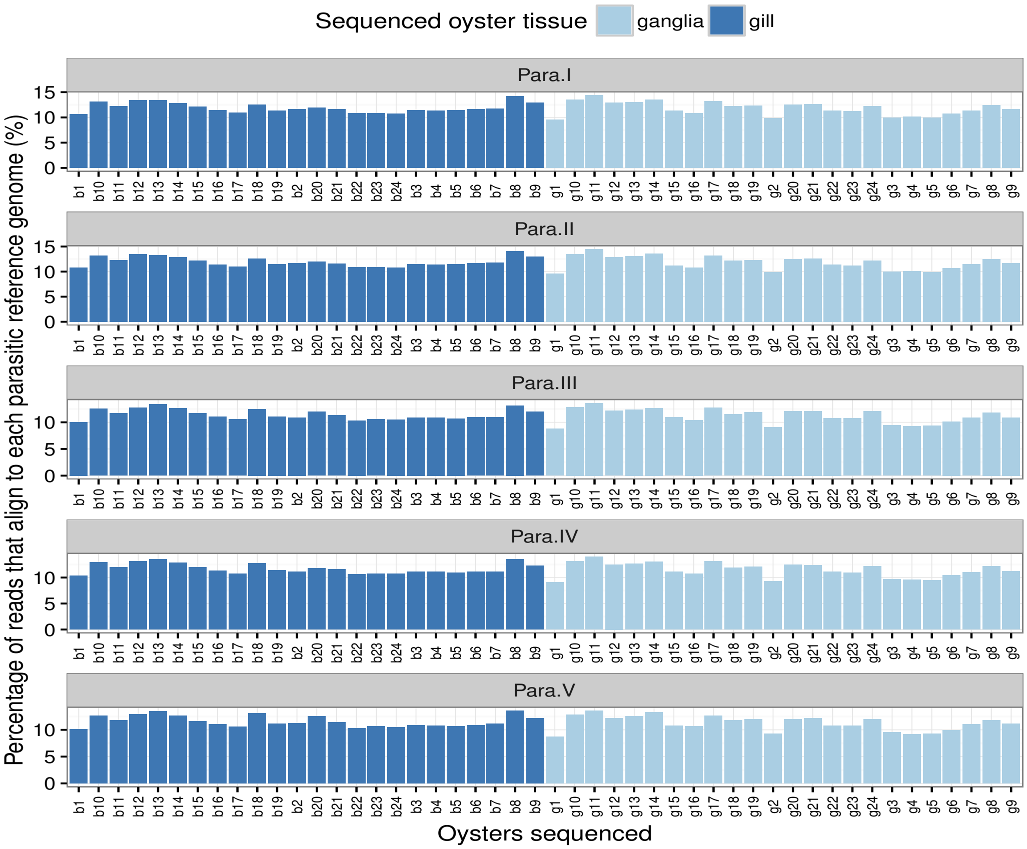


3.4 How many reads map by pair mates to each parasite genome?

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)
```



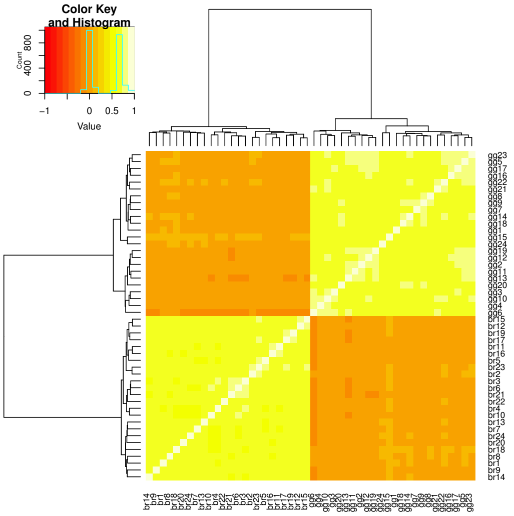
4 Network inference

We will be using weighted genes co-expression analyses from Zhang and Horrath 2005. Machine learning regularization techniques will be applied after to emphasize selection of significant gene modules.

4.1 Correlation analysis

The analysis sill include: Sample verification and clustering. Similarity matrix between samples. Conversion of similarity matrix to adjacency matrix. Detection of co-expression modules.

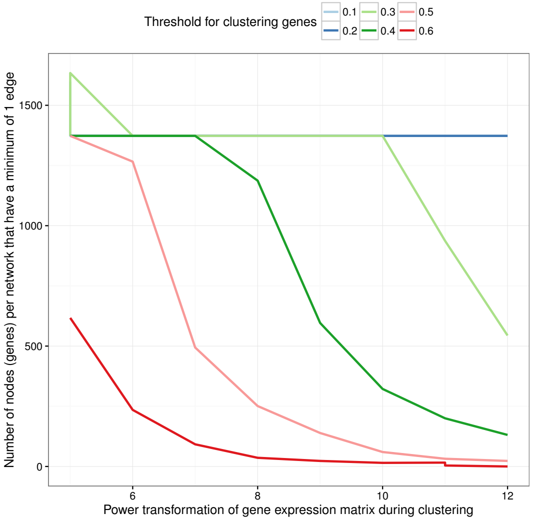
```
counts <- read.table("./data/diffExpr.P1e-4_C2.matrix.log2.dat", header = T)
heatmap.2(cor(counts), trace = 'none')
```



4.2 Network reconstruction

Plot number of nodes and edges after network construction at different power transformations of gene expression FPKMs, using different thresholds for significant cluster sizes. Power transformation reduces low/false correlations between genes Zhang and Horvath (2005 PDF).

```
#read.xlsx("./data/network.stats.xlsx", header = TRUE, sheetIndex = 1) %>%
read.table("./data/network.stats.txt", header = TRUE) %>%
    ggplot(aes(x = power_trans,
               y = node)) +
    theme_bw() +
    geom_line(aes(color = factor(threshold)),
              size = 1  )+
    labs(x = "Power transformation of gene expression matrix during clustering",
         y = "Number of nodes (genes) per network that have a minimum of 1 edge") +
    theme(legend.position = "top") +
    scale_color_brewer(type = "qual", palette = "Paired",
                       name = "Threshold for clustering genes")
```



4.3 Cluster composition

After computing correlations between genes, networks were inferred. Networks are organized into clusters and each contain modules of highly correlated and co-functional genes. The selected network is based on **power transformation of 7** to avoid selecting low correlated scores, **threshold of 0.4** for significance, and **maximum number of genes per module of 100**. Detail of the scores:

- **Nodes**, differentially expressed transcripts
- **Modules**, group of genes per cluster (represented by different colors)

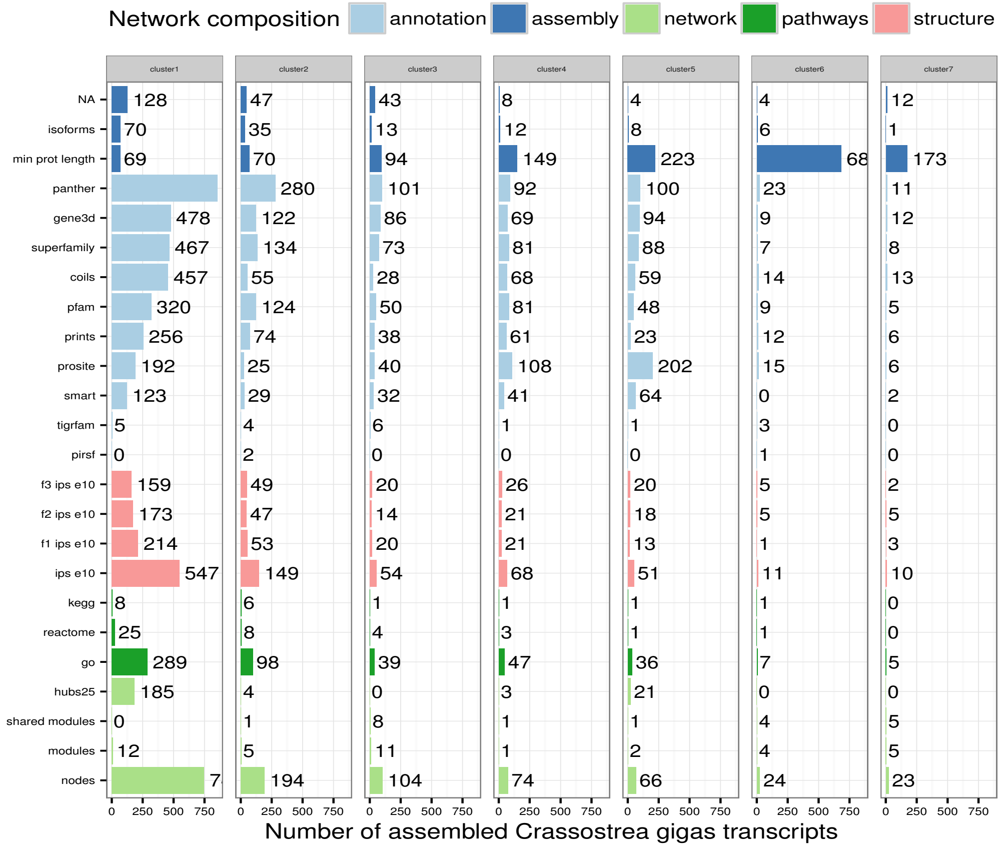
- **Shared modules**, genes corresponding to one module and appear in two clusters or more
- **Hubs25**, genes that have at least 25 edges (degree)
- **GO**, Gene Ontology terms and not inferred from electronic annotation
- **Reactome**, reactome IDs found
- **KEGG**, KEGG IDs found
- **NA**, non annotated contigs found in any of sequence database but still found differentially expressed
- **Isoforms**, transcripts that have been assembled similarly from the same read bins
- **IPS e10**, annotations at a minimum e-value of 10^{-10}
- **Frame1**, first frameshift annotations of translated transcripts at a minimum e-value of 10^{-10}
- **Frame2**, second frameshift annotations of translated transcripts at a minimum e-value of 10^{-10}
- **Frame3**, third frameshift annotations of translated transcripts at a minimum e-value of 10^{-10}
- **PIRSF**, domain classification (database), e-value 10^{-10}
- **TIGRFAM**, subfamilies classification (database), e-value 10^{-10}
- **SMART**, Swissprot, Trembl, and Ensembl domains (database), e-value 10^{-10}
- **PROSITE**, functional domains (database), e-value 10^{-10}
- **PRINTS**, conserved motifs in SwissProts, fingerprinting (database), e-value 10^{-10}
- **PFAM**, protein domains (database), e-value 10^{-10}
- **COILS**, coiled coil confirmation (database), e-value 10^{-10}
- **SUPERFAMILY**, domains, phylogeny and taxon (database), e-value 10^{-10}
- **Gene3D**, domain families (database), e-value 10^{-10}
- **PANTHER**, domains and pathways (database), e-value 10^{-10}
- **Min prot length**, the minimum length of annotated proteins, e-value 10^{-10} (no frameshift preselection)

```
df <- read.table("./data/cluster.stats.txt", header = TRUE, sep = "\t")
```

```
head(df)
```

```
      categories cluster1 cluster2 cluster3 cluster4 cluster5
1          nodes      748      194      104       74       66
2        modules       12        5       11        1        2
3 shared modules        0        1        8        1        1
4         hubs25      185        4        0        3       21
5             go      289       98       39       47       36
6       reactome       25        8        4        3        1
  cluster6 cluster7    label
1       24       23  network
2        4        5  network
3        4        5  network
4        0        0  network
5        7        5 pathways
6        1        0 pathways
```

```
df$categories <- factor(df$categories, levels = df[, 1])
gather(df[-c(25:27), ], "clusters", "counts", 2:8) %>%
    ggplot(aes(x = categories,
               y = counts,
               fill = label)) +
    theme_bw() +
    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)) +
    coord_flip() +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = categories,
                  y = counts,
                  ymax = counts,
                  label = counts,
                  hjust = 0),
              hjust = -.15,
              size = 3.5) +
    facet_wrap( ~ clusters,
               ncol = 7) +
    theme(legend.position = "top") +
    scale_fill_brewer(type = "qual", palette = "Paired",
                      name = "Network composition") +
    labs(y = "Number of assembled Crassostrea gigas transcripts",
         x = "")
```

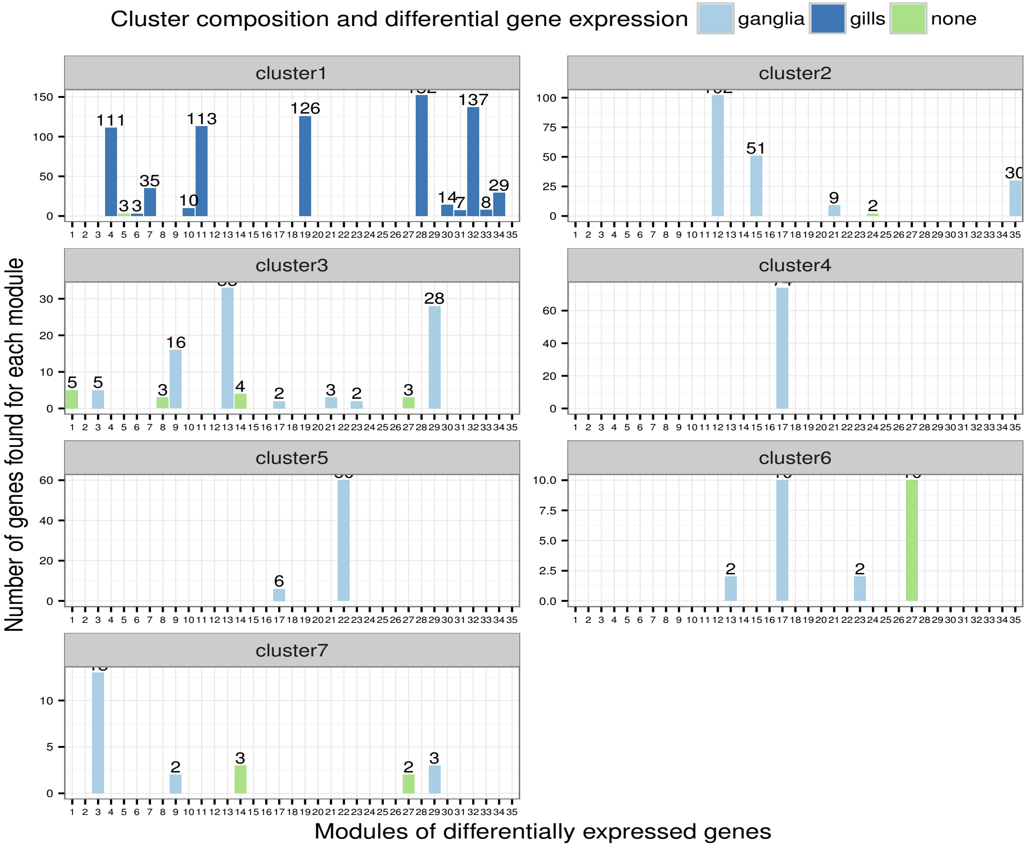


4.4 Module composition

A module is a group of highly correlated genes. Each gene belong to one module. A module can appear in more than one cluster. Below is the composition of each cluster, each separated **based only on differentially expressed genes** between gills and ganglia oyster tissues. Hierachical clustering was done on each module separately to group genes within modules using 5000 bootstraps.

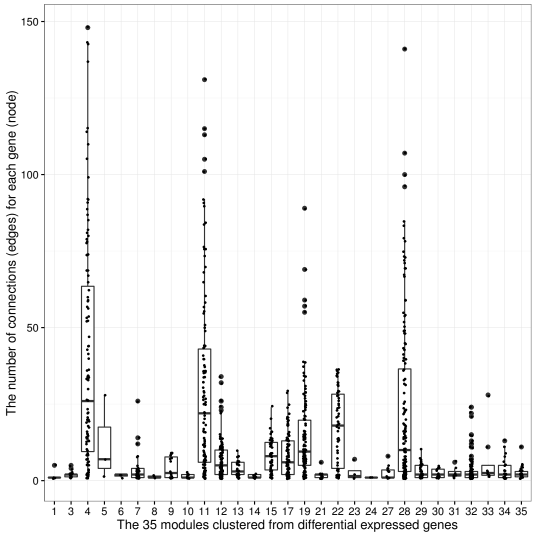
↰ Modules labeled in green contain less than 10 genes, thus hierarchical clustering cannot be done. They are although expressed in Ganglia except module 5.

```
df <- read.table("./data/module.stats.txt", header = T, sep = "\t", fill = T)
df$modules <- as.factor(df$modules)
gather(df, "clusters", "counts", 2:8) %>%
    ggplot(aes(x = modules,
               y = counts,
               fill = network)) +
    theme_bw() +
#    coord_flip() +
    theme(legend.position = "top",
          axis.text.x = element_text(vjust = .5, size = 4.5),
          axis.text.y = element_text(vjust = .5, size = 6)) +
    geom_bar(stat = "identity",
             position = "dodge") +
    geom_text(aes(x = modules,
                  y = counts,
                  ymax = counts,
                  label = counts),
              position = position_dodge(width = 1),
              size = 3,
              vjust = -.2,
              hjust = .5) +
    labs(x = "Modules of differentially expressed genes",
         y = "Number of genes found for each module") +
    facet_wrap( ~ clusters, ncol = 2, scales = "free") +
    scale_fill_brewer(type = "qual", palette = "Paired",
                      name = "Cluster composition and differential gene expression")

Warning:  Removed 204 rows containing missing values (geom_text).
```



Here we show a distribution of the number of edges per node as to showcase hubs and their degrees.

```
df <- read.table("./data/degree.stats.txt", header = T, sep = "\t")
#boxplot(Degree~modules, data=df)

df$modules <- as.factor(df$modules)
ggplot(df, aes(x = modules,
               y = Degree)) +
    geom_boxplot() +
    geom_jitter(width = .2, size = .5) +
    theme_bw() +
    labs(x = "The 35 modules clustered from differential expressed genes",
         y = "The number of connections (edges) for each gene (node)")
```

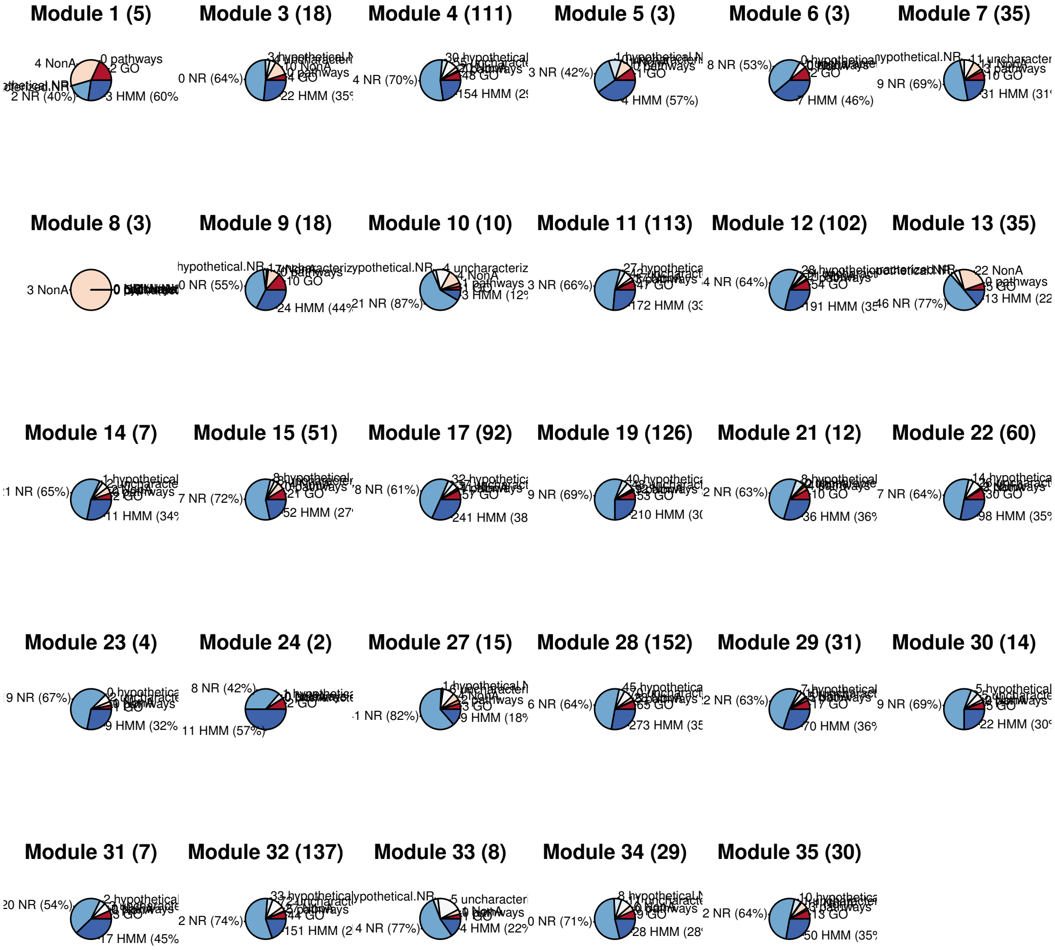


Each module consists of assembled transcripts and grouped together based on their gene expression. These genes have been annotated and assigned a pathway (if a hit score was found). At an e-value of

10^{-10} the top NR and HMM annotations were retained. One transcript can have one or more annotations within each database. The NR database of NCBI contains many uncharacterized genes or hypothetical protein functions.

- **GO**, Gene Ontology terms and not inferred from electronic annotation
- **Pathways**, involve either KEGG and/or reactome IDs
- **NonA**, non annotated transcripts in either NR or protein domain databases
- **Uncharacterized**, the NR database assign this label to genes that have not been assigned a function
- **Hypothetical**, the NR database assign this label to half-structured and potentially functional genes
- **NR**, Non redundant database from NCBI that contains gene names and species associations, annotations are based on the BLAST algorithm
- **HMM**, Hidden Markov Models that annotate peptides based on HMM aligned profiles against 12 protein databases (See Subsection 4.3)

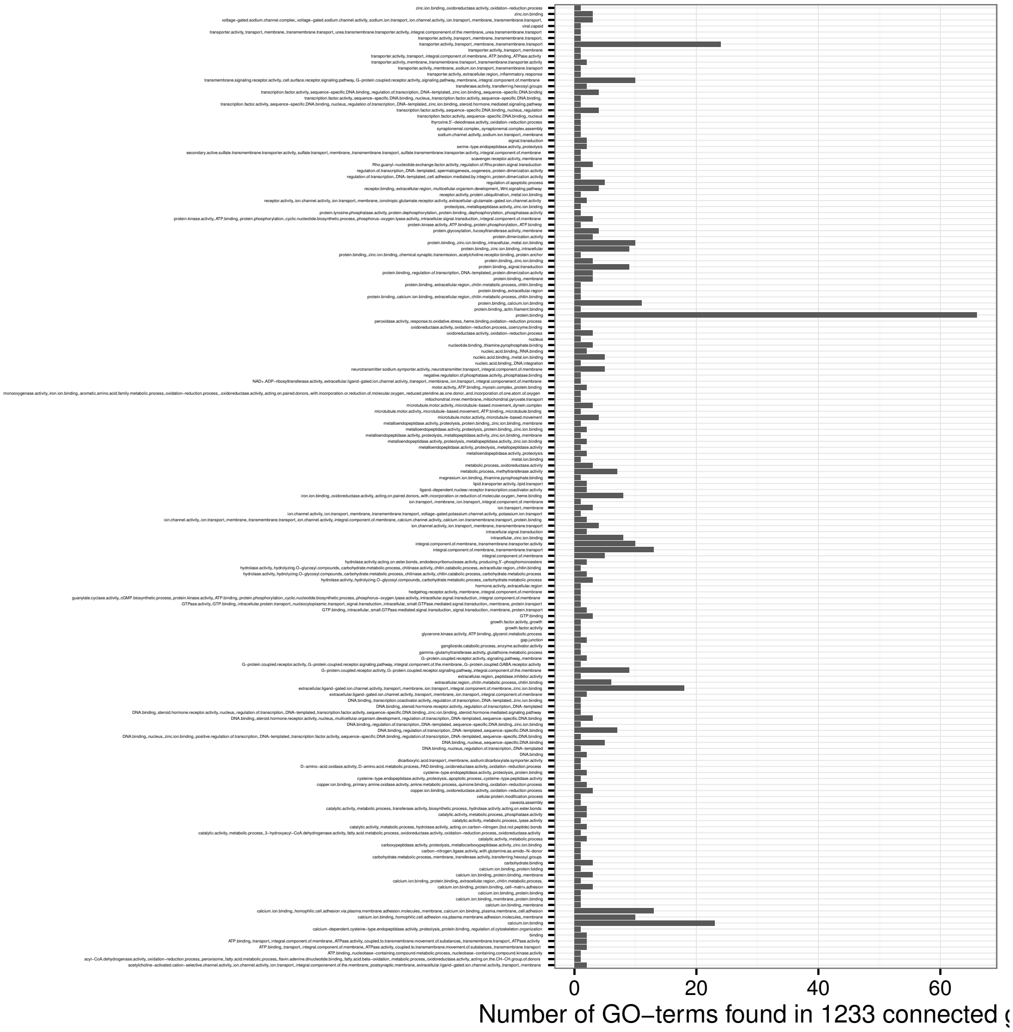
```
par(mfrow=c(5,6), pin = c(.75, .75), mai = c(.5, .4, .4, .4))
x <- c(1, 3:15,17,19,21:24,27:35)
for (i in x){
df <- read.table("./data/module.stats.txt", header = T, sep = "\t")
df <- df[, -c(2:9)]
df <- gather(df[i,], "categories", "counts", c(4,7:12))
df
df$label <- paste(df$counts, df$categories, sep = " ")
df$label[6] <- paste(df$label[6], " (", floor((df$counts[6]/df$Total[1])*100), "%)", sep = "")
df$label[7] <- paste(df$label[7], " (", floor((df$counts[7]/df$Total[1])*100), "%)", sep = "")
title.per.module <- paste("Module ", as.factor(df$modules[1]), " (", df$nodes[1], ")", sep = "")
color.per.module <- c("#b2182b", "#ef8a62", "#fddbc7", "#f7f7f7", "#d1e5f0",
                      "#67a9cf", "#2166ac")
pie(df$counts, labels = df$label,
    cex = .8, main = title.per.module,
    col = color.per.module)
}
```



4.5 GO-terms for all networks

From the 1233 genes connected in the network, 521 GO entries were found.

```
read.table("./data/counted.go.terms.inFull.networks.txt", header = F, sep = "") %>%
    ggplot(aes(x = V2,
               y = V1)) +
    theme_bw() +
    coord_flip() +
    geom_bar(stat = "identity",
             position = "dodge") +
    labs(x="", y = "Number of GO-terms found in 1233 connected genes") +
    theme(axis.text.y = element_text(vjust = .5, size = 1.5))
```



5 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 3.3.1 (2016-06-21)
Platform: x86_64-pc-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=C                 
 [9] LC_ADDRESS=C               LC_TELEPHONE=C            
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C       

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

other attached packages:
 [1] stringr_1.0.0         reshape2_1.4.1        WGCNA_1.51           
 [4] fastcluster_1.1.20    dynamicTreeCut_1.63-1 gplots_3.0.1         
 [7] tidyr_0.5.1           dplyr_0.5.0           latticeExtra_0.6-28  
[10] RColorBrewer_1.1-2    glmnet_2.0-5          foreach_1.4.3        
[13] Matrix_1.2-6          leaps_2.9             caret_6.0-70         
[16] ggplot2_2.1.0         lattice_0.20-33       knitr_1.13           

loaded via a namespace (and not attached):
 [1] Biobase_2.34.0        splines_3.3.1         gtools_3.5.0         
 [4] Formula_1.2-1         assertthat_0.1        highr_0.6            
 [7] stats4_3.3.1          impute_1.48.0         RSQLite_1.0.0        
[10] quantreg_5.26         chron_2.3-47          digest_0.6.9         
[13] minqa_1.2.4           colorspace_1.2-6      preprocessCore_1.36.0
[16] plyr_1.8.4            SparseM_1.7           GO.db_3.4.0          
[19] scales_0.4.0          gdata_2.17.0          lme4_1.1-12          
[22] MatrixModels_0.4-1    tibble_1.0            mgcv_1.8-12          
[25] IRanges_2.8.1         car_2.1-2             nnet_7.3-12          
[28] BiocGenerics_0.20.0   lazyeval_0.2.0        pbkrtest_0.4-6       
[31] survival_2.39-5       magrittr_1.5          evaluate_0.9         
[34] doParallel_1.0.10     nlme_3.1-128          MASS_7.3-45          
[37] foreign_0.8-66        tools_3.3.1           data.table_1.9.6     
[40] formatR_1.4           matrixStats_0.50.2    S4Vectors_0.12.0     
[43] munsell_0.4.3         cluster_2.0.4         AnnotationDbi_1.36.0 
[46] compiler_3.3.1        caTools_1.17.1        nloptr_1.0.4         
[49] iterators_1.0.8       bitops_1.0-6          labeling_0.3         
[52] gtable_0.2.0          codetools_0.2-14      DBI_0.4-1            
[55] R6_2.1.2              gridExtra_2.2.1       Hmisc_3.17-4         
[58] KernSmooth_2.23-15    stringi_1.1.1         parallel_3.3.1       
[61] Rcpp_0.12.5           RevoUtils_10.0.1      rpart_4.1-10         
[64] acepack_1.3-3.3      
```