Apr 15, 2019

Working

SYSB3036 W11: Transcriptomic Analyses

Frank O. . Aylward[1]

[1]Virginia Tech

dx.doi.org/10.17504/protocols.io.z3mf8k6

Frank O. . Aylward
Virginia Tech

1 First let's get the data from GitHub and move inside the new folder we download:

git clone https://github.com/faylward/transcriptomics_tutorial

cd transcriptomics_tutorial

ls

You should see two files: "homework_rnaseq_data.txt" and counts <- as.matrix(read.table(file="homework_rnaseq_data.txt", sep="\t", header=T, row.names=1))". We will be using the second one for the tutorial today.

2 For the rest of the tutorial we will be working in the R console, so move to that and read in the "test_rnaseq_data.txt" file.
This file is a transcriptomic data matrix with Samples as columns and Genes as rows. The values indicate the abundance of different genes in different samples, as determined by a read-mapping analysis (as we went over last week).

counts <- as.matrix(read.table("week11/transcriptomics_tutorial/test_rnaseq_data.txt", sep="\t", header=T, row.names=1))

Note that you will need to make sure the full path to the file is given, not just the file name.
Note that we are also using the "as.matrix" command, since we want R to know this is a numeric matrix of numbers.

3 To get the general size of the new matrix we just loaded we can use "dim":

dim(counts)

and for the first few lines:

head(counts)

4 In this particular dataset the samples belong to different time-points, so we are measuring changes in gene expression over time. Samples were taken every 4 hours over the course of 3 days in a microbial community.

To get an idea of how the gene expression changes over time, we can plot individual genes:

```
plot(counts["Gene1",], type="l", lwd=2, axes=F, xlab=NA)
axis(1, at=c(1:25), colnames(counts), las=2)
axis(2)
```

Note that many genes appear to have a periodic gene expression pattern that peaks once every 24 hours (a "diel cycle").

5 One useful thing we can visualize in these tables is the Rank Abundance Distribution. This tells us how abundant the different genes are

when we order them from most-least abundant. The shape of this curve can help us see the discrepency between the abundance of the different genes, which is often quite severe.

rankAbundance <- sort(rowSums(counts), decreasing=T)

Here we are creating one vector which contains the summed abundance of all rows. To plot this we can use:

plot(rankAbundance, type="l")

Typically this shows quite a dropoff after the first few genes. To get the standard deviation we can use:

sd(rankAbundance)

6 We can also get an idea of the gene abundance distribution by making a histogram:

hist(rankAbundance, breaks=20, col="blue")

7

Now as it turns out analyzing raw counts can be problematic. The abundance between the different gene expression values varies greatly between genes, and this can compromise downstream statistical analyses and visualization. So we will want to transform our data so it can be more useful. One common approach is to log-transform the data, so we will do that here.

log <- as.matrix(log10(counts+1))

Note that we need to add 1 first, since log10(0) is undefined. These added values are called "pseudocounts".
After this we can do the same things we did above with the regular count table.

rankAbundance <- sort(rowSums(log), decreasing=T)

sd(rankAbundance)

plot(rankAbundance, type="l")

hist(rankAbundance, breaks=20, col="blue")

8

Next we will want to visualize the data with clustering and a heatmap. For this we will need to R packages, "gplots" and "RColorBrewer", which you will need to install in your workspace.

install.packages("gplots")

and

install.packages("RColorBrewer")

and then load the packages:

library(gplots)
library(RColorBrewer)

Now with RColorBrewer we can choose a color palette to use when plotting a heatmap:

palette = brewer.pal(9, "GnBu")

And then we can plot the heatmap:

heatmap.2(counts, trace="none", col=palette)

9 And we can do the same for the log-transformed data:

```
heatmap.2(log, trace="none", col=palette)
```