Next Generation Sequencing Technologies: Data Analysis and Applications

Analysis Results and Functional Enrichment Analysis

Dr. Riddhiman Dhar, Department of Biotechnology
Indian Institute of Technology Kharagpur

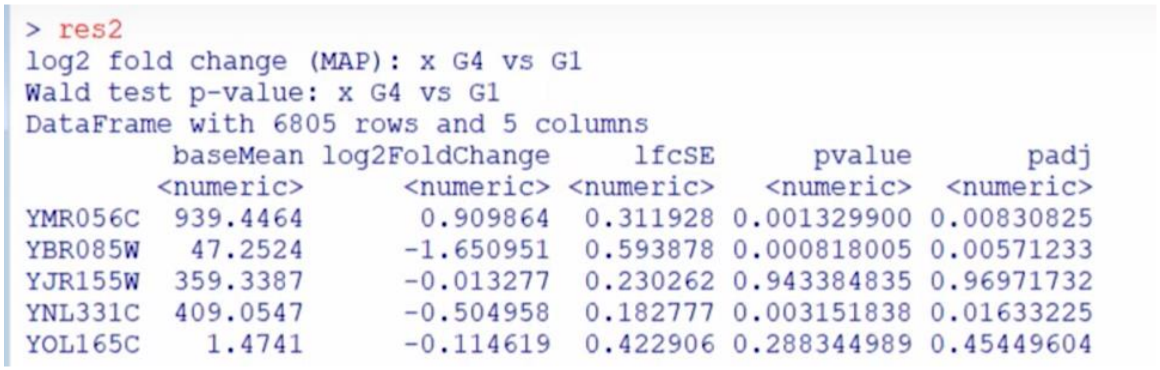
Good day, everyone. Welcome to the course on Next Generation Sequencing Technologies: Data Analysis and Application. In the hands-on, we have done the differential expression analysis for an RNA-seq dataset, right? We have used DEseq2 and subsequently, we used a dataset with the spike-ins in there, the ERCC spike-ins, which we used as a control for our normalization. We used RUVg using the RUVseq package. Now we have the results, so what we want to do is look at the results and maybe visualize some of them.

We have used MA plot already, but maybe we can try the volcano plot, learn how to create this volcano plot, and maybe use a part of the data as a heatmap. Okay, so we have talked about this in theory class, so I would like to show you how we can generate these plots. And then, followed by that, we will talk about functional enrichment analysis very briefly using a package in R again, and we will give you some of these enrichment results, okay? So, that's the agenda for this class. We will try to visualize some of the results that we get and then do the functional enrichment analysis using the R package. Here are the steps. As we have seen, we have now completed up to the differential gene expression analysis, and we have gotten the results. We have seen the results from two types of differential expression analysis, one without spiking and the other with spiking. And we have done the MA plot right for this analysis.

What we are going to do today is look at the volcano plot, the heat map, and then functional analysis. The tools that we are going to use for the heat map are in this Gplots package. There is a heatmap dot 2 function that we would use for our analysis. And for functional enrichment analysis, we can use different types of tools. We have mentioned different tools in the theory class. What we will use in this class is the R package gprofiler2. This is a package that utilizes the hypergeometric test. So, it is a hypergeometric test-based enrichment, and we have talked about this in the theory class. How does the hypergeometric test work? And here is the link to the package and the references if you are interested in the implementations and the actual package, depending on how it is done.

So, you will see that many of these tools are available as web servers. So, you can simply list your genes there, put the list on the server, and select perhaps the organism's name right, and then you can get the results very easily. But sometimes you probably have to do this for a large number of samples repeatedly, and in that case, that becomes very tedious. So, in those cases, you probably would prefer a command-line-based tool that you can use. So, perhaps a package like Profiler, where you can automatically do this analysis for a large number of sets,

So, this is the tool we will use, and we will go step by step, ok? So, the first thing we will do is the volcano plot. We have talked about this in theory, so let us now go to R, and we will actually do the volcano plot first, followed by the heat map, and then we will move on to the functional enrichment analysis. So, for the results part, I can simply show you the results from right to right, and this is what we got after the last analysis. Remember, we have this RUVg method, then the DESeq2 we have run, and this is the final result that we got. So, what you want to do using these results is generate this volcano plot, right? So, you remember the volcano plot, right?



```
> res2
log2 fold change (MAP): x G4 vs G1
Wald test p-value: x G4 vs G1
DataFrame with 6805 rows and 5 columns
         baseMean log2FoldChange     lfcSE      pvalue       padj
        <numeric>      <numeric> <numeric>   <numeric>  <numeric>
YMR056C  939.4464       0.909864  0.311928 0.001329900 0.00830825
YBR085W   47.2524      -1.650951  0.593878 0.000818005 0.00571233
YJR155W  359.3387      -0.013277  0.230262 0.943384835 0.96971732
YNL331C  409.0547      -0.504958  0.182777 0.003151838 0.01633225
YOL165C    1.4741      -0.114619  0.422906 0.288344989 0.45449604
```

So, what do you have in the volcano plot? You have this p value right log minus log 10 p value in the y axis, and in the x axis is the log full change, ok? So, this kind of gives you the significance of the results or significance of this log full change that are observed for change. So, what we have to do first is load the data and see what we get. So, as I have told you right here in R, we can access these results using this dollar sign. What are the results that are available? So, we have this raise in the dollar log to full change. I can simply write a state here, ok, but that will give you all these values for all genes, ok? I can also just check right.

So, for genes, you get these values, and then you can also look at the p value. If you have p values, we can also yeah. So, we have a p value, and we are sorry for giving a p adjusted, ok? So, maybe we can be sorry, yeah. So, here is this right. So, you get p adjusted, and if you just want the p value, not the adjusted one, you can, ok?

So, you can try anything; it does not really matter. So, what we will do is generate this volcano plot, and we will first export it. export this data into some variables or assign some variables for this maybe not x right. So, or maybe x value because x we have used before already. So, for the x value, we can simply assign this data log to full change, ok, and the y value is remembered minus log 10, and then we have p adjusted, so we can say that is ok all right. So, now that we have calculated this and assigned the two x val and y val, what we have to do is plot these points right and generate that graph.

So, the command for doing that is very simple, right? We can say x, y, val n this is something we get, ok? Now, the problem, as you can see, is that most of the points are clustered around here. There are some outliers, and perhaps we can shift this axis a little bit right and change the range of the y axis because that would be better. So, we will do this, and we can make a nicer plot right from this really basic one, ok? So, let us try that, ok? So, what we will do is change the limits of the y axis from 0 to 10. Let us say 10; ok, above 10, we have very few points.

So, maybe these are not really critical for this visualization. And we can see right here that the other option would be to make it logarithmic, which will change the shape of this curve, and similarly for this x axis right. As you can see, most of these values are contained within minus 3 and plus 3 right. So, we can change the range again, and this kind of gives you a very nice shape, and this is expected because the genes that show very low lock will change, which is unlikely to be significant in terms of statistical value. So, maybe it is better if we just also level this x axis and this y axis, because then that will make it clear what this x is we are looking at. So, log2-fold change, we can simply write and y lab equals minus log 10 p dot adjusted.

So, because we have taken the adjusted p value right, we can also give a main right, which is, let us say, a wall kind of plot, all right. We can change the font size; we talked about this in the last class. So, simply saying, all right, and this looks a bit nicer, right? We can also put the thresholds for this p adjusted right because if you set, let us say, FDR at 0,

1. So, we are looking at minus log 10 p adjusted to be 1, and maybe we want to put this cut-off and show like which genes are above this cut-off right in because when you transform this p at p-value by minus log 10 right, So, a higher value of minus log 10 p adjusted means more significant results, right? So, we can set the cutoff now. So, this will be using this command, right? So, every line because this is a horizontal line, we are setting a cutoff for minus log 10 value.

So, this is a horizontal line. So, we can use h equals 1, ok? We want this in different colors, perhaps, and also maybe a dotted line, ok? So, this is something we can achieve by just saying color equals 2 ok, and here is the cutoff that you get ok. Now, it might look a bit nicer if you could try to make these points with different colors.

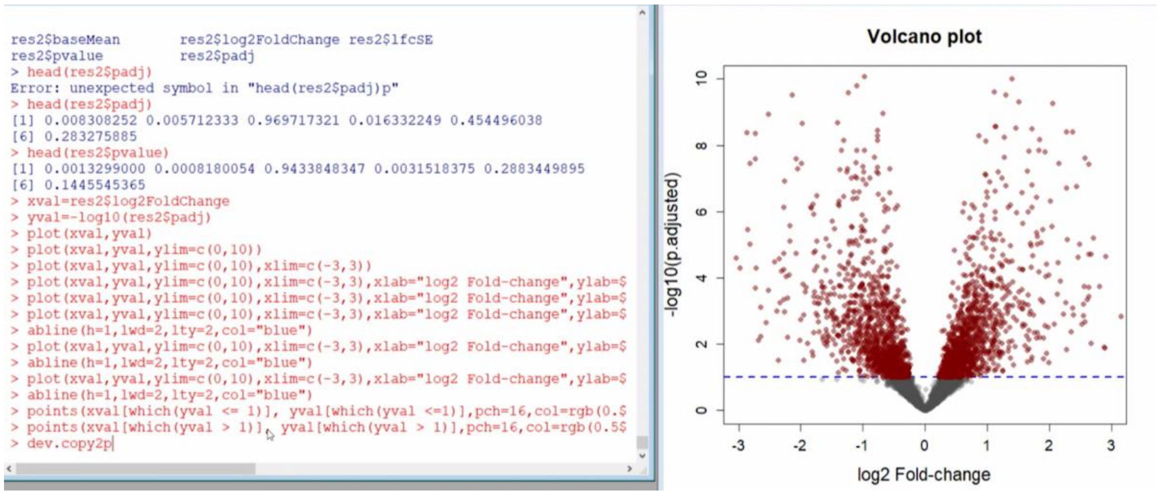
So, for example, some of the points may be the ones that are not significant; maybe they are gray. So, the points that are significant are in under the color, let us say red or something. So, to do that first, we can try something like this, which is h equals 16, right? So, this will change the open circles to fill circles; we can say color equals to RGB, right? So, as we have talked about before, we are setting this RGB scale right, and this looks a bit nicer than before, ok?

And it would be even nicer if we could get different colors for them, and this is something we will do now, ok? So, for doing what I will do, I will take a very simple kind of step, ok? We can let us say first that I will color these points white, ok? Why am I doing that? You will realize in a moment, ok, and then on top of this, ok, in R, you can actually add points, ok, and you can keep on adding points in different colors, ok.

So, that would be great. So, first, let us say, Let us add this line. Ok, I will add this line. Now this is the cutoff. So, for adding points, is the command, okay? So, x which x values sorry y values are less than equals to 1 ok. So, I am plotting only those points now that show this, for which the y values are less than or equal to 1. This will give me the points that are below this line, OK, and similarly, for the y value, we will choose only the ones that are below 1, OK.

PCH equals the 16th right. Now, we want the colors; we do not have to add any axis levels, etcetera, because these are already there and then this color equals to RGB. The last one is for transparency, okay? So, you see, we have added these points that are not significant, and these are in gray; they appear black because there are so many points in there. What you want to do is now add these points that are on top, but in a different color, maybe red or something.

So, let us go ahead, okay? So, we will simply change the condition and use the same command. So, points right, we are adding points to the same plot; we are not creating a new plot, ok? If you create a new plot, these points will be gone, and you have to also add the axis, etcetera. The only thing we need to change now is the color. So, maybe we can add this color right here in red and let us see how it looks.



```
res2$baseMean        res2$log2FoldChange res2$lfcSE
res2$pvalue          res2$padj
> head(res2$padj)
Error: unexpected symbol in "head(res2$padj)p"
> head(res2$padj)
[1] 0.008308252 0.005712333 0.969717321 0.016332249 0.454496038
[6] 0.283275885
> head(res2$pvalue)
[1] 0.0013299000 0.0008180054 0.9433848347 0.0031518375 0.2883449895
[6] 0.1445545365
> xval=res2$log2FoldChange
> yval=-log10(res2$padj)
> plot(xval,yval)
> plot(xval,yval,ylim=c(0,10))
> plot(xval,yval,ylim=c(0,10),xlim=c(-3,3))
> plot(xval,yval,ylim=c(0,10),xlim=c(-3,3),xlab="log2 Fold-change",ylab=$
> plot(xval,yval,ylim=c(0,10),xlim=c(-3,3),xlab="log2 Fold-change",ylab=$
> plot(xval,yval,ylim=c(0,10),xlim=c(-3,3),xlab="log2 Fold-change",ylab=$
> abline(h=1,lwd=2,lty=2,col="blue")
> plot(xval,yval,ylim=c(0,10),xlim=c(-3,3),xlab="log2 Fold-change",ylab=$
> abline(h=1,lwd=2,lty=2,col="blue")
> plot(xval,yval,ylim=c(0,10),xlim=c(-3,3),xlab="log2 Fold-change",ylab=$
> abline(h=1,lwd=2,lty=2,col="blue")
> points(xval[which(yval <= 1)], yval[which(yval <=1)],pch=16,col=rgb(0.$
> points(xval[which(yval > 1)], yval[which(yval > 1)],pch=16,col=rgb(0.5$
> dev.copy2p
```

So, I have just changed the colors here, as you can see. So, this is for red. So, I have said okay, red intensity 0.5, and then also transparency 0.5, ok, and this is what you get right.

You have seen this different color in the theory class, right? So, I have shown you some of these figures, and they look something like this, ok? So, you have these points in different colors, and you can then easily identify which genes show a significant difference in expression in these two different samples. Now, in some cases, when you have a lot of genes, you might want to color them based on their lock-full change as well. So, maybe you want to set up a cutoff of plus 1 and minus 1, and you can change this color again, ok?

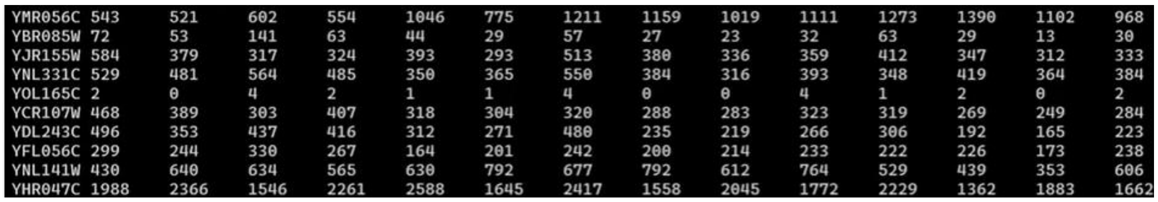
So, again, you can do this in the same way. I am not going to do that, but you can do this in the same way that we have done, ok? The only thing you need are more conditions here, right? So, instead of just saying which y value is greater than 1, you have to also add where lock full change is or whether the x value is greater than 1 or minus 1 right. So, that kind of condition you have to set, and you can get different colors, ok? And you can, of course, save this right. We have talked about how you can save this in R in PDF format, and we prefer the PDF format because of its resolution.

So, it will maintain the resolution; otherwise, if you export in JPEG or BMP, you would lose resolution. So, you can simply save this using this step: dot copy to PDF. So, this output will be copied and then switched off. So, this part is done right. The next thing you can do is make the heat maps, ok? So, for heat maps, we can use this package called G-Plots. ok I have installed this again and it is very easy to again install you can simply if you send a bi-conductor package right you can simply go to Bioconductor page and use this Biocmanager ok.

And in this package, we will use this heat map dot 2 function. OK, this is a function that we will use; it has a lot of options, and this usage is given right. It will have a lot of options, but do not worry about this; it will start with a very simple thing from our data set. So, again, I have prepared these commands, which we will use for generating this heat map, because I want to show this in a very short period of time. So, we will load this package library gplots. Sorry, there is a spelling mistake. So, here is a gplots package we have loaded, and now we want to use this heat map function, but for the heat map function, right?

So, the first thing you require is for this x to be formatted properly. So, what is this x? x is this matrix of these values that we have to plot. So, in our data, what we have is what I have done right. So, for the heat map, as you have seen, we want to look at different samples. So, what I have done is take this count data, and I have created a file that you can use for doing this heat map analysis.

So, I have created this heat map example data. As you can see, by taking only a few genes and their expression values, this could be a raw count or a normalized count. The process of heat map generation is the same. So, we will load this heat map example data in R, and then we will use this heat map dot 2 function to actually do the heat map to generate the heat map. So, let us load this quickly, ok? So, here it is, ok.



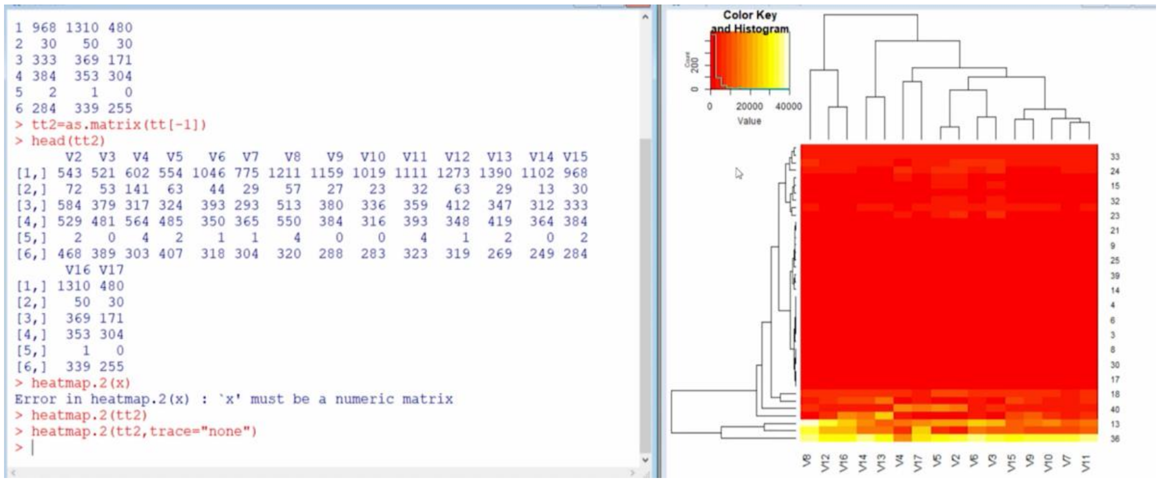
```
YMR056C 543     521     602     554     1046    775     1211    1159    1019    1111    1273    1390    1102    968
YBR085W 72      53      141     63      44      29      57      27      23      32      63      29      13      30
YJR155W 584     379     317     324     393     293     513     380     336     359     412     347     312     333
YNL331C 529     481     564     485     350     365     550     384     316     393     348     419     364     384
YOL165C 2       0       4       2       1       1       4       0       0       4       1       2       0       2
YCR107W 468     389     303     407     318     304     320     288     283     323     319     269     249     284
YDL243C 496     353     437     416     312     271     480     235     219     266     306     192     165     223
YFL056C 299     244     330     267     164     201     242     200     214     233     222     226     173     238
YNL141W 430     640     634     565     630     792     677     792     612     764     529     439     353     606
YHR047C 1988    2366    1546    2261    2588    1645    2417    1558    2045    1772    2229    1362    1883    1662
```

As you notice, there is no header. So, I have not given the header. So, the default option is false, and if there is no header, R will add its own header, v 1, v 2, etcetera. Now, what I need to do is convert this to a matrix, and we also need to get rid of this first row, or the gene names, because the heatmap 2 function expects only a numerical matrix and only numeric data. So, to do that, we will do this conversion.

So, as a dot matrix, So, we are getting this data as matrix dt minus 1. So, we are removing the first column. So, minus 1 means we are removing this first column here because this is the gene name. We do not need this or heatmap2 cannot use this information, and we have generated this new data. So, if you can check this, you will see only these values are now remaining OK. And since this is a matrix, we can now use the heat map function, and we can simply say dot x.

So, it is still giving sorry, not x, yeah, tt 2 ok. So, here it is: we got the heat map, ok, and in the heat map, what you have along these rows are the genes, and along the columns are the samples ok, and you have some sort of clustering that is done here, ok, both on the samples as well as on the genes. If you do not want these clusters right, or if you do not want this clustering in the higher case of clustering, you can switch them off, and you have the option to switch it off along one axis. You can say, Do not cluster along these gene names or do not cluster along the sample names. So, sometimes we do not prefer to cluster along the sample names because we do not want to change the order of the samples. So, that is something you can decide on; you have that option, ok?

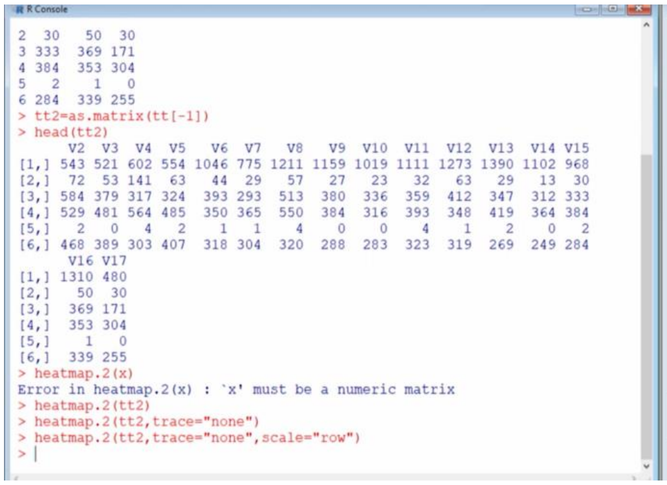
So, what I will do is make it a bit nicer. One of the things you probably see is some lines that are going through this heat map. So, first, let us remove those, and you can see the scale here on top right. So, red means negative or is close to 0, and yellow means positive numbers, like a large positive number. Now, in heat mapping, when you are generating heat maps, one thing you would have to be very careful about is that you are working with genes that show a very high degree of difference in their expression pattern, and if you are using those genes for clustering, the ones with very high expression will dominate the clustering process. So, to remove that effect or minimize that effect, what we do is do something called scaling.



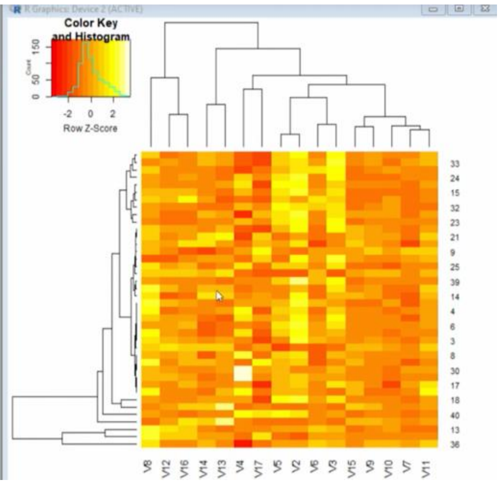
```
1 968 1310 480
2  30   50  30
3 333  369 171
4 384  353 304
5   2    1   0
6 284  339 255
> tt2=as.matrix(tt[-1])
> head(tt2)
      V2  V3  V4  V5   V6  V7   V8   V9  V10  V11  V12  V13  V14 V15
[1,] 543 521 602 554 1046 775 1211 1159 1019 1111 1273 1390 1102 968
[2,]  72  53 141  63   44  29   57   27   23   32   63   29   13  30
[3,] 584 379 317 324  393 293  513  380  336  359  412  347  312 333
[4,] 529 481 564 485  350 365  550  384  316  393  348  419  364 384
[5,]   2   0   4   2    1   1    4    0    0    4    1    2    0   2
[6,] 468 389 303 407  318 304  320  288  283  323  319  269  249 284
      V16 V17
[1,] 1310 480
[2,]   50  30
[3,]  369 171
[4,]  353 304
[5,]    1   0
[6,]  339 255
> heatmap.2(x)
Error in heatmap.2(x) : `x' must be a numeric matrix
> heatmap.2(tt2)
> heatmap.2(tt2,trace="none")
> |
```

So, scaling will bring all the gene expression across samples to the same level. So, all these genes will get a similar weight in the clustering process, ok? So, by clustering, we just want to see some of the expression of genes that are very similar to each other across these samples. So, what I will do is use this scale function, and as you can see now this looks much nicer. We have this transformed because we have scaled, and you can see this Z score because this is after the scaling of the new score that has been calculated, and this is for the different genes.

So, we can now compare these expression patterns across these different samples. One of the last things I want to do is switch off the clustering for the samples because, as I said, we do not want this. So, we can actually switch this off using this cluster, right? So, dendogram we do not want, I think, and then we can also switch off the clustering somewhere here we have to look at again. So, there would be an option to actually switch off this clustering.



```
2  30   50  30
3 333  369 171
4 384  353 304
5   2    1   0
6 284  339 255
> tt2=as.matrix(tt[-1])
> head(tt2)
      V2  V3  V4  V5   V6  V7   V8   V9  V10  V11  V12  V13  V14 V15
[1,] 543 521 602 554 1046 775 1211 1159 1019 1111 1273 1390 1102 968
[2,]  72  53 141  63   44  29   57   27   23   32   63   29   13  30
[3,] 584 379 317 324  393 293  513  380  336  359  412  347  312 333
[4,] 529 481 564 485  350 365  550  384  316  393  348  419  364 384
[5,]   2   0   4   2    1   1    4    0    0    4    1    2    0   2
[6,] 468 389 303 407  318 304  320  288  283  323  319  269  249 284
      V16 V17
[1,] 1310 480
[2,]   50  30
[3,]  369 171
[4,]  353 304
[5,]    1   0
[6,]  339 255
> heatmap.2(x)
Error in heatmap.2(x) : `x' must be a numeric matrix
> heatmap.2(tt2)
> heatmap.2(tt2,trace="none")
> heatmap.2(tt2,trace="none",scale="row")
> |
```



So, if you do not get this reordering of the sample, okay. So, this is again, maybe I am not able to find this right now, but you have this option where you can actually do this for your sample, alright? So, let us now move on to the next part, which is the functional enrichment analysis. So, we have now seen how we can generate this heat map right, how we can generate this fall kind of plot, and of course, there are other options and other visualizations that you can make and explore, but we will not be able to cover all those in the limited time that we have. So, the next part is the functional enrichment analysis, which I will explain very briefly and show you how we can do that using a package called gprofiler2. So, I will switch off or move this window right and clear the screen, ok?

So, let us see the package the gprofiler package, here again. This is a package that you can download very easily or install using this R console, and you have the manuals, etcetera. All the details are here, and here is the web page you can see right here. All the information is available, and in details, ok? So, the main function that we use is this GOST. So, this is the enrichment, and all the instructions are given. You can see the installation instructions for the library you have to load, and the function that you will have to use is this one. So, this is the result that we get from the function that we use: GOST the query list. This will be the list of genes you can provide, ok? You can simply mention the organism's short name, right?

So, this is H. sapiens. This is the human data. In the case of working with Saccharomyces cerevisiae data, you can simply say S. cerevisiae. So, these name-naming conventions are also given on another page, which we will see in a moment. Then you also have these options for different attributes that you can use; for example, you can use an ordered query. So, this is false, which means whether you are sorting this list according to let us say the full change or something, if that is the case, you can say this is true. Whether it is multi-query right you can also have multiple lists that you can give, and we will see them down there.

If you have multiple lists, you can also exclude some of these annotations, some of which are in silico annotations. So maybe they are not very reliable. So, you may not want to use those annotations. You also have this measure under-representation, which means whether you want to measure word representation or under representation, which you can specify using this right. You can then choose the threshold p-value that will use the correction method. This is a multiple-hypothesis testing correction method.

Here, the default is a method for this package that is designed by this package; you can also change it to FDR. And then there are other options you can see if, right again, these are also explained on this page. So, what we will do is take a list of genes, and we will try to do this run and see what we get. Again, this is all explained in very detail here, and for multiple queries, you also have these options you can find somewhere down there. It is also written again, so you can see this query in different query lists. So, different lists can also have this, and you can say multi-query is also true.

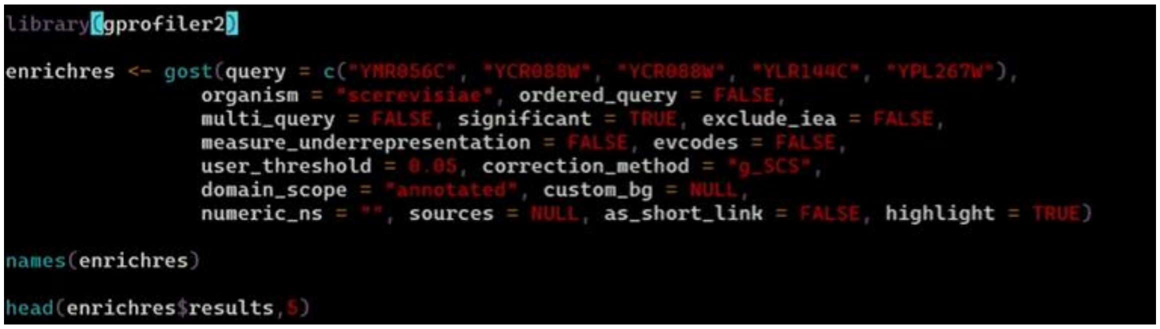
Now, coming to the organism's name, right? So, if you are working with a specific organism, So, let us say you are working with Saccharomyces cerevisiae, which we are working with. We can simply search, and this is the data that is already present, and you have to use this certificate, ok? So, this list is also part of the gprofiler package on the webpage, and you can find this list and then for different organisms, which are given here. So, what we will do is we will run this using this G O S T command from a with a list of genes right random list of genes that we find ok.

In reality, what will we do once we have identified these differentially expressed genes? Let us say we find 100 or 200 genes that are differentially expressed and we have filtered them in all possible ways. Then we want to do this functional enrichment analysis. And so, what we will do is create this list of these differential expressions, give it to this program, and then identify the function classes that are enriched in this process. So, what we will do then is let us go to the R console, ok, and I have already installed this g profiler 2. Of course, it is giving me an error that is giving me a warning, not an error. So, that is ok; it is just saying that perhaps we need to update this R version because this is not the latest version that I am running, but it is ok; it will run.

And then what we will do is have a look at the list, okay? So, the list of genes that we got, and we can perhaps choose at least some of these genes, and we can create a list and then give that list for this analysis, So, I am choosing this at random because we do not have time to find out which genes we choose because sometimes that requires a lot of careful consideration and maybe identifying those genes that you think are important right after those filtering, lock pull change, etcetera, etcetera. So, once you have created this list, you can, once you have the data, create this list very easily and simply load that list in R rather than writing or creating it like this. You can simply load it, and then you can give it or give it that list as an input to this program, ok? So, what I will do is just copy some of these genes here in names you can see, and perhaps we can send this to this program, ok?

So, here we are, ok? I think this should be ok, right? We have given five genes, and let us now run the analysis, ok? So, how do you run this analysis, okay? So, again, we have this code. So, just to refer, right? So, the results that we will get are the enriched results. So, come to G O S T right, then query equals to list, right? In case you are loading this, you just simply give that variable name where you have loaded your data and simply specify here.

And then we have to say the organism's name, right? So, this is required because the program will then search for these gene names and then associate them with specific functional classes. So, or maybe K classes, right? So, here are G-O classes, right? So, this information is required because then the program can find these genes and associate them with the functional classes. And in the second part, it will also look at the global list of genes and their functional associations.



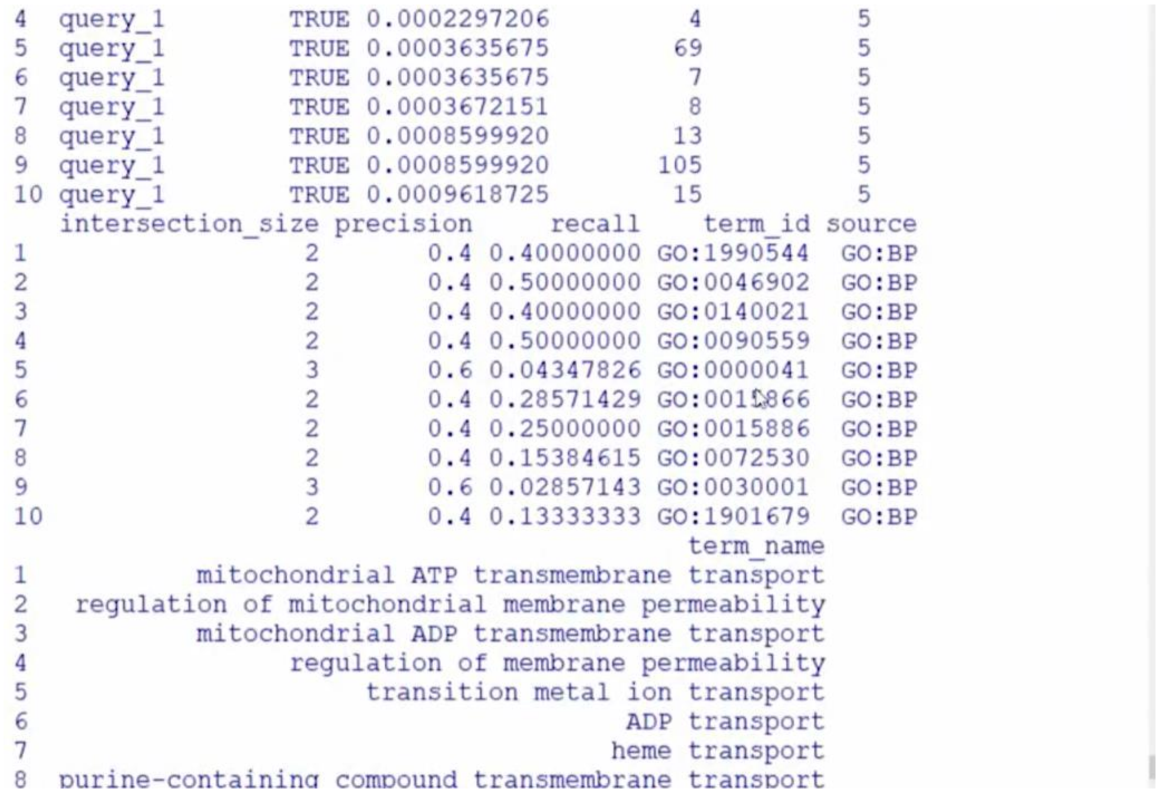
```
library(gprofiler2)

enrichres <- gost(query = c("YMR056C", "YCR088W", "YCR088W", "YLR144C", "YPL267W"),
                organism = "scerevisiae", ordered_query = FALSE,
                multi_query = FALSE, significant = TRUE, exclude_iea = FALSE,
                measure_underrepresentation = FALSE, evcodes = FALSE,
                user_threshold = 0.05, correction_method = "g_SCS",
                domain_scope = "annotated", custom_bg = NULL,
                numeric_ns = "", sources = NULL, as_short_link = FALSE, highlight = TRUE)

names(enrichres)

head(enrichres$results,5)
```

So, if you remember the hypergeometric test, you remember right, we have a global list and the sample OK. So, the sample in our case is the list of differentially expressed genes. So, the rest of the things we really do not need to mention are right. So, unless you want to change them, they will be taken as default values. And the only thing we can perhaps do is look at this correction method, right?



```
4  query_1         TRUE 0.0002297206         4          5
5  query_1         TRUE 0.0003635675        69          5
6  query_1         TRUE 0.0003635675         7          5
7  query_1         TRUE 0.0003672151         8          5
8  query_1         TRUE 0.0008599920        13          5
9  query_1         TRUE 0.0008599920       105          5
10 query_1         TRUE 0.0009618725        15          5
   intersection_size precision     recall    term_id source
1                  2       0.4 0.40000000 GO:1990544  GO:BP
2                  2       0.4 0.50000000 GO:0046902  GO:BP
3                  2       0.4 0.40000000 GO:0140021  GO:BP
4                  2       0.4 0.50000000 GO:0090559  GO:BP
5                  3       0.6 0.04347826 GO:0000041  GO:BP
6                  2       0.4 0.28571429 GO:0015866  GO:BP
7                  2       0.4 0.25000000 GO:0015886  GO:BP
8                  2       0.4 0.15384615 GO:0072530  GO:BP
9                  3       0.6 0.02857143 GO:0030001  GO:BP
10                 2       0.4 0.13333333 GO:1901679  GO:BP
                                             term_name
1            mitochondrial ATP transmembrane transport
2   regulation of mitochondrial membrane permeability
3            mitochondrial ADP transmembrane transport
4                  regulation of membrane permeability
5                        transition metal ion transport
6                                        ADP transport
7                                      heme transport
8  purine-containing compound transmembrane transport
```

So, maybe we can change that correction method to FDR. Again, the program mentions, right, what are the viable options? You can also give a bonferroni correction, ok? And we have got this now, okay? What we want is perhaps okay. So, enriched results are the names of enriched results. So, as you can see, we have found the variable names that are present. So, from the enriched results, we have this query: this is the least p-value, right, and term ID term name, which are given OK.

So, what I will do is then have a look at the first few lines. Let us say first, OK. And you can see where this part of query 1 is significant, whether it is significant or not. There is some p-value, right, and then you have the term names. So, you have the geo-id right. So, you have this term ID; this is the geo ID, right? The source is GO-BP, the biological process. We have again talked about these sub-anthologies. Then you have the term name, right? These are the functions that are associated with these genes in this query, ok?

So, we can now expand this right, of course; you can take bigger lists, and we can identify the functional classes again according to the biological process, and so on. There are other tools as well that you can explore, but the idea is very similar. You can have these hypergeometric test-based tools, or you can go with the GACLI overrepresentation analysis. So, that is it for the hands-on RNA sequencing data set. So, we have covered a lot. Actually, we have started with the data mapping, where we actually map using HISAT2. And we have done the raw count raw counting right raw count data we got after using htseq-count.

And in this hands-on we actually process that data through all the steps. So, from bias correction to normalization to differential expression analysis, and finally, function enrichment analysis, So, in the next class, we will be talking about genome assembly and how we can use NGIS data for the assembly process. What are the algorithms that are available for assembly, and which one is the preferred one? And finally, we will talk about the study of epigenetic modifications using next-generation sequencing technologies. Thank you very much.