Next Generation Sequencing Technologies: Data Analysis and Applications

RNA-seq data analysis – Hands-on 2

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 Applications. In the last few classes, we have discussed the full analysis pipeline for transcriptomic data. So we started with read mapping, and we have talked about the specific requirements for read mapping in the case of transcriptomic datasets because we will have to deal with exon and intron boundaries. So we can have splice variants, so we need to have sequence-splice variant-aware alignment methods. We specifically talked about a few tools for that. We then discussed the quantification part, followed by a discussion on the biases in RNA sequencing data.

So there are different types of bias that we might encounter when we are processing these datasets, and we also talked about some of the methods that we can use to actually remove, eliminate, or minimize these biases. Following that, we talked about the normalization method. So if we are doing comparisons between different samples obtained from different conditions or different sample types, for example, disease versus LD, we want to do a normalization before we can compare the expression levels of genes between them. And then finally, we perform the differential expression analysis.

So we talked about the software tools that we use to do this differential gene expression analysis, specifically DESeq2 and EHR. And finally, we talked about the interpretation of the data and how we actually visualize the results that we get, and we then talked about the functional enrichment analysis. This is an interpretation of the results that come out of differential gene expression analysis. So what we are going to do now is implement all those. So what we will do is go through a hands-on experience where we will actually analyze these real datasets.

So we will take one dataset, and we will analyze this dataset through all these steps. So that is the agenda, which is what we will start with this class, and subsequently, a few more

classes will be devoted to this hands-on. So today's agenda will start talking about a programming language called R. So this is a platform where we can do a lot of statistical analysis. A lot of the packages are built on this R platform, and as we have talked about when you are talking about the analysis part of the theory, we mentioned there are packages that are implemented in R, and we will see that there are so many tools that you can use for doing this transcriptomic analysis. So what is the goal of this hands-on for the next few classes? To perform differential gene expression analysis on a real dataset and identify genes that show differences in expression between samples obtained from two different conditions.

So this is just a specific example of course, you can take any example you like. I will also show you where you can get such examples or download such examples, and you can try your codes or try your hands at those. So what we will do is start with this platform, R, today. So this is where we will do all the analysis. So briefly, we have analyzed the transcriptome data.

I have shown you how to actually do the mapping with HiSAT-2 with a small example dataset, and we obtain the raw counts using htseq-count. So hopefully you remember that and the rest, and from that point on, we can now take this forward on the R platform. So HiSAT-2 runs on the terminal of Linux, but from there on, right also the htseq-count we have used in the terminal. From there on, we can now use the R platform to do all the analysis that we would like to do. So just to remind you of all the steps or to summarize the steps

So the first step is that we have to set up the system in R. So you might be familiar with R or might not. So I will show you right away, at least very briefly, how to set up this system where you will get all the tools, etc. How do you install these packages that are available in R, and how can we use these packages? Where do you get the manuals, etc.? All the details

So this is the first part that we have to get the system running. Then we take the count data

that we have already generated, and you can get this kind of count data or count data from different sites. I will again show you where you can get those. Then the first part of this analysis is the preliminary data analysis. So you remember we talked about two different methods?

One was distance-based clustering, and the second was principal component analysis. So these preliminary data analysis methods allow us to understand what would be the best comparisons to make. Whether some samples are similar to each other or whether some samples are different And it can identify the best groups for comparison. So we will go through this preliminary data analysis with our data set, and then we will also do the bias correction part.

So we have the count data, so we will do the bias correction again using an R package that can allow us to do some sort of bias correction. And then we will actually go to the differential gene expression analysis, which is using DEseq2. So we will try two different methods. We will do DEseq2 first, and then we will also try this differential gene expression analysis through the RUVSeq method. So where will we utilize these ERCC spike-ins to actually normalize the data?

So hopefully you remember what these spike-ins are. We have talked about this and how we can utilize those spike-ins to normalize the data. So we will go through these two methods. And then we will have the results and the visualization part. So once we have generated the results from differential gene expression analysis, we will talk about how to visualize.

So we will see what kind of code we can use. So I have shown you some of the visualizations, but in this hands-on, we will write these codes to generate those visualizations. For example, the MA plot or the heat map So those we will do here, hands-on. And then, finally, I will also talk about the functional enrichment analysis.

There are different tools that you can use but we will use one tool to actually do the

enrichment analysis on our dataset. So this is the agenda. So let us start with the first part, which is the setting of the system. So we will first have a brief introduction to R, how to set up R, and very basic commands in R, followed by setting up the system for the data analysis. So and then, if time permits, we will also talk about the data loading, and we can load the count data and look at some of the characteristics of the data.

So let us now move on to the system. So I will start with the R platform. So here it is. So this is the website where you can actually get this tool. You can download R for your system.

So R is a free tool that is actually used for computing and also generating very nice graphics. And it can run on all three platforms. So it can run on Unix-based platforms, Mac, as well as Windows). So if you are running Windows, you can simply choose the right version. So if you click on download R, you have a lot of these options for different meters from which you can download, or you can simply go here and download the newest version.

So here is the.tar.gz file we have talked about. These are the compressions and you can download and decompress them. So I have installed this R 4.

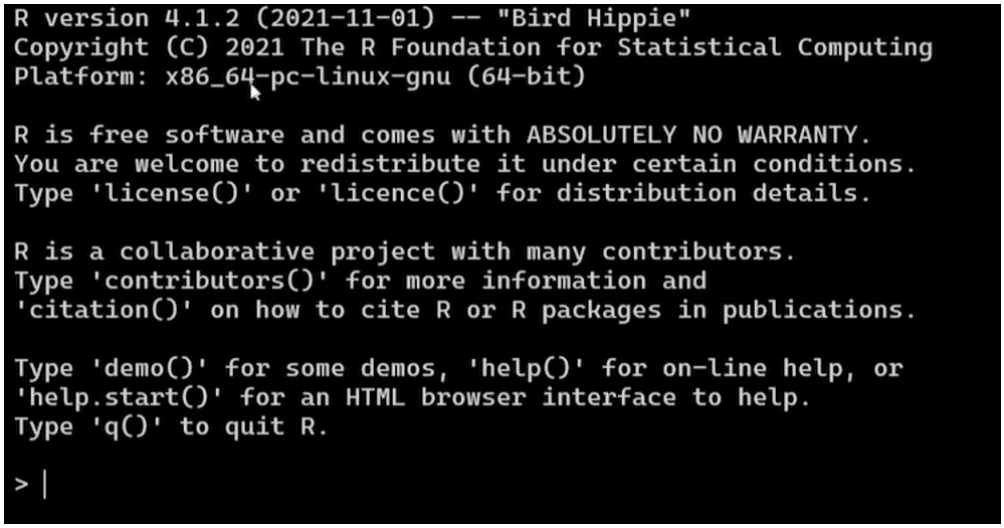
3.0. So as you can see, 4.3.1 is the latest one, but what I have in the system is 4.3.0. So we will utilize this; you can use 4.

3.1 There won't be too much difference in terms of the use of packages and the updates. So let's look at this. So if you want to install, you can download this and install it on your system. Again, you can use different systems, whether you need a Windows system or not. For the Windows release you can actually go here and see all these meters, etc.

all these things. All right, so I'll show you that you can probably go here and again download R for Linux. So if you are running this Linux system, you can download it from here. If you are running macOS, you can also download from here, and for Windows, you

can simply click, and it will download all the tools for you. So some of the tools are required with this base. So you can see that this is done when you are installing R for the first time, and you also need to download this R tool. And some of the CRAN packages you probably want to download as well.

So if you do not download these packages for R, we can install them later on from inside R, and I'll show you how we can do that. For the Linux system, if you are running a Linux-based system, I can show you in WSL right the Windows Subsystem for Linux, and here you can also install by simply saying sudo. If you have sudo permission, of course, install R base. So in my system, it's already installed, so in this WSL system, I won't run it. So in the Linux system, if I just want to run R, I'll just type capital R, and then I'll enter, and you can see here that the version is different, but you can see this version here.



This is a slightly older version that is installed here, and if you can write your codes you can generate all sorts of plots, whatever you want, by simply typing here. Okay, so you can write your commands. For example, you can say equals to 3 or something. You can run all sorts of commands here, and you can generate all the graphs or do statistical analysis whatever you want inside this terminal. If you want to quit right now, if you come, if you want to come out of R from here, you can simply say q brackets, right, and then you can say whether you want to save the workspace image or not, usually no, so we are back to the terminal again. Okay, so it's that simple, but if you have a very long R code to run, let's say you have generated a very long code of R, then you probably do not want to run in this

way. You have to copy and paste all those things, so that's very cumbersome, so you can run it as a script here. okay, so you can simply run, let's say, R and then minus f and then script name. Okay, so this is also possible will not get there because we are not going to write very complicated scripts. I'll show you small scripts one at a time and we'll run those and execute those and we'll generate some of the results. for this class and demonstration, I'll use the windows version for a very simple reason because if I run this from this window subsystem for Linux all the codes the plots that will be generated that will be saved in files. So I'd have to always save them I will not be able to visualize them here because it's not set up like that so that's why I'll use the R from Windows because then I can see the side by side the plots that are generated. You can also download something called R studio so I also have this installed here okay'so this is the R I have it install this is 4.3.0 and I'll run everything from inside here. I can probably zoom in a bit I can make it a bigger slightly bigger I think and then we can probably see this better and here it is right okay, so this is bigger. So, once I run the course from here the plots will also be generated side by side. So, I can see the plots in this part so that will be very useful right. So I can see and change if I want to change the plots and outcomes that is it there is also something called R studio right this is also very helpful for running R codes because in here you are more aware of your environment. So, what you have is on this side you have the global and environment right so it means which variables are there in the environment right what those values are etc will be given here so maybe if I just write for example okay, maybe later so if I let's say write a equals to 3 you see this is there okay, so a equals to 3 is now in the environment. So, that's in there I can remove this from the environment by just saying RMA remove a so this is gone from time, right? So this is also possible for other types of data structures in R as well, right so I will talk about that later on the right hand corner here. You have these plots that you can see you can also see the packages that are already available in the library and which are installed right of course you can see some of them are ticked right so it means they are also available and installed right and we this is very helpful because then we can see the plots around here right so whatever plot I am generating I will be able to see this here and here we can write the code etc. So, this is the console you can also see a terminal right if you want to use the terminal here you can also use it from here. So, for keeping this simple now we'll use the just R. not R studio once you get more familiar you

can start using the R studio . So, I'll quit this now alright so now going back to R here. So, R can support a lot of different types of data types right so you can have this you can have vectors you can have matrix so different types of data can be used in R, right so to create a vector you can simply assign a vector right so let's say we have a variable name that we say it's vector okay and to assign we usually use this arrow sign right we are assigning these values to the vector okay but you can also use this equal sign okay it's simply we are assigning these values in vector. so you'll see both of these, right? You will see this assignment, okay, as well as the other time. So, a vector is a list that contains objects of same type, right so it can be a string it could be number so I hope you are familiar with the numbers string etc those concepts right so you can have let's say a right character simply right and to see what is there in in vector you simply type vector. So, this is what we see we can also define list right so let's say we say list lsd and least can contain different types of objects right so it can contain numbers as well as the strings or characters right so it's kind of like a mixed type of thing right and you just want to see right what is there say lsd okay now if you want to access these elements right so you can see these numbers right so lsd is one right and two this is the third element is t right but if you want to access these individual elements right you can simply write and it will give you these elements. So, the other one you can write is this and you get this okay you can also define matrix in R right so this is the type of data that will encounter most we'll see we'll work with mostly matrix data because we are talking about count data right in transcriptomic data so round data means we have genes in the rows and have the samples in the mat in the columns right so this will be a matrix okay which would again the dimensions will be determined on the number of genes that you have in the sample and the number of samples right so we'll come to that in a moment when you actually start working with the sieve data. So, you can say matrix or as matrix right so you can have let's say so we have to mention right how we want to distribute this expected comma I think is the data.

```
> vector <- c("A","B","C")
> vector
[1] "A" "B" "C"
> lst <- list("A",1,"D")
> lst
[[1]]
[1] "A"

[[2]]
[1] 1

[[3]]
[1] "D"

> lst[[1]]
[1] "A"
> lst[[3]]
[1] "D"
> mat <- matrix(("A","B","C","D","E","F"),nrow=2,ncol=3)
Error: unexpected ',' in "mat <- matrix(("A","
> mat <- matrix(c("A","B","C","D","E","F"),nrow=2,ncol=3)
> mat
     [,1] [,2] [,3]
[1,] "A"  "C"  "E"
[2,] "B"  "D"  "F"
> mat[1,2]
[1] "C"
```

So, we need to give it give C right so it will say okay this is the value value these are the values that we have in the matrix and then number of rows is 2 so we have 6 elements right so you just have to tell how you want to organize these elements in the matrix right so here I want let's say n row equals to 2 and n column equals to 3 so 3 columns and 2 rows okay so let's now see okay this is how it is organized now you see that we have 3 columns and 2 rows okay now how do you access these elements in this matrix we say 1 comma 2 right so the first number is for the row right and the second number is for column. So, this will be C is this element right so 1 2 will be this element and similarly you can try other ones right so you can say 2 3 this would be the last element here we can also try something like this okay this when I just say 1 comma gap so it will give it will it will mean that we want the full first row. So, that's what we have we are getting here right we are getting the full first row and similarly we have if you let's say one the last column full last column so we can say gap comma 3 so we will get the third column okay the values in the third column which are E and F. So, remember this matrix part because this is very very important this will actually come when we start loading our data there is another concept which is called the data frame which could be combinations of all these data types okay all right so I think this is clear right how we actually get to the different types of data how we actually start working on this. So, what I'll do now is I'll load the data that we have right the raw count

data that we have generated and we will start performing some operations on this data set. So, we'll start loading the data first but let's have a look at the data. So, for that I'll go to the terminal because here I have loaded the data it's here inside this folder so this is the ingest data and so on test and this is the file that actually contains the raw count data. So, we can check this right in the terminal using VI okay and one thing I'll do first hopefully remember the commands in VI colon SC node act so so what you see here on top are the sample names right so you have this s1 s5 s9 so on so we have 16 samples up to s16 okay and along the rows we have the gene names.

	S1	S5	S9	S13	S2	S6	S10	S14	S3	S7	S11	S15	S4	S8	S12
YMR056C	543	521	602	554	1046	775	1211	1159	1019	1111	1273	1390	1102	968	1310
YBR085W	72	53	141	63	44	29	57	27	23	32	63	29	13	30	50
YJR155W	584	379	317	324	393	293	513	380	336	359	412	347	312	333	369
YNL331C	529	481	564	485	350	365	550	384	316	393	348	419	364	384	353
YOL165C	2	0	4	2	1	1	4	0	0	4	1	2	0	2	1
YCR107W	468	389	303	407	318	304	320	288	283	323	319	269	249	284	339
YDL243C	496	353	437	416	312	271	480	235	219	266	306	192	165	223	261
YFL056C	299	244	330	267	164	201	242	200	214	233	222	226	173	238	249
YNL141W	430	640	634	565	630	792	677	792	612	764	529	439	353	606	395
YHR047C	1988	2366	1546	2261	2588	1645	2417	1558	2045	1772	2229	1362	1883	1662	2646
YBL074C	178	149	144	169	131	135	208	152	156	152	149	160	174	170	175
YKL106W	230	238	243	160	224	299	421	359	333	398	452	410	436	372	411
YLR027C	20002	22397	15019	20466	25754	12420	40214	15336	18698	13484	32827	12284	17788	11076	26448
YBR236C	1619	1417	1333	1309	1279	1217	1541	1371	1275	1434	1226	1478	1547	1433	1236
YKL112W	3395	2821	2215	2732	1871	1415	2390	1275	1468	1609	2157	1224	1571	1290	2142
YMR072W	5105	5132	3057	4706	3761	3516	4637	3923	3598	4127	3415	4100	4128	3468	3726
YJR108W	55	59	42	48	39	43	45	39	26	57	40	40	41	50	50
YCR088W	8531	7664	6257	8097	6169	5220	6477	5725	5942	6190	5819	6096	6420	6073	5745
YOR239W	4174	3699	1785	3659	2788	2618	2991	3070	2878	3053	2523	3020	3138	2901	2543
YNR033W	2888	2627	2788	2989	1923	1849	2228	2058	1989	2110	1878	2108	2053	1888	2001
YMR289W	763	720	557	670	652	677	892	801	780	887	724	975	943	833	779
YER045C	626	623	583	495	403	298	543	369	366	431	462	279	318	376	431
YGR037C	5423	5479	3685	4675	2468	2382	2119	2303	2030	2273	1839	2219	1933	1848	1760
YNR016C	7997	7523	5468	6499	2749	2907	4095	1990	2405	3226	3073	2244	2591	2278	3501

So, you can see the gene names we can check how many columns or how many rows we have so if you if I just go to the end of the file right, I get six thousand eight hundred six, okay? So in total, we have six thousand eight hundred six values for genes, etc., and then for each sample, you have the count value right, so you can see this for this gene. YMR 0 5 6 C right we have these count values across these samples now these samples are organized in certain ways which will come about later on when you actually go to the differential expression and this is that's when the whole thing will be more relevant right so for how these samples are organized which are the replicates etc that information we'll talk about later on. So, as you can see this is a matrix right so we will like to load this in R and then we'll start working with that right so to load this in the R right so we will again go here right one of the things we'd have to do is we have to give the path to the file right so we'll only then we can load the pipe right so to get the current working directory where we are right inside our we can use the command called getwd. So, get W T is the current working directory right so working directory so it gives me the path where we are right

now okay now to change this to the directory right where you want to go right we have to say set working directory or setwd okay now in your system this path will be different of course right so you can then go around and go to the appropriate path where you are getting these files. So, you say setwd I know where it is right so it's on if we go up right so it's again you can see that this is like Linux commands we are using right so dot dot slash means we are going back up right one directory up then we are going in desktop where I have the data right and in hands-on I showed you right where this data is and that's it right so this will change the data it's like CD right in terminal we use this CD here we just use the setwd and within double quotes right the whole path should be within double quotes okay this is the format in R okay and we can again check whether we have changed the directory and we are in the right directory we are right so so the next thing we want to do is we want to load the data. So, to load that we can load a matrix as a table and the command that we have in R is called read dot table. So, the first thing is we need to define a variable where do we want to load the data right so let's say we say call it as tab right on the table short for table right and then read dot table and we have to now give the file name. So, which is this one okay and again this file name should be within double quotes The other parameter that we need to specify often is whether we have a header line in the file or not.

```
> getwd()
[1] "C:/Users/Dhar/Documents"
> setwd("../Desktop/NGS_Data_Analysis_HandsOn2/Test/")
> getwd()
[1] "C:/Users/Dhar/Desktop/NGS_Data_Analysis_HandsOn2/Test"
> tab <- read.table("RUN3_all_S1-S16_analysis.txt",header=TRUE)
```

Okay, so header equals true, right? This is a logical variable, so header equals to true, which means there is a header line on top of the file, okay, and we need to consider that okay, so you can try this without header and with header and see what the outcome is because this will impact how we actually call the data. So, we have a header line right so I showed you we have this s1 s5 s9 etc those names right in the first line so that's the header line so that's why I mentioned header equals to true. So, this is what we get now header equals to true and we can now actually simply type tab so it will print the whole data here okay this is something we do not want what you want to do is we want to check let's say first few lines.

```
> tab <- read.table("RUN3_all_S1-S16_analysis.txt",header=TRUE)
> head(tab)
         S1  S5  S9 S13   S2  S6  S10  S14   S3   S7  S11  S15   S4  S8
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
         S12 S16
YMR056C 1310 480
YBR085W   50  30
YJR155W  369 171
YNL331C  353 304
YOL165C    1   0
YCR107W  339 255
```

So, we can simply say head tab so head will again show you first few lines you can see this is very similar to the command lines right so the terminal what we have done before right so you can simply say head tab so it will show you this part of this matrix so you can see these first six teams here along with the header that you have. So, the next thing right so you can also try to see the last few lines again we can use the tail again this very similar to the command line you can also specify number of lines you want to see you can say I want to see first three lines.So, it's a head tab three so it will show you only first three things If you say head tab 10 right, you can see the first ten lines.

```
> head(tab,10)
         S1   S5   S9  S13   S2   S6  S10  S14   S3   S7  S11  S15   S4
YMR056C 543  521  602  554 1046  775 1211 1159 1019 1111 1273 1390 1102
YBR085W  72   53  141   63   44   29   57   27   23   32   63   29   13
YJR155W 584  379  317  324  393  293  513  380  336  359  412  347  312
YNL331C 529  481  564  485  350  365  550  384  316  393  348  419  364
YOL165C   2    0    4    2    1    1    4    0    0    4    1    2    0
YCR107W 468  389  303  407  318  304  320  288  283  323  319  269  249
YDL243C 496  353  437  416  312  271  480  235  219  266  306  192  165
YFL056C 299  244  330  267  164  201  242  200  214  233  222  226  173
YNL141W 430  640  634  565  630  792  677  792  612  764  529  439  353
YHR047C 1988 2366 1546 2261 2588 1645 2417 1558 2045 1772 2229 1362 1883
         S8  S12  S16
YMR056C 968 1310  480
YBR085W  30   50   30
YJR155W 333  369  171
YNL331C 384  353  304
YOL165C   2    1    0
YCR107W 284  339  255
YDL243C 223  261  157
YFL056C 238  249  180
YNL141W 606  395  897
YHR047C 1662 2646 1475
```

Okay, so this is how the whole thing would work. Now, let's say you want to see right now. The difference will come if we say we want to see the first row right, so you can see

YMR056C because this is stored as a matrix. This data is right, and you can now access each element. You can access rows as well as columns. Okay, so now the next thing is we can access the columns also. Let's say we want to say tab 5 right, so this is the fifth column. Of course, we get all these six thousand eight hundred five data points right, so maybe you want the fifth column only the head right, so the first five or six values we get here. Okay, now that it turns out in R, you can actually access the columns in a different way as well, and that's actually very convenient, as we will see later on, so I'll just type head again right and look at the column names. So in R when you are dealing with really big data you can actually access the columns by this tab dollar the column names.

So, I'll say tab dollar S5 okay and it will give you the values corresponding to S5 column. So, again I have used head so we get the first six values here and these are the values you can see this is this is what we have under this S5 so what you what is what is really convenient here you don't have to remember right which column contains which sample you can simply call by the sample name or the variable name right and see what the data is or you can do all sorts of analysis on that. So, this is actually really convenient for while you are using R. So, then you can also generate something called summary tab etc okay and it will give you these statistics for all sorts of for all the columns that you have in the data. So, you can get this minimum value at the first quartile you have the median you have the mean the third quarter and the maximum value etc so for each sample is one from S1 to S16 you will get these statistics when you write this summary you can also see that what are these variables that are available using this dollar approach. So, you can simply type tab dollar and it will say S and it will then show you what are these variables or the names that are available with this dollar axis okay the other option that you have is you can also type names okay tab and it will show you these names or the variable the column names that are actually available okay and you can also try row names I think and if you get these names of the genes that are available. So, you can try this column names or simply names and you can also try row names right and this will give you the list of genes that are present in your data. So, I think this is this part is clear so if you do not have your own data right,

I'll show you in the next class where we can actually download this data right and we can try this out on these different things on them. So, if you have your own data definitely try this and try to load it and then you can start working on this so in the next class we'll actually do a bit more on this data set one since we have loaded it now we will actually now we can do a lot of things and I will also show you how to start loading the packages that will be required for our analysis so that's it for this class and thank you.