BIMM143_Project2_R Notebook

Code

Introduction

{5 points for specific, measurable, and clear scientific question}

Scientific Question: What is the mutation in the huntingtin (HTT) gene that leads to the symptoms of Huntington's disease, and what are the consequences of such mutation?

{5 points for background on the protein/gene/species of interest and where the data is sourced from}

Huntington's disease, affecting 1 per 7300 people, is one of the most prevalent monogenic neurological disorder in the world (Fisher, 2014). The cause of this disease involves in a mutation in the Huntingtin (HTT) gene, which lead to an exceptionally long expansion of CAG trinucleotide repeat (MacDonald, 1993). This mutation results in a structural change in the Huntingtin (Htt) protein, a protein which is ubiquitously expressed throughout the body but its major function remains unknown.
Although the specific function of the Htt protein is still a mystery, recent researches have revealed interaction of this protein with other proteins that are involved in the process of tranlsation, such as Prkra, Rps6, and Gnb2l1 (Culver, 2012). Specifically, introducing the mutated Htt protein can increase ribosome stalling, while removing it rescues the speed of translation (Eshraghi, 2021). Other researches have shown that the expression of mutated HTT gene could lead to translation deficit in multiple animal models (Joag, 2019), and can cause differential expression of genes involving in multiple functions such as protein folding and ribosome biogenesis (Tauber, 2011). The question here is, what specific genes are up-regulated or down-regulated which leads to the dysfunction of protein synthesis in subjects with mutated HTT gene.

{5 points for clear, specific, and measurable scientific hypothesis that is in the form of an if-then statement}

Scientific Hypothesis: If there is a mutation such as a polyglutamine repeat expansion in one or both of the alleles of the HTT gene, then there would be up-regulation and down-regulation of various genes and proteins that lead to the impairment of protein synthesis.

{5 points for description of what analyses were done and how the data was downloaded for the project}

A pairwise sequence alignment is done to compare the Huntingtin protein produced by the normal HTT gene and that by the mutated HTT gene in Homo sapiens to see what exactly is the mutation that leads to the expression of the pathological phenotype of Huntington's disease by looking at the parts that fail to align, which will answer the first part of my scientific question. The data used for this analysis are two FASTA files containing protein sequences of the variants of the Huntingtin protein, obtained from the UniProt online database.

After confirming the presence of the mutation in the Huntingtin protein, an RNA-seq analysis is done to find out which genes and proteins are up-regulated or down-regulated in the mutated groups compared to the control group, attempting to answer the second part of my scientific question. The raw count csv file was obtained from

the paper: Mutant Huntingtin stalls ribosomes and represses protein synthesis in a cellular model of Huntington disease. The authors did a study involving three groups, the mHTT homozygous group, the mHTT heterozygous group, and the control group, which is homozygous for normal HTT, and got the raw counts for various genes expressed under these three conditions.

The results from the above analysis are presented by a heatmap, featuring the top differfentially expressed genes from the RNA-seq, and a GO annotation to see the specific functions of these genes, testing the second part of the hypothesis and revealing whether HTT mutation can lead to impairment in protein synthesis.

Loading in Packages

{10 points for definition of each of the packages loaded}

Packages needed for following analysis:
(1) **seqinr**: used for reading in FASTA formatted files needed for pairwise sequence alignment and converting character vectors into strings.
install.packages("seqinr")

(2) **Biostrings**: used for pulling out the scorring matrix for pairwise alignment, performing the pairwise alignment, and displaying the results from the pairwise alignment.
if (!require("BiocManager", quietly = TRUE))
install.packages("BiocManager")
BiocManager::install("Biostrings")

(3) **edgeR**: used for normalizing the count data needed for RNA-seq, filtering out low-count genes,and performing the RNA-seq pipeline.
if (!require("BiocManager", quietly = TRUE))
install.packages("BiocManager")
BiocManager::install("edgeR")

(4) **gplots**: used for plotting the heatmap.
install.packages('gplots')

(5) **RColorBrewer**: used for coloring the heatmap.
install.packages("RColorBrewer")

(6) **org.Mm.eg.db**: used for mapping the gene symbol of top expressed genes to GO ids.
if (!require("BiocManager", quietly = TRUE))
install.packages("BiocManager")
BiocManager::install("org.Mm.eg.db")

(7) **GO.db**: used for mapping GO ids to GO terms for functional annotation.
if (!require("BiocManager", quietly = TRUE))
install.packages("BiocManager")
BiocManager::install("GO.db")

{5 points for correctly loading all of the packages needed and stating anything that needs to be done to load the packages (downloading the packages)}

Hide

```
library(seqinr)
library(Biostrings)
```

```
Loading required package: BiocGenerics

Attaching package: ‘BiocGenerics’

The following objects are masked from ‘package:stats’:

    IQR, mad, sd, var, xtabs

The following objects are masked from ‘package:base’:

    anyDuplicated, append, as.data.frame, basename, cbind,
    colnames, dirname, do.call, duplicated, eval, evalq,
    Filter, Find, get, grep, grepl, intersect, is.unsorted,
    lapply, Map, mapply, match, mget, order, paste, pmax,
    pmax.int, pmin, pmin.int, Position, rank, rbind, Reduce,
    rownames, sapply, setdiff, sort, table, tapply, union,
    unique, unsplit, which.max, which.min

Loading required package: S4Vectors
Loading required package: stats4

Attaching package: ‘S4Vectors’

The following objects are masked from ‘package:base’:

    expand.grid, I, unname

Loading required package: IRanges

Attaching package: ‘IRanges’

The following object is masked from ‘package:grDevices’:

    windows

Loading required package: XVector
Loading required package: GenomeInfoDb

Attaching package: ‘Biostrings’

The following object is masked from ‘package:seqinr’:

    translate

The following object is masked from ‘package:base’:

    strsplit
```

Hide

```
library(edgeR)
```

```
Loading required package: limma

Attaching package: ‘limma’

The following object is masked from ‘package:BiocGenerics’:

    plotMA

The following object is masked from ‘package:seqinr’:

    zscore
```

Hide

```
library(gplots)
```

```

Attaching package: ‘gplots’

The following object is masked from ‘package:IRanges’:

    space

The following object is masked from ‘package:S4Vectors’:

    space

The following object is masked from ‘package:stats’:

    lowess
```

Hide

```
library(RColorBrewer)
library(org.Mm.eg.db)
```

```
Loading required package: AnnotationDbi
Loading required package: Biobase
Welcome to Bioconductor

    Vignettes contain introductory material; view with
    'browseVignettes()'. To cite Bioconductor, see
    'citation("Biobase")', and for packages
    'citation("pkgname")'.
```

Hide

```
library(GO.db)
```

Performing Bioinformatics Analysis

Bioinfo method 1: Pairwise Sequence Alignment

Below is an pairwise sequence alignment comparing two protein sequences: one of which is the normal Huntingtin protein, the other is the mutated Huntingtin protein. These human protein sequences are obtained from the website UniProt as two FASTA files, and are analyzed using the Biostrings package. Pairwise sequence alignment is a way to see how similar two proteins are structural-wise, and could be used to answer the first part of my scientific question, and reveal the possible mutations involved in the Huntingtin protein in Huntington's Diseased patients.

Hide

```
# Read in the FASTA files containing protein sequences of the normal and the mutated Huntingtin
 protein downloaded from UniProt
Htt <- read.fasta("Human_HTT_normal.fasta")
mHtt <- read.fasta("Human_HTT_diseased.fasta")

# Store the protein sequences as vectors
HttSeq <- Htt[[1]]
mHttSeq <- mHtt[[1]]

# Convert the vectors to strings
HttStr <- c2s(HttSeq)
mHttStr <- c2s(mHttSeq)

# Convert the AA characters into uppercase for alignment
HttStr <- toupper(HttStr)
mHttStr <- toupper(mHttStr)

# Load the BLOSUM50 matrix for the Needleman-Wunsch algorithm
data("BLOSUM50")

# Perform a pairwise alignment
globalAlignHtt <- pairwiseAlignment(HttStr, mHttStr, substitutionMatrix = BLOSUM50, gapOpening =
-2, gapExtension = -8, scoreOnly = FALSE)
```

Hide

```
# Write a function to display our alignment results
printPairwiseAlignment <- function(alignment, chunksize=60, returnlist=FALSE)
  {
     # Get the packages required to run this function
     require(Biostrings)

     # Get the alignment for the first sequence
     seq1aln <- pattern(alignment)
     # Get the alignment for the second sequence
     seq2aln <- subject(alignment)

     # Find the number of columns in the alignment
     alnlen  <- nchar(seq1aln)
     starts  <- seq(1, alnlen, by=chunksize)
     n       <- length(starts)
     seq1alnresidues <- 0
     seq2alnresidues <- 0
     for (i in 1:n) {
        chunkseq1aln <- substring(seq1aln, starts[i], starts[i]+chunksize-1)
        chunkseq2aln <- substring(seq2aln, starts[i], starts[i]+chunksize-1)

        # Find out how many gaps there are in chunkseq1aln
        gaps1 <- countPattern("-",chunkseq1aln)
        # Find out how many gaps there are in chunkseq2aln
        gaps2 <- countPattern("-",chunkseq2aln)

        # Calculate how many residues of the first sequence we have printed so far in the alignm
ent
        seq1alnresidues <- seq1alnresidues + chunksize - gaps1
        # Calculate how many residues of the second sequence we have printed so far in the align
ment
        seq2alnresidues <- seq2alnresidues + chunksize - gaps2
        if (returnlist == 'FALSE')
        {
           print(paste(chunkseq1aln,seq1alnresidues))
           print(paste(chunkseq2aln,seq2alnresidues))
           print(paste(' '))
        }
     }
     if (returnlist == 'TRUE')
     {
        vector1 <- s2c(substring(seq1aln, 1, nchar(seq1aln)))
        vector2 <- s2c(substring(seq2aln, 1, nchar(seq2aln)))
        mylist <- list(vector1, vector2)
        return(mylist)
     }
}
```

Hide

```
# Apply the function to our results for visualization
printPairwiseAlignment(globalAlignHtt, 60)
```

```
[1] "MATLEKLMKAFESLKSFQQQQQQQQQQQQQQQQQQQQQ---------------------- 38"
[1] "MATLEKLMKAFESLKSFQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ 60"
[1] " "
[1] "----------------------------------PPPPPPPPPPPQLPQPPPQAQPLLP 63"
[1] "QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQPPPPPPPPPPPQLPQPPPQAQPLLP 120"
[1] " "
[1] "QPQPPPPPPPPPPGPAVAEEPLHRPKKELSATKKDRVNHCLTICENIVAQSVRNSPEFQK 123"
[1] "QPQPPPPPPPPPPGPAVAEEPLHRPKKELSATKKDRVNHCLTICENIVAQSVRNSPEFQK 180"
[1] " "
[1] "LLGIAMELFLLCSDDAESDVRMVADECLNKVIKALMDSNLPRLQLELYKEIKKNGAPRSL 183"
[1] "LLGIAMELFLLCSDDAESDVRMVADECLNKVIKALMDSNLPRLQLELYKEIKKNGAPRSL 240"
[1] " "
[1] "RAALWRFAELAHLVRPQKCRPYLVNLLPCLTRTSKRPEESVQETLAAAVPKIMASFGNFA 243"
[1] "RAALWRFAELAHLVRPQKCRPYLVNLLPCLTRTSKRPEESVQETLAAAVPKIMASFGNFA 300"
[1] " "
[1] "NDNEIKVLLKAFIANLKSSSPTIRRTAAGSAVSICQHSRRTQYFYSWLLNVLLGLLVPVE 303"
[1] "NDNEIKVLLKAFIANLKSSSPTIRRTAAGSAVSICQHSRRTQYFYSWLLNVLLGLLVPVE 360"
[1] " "
[1] "DEHSTLLILGVLLTLRYLVPLLQQQVKDTSLKGSFGVTRKEMEVSPSAEQLVQVYELTLH 363"
[1] "DEHSTLLILGVLLTLRYLVPLLQQQVKDTSLKGSFGVTRKEMEVSPSAEQLVQVYELTLH 420"
[1] " "
[1] "HTQHQDHNVVTGALELLQQLFRTPPPELLQTLTAVGGIGQLTAAKEESGGRSRSGSIVEL 423"
[1] "HTQHQDHNVVTGALELLQQLFRTPPPELLQTLTAVGGIGQLTAAKEESGGRSRSGSIVEL 480"
[1] " "
[1] "IAGGGSSCSPVLSRKQKGKVLLGEEEALEDDSESRSDVSSSALTASVKDEISGELAASSG 483"
[1] "IAGGGSSCSPVLSRKQKGKVLLGEEEALEDDSESRSDVSSSALTASVKDEISGELAASSG 540"
[1] " "
[1] "VSTPGSAGHDIITEQPRSQHTLQADSVDLASCDLTSSATDGDEEDILSHSSSQVSAVPSD 543"
[1] "VSTPGSAGHDIITEQPRSQHTLQADSVDLASCDLTSSATDGDEEDILSHSSSQVSAVPSD 600"
[1] " "
[1] "PAMDLNDGTQASSPISDSSQTTTEGPDSAVTPSDSSEIVLDGTDNQYLGLQIGQPQDEDE 603"
[1] "PAMDLNDGTQASSPISDSSQTTTEGPDSAVTPSDSSEIVLDGTDNQYLGLQIGQPQDEDE 660"
[1] " "
[1] "EATGILPDEASEAFRNSSMALQQAHLLKNMSHCRQPSDSSVDKFVLRDEATEPGDQENKP 663"
[1] "EATGILPDEASEAFRNSSMALQQAHLLKNMSHCRQPSDSSVDKFVLRDEATEPGDQENKP 720"
[1] " "
[1] "CRIKGDIGQSTDDDSAPLVHCVRLLSASFLLTGGKNVLVPDRDVRVSVKALALSCVGAAV 723"
[1] "CRIKGDIGQSTDDDSAPLVHCVRLLSASFLLTGGKNVLVPDRDVRVSVKALALSCVGAAV 780"
[1] " "
[1] "ALHPESFFSKLYKVPLDTTEYPEEQYVSDILNYIDHGDPQVRGATAILCGTLICSILSRS 783"
[1] "ALHPESFFSKLYKVPLDTTEYPEEQYVSDILNYIDHGDPQVRGATAILCGTLICSILSRS 840"
[1] " "
[1] "RFHVGDWMGTIRTLTGNTFSLADCIPLLRKTLKDESSVTCKLACTAVRNCVMSLCSSSYS 843"
[1] "RFHVGDWMGTIRTLTGNTFSLADCIPLLRKTLKDESSVTCKLACTAVRNCVMSLCSSSYS 900"
[1] " "
[1] "ELGLQLIIDVLTLRNSSYWLVRTELLETLAEIDFRLVSFLEAKAENLHRGAHHYTGLLKL 903"
[1] "ELGLQLIIDVLTLRNSSYWLVRTELLETLAEIDFRLVSFLEAKAENLHRGAHHYTGLLKL 960"
[1] " "
[1] "QERVLNNVVIHLLGDEDPRVRHVAAASLIRLVPKLFYKCDQGQADPVVAVARDQSSVYLK 963"
[1] "QERVLNNVVIHLLGDEDPRVRHVAAASLIRLVPKLFYKCDQGQADPVVAVARDQSSVYLK 1020"
[1] " "
[1] "LLMHETQPPSHFSVSTITRIYRGYNLLPSITDVTMENNLSRVIAAVSHELITSTTRALTF 1023"
```

```
[1] "LLMHETQPPSHFSVSTITRIYRGYNLLPSITDVTMENNLSRVIAAVSHELITSTTRALTF 1080"
[1] " "
[1] "GCCEALCLLSTAFPVCIWSLGWHCGVPPLSASDESRKSCTVGMATMILTLLSSAWFPLDL 1083"
[1] "GCCEALCLLSTAFPVCIWSLGWHCGVPPLSASDESRKSCTVGMATMILTLLSSAWFPLDL 1140"
[1] " "
[1] "SAHQDALILAGNLLAASAPKSLRSSWASEEEANPAATKQEEVWPALGDRALVPMVEQLFS 1143"
[1] "SAHQDALILAGNLLAASAPKSLRSSWASEEEANPAATKQEEVWPALGDRALVPMVEQLFS 1200"
[1] " "
[1] "HLLKVINICAHVLDDVAPGPAIKAALPSLTNPPSLSPIRRKGKEKEPGEQASVPLSPKKG 1203"
[1] "HLLKVINICAHVLDDVAPGPAIKAALPSLTNPPSLSPIRRKGKEKEPGEQASVPLSPKKG 1260"
[1] " "
[1] "SEASAASRQSDTSGPVTTSKSSSLGSFYHLPSYLKLHDVLKATHANYKVTLDLQNSTEKF 1263"
[1] "SEASAASRQSDTSGPVTTSKSSSLGSFYHLPSYLRLHDVLKATHANYKVTLDLQNSTEKF 1320"
[1] " "
[1] "GGFLRSALDVLSQILELATLQDIGKCVEEILGYLKSCFSREPMMATVCVQQLLKTLFGTN 1323"
[1] "GGFLRSALDVLSQILELATLQDIGKCVEEILGYLKSCFSREPMMATVCVQQLLKTLFGTN 1380"
[1] " "
[1] "LASQFDGLSSNPSKSQGRAQRLGSSSVRPGLYHYCFMAPYTHFTQALADASLRNMVQAEQ 1383"
[1] "LASQFDGLSSNPSKSQGRAQRLGSSSVRPGLYHYCFMAPYTHFTQALADASLRNMVQAEQ 1440"
[1] " "
[1] "ENDTSGWFDVLQKVSTQLKTNLTSVTKNRADKNAIHNHIRLFEPLVIKALKQYTTTTCVQ 1443"
[1] "ENDTSGWFDVLQKVSTQLKTNLTSVTKNRADKNAIHNHIRLFEPLVIKALKQYTTTTCVQ 1500"
[1] " "
[1] "LQKQVLDLLAQLVQLRVNYCLLDSDQVFIGFVLKQFEYIEVGQFRESEAIIPNIFFFLVL 1503"
[1] "LQKQVLDLLAQLVQLRVNYCLLDSDQVFIGFVLKQFEYIEVGQFRESEAIIPNIFFFLVL 1560"
[1] " "
[1] "LSYERYHSKQIIGIPKIIQLCDGIMASGRKAVTHAIPALQPIVHDLFVLRGTNKADAGKE 1563"
[1] "LSYERYHSKQIIGIPKIIQLCDGIMASGRKAVTHAIPALQPIVHDLFVLRGTNKADAGKE 1620"
[1] " "
[1] "LETQKEVVVSMLLRLIQYHQVLEMFILVLQQCHKENEDKWKRLSRQIADIILPMLAKQQM 1623"
[1] "LETQKEVVVSMLLRLIQYHQVLEMFILVLQQCHKENEDKWKRLSRQIADIILPMLAKQQM 1680"
[1] " "
[1] "HIDSHEALGVLNTLFEILAPSSLRPVDMLLRSMFVTPNTMASVSTVQLWISGILAILRVL 1683"
[1] "HIDSHEALGVLNTLFEILAPSSLRPVDMLLRSMFVTPNTMASVSTVQLWISGILAILRVL 1740"
[1] " "
[1] "ISQSTEDIVLSRIQELSFSPYLISCTVINRLRDGDSTSTLEEHSEGKQIKNLPEETFSRF 1743"
[1] "ISQSTEDIVLSRIQELSFSPYLISCTVINRLRDGDSTSTLEEHSEGKQIKNLPEETFSRF 1800"
[1] " "
[1] "LLQLVGILLEDIVTKQLKVEMSEQQHTFYCQELGTLLMCLIHIFKSGMFRRITAAATRLF 1803"
[1] "LLQLVGILLEDIVTKQLKVEMSEQQHTFYCQELGTLLMCLIHIFKSGMFRRITAAATRLF 1860"
[1] " "
[1] "RSDGCGGSFYTLDSLNLRARSMITTHPALVLLWCQILLLVNHTDYRWWAEVQQTPKRHSL 1863"
[1] "RSDGCGGSFYTLDSLNLRARSMITTHPALVLLWCQILLLVNHTDYRWWAEVQQTPKRHSL 1920"
[1] " "
[1] "SSTKLLSPQMSGEEEDSDLAAKLGMCNREIVRRGALILFCDYVCQNLHDSEHLTWLIVNH 1923"
[1] "SSTKLLSPQMSGEEEDSDLAAKLGMCNREIVRRGALILFCDYVCQNLHDSEHLTWLIVNH 1980"
[1] " "
[1] "IQDLISLSHEPPVQDFISAVHRNSAASGLFIQAIQSRCENLSTPTMLKKTLQCLEGIHLS 1983"
[1] "IQDLISLSHEPPVQDFISAVHRNSAASGLFIQAIQSRCENLSTPTMLKKTLQCLEGIHLS 2040"
[1] " "
[1] "QSGAVLTLYVDRLLCTPFRVLARMVDILACRRVEMLLAANLQSSMAQLPMEELNRIQEYL 2043"
[1] "QSGAVLTLYVDRLLCTPFRVLARMVDILACRRVEMLLAANLQSSMAQLPMEELNRIQEYL 2100"
```

```
[1] " "
[1] "QSSGLAQRHQRLYSLLDRFRLSTMQDSLSPSPPVSSHPLDGDGHVSLETVSPDKDWYVHL 2103"
[1] "QSSGLAQRHQRLYSLLDRFRLSTMQDSLSPSPPVSSHPLDGDGHVSLETVSPDKDWYVHL 2160"
[1] " "
[1] "VKSQCWTRSDSALLEGAELVNRIPAEDMNAFMMNSEFNLSLLAPCLSLGMSEISGGQKSA 2163"
[1] "VKSQCWTRSDSALLEGAELVNRIPAEDMNAFMMNSEFNLSLLAPCLSLGMSEISGGQKSA 2220"
[1] " "
[1] "LFEAAREVTLARVSGTVQQLPAVHHVFQPELPAEPAAYWSKLNDLFGDAALYQSLPTLAR 2223"
[1] "LFEAAREVTLARVSGTVQQLPAVHHVFQPELPAEPAAYWSKLNDLFGDAALYQSLPTLAR 2280"
[1] " "
[1] "ALAQYLVVVSKLPSHLHLPPEKEKDIVKFVVATLEALSWHLIHEQIPLSLDLQAGLDCCC 2283"
[1] "ALAQYLVVVSKLPSHLHLPPEKEKDIVKFVVATLEALSWHLIHEQIPLSLDLQAGLDCCC 2340"
[1] " "
[1] "LALQLPGLWSVVSSTEFVTHACSLIYCVHFILEAVAVQPGEQLLSPERRTNTPKAISEEE 2343"
[1] "LALQLPGLWSVVSSTEFVTHACSLIHCVHFILEAVAVQPGEQLLSPERRTNTPKAISEEE 2400"
[1] " "
[1] "EEVDPNTQNPKYITAACEMVAEMVESLQSVLALGHKRNSGVPAFLTPLLRNIIISLARLP 2403"
[1] "EEVDPNTQNPKYITAACEMVAEMVESLQSVLALGHKRNSGVPAFLTPLLRNIIISLARLP 2460"
[1] " "
[1] "LVNSYTRVPPLVWKLGWSPKPGGDFGTAFPEIPVEFLQEKEVFKEFIYRINTLGWTSRTQ 2463"
[1] "LVNSYTRVPPLVWKLGWSPKPGGDFGTAFPEIPVEFLQEKEVFKEFIYRINTLGWTSRTQ 2520"
[1] " "
[1] "FEETWATLLGVLVTQPLVMEQEESPPEEDTERTQINVLAVQAITSLVLSAMTVPVAGNPA 2523"
[1] "FEETWATLLGVLVTQPLVMEQEESPPEEDTERTQINVLAVQAITSLVLSAMTVPVAGNPA 2580"
[1] " "
[1] "VSCLEQQPRNKPLKALDTRFGRKLSIIRGIVEQEIQAMVSKRENIATHHLYQAWDPVPSL 2583"
[1] "VSCLEQQPRNKPLKALDTRFGRKLSIIRGIVEQEIQAMVSKRENIATHHLYQAWDPVPSL 2640"
[1] " "
[1] "SPATTGALISHEKLLLQINPERELGSMSYKLGQVSIHSVWLGNSITPLREEEWDEEEEEE 2643"
[1] "SPATTGALISHEKLLLQINPERELGSMSYKLGQVSIHSVWLGNSITPLREEEWDEEEEEE 2700"
[1] " "
[1] "ADAPAPSSPPTSPVNSRKHRAGVDIHSCSQFLLELYSRWILPSSSARRTPAILISEVVRS 2703"
[1] "ADAPAPSSPPTSPVNSRKHRAGVDIHSCSQFLLELYSRWILPSSSARRTPAILISEVVRS 2760"
[1] " "
[1] "LLVVSDLFTERNQFELMYVTLTELRRVHPSEDEILAQYLVPATCKAAAVLGMDKAVAEPV 2763"
[1] "LLVVSDLFTERNQFELMYVTLTELRRVHPSEDEILAQYLVPATCKAAAVLGMDKAVAEPV 2820"
[1] " "
[1] "SRLLESTLRSSHLPSRVGALHGVLYVLECDLLDDTAKQLIPVISDYLLSNLKGIAHCVNI 2823"
[1] "SRLLESTLRSSHLPSRVGALHGVLYVLECDLLDDTAKQLIPVISDYLLSNLKGIAHCVNI 2880"
[1] " "
[1] "HSQQHVLVMCATAFYLIENYPLDVGPEFSASIIQMCGVMLSGSEESTPSIIYHCALRGLE 2883"
[1] "HSQQHVLVMCATAFYLIENYPLDVGPEFSASIIQMCGVMLSGSEESTPSIIYHCALRGLE 2940"
[1] " "
[1] "RLLLSEQLSRLDAESLVKLSVDRVNVHSPHRAMAALGLMLTCMYTGKEKVSPGRTSDPNP 2943"
[1] "RLLLSEQLSRLDAESLVKLSVDRVNVHSPHRAMAALGLMLTCMYTGKEKVSPGRTSDPNP 3000"
[1] " "
[1] "AAPDSESVIVAMERVSVLFDRIRKGFPCEARVVARILPQFLDDFFPPQDIMNKVIGEFLS 3003"
[1] "AAPDSESVIVAMERVSVLFDRIRKGFPCEARVVARILPQFLDDFFPPQDIMNKVIGEFLS 3060"
[1] " "
[1] "NQQPYPQFMATVVYKVFQTLHSTGQSSMVRDWVMLSLSNFTQRAPVAMATWSLSCFFVSA 3063"
[1] "NQQPYPQFMATVVYKVFQTLHSTGQSSMVRDWVMLSLSNFTQRAPVAMATWSLSCFFVSA 3120"
[1] " "
```

```
[1] "STSPWVAAILPHVISRMGKLEQVDVNLFCLVATDFYRHQIEEELDRRAFQSVLEVVAAPG 3123"
[1] "STSPWVAAILPHVISRMGKLEQVDVNLFCLVATDFYRHQIEEELDRRAFQSVLEVVAAPG 3180"
[1] " "
[1] "SPYHRLLTCLRNVHKVTTC 3183"
[1] "SPYHRLLTCLRNVHKVTTC 3240"
[1] " "
```

Bioinfo method 2: RNAseq

The following chunks of code performs a differential expression analysis, which is also known as RNAseq, on a csv file I obtained from the NCBI(GEO) dataset. This csv file contains the raw counts of the expression for various genes in three major test groups. One of the groups is homozygous for the mutated HTT gene, another heterozygous for the mutated HTT gene, and the last one homozygous for the normal HTT gene as a control. By normalizing these count data and perform RNAseq pipeline on them, the second part of my scientific question could be answered, and we could see whether the most differentially expressed genes are associate with protein synthesis.

Hide

```
# Read in the csv file containing count data for the RNA-seq experiment, obtained from NCBI(GEO)
dataset
seqdata <- read.csv("GSE146673_Subramaniam_RNAseq_genecounts.csv")

# Have a look at our data
head(seqdata)
```

	X <chr>	control1_rna <int>	control2_rna <int>	control3_rna <int>	het1_rna <int>	het2_rna <int>	het3_rna <int>	hom
1	0610005C13Rik	0	0	0	0	0	0	
2	0610006L08Rik	0	0	0	0	0	0	
3	0610009B22Rik	213	211	207	345	354	349	
4	0610009E02Rik	0	0	0	0	0	0	
5	0610009L18Rik	0	0	0	0	0	0	
6	0610010B08Rik	0	0	1	0	0	0	

6 rows | 1-9 of 10 columns

Hide

```
# Organize the data by excluding the genes with 0 read count across all samples and rename the r
ow names by gene names
countdata <- seqdata[rowSums(seqdata[ ,c(2:ncol(seqdata))]) > 0, ]
renamed_countdata <- countdata[ ,-1]
rownames(renamed_countdata) <- make.names(countdata$X, unique = TRUE)

# Have a look at our data
head(renamed_countdata)
```

	control1_rna <int>	control2_rna <int>	control3_rna <int>	het1_rna <int>	het2_rna <int>	het3_rna <int>	hom
X0610009B22Rik	213	211	207	345	354	349	
X0610010B08Rik	0	0	1	0	0	0	
X0610010F05Rik	1807	2050	1525	2104	2237	1958	
X0610010K14Rik	1100	1106	1108	810	774	820	
X0610030E20Rik	968	942	780	1020	866	834	
X0610037L13Rik	674	692	598	773	756	682	

6 rows | 1-8 of 9 columns

Hide

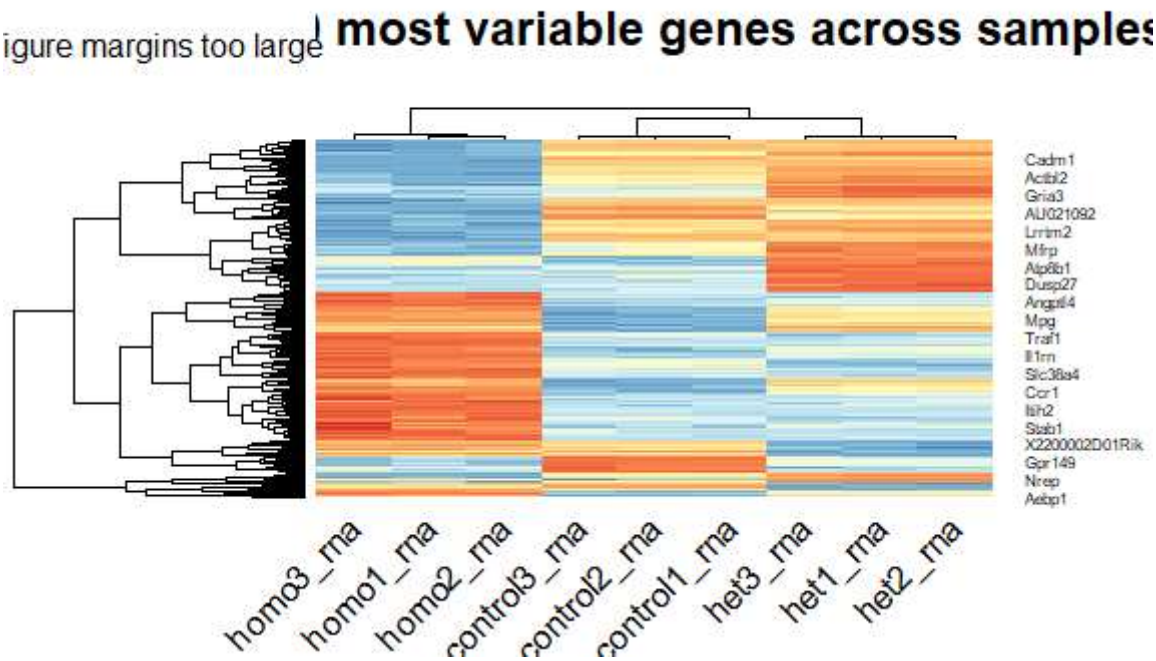
```
# Create a DGEList object
y <- DGEList(renamed_countdata)

# Obtain counts-per-million
myCPM <- cpm(renamed_countdata)

# Filter lowly expressed genes
thresh <- myCPM > 0.5
keep <- rowSums(thresh) >= 2

# Filter the DGEList object
y <- y[keep, keep.lib.sizes=FALSE]

# Normalize the counts
logcounts <- cpm(y,log=TRUE)
var_genes <- apply(logcounts, 1, var)

# Get the top 500 most variable genes
select_var <- names(sort(var_genes, decreasing=TRUE))[1:500]
highly_variable_lcpm <- logcounts[select_var,]
```

Data analysis method 1: Heat map

The following codes select the top 500 differentially expressed genes from the RNAseq result and present their relative expression by a heat map. The blue indicates the down-regulation of gene, and red indicates the up-regulation of a gene. Each column represents a subject from the original experiment, and we can see the three groups: mHTT homozygous, control, and mHTT heterozygous on the labels below. Each row represents a gene, and on the right are some examples of the most differentially expressed genes.

Hide

```
# Plot a heatmap to show the differentially expressed genes from the RNAseq data
mypalette <- brewer.pal(11,"RdYlBu")
morecols <- colorRampPalette(mypalette)
heatmap.2(highly_variable_lcpm,col=rev(morecols(50)),trace="none", main="Top 500 most variable g
enes across samples",scale="row",srtCol=45)
```



Data analysis method 2: GO annotation Table

The following codes select the top 10 differentially expressed genes from the RNAseq result, and mapped their gene symbols to their GO ids. The GO ids are then used to annotate their corresponding GO terms to see the specific functions these genes are involved in.

Hide

```
# Get the top 10 differentially expressed genes for GO annotation
top10 <- names(sort(var_genes, decreasing=TRUE))[1:10]

# Have a look at the list
top10
```

```
 [1] "Thbd"   "Ppbp"   "Megf10" "Scn3a"  "Selp"   "Tenm3"  "Aebp1"  "Lxn"    "Lrrn1"  "Ccl5"
```

Hide

```
# Get the gene ontology and KEGG pathway of the genes
top10_table <- select(org.Mm.eg.db, keys=top10, columns=c("GENENAME", "GO", "PATH"), keytype="SY
MBOL")
```

```
'select()' returned 1:many mapping between keys and columns
```

Hide

```
# Have a look at this table
top10_table
```

SY… <chr>	GENENAME <chr>	GO <chr>	EVIDE… <chr>	ONTO… <chr>
Thbd	thrombomodulin	GO:0004888	IEA	MF
Thbd	thrombomodulin	GO:0005509	IEA	MF
Thbd	thrombomodulin	GO:0005615	ISO	CC
Thbd	thrombomodulin	GO:0005774	ISO	CC
Thbd	thrombomodulin	GO:0005886	ISO	CC
Thbd	thrombomodulin	GO:0005886	TAS	CC
Thbd	thrombomodulin	GO:0005887	IEA	CC
Thbd	thrombomodulin	GO:0007565	IMP	BP
Thbd	thrombomodulin	GO:0007596	TAS	BP
Thbd	thrombomodulin	GO:0007599	IEA	BP

1-10 of 1,091 rows Previous **1** 2 3 4 5 6 … 100 Next

Hide

```
# Annotate the table based on GO ids
GOids <- top10_table$GO
annotated_table <- select(GO.db, keys=GOids, columns=c("TERM"), keytype="GOID")
```

```
'select()' returned many:1 mapping between keys and columns
```

Hide

```
# Have a look at the annotation
annotated_table
```

GOID <chr>	TERM <chr>

GOID <chr>	TERM <chr>
GO:0004888	transmembrane signaling receptor activity
GO:0005509	calcium ion binding
GO:0005615	extracellular space
GO:0005774	vacuolar membrane
GO:0005886	plasma membrane
GO:0005886	plasma membrane
GO:0005887	integral component of plasma membrane
GO:0007565	female pregnancy
GO:0007596	blood coagulation
GO:0007599	hemostasis

1-10 of 1,091 rows Previous **1** 2 3 4 5 6 … 100 Next

Analysis of Results

Based on the pairwise sequence alignment, we confirm the structural mutation of the Huntingtin protein. There seems to be a poly-Q mutation on the Huntingtin protein, which lead to its malfunction and ultimately causes the pathology of Huntington's Disease.

From the heat map, we could see that the genetic expression of each experimental group are distinct from one another, and there exists similarity across subjects within each group. Specifically, the group that is homozygous for mutated HTT gene is most different from the other groups, as it shows down-regulation of genes like Ccr1, and up-regulation of genes such as Aebp1. The other two groups are somewhat alike, but the heterozygous group still show up-regulation in genes such as Dusp27, which is not seen in the control group.

The data from the Gene Ontology annotation reveals a more striking result. Contrary to the hypothesis that these genes may be involved in functions that alters protein synthesis, specifically in the translation phase, the top 10 differentially expressed genes are more associated to inflammatory response and neurotoxicity. Only a few GO terms describe functions related to protein synthesis. This result may be due to the limited numebr of top expressed genes chosen for the annotation.