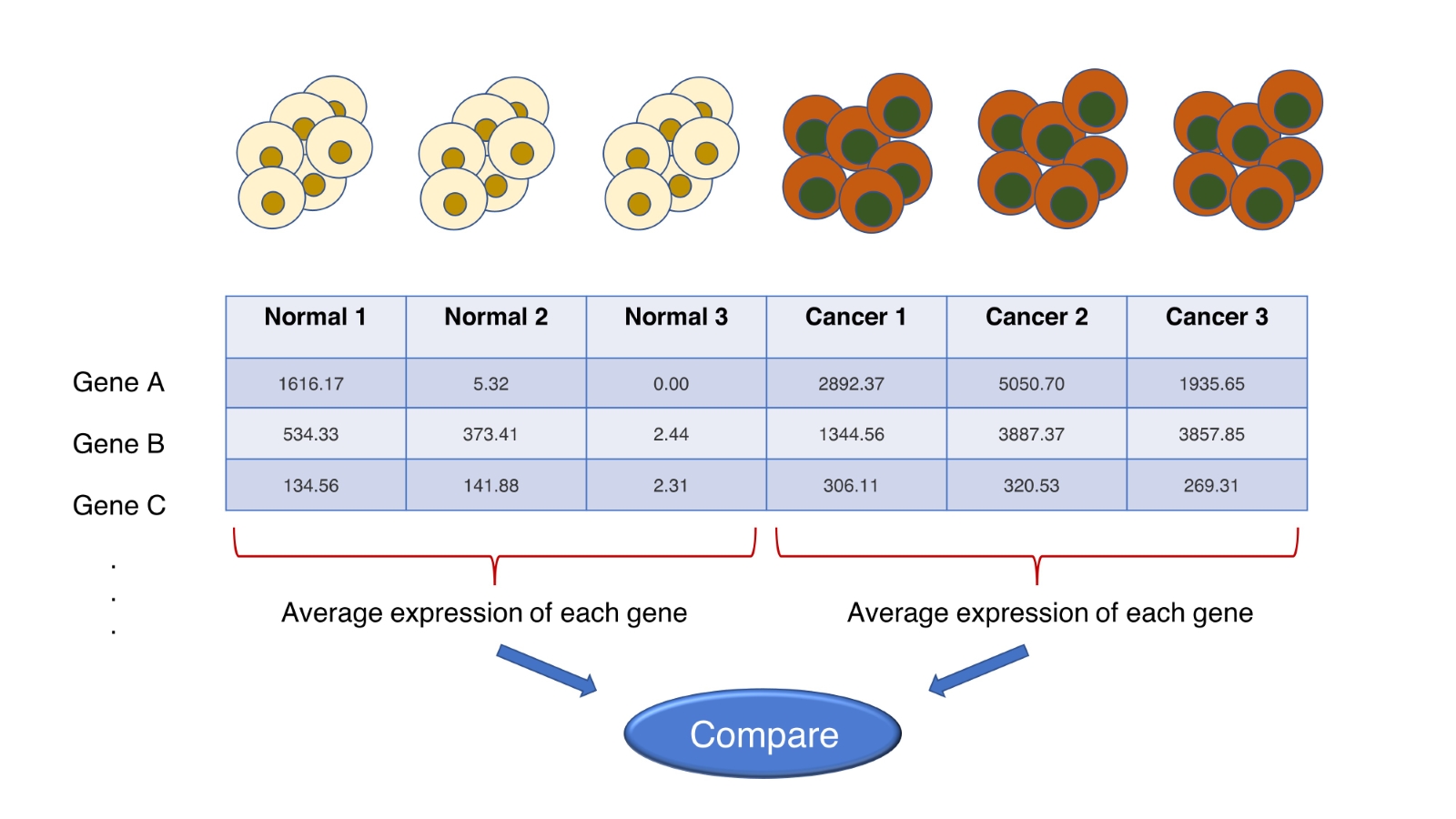
# RNA-seq data analysis

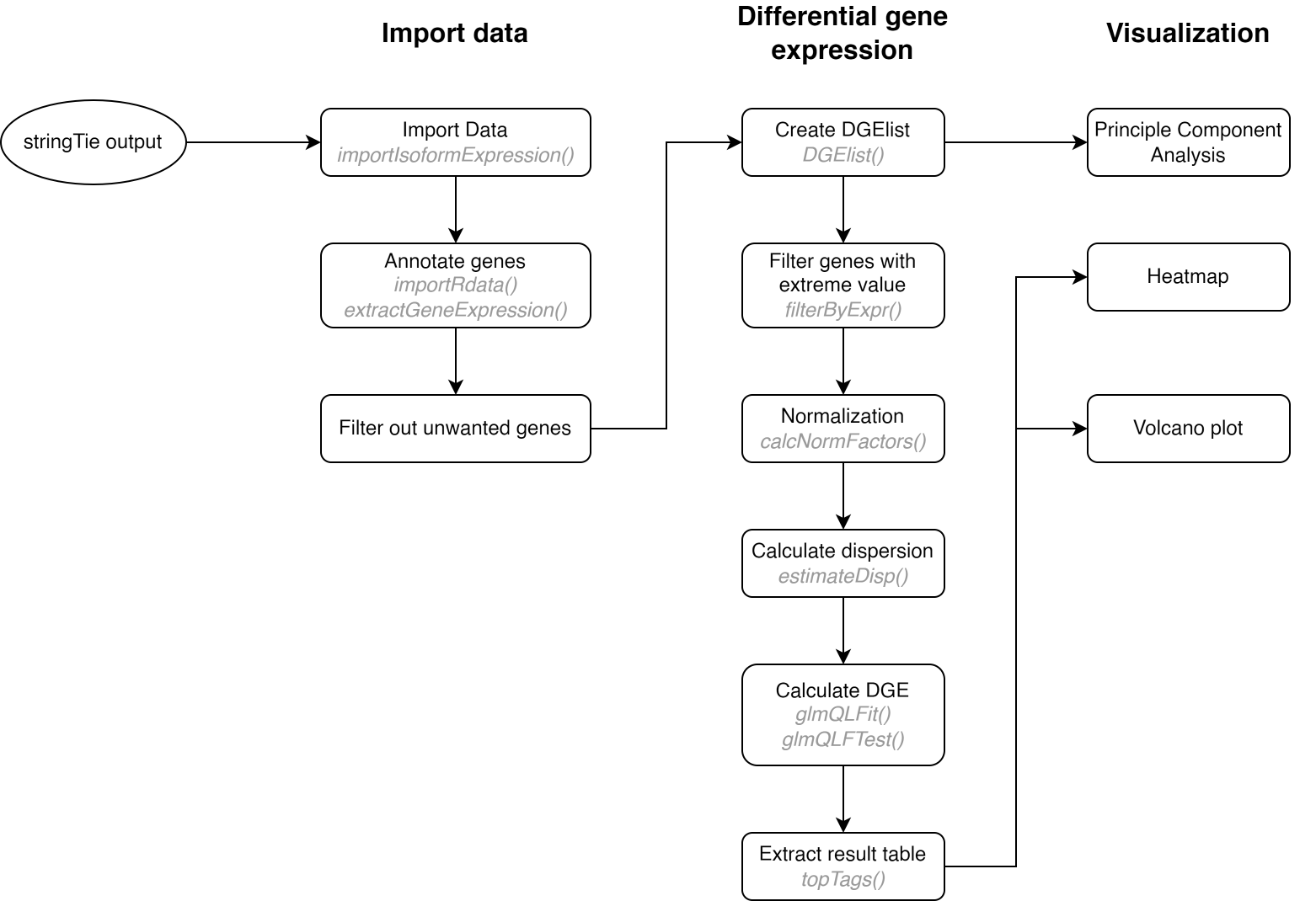
## What is differential gene expression?

Differential gene expression คือ การหาความแตกต่างของการแสดงออกของ gene ระหว่างกลุ่มตัวอย่างสองกลุ่มขึ้นไป เพื่อให้ได้ผลลัพธ์ว่ามี gene ตัวใดตัวหนึ่งแสดงออกมากหรือน้อยกว่าผิดปกติ เมื่อเทียบกับกลุ่มอื่นๆ โดยค่าที่นำมาใช้เปรียบเทียบนั้น จะได้มาจากขึ้นตอน quantification



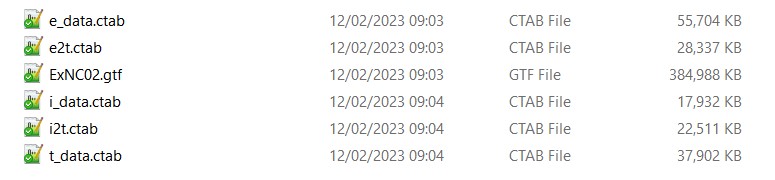
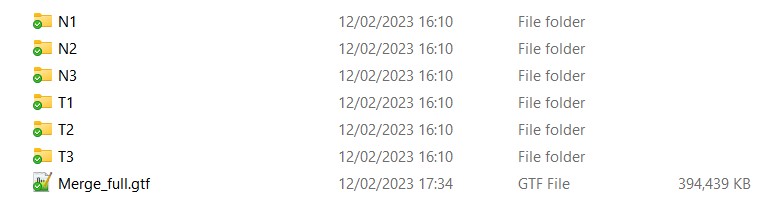
## Differential gene expression workflow

หลังจากเราได้ข้อมูล Gene expression quantification จาก StringTie แล้ว เราจะนำข้อมูลมาผ่านกระบวนการต่างๆ เพื่อหาความแตกต่างของ gene แต่ละกลุ่ม



## Import data

ในการที่จะนำไฟล์ RNA analysis เข้าสู่ R นั้น จำเป็นที่จะต้องเตรียมข้อมูลให้เหมาะกับ function ที่เราจะใช้ในการอ่านข้อมูล โดยในแต่ละ sample นั้น จะประกอบด้วยไฟล์ .gtf และ .ctab ที่ได้จากการวิเคราะห์ก่อนหน้านี้



### Import quantification

โดยเราจะใช้ function importIsoformExpression ในการนำข้อมูลเข้าให้อยู่ในรูปของ dataframe

stringTie\_quant <- importIsoformExpression(  
 parentDir = "./Source/bladder",  
 addIsofomIdAsColumn = FALSE,  
 readLength = 150  
)

## Step 1 of 3: Identifying which algorithm was used...

## The quantification algorithm used was: StringTie

## Found 6 quantification file(s) of interest

## Step 2 of 3: Reading data...

## reading in files with read\_tsv

## 1 2 3 4 5 6   
## Step 3 of 3: Normalizing abundance values (not counts) via edgeR...  
## Done

head(stringTie\_quant$abundance, 10)

## N1 N2 N3 T1 T2 T3  
## MSTRG.24.1 0.7207532 0.000000000 0 0.000000000 0.00000000 0.00000000  
## MSTRG.24.2 0.0000000 1.669273712 0 0.000000000 0.00000000 0.00000000  
## MSTRG.24.4 0.0000000 0.493513530 0 0.000000000 0.00000000 0.00000000  
## MSTRG.24.5 0.0000000 0.006439551 0 0.000000000 0.00000000 0.00000000  
## MSTRG.24.3 0.3298251 1.521147453 0 0.004002289 0.00000000 0.00000000  
## MSTRG.24.6 1.7884048 1.259405992 0 0.000000000 0.01820757 0.00000000  
## ENST00000456328.2 0.5592946 0.131153423 0 0.049252195 0.01158960 0.00000000  
## ENST00000450305.2 0.0000000 0.000000000 0 0.000000000 0.00000000 0.00000000  
## MSTRG.26.1 0.0000000 0.000000000 0 1.063266717 0.18918203 0.42427910  
## MSTRG.26.4 0.0000000 0.000000000 0 0.000000000 0.04498711 0.09408183

จะเห็นว่าในไฟล์นั้นประกอบด้วย ส่วนแถว ซึ่งเป็นชื่อ isoform ของ RNA นั้นๆ และ ส่วนคอลัมน์ ซึ่งเป็นชื่อของ sample ที่เราศึกษา โดยข้อมูลแต่ละจุดคือ ค่าของ expression ที่ได้จากการวิเคราะห์

### Make a design matrix

หลังจากนั้น เราต้องสร้าง condition matrix ซึ่งประกอบด้วย แต่ละ sample ที่ต้องการศึกษา และ condition ของตัวอย่างนั้น ซึ่งในที่นี้เราจะแบ่งเป็นสองกลุ่ม ก็คือ Normal และ Tumor

design <- data.frame(  
 sampleID = colnames(stringTie\_quant$abundance),  
 condition = gsub(".{1}$", "", colnames(stringTie\_quant$abundance)) # Remove number  
)  
  
design

## sampleID condition  
## 1 N1 N  
## 2 N2 N  
## 3 N3 N  
## 4 T1 T  
## 5 T2 T  
## 6 T3 T

### Create a list of files

หลังจากนั้นเราจะต้องรวมไฟล์เข้ากันกับ annotation file ซึ่งจะทำการ annotate ชื่อ gene นั้น จาก Ensemble format เป็น gene id

switch\_analyze\_Rlist <- importRdata(  
 isoformCountMatrix = stringTie\_quant$counts,  
 isoformRepExpression = stringTie\_quant$abundance,  
 designMatrix = design,  
 isoformExonAnnoation = "./Source/bladder/BCaMerge.gtf",  
)

## comparison estimated\_genes\_with\_dtu  
## 1 N vs T 4768 - 7948

names(switch\_analyze\_Rlist)

## [1] "isoformFeatures" "exons" "conditions"   
## [4] "designMatrix" "sourceId" "isoformCountMatrix"   
## [7] "isoformRepExpression" "runInfo" "isoformRepIF"

สังเกตว่าภายใน 1 list นั้นจะประกอบด้วยหลายหัวข้อ ซึ่งเราสามารถดึงออกมาใช้ได้ด้วย operator $

### Extract gene count matrix

ต่อไปเราจะใช้แค่ gene count matrix จาก list ที่เราสร้างขึ้นมา

gene\_count <- extractGeneExpression(  
 switch\_analyze\_Rlist,  
 extractCounts = TRUE # set to FALSE for abundances  
)  
  
head(gene\_count,10)

## gene\_id gene\_name N1 N2 N3 T1  
## 1 ENSG00000000003.15 TSPAN6 1616.16632 5.323296 0.000000 2892.36698  
## 2 ENSG00000000419.14 DPM1 534.33260 373.409670 2.438447 1344.56193  
## 3 ENSG00000000457.14 SCYL3 134.55807 141.880574 2.312303 306.11165  
## 4 ENSG00000000460.17 C1orf112 68.44894 47.264863 17.260795 117.66245  
## 5 ENSG00000000938.13 FGR 40.13006 3177.326075 62.927494 10.98451  
## 6 ENSG00000000971.16 CFH 801.43355 8.169078 5.188639 1304.66135  
## 7 ENSG00000001036.14 FUCA2 246.33240 104.995451 5.653375 1192.91176  
## 8 ENSG00000001084.13 GCLC 374.82061 77.947080 6.817594 4565.35564  
## 9 ENSG00000001167.15 NFYA 128.75396 215.173089 0.000000 526.52422  
## 10 ENSG00000001460.18 STPG1 105.25642 27.359621 3.458716 253.40639  
## T2 T3  
## 1 5050.6964 1935.65287  
## 2 3887.3732 3857.84694  
## 3 320.5272 269.31308  
## 4 207.5864 72.52807  
## 5 192.8986 109.53769  
## 6 13678.3377 2457.62369  
## 7 3405.1256 1728.34830  
## 8 1549.1920 3010.92600  
## 9 783.7322 664.80964  
## 10 370.8472 179.81425

จะเห็นว่าขณะนี้เรามีทั้ง gene\_id และ gene\_name แล้ว

### Filter out lncRNA

ต่อไป เราจะนำรายชื่อของ RNA ที่เราไม่สนใจออกไป ซึ่งในที่นี้คือ long-noncoding RNA ซึ่งมักจะไม่ถูกเปลี่ยนไปเป็นโปรตีน แต่จะใช้สำหรับ function อื่นๆ ในร่างกาย

ก่อนอื่น เราต้อง import file ที่มีการ annotate ชนิดของ RNA เข้ามาใน R ก่อน โดยใช้ function rtracklayer::import()

V38.gtf <- rtracklayer::import("./Source/gencode.v38.annotation.gtf")  
unique(V38.gtf$gene\_type)

## [1] "transcribed\_unprocessed\_pseudogene" "unprocessed\_pseudogene"   
## [3] "miRNA" "lncRNA"   
## [5] "protein\_coding" "processed\_pseudogene"   
## [7] "snRNA" "transcribed\_processed\_pseudogene"   
## [9] "misc\_RNA" "TEC"   
## [11] "transcribed\_unitary\_pseudogene" "snoRNA"   
## [13] "scaRNA" "rRNA\_pseudogene"   
## [15] "unitary\_pseudogene" "polymorphic\_pseudogene"   
## [17] "pseudogene" "rRNA"   
## [19] "IG\_V\_pseudogene" "scRNA"   
## [21] "IG\_V\_gene" "IG\_C\_gene"   
## [23] "IG\_J\_gene" "sRNA"   
## [25] "ribozyme" "translated\_processed\_pseudogene"   
## [27] "vault\_RNA" "TR\_C\_gene"   
## [29] "TR\_J\_gene" "TR\_V\_gene"   
## [31] "TR\_V\_pseudogene" "translated\_unprocessed\_pseudogene"   
## [33] "TR\_D\_gene" "IG\_C\_pseudogene"   
## [35] "TR\_J\_pseudogene" "IG\_J\_pseudogene"   
## [37] "IG\_D\_gene" "IG\_pseudogene"   
## [39] "Mt\_tRNA" "Mt\_rRNA"

จะเห็นว่ามีชนิดของ RNA มากมายหลายชนิดในไฟล์นี้ เราจะทำการเลือกชื่อ RNA ที่เราไม่สนใจ ซึ่งก็คือ lncRNA มากรองข้อมูลในส่วนที่เราไม่ต้องการออกในไฟล์ต้นฉบับของเรา

หลังจากนั้นเราจะนำ column gene\_id ออก และเปลี่ยน gene\_name ให้เป็นชื่อแถว

**Note:** ในที่ข้อมูลนี้เราจะทำการตัด RNA ที่มีชื่อซ้ำออกไป เพื่อให้ง่ายแก่การสอน ซึ่งในการวิเคราะห์จริงอาจจะต้องใช้วิธีอื่นในการวิเคราะห์ชื่อ RNA ที่ซ้ำกันใน sample เดียวกัน

lncRNA\_subset <- V38.gtf$gene\_type == "lncRNA"  
lncRNA <- V38.gtf[lncRNA\_subset]$gene\_name  
  
gene\_count\_no\_lncRNA <- gene\_count %>% filter(!(gene\_name %in% unique(lncRNA)))  
head(gene\_count\_no\_lncRNA, 10)

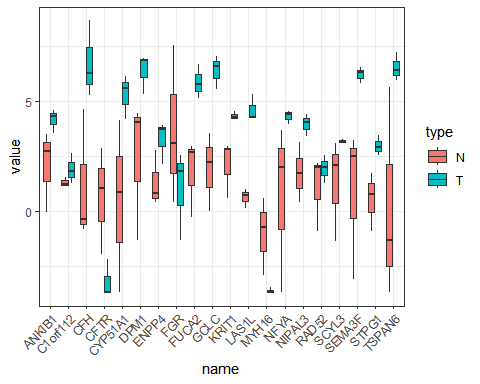
## gene\_id gene\_name N1 N2 N3 T1  
## 1 ENSG00000000003.15 TSPAN6 1616.16632 5.323296 0.000000 2892.36698  
## 2 ENSG00000000419.14 DPM1 534.33260 373.409670 2.438447 1344.56193  
## 3 ENSG00000000457.14 SCYL3 134.55807 141.880574 2.312303 306.11165  
## 4 ENSG00000000460.17 C1orf112 68.44894 47.264863 17.260795 117.66245  
## 5 ENSG00000000938.13 FGR 40.13006 3177.326075 62.927494 10.98451  
## 6 ENSG00000000971.16 CFH 801.43355 8.169078 5.188639 1304.66135  
## 7 ENSG00000001036.14 FUCA2 246.33240 104.995451 5.653375 1192.91176  
## 8 ENSG00000001084.13 GCLC 374.82061 77.947080 6.817594 4565.35564  
## 9 ENSG00000001167.15 NFYA 128.75396 215.173089 0.000000 526.52422  
## 10 ENSG00000001460.18 STPG1 105.25642 27.359621 3.458716 253.40639  
## T2 T3  
## 1 5050.6964 1935.65287  
## 2 3887.3732 3857.84694  
## 3 320.5272 269.31308  
## 4 207.5864 72.52807  
## 5 192.8986 109.53769  
## 6 13678.3377 2457.62369  
## 7 3405.1256 1728.34830  
## 8 1549.1920 3010.92600  
## 9 783.7322 664.80964  
## 10 370.8472 179.81425

## Get only count matrix  
count\_matrix <- gene\_count\_no\_lncRNA %>%  
 distinct(gene\_name, .keep\_all = TRUE) %>% # Remove duplicate gene\_name   
 column\_to\_rownames("gene\_name") %>%   
 select(-gene\_id) %>%   
 as.matrix  
  
head(count\_matrix, 10)

## N1 N2 N3 T1 T2 T3  
## TSPAN6 1616.16632 5.323296 0.000000 2892.36698 5050.6964 1935.65287  
## DPM1 534.33260 373.409670 2.438447 1344.56193 3887.3732 3857.84694  
## SCYL3 134.55807 141.880574 2.312303 306.11165 320.5272 269.31308  
## C1orf112 68.44894 47.264863 17.260795 117.66245 207.5864 72.52807  
## FGR 40.13006 3177.326075 62.927494 10.98451 192.8986 109.53769  
## CFH 801.43355 8.169078 5.188639 1304.66135 13678.3377 2457.62369  
## FUCA2 246.33240 104.995451 5.653375 1192.91176 3405.1256 1728.34830  
## GCLC 374.82061 77.947080 6.817594 4565.35564 1549.1920 3010.92600  
## NFYA 128.75396 215.173089 0.000000 526.52422 783.7322 664.80964  
## STPG1 105.25642 27.359621 3.458716 253.40639 370.8472 179.81425

เมื่อลองนำข้อมูลมาสร้าง boxplot คร่าวๆ จะพบว่ามีหลาย gene ที่มีความแตกต่างกัน ซึ่งต่อไปเราจะนำมาเข้าสู่กระบวนการหา differential gene expression เพื่อดูว่ามี gene ใดบ้างที่มีความแตกต่างกันระหว่างสองกลุ่มอย่างมีนัยสำคัญ

count\_matrix %>%   
 edgeR::cpm(log=TRUE) %>%   
 head(20) %>%   
 t %>%   
 as.data.frame() %>%   
 rownames\_to\_column("type") %>%   
 tidyr::pivot\_longer(-type) %>%  
 mutate(type = gsub("\\d", "", type)) %>%   
 ggplot(aes(x = name, y =value, fill = type)) + geom\_boxplot() +   
 theme\_bw() +  
 theme(axis.text.x = element\_text(angle = 45, vjust = 1, hjust=1))



## Differential gene expression analysis

ก่อนที่เราจะทำการ visualize ข้อมูลนั้น เราจะต้องทำการวิเคราะห์ก่อนว่า RNA ไหนที่มีการแสดงออกระหว่างสองกลุ่มที่แตกต่างอย่างมีนัยสำคัญ

โดยเราจะเริ่มจากการสร้าง design matrix ซึ่งบ่งบอกว่าใครอยู่กลุ่มไหน

library(edgeR)  
  
colnames(count\_matrix)

## [1] "N1" "N2" "N3" "T1" "T2" "T3"

group <- design$condition  
diff\_design <- model.matrix(~0+group)  
diff\_design

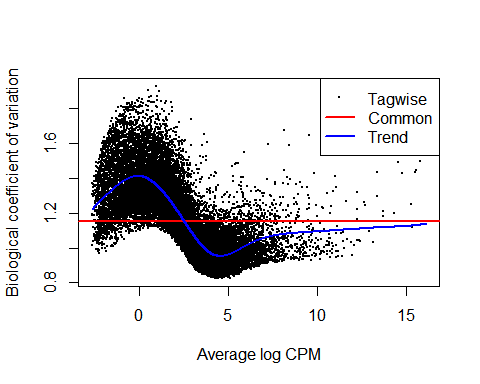
## groupN groupT  
## 1 1 0  
## 2 1 0  
## 3 1 0  
## 4 0 1  
## 5 0 1  
## 6 0 1  
## attr(,"assign")  
## [1] 1 1  
## attr(,"contrasts")  
## attr(,"contrasts")$group  
## [1] "contr.treatment"

สิ่งที่เราเห็นคือ design matrix ของกลุ่มที่เราต้องการ โดยหมายเลข 1 คือตัวบ่งบอกว่า sample เราอยู่ในกลุ่มนั้นๆ โดยในที่นี่ sample 1-3 จะอยู่ในกลุ่ม Normal ส่วน sample 4-6 จะอยู่ในกลุ่ม Tumor

### Normalization

หลังจากนั้น เราจะต้องทำการ normalize ค่าการแสดงออกของ RNA เนื่องจากการ run RNA seq ในแต่ละ sample นั้น สภาวะของเครื่องอาจจะมีความแตกต่างกันบ้างเล็กน้อย ส่งผลให้ค่า signal intensity พื้นหลังนั้นมีไม่เท่ากัน

dge <- DGEList(counts=count\_matrix, group=group)   
  
keep <- filterByExpr(dge, group=group,min.count=2, min.prob=0.5)  
  
dge <- dge[keep,]  
  
# Calculate normalization factor  
genexp <- calcNormFactors(dge)  
  
# GLM Common dispersion  
genexp <- estimateGLMCommonDisp(dge, diff\_design)  
  
# Estimate GLM trended dispersions  
genexp <- estimateGLMTrendedDisp(genexp, diff\_design)  
  
# Tagwise dispersion of each gene  
genexp <- estimateGLMTagwiseDisp(genexp, diff\_design)  
  
plotBCV(genexp)



head(genexp$counts)

## N1 N2 N3 T1 T2 T3  
## TSPAN6 1616.16632 5.323296 0.000000 2892.36698 5050.6964 1935.65287  
## DPM1 534.33260 373.409670 2.438447 1344.56193 3887.3732 3857.84694  
## SCYL3 134.55807 141.880574 2.312303 306.11165 320.5272 269.31308  
## C1orf112 68.44894 47.264863 17.260795 117.66245 207.5864 72.52807  
## FGR 40.13006 3177.326075 62.927494 10.98451 192.8986 109.53769  
## CFH 801.43355 8.169078 5.188639 1304.66135 13678.3377 2457.62369

หลังจาก normalize แล้ว เราจะทำการวิเคราะห์ differential gene expression โดยการใช้การวิเคราะห์ทางสถิติที่เรียกว่า negative binomial generalized log-linear model ซึ่งโดยสรุปคร่าวๆ คือการเปรียบเทียบ average log RNA expression ระหว่างสองกลุ่ม แต่ซับซ้อนกว่าเพื่อลด ผลบวกลวง

**Note:** package ที่นิยมใช้ในปัจจุบัน ได้แก่ limma, edgeR, และ DEseq โดยจะมีความแตกต่างกันเล็กน้อยในส่วนของการวิเคราะห์ทางสถิติ สำหรับผู้ที่สนใจสามารถศึกษาเพิ่มเติมได้ที่ <https://www.biostars.org/p/284775/>

หลังจากนั้นเราจะใช้ funtion topTags() เพื่อทำการดึงตารางผลของ differential RNA expression ออกมา

fit <- glmQLFit(genexp, diff\_design)  
  
genediff <- glmQLFTest(fit, contrast=c(-1,1))  
  
# All genes  
all\_gene <- topTags(genediff, n = Inf, p.value = 1, adjust.method = "fdr")  
  
all\_gene$table %>%   
 rownames\_to\_column("gene\_name") %>%   
readr::write\_csv("all\_gene.csv")  
# Only significant value  
sig\_gene <- topTags(genediff, n = Inf, p.value = 0.05,   
 adjust.method = "fdr", sort.by = "logFC")  
  
# Total diffenetiated gene  
summary(decideTests(genediff))

## -1\*groupN 1\*groupT  
## Down 1792  
## NotSig 17504  
## Up 939

# Summary table  
(sig\_gene$table)

## logFC logCPM F PValue FDR  
## HBD -18.273967 9.562807862 46.650297 8.545057e-12 1.729092e-07  
## ENSG10010139367.1 -17.649666 8.938617764 45.143875 1.842839e-11 1.864493e-07  
## AQP9 -15.704149 7.349606642 40.412758 2.066846e-10 1.099827e-06  
## CXCR1 -14.963082 6.252996881 39.713938 2.955267e-10 1.099827e-06  
## MEFV -14.752717 6.042819257 39.573130 3.176099e-10 1.099827e-06  
## FPR2 -14.641776 5.931976369 39.521492 3.261161e-10 1.099827e-06  
## ADGRE3 -13.942470 5.233551274 39.032417 4.188918e-10 1.210896e-06  
## ALAS2 -13.895256 8.159710331 36.205749 1.783055e-09 2.405342e-06  
## ADGRG3 -13.615974 4.907601442 38.336422 5.982575e-10 1.451147e-06  
## PROK2 -13.564091 5.955987834 36.981569 1.197829e-09 1.897911e-06  
## MT-ATP8 13.558617 13.119447847 32.113716 1.458995e-08 7.380690e-06  
## GLT1D1 -13.536189 4.828029253 38.087774 6.795202e-10 1.451147e-06  
## FCAR -13.505521 4.797405480 37.982573 7.171468e-10 1.451147e-06  
## FCGR3B -13.473789 8.385197697 35.019121 3.277923e-09 3.158513e-06  
…

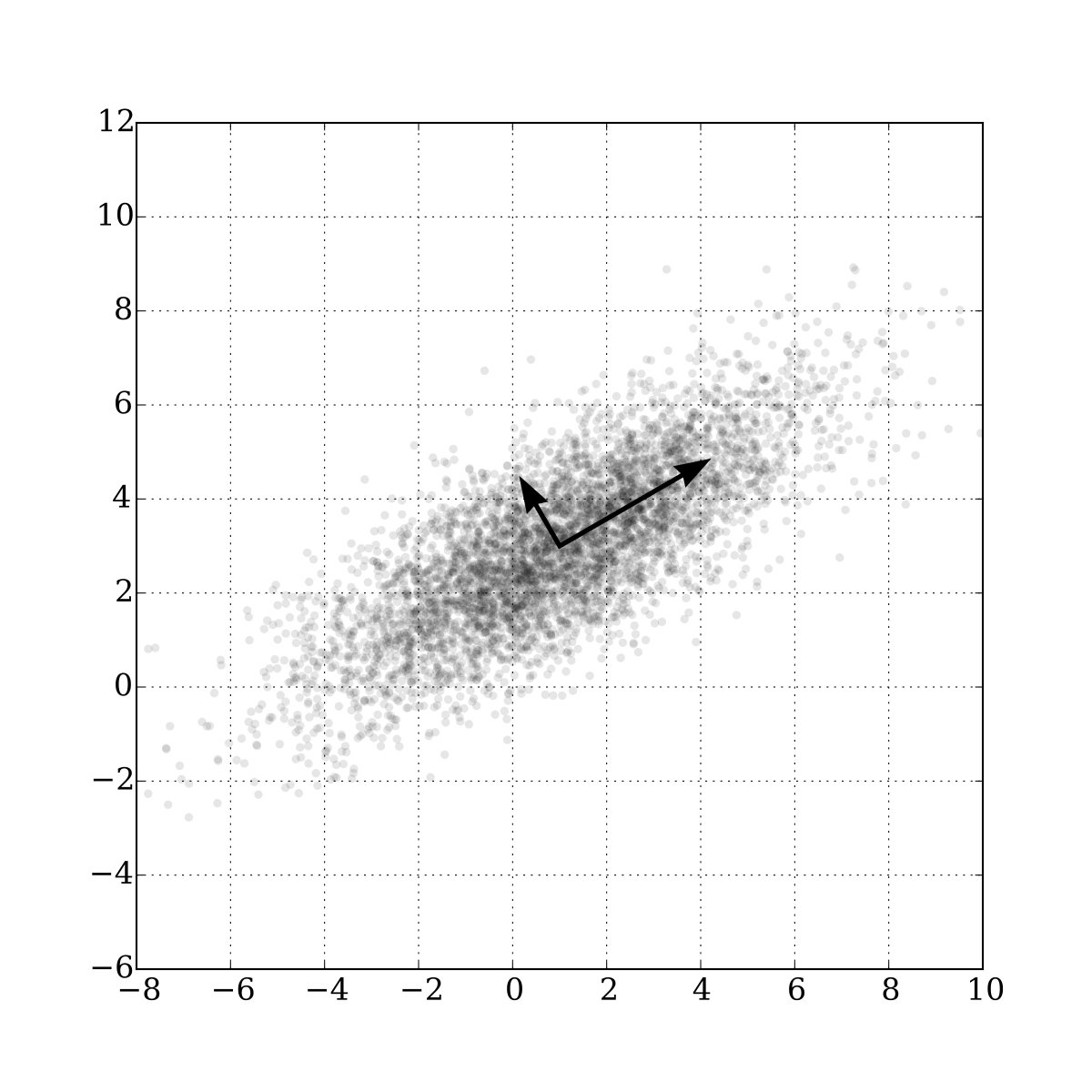
โดยจากตาราง จะพบว่ามีการแสดงค่าต่างๆ โดยที่เราสนใจมักจะเป็น

* logFC ซึ่งก็คือ fold change ของ RNA expression ระหว่างกลุ่ม Normal vs Tumor
* pvalue โดยเรามักจะต้องปรับผลเพื่อลดภาวะผลบวกลวงออกไปด้วย เราจึงใช้ column FDR ไม่ใช่ PValue

## Visualization

### Principal Component Analysis (PCA)

PCA คือการลดมิติของปริมาณข้อมูลลงเพือทำให้เกิดความง่ายขึ้นในการวิเคราะห์ โดยใช้หลักการรวมข้อมูลแบบ linear combination ที่มีความแปรปรวนใกล้เคียง ซึ่งโดยทั่วไปแล้ว จะนำมาใช้ในการดูความแตกต่างกันของลักษณะข้อมูลในแต่ละกลุ่มแบบคร่าวๆ หรือใช้ในการค้นหาความผิดปกติของข้อมูลที่เกินจากสภาวะที่ต่างกัน (batch effect) โดยที่ข้อมูลที่มีลักษณะใกล้เคียงกันจะอยู่ในตำแหน่งที่ใกล้เคียงกัน



รูปจาก: <https://en.wikipedia.org/wiki/Principal_component_analysis>

#### Requirement

* ข้อมูลควรมีการถูก normalized โดยอาจจะ centered (ทำให้ scale เริ่มต้นที่ 0) หรือไม่ก็ได้
* ต้องไม่มี missing value

library(PCAtools)

## Loading required package: ggrepel

##   
## Attaching package: 'PCAtools'

## The following objects are masked from 'package:stats':  
##   
## biplot, screeplot

# Calculate log-counts-per-million  
logcpm <- cpm(dge, prior.count = 2, log = TRUE)  
  
# Create a metadata table  
metadata <- data.frame(row.names = colnames(logcpm),  
 group = c(rep(1,3), rep(2,3)))  
  
(metadata)

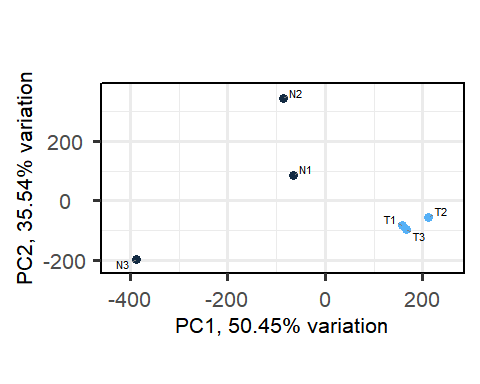
## group  
## N1 1  
## N2 1  
## N3 1  
## T1 2  
## T2 2  
## T3 2

โดยการแปลผล PCA นั้น ควรดูไล่ไปทีละแกน (มิติ 1 -> มิติ 2 ไม่ใช่ดู 2 มิติพร้อมกัน)

# Perform PCA analysis  
pc <- pca(logcpm, metadata = metadata, removeVar = 0.1)

## -- removing the lower 10% of variables based on variance

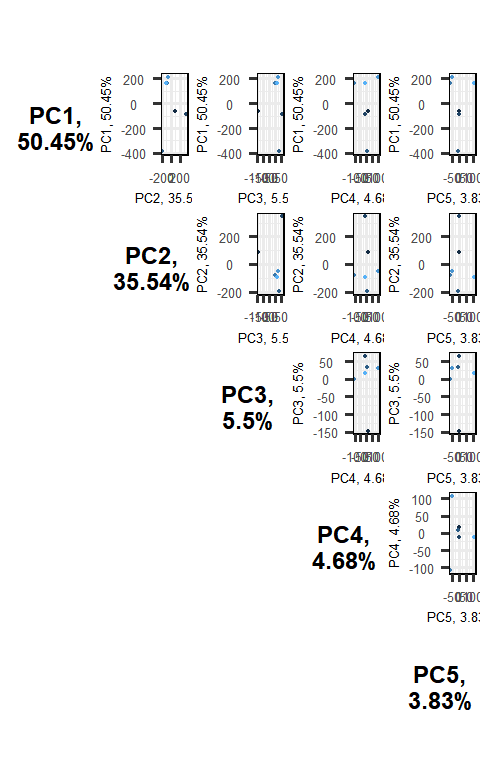
# Create PCA plot  
biplot(pc, colby = "group")



จะเห็นได้ว่า ในส่วนของ T1, T2 และ T3 นั้นค่อนข้างเกาะกลุ่มกัน แต่ N นั้น มีความแตกต่างกันพอสมควรในทั้งสองมิติ

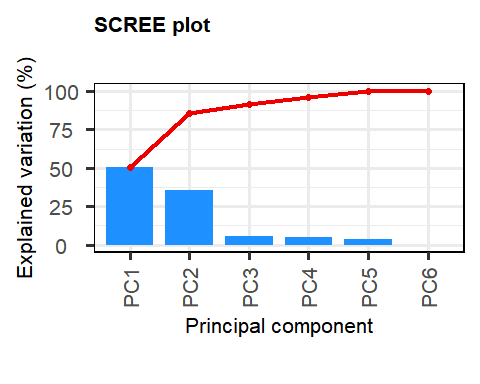
แม้ว่าในกราฟจะมีแค่ 2 มิติ แต่โดยที่จริงแล้วมิตินั้นจะโดนลดลงเหลือ n มิติ

pairsplot(pc)



ซึ่งเราสามารถดูความมากน้อยของผลกระทบของในแต่ละมิติได้โดยใช้ Scree plot โดยมิติแรกจะมีผลมากกว่ามิติหลังเสมอ

screeplot(pc)



ในส่วนของข้อมูลเชิงลึกของ PCA สามารถศึกษาเพิ่มเติมได้ในเอกสารแนบ: <http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf>

ตัวอย่างการใช้งานเพิ่มเติม: <https://www.bioconductor.org/packages/release/bioc/vignettes/PCAtools/inst/doc/PCAtools.html>

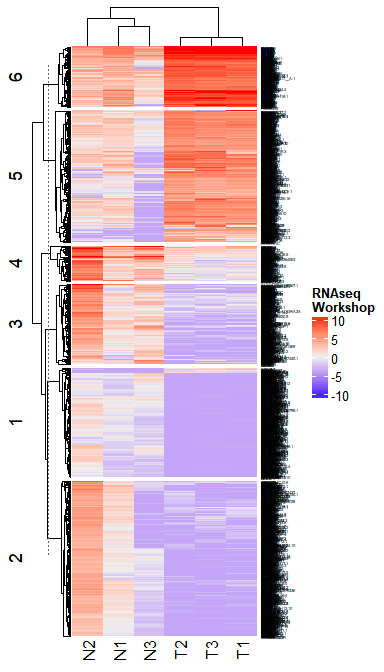
### Heatmap

Heatmap คือการเปลี่ยนข้อมูลที่มีให้อยู่ในรูปของสี ซึ่งจะแสดงความแตกต่างตามค่าที่มากหรือน้อย โดยการสร้าง heatmap นั้นจะใช้ข้อมูลดิบ (ก่อนทำ differential expression) ซึ่งจะทำให้เห็นภาพรวมของข้อมูลแต่จะไม่ให้ข้อมูลความแตกต่างทางด้านสถิติมากนัก

ซึ่งโดยปกติถ้านำข้อมูลทั้งหมดมาสร้าง heatmap จะทำให้รูปมีขนาดใหญ่เกินไป ดังนั้น เรามักจะกรองข้อมูลที่เราต้องการจะนำเสนอก่อนที่จะนำมาสร้าง

library(ComplexHeatmap)

# Filter only significant DEGs gene (from EdgeR)  
DEGGene <- logcpm[rownames(sig\_gene$table),]  
  
Heatmap(DEGGene, row\_km = 6, name = "RNAseq\nWorkshop",  
 row\_names\_gp = gpar(fontsize = 3))

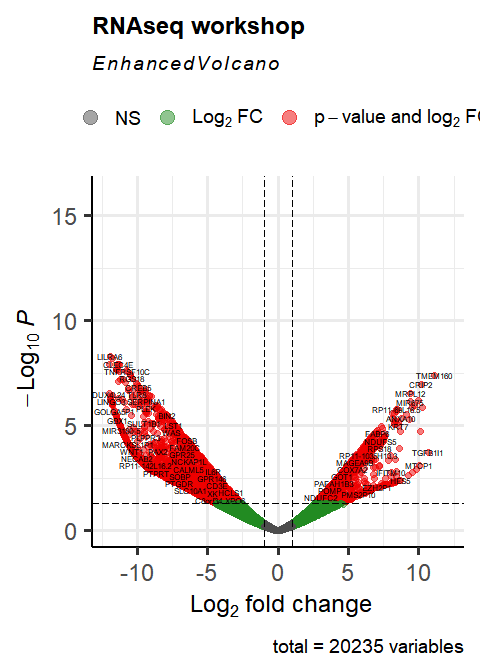


ตัวอย่างการใช้งานเพิ่มเติม: <https://jokergoo.github.io/ComplexHeatmap-reference/book/>

### Volcano plot

Volcano plot คือกราฟที่แสดงความแตกต่างของการแสดงออกของ RNA ระหว่างสองกลุ่ม โดยมีแกน x คือ log fold change และ y คือ -log10(p-value) เหตุผลที่แกน y ต้องเป็น -log10(p-value) เพื่อที่จะปรับค่า p-value ที่เป็นทศนิยมนั้นให้อยู่ในหลักจำนวนเต็ม ซึ่งจะทำให้ได้กราฟที่มีรูปร่างคล้ายภูเขาไฟหัวกลับ

library(EnhancedVolcano)  
  
# Create Volcano plot  
EnhancedVolcano(all\_gene$table, lab = rownames(all\_gene$table), x = "logFC", y = "PValue",   
 xlim = c(-12, 12), labSize = 2.0, pCutoff = 0.05,  
 title = "RNAseq workshop", max.overlaps = 50)



ค่าที่ cut-off ที่เราสนใจนั้นมักจะเป็นที่ logFC > 1-2, และ -log10(p-value) > 1.3-2 (p-value < 0.01-0.05)

ตัวอย่างการใช้งานเพิ่มเติม: <https://bioconductor.org/packages/release/bioc/vignettes/EnhancedVolcano/inst/doc/EnhancedVolcano.html>