Final Doc Research P2

Gabrielle Salamanca

July 13, 2023

This document is Part 2 of the final doc research. This will explain the work with the full dataset but with Y1. Everything is the same up until the columns part.

### Dataset

gene <- read\_excel("~/TCR-Project/Datasets/fullgenes.xlsx")  
attach(gene)

### SKAT Prep

Partial strings:

# v gene  
stringv1 <- "TRBV10-1"  
stringv2 <- "TRBV10-2"  
stringv3 <- "TRBV10-3"  
stringv4 <- "TRBV11-1"  
stringv5 <- "TRBV11-2"  
stringv6 <- "TRBV11-3"  
stringv7 <- "TRBV12-1"  
stringv8 <- "TRBV12-3"  
stringv9 <- "TRBV12-4"  
stringv10 <- "TRBV12-5"  
  
stringv11 <- "TRBV13"  
stringv12 <- "TRBV14"  
stringv13 <- "TRBV15"  
stringv14 <- "TRBV18"  
stringv15 <- "TRBV19"  
stringv16 <- "TRBV2"  
stringv17 <- "TRBV20-1"  
stringv18 <- "TRBV21-1"  
stringv19 <- "TRBV23-1"  
stringv20 <- "TRBV24-1"  
  
stringv21 <- "TRBV25-1"  
stringv22 <- "TRBV27"  
stringv23 <- "TRBV28"  
stringv24 <- "TRBV29-1"  
stringv25 <- "TRBV3-2"  
stringv26 <- "TRBV30"  
stringv27 <- "TRBV4-1"  
stringv28 <- "TRBV4-2"  
stringv29 <- "TRBV4-3"  
stringv30 <- "TRBV5-1"  
  
stringv31 <- "TRBV5-3"  
stringv32 <- "TRBV5-4"  
stringv33 <- "TRBV5-5"  
stringv34 <- "TRBV5-6"  
stringv35 <- "TRBV5-7"  
stringv36 <- "TRBV5-8"  
stringv37 <- "TRBV6-1"  
stringv38 <- "TRBV6-2"  
stringv39 <- "TRBV6-3"  
stringv40 <- "TRBV6-4"  
  
stringv41 <- "TRBV6-5"  
stringv42 <- "TRBV6-6"  
stringv43 <- "TRBV6-7"  
stringv44 <- "TRBV6-8"  
stringv45 <- "TRBV6-9"  
stringv46 <- "TRBV7-2"  
stringv47 <- "TRBV7-3"  
stringv48 <- "TRBV7-4"  
stringv49 <- "TRBV7-5"  
stringv50 <- "TRBV7-6"  
  
# j gene  
stringj1 <- "TRBJ1-1"  
stringj2 <- "TRBJ1-2"  
stringj3 <- "TRBJ1-3"  
stringj4 <- "TRBJ1-4"  
stringj5 <- "TRBJ1-5"  
stringj6 <- "TRBJ1-6"  
stringj7 <- "TRBJ2-1"  
stringj8 <- "TRBJ2-2"  
stringj9 <- "TRBJ2-3"  
stringj10 <- "TRBJ2-4"  
  
stringj11 <- "TRBJ2-5"  
stringj12 <- "TRBJ2-6"  
stringj13 <- "TRBJ2-7"

Columns:

# v gene  
colv1 <- grep(stringv1, names(gene), value = TRUE)  
colv2 <- grep(stringv2, names(gene), value = TRUE)  
colv3 <- grep(stringv3, names(gene), value = TRUE)  
colv4 <- grep(stringv4, names(gene), value = TRUE)  
colv5 <- grep(stringv5, names(gene), value = TRUE)  
colv6 <- grep(stringv6, names(gene), value = TRUE)  
colv7 <- grep(stringv7, names(gene), value = TRUE)  
colv8 <- grep(stringv8, names(gene), value = TRUE)  
colv9 <- grep(stringv9, names(gene), value = TRUE)  
colv10 <- grep(stringv10, names(gene), value = TRUE)  
  
colv11 <- grep(stringv11, names(gene), value = TRUE)  
colv12 <- grep(stringv12, names(gene), value = TRUE)  
colv13 <- grep(stringv13, names(gene), value = TRUE)  
colv14 <- grep(stringv14, names(gene), value = TRUE)  
colv15 <- grep(stringv15, names(gene), value = TRUE)  
colv16 <- grep(stringv16, names(gene), value = TRUE)  
colv17 <- grep(stringv17, names(gene), value = TRUE)  
colv18 <- grep(stringv18, names(gene), value = TRUE)  
colv19 <- grep(stringv19, names(gene), value = TRUE)  
colv20 <- grep(stringv20, names(gene), value = TRUE)  
  
colv21 <- grep(stringv21, names(gene), value = TRUE)  
colv22 <- grep(stringv22, names(gene), value = TRUE)  
colv23 <- grep(stringv23, names(gene), value = TRUE)  
colv24 <- grep(stringv24, names(gene), value = TRUE)  
colv25 <- grep(stringv25, names(gene), value = TRUE)  
colv26 <- grep(stringv26, names(gene), value = TRUE)  
colv27 <- grep(stringv27, names(gene), value = TRUE)  
colv28 <- grep(stringv28, names(gene), value = TRUE)  
colv29 <- grep(stringv29, names(gene), value = TRUE)  
colv30 <- grep(stringv30, names(gene), value = TRUE)  
  
colv31 <- grep(stringv31, names(gene), value = TRUE)  
colv32 <- grep(stringv32, names(gene), value = TRUE)  
colv33 <- grep(stringv33, names(gene), value = TRUE)  
colv34 <- grep(stringv34, names(gene), value = TRUE)  
colv35 <- grep(stringv35, names(gene), value = TRUE)  
colv36 <- grep(stringv36, names(gene), value = TRUE)  
colv37 <- grep(stringv37, names(gene), value = TRUE)  
colv38 <- grep(stringv38, names(gene), value = TRUE)  
colv39 <- grep(stringv39, names(gene), value = TRUE)  
colv40 <- grep(stringv40, names(gene), value = TRUE)  
  
colv41 <- grep(stringv41, names(gene), value = TRUE)  
colv42 <- grep(stringv42, names(gene), value = TRUE)  
colv43 <- grep(stringv43, names(gene), value = TRUE)  
colv44 <- grep(stringv44, names(gene), value = TRUE)  
colv45 <- grep(stringv45, names(gene), value = TRUE)  
colv46 <- grep(stringv46, names(gene), value = TRUE)  
colv47 <- grep(stringv47, names(gene), value = TRUE)  
colv48 <- grep(stringv48, names(gene), value = TRUE)  
colv49 <- grep(stringv49, names(gene), value = TRUE)  
colv50 <- grep(stringv50, names(gene), value = TRUE)  
  
# j gene  
colj1 <- grep(stringj1, names(gene), value = TRUE)  
colj2 <- grep(stringj2, names(gene), value = TRUE)  
colj3 <- grep(stringj3, names(gene), value = TRUE)  
colj4 <- grep(stringj4, names(gene), value = TRUE)  
colj5 <- grep(stringj5, names(gene), value = TRUE)  
colj6 <- grep(stringj6, names(gene), value = TRUE)  
colj7 <- grep(stringj7, names(gene), value = TRUE)  
colj8 <- grep(stringj8, names(gene), value = TRUE)  
colj9 <- grep(stringj9, names(gene), value = TRUE)  
colj10 <- grep(stringj10, names(gene), value = TRUE)  
  
colj11 <- grep(stringj11, names(gene), value = TRUE)  
colj12 <- grep(stringj12, names(gene), value = TRUE)  
colj13 <- grep(stringj13, names(gene), value = TRUE)

Now after the columns, we will make three subsets based on Y1. This had caused a problem when trying to run the for loops, which we’ll dive in during that part. Right now, we need to fill in the NAs for Y1.

set.na1 <- c(22)  
set.na2 <- c(94:109)  
Y1 <- gene$Y1  
Y1[set.na1] <- "active"  
Y1[set.na2] <- "healthy"

Now, we will make three subsets and null models based on Y1’s values: active, recovered, and healthy. The reason why we’re doing three pairs is because SKATBinary only takes 1’s and 0’s. So the pairs are: active/recovered (actRec), active/healthy (actHea), and recovered/healthy (recHea). The 1 is active for actRec, active for actHea, and recovered for recHea.

# subsets  
actRec <- subset(gene, Y1 == "active" | Y1 == "recovered")  
Y.ar <- rep(0, length(actRec$Y1))  
Y.ar[which(actRec$Y1 == "active")] = 1  
  
actHea <- subset(gene, Y1 == "active" | Y1 == "healthy")  
Y.ah <- rep(0, length(actHea$Y1))  
Y.ah[which(actHea$Y1 == "active")] = 1  
  
  
recHea <- subset(gene, Y1 == "recovered" | Y1 == "healthy")  
Y.rh <- rep(0, length(recHea$Y1))  
Y.rh[which(recHea$Y1 == "recovered")] = 1  
  
# null models  
obj.ar <- SKAT\_Null\_Model(Y.ar ~ 1, out\_type = "D")

## Sample size (non-missing y and X) = 69, which is < 2000. The small sample adjustment is applied!

obj.ah <- SKAT\_Null\_Model(Y.ah ~ 1, out\_type = "D")

## Sample size (non-missing y and X) = 73, which is < 2000. The small sample adjustment is applied!

obj.rh <- SKAT\_Null\_Model(Y.rh ~ 1, out\_type = "D")

## Sample size (non-missing y and X) = 42, which is < 2000. The small sample adjustment is applied!

Finally, we can run the for loops. Now, as I mentioned, there were problems I ran into when trying to run them initially. So, I did each one individually to see what were the warnings that were popping up. The first one was this: **missing value where TRUE/FALSE needed**. I’m not sure what this meant, but the rest of the warnings had to do with dimension problems.

The dimension problem possibly occurred, because the original subsets had only the gene dataset in mind, so that had the for loop stopping at one p-value. So, after some fiddling from the professor, she figured out that I needed add in a column index for the subset to finally run smoothly and fix the dimension problem.

To note, the p-values must be less than .

## Active/Recovered

Here’s the v gene for actRec:

p.ar <- rep(0,50)  
ar.val <- rep(0,13)  
  
# loop  
for (i in 1:50) {  
 col.idx <- get(paste0("colv", i,sep=""))  
 sub <- as.matrix(actRec[,col.idx])  
 out <- SKATBinary(sub, obj.ar, kernel = "linear.weighted")  
 p <- out$p.value  
 p.ar[i] <- p  
}

ar.v <- data.frame(cbind(c(1:50), p.ar))  
colnames(ar.v) <- c("vgene.idx","pvalue")  
ar.v

## vgene.idx pvalue  
## 1 1 4.559683e-04  
## 2 2 3.161450e-04  
## 3 3 9.995234e-06  
## 4 4 1.273809e-03  
## 5 5 9.885349e-03  
## 6 6 1.000000e+00  
## 7 7 1.000000e+00  
## 8 8 6.402726e-04  
## 9 9 4.915437e-03  
## 10 10 1.000000e+00  
## 11 11 1.207556e-01  
## 12 12 7.775790e-03  
## 13 13 7.369275e-05  
## 14 14 1.294020e-04  
## 15 15 3.747397e-04  
## 16 16 1.266989e-03  
## 17 17 3.986752e-02  
## 18 18 2.628744e-01  
## 19 19 1.000000e+00  
## 20 20 1.824169e-03  
## 21 21 1.000000e+00  
## 22 22 1.486869e-03  
## 23 23 5.862270e-02  
## 24 24 1.989339e-04  
## 25 25 1.000000e+00  
## 26 26 8.115029e-03  
## 27 27 9.923313e-04  
## 28 28 2.410232e-01  
## 29 29 2.011782e-01  
## 30 30 1.745082e-03  
## 31 31 6.095097e-04  
## 32 32 1.674971e-06  
## 33 33 1.719501e-05  
## 34 34 4.971909e-03  
## 35 35 1.003743e-04  
## 36 36 1.000000e+00  
## 37 37 4.665960e-03  
## 38 38 1.000000e+00  
## 39 39 3.508247e-03  
## 40 40 3.646228e-03  
## 41 41 2.623802e-03  
## 42 42 5.056259e-05  
## 43 43 1.000000e+00  
## 44 44 1.000000e+00  
## 45 45 5.363037e-03  
## 46 46 3.868574e-05  
## 47 47 1.193209e-04  
## 48 48 1.000000e+00  
## 49 49 8.354814e-03  
## 50 50 4.662387e-03

The p-values are far better when we do these kinds of pairs. The notable p-values are: colv1, colv2, colv3, colv4, colv5, colv8, colv9, colv11, colv12, colv13, colv14, colv15, colv16, colv17, colv18, colv20, colv22, colv23, colv24, colv26, colv27, colv28, colv29, colv30, colv31, colv32, colv33, colv34, colv35, colv37, colv39, colv40, colv41, colv42, colv45, colv46, colv47, colv49, and colv50.

Here’s the j gene for actRec:

for (i in 1:13) {  
 col.idx <- get(paste0("colj", i,sep=""))  
 sub <- as.matrix(actRec[,col.idx])  
 out <- SKATBinary(sub, obj.ar, kernel = "linear.weighted")  
 p <- out$p.value  
 ar.val[i] <- p  
}

ar.j <- data.frame(cbind(c(1:13),ar.val))  
colnames(ar.j) <- c("jgene.idx","p-value")  
ar.j

## jgene.idx p-value  
## 1 1 2.263020e-01  
## 2 2 6.122725e-02  
## 3 3 1.717002e-02  
## 4 4 2.544960e-05  
## 5 5 8.060797e-05  
## 6 6 1.257839e-04  
## 7 7 3.183579e-03  
## 8 8 1.951967e-02  
## 9 9 7.633952e-03  
## 10 10 1.274398e-05  
## 11 11 1.719514e-03  
## 12 12 1.940428e-03  
## 13 13 2.325683e-04

Here, all the j genes were notable.

## Active/Healthy

Here’s the v gene for actHea:

p.ah <- rep(0,50)  
ah.val <- rep(0,13)  
  
# loop  
for (i in 1:50) {  
 col.idx <- get(paste0("colv", i,sep=""))  
 sub <- as.matrix(actHea[,col.idx])  
 out <- SKATBinary(sub, obj.ah, kernel = "linear.weighted")  
 p <- out$p.value  
 p.ah[i] <- p  
}

ah.v <- data.frame(cbind(c(1:50), p.ah))  
colnames(ah.v) <- c("vgene.idx","pvalue")  
ah.v

## vgene.idx pvalue  
## 1 1 8.992880e-01  
## 2 2 1.000000e+00  
## 3 3 7.348150e-03  
## 4 4 3.713946e-03  
## 5 5 2.193674e-01  
## 6 6 1.000000e+00  
## 7 7 1.000000e+00  
## 8 8 3.911939e-01  
## 9 9 4.770466e-05  
## 10 10 1.000000e+00  
## 11 11 2.456669e-01  
## 12 12 4.734803e-02  
## 13 13 1.362616e-02  
## 14 14 7.512294e-02  
## 15 15 5.745317e-02  
## 16 16 5.749620e-03  
## 17 17 4.340714e-03  
## 18 18 3.532945e-01  
## 19 19 1.000000e+00  
## 20 20 1.600537e-01  
## 21 21 1.000000e+00  
## 22 22 1.630917e-01  
## 23 23 4.813755e-01  
## 24 24 1.590054e-03  
## 25 25 1.000000e+00  
## 26 26 6.279928e-04  
## 27 27 1.884421e-01  
## 28 28 3.542358e-01  
## 29 29 1.328105e-01  
## 30 30 2.801180e-02  
## 31 31 1.943095e-01  
## 32 32 3.719164e-01  
## 33 33 2.297951e-01  
## 34 34 5.448104e-02  
## 35 35 1.414282e-02  
## 36 36 1.000000e+00  
## 37 37 4.344294e-02  
## 38 38 1.000000e+00  
## 39 39 9.583909e-02  
## 40 40 7.630273e-03  
## 41 41 8.357911e-03  
## 42 42 1.979785e-02  
## 43 43 1.000000e+00  
## 44 44 1.000000e+00  
## 45 45 4.060754e-05  
## 46 46 4.796944e-01  
## 47 47 2.427434e-01  
## 48 48 2.174970e-02  
## 49 49 1.183233e-01  
## 50 50 3.853965e-03

The notable p-values are: colv1, colv3, colv4, colv5, colv8, colv9, colv11, colv12, colv13, colv14, colv15, colv16, colv17, colv18, colv20, colv22, colv23, colv24, colv26, colv27, colv28, colv29, colv30, colv31, colv32, colv33, colv34, colv35, colv37, colv39, colv40, colv41, colv42, colv45, colv46, colv47, colv49, and colv50.

Here’s the j gene for actHea:

for (i in 1:13) {  
 col.idx <- get(paste0("colj", i,sep=""))  
 sub <- as.matrix(actHea[,col.idx])  
 out <- SKATBinary(sub, obj.ah, kernel = "linear.weighted")  
 p <- out$p.value  
 ah.val[i] <- p  
}

ah.j <- data.frame(cbind(c(1:13),ah.val))  
colnames(ah.j) <- c("jgene.idx","p-value")  
ah.j

## jgene.idx p-value  
## 1 1 0.2104389394  
## 2 2 0.0096603839  
## 3 3 0.8767350099  
## 4 4 0.0681701842  
## 5 5 0.0471951732  
## 6 6 0.0660849810  
## 7 7 0.0006782331  
## 8 8 0.0008721123  
## 9 9 0.1173672771  
## 10 10 0.5000000000  
## 11 11 0.0098412427  
## 12 12 0.1712072117  
## 13 13 0.0150592306

Here, the notable j genes are: colj2, colj5, colj7, colj8, colj11, and colj13.

## Recovered/Healthy

Here’s the v gene for recHea:

p.rh <- rep(0,50)  
rh.val <- rep(0,13)  
  
# loop  
for (i in 1:50) {  
 col.idx <- get(paste0("colv", i,sep=""))  
 sub <- as.matrix(recHea[,col.idx])  
 out <- SKATBinary(sub, obj.rh, kernel = "linear.weighted")  
 p <- out$p.value  
 p.rh[i] <- p  
}

rh.v <- data.frame(cbind(c(1:50), p.rh))  
colnames(rh.v) <- c("vgene.idx","pvalue")  
rh.v

## vgene.idx pvalue  
## 1 1 5.102107e-02  
## 2 2 2.586076e-03  
## 3 3 1.274080e-02  
## 4 4 6.547944e-04  
## 5 5 1.211450e-01  
## 6 6 1.000000e+00  
## 7 7 1.000000e+00  
## 8 8 1.007064e-03  
## 9 9 3.179350e-03  
## 10 10 1.000000e+00  
## 11 11 1.845926e-02  
## 12 12 1.302581e-01  
## 13 13 1.246037e-01  
## 14 14 5.290429e-03  
## 15 15 1.417283e-05  
## 16 16 3.808133e-03  
## 17 17 1.253626e-03  
## 18 18 2.992410e-02  
## 19 19 1.000000e+00  
## 20 20 2.457415e-02  
## 21 21 1.000000e+00  
## 22 22 8.596411e-06  
## 23 23 1.669464e-02  
## 24 24 5.115295e-02  
## 25 25 1.000000e+00  
## 26 26 1.661752e-01  
## 27 27 6.715324e-05  
## 28 28 2.186395e-01  
## 29 29 1.540540e-01  
## 30 30 4.250155e-04  
## 31 31 2.213975e-02  
## 32 32 1.171462e-02  
## 33 33 7.099891e-02  
## 34 34 4.554505e-02  
## 35 35 7.620747e-02  
## 36 36 1.000000e+00  
## 37 37 3.804086e-01  
## 38 38 1.000000e+00  
## 39 39 6.094660e-02  
## 40 40 3.110257e-01  
## 41 41 5.462094e-02  
## 42 42 3.382496e-03  
## 43 43 1.000000e+00  
## 44 44 1.000000e+00  
## 45 45 5.334715e-01  
## 46 46 6.818814e-02  
## 47 47 1.035879e-04  
## 48 48 2.788012e-04  
## 49 49 3.201808e-04  
## 50 50 1.008724e-02

The notable p-values are: colv1, colv2, colv3, colv4, colv5, colv8, colv9, colv11, colv12, colv13, colv14, colv15, colov16, colv17, colv18, colv20, colv22, colv23, colv24, colv26, colv27, colv28, colv29, colv30, colv31, colv32, colv33, colv34, colv35, colv37, colv39, colv40, colv41, colv42, colv45, colv46, colv47, colv48, colv49, and colv50.

Here’s the j gene for recHea:

for (i in 1:13) {  
 col.idx <- get(paste0("colj", i,sep=""))  
 sub <- as.matrix(recHea[,col.idx])  
 out <- SKATBinary(sub, obj.rh, kernel = "linear.weighted")  
 p <- out$p.value  
 rh.val[i] <- p  
}

rh.j <- data.frame(cbind(c(1:13),rh.val))  
colnames(rh.j) <- c("jgene.idx","p-value")  
rh.j

## jgene.idx p-value  
## 1 1 6.438531e-01  
## 2 2 3.944860e-02  
## 3 3 2.380597e-05  
## 4 4 1.312574e-02  
## 5 5 3.017598e-03  
## 6 6 7.934859e-02  
## 7 7 2.528439e-03  
## 8 8 1.116607e-02  
## 9 9 8.186679e-02  
## 10 10 9.338607e-04  
## 11 11 3.599281e-02  
## 12 12 1.563543e-01  
## 13 13 5.252478e-03

Here, all the j genes are notable.

Now, let’s do the p-value adjustments on each one

## P-value Adjustment: actRec

Here’s for the actRec’s v genes:

# v gene  
ar.pv <- ar.v$pvalue  
ar.paV <- p.adjust(ar.pv, method = p.adjust.methods, n = length(ar.pv))  
ar.vRes <- data.frame(cbind(c(1:50), ar.paV))  
colnames(ar.vRes) <- c("vgene.idx","p-value")  
ar.vRes

## vgene.idx p-value  
## 1 1 1.732680e-02  
## 2 2 1.264580e-02  
## 3 3 4.897664e-04  
## 4 4 4.307761e-02  
## 5 5 1.779363e-01  
## 6 6 1.000000e+00  
## 7 7 1.000000e+00  
## 8 8 2.304981e-02  
## 9 9 1.212221e-01  
## 10 10 1.000000e+00  
## 11 11 1.000000e+00  
## 12 12 1.632916e-01  
## 13 13 3.316174e-03  
## 14 14 5.434882e-03  
## 15 15 1.461485e-02  
## 16 16 4.307761e-02  
## 17 17 6.777479e-01  
## 18 18 1.000000e+00  
## 19 19 1.000000e+00  
## 20 20 5.472507e-02  
## 21 21 1.000000e+00  
## 22 22 4.757981e-02  
## 23 23 9.379633e-01  
## 24 24 8.156292e-03  
## 25 25 1.000000e+00  
## 26 26 1.632916e-01  
## 27 27 3.473160e-02  
## 28 28 1.000000e+00  
## 29 29 1.000000e+00  
## 30 30 5.409754e-02  
## 31 31 2.255186e-02  
## 32 32 8.374853e-05  
## 33 33 8.253603e-04  
## 34 34 1.212221e-01  
## 35 35 4.416470e-03  
## 36 36 1.000000e+00  
## 37 37 1.212221e-01  
## 38 38 1.000000e+00  
## 39 39 9.823091e-02  
## 40 40 9.844814e-02  
## 41 41 7.609025e-02  
## 42 42 2.325879e-03  
## 43 43 1.000000e+00  
## 44 44 1.000000e+00  
## 45 45 1.212221e-01  
## 46 46 1.818230e-03  
## 47 47 5.130800e-03  
## 48 48 1.000000e+00  
## 49 49 1.632916e-01  
## 50 50 1.212221e-01

The notable v gene columns are: 1, 2, 3, 4, 5, 8, 9, 12, 13, 14, 15, 16, 17, 20, 22, 23, 24, 26, 27, 30, 31, 32, 33, 34, 35, 37, 39, 40, 41, 42, 45, 46, 47, 49, and 50.

And here’s for the j genes:

# j gene  
ar.pj <- ar.j$`p-value`  
ar.paJ <- p.adjust(ar.pj, method = p.adjust.methods, n = length(ar.pj))  
ar.jRes <- data.frame(cbind(c(1:13), ar.paJ))  
colnames(ar.jRes) <- c("jgene.idx","p-value")  
ar.jRes

## jgene.idx p-value  
## 1 1 0.2263020057  
## 2 2 0.1224544978  
## 3 3 0.0686800872  
## 4 4 0.0003053952  
## 5 5 0.0008866877  
## 6 6 0.0012578389  
## 7 7 0.0191014761  
## 8 8 0.0686800872  
## 9 9 0.0381697581  
## 10 10 0.0001656718  
## 11 11 0.0137561108  
## 12 12 0.0137561108  
## 13 13 0.0020931150

The notable j gene columns are: 4, 5, 6, 7, 9, 10, 11, 12, and 13.

## P-value Adjustment: actHea

Here’s for the actHea’s v genes:

# v gene  
ah.pv <- ah.v$pvalue  
ah.paV <- p.adjust(ah.pv, method = p.adjust.methods, n = length(ah.pv))  
ah.vRes <- data.frame(cbind(c(1:50), ah.paV))  
colnames(ah.vRes) <- c("vgene.idx","p-value")  
ah.vRes

## vgene.idx p-value  
## 1 1 1.000000000  
## 2 2 1.000000000  
## 3 3 0.308622318  
## 4 4 0.170841538  
## 5 5 1.000000000  
## 6 6 1.000000000  
## 7 7 1.000000000  
## 8 8 1.000000000  
## 9 9 0.002337528  
## 10 10 1.000000000  
## 11 11 1.000000000  
## 12 12 1.000000000  
## 13 13 0.531420147  
## 14 14 1.000000000  
## 15 15 1.000000000  
## 16 16 0.247233661  
## 17 17 0.190991427  
## 18 18 1.000000000  
## 19 19 1.000000000  
## 20 20 1.000000000  
## 21 21 1.000000000  
## 22 22 1.000000000  
## 23 23 1.000000000  
## 24 24 0.074732516  
## 25 25 1.000000000  
## 26 26 0.030143656  
## 27 27 1.000000000  
## 28 28 1.000000000  
## 29 29 1.000000000  
## 30 30 0.980412912  
## 31 31 1.000000000  
## 32 32 1.000000000  
## 33 33 1.000000000  
## 34 34 1.000000000  
## 35 35 0.537427323  
## 36 36 1.000000000  
## 37 37 1.000000000  
## 38 38 1.000000000  
## 39 39 1.000000000  
## 40 40 0.312841211  
## 41 41 0.334316456  
## 42 42 0.732520455  
## 43 43 1.000000000  
## 44 44 1.000000000  
## 45 45 0.002030377  
## 46 46 1.000000000  
## 47 47 1.000000000  
## 48 48 0.782989294  
## 49 49 1.000000000  
## 50 50 0.173428416

The notable v gene columns are: 9, 26, and 45.

And here’s for the j genes:

# j gene  
ah.pj <- ah.j$`p-value`  
ah.paJ <- p.adjust(ah.pj, method = p.adjust.methods, n = length(ah.pj))  
ah.jRes <- data.frame(cbind(c(1:13), ah.paJ))  
colnames(ah.jRes) <- c("jgene.idx","p-value")  
ah.jRes

## jgene.idx p-value  
## 1 1 0.68482885  
## 2 2 0.10626422  
## 3 3 1.00000000  
## 4 4 0.46259487  
## 5 5 0.37756139  
## 6 6 0.46259487  
## 7 7 0.00881703  
## 8 8 0.01046535  
## 9 9 0.58683639  
## 10 10 1.00000000  
## 11 11 0.10626422  
## 12 12 0.68482885  
## 13 13 0.13553308

The notable j gene columns are: 7 and 8.

## P-value Adjustment: recHea

Here’s for the recHea’s v genes:

# v gene  
rh.pv <- rh.v$pvalue  
rh.paV <- p.adjust(rh.pv, method = p.adjust.methods, n = length(rh.pv))  
rh.vRes <- data.frame(cbind(c(1:50), rh.paV))  
colnames(rh.vRes) <- c("vgene.idx","p-value")  
rh.vRes

## vgene.idx p-value  
## 1 1 1.0000000000  
## 2 2 0.1034430303  
## 3 3 0.4204465358  
## 4 4 0.0281561601  
## 5 5 1.0000000000  
## 6 6 1.0000000000  
## 7 7 1.0000000000  
## 8 8 0.0422967090  
## 9 9 0.1239946669  
## 10 10 1.0000000000  
## 11 11 0.5722371375  
## 12 12 1.0000000000  
## 13 13 1.0000000000  
## 14 14 0.1904554309  
## 15 15 0.0006944686  
## 16 16 0.1409009300  
## 17 17 0.0513986820  
## 18 18 0.8378747090  
## 19 19 1.0000000000  
## 20 20 0.7126504345  
## 21 21 1.0000000000  
## 22 22 0.0004298205  
## 23 23 0.5342284277  
## 24 24 1.0000000000  
## 25 25 1.0000000000  
## 26 26 1.0000000000  
## 27 27 0.0032233553  
## 28 28 1.0000000000  
## 29 29 1.0000000000  
## 30 30 0.0187006822  
## 31 31 0.6641923950  
## 32 32 0.3982970941  
## 33 33 1.0000000000  
## 34 34 1.0000000000  
## 35 35 1.0000000000  
## 36 36 1.0000000000  
## 37 37 1.0000000000  
## 38 38 1.0000000000  
## 39 39 1.0000000000  
## 40 40 1.0000000000  
## 41 41 1.0000000000  
## 42 42 0.1285348499  
## 43 43 1.0000000000  
## 44 44 1.0000000000  
## 45 45 1.0000000000  
## 46 46 1.0000000000  
## 47 47 0.0048686317  
## 48 48 0.0128248537  
## 49 49 0.0144081364  
## 50 50 0.3530534833

The notable v gene columns are: 4, 8, 15, 17 (debatable), 22, 27, 30, 47, 48, and 49.

And here’s for the j genes:

# j gene  
rh.pj <- rh.j$`p-value`  
rh.paJ <- p.adjust(rh.pj, method = p.adjust.methods, n = length(rh.pj))  
rh.jRes <- data.frame(cbind(c(1:13), rh.paJ))  
colnames(rh.jRes) <- c("jgene.idx","p-value")  
rh.jRes

## jgene.idx p-value  
## 1 1 0.6438530532  
## 2 2 0.2159568379  
## 3 3 0.0003094776  
## 4 4 0.0918801590  
## 5 5 0.0301759834  
## 6 6 0.3173943736  
## 7 7 0.0278128248  
## 8 8 0.0893285845  
## 9 9 0.3173943736  
## 10 10 0.0112063283  
## 11 11 0.2159568379  
## 12 12 0.3173943736  
## 13 13 0.0472723004

The notable j gene columns are: 3, 5, 7, 10, and 13.

### PCA Plots

Now, we will make PCA plots for each subset and the entire dataset to compare. We will use the p-adjusted values for this one.

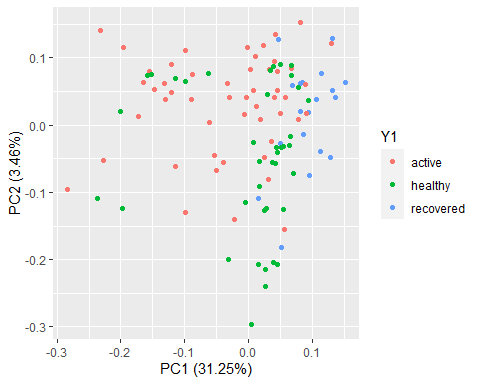
First, we need to do some prep for the plots.

# dataframe  
gene$Y1 <- Y1  
dfull <- gene[3:630]  
  
# v gene  
ar.v <- actRec[, c(colv1, colv2, colv3, colv4, colv5, colv8, colv9, colv12,   
 colv13, colv14, colv15, colv16, colv17, colv20, colv22,   
 colv23, colv24, colv26, colv27, colv30, colv31, colv32,   
 colv33, colv34, colv35, colv37, colv39, colv40, colv41,   
 colv42, colv45, colv46, colv47, colv49, colv50)]   
  
ah.v <- actHea[, c(colv9, colv26, colv45)]   
  
rh.v <- recHea[, c(colv4, colv8, colv15, colv17, colv22, colv27, colv30,   
 colv47, colv48, colv49)]  
  
# j gene  
ar.j <- actRec[, c(colj4, colj5, colj6, colj7, colj9, colj10, colj11, colj12,   
 colj13)]  
ar.j <- ar.j[, which(apply(ar.j, 2, var) != 0)]  
  
ah.j <- actHea[, c(colj7, colj8)]   
  
rh.j <- recHea[, c(colj3, colj5, colj7, colj10, colj13)]  
rh.j <- rh.j[, which(apply(rh.j, 2, var) != 0)]  
  
# pca res  
pcaFull <- prcomp(dfull, scale. = TRUE)  
# v gene  
pca.arV <- prcomp(ar.v, scale. = TRUE)  
pca.ahV <- prcomp(ah.v, scale. = TRUE)  
pca.rhV <- prcomp(rh.v, scale. = TRUE)   
  
# j gene  
pca.arJ <- prcomp(ar.j, scale. = TRUE)  
pca.ahJ <- prcomp(ah.j, scale. = TRUE)  
pca.rhJ <- prcomp(rh.j, scale. = TRUE)

Finally, we can do the v and j gene plots for each pair and one for the full dataset.

# plot  
ARplotV <- autoplot(pca.arV, data = actRec, colour = 'Y1')  
AHplotV <- autoplot(pca.ahV, data = actHea, colour = 'Y1')  
RHplotV <- autoplot(pca.rhV, data = recHea, colour = 'Y1')  
ARplotJ <- autoplot(pca.arJ, data = actRec, colour = 'Y1')  
AHplotJ <- autoplot(pca.ahJ, data = actHea, colour = 'Y1')  
RHplotJ <- autoplot(pca.rhJ, data = recHea, colour = 'Y1')  
autoplot(pcaFull, data = gene, colour = 'Y1')

**PCA Plot: fullgenes, Y1 coloring**



# arrange  
grid.arrange(ARplotV, ARplotJ, ncol = 2)

grid.arrange(AHplotV, AHplotJ, ncol = 2)

grid.arrange(RHplotV, RHplotJ, ncol = 2)

knitr::opts\_chunk$set(echo = TRUE)  
library(dplyr)  
library(factoextra)  
library(ggfortify)  
library(ggplot2)  
library(gridExtra)  
library(psych)  
library(readr)  
library(readxl)  
library(SKAT)  
library(tidyr)  
gene <- read\_excel("~/TCR-Project/Datasets/fullgenes.xlsx")  
attach(gene)  
# v gene  
stringv1 <- "TRBV10-1"  
stringv2 <- "TRBV10-2"  
stringv3 <- "TRBV10-3"  
stringv4 <- "TRBV11-1"  
stringv5 <- "TRBV11-2"  
stringv6 <- "TRBV11-3"  
stringv7 <- "TRBV12-1"  
stringv8 <- "TRBV12-3"  
stringv9 <- "TRBV12-4"  
stringv10 <- "TRBV12-5"  
  
stringv11 <- "TRBV13"  
stringv12 <- "TRBV14"  
stringv13 <- "TRBV15"  
stringv14 <- "TRBV18"  
stringv15 <- "TRBV19"  
stringv16 <- "TRBV2"  
stringv17 <- "TRBV20-1"  
stringv18 <- "TRBV21-1"  
stringv19 <- "TRBV23-1"  
stringv20 <- "TRBV24-1"  
  
stringv21 <- "TRBV25-1"  
stringv22 <- "TRBV27"  
stringv23 <- "TRBV28"  
stringv24 <- "TRBV29-1"  
stringv25 <- "TRBV3-2"  
stringv26 <- "TRBV30"  
stringv27 <- "TRBV4-1"  
stringv28 <- "TRBV4-2"  
stringv29 <- "TRBV4-3"  
stringv30 <- "TRBV5-1"  
  
stringv31 <- "TRBV5-3"  
stringv32 <- "TRBV5-4"  
stringv33 <- "TRBV5-5"  
stringv34 <- "TRBV5-6"  
stringv35 <- "TRBV5-7"  
stringv36 <- "TRBV5-8"  
stringv37 <- "TRBV6-1"  
stringv38 <- "TRBV6-2"  
stringv39 <- "TRBV6-3"  
stringv40 <- "TRBV6-4"  
  
stringv41 <- "TRBV6-5"  
stringv42 <- "TRBV6-6"  
stringv43 <- "TRBV6-7"  
stringv44 <- "TRBV6-8"  
stringv45 <- "TRBV6-9"  
stringv46 <- "TRBV7-2"  
stringv47 <- "TRBV7-3"  
stringv48 <- "TRBV7-4"  
stringv49 <- "TRBV7-5"  
stringv50 <- "TRBV7-6"  
  
# j gene  
stringj1 <- "TRBJ1-1"  
stringj2 <- "TRBJ1-2"  
stringj3 <- "TRBJ1-3"  
stringj4 <- "TRBJ1-4"  
stringj5 <- "TRBJ1-5"  
stringj6 <- "TRBJ1-6"  
stringj7 <- "TRBJ2-1"  
stringj8 <- "TRBJ2-2"  
stringj9 <- "TRBJ2-3"  
stringj10 <- "TRBJ2-4"  
  
stringj11 <- "TRBJ2-5"  
stringj12 <- "TRBJ2-6"  
stringj13 <- "TRBJ2-7"  
# v gene  
colv1 <- grep(stringv1, names(gene), value = TRUE)  
colv2 <- grep(stringv2, names(gene), value = TRUE)  
colv3 <- grep(stringv3, names(gene), value = TRUE)  
colv4 <- grep(stringv4, names(gene), value = TRUE)  
colv5 <- grep(stringv5, names(gene), value = TRUE)  
colv6 <- grep(stringv6, names(gene), value = TRUE)  
colv7 <- grep(stringv7, names(gene), value = TRUE)  
colv8 <- grep(stringv8, names(gene), value = TRUE)  
colv9 <- grep(stringv9, names(gene), value = TRUE)  
colv10 <- grep(stringv10, names(gene), value = TRUE)  
  
colv11 <- grep(stringv11, names(gene), value = TRUE)  
colv12 <- grep(stringv12, names(gene), value = TRUE)  
colv13 <- grep(stringv13, names(gene), value = TRUE)  
colv14 <- grep(stringv14, names(gene), value = TRUE)  
colv15 <- grep(stringv15, names(gene), value = TRUE)  
colv16 <- grep(stringv16, names(gene), value = TRUE)  
colv17 <- grep(stringv17, names(gene), value = TRUE)  
colv18 <- grep(stringv18, names(gene), value = TRUE)  
colv19 <- grep(stringv19, names(gene), value = TRUE)  
colv20 <- grep(stringv20, names(gene), value = TRUE)  
  
colv21 <- grep(stringv21, names(gene), value = TRUE)  
colv22 <- grep(stringv22, names(gene), value = TRUE)  
colv23 <- grep(stringv23, names(gene), value = TRUE)  
colv24 <- grep(stringv24, names(gene), value = TRUE)  
colv25 <- grep(stringv25, names(gene), value = TRUE)  
colv26 <- grep(stringv26, names(gene), value = TRUE)  
colv27 <- grep(stringv27, names(gene), value = TRUE)  
colv28 <- grep(stringv28, names(gene), value = TRUE)  
colv29 <- grep(stringv29, names(gene), value = TRUE)  
colv30 <- grep(stringv30, names(gene), value = TRUE)  
  
colv31 <- grep(stringv31, names(gene), value = TRUE)  
colv32 <- grep(stringv32, names(gene), value = TRUE)  
colv33 <- grep(stringv33, names(gene), value = TRUE)  
colv34 <- grep(stringv34, names(gene), value = TRUE)  
colv35 <- grep(stringv35, names(gene), value = TRUE)  
colv36 <- grep(stringv36, names(gene), value = TRUE)  
colv37 <- grep(stringv37, names(gene), value = TRUE)  
colv38 <- grep(stringv38, names(gene), value = TRUE)  
colv39 <- grep(stringv39, names(gene), value = TRUE)  
colv40 <- grep(stringv40, names(gene), value = TRUE)  
  
colv41 <- grep(stringv41, names(gene), value = TRUE)  
colv42 <- grep(stringv42, names(gene), value = TRUE)  
colv43 <- grep(stringv43, names(gene), value = TRUE)  
colv44 <- grep(stringv44, names(gene), value = TRUE)  
colv45 <- grep(stringv45, names(gene), value = TRUE)  
colv46 <- grep(stringv46, names(gene), value = TRUE)  
colv47 <- grep(stringv47, names(gene), value = TRUE)  
colv48 <- grep(stringv48, names(gene), value = TRUE)  
colv49 <- grep(stringv49, names(gene), value = TRUE)  
colv50 <- grep(stringv50, names(gene), value = TRUE)  
  
# j gene  
colj1 <- grep(stringj1, names(gene), value = TRUE)  
colj2 <- grep(stringj2, names(gene), value = TRUE)  
colj3 <- grep(stringj3, names(gene), value = TRUE)  
colj4 <- grep(stringj4, names(gene), value = TRUE)  
colj5 <- grep(stringj5, names(gene), value = TRUE)  
colj6 <- grep(stringj6, names(gene), value = TRUE)  
colj7 <- grep(stringj7, names(gene), value = TRUE)  
colj8 <- grep(stringj8, names(gene), value = TRUE)  
colj9 <- grep(stringj9, names(gene), value = TRUE)  
colj10 <- grep(stringj10, names(gene), value = TRUE)  
  
colj11 <- grep(stringj11, names(gene), value = TRUE)  
colj12 <- grep(stringj12, names(gene), value = TRUE)  
colj13 <- grep(stringj13, names(gene), value = TRUE)  
set.na1 <- c(22)  
set.na2 <- c(94:109)  
Y1 <- gene$Y1  
Y1[set.na1] <- "active"  
Y1[set.na2] <- "healthy"  
# subsets  
actRec <- subset(gene, Y1 == "active" | Y1 == "recovered")  
Y.ar <- rep(0, length(actRec$Y1))  
Y.ar[which(actRec$Y1 == "active")] = 1  
  
actHea <- subset(gene, Y1 == "active" | Y1 == "healthy")  
Y.ah <- rep(0, length(actHea$Y1))  
Y.ah[which(actHea$Y1 == "active")] = 1  
  
  
recHea <- subset(gene, Y1 == "recovered" | Y1 == "healthy")  
Y.rh <- rep(0, length(recHea$Y1))  
Y.rh[which(recHea$Y1 == "recovered")] = 1  
  
# null models  
obj.ar <- SKAT\_Null\_Model(Y.ar ~ 1, out\_type = "D")  
obj.ah <- SKAT\_Null\_Model(Y.ah ~ 1, out\_type = "D")  
obj.rh <- SKAT\_Null\_Model(Y.rh ~ 1, out\_type = "D")  
p.ar <- rep(0,50)  
ar.val <- rep(0,13)  
  
# loop  
for (i in 1:50) {  
 col.idx <- get(paste0("colv", i,sep=""))  
 sub <- as.matrix(actRec[,col.idx])  
 out <- SKATBinary(sub, obj.ar, kernel = "linear.weighted")  
 p <- out$p.value  
 p.ar[i] <- p  
}  
ar.v <- data.frame(cbind(c(1:50), p.ar))  
colnames(ar.v) <- c("vgene.idx","pvalue")  
ar.v  
for (i in 1:13) {  
 col.idx <- get(paste0("colj", i,sep=""))  
 sub <- as.matrix(actRec[,col.idx])  
 out <- SKATBinary(sub, obj.ar, kernel = "linear.weighted")  
 p <- out$p.value  
 ar.val[i] <- p  
}  
ar.j <- data.frame(cbind(c(1:13),ar.val))  
colnames(ar.j) <- c("jgene.idx","p-value")  
ar.j  
p.ah <- rep(0,50)  
ah.val <- rep(0,13)  
  
# loop  
for (i in 1:50) {  
 col.idx <- get(paste0("colv", i,sep=""))  
 sub <- as.matrix(actHea[,col.idx])  
 out <- SKATBinary(sub, obj.ah, kernel = "linear.weighted")  
 p <- out$p.value  
 p.ah[i] <- p  
}  
ah.v <- data.frame(cbind(c(1:50), p.ah))  
colnames(ah.v) <- c("vgene.idx","pvalue")  
ah.v  
for (i in 1:13) {  
 col.idx <- get(paste0("colj", i,sep=""))  
 sub <- as.matrix(actHea[,col.idx])  
 out <- SKATBinary(sub, obj.ah, kernel = "linear.weighted")  
 p <- out$p.value  
 ah.val[i] <- p  
}  
ah.j <- data.frame(cbind(c(1:13),ah.val))  
colnames(ah.j) <- c("jgene.idx","p-value")  
ah.j  
p.rh <- rep(0,50)  
rh.val <- rep(0,13)  
  
# loop  
for (i in 1:50) {  
 col.idx <- get(paste0("colv", i,sep=""))  
 sub <- as.matrix(recHea[,col.idx])  
 out <- SKATBinary(sub, obj.rh, kernel = "linear.weighted")  
 p <- out$p.value  
 p.rh[i] <- p  
}  
rh.v <- data.frame(cbind(c(1:50), p.rh))  
colnames(rh.v) <- c("vgene.idx","pvalue")  
rh.v  
for (i in 1:13) {  
 col.idx <- get(paste0("colj", i,sep=""))  
 sub <- as.matrix(recHea[,col.idx])  
 out <- SKATBinary(sub, obj.rh, kernel = "linear.weighted")  
 p <- out$p.value  
 rh.val[i] <- p  
}  
rh.j <- data.frame(cbind(c(1:13),rh.val))  
colnames(rh.j) <- c("jgene.idx","p-value")  
rh.j  
# v gene  
ar.pv <- ar.v$pvalue  
ar.paV <- p.adjust(ar.pv, method = p.adjust.methods, n = length(ar.pv))  
ar.vRes <- data.frame(cbind(c(1:50), ar.paV))  
colnames(ar.vRes) <- c("vgene.idx","p-value")  
ar.vRes  
# j gene  
ar.pj <- ar.j$`p-value`  
ar.paJ <- p.adjust(ar.pj, method = p.adjust.methods, n = length(ar.pj))  
ar.jRes <- data.frame(cbind(c(1:13), ar.paJ))  
colnames(ar.jRes) <- c("jgene.idx","p-value")  
ar.jRes  
# v gene  
ah.pv <- ah.v$pvalue  
ah.paV <- p.adjust(ah.pv, method = p.adjust.methods, n = length(ah.pv))  
ah.vRes <- data.frame(cbind(c(1:50), ah.paV))  
colnames(ah.vRes) <- c("vgene.idx","p-value")  
ah.vRes  
# j gene  
ah.pj <- ah.j$`p-value`  
ah.paJ <- p.adjust(ah.pj, method = p.adjust.methods, n = length(ah.pj))  
ah.jRes <- data.frame(cbind(c(1:13), ah.paJ))  
colnames(ah.jRes) <- c("jgene.idx","p-value")  
ah.jRes  
# v gene  
rh.pv <- rh.v$pvalue  
rh.paV <- p.adjust(rh.pv, method = p.adjust.methods, n = length(rh.pv))  
rh.vRes <- data.frame(cbind(c(1:50), rh.paV))  
colnames(rh.vRes) <- c("vgene.idx","p-value")  
rh.vRes  
# j gene  
rh.pj <- rh.j$`p-value`  
rh.paJ <- p.adjust(rh.pj, method = p.adjust.methods, n = length(rh.pj))  
rh.jRes <- data.frame(cbind(c(1:13), rh.paJ))  
colnames(rh.jRes) <- c("jgene.idx","p-value")  
rh.jRes  
# dataframe  
gene$Y1 <- Y1  
dfull <- gene[3:630]  
  
# v gene  
ar.v <- actRec[, c(colv1, colv2, colv3, colv4, colv5, colv8, colv9, colv12,   
 colv13, colv14, colv15, colv16, colv17, colv20, colv22,   
 colv23, colv24, colv26, colv27, colv30, colv31, colv32,   
 colv33, colv34, colv35, colv37, colv39, colv40, colv41,   
 colv42, colv45, colv46, colv47, colv49, colv50)]   
  
ah.v <- actHea[, c(colv9, colv26, colv45)]   
  
rh.v <- recHea[, c(colv4, colv8, colv15, colv17, colv22, colv27, colv30,   
 colv47, colv48, colv49)]  
  
# j gene  
ar.j <- actRec[, c(colj4, colj5, colj6, colj7, colj9, colj10, colj11, colj12,   
 colj13)]  
ar.j <- ar.j[, which(apply(ar.j, 2, var) != 0)]  
  
ah.j <- actHea[, c(colj7, colj8)]   
  
rh.j <- recHea[, c(colj3, colj5, colj7, colj10, colj13)]  
rh.j <- rh.j[, which(apply(rh.j, 2, var) != 0)]  
  
# pca res  
pcaFull <- prcomp(dfull, scale. = TRUE)  
# v gene  
pca.arV <- prcomp(ar.v, scale. = TRUE)  
pca.ahV <- prcomp(ah.v, scale. = TRUE)  
pca.rhV <- prcomp(rh.v, scale. = TRUE)   
  
# j gene  
pca.arJ <- prcomp(ar.j, scale. = TRUE)  
pca.ahJ <- prcomp(ah.j, scale. = TRUE)  
pca.rhJ <- prcomp(rh.j, scale. = TRUE)  
# plot  
ARplotV <- autoplot(pca.arV, data = actRec, colour = 'Y1')  
AHplotV <- autoplot(pca.ahV, data = actHea, colour = 'Y1')  
RHplotV <- autoplot(pca.rhV, data = recHea, colour = 'Y1')  
ARplotJ <- autoplot(pca.arJ, data = actRec, colour = 'Y1')  
AHplotJ <- autoplot(pca.ahJ, data = actHea, colour = 'Y1')  
RHplotJ <- autoplot(pca.rhJ, data = recHea, colour = 'Y1')  
autoplot(pcaFull, data = gene, colour = 'Y1')  
  
# arrange  
grid.arrange(ARplotV, ARplotJ, ncol = 2)  
grid.arrange(AHplotV, AHplotJ, ncol = 2)  
grid.arrange(RHplotV, RHplotJ, ncol = 2)