Multi-Omic Dimension Reduction

Jack Pattee

4/18/2023

```
###define local path to save out plot results
localPath = "~/Documents/Miscellaneous/SISG_plots/"

###Load multi-omic data breast cancer tumor data from The Cancer Genome Atlas
data(BRCA_data)

###Gene expression data
expressionDat = Data$Expression

###Methylation data
methylationDat = Data$Methylation

###micro-RNA data
mirnaDat = Data$miRNA

###run JIVE
###too slow for live session, but determines the tuning parameters
#jive(Data)
###with the tuning parameters already determined
results = jive(Data,method="given",rankJ=2,rankA=c(27,26,25))
```

```
## Running JIVE algorithm for ranks:
## joint rank: 2 , individual ranks: 27 26 25
## JIVE algorithm converged after  98  iterations.
```

```
###use TCGA biolinks to get clinical phenotype data
dataClin <- GDCquery_clinic(project = "TCGA-BRCA","clinical")
###get tumor subtype from 'morphology' variable
###8500/3: Invasive breast carcinoma of no special type (aka Ductal Carcinoma) (estimated ~75%)
###8520/3: Lobular carcinoma (estimated ~12 %)
###call the rest 'other'
###do some formatting and matching
sampFormatted = rep(NA,ncol(expressionDat))
for(i in 1:ncol(expressionDat)){
  tempSplit = strsplit(colnames(expressionDat)[i],"\\.")
  sampFormatted[i] = paste0(tempSplit[[1]][1],"-",tempSplit[[1]][2],"-",tempSplit[[1]][3])
}
m = match(sampFormatted,dataClin$submitter_id)
subtypeVec = dataClin$morphology[m]
###Define two variables
###First: 1 for DC, 0 for other
###Second: 1 for DC, 2 for lobular carcinoma, 0 for other
subtypeTwo = rep(0, length(subtypeVec))
```

```
subtypeTwo[subtypeVec=="8500/3"] = 1
subtypeThree = rep(0, length(subtypeVec))
subtypeThree[subtypeVec=="8500/3"] = 1
subtypeThree[subtypeVec=="8520/3"] = 2

###define survival endpoint, censored by 'days to last follow up'
survivalTime = dataClin$days_to_death[m]
survivalInd = rep(1, length(survivalTime))
survivalInd[which(is.na(survivalTime))] = 0
survivalTime[which(is.na(survivalTime))] = dataClin$days_to_last_follow_up[m][which(is.na(survivalTime)
###define age variable
age = dataClin$age_at_index[m]
```

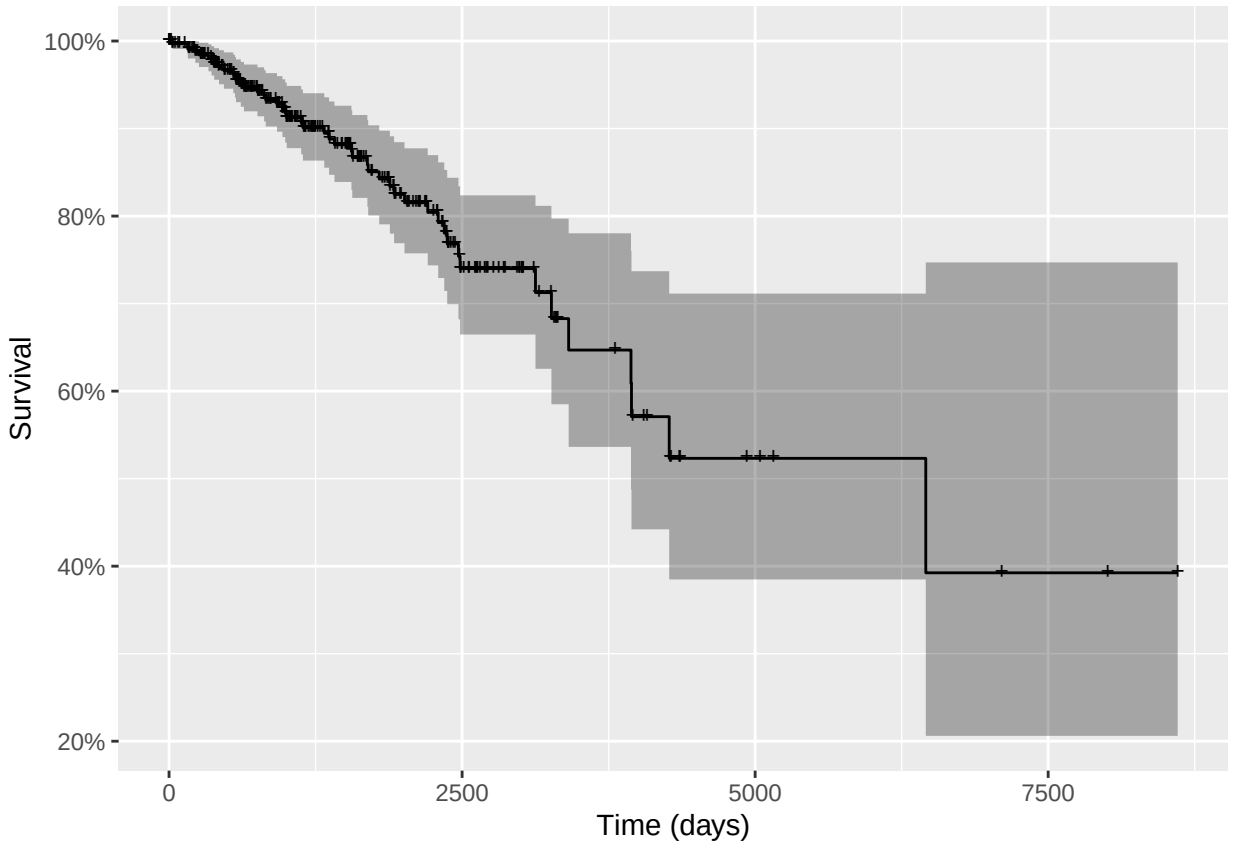
This document details an application of dimension reduction methods to multi-omic data from a sample of breast cancer tumors collected via The Cancer Genome Atlas. Data is available for 348 subjects. Data is available for three genomic variables: gene expression (with 645 measured genes), methylation (with 574 probes), and micro RNA (with 423 molecules). Our goal is to explore the distribution of this data via dimension reduction approaches. We will use the JIVE method, which considers 'omic-specific and mixed factors, and stacked PCA, which does not.

Although dimension reduction is a fundamentally unsupervised method, we demonstrate the utility of the dimension reduction approache with respect to some clinical variables. In particular, we consider tumor subtype and survival time. Tumor subtype is considered in two forms: dichotomized into 'ductal carcinoma' and 'other', and trichotomized into 'ductal carcinoma', 'lobular carcinoma', and 'other'. Survival is defined as the number of days to death, and is censored by the number of days to last follow up.

```
###describe the distribution of tumor subtype
print(summary(factor(subtypeThree, levels = c(0,1,2), labels = c("Other", "Ductal Carcinoma", "Lobular
```

```
##             Other  Ductal Carcinoma Lobular Carcinoma
##                24               297                27
```

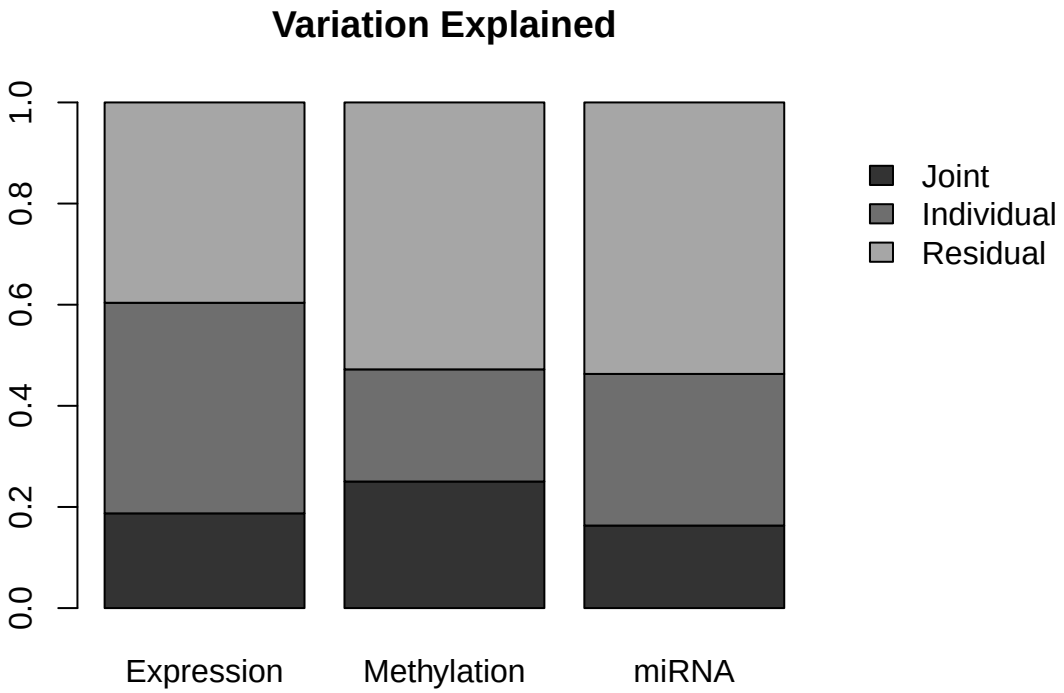
```
###plot the survival curve
autoplot(survfit(Surv(survivalTime,survivalInd)~1)) + xlab("Time (days)") + ylab("Survival")
```



Jive uses a permutation methodology to parse structure from residual noise, i.e., to determine the number of factors to use in the reconstruction of the joint structure and the per-omic individual structure. This permutation process can be run with the commented-out code in the code block below; however, to reduce processing time, the selected ranks have been input manually.

Show the variance explained in each of the omic data types.

```
###show variance explained
showVarExplained(results)
```



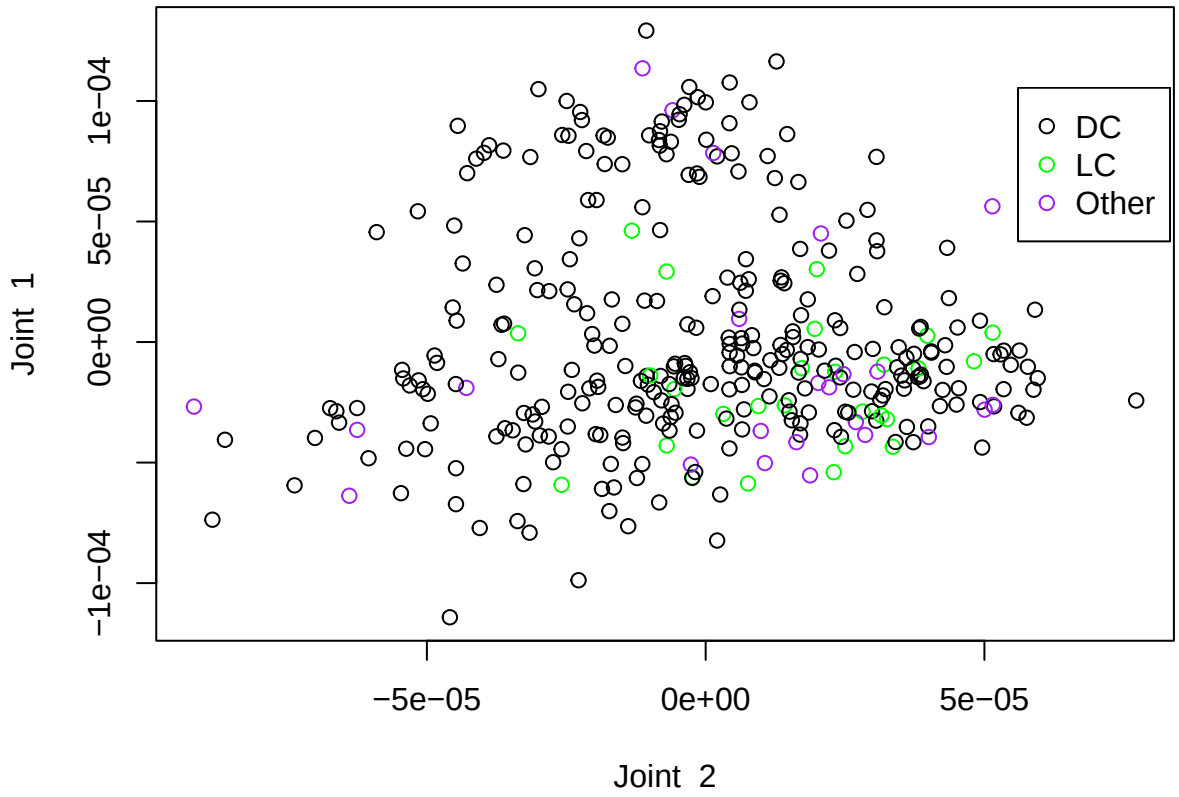
Plot the heatmap of joint and individual variance. This needs to be saved out to a file as it does not display well within R or RMarkdown. Red represents positive values, blue represents negative values.

```
png(paste0(localPath,"HeatmapsBRCA.png"),height=700,width=850)
showHeatmaps(results)
dev.off()
```

```
## pdf
##   2
```

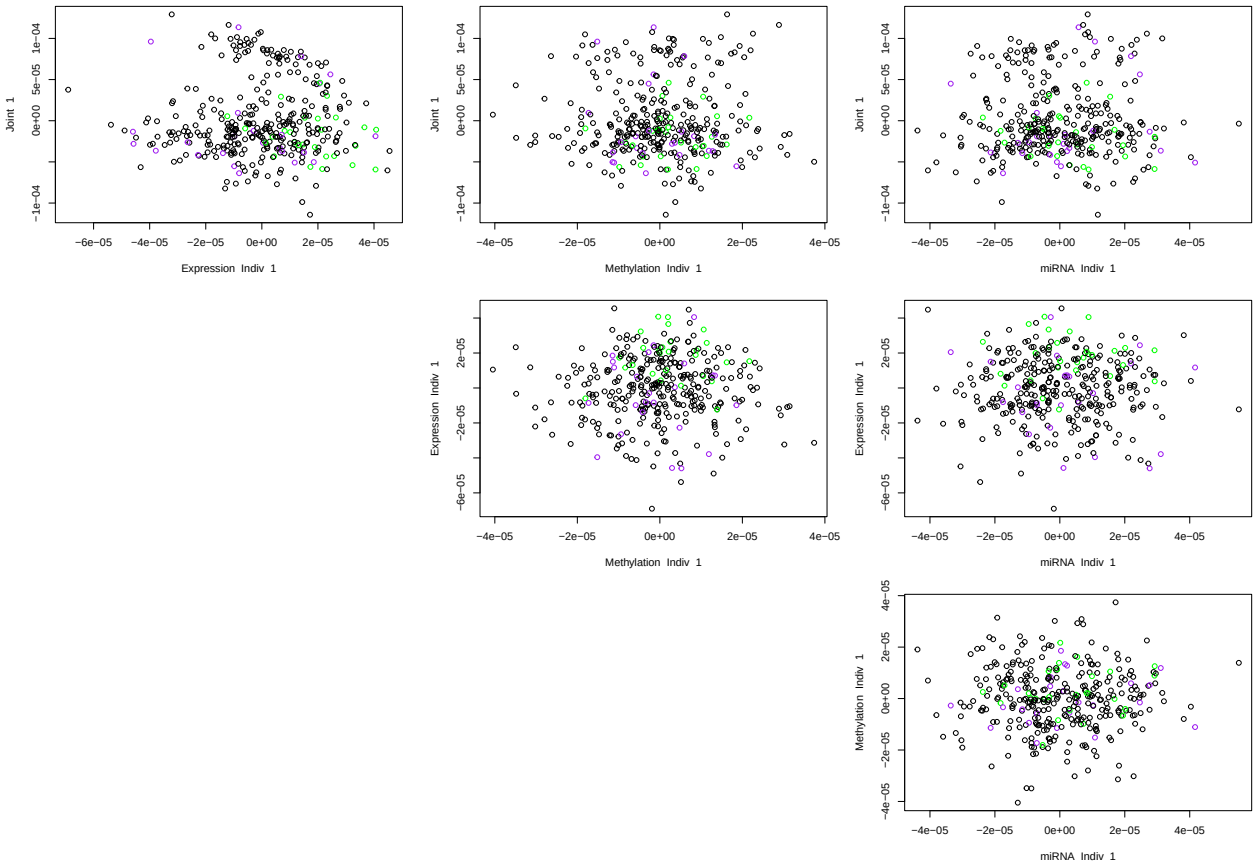
Visualize the joint variation projected into two dimensions, with points colored according to the trichotomous representation of tumor subtype.

```
Colors = rep('black',348)
Colors[subtypeThree==2] = 'green'
Colors[subtypeThree==0] = 'purple'
showPCA(results,n_joint=2,Colors=Colors)
legend(x = "topright", # Position
       legend = c("DC", "LC", "Other"),   # Legend texts
       pch = c(1,1,1),
       col = c("black","green","purple"))  # Line colors
```



Plot the 1-dimensional reduction of each data type-specific variation against one another to look for trends.

```
showPCA(results,n_joint=1,n_indiv=c(1,1,1),Colors=Colors)
```



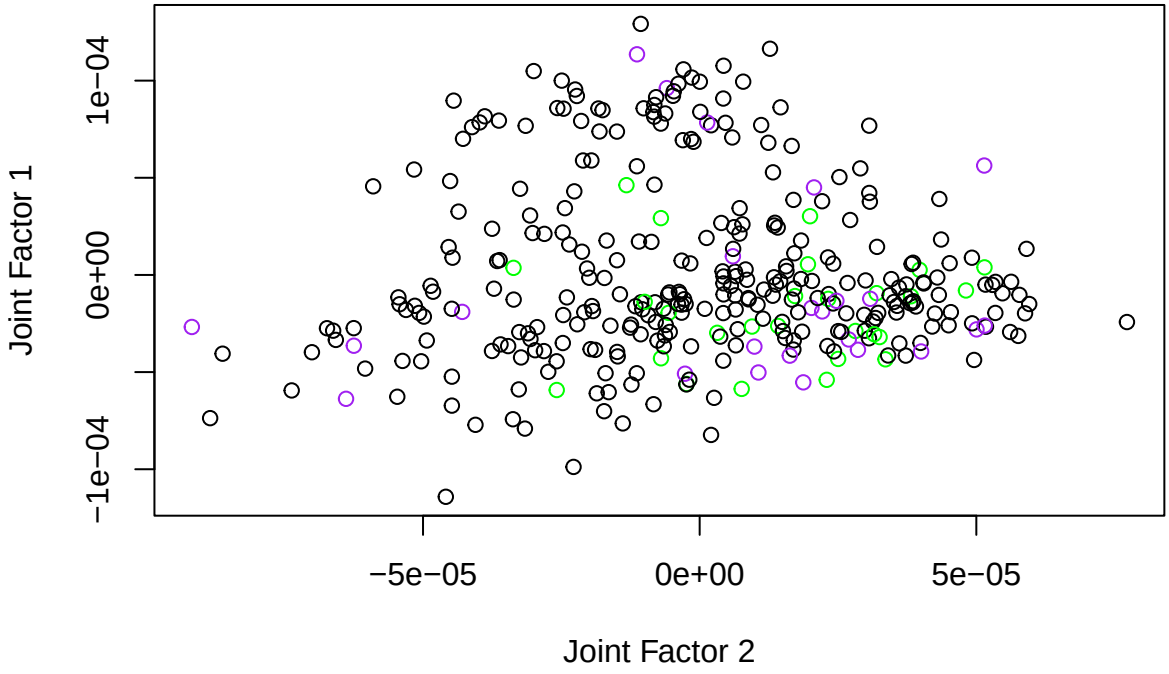
The above plots are generated via r.jive plotting functions. To have greater control of our plotting and modeling options, we extract the joint and individual factors from the JIVE decomposition as below. We will then investigate clustering and association modeling approaches, and compare how using JIVE factors perform versus PCs from stacked PCA.

```
rankJV <- results$rankJ
J<-numeric(0)
ng<-0

###get joint factors for use in plotting, clustering
for(j in 1:length(Data)){
  J <- rbind(J,results$joint[[j]]);
  ng<-c(ng,dim(results$joint[[j]])[1])
}
svd.o <- svd(J)
jV <- svd.o$v %*% diag(svd.o$d);
factorMat=jV[,1:rankJV]

###get factor loadings for individual data
###code ported from jive.r 'showPCA()' function
n_joint = 0
n_indiv = c(2,2,2)
l <- length(results$data)

nPCs = sum(n_indiv)
indivPCs = matrix(nrow=nPCs,ncol = dim(results$data[[1]])[2])
PC_names = rep('',nPCs)
```

```
for(i in 1:l){
  if(n_indiv[i]>0){
    SVD = svd(results$individual[[i]],nu=n_indiv[i],nv=n_indiv[i])
    indices = (n_joint+sum(n_indiv[0:(i-1)])+1):(n_joint+sum(n_indiv[0:i]))
    indivPCs[indices,] = diag(SVD$d)[1:n_indiv[i],1:n_indiv[i]]%*%t(SVD$v[,1:n_indiv[i]])
    PC_names[indices] = paste(names(results$data)[i]," Indiv ",1:n_indiv[i])
  }}
indivPCs = t(indivPCs)
colnames(indivPCs) = PC_names
```

To test whether this process was performed correctly, we can plot the first two joint factors against one another and compare to the 'showPCA()' plot from r.jive. These plots should be identical. We see that they are.

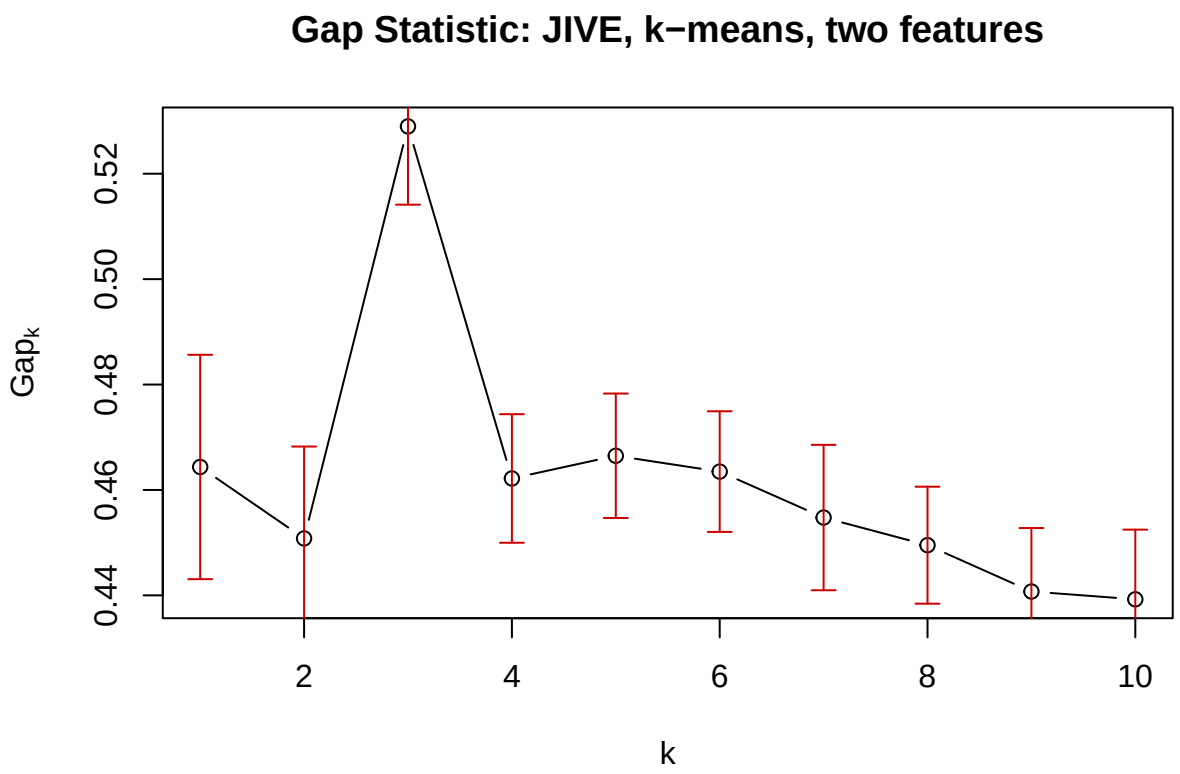
```
###this should match the 'showPCA' plot above
plot(factorMat[,2],factorMat[,1],col = Colors, xlab = "Joint Factor 2", ylab = "Joint Factor 1")
```



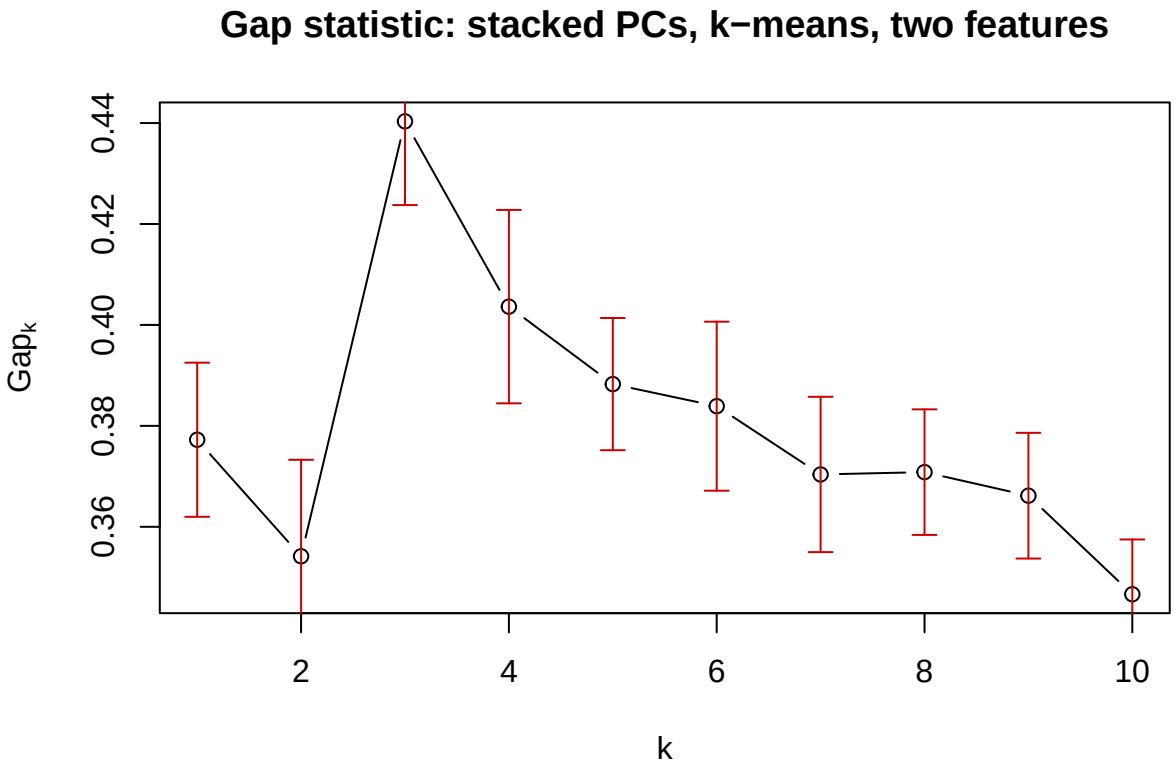
We will now explore some clustering analyses using the JIVE factors, and compare these approaches to clustering using the stacked principal components. First, we will compare clustering using the two joint factors from JIVE and the first two PCs from stacked principal components. We will first investigate k-means clustering and use the gap statistic to determine the optimal number of clusters.

```
###normalize JIVE factors for modeling
factorMat = scale(factorMat)
indivPCs = scale(indivPCs)

###scale and norm each data type before PCA
scaleExp = scale(t(expressionDat),scale = FALSE)
scaleMeth = scale(t(methylationDat),scale = FALSE)
scaleMi = scale(t(mirnaDat),scale = FALSE)
stackedData = cbind(scaleExp/sum(abs(scaleExp)),
                    scaleMeth/sum(abs(scaleMeth)),
                    scaleMi/sum(abs(scaleMi)))
```

```
stackedPcs = prcomp(stackedData, retx = TRUE)
stackedPcFactors = scale(stackedPcs$x)

jiveGap = clusGap(factorMat, FUN = kmeans, nstart = 10, K.max = 10, B = 10)
plot(jiveGap, main = "Gap Statistic: JIVE, k-means, two features")
```

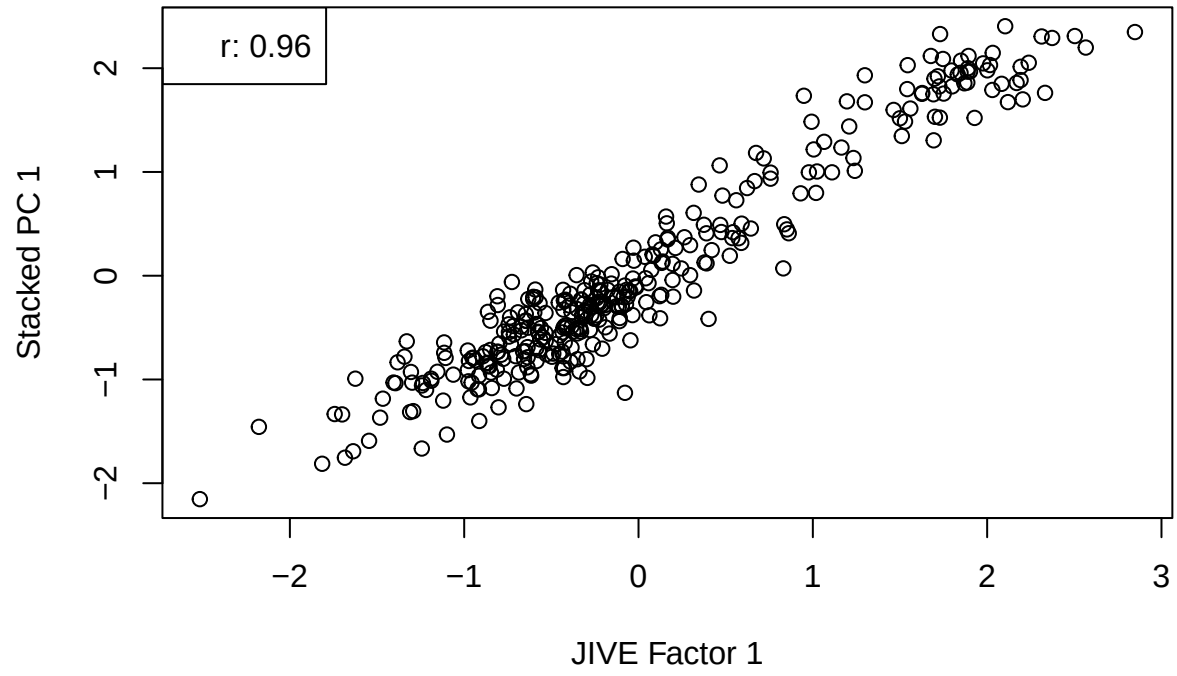



```
stackedGap = clusGap(stackedPcFactors[,1:2], FUN = kmeans, nstart = 10, K.max = 10, B = 10)
plot(stackedGap, main = "Gap statistic: stacked PCs, k-means, two features")
```

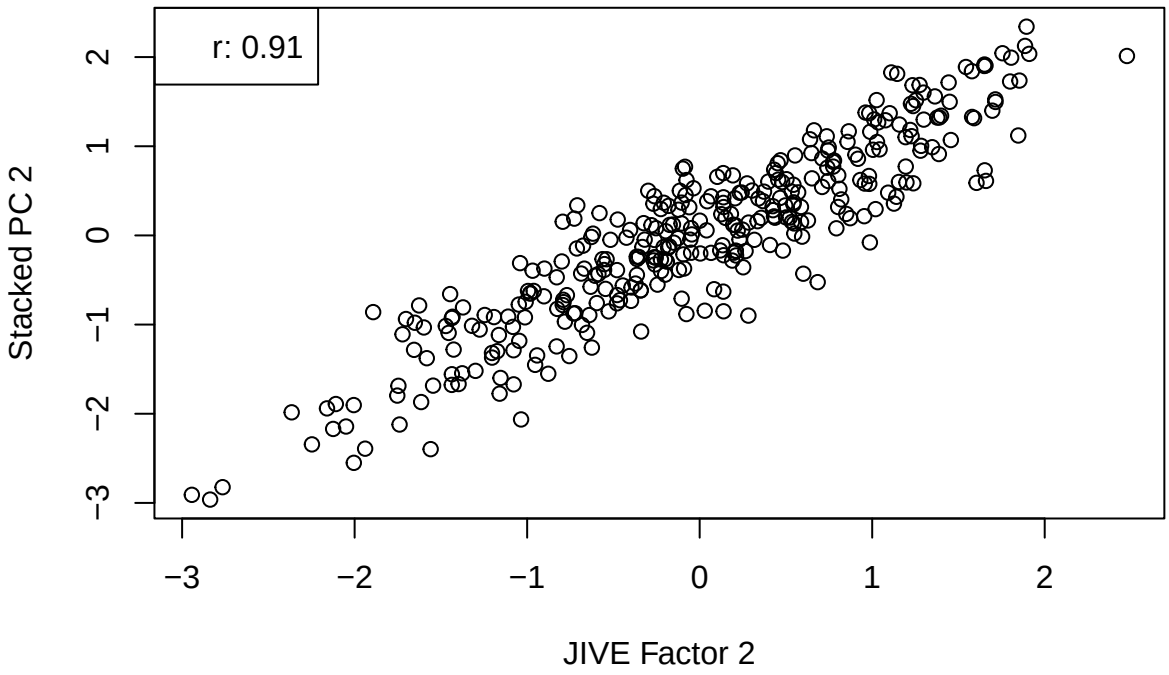


Plot the first and second joint factor against the first and second joint PC.

```
plot(factorMat[,1],stackedPcFactors[,1],xlab = "JIVE Factor 1", ylab = "Stacked PC 1")
legend("topleft", legend = paste0("r: ",round(cor(factorMat[,1],stackedPcFactors[,1]),2)))
```




```
plot(factorMat[,2],stackedPcFactors[,2],xlab = "JIVE Factor 2", ylab = "Stacked PC 2")
legend("topleft", legend = paste0("r: ",round(cor(factorMat[,2],stackedPcFactors[,2]),2)))
```



It appears that three clusters is optimal for both approaches. To investigate the 'quality' of these clusterings, we will compare the three cluster assignments to the trichotomous representation of tumor subtypes. Note: it could be that the clusterings are representative of structure from a different source, so this comparison is by no means an 'absolute' measure of clustering quality.

```
jiveClust = kmeans(factorMat,centers = 3, nstart = 5)
print(paste0("Adjusted Rand index for two-factor JIVE k-means clustering vs trichotomous tumor subtype:
```

```
## [1] "Adjusted Rand index for two-factor JIVE k-means clustering vs trichotomous tumor subtype: -0.02
```

```
table(data.frame(Truth = subtypeThree, JIVE = jiveClust$cluster))
```

```
##       JIVE
## Truth   1   2   3
##     0   5   4  15
##     1 111  64 122
##     2   6   2  19
```

```
stackedClust = kmeans(stackedPcFactors[,1:2],centers = 3, nstart = 5)
print(paste0("Adjusted Rand index for two-factor stacked PCA k-means clustering vs trichotomous tumor su
```

```
## [1] "Adjusted Rand index for two-factor stacked PCA k-means clustering vs trichotomous tumor subtype
```

```
table(data.frame(Truth = subtypeThree, Stacked = stackedClust$cluster))
```

```
##       Stacked
## Truth   1   2   3
##     0  15   5   4
##     1 126  98  73
##     2  22   3   2
```

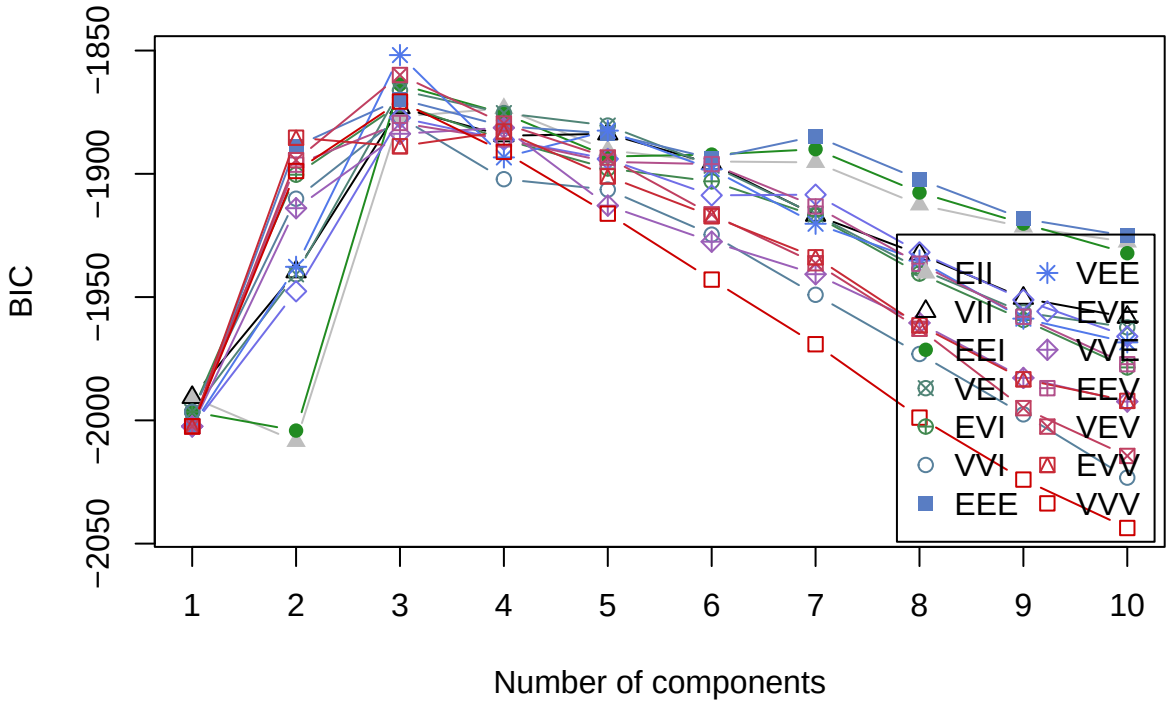
```
table(data.frame(Stacked = stackedClust$cluster, JIVE = jiveClust$cluster))
```

```
##         JIVE
## Stacked   1   2   3
##       1  23   0 140
```

```
##        2  94   0  12
##        3   5  70   4
```

K-means clustering assumes spherical cluster shape, which may not be the case in our data. Allow for a more flexible clustering with Gaussian mixture models, with the BIC used to select the number of clusters and the covariance structure.

```
jiveGmmBic = mclustBIC(factorMat, G = c(1:10))
plot(jiveGmmBic, main = "BIC: Jive, GMM, two features")
```




```
summary(jiveGmmBic)
```

```
## Best BIC values:
##              VEE,3        VEV,3       EEI,3
## BIC      -1851.841 -1859.913080 -1863.74280
## BIC diff     0.000    -8.072409   -11.90213
```

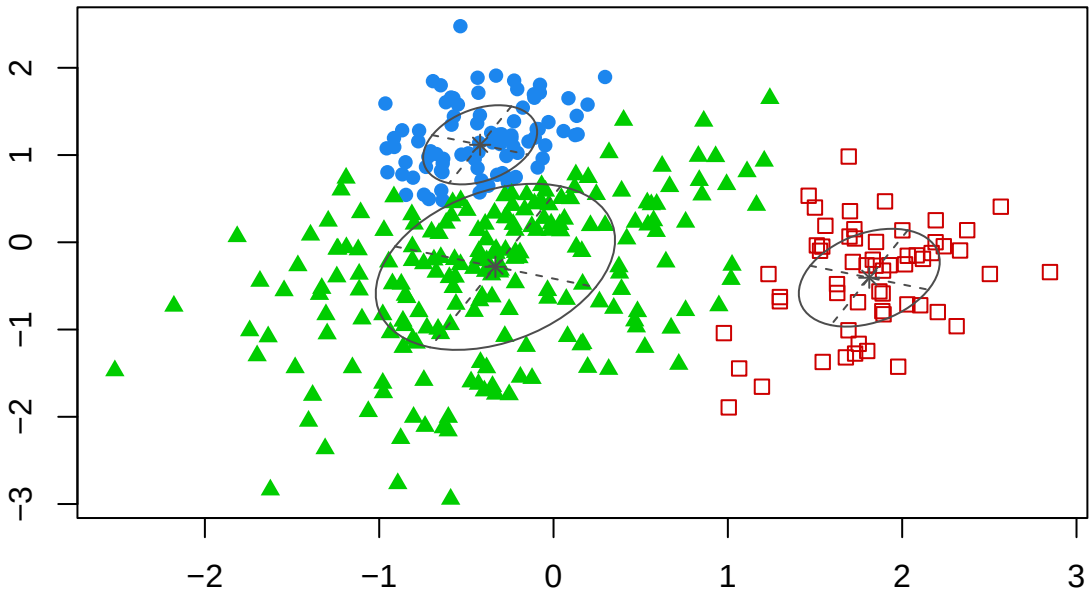
```
jiveGmmClust = Mclust(factorMat,x = jiveGmmBic)
print(paste0("Adjusted Rand index for two-factor JIVE GMM clustering vs trichotomous tumor subtype: ",r
```

```
## [1] "Adjusted Rand index for two-factor JIVE GMM clustering vs trichotomous tumor subtype: 0.01"
```

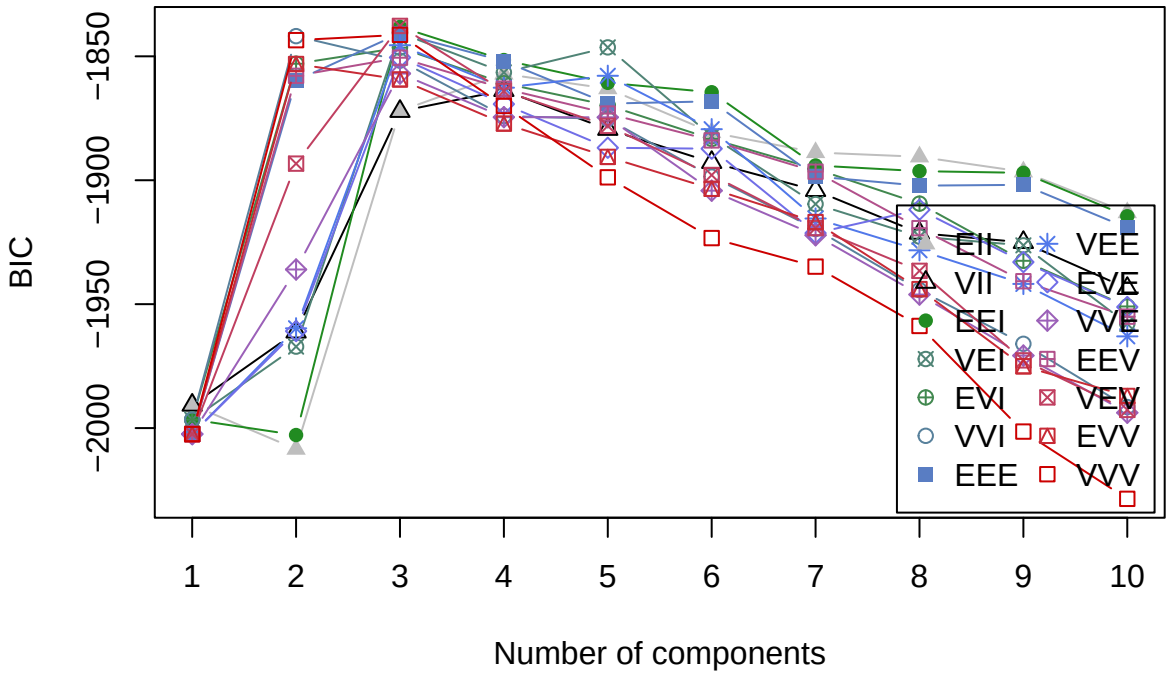
```
table(data.frame(Truth = subtypeThree, JIVE = jiveGmmClust$classification))
```

```
##      JIVE
## Truth   1   2   3
##     0   9   3  12
##     1  66  55 176
##     2  11   0  16
```

```
plot(jiveGmmClust, what = "classification", main = "GMM Classification: JIVE, two features")
```



```
pcGmmBic = mclustBIC(stackedPcFactors[,1:2], G = c(1:10))
plot(pcGmmBic, main = "BIC: Stacked PCA, GMM, two features")
```




```
summary(pcGmmBic)
```

```
## Best BIC values:
##              VEV,3         EEI,3         VEI,3
## BIC      -1837.738 -1838.1362559 -1840.573125
## BIC diff     0.000    -0.3984626    -2.835332
```

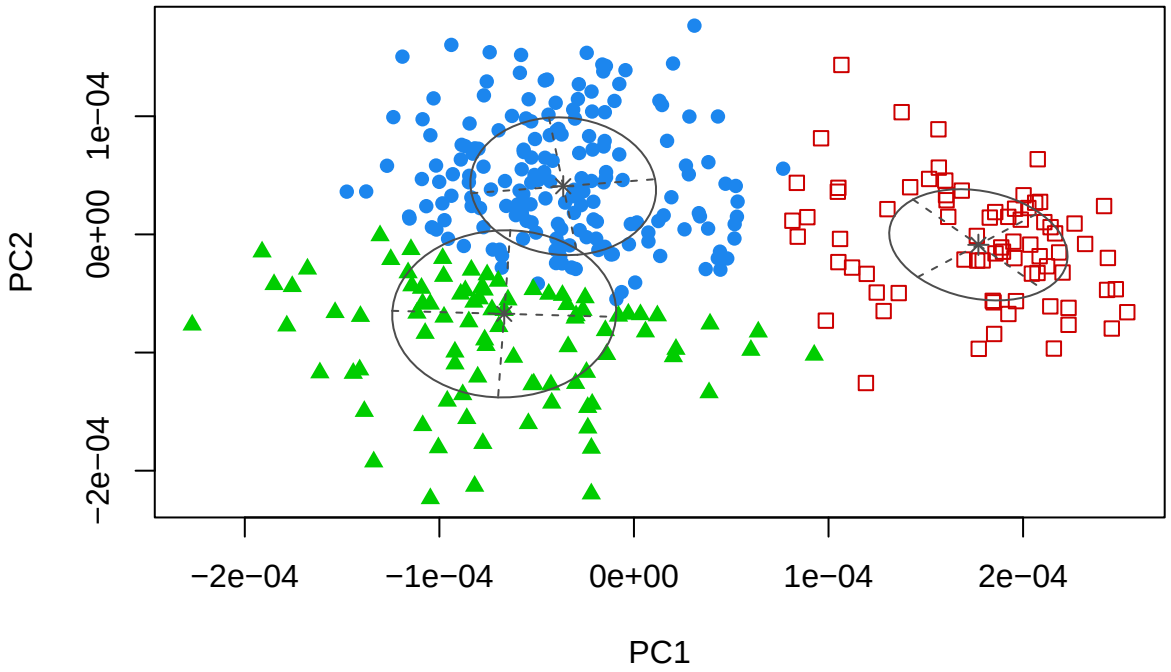
```
pcGmmClust = Mclust(stackedPcs$x[,1:2],x = pcGmmBic)
print(paste0("Adjusted Rand index for two-factor stacked PCA GMM clustering vs trichotomous tumor subty
```

```
## [1] "Adjusted Rand index for two-factor stacked PCA GMM clustering vs trichotomous tumor subtype: -0
```

```
table(data.frame(Truth = subtypeThree, Stacked = pcGmmClust$classification))
```

```
##       Stacked
## Truth   1   2   3
##     0  14   5   5
##     1 148  68  81
##     2  22   2   3
```

```
plot(pcGmmClust, what = "classification", main = "GMM Classification: stacked PCA, two features")
```




```
print("Compare clustering assignments for JIVE and stacked PCA: three clusters")
```

```
## [1] "Compare clustering assignments for JIVE and stacked PCA: three clusters"
```

```
tempDatThree = data.frame(Stacked = pcGmmClust$classification, JIVE = jiveGmmClust$classification)
table(tempDatThree)
```

```
##         JIVE
## Stacked   1   2   3
##       1  84   0 100
##       2   0  58  17
##       3   2   0  87
```

It appears that JIVE performs somewhat better than stacked PCs; however, the adjusted rand indices are still quite low. Let's investigate clustering with two clusters, and see how this compares to the dichotomous representation of tumor subtypes. We will again use Gaussian mixture modeling given the apparently non-spherical cluster structure.

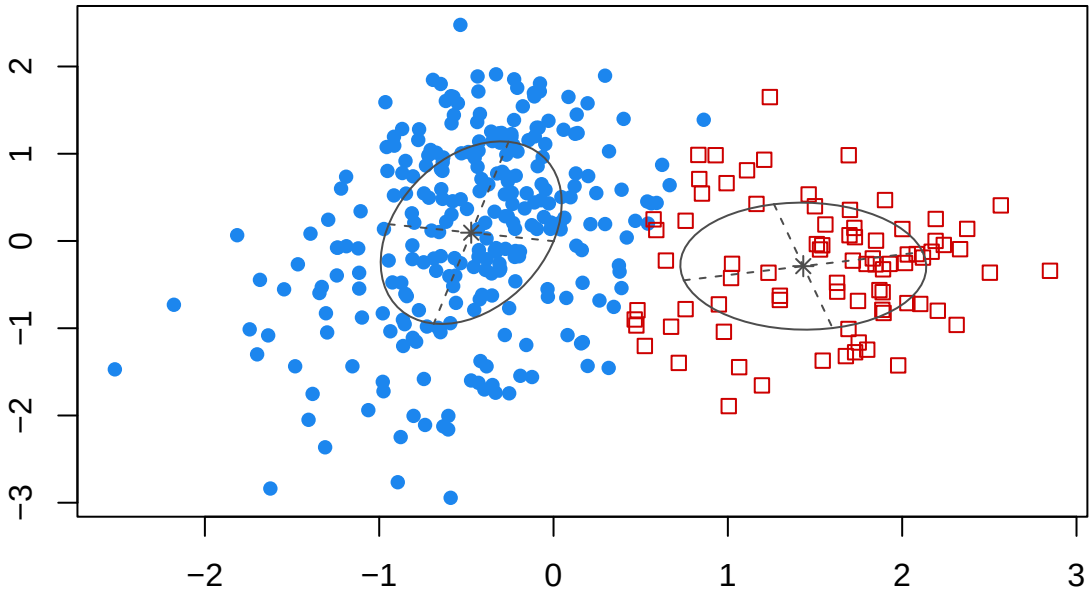
```
jiveGmmBicTwo = mclustBIC(factorMat,G = 2)
#summary(jiveGmmBicTwo)
jiveGmmClustTwo = Mclust(factorMat,x = jiveGmmBicTwo)
print(paste0("Adjusted Rand index for two-factor JIVE GMM clustering vs dichotomous tumor subtype: ",ro
```

```
## [1] "Adjusted Rand index for two-factor JIVE GMM clustering vs dichotomous tumor subtype: -0.05"
```

```
table(data.frame(Truth = subtypeTwo,JIVE = jiveGmmClustTwo$classification))
```

```
##       JIVE
## Truth   1   2
##     0  44   7
##     1 223  74
```

```
plot(jiveGmmClustTwo, what = "classification", main = "GMM Classification: JIVE, two features")
```



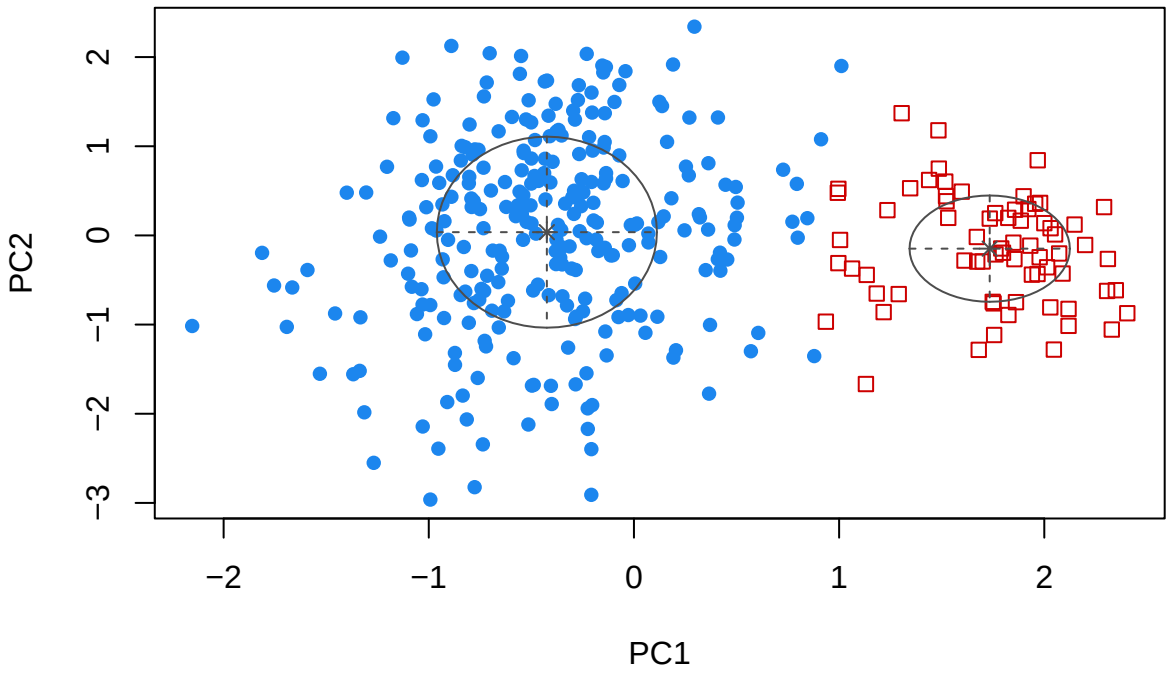

```
pcGmmBicTwo = mclustBIC(stackedPcFactors[,1:2],G = 2)
#summary(pcGmmBicTwo)
pcGmmClustTwo = Mclust(stackedPcFactors[,1:2],x = pcGmmBicTwo)
print(paste0("Adjusted Rand index for two-factor stacked PCA GMM clustering vs dichotomous tumor subtyp
```

```
## [1] "Adjusted Rand index for two-factor stacked PCA GMM clustering vs dichotomous tumor subtype: -0.
```

```
table(data.frame(Truth = subtypeTwo, Stacked = pcGmmClustTwo$classification))
```

```
##       Stacked
## Truth   1   2
##     0  47   4
##     1 232  65
```

```
plot(pcGmmClustTwo, what = "classification", main = "GMM Classification: stacked PCA, two features")
```




```
print("Compare clustering assignments for JIVE and stacked PCA: two clusters")
```

```
## [1] "Compare clustering assignments for JIVE and stacked PCA: two clusters"
```

```
tempDatTwo = data.frame(Stacked = pcGmmClustTwo$classification, JIVE = jiveGmmClustTwo$classification)
table(tempDatTwo)
```

```
##         JIVE
## Stacked   1   2
##       1 267  12
##       2   0  69
```

Let's see if adding additional factors changes the clustering assignment at all. We will add the first data-specific factor for each of the three data types in JIVE, and the three subsequent PCs for stacked PCA. Compare to the dichotomous representation of tumor subtype.

```
jive5Factors = cbind(factorMat,indivPCs[,c(1,3,5)])
jive5GmmBic = mclustBIC(jive5Factors, G = 2)
#summary(jive5GmmBic)
jive5GmmClust = Mclust(jive5Factors,x = jive5GmmBic)
print(paste0("Adjusted Rand index for five-factor JIVE GMM clustering vs dichotomous tumor subtype: ",r
```

```
## [1] "Adjusted Rand index for five-factor JIVE GMM clustering vs dichotomous tumor subtype: -0.07"
```

```
table(data.frame(Truth = subtypeTwo,JIVE = jive5GmmClust$classification))
```

```
##       JIVE
## Truth   1   2
##     0  48   3
##     1 245  52
```

```
pc5GmmBic = mclustBIC(stackedPcFactors[,1:5], G = 2)
#summary(pc5GmmBic)
pc5GmmClust = Mclust(stackedPcFactors[,1:5],x = pc5GmmBic)
print(paste0("Adjusted Rand index for five-factor stacked PCA GMM clustering vs dichotomous tumor subty
```

```
## [1] "Adjusted Rand index for five-factor stacked PCA GMM clustering vs dichotomous tumor subtype: -0
```

```
table(data.frame(Truth = subtypeTwo, Stacked = pc5GmmClust$classification))
```

```
##      Stacked
## Truth   1   2
##     0  43   8
##     1 215  82
```

Estimate logistic regression models to associate derived features with the dichotomous representation of tumor subtype. Age is also included as a covariate.

```
jiveDat = cbind.data.frame(subtypeTwo,jive5Factors,age)
colnames(jiveDat) = c("y","joint1","joint2","express1","meth1","mirna1","age")
jiveGlm = glm(y~.,data = jiveDat, family = "binomial")
print(summary(jiveGlm))
```

```
##
## Call:
## glm(formula = y ~ ., family = "binomial", data = jiveDat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8474   0.3207   0.4553   0.5867   1.1793
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.321031   0.768115   3.022  0.00251 **
## joint1       0.427849   0.196638   2.176  0.02957 *
## joint2      -0.427106   0.162131  -2.634  0.00843 **
## express1    -0.469966   0.165490  -2.840  0.00451 **
## meth1       -0.150825   0.171172  -0.881  0.37825
## mirna1      -0.195113   0.156440  -1.247  0.21232
## age         -0.005947   0.012763  -0.466  0.64127
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 290.01  on 347  degrees of freedom
## Residual deviance: 265.58  on 341  degrees of freedom
## AIC: 279.58
##
## Number of Fisher Scoring iterations: 5
```

```
pcDat = cbind.data.frame(subtypeTwo, stackedPcFactors[,1:5],age)
colnames(pcDat) = c("y",paste0("pc",1:5),"age")
pcMod = glm(y~.,data = pcDat, family = "binomial")
summary(pcMod)
```

```
##
## Call:
## glm(formula = y ~ ., family = "binomial", data = pcDat)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7786   0.3055   0.4485   0.5835   1.3330
```

```
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.436856   0.772890   3.153  0.00162 **
## pc1          0.415533   0.196344   2.116  0.03432 *
## pc2         -0.507048   0.161721  -3.135  0.00172 **
## pc3          0.158180   0.171085   0.925  0.35519
## pc4         -0.424836   0.163004  -2.606  0.00915 **
## pc5         -0.234584   0.166138  -1.412  0.15796
## age         -0.007699   0.012807  -0.601  0.54771
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 290.01  on 347  degrees of freedom
## Residual deviance: 262.98  on 341  degrees of freedom
## AIC: 276.98
##
## Number of Fisher Scoring iterations: 5
```

Estimate cox proportional hazards models to associate derived features with survival outcome. Age is also included as a covariate.

```
jiveDatSurv = cbind.data.frame(survivalTime,survivalInd,jive5Factors,age)
colnames(jiveDatSurv) = c("survTime","survInd","joint1","joint2","express1","meth1","mirna1","age")
jiveModPh = coxph(Surv(survTime,survInd)~., data = jiveDatSurv)
summary(jiveModPh)
```

```
## Call:
## coxph(formula = Surv(survTime, survInd) ~ ., data = jiveDatSurv)
##
##   n= 348, number of events= 48
##
##              coef exp(coef) se(coef)      z Pr(>|z|)
## joint1   -0.01122   0.98884  0.15451 -0.073  0.94211
## joint2   -0.07668   0.92619  0.13862 -0.553  0.58017
## express1 -0.13460   0.87407  0.15939 -0.844  0.39841
## meth1    -0.09847   0.90622  0.14658 -0.672  0.50173
## mirna1   -0.16111   0.85120  0.13828 -1.165  0.24399
## age       0.03760   1.03832  0.01264  2.976  0.00292 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##          exp(coef) exp(-coef) lower .95 upper .95
## joint1      0.9888     1.0113    0.7305     1.339
## joint2      0.9262     1.0797    0.7058     1.215
## express1    0.8741     1.1441    0.6395     1.195
## meth1       0.9062     1.1035    0.6799     1.208
## mirna1      0.8512     1.1748    0.6491     1.116
## age         1.0383     0.9631    1.0129     1.064
##
## Concordance= 0.648  (se = 0.046 )
## Likelihood ratio test= 14.74  on 6 df,   p=0.02
## Wald test            = 14.52  on 6 df,   p=0.02
```

```
## Score (logrank) test = 15.08  on 6 df,   p=0.02
```

```
pcDatSurv = cbind.data.frame(survivalTime,survivalInd,stackedPcFactors[,1:5],age)
colnames(pcDatSurv) = c("survTime","survInd",paste0("pc",1:5),"age")
pcModPh = coxph(Surv(survTime,survInd)~., data = pcDatSurv)
summary(pcModPh)
```

```
## Call:
## coxph(formula = Surv(survTime, survInd) ~ ., data = pcDatSurv)
##
##   n= 348, number of events= 48
##
##         coef exp(coef) se(coef)      z Pr(>|z|)
## pc1  0.08148   1.08489  0.15157  0.538  0.59087
## pc2 -0.17008   0.84360  0.14488 -1.174  0.24042
## pc3  0.06455   1.06668  0.15114  0.427  0.66931
## pc4 -0.13693   0.87203  0.16454 -0.832  0.40531
## pc5 -0.18746   0.82906  0.14829 -1.264  0.20616
## age  0.03965   1.04045  0.01296  3.059  0.00222 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##     exp(coef) exp(-coef) lower .95 upper .95
## pc1    1.0849     0.9217    0.8061     1.460
## pc2    0.8436     1.1854    0.6351     1.121
## pc3    1.0667     0.9375    0.7932     1.434
## pc4    0.8720     1.1467    0.6316     1.204
## pc5    0.8291     1.2062    0.6200     1.109
## age    1.0405     0.9611    1.0143     1.067
##
## Concordance= 0.665  (se = 0.041 )
## Likelihood ratio test= 15.86  on 6 df,   p=0.01
## Wald test            = 15.08  on 6 df,   p=0.02
## Score (logrank) test = 15.72  on 6 df,   p=0.02
```