2021 SISBID Clustering Demos

Genevera I. Allen, Yufeng Liu, Hui Shen, Camille Little

Load packages

```
library(ggplot2)
library(animation)
library(ISLR)
library(clustRviz)
library(sigclust)
library(kknn)
```

K-means Clustering

1. Data set 1 - Simulated Data

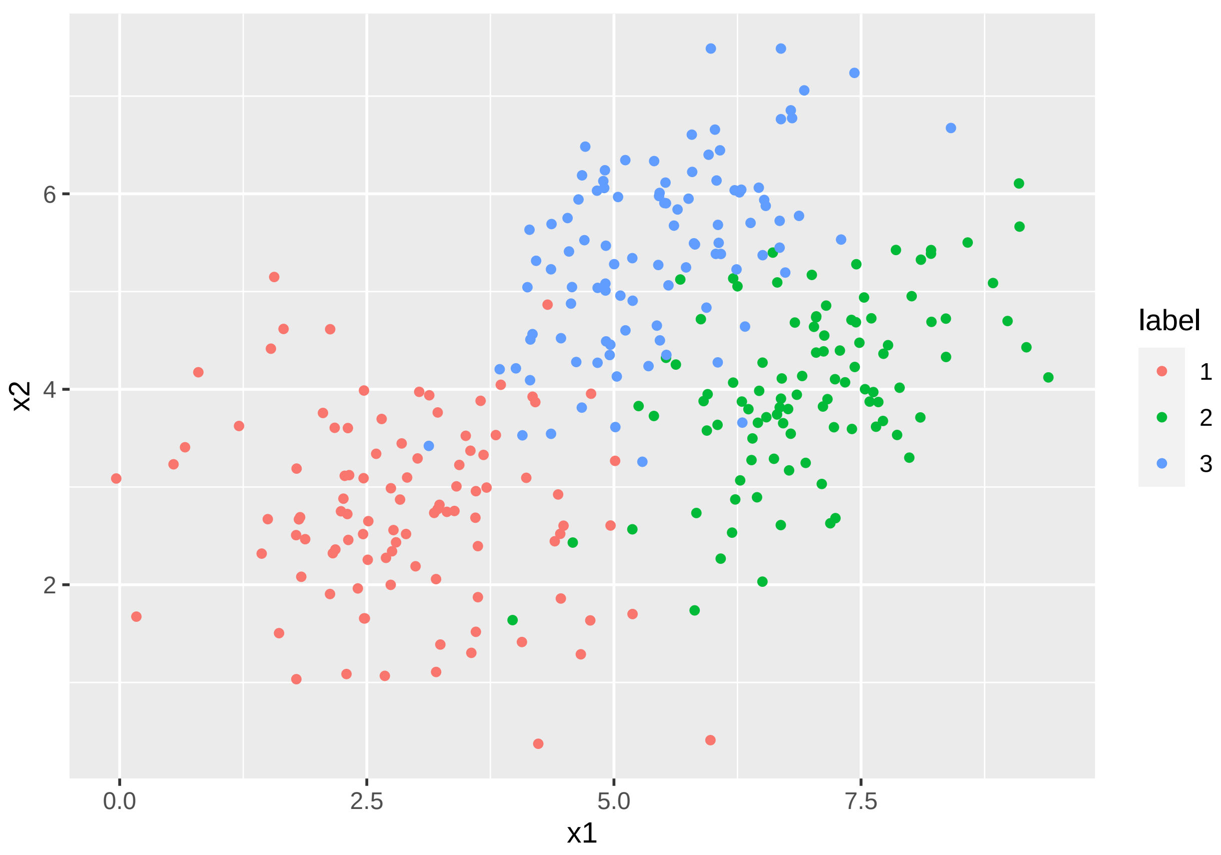
- small simulated data set to demonstrate concepts with k-means clustering

Simulate data: generate data from a mixture of three normal distribution

```
n = 300
mu1 = c(3,3)
mu2 = c(7,4)
mu3 = c(5.5,5.5)
Sig = matrix(c(1,.5,.5,1),2,2)
x1 = t(matrix(mu1,2,n/3)) + matrix(rnorm(n*2/3),n/3,2)
xx = matrix(rnorm(n*2/3),n/3,2)
x2 = t(matrix(mu2,2,n/3)) + xx%*%chol(Sig)
xx = matrix(rnorm(n*2/3),n/3,2)
x3 = t(matrix(mu3,2,n/3)) + xx%*%chol(Sig)
X = rbind(x1,x2,x3)
Y = c(rep(1,n/3),rep(2,n/3),rep(3,n/3))
Data = cbind(X,Y)
Data = data.frame(Data)
colnames(Data) = c("x1","x2","label")
Data$label = factor(Data$label)
```

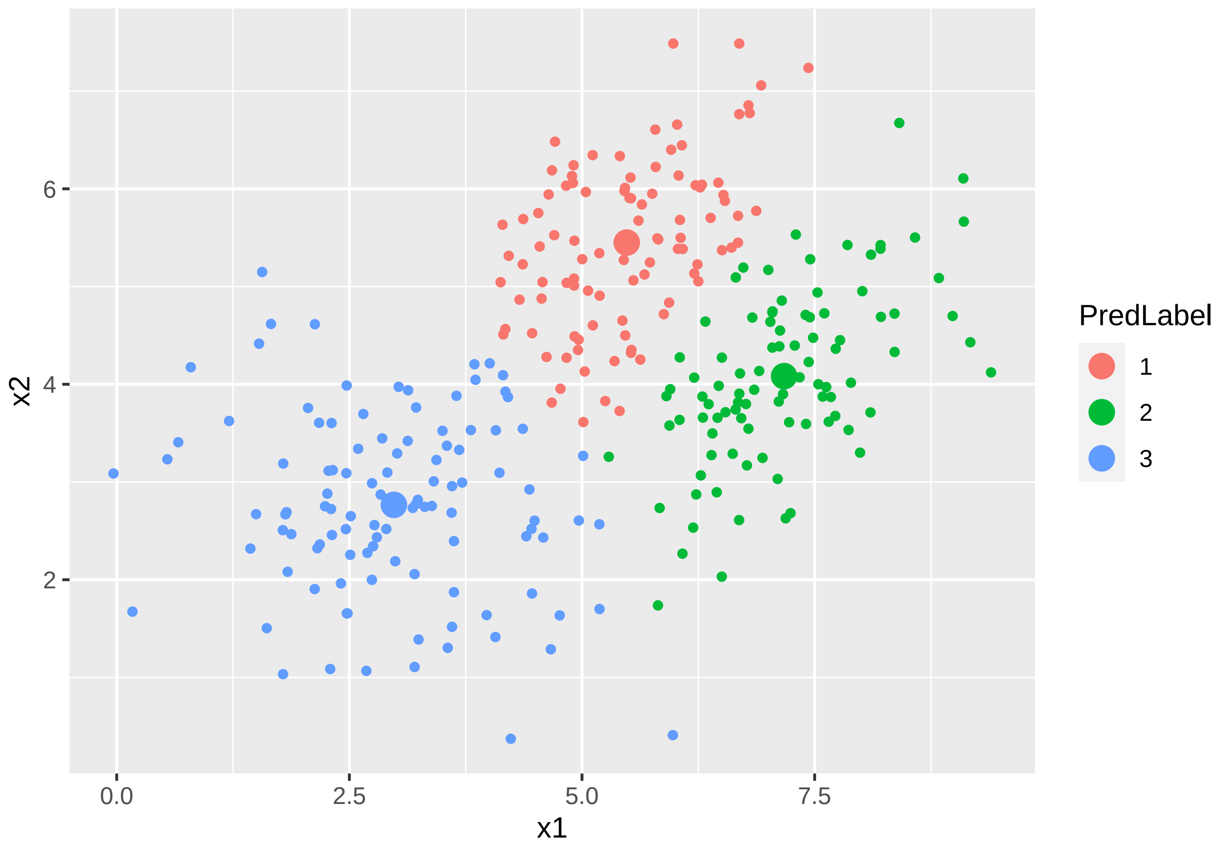
Plot with true labels

```
ggplot(data = Data) +
  geom_point(mapping = aes(x = x1,y = x2,color = label),pch = 16)
```



Apply k-means

```
k = 3
km = kmeans(X,centers=k)
gd = data.frame(km$centers)
gd$label = rownames(gd)
colnames(gd) = c("x1","x2","label")
Data$PredLabel = factor(km$cluster)
ggplot() +
    geom_point(data = Data,mapping = aes(x = x1,y = x2,color = PredLabel), pch = 16)  +
    geom_point(gd,mapping = aes(x = x1,y = x2,color= factor(label)),size = 4)
```



Code to understand K-means algorithm: raw code for k-means

```
mv.kmeans = function(x,k,cens=NULL){
  n = nrow(x)
  if(is.null(cens)){
      cens = x[sample(1:n,k),]
    }
  plot(x[,1],x[,2],pch=16)
  points(cens[,1],cens[,2],col=1:k,pch=16,cex=3)
  thr = 1e-6; ind = 1; iter = 1;
  while( ind>thr)
    {
      oldcen = cens
      km = kmeans(x,centers=cens,iter.max=1,nstart=1,algorithm="MacQueen")
      plot(x[,1],x[,2],col=km$cluster,pch=16)
      points(cens[,1],cens[,2],col=1:k,pch=16,cex=3)
      cens = km$centers
      #print(cens)
      plot(x[,1],x[,2],col=km$cluster,pch=16)
      points(cens[,1],cens[,2],col=1:k,pch=16,cex=3)
      ind = sum(diag((oldcen-cens)%*%t(oldcen-cens)))
      #print(ind)
    }
}
```

watch K-means algorithm movie
start from random starting points

```
saveHTML(mv.kmeans(X,3,cens=NULL),htmlfile="2021.html")
```

```
## HTML file created at: 2021.html
```

2. Data set 2 - NCI Microarray data: The data contains expression levels on 6830 genes from 64 cancer cell lines. Cancer type is also recorded.

- Apply K-means to cluster a high-dimensional data set.
- Apply hierarchical clustering & try out different linkages.
- Apply biclustering (Cluster heatmap) to visualize data.

```
ncidat = NCI60$data
rownames(ncidat) = NCI60$labs # cancer type
dim(ncidat)
```

```
## [1]   64 6830
```

```
table(NCI60$labs)
```

```
##
##      BREAST         CNS       COLON K562A-repro K562B-repro    LEUKEMIA
##           7           5           7           1           1           6
## MCF7A-repro MCF7D-repro    MELANOMA       NSCLC     OVARIAN    PROSTATE
##           1           1           8           9           6           2
##       RENAL     UNKNOWN
##           9           1
```

Apply K-means

```
K = 9
km = kmeans(ncidat,centers=K)
```

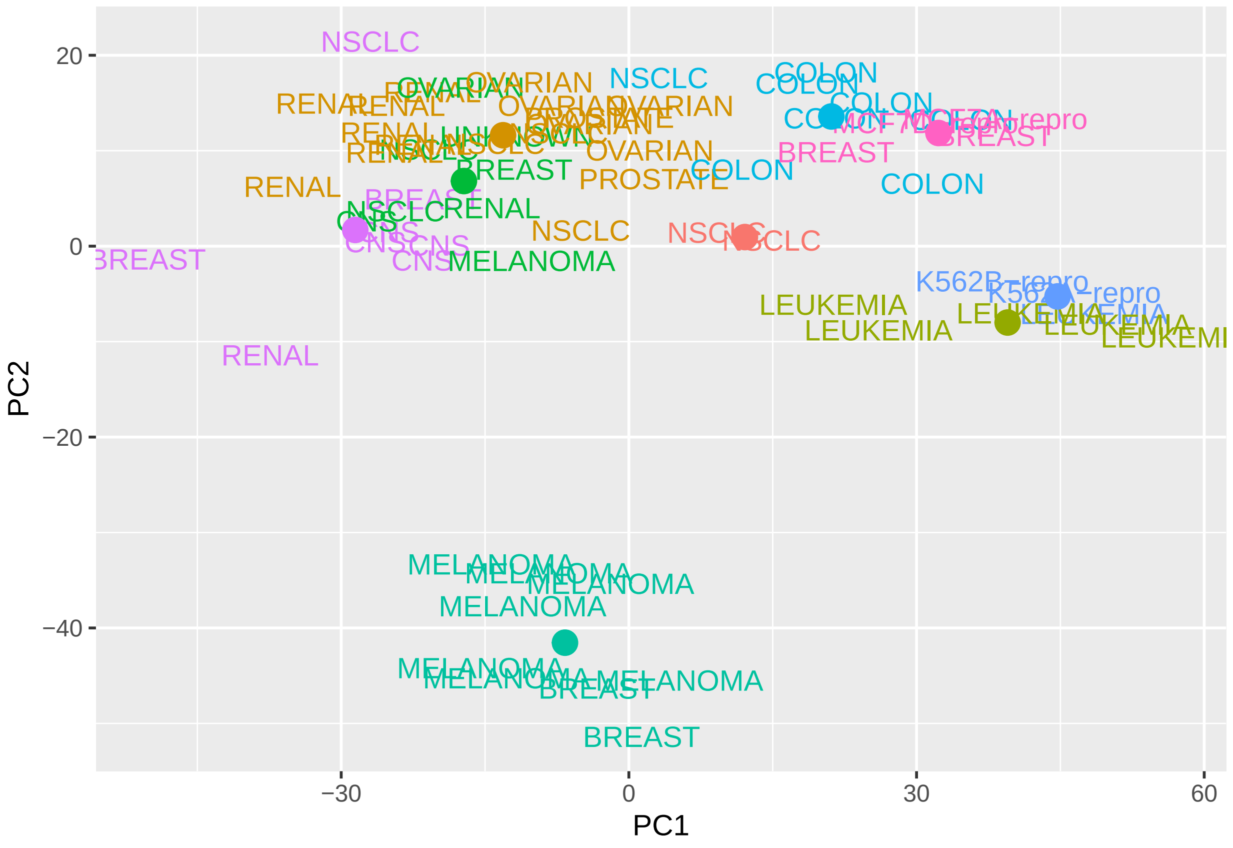
How do we visualize K-means results?
PCA - take SVD to get solution
Center genes, but don't scale

```
X = scale(ncidat,center=TRUE,scale=FALSE)
sv = svd(X)
U = sv$u
V = sv$v
D = sv$d
Z = X%*%V
```

Visualization

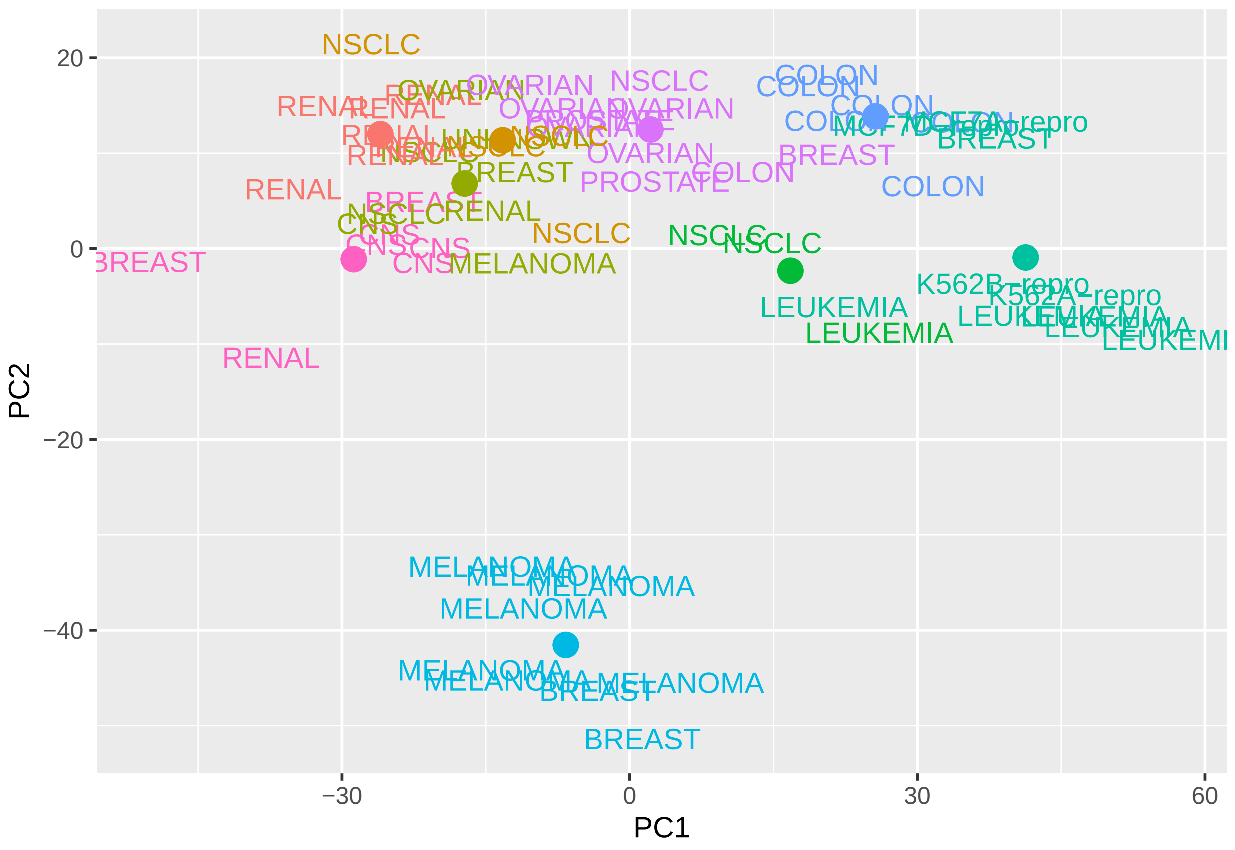
```
# projected data
PCData = data.frame(cbind(Z[,1],Z[,2],km$cluster,NCI60$labs),stringsAsFactors = FALSE)
colnames(PCData) = c("PC1","PC2","PredLabel","CancerType")
PCData$PC1 = as.numeric(PCData$PC1)
PCData$PC2 = as.numeric(PCData$PC2)
# projected k-means centers
GroupData = data.frame(km$centers%*%V[,1:2])
GroupData$label = rownames(GroupData)
colnames(GroupData) = c("PC1","PC2","PredLabel")
```

```r
ggplot(PCData,mapping=aes(x = PC1,y= PC2,color = PredLabel)) +
  geom_text(mapping=aes(label = CancerType)) +
  geom_point(data = GroupData,size = 4) +
  theme(legend.position="none")
```



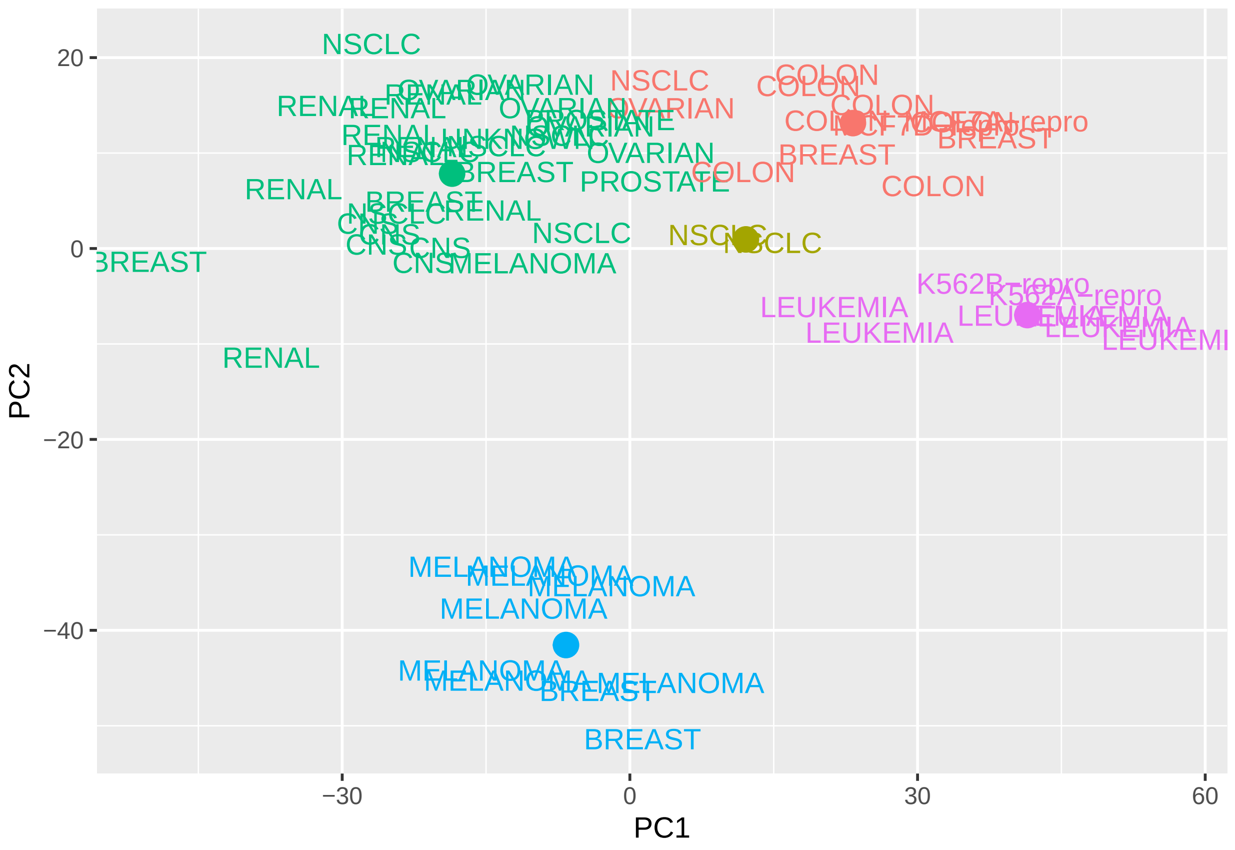
Re-run and see if solution changes

```r
K = 9
km = kmeans(ncidat,centers=K)
PCData$PredLabel = as.factor(km$cluster)

# projected k-means centers
GroupData = data.frame(km$centers%*%V[,1:2])
GroupData$label = rownames(GroupData)
colnames(GroupData) = c("PC1","PC2","PredLabel")

# plot
ggplot(PCData,mapping=aes(x = PC1,y= PC2,color = PredLabel)) +
  geom_text(mapping=aes(label = CancerType)) +
    geom_point(data = GroupData,size = 4) +
  theme(legend.position="none")
```



Try different K

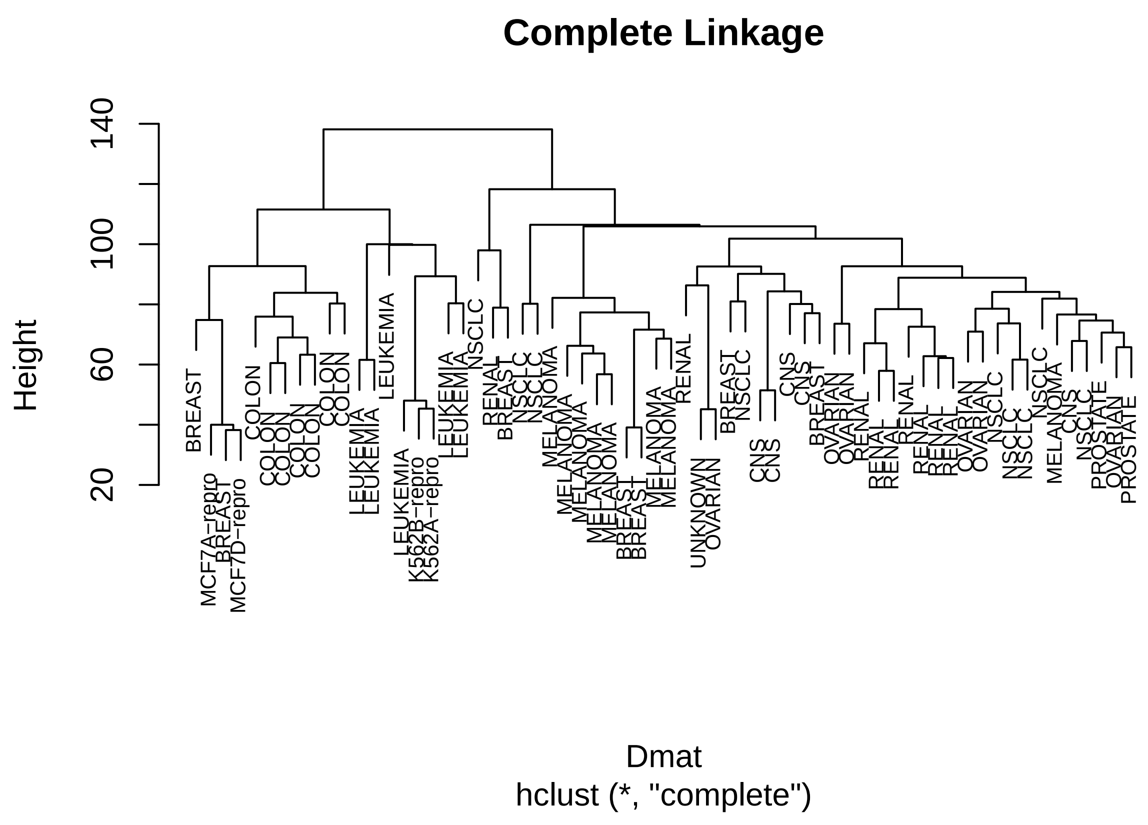
```
K = 5
km = kmeans(ncidat,centers=K)
PCData$PredLabel = as.factor(km$cluster)

# projected k-means centers
GroupData = data.frame(km$centers%*%V[,1:2])
GroupData$label = rownames(GroupData)
colnames(GroupData) = c("PC1","PC2","PredLabel")

# plot
ggplot(PCData,mapping=aes(x = PC1,y= PC2,color = PredLabel)) +
  geom_text(mapping=aes(label = CancerType)) +
    geom_point(data = GroupData,size = 4) +
  theme(legend.position="none")
```



Hierarchical clustering

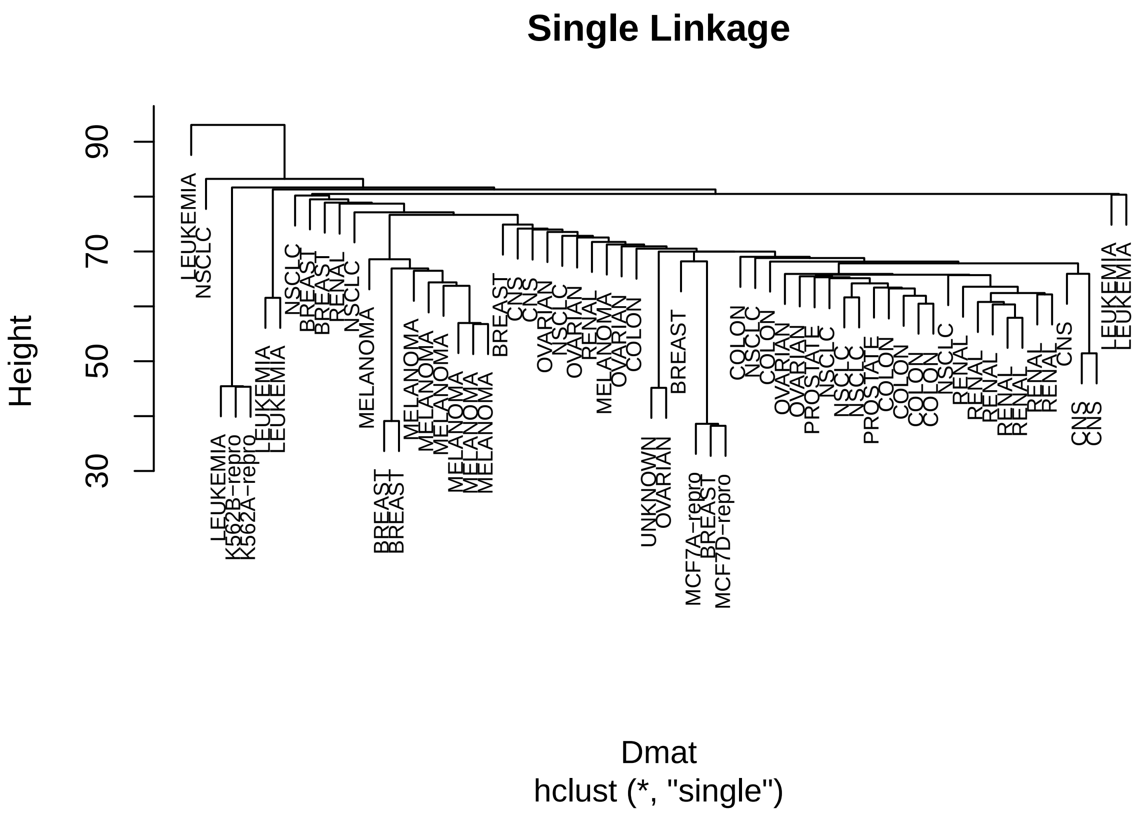
Real Data: NCI 60 data in ISLR package
Complete linakge - Euclidean distance

```r
cols = as.numeric(as.factor(rownames(ncidat)))
Dmat = dist(ncidat)
com.hclust = hclust(Dmat,method="complete")
plot(com.hclust,cex=.7,main="Complete Linkage")
```



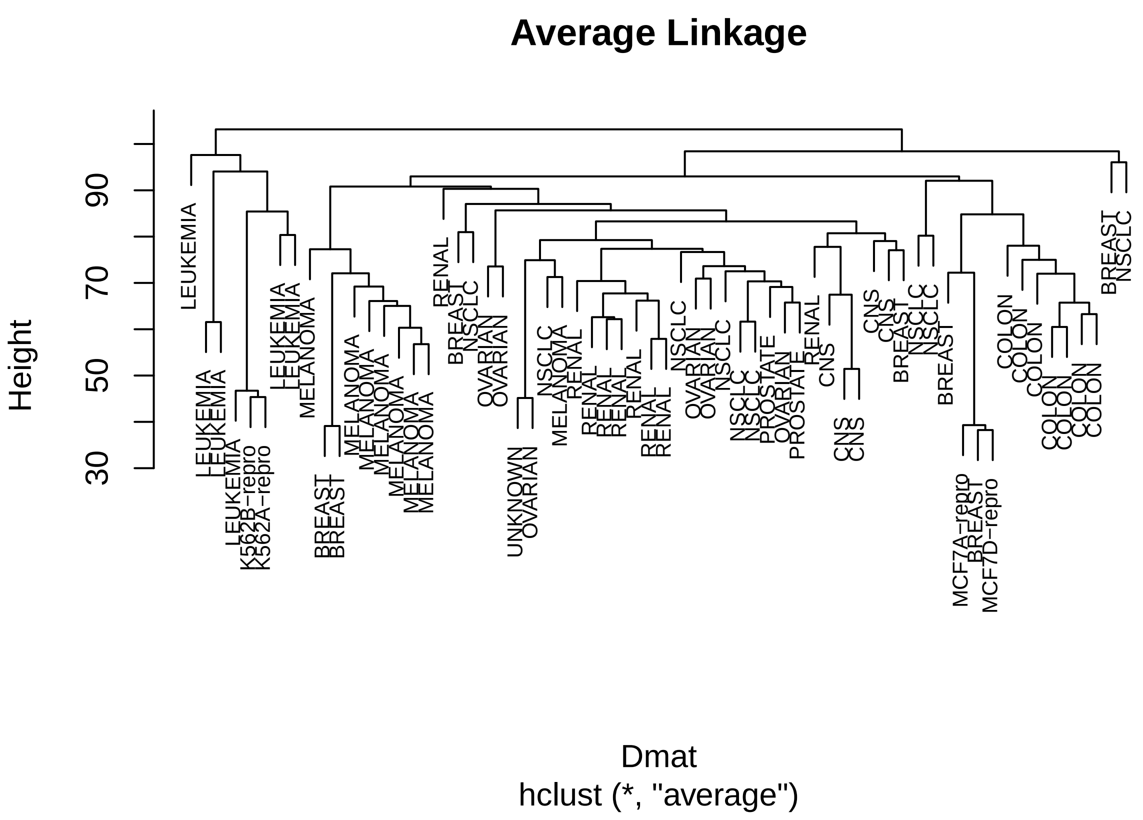
Single linakge

```
sing.hclust = hclust(Dmat,method="single")
plot(sing.hclust,cex=.7,main="Single Linkage")
```



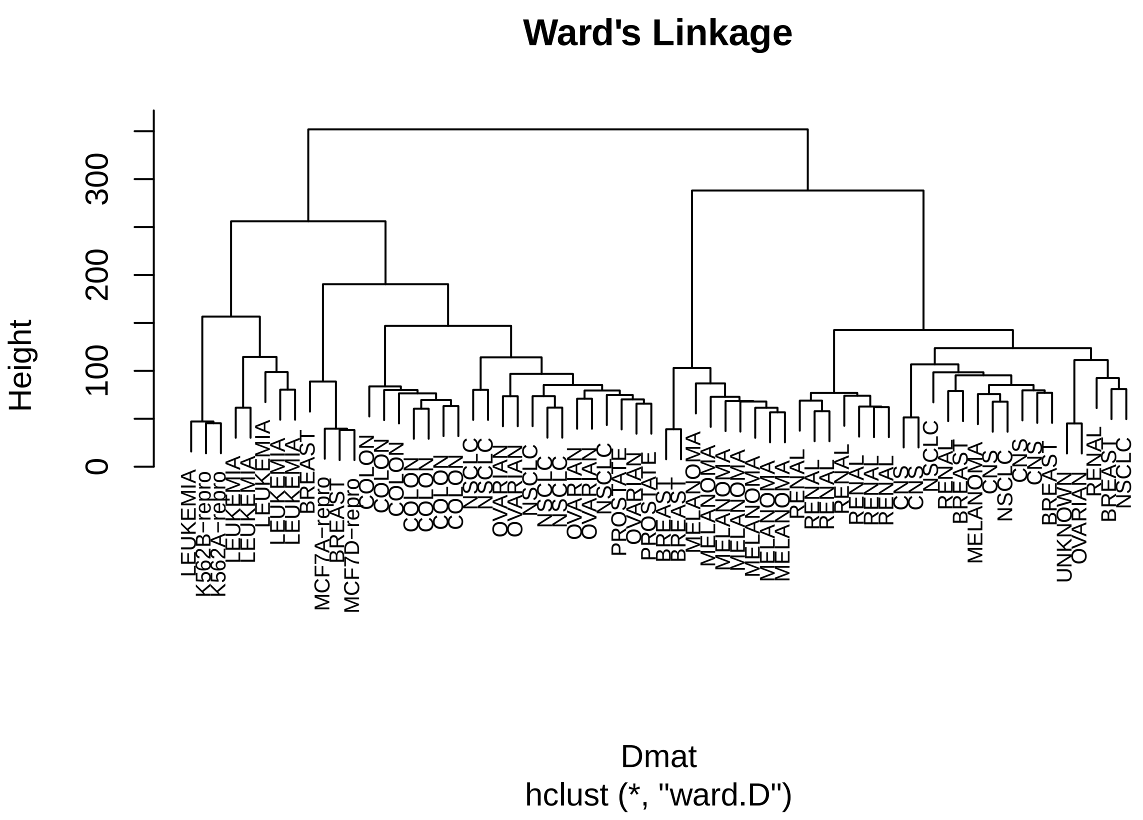
Average linakge

```
ave.hclust = hclust(Dmat,method="average")
plot(ave.hclust,cex=.7,main="Average Linkage")
```



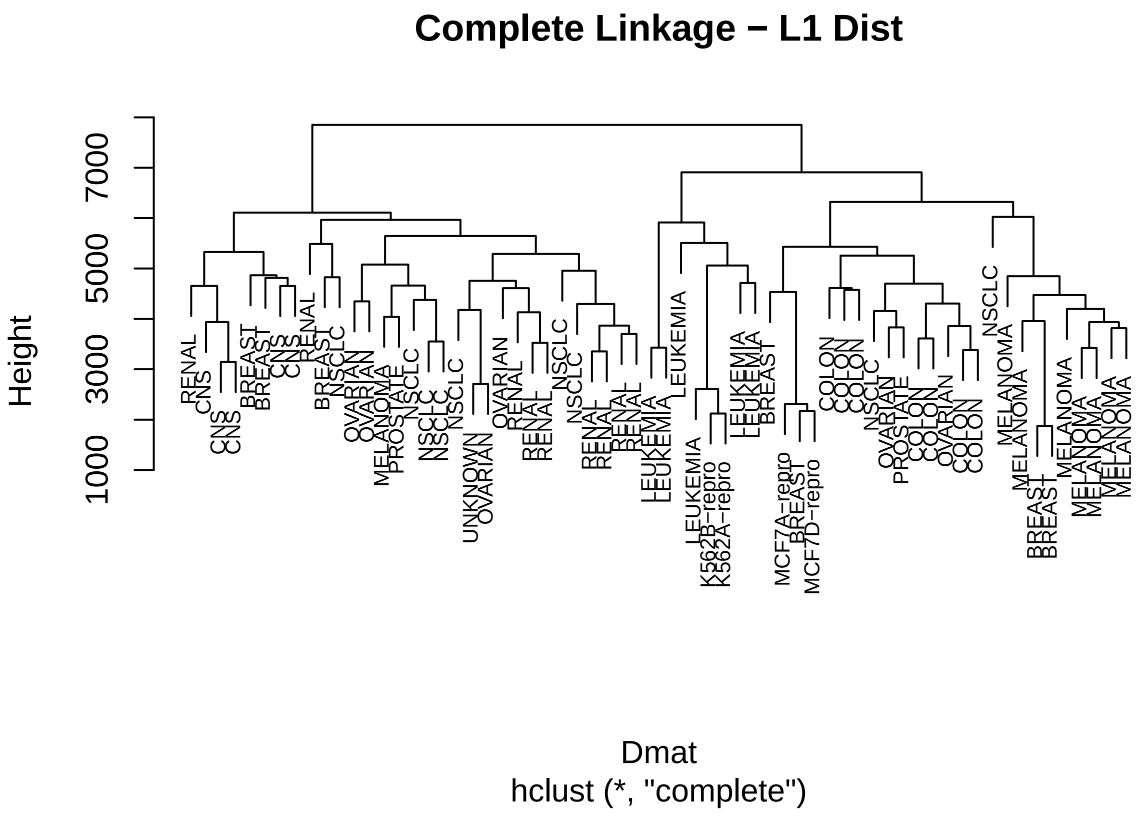
Ward's linakge

```
ward.hclust = hclust(Dmat,method="ward.D")
plot(ward.hclust,cex=.7,main="Ward's Linkage")
```



Complete linkage with different distances - L1 distance

```
Dmat = dist(ncidat,method="manhattan") #L1 distance
com.hclust = hclust(Dmat,method="complete")
plot(com.hclust,cex=.7,main="Complete Linkage - L1 Dist")
```



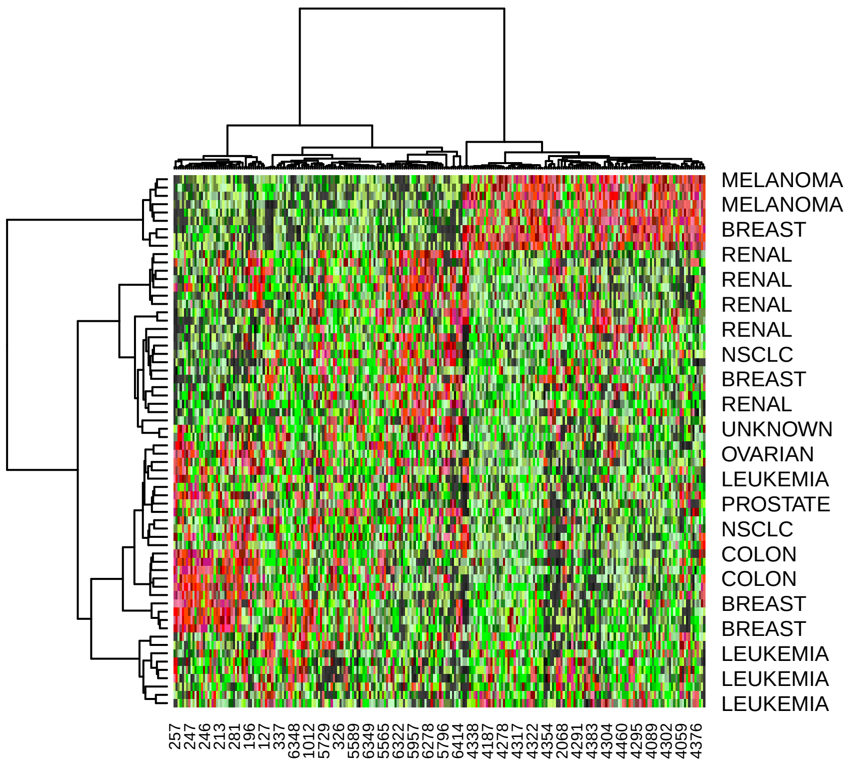
Biclustering - Cluster Heatmap

Filter genes using PCA
PC loadings - visualize data by limiting to top genes in magnitude in the PC loadings

```r
aa = grep("grey",colors())
bb = grep("green",colors())
cc = grep("red",colors())
gcol2 = colors()[c(aa[1:30],bb[1:20],rep(cc,2))]

j = 2
ord = order(abs(V[,j]),decreasing=TRUE)
x = as.matrix(X[,ord[1:250]])
```

#cluster heatmap - uses Ward's linkage (complete is default)

```r
heatmap(x,col=gcol2,hclustfun=function(x) hclust(x,method="ward.D"))
```



Convex Clustering & Biclustering

Dataset : presidential_speech
We demonstrate the use of these two functions on a text minining data set, presidential_speech, which measures how often the 44 U.S. presidents used certain words in their public addresses.

```
data(presidential_speech)
presidential_speech[1:6, 1:6]
```

```
##                      amount appropri  british     cent commerci commission
## Abraham Lincoln    3.433987 2.397895 1.791759 2.564949 2.708050   2.079442
## Andrew Jackson     4.248495 4.663439 2.995732 1.945910 3.828641   3.218876
## Andrew Johnson     4.025352 3.091042 2.833213 3.332205 2.772589   2.079442
## Barack Obama       1.386294 0.000000 0.000000 1.386294 0.000000   0.000000
## Benjamin Harrison 4.060443 4.174387 2.302585 4.304065 3.663562   3.465736
## Calvin Coolidge    3.713572 4.094345 1.386294 3.555348 2.639057   1.609438
```

Clustering

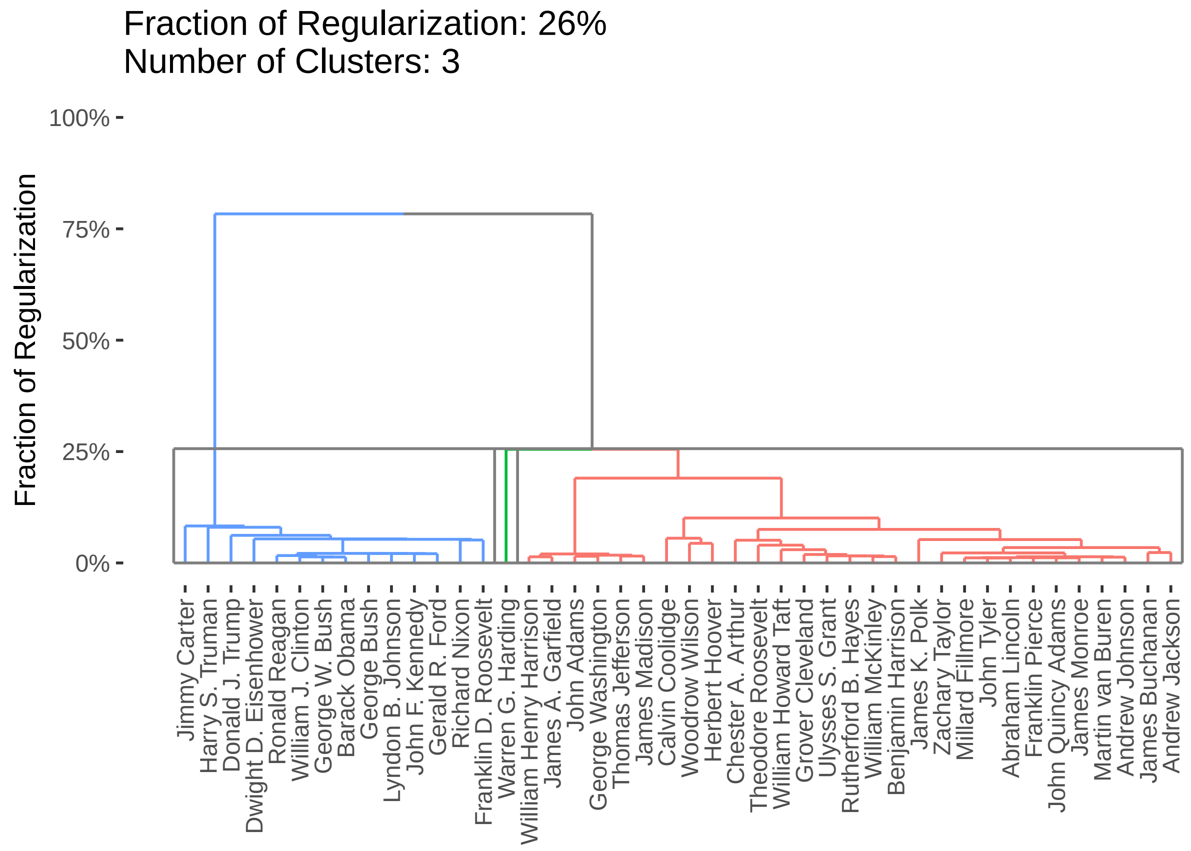
```
carp_fit <- CARP(presidential_speech)
```

```
## Pre-computing weights and edge sets

## Computing Convex Clustering [CARP] Path

## Post-processing
```

```
print(carp_fit)
```

```
## CARP Fit Summary
## ====================
##
## Algorithm: CARP (t = 1.05)
## Fit Time: 0.244 secs
## Total Time: 1.371 secs
##
## Number of Observations: 44
## Number of Variables:    75
##
## Pre-processing options:
##  - Columnwise centering: TRUE
##  - Columnwise scaling:   FALSE
##
## Weights:
##  - Source: Radial Basis Function Kernel Weights
##  - Distance Metric: Euclidean
##  - Scale parameter (phi): 0.01 [Data-Driven]
##  - Sparsified: 4 Nearest Neighbors [Data-Driven]
```

```
plot(carp_fit, type = "dendrogram", k = 3)
```



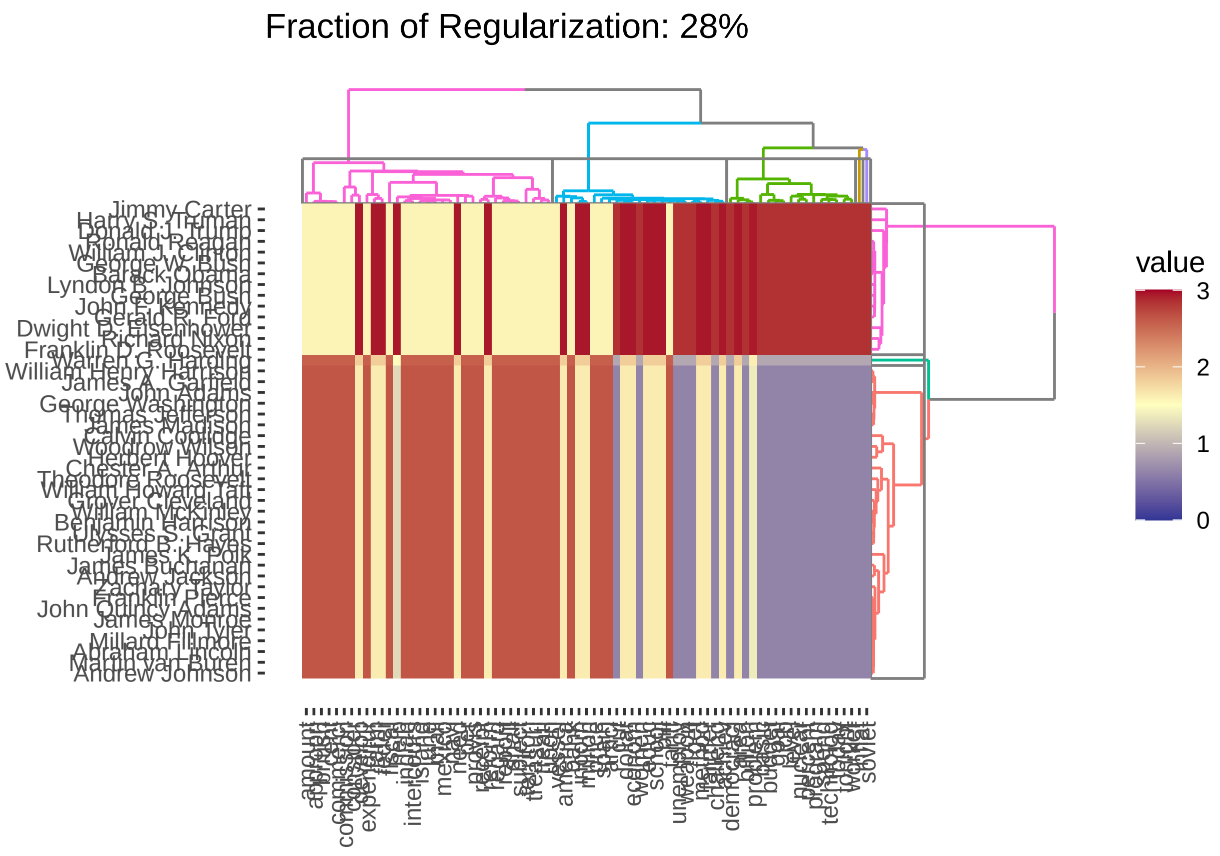
```
# plot(carp_fit, type = "path", dynamic = TRUE)
```

Convex Biclustering

```
cbass_fit <- CBASS(presidential_speech)
```

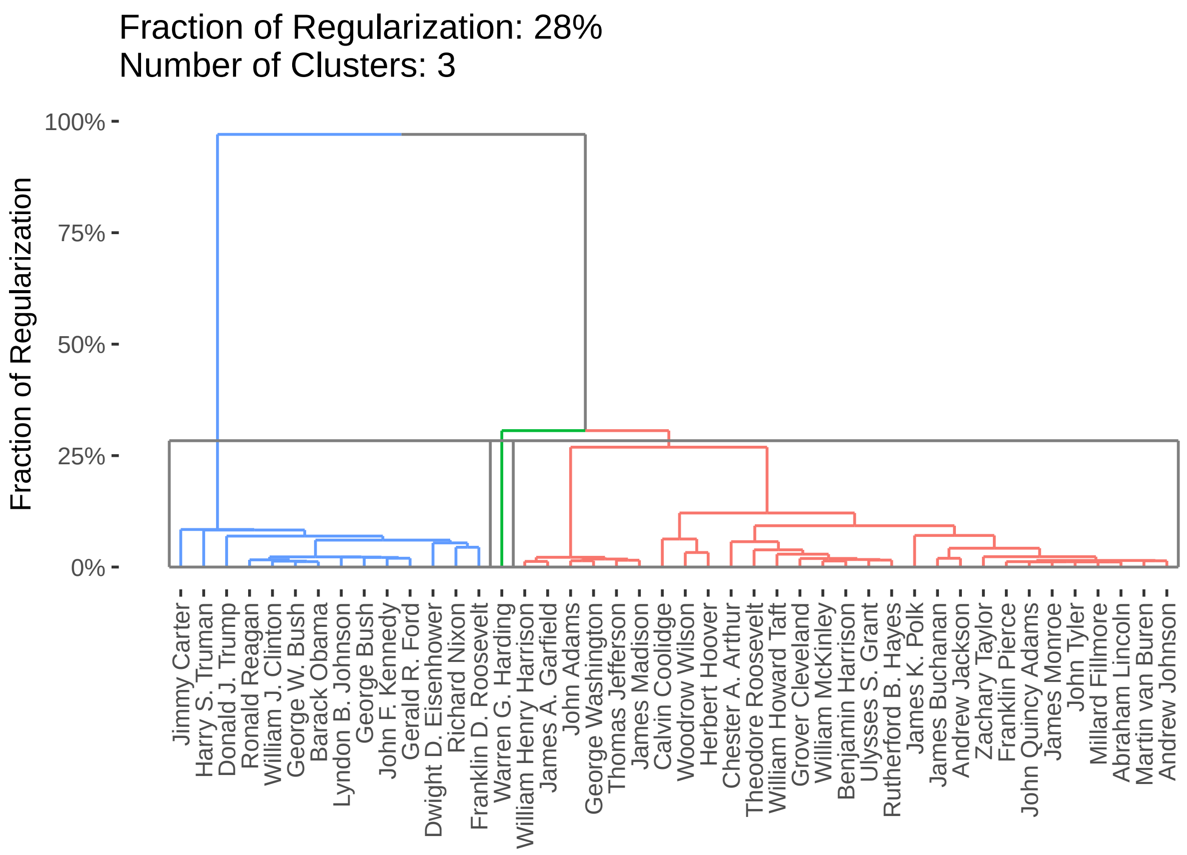
```
## Pre-computing column weights and edge sets

## Pre-computing row weights and edge sets

## Computing Convex Bi-Clustering [CBASS] Path
## Post-processing rows

## Post-processing columns
```

```
plot(cbass_fit, k.row = 3)
```



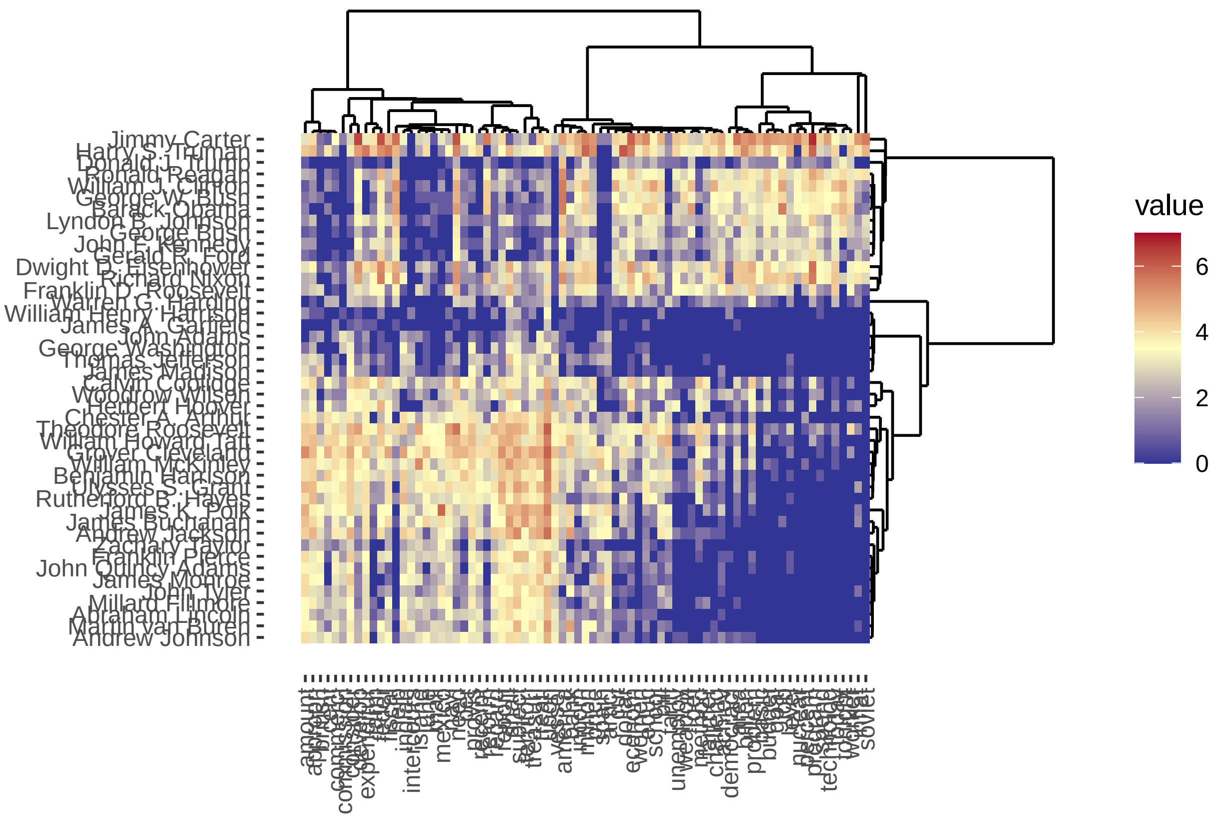
By default, plotting the result of CBASS gives the traditional cluster heatmap, but we can also get the row or column dendrograms as well:

```
plot(cbass_fit, type = "row.dendrogram", k.row = 3)
```



Setting type = "heatmap" gives the traditional cluster heatmap:

```
plot(cbass_fit, type = "heatmap")
```



Significance of Clustering (SigClust)

Simulated data

```r
## Simulate a dataset from a collection of mixtures of two
## multivariate Gaussian distribution with different means.
mu <- 5
n <- 30
p <- 500
dat <- matrix(rnorm(p*2*n),2*n,p)
dat[1:n,1] <- dat[1:n,1]+mu
dat[(n+1):(2*n),1] <- dat[(n+1):(2*n),1]-mu

nsim <- 1000
nrep <- 1
icovest <- 1
pvalue <- sigclust(dat,nsim=nsim,nrep=nrep,labflag=0,icovest=icovest)
```

```r
slot(pvalue,"pval")
```

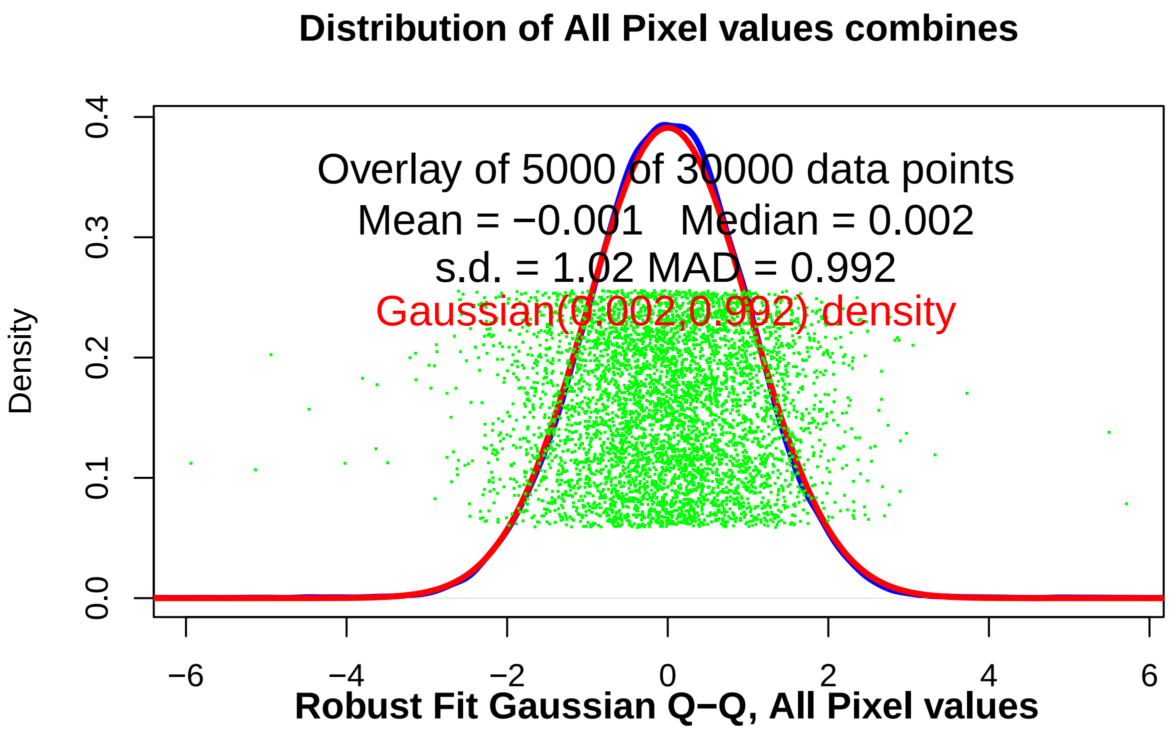
```
## [1] 0.002
```

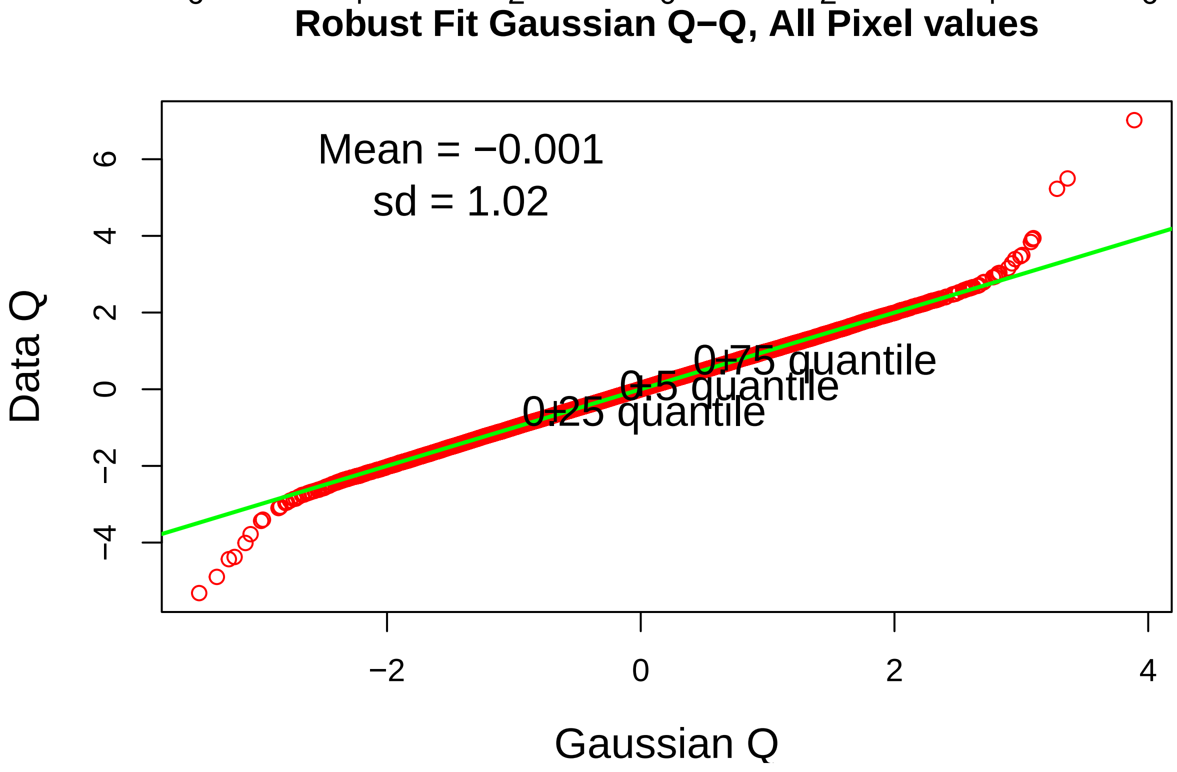
```r
slot(pvalue,"pvalnorm")
```

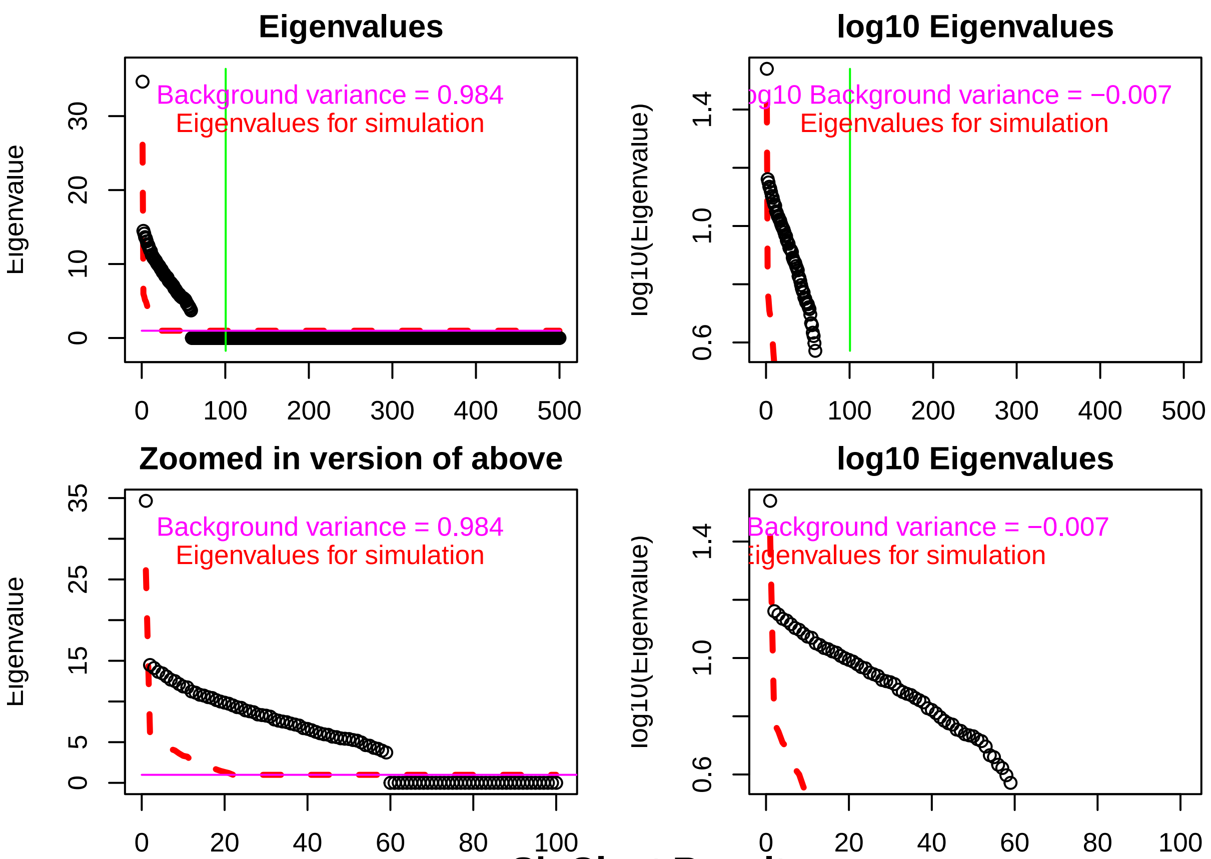
```
## [1] 0.0006893125
```

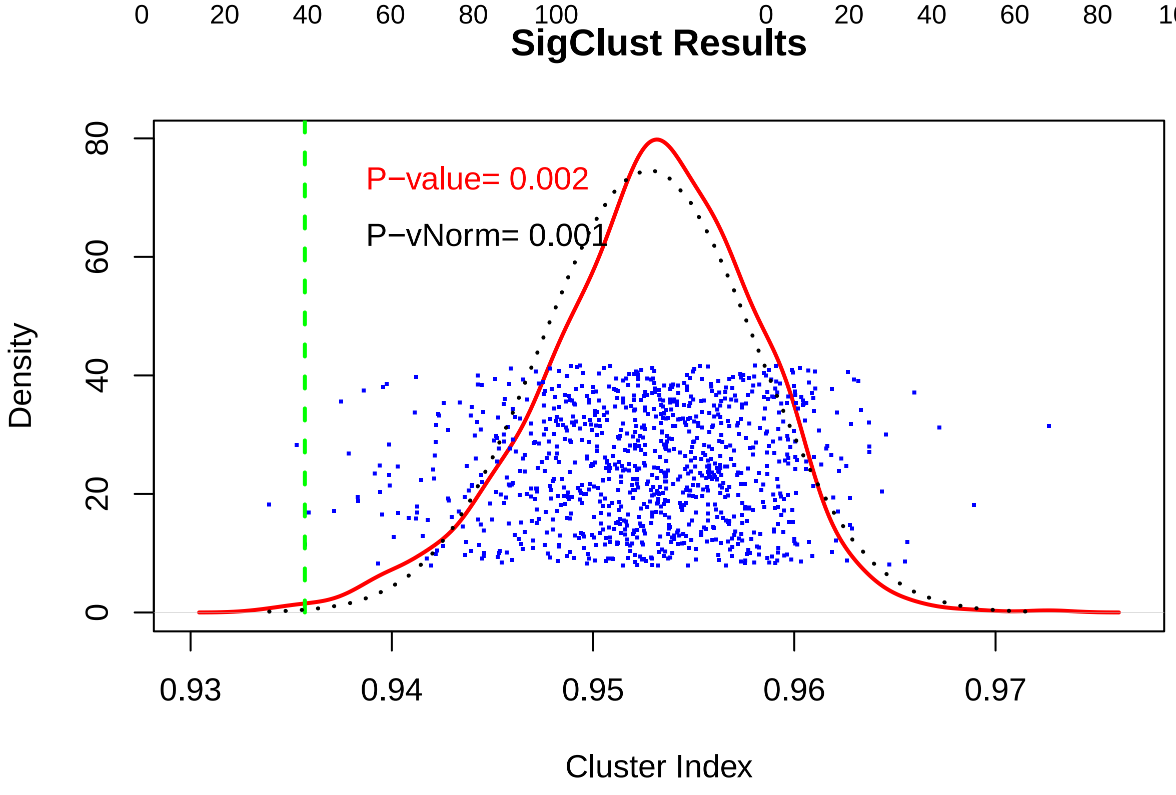
SigClust plot

```r
plot(pvalue)
```


Distribution of All Pixel values combines
Overlay of 5000 of 30000 data points
Mean = −0.001   Median = 0.002
s.d. = 1.02 MAD = 0.992
Gaussian(0.002,0.992) density
Density
0.0
0.1
0.2
0.3
0.4
−6
−4
−2
0
2
4
6


Robust Fit Gaussian Q−Q, All Pixel values
Mean = −0.001
sd = 1.02
0.75 quantile
0.5 quantile
0.25 quantile
Data Q
Gaussian Q
−4
−2
0
2
4
6
−2
0
2
4


Eigenvalues
Background variance = 0.984
Eigenvalues for simulation
Eigenvalue
log10 Eigenvalues
log10 Background variance = −0.007
Eigenvalues for simulation
log10(Eigenvalue)
Zoomed in version of above
Background variance = 0.984
Eigenvalues for simulation
Eigenvalue
log10 Eigenvalues
Background variance = −0.007
Eigenvalues for simulation
log10(Eigenvalue)

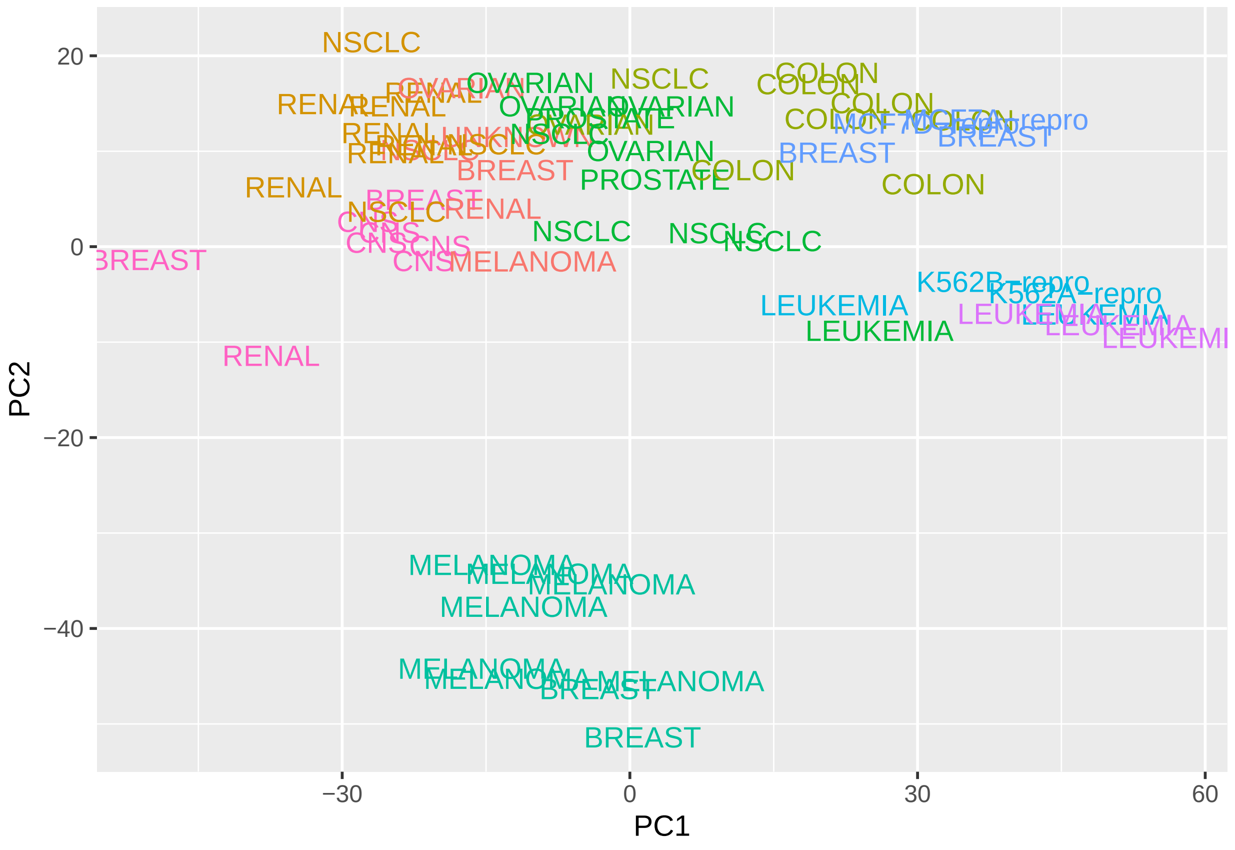

SigClust Results
P−value= 0.002
P−vNorm= 0.001
Density
Cluster Index

Spectral clustering

```
K = 9
SC_NCI = specClust(ncidat, centers=K, nn = 7, method = "symmetric", gmax=NULL)
```

Visualization

```
X = scale(ncidat,center=TRUE,scale=FALSE)
sv = svd(X)
U = sv$u
V = sv$v
D = sv$d
Z = X%*%V

# projected data
SCData = data.frame(cbind(Z[,1],Z[,2],SC_NCI$cluster,NCI60$labs),stringsAsFactors = FALSE)
colnames(SCData) = c("PC1","PC2","PredLabel","CancerType")
SCData$PC1 = as.numeric(SCData$PC1)
SCData$PC2 = as.numeric(SCData$PC2)
# plot
ggplot(SCData,mapping=aes(x = PC1,y= PC2,color = PredLabel)) +
  geom_text(mapping=aes(label = CancerType)) +
  theme(legend.position="none")
```


PC2
PC1
NSCLC
RENAL
OVARIAN
COLON
PROSTATE
BREAST
CNS
MELANOMA
LEUKEMIA
K562B-repro
K562A-repro
MCF7D-repro
MCF7A-repro
UNKNOWN