Análise Exploratória

KMeans

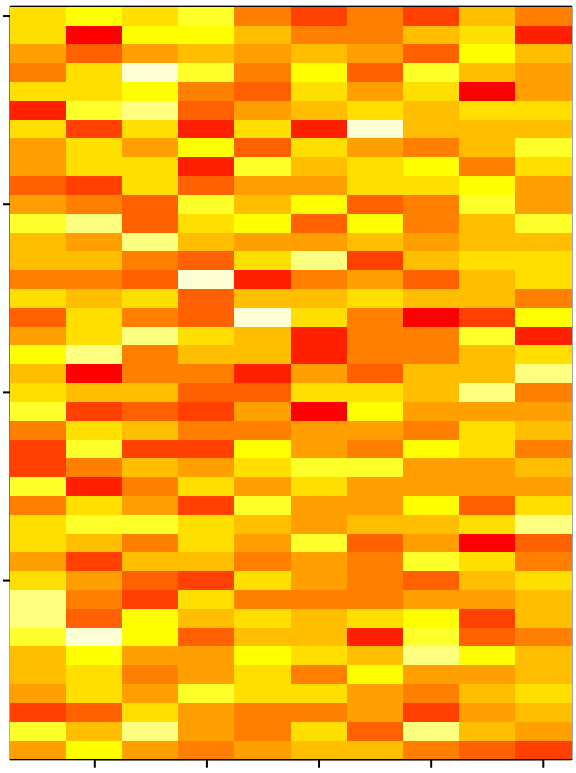
Delermando Branquinho Filho

Matrix data

```
set.seed(12345); par(mar=rep(0.2,4))
dataMatrix <- matrix(rnorm(400),nrow=40)
dim(dataMatrix)
```

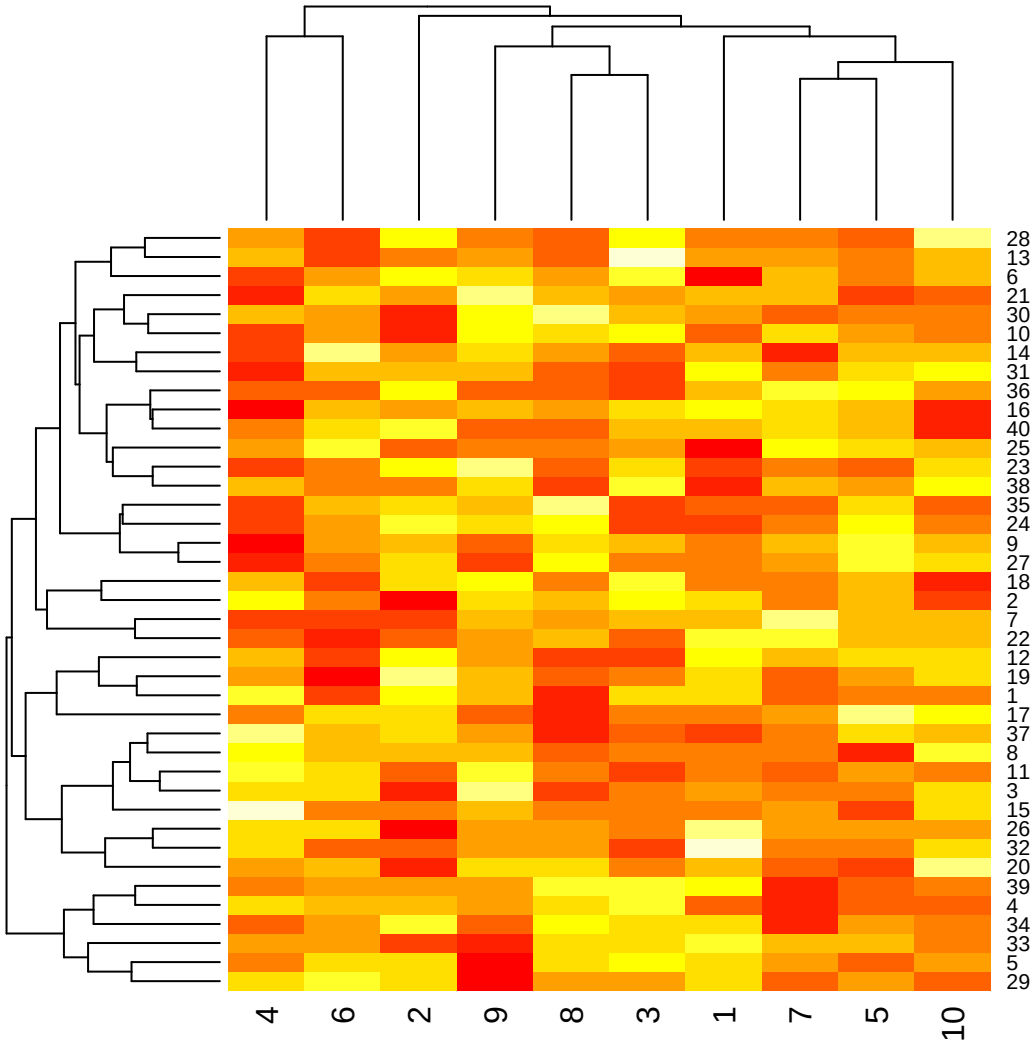
```
## [1] 40 10
```

```
image(1:10,1:40,t(dataMatrix)[,nrow(dataMatrix):1])
```



Cluster the data

```
par(mar=rep(0.2,4))
heatmap(dataMatrix)
```

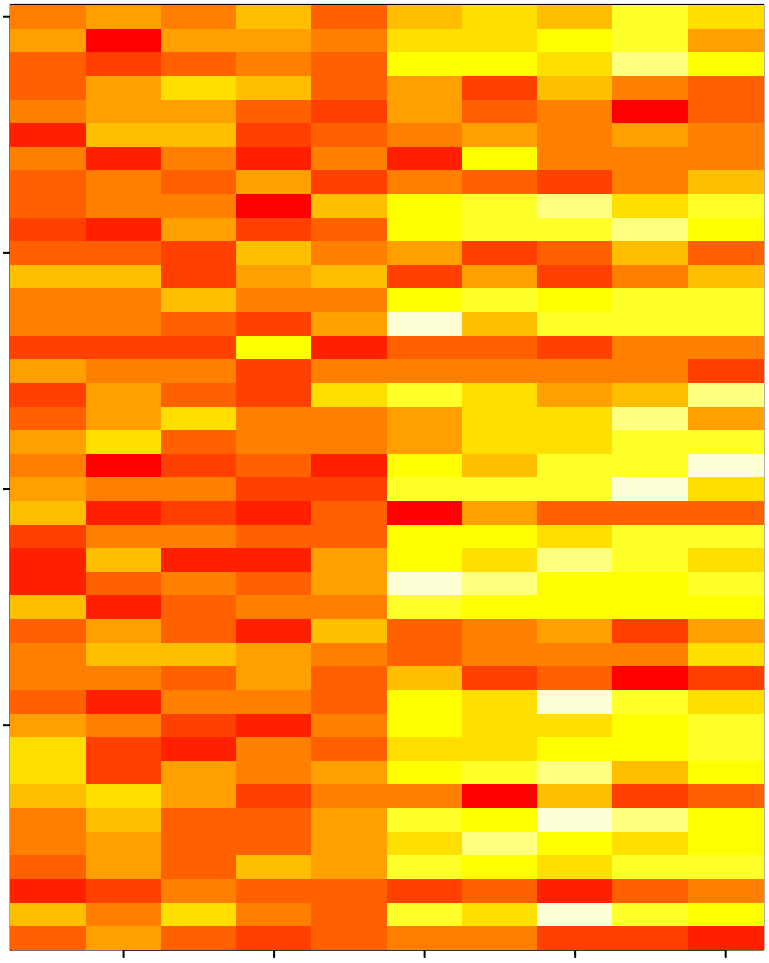


E se nós adicionarmos um padrão?

```
set.seed(678910)
for(i in 1:40){
  # flip a coin
  coinFlip <- rbinom(1,size=1,prob=0.5)
  # if coin is heads add a common pattern to that row
  if(coinFlip){
    dataMatrix[i,] <- dataMatrix[i,] + rep(c(0,3),each=5)
  }
}
```

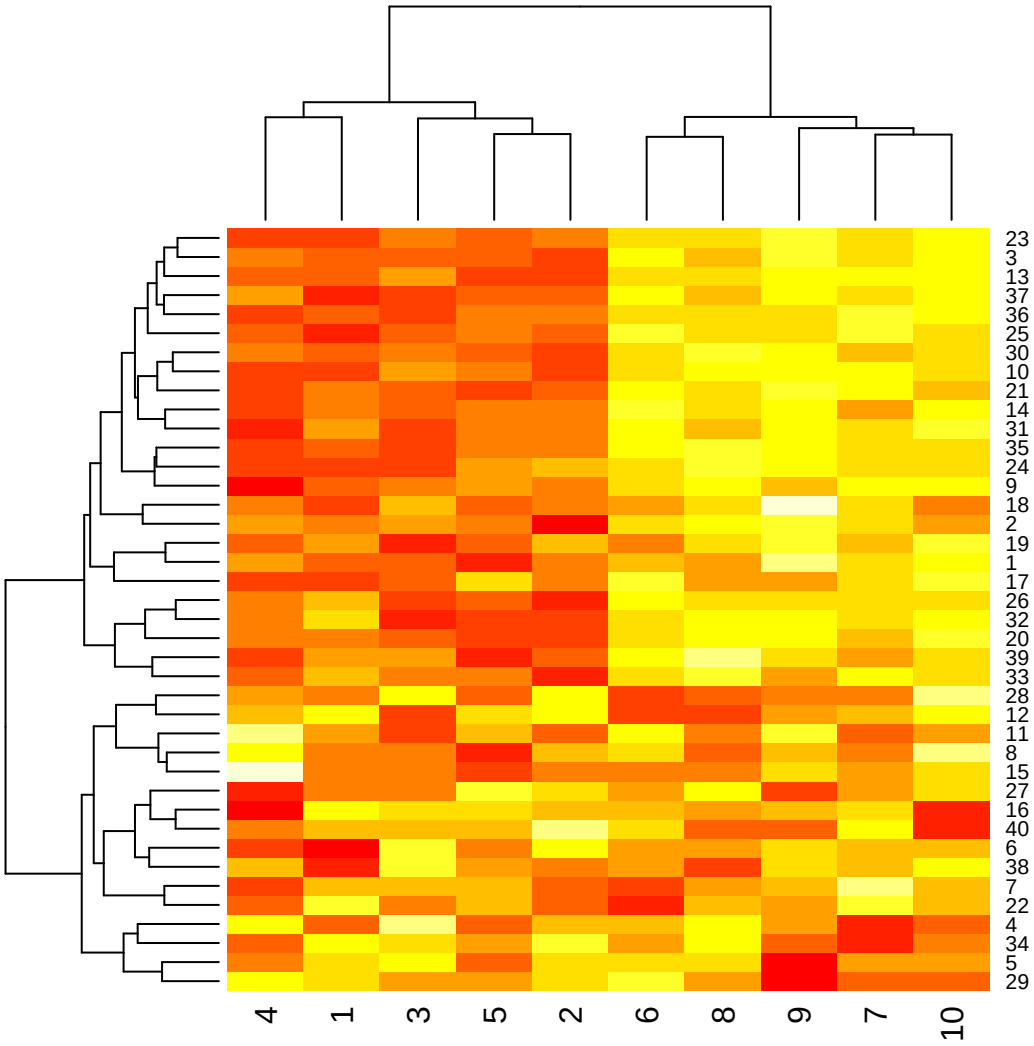
E se nós adicionarmos um padrão? - os dados

```
par(mar=rep(0.2,4))
image(1:10,1:40,t(dataMatrix)[,nrow(dataMatrix):1])
```



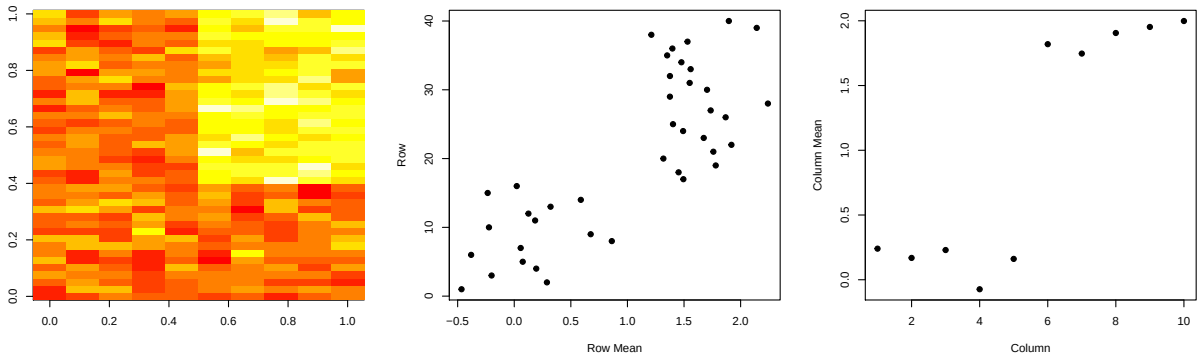
What if we add a pattern? - Os dados agrupados

```
par(mar=rep(0.2,4))
heatmap(dataMatrix)
```



Padrões em linhas e colunas

```
hh <- hclust(dist(dataMatrix));
dataMatrixOrdered <- dataMatrix[hh$order,]
par(mfrow=c(1,3))
image(t(dataMatrixOrdered)[,nrow(dataMatrixOrdered):1])
plot(rowMeans(dataMatrixOrdered),40:1,xlab="Row Mean",ylab="Row",pch=19)
plot(colMeans(dataMatrixOrdered),xlab="Column",ylab="Column Mean",pch=19)
```



Soluções relacionadas - PCA / SVD

SVD

Se X é uma matriz com cada variável em uma coluna e cada observação em uma linha, em seguida, o SVD (Singular value decomposition) é uma "decomposição da matriz". Formalmente, a decomposição em valores singulares de uma matriz m×n real ou complexa M é uma fatoração ou fatorização na forma:

$$X = UDV^T$$

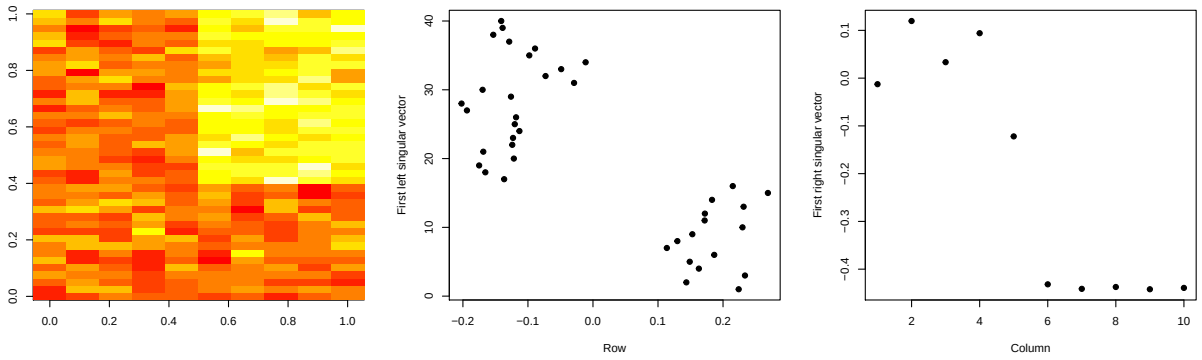
Onde as colunas de U são ortogonais (vetores singulares esquerdos), as colunas de V são ortogonais (vetores singulares diretos) e D é uma matriz diagonal (valores singulares).

PCA

The principal components are equal to the right singular values if you first scale (subtract the mean, divide by the standard deviation) the variables.

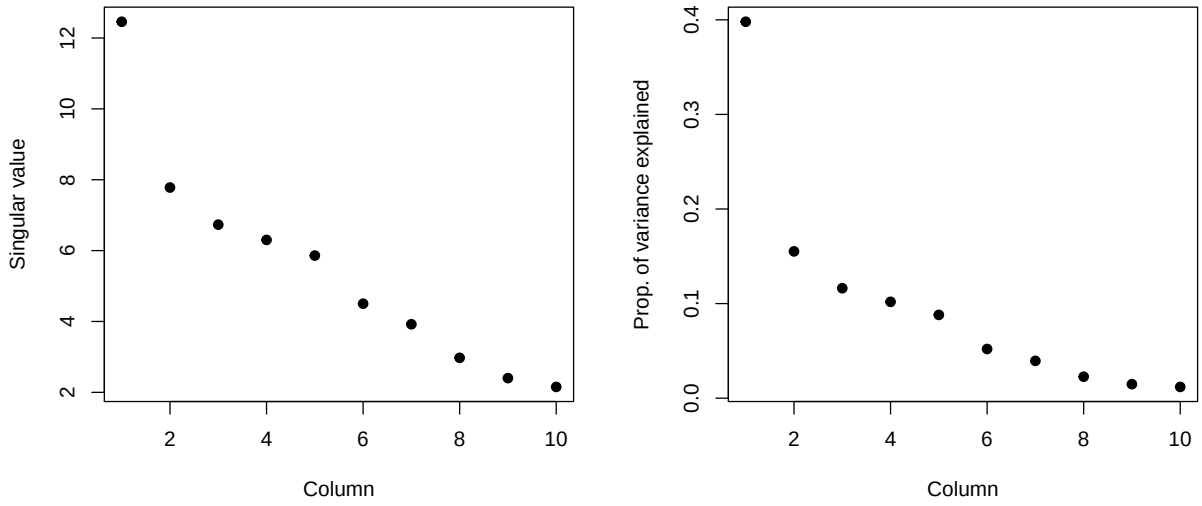
Components of the SVD - u and v

```
svd1 <- svd(scale(dataMatrixOrdered))
par(mfrow=c(1,3))
image(t(dataMatrixOrdered)[,nrow(dataMatrixOrdered):1])
plot(svd1$u[,1],40:1,xlab="Row",ylab="First left singular vector",pch=19)
plot(svd1$v[,1],xlab="Column",ylab="First right singular vector",pch=19)
```



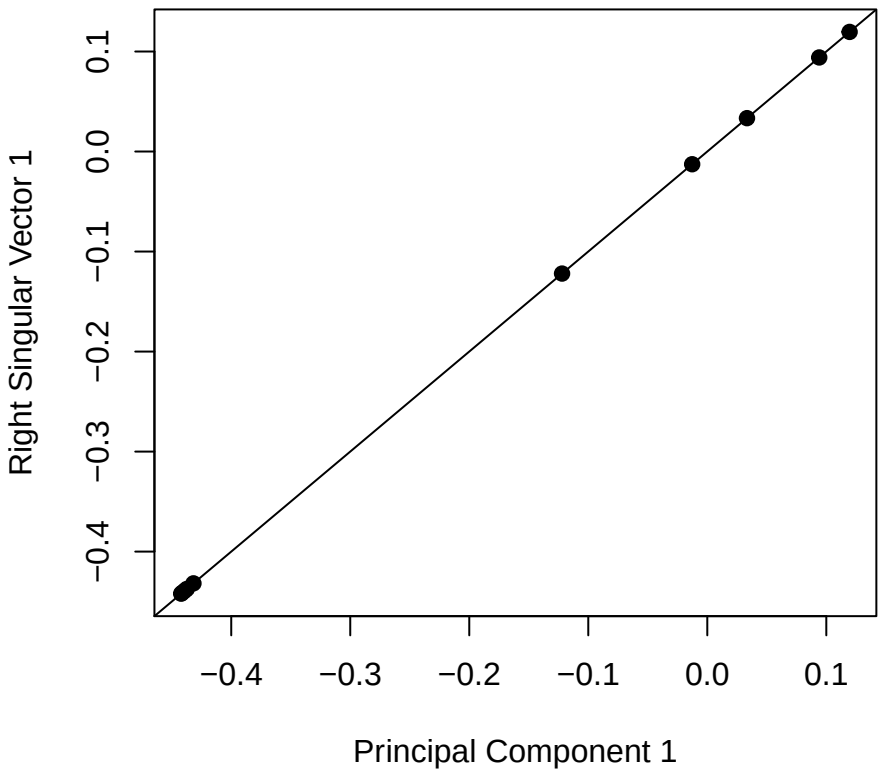
Componentes da SVD - Variância explicada

```
par(mfrow=c(1,2))
plot(svd1$d,xlab="Column",ylab="Singular value",pch=19)
plot(svd1$d^2/sum(svd1$d^2),xlab="Column",ylab="Prop. of variance explained",pch=19)
```

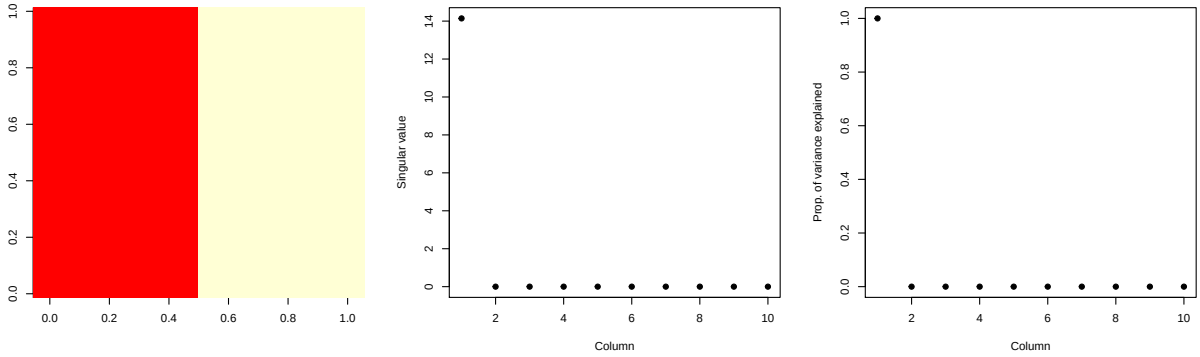


Relação com os componentes principais

```
svd1 <- svd(scale(dataMatrixOrdered))
pca1 <- prcomp(dataMatrixOrdered,scale=TRUE)
plot(pca1$rotation[,1],svd1$v[,1],pch=19,xlab="Principal Component 1",ylab="Right Singular Vector 1")
abline(c(0,1))
```



```
constantMatrix <- dataMatrixOrdered*0
for(i in 1:dim(dataMatrixOrdered)[1]){constantMatrix[i,] <- rep(c(0,1),each=5)}
svd1 <- svd(constantMatrix)
par(mfrow=c(1,3))
image(t(constantMatrix)[,nrow(constantMatrix):1])
plot(svd1$d,xlab="Column",ylab="Singular value",pch=19)
plot(svd1$d^2/sum(svd1$d^2),xlab="Column",ylab="Prop. of variance explained",pch=19)
```

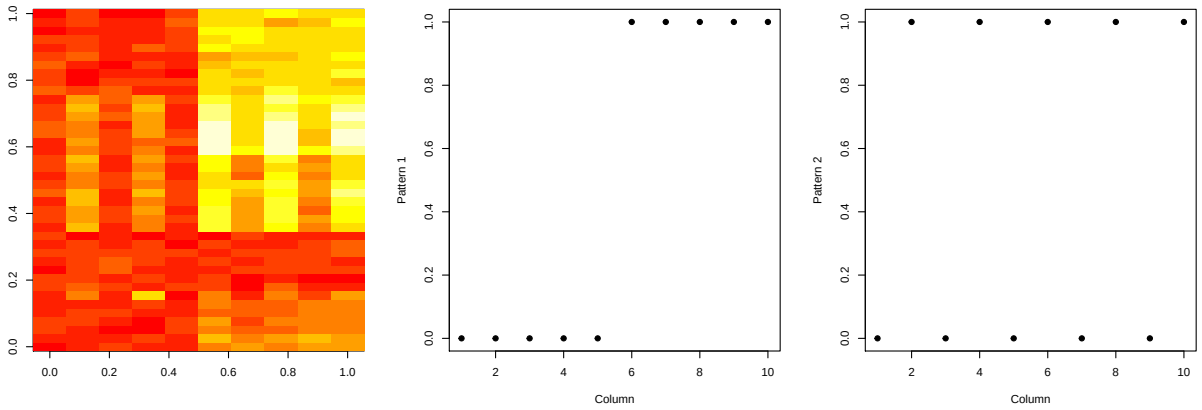


E se acrescentarmos um segundo padrão?

```
set.seed(678910)
for(i in 1:40){
  # flip a coin
  coinFlip1 <- rbinom(1,size=1,prob=0.5)
  coinFlip2 <- rbinom(1,size=1,prob=0.5)
  # if coin is heads add a common pattern to that row
  if(coinFlip1){
    dataMatrix[i,] <- dataMatrix[i,] + rep(c(0,5),each=5)
  }
  if(coinFlip2){
    dataMatrix[i,] <- dataMatrix[i,] + rep(c(0,5),5)
  }
}
hh <- hclust(dist(dataMatrix)); dataMatrixOrdered <- dataMatrix[hh$order,]
```

Decomposição do valor singular - padrões verdadeiros

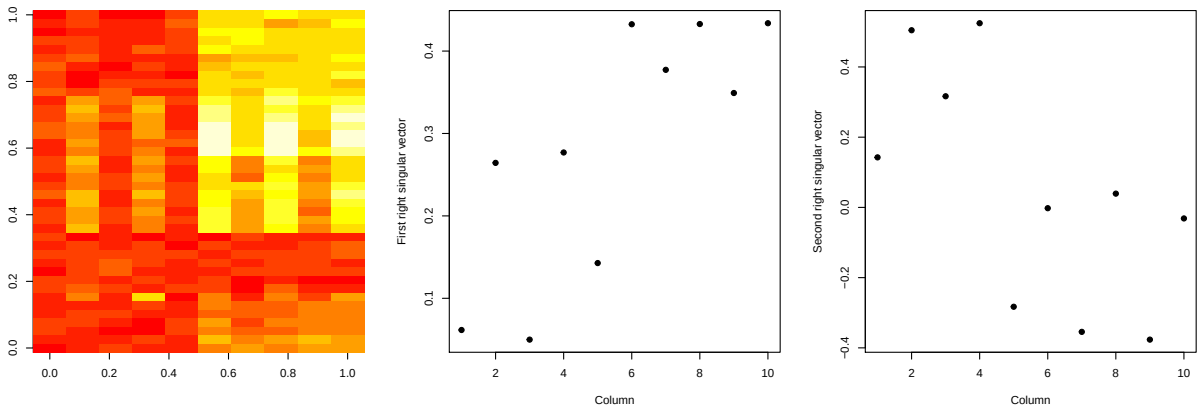
```
svd2 <- svd(scale(dataMatrixOrdered))
par(mfrow=c(1,3))
image(t(dataMatrixOrdered)[,nrow(dataMatrixOrdered):1])
plot(rep(c(0,1),each=5),pch=19,xlab="Column",ylab="Pattern 1")
plot(rep(c(0,1),5),pch=19,xlab="Column",ylab="Pattern 2")
```



v e padrões de variação em linhas

```
svd2 <- svd(scale(dataMatrixOrdered))
par(mfrow=c(1,3))
```

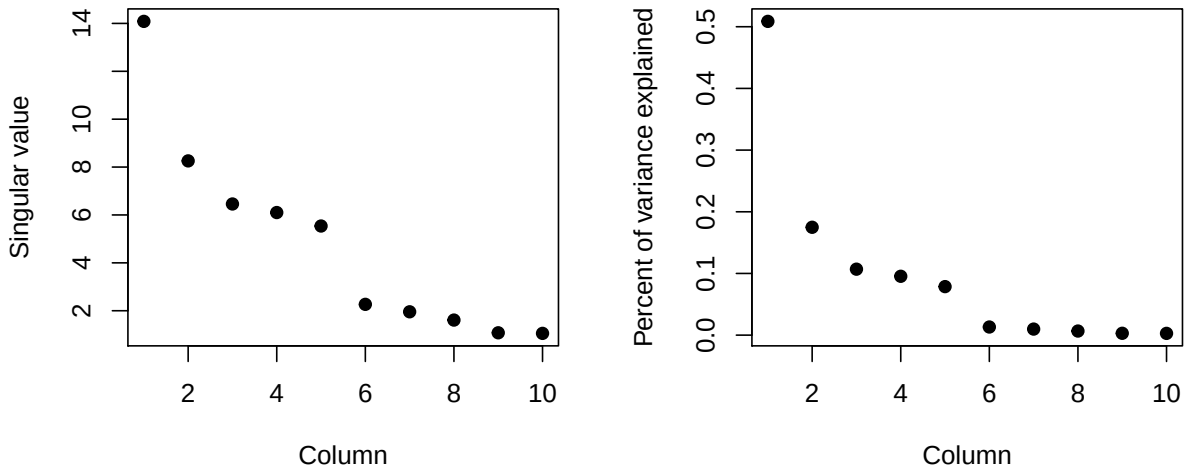
```
image(t(dataMatrixOrdered)[,nrow(dataMatrixOrdered):1])
plot(svd2$v[,1],pch=19,xlab="Column",ylab="First right singular vector")
plot(svd2$v[,2],pch=19,xlab="Column",ylab="Second right singular vector")
```



---

d e variação explicada

```
svd1 <- svd(scale(dataMatrixOrdered))
par(mfrow=c(1,2))
plot(svd1$d,xlab="Column",ylab="Singular value",pch=19)
plot(svd1$d^2/sum(svd1$d^2),xlab="Column",ylab="Percent of variance explained",pch=19)
```



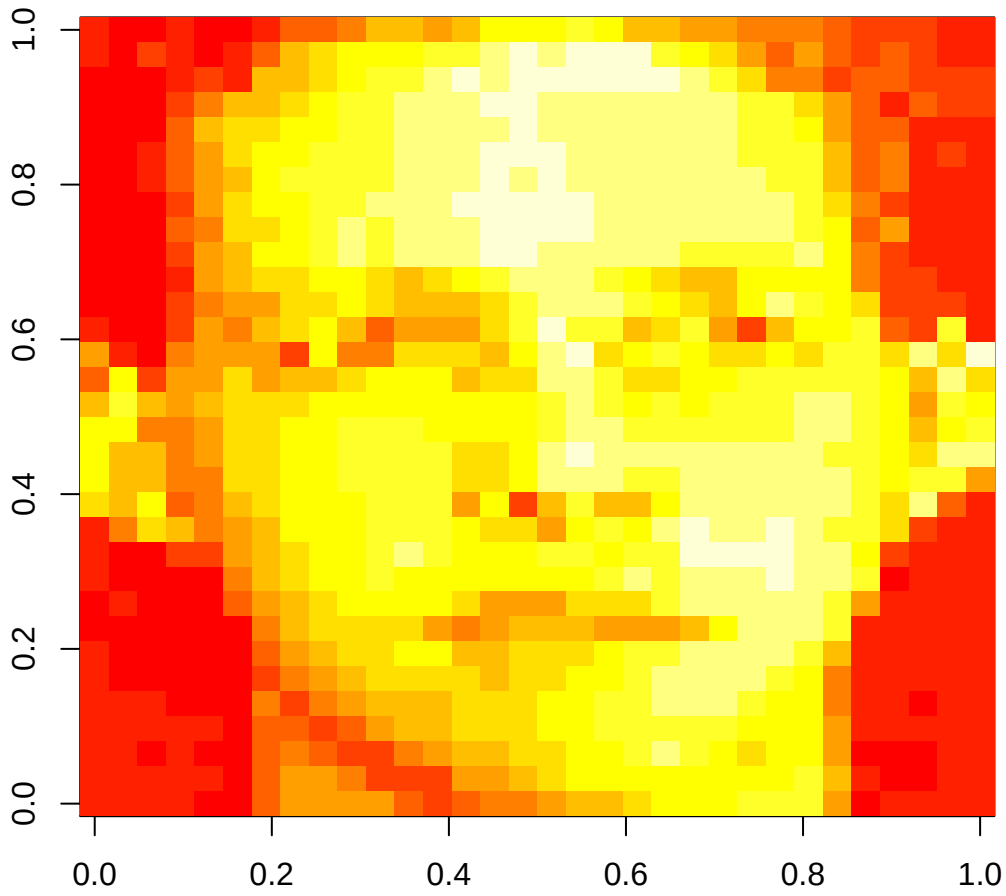
Missing values

```
dataMatrix2 <- dataMatrixOrdered
## Inserir aleatoriamente alguns dados em falta
dataMatrix2[sample(1:100,size=40,replace=FALSE)] <- NA
svd1 <- svd(scale(dataMatrix2))  ## Não funciona!
```

```
## Error in svd(scale(dataMatrix2)): infinite or missing values in 'x'
```

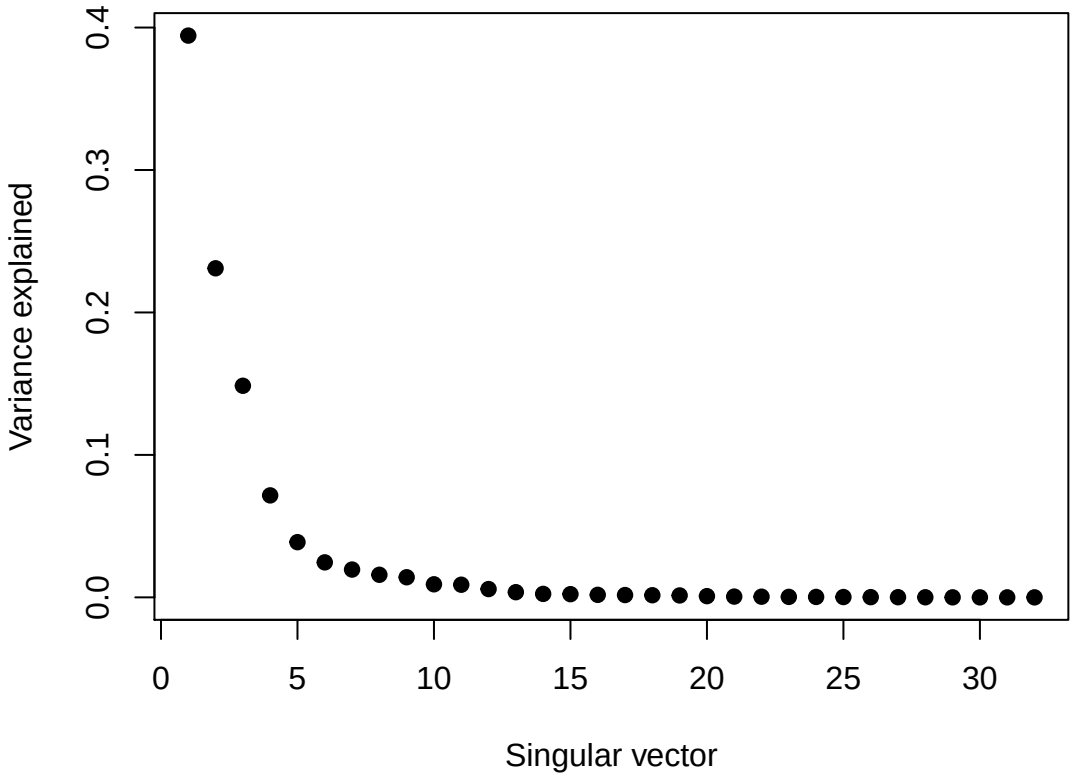
Exemplo de rosto

```
load("data/face.rda")
image(t(faceData)[,nrow(faceData):1])
```



Exemplo de face - variação explicada

```
svd1 <- svd(scale(faceData))
plot(svd1$d^2/sum(svd1$d^2),pch=19,xlab="Singular vector",ylab="Variance explained")
```



Exemplo de rosto - criar aproximações

```
svd1 <- svd(scale(faceData))

## Note que %*% é a multiplicação matricial
# Aqui svd1$d[1] é uma constante
approx1 <- svd1$u[,1] %*% t(svd1$v[,1]) * svd1$d[1]

# Nestes exemplos, precisamos fazer a matriz diagonal fora de d
approx5 <- svd1$u[,1:5] %*% diag(svd1$d[1:5])%*% t(svd1$v[,1:5])
approx10 <- svd1$u[,1:10] %*% diag(svd1$d[1:10])%*% t(svd1$v[,1:10])
```

Exemplo de rosto - criar aproximações - Plot

```
par(mfrow=c(1,4))
image(t(approx1)[,nrow(approx1):1], main = "(a)")
image(t(approx5)[,nrow(approx5):1], main = "(b)")
image(t(approx10)[,nrow(approx10):1], main = "(c)")
image(t(faceData)[,nrow(faceData):1], main = "(d)")  ## Original data
```