

# TD2 - SVD pour la compression et la réduction de dimension

Courtenay Rebecca & Ducros Chloé & Lasson Marie

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Chargement de la librairie . . . . .	1
1.2	Tracer des graphes . . . . .	1
1.3	Fonction SVD . . . . .	3
<b>2</b>	<b>Questions</b>	<b>3</b>
2.1	Question 1 . . . . .	3
2.2	Question 2 . . . . .	5
2.3	Question 3 . . . . .	6
2.4	Question 4 . . . . .	7
2.5	Question 5 . . . . .	8
2.6	Question 6 . . . . .	9
2.7	Question BONUS . . . . .	10

## 1 Introduction

Dans ce TD, on procède à une compression d'image en utilisant la SVD. Vous retrouverez toutes les fonctions dont vous aurez besoin dans la partie 2.11 du cours (page 14 du PDF).

Voici en plus quelques autres fonctions utiles.

### 1.1 Chargement de la librairie

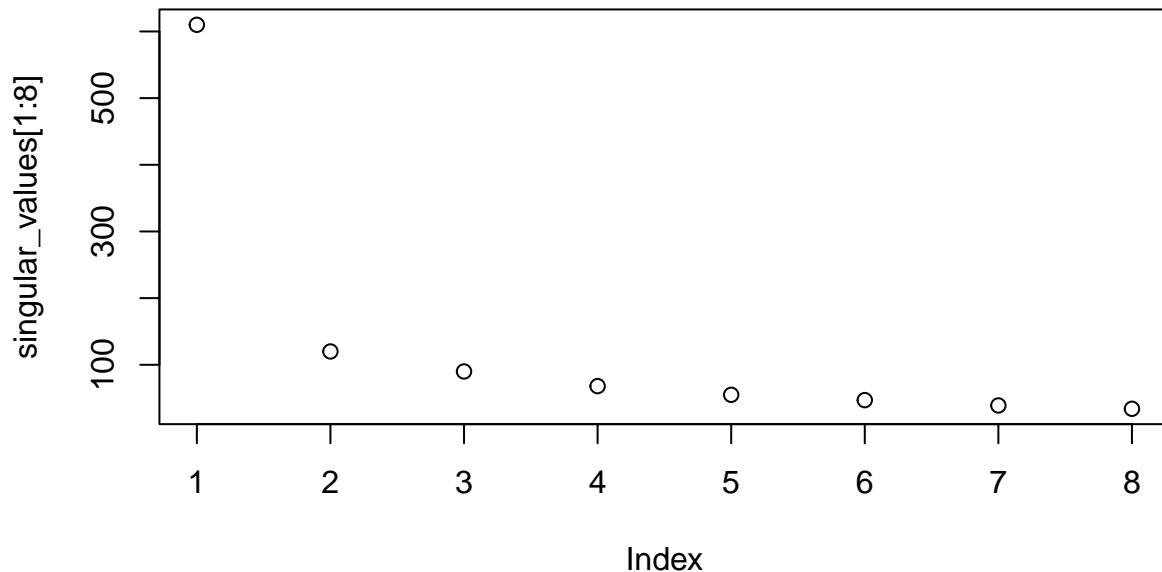
```
library(jpeg)
```

### 1.2 Tracer des graphes

Afficher les 8 premières valeurs singulières dans un graphe:

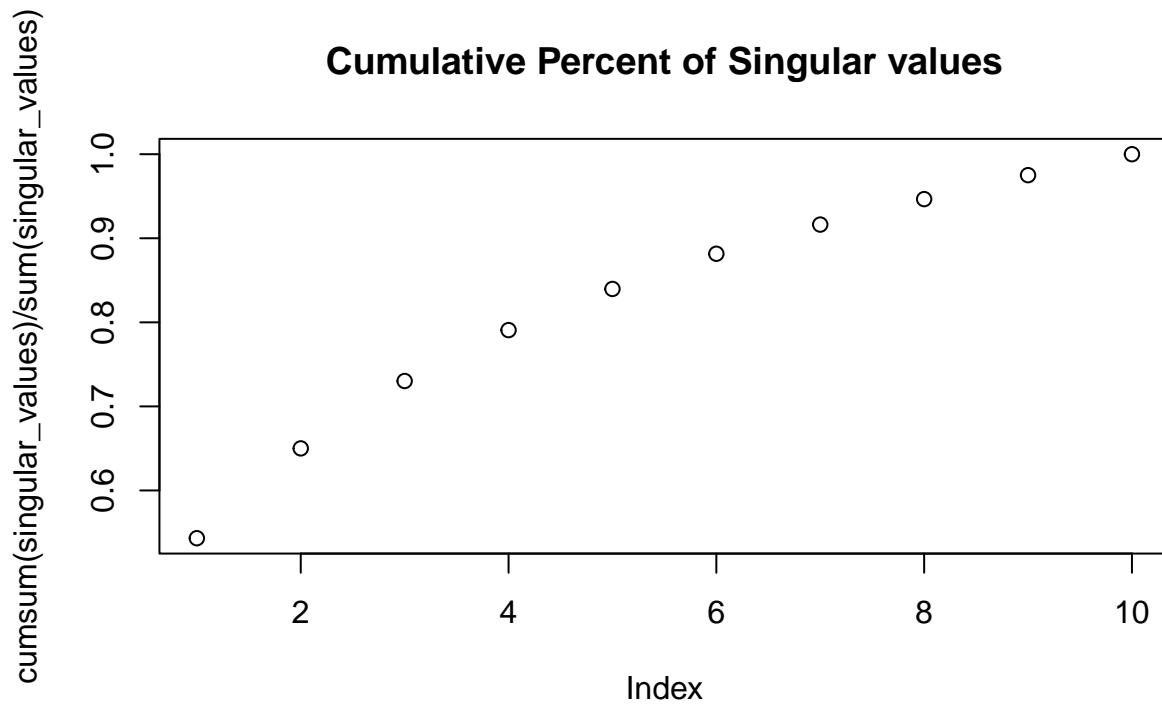
```
singular_values <- c(610,120,90,68,55,47,39,34,32,28)
plot(singular_values[1:8], main="8 first Singular values")
```

## 8 first Singular values



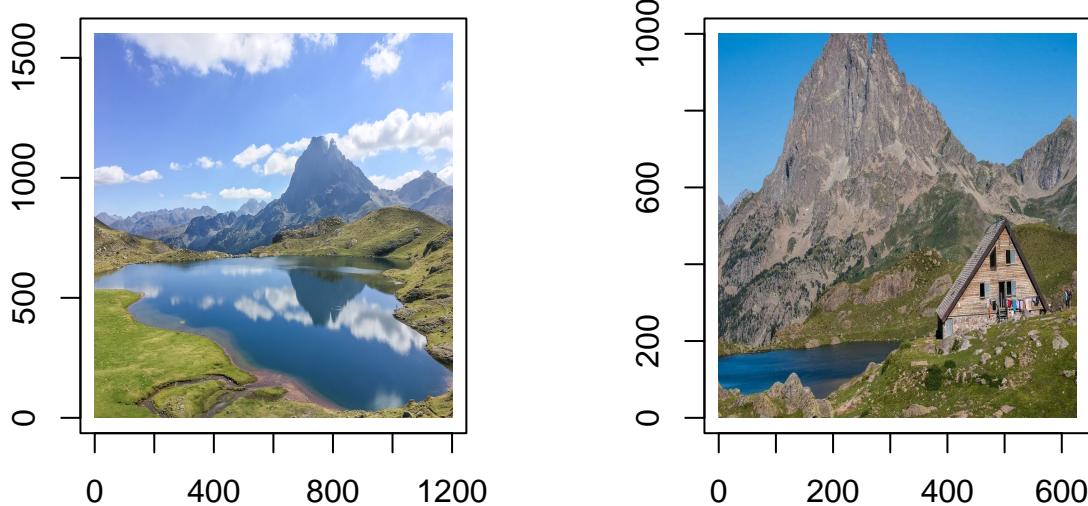
A partir de valeurs singulières, tracer le pourcentage cumulatif de leur impact :

```
singular_values <- c(610,120,90,68,55,47,39,34,32,28)
plot(cumsum(singular_values) / sum(singular_values),
     main="Cumulative Percent of Singular values")
```



Pour tracer des graphes ou des images sur  $n$  lignes et  $p$  colonnes on utilise la fonction `par(mfrow = c(n,p))` suivie de tous les appels à `plot`.

```
par(mfrow=c(1,2))
plot.image(PIC)
plot.image(refuge_ossau)
```



### 1.3 Fonction SVD

La fonction `svd` de R retourne un triplé  $(d, u, v)$  avec  $d$  le vecteur des valeurs singulières, et  $u$  et  $v$  les deux matrices orthogonales de gauche et de droite. Pour accéder par exemple au vecteur des valeurs singulières, il faut donc utiliser `X.svd$d` en supposant qu'on a stocké le résultat de la `svd` dans la variable `X.svd`.

## 2 Questions

L'ensemble des questions vaut 18 points, en plus des 2 points bonus de la question 7. La qualité du rapport sera notée sur 4 points: vous aurez ces points par exemple en labellisant vos graphes et leurs axes, en faisant un affichage correct etc. Je vous invite donc à chercher sur Internet les différents paramètres possibles pour la fonction `plot`.

### 2.1 Question 1

Importer l'image du pic du midi d'Ossau, en faire la SVD, puis tracer côté à côté l'image originale et l'image issue de la compression avec 40 composantes.

#### 2.1.1 SVD

```
r <- PIC[, , 1] #rouge
g <- PIC[, , 2] #vert
```

```

b <- PIC[, , 3] #bleu
#SVD pour chaque bande de couleurs
PIC_R_SVD <- svd(r)
PIC_G_SVD <- svd(g)
PIC_B_SVD <- svd(b)
SVD <- list(PIC_R_SVD, PIC_G_SVD, PIC_B_SVD)

```

### 2.1.2 Image et compression

```

PLOT_IMG <- function(pic, main = "") {
  h <- dim(pic)[1]
  w <- dim(pic)[2]
  plot(x = c(0, h), y = c(0, w), type = "n", main = main)
  rasterImage(pic, 0, 0, h, w)
}

VP <- function(i) {
  list(d = i$d[1:40], #racine carrée de valeurs propres [valeurs singulières]
       u = i$u[, 1:40], #vecteurs propres de droite
       v = i$v[, 1:40]) #vecteurs propres de gauche
}

COMPRESS_IMG <- function(i) {
  img.compressed <- i$u %*% diag(i$d) %*% t(i$v)
}

IMG <- function(svd) {
  dim_mini <- lapply(svd, VP)
  img <- sapply(dim_mini, COMPRESS_IMG, simplify = 'array')
  img[img < 0] <- 0
  img[img > 1] <- 1
  return(list(img = img, svd.reduced = dim_mini))
}

```

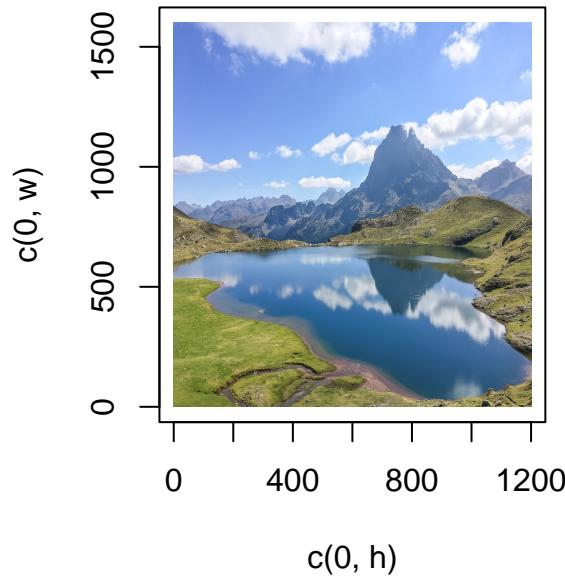
### 2.1.3 Graphique des deux images

```

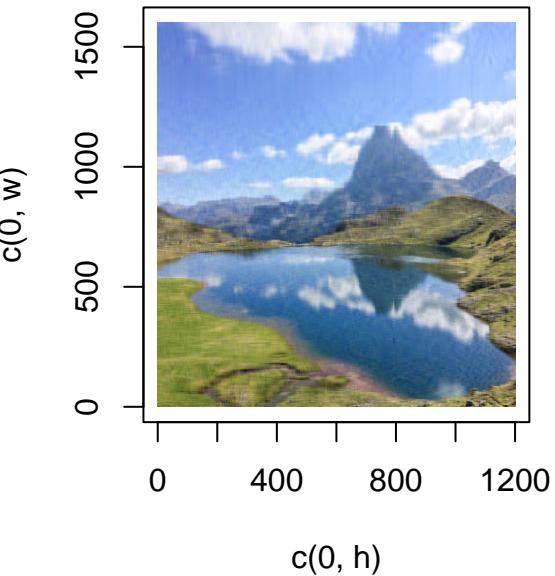
par(mfrow = c(1, 2))
PLOT_IMG(PIC, "Image d'origine")
PLOT_IMG(IMG(SVD)$img,
paste("SVD avec 40 composantes"))

```

**Image d'origine**



**SVD avec 40 composantes**

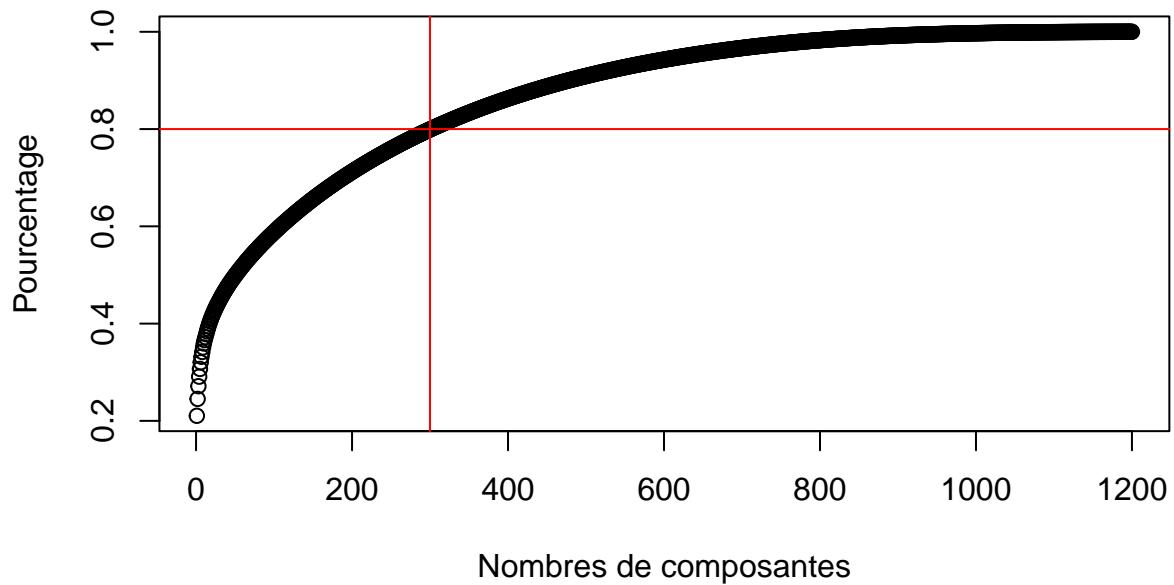


## 2.2 Question 2

Tracer le graphe du pourcentage cumulatif des valeurs singulières pour la bande rouge. Avec environ combien de composantes transmet-on (environ) 80% de l'information de l'image initiale sur la bande rouge ?

```
plot(cumsum(PIC_R_SVD$d) / sum(PIC_R_SVD$d),
     main="Pourcentage cumulatif des valeurs singulières pour la bande rouge",
     ylab="Pourcentage", xlab="Nombres de composantes", pch=1)
abline(h=0.80, col="red")
abline(v=300, col="red")
```

## Pourcentage cumulatif des valeurs singulières pour la bande rouge



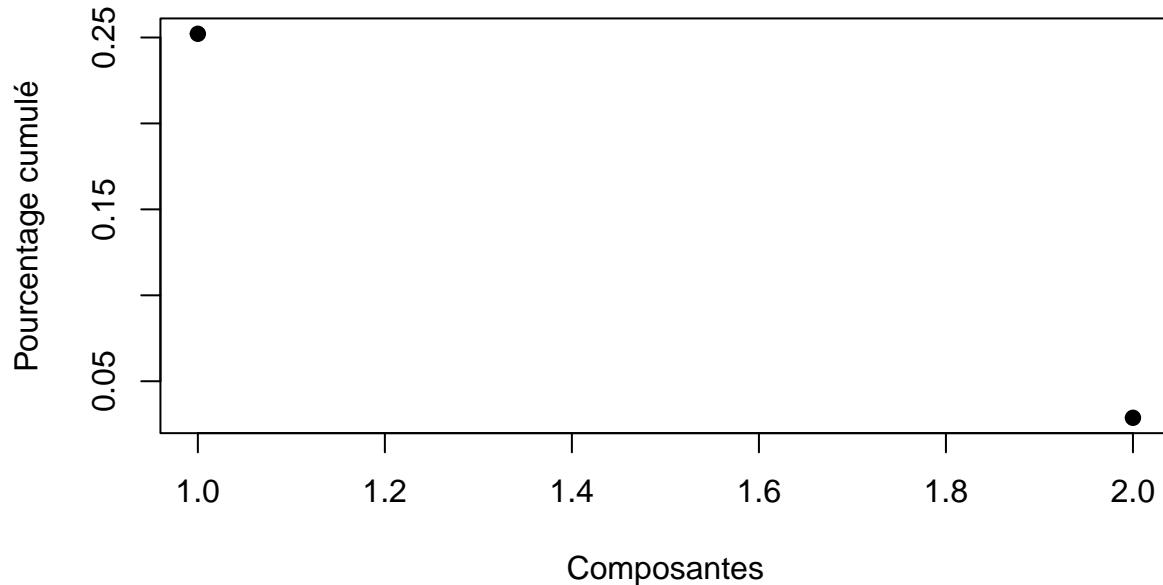
Il faut environ 300 composantes pour transmettre (environ) 80% de l'information de l'image initiale sur la bande rouge.

### 2.3 Question 3

Pour la bande verte, combien de pourcentage de l'information est représenté exactement avec la première composante (i.e. quel est le poids de la première valeur singulière par rapport au total) ? Avec la deuxième ?

```
plot((PIC_G_SVD$d [1:2] / sum(PIC_G_SVD$d) ,  
      main="Pourcentage cumulatif des valeurs singulières 1 et 2",  
      ylab="Pourcentage cumulé", xlab="Composantes", pch=19)
```

## Pourcentage cumulatif des valeurs singulières 1 et 2



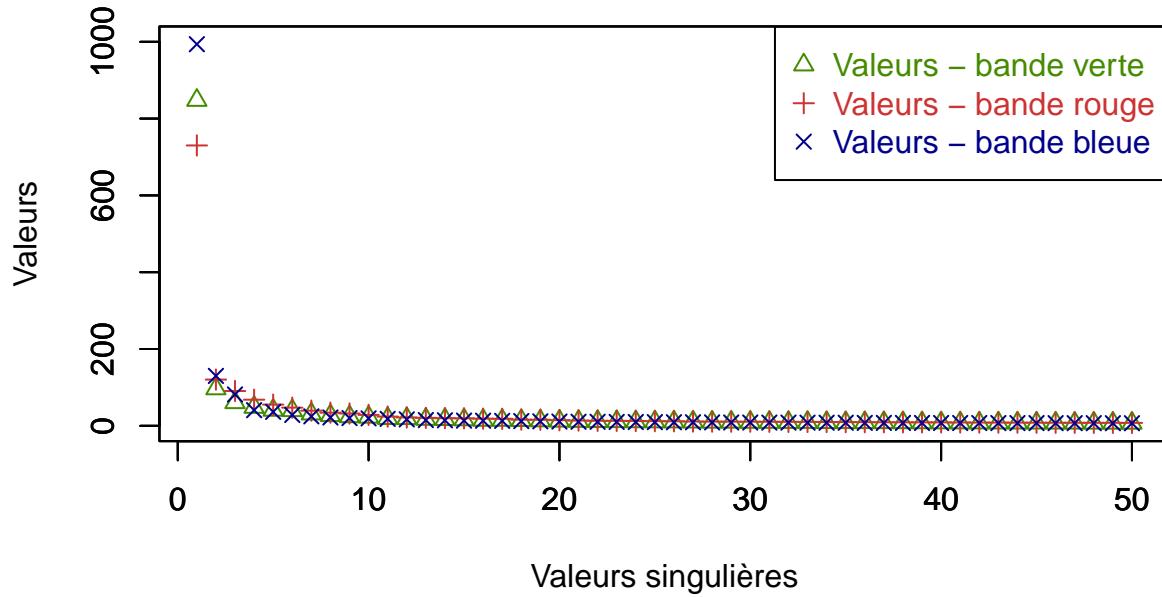
Pour la 1er valeur singulière, son poids par rapport au total est de 25%. Pour la deuxième valeur singulière, son poids est d'environ 2%.

### 2.4 Question 4

Afficher sur un graphique côté à côté (les trois sur la même ligne), les valeurs des 50 premières valeurs singulières pour chacune des trois bandes (RGB). Donner un titre à chaque graphe et labelliser les axes.

```
plot(PIC_G_SVD$d[1:50], pch=2, col="chartreuse4", ylab="Valeurs", xlab="Valeurs singulières",
      ylim=c(0,1000), main="50 premières valeurs pour les 3 bandes")
par(new = T)
plot(PIC_R_SVD$d[1:50], pch=3, col="brown3", ylab="", xlab="", ylim=c(0,1000))
par(new = T)
plot(PIC_B_SVD$d[1:50], pch=4, col="darkblue", ylab="", xlab="", ylim=c(0,1000))
legend(x="topright",
       legend=c("Valeurs - bande verte", "Valeurs - bande rouge", "Valeurs - bande bleue"),
       pch=c(2,3,4), col=c("chartreuse4", "brown3", "darkblue"),
       text.col=c("chartreuse4", "brown3", "darkblue"))
```

## 50 premières valeurs pour les 3 bandes

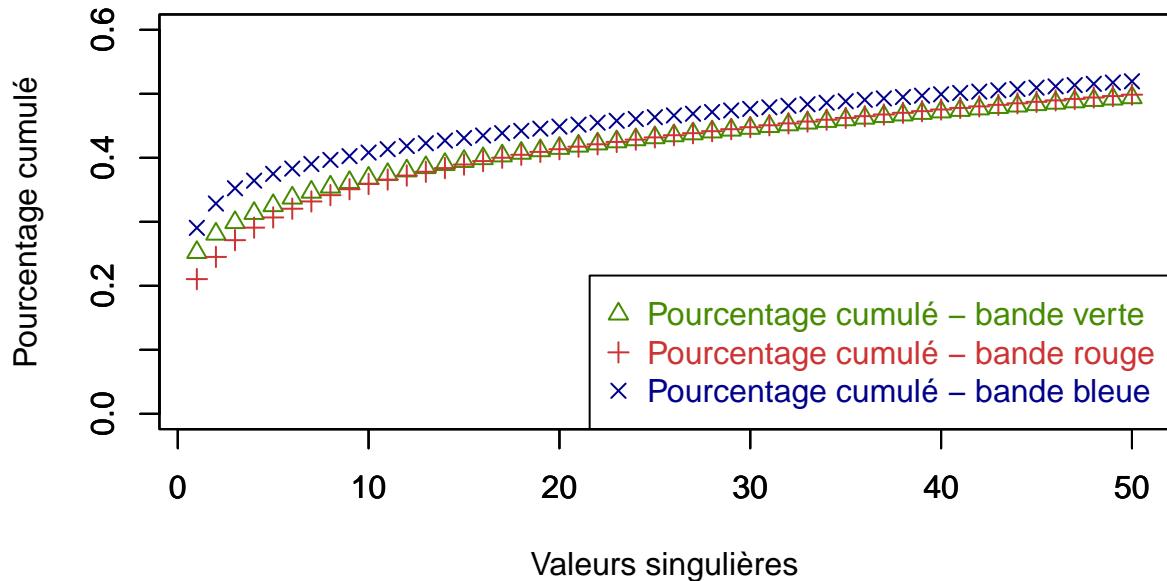


### 2.5 Question 5

Afficher sur un graphique côté à côté (les trois sur la même ligne), le pourcentage cumulatif des 100 premières valeurs singulières pour chacune des trois bandes (RGB). Donner un titre à chaque graphe et labelliser les axes.

```
plot(cumsum(PIC_G_SVD$d) [1:50] / sum(PIC_G_SVD$d), pch=2, col="chartreuse4",
      ylab="Pourcentage cumulé", xlab="Valeurs singulières", ylim=c(0,0.6),
      main="Pourcentage cumulé des 50 premières valeurs pour les 3 bandes")
par(new = T)
plot(cumsum(PIC_R_SVD$d) [1:50] / sum(PIC_R_SVD$d), pch=3, col="brown3",
      ylab="", xlab="", ylim=c(0,0.6))
par(new = T)
plot(cumsum(PIC_B_SVD$d) [1:50] / sum(PIC_B_SVD$d), pch=4, col="darkblue",
      ylab="", xlab="", ylim=c(0,0.6))
legend(x="bottomright",
       legend=c("Pourcentage cumulé - bande verte",
               "Pourcentage cumulé - bande rouge",
               "Pourcentage cumulé - bande bleue"),
       pch=c(2,3,4), col=c("chartreuse4", "brown3", "darkblue"),
       text.col=c("chartreuse4", "brown3", "darkblue"))
```

## Pourcentage cumulé des 50 premières valeurs pour les 3 bandes



### 2.6 Question 6

Tracer un graphe sur lequel apparaît le poids de l'image (en MB) en fonction du nombre de composantes sélectionnées. Le tracer avec un pas de 20, pour les 400 premières composantes.

```

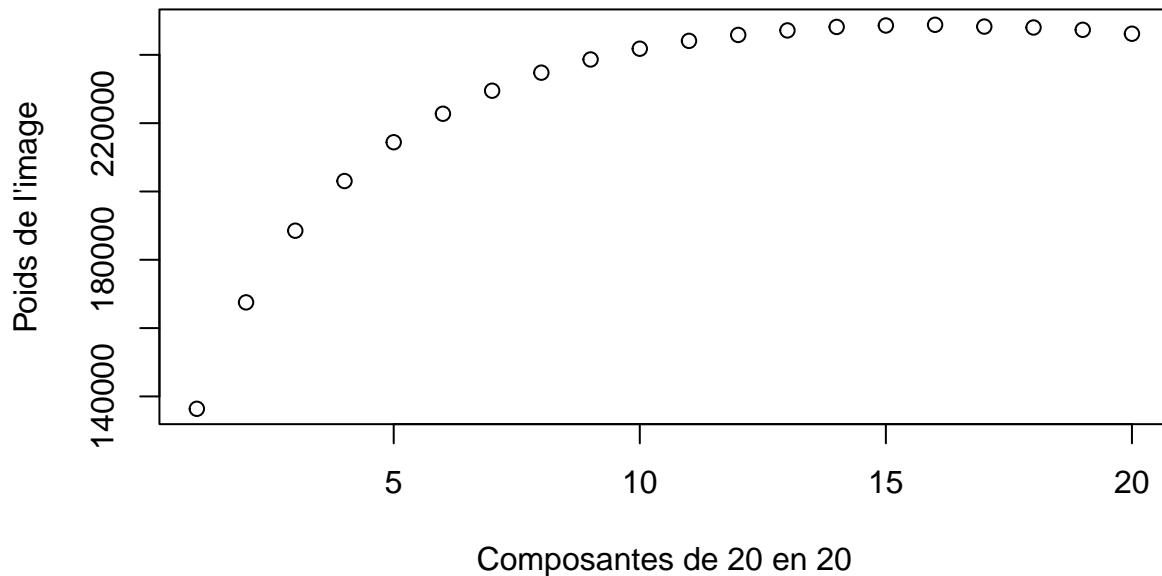
vecteur <- c()

IMG_BIS <- function(svd, nb.comp) {
  dim_mini <- lapply(svd, function(i) list(d = i$d[1:nb.comp],
                                             u = i$u[, 1:nb.comp],
                                             v = i$v[, 1:nb.comp]))
  img <- sapply(dim_mini, COMPRESS_IMG, simplify = 'array')
  img[img < 0] <- 0
  img[img > 1] <- 1
  return(list(img = img, svd.reduced = dim_mini))
}

for (index in seq.int(20, 400, 20)){
  img_size <- object.size(writeJPEG(IMG_BIS(SVD, index)$img))
  vecteur <- append(vecteur, img_size, index)
}
plot.new()
plot(vecteur, main="Poids de l'image en fonction du nombre de composantes",
      xlab="Composantes de 20 en 20", ylab="Poids de l'image")

```

## Poids de l'image en fonction du nombre de composantes



### 2.7 Question BONUS

Imaginons le scénario suivant : nous avons un satellite qui prend un certain nombre de photos par jour de la Terre, et qui transmet tous les jours ses données à un datacenter sur Terre. Il ne peut envoyer que 8GB de données par jour. Pour pouvoir envoyer plus d'images, on estime qu'on peut envoyer une image qui retransmet 50% (environ) de l'information contenue dans l'image initiale. Comparer le nombre d'images à pleine résolution au nombre d'images à 50% qu'il est possible d'envoyer. On admettra qu'une compression par SVD sur l'image satellite a le même impact qu'une compression sur l'image du pic du Midi d'Ossau, et on admettra que la taille des images est similaire. Pour répondre à cette question, vous aurez donc besoin de déterminer le nombre de composantes à utiliser pour parvenir à une résolution d'environ 50%, puis de compresser l'image et de comparer sa taille avec l'image initiale. Vous pourrez vous aider des graphiques de la question 5).

En nous aidant de la question 5, on remarque qu'en prenant 50 composantes, on peut avoir 50% de l'information, toutes bandes de couleur confondues.

#### 2.7.1 Tailles

```
img_size <- object.size(writeJPEG(IMG_BIS(SVD, 50)$img))  
img_size
```

```
## 178904 bytes  
pic_size <- object.size(PIC)  
pic_size
```

```
## 46080224 bytes
```

#### 2.7.2 Pour 8GB

De ce fait : 8GB = 8000000000 Bytes

```
img_mem <- 80000000000/img_size
img_mem

## 44716.7 bytes
pic_mem <- 80000000000/pic_size
pic_mem

## 173.6 bytes
img_mem - pic_mem

## 44543.1 bytes
```

On remarque une grande différence entre les deux mémoires. De ce fait, nous pouvons donc passer de 173 images à 44 716 images en prenant uniquement 50 composantes.