```
library(pixmap)
x <- read.pnm("xray.pnm")
plot(x)

#To reconstruct the colours correctly, the data must not be centered
red=prcomp(x@red,center = FALSE)
blue=prcomp(x@blue,center = FALSE)
green=prcomp(x@green,center = FALSE)

reconstruct <- function(k)
{
  x.reconstructed=x

  #To be able to plot the picture, the values of the colours have to be between 0 and 1.
  #So I simply replaced the wrong values by 0 or 1.
  reconst.red=as.matrix(red$x[,1:k]) %*% t(as.matrix(red$rotation))[1:k,]
  reconst.red=replace(reconst.red,reconst.red<0,0)
  reconst.red=replace(reconst.red,reconst.red>1,1)

  reconst.blue=as.matrix(blue$x[,1:k]) %*% t(as.matrix(blue$rotation))[1:k,]
  reconst.blue=replace(reconst.blue,reconst.blue<0,0)
  reconst.blue=replace(reconst.blue,reconst.blue>1,1)

  reconst.green=as.matrix(green$x[,1:k]) %*% t(as.matrix(green$rotation))[1:k,]
  reconst.green=replace(reconst.green,reconst.green<0,0)
  reconst.green=replace(reconst.green,reconst.green>1,1)

  x.reconstructed@red=reconst.red
  x.reconstructed@blue=reconst.blue
  x.reconstructed@green=reconst.green

  plot(x.reconstructed)
}

#The function returns the amount of variation that cannot be explained by the first k PCs.
info.loss <- function(k)
{
  d=data.frame(red=1-sum(red$sdev[1:k]^2)/sum(red$sdev^2),
               blue=1-sum(blue$sdev[1:k]^2)/sum(blue$sdev^2),
               green=1-sum(green$sdev[1:k]^2)/sum(green$sdev^2),
               overall=1-
sum(red$sdev[1:k]^2+blue$sdev[1:k]^2+green$sdev[1:k]^2)/sum(green$sdev^2+


          red$sdev^2+blue$sdev^2))
  return(d)
}

#Using the PCs to reconstruct the picture I only have to save a (1023 x k) matrix
(scores)
#and a (k x 746) matrix (loadings) instead of one 1023x746 matrix
comp.factor <- function(k)
{
  k*(1023+746)/(1023*746)
}

#Using only 8 principle components the foreign body is already visible. With 8 principle
#components less than 2% of the variations of every colour cannot be explained. And
#only 1.8% of the original amount of storage is needed to save the picture.
```

```
reconstruct(8)
info.loss(8)
comp.factor(8)
```