Multivariate Statistics HW1

Namwoo Kwon (UNIST Biztics 20236002)

2023-03-10

Goal : Perform the PCA of ANY photo.

Process : Create different versions of the photo

1. Intro 1 ~ 4
2. Two largest eigenvalues
3. Ten largest eigenvalues
4. 11th through 100th eigenvalues
5. 100 largest eigenvalues
6. First half of the eigenvalues
7. All the eigenvalues
8. Save the Final Result

Intro 1 : Set path of file

```
setwd('/Users/namwoo/Desktop/UNIST/lecture/1-1/Multivaraiate Statistics')
getwd()
```

```
## [1] "/Users/namwoo/Desktop/UNIST/lecture/1-1/Multivaraiate Statistics"
```

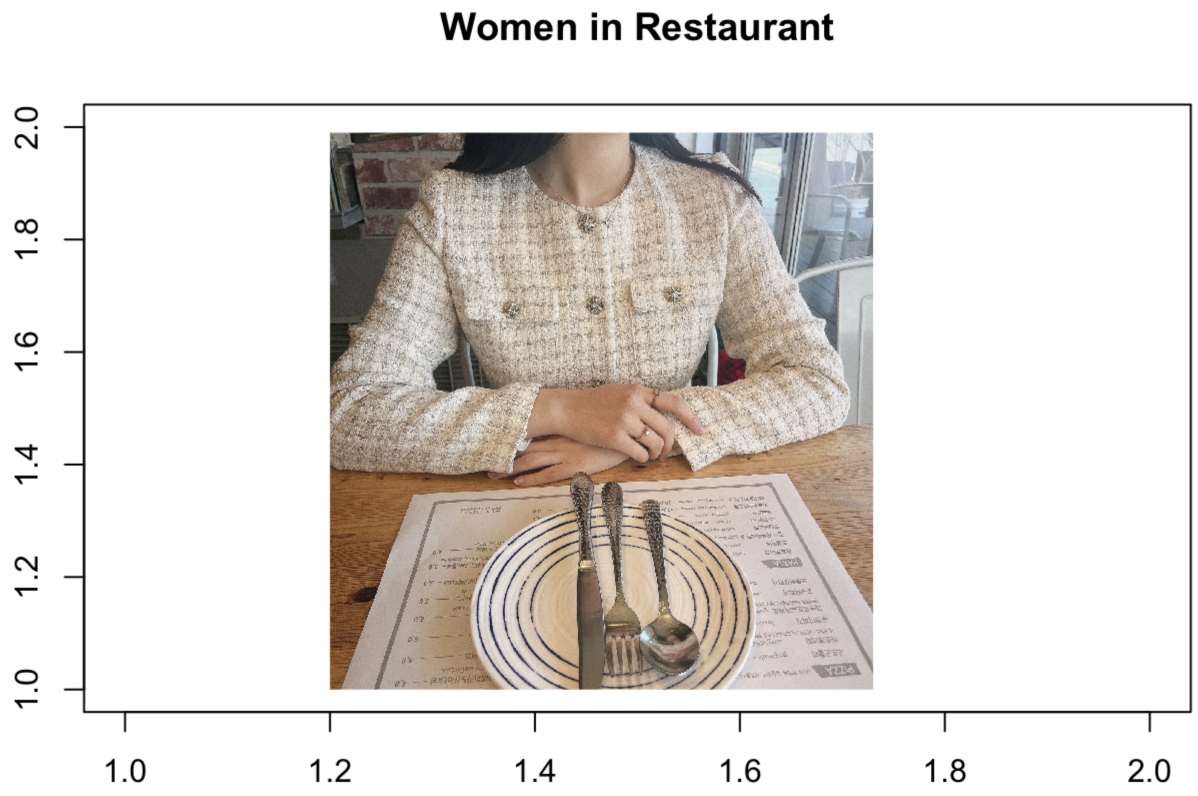
Intro 2 : Load Image File

```
# install.packages('jpeg')
library(jpeg)
img <- readJPEG('Home_Work_1_image.jpeg')
```

```
dim(img) # image size 1440, 1440, 3
```

```
## [1] 1440 1440    3
```

```
plot(1:2, type='n', main='Women in Restaurant', xlab='', ylab='')
rasterImage(as.raster(img[,,1:3]), 1.2, 1.0, 1.73, 1.99, interpolate=FALSE)
```



Intro 3 : Separate Matrix by RGB

```
r <- img[,,1] # Red
g <- img[,,2] # Green
b <- img[,,3] # Blue
```

Intro 4 : Let’s PCA

```
# 'center=F' : To reverse the data back to the original without reprocessing the mean
img.r.pca <- prcomp(r, center=F)
img.g.pca <- prcomp(g, center=F)
img.b.pca <- prcomp(b, center=F)

# Check importance of principal component
r_importance <- summary(img.r.pca)$importance
g_importance <- summary(img.g.pca)$importance
b_importance <- summary(img.b.pca)$importance
```

```
# Check 5 of PC
r_importance[, 1:5]
```

```
##                             PC1      PC2      PC3      PC4      PC5
## Standard deviation     27.56183 2.357403 2.260893 1.802444 1.139824
## Proportion of Variance  0.95615 0.006990 0.006430 0.004090 0.001640
## Cumulative Proportion   0.95615 0.963150 0.969580 0.973670 0.975300
```

```
g_importance[, 1:5]
```

```
##                             PC1     PC2      PC3      PC4      PC5
## Standard deviation     25.56004 2.29914 2.214335 1.994816 1.225098
## Proportion of Variance  0.94714 0.00766 0.007110 0.005770 0.002180
## Cumulative Proportion   0.94714 0.95480 0.961910 0.967680 0.969860
```

```
b_importance[, 1:5]
```

```
##                             PC1      PC2      PC3      PC4      PC5
## Standard deviation     23.91446 2.470779 2.254042 2.064386 1.472278
## Proportion of Variance  0.93572 0.009990 0.008310 0.006970 0.003550
## Cumulative Proportion   0.93572 0.945710 0.954030 0.961000 0.964550
```

What you can see

- PC1 (1st eigenvalue) has most of the distributed data in the original data

```
# Save the result of PCA
rgb.pca <- list(img.r.pca, img.g.pca, img.b.pca)
```

Do Home Work 1

```
# Numbers of Principal Component
num = c(2, # Two largest eigenvalues
        10, # Ten largest eigenvalues
        11, # 11th through 100th eigenvalues
        100, # 100 largest eigenvalues
        720, # First half of the eigenvalues
        1440) # All the eigenvalues
```

```
for (i in num) {
  pca.img <- sapply(rgb.pca, function(j) {
    # Restore compressed images based on their principal components
    # If HW1 part 3, Change the parameter
    if (i == 11) { compressed.img <- j$x[, 11:100]%*%t(j$rotation[, 11:100])}
    else {compressed.img <- j$x[, 1:i]%*%t(j$rotation[, 1:i])}

  }, simplify = 'array')
  # Save the Image
  # If HW1 part 3, Change the parameter
  if (i==11){writeJPEG(pca.img, paste('HW_1_PCA_', i, '- 100', '.jpg', sep=''))}
  else {writeJPEG(pca.img, paste('HW_1_PCA_', i, '.jpg', sep=''))}
}
```

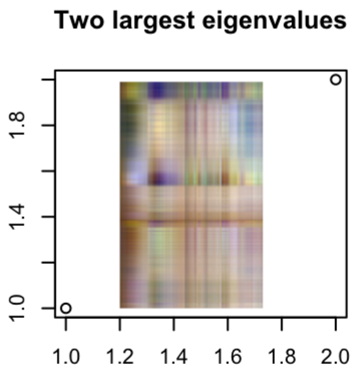
View an image of My homework’s final product

```
# Save the result of PCA
rgb.pca <- list(img.r.pca, img.g.pca, img.b.pca)
```

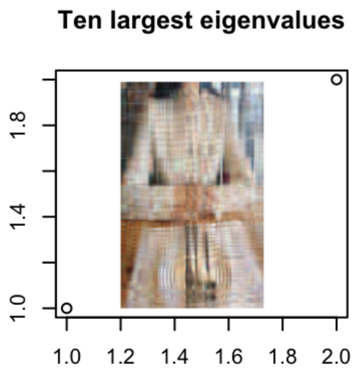
```
# Load Saved Images
img_2 <- readJPEG('HW_1_PCA_2.jpg')
img_10 <- readJPEG('HW_1_PCA_10.jpg')
img_11 <- readJPEG('HW_1_PCA_11- 100.jpg')
img_100 <- readJPEG('HW_1_PCA_100.jpg')
img_720 <- readJPEG('HW_1_PCA_720.jpg')
img_1440 <- readJPEG('HW_1_PCA_1440.jpg')
```

View Images in Report

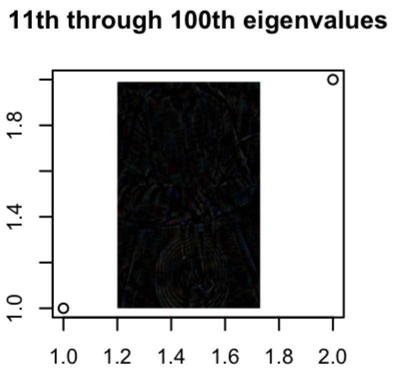
```
par(mfrow=c(2:3))

# Picture quality is very choppy
plot(1:2, main="Two largest eigenvalues",
     xlab="", ylab='',
     sub="Picture quality is very choppy")
rasterImage(img_2, 1.2, 1.0, 1.73, 1.99)

# Better than before, but not as good quality
plot(1:2, main="Ten largest eigenvalues",
     xlab="", ylab='',
     sub="Better than before, but not as good quality")
rasterImage(img_10, 1.2, 1.0, 1.73, 1.99)

# Fine lines indicate contours
plot(1:2, main="11th through 100th eigenvalues",
     xlab="", ylab='',
     sub="Fine lines indicate contours")
rasterImage(img_11, 1.2, 1.0, 1.73, 1.99)

# Shows much better quality images
plot(1:2, main="100 largest eigenvalues",
     xlab="", ylab='',
     sub=" Shows much better quality images")
rasterImage(img_100, 1.2, 1.0, 1.73, 1.99)

# Doesn't look much different from the original image
plot(1:2, main="First half of the eigenvalues",
     xlab="", ylab='',
     sub="Doesn't look much different \nfrom the original image")
rasterImage(img_720,1.2, 1.0, 1.73, 1.99)

# Doesn't look much different from the original image
plot(1:2, main="All the eigenvalues",
     xlab="", ylab='',
     sub="Doesn't look much different \nfrom the original image")
rasterImage(img_1440, 1.2, 1.0, 1.73, 1.99)
```



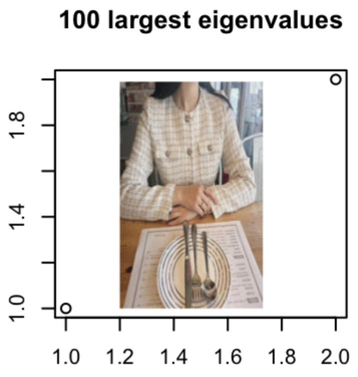
Picture quality is very choppy



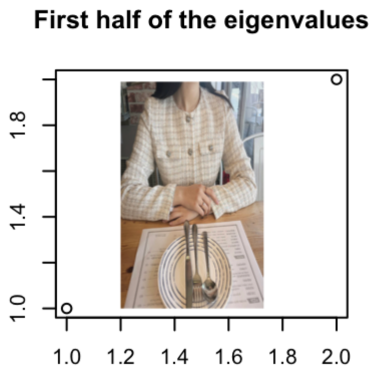
Better than before, but not as good quality



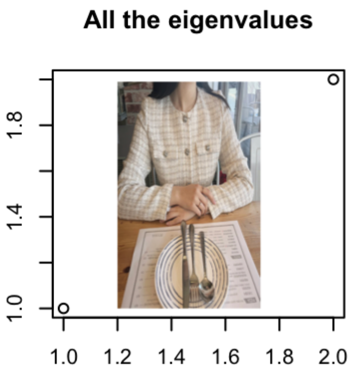
Fine lines indicate contours



Shows much better quality images



Doesn't look much different from the original image



Doesn't look much different from the original image

What you can see

- 1st through 10th eigenvalues reflect most of the variance information of the original data, but not the details.
- If we look at the case of the ‘11th through 100th eigenvalues’, we can see the details.
- So if we incorporate all the information from 1st through 100th eigenvalues, we can restore the original similar image.

Save Final Result

```
jpeg(filename="Final Result by HW1.png",
     width=600,
     height=300,
     unit="px",
     bg="transparent")

par(mfrow=c(2:3))
plot(1:2, main="Two largest eigenvalues",
     xlab="", ylab='',
     sub="Picture quality is very choppy")
rasterImage(img_2, 1.2, 1.0, 1.73, 1.99)

# Better than before, but not as good quality
plot(1:2, main="Ten largest eigenvalues",
     xlab="", ylab='',
     sub="Better than before, but not as good quality")
rasterImage(img_10, 1.2, 1.0, 1.73, 1.99)

# Fine lines indicate contours
plot(1:2, main="11th through 100th eigenvalues",
     xlab="", ylab='',
     sub="Fine lines indicate contours")
rasterImage(img_11, 1.2, 1.0, 1.73, 1.99)

# Shows much better quality images
plot(1:2, main="100 largest eigenvalues",
     xlab="", ylab='',
     sub=" Shows much better quality images")
rasterImage(img_100, 1.2, 1.0, 1.73, 1.99)

# Doesn't look much different from the original image
plot(1:2, main="First half of the eigenvalues",
     xlab="", ylab='',
     sub="Doesn't look much different \nfrom the original image")
rasterImage(img_720,1.2, 1.0, 1.73, 1.99)

# Doesn't look much different from the original image
plot(1:2, main="All the eigenvalues",
     xlab="", ylab='',
     sub="Doesn't look much different \nfrom the original image")
rasterImage(img_1440, 1.2, 1.0, 1.73, 1.99)

dev.off()
```

```
## quartz_off_screen
##                 2
```