STAT0017 ICA1

Question 1

(a) Three important properties of PCA are listed below:

i. Let $\mathbf{X} = [X_1, ..., X_p]^T$ have covariance matrix Σ with eigenvalue-eigenvector pairs $(\lambda_1, \mathbf{e}_1), ..., (\lambda_p, \mathbf{e}_p)$, where $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_p \geq 0$ and $\mathbf{e}_i = [e_{i1}, ..., e_{ip}]^T$. Then the i^{th} PC is given by

$$Y_i = e_{i1}X_1 + e_{i2}X_2 + ... + e_{ip}X_p = e_i^T\mathbf{X}$$

With them, we have $Var(Y_i) = e_i^T\Sigma e_i = \lambda_i$ and $Cov(Y_i, Y_k) = e_i^T\Sigma e_k = 0$.

ii. For the PCs Y_i defined above, we have $\sum_{i=1}^p Var(Y_i) = \sum_{i=1}^p Var(X_i)$; that is,

$$\lambda_1 + ... + \lambda_p = \sigma_{11} + ... + \sigma_{pp}$$

iii. For the PCs Y_i defined above, we have the correlation coefficients between the PC Y_i and the original variable X_k

$$\rho_{Y_i,X_k} = \frac{Cov(Y_i, X_k)}{\sqrt{Var(Y_i)}\sqrt{Var(X_k)}} = \frac{e_{ik}\lambda_i}{\sqrt{\lambda_i}\sqrt{\sigma_{kk}}} = \frac{e_{ik}\sqrt{\lambda_i}}{\sqrt{\sigma_{kk}}}$$

(b)

1. In this case the variances for $X_1, ..., X_4$ are 0.69, 0.19, 3.12, 0.58 respectively, which are fairly similar (on the same scale), thus we can use the covariance matrix directly.
2. The first priciple component is $Y_1 = -0.361X_1 + 0.085X_2 - 0.857X_3 - 0.358X_4$; the second principle component is $Y_2 = 0.657X_1 + 0.730X_2 - 0.173X_3 - 0.075X_4$.
3. The variance explained by each PC is 4.23, 0.24, 0.08, 0.02 respectively, and the scree plot is as follows

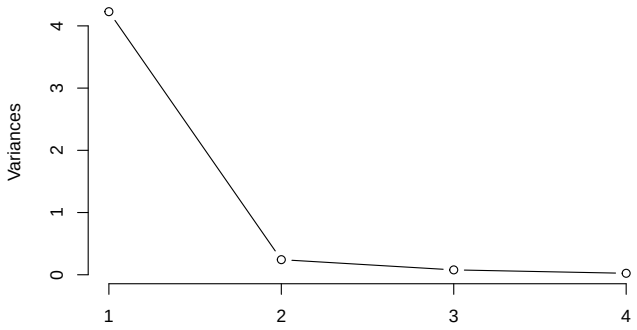

Figure 1: Scree plot of 4 PCs

We can see that the plot clearly flattens out after the second component, thus we can conclude that only the first two PCs should be retained to reduce the dimension of this dataset.

4.

i. The following code indicates that $Var(Y_i) = \lambda_i$ and $Cov(Y_i, Y_k) = 0$ which approves Property i.

```
round(cov(PCA$x), 5)
```

```
##         PC1     PC2     PC3     PC4
## PC1 4.22824 0.00000 0.00000 0.00000
## PC2 0.00000 0.24267 0.00000 0.00000
## PC3 0.00000 0.00000 0.07821 0.00000
## PC4 0.00000 0.00000 0.00000 0.02384
```

```
round(lambda <- PCA$sdev^2, 5)
```

```
## [1] 4.22824 0.24267 0.07821 0.02384
```

ii. The following code indicates that $\sum_{i=1}^{p} Var(Y_i) = \sum_{i=1}^{p} Var(X_i)$ which approves Property ii.

```
sum(cov(PCA$x))
```

```
## [1] 4.572957
```

```
sum(apply(dta[, 1:4], 2, var))
```

```
## [1] 4.572957
```

iii. The following code indicates that $Cov(Y_i, X_k) = e_{ik}\lambda_i$ which approves Property iii.

```
cov(PCA$x, dta[, 1:4])
```

```
##                X1          X2           X3          X4
## PC1 -1.528029859  0.35738162 -3.622210384 -1.51493333
## PC2  0.159334888  0.17718882 -0.042072474 -0.01831704
## PC3  0.045520264 -0.04676231 -0.005962385 -0.04268920
## PC4 -0.007519667  0.00762063  0.011437007 -0.01796349
```

```
t(PCA$rotation) * lambda
```

```
##                X1          X2           X3          X4
## PC1 -1.528029859  0.35738162 -3.622210384 -1.51493333
## PC2  0.159334888  0.17718882 -0.042072474 -0.01831704
## PC3  0.045520264 -0.04676231 -0.005962385 -0.04268920
## PC4 -0.007519667  0.00762063  0.011437007 -0.01796349
```

Question 2

(a) For $k, l = 1, ..., p$, the canonical variables have the following properties:

i. $Var(U_k) = Var(V_k) = 1$

ii. $Corr(U_k, U_l) = Corr(V_k, V_l) = 0$

iii. $Corr(U_k, V_l) = 0$

(b)

1. The first pair of canonical variable U_1 and V_1 is

$$(U_1, V_1) = (-1.069X_1 + 0.855X_2, -0.849X_3 + 0.694X_4)$$

2. The following code calculates the covariance matrix of canonical variables

```
U <- CCA$scores$xscores
V <- CCA$scores$yscores
round(cov(cbind(U, V)), 4)
```

```
##       [,1]   [,2]  [,3]   [,4]
## [1,] 1.000 0.0000 0.941 0.0000
## [2,] 0.000 1.0000 0.000 0.1239
## [3,] 0.941 0.0000 1.000 0.0000
## [4,] 0.000 0.1239 0.000 1.0000
```

We see that the diagonal of the above matrix are all 1, which approves Property i. Also, all the entries in the upper-left 2×2 sub-matrix and lower-right 2×2 sub-matrix except their diagonals are zero, which approves Property ii. Furthermore, all the entries in the upper-right 2×2 sub-matrix and lower-left 2×2 sub-matrix except their diagonals (the canonical correlations) are zero, which approves Property iii.

Question 3

Three similarities between PCA, CCA and Fisher's multi-class LDA are listed below:

1. All three methods are "linear", that is, they are all used to reduce the demension of a high-dimensional dataset by mapping points via a linear projection into a lower-dimensional section.
2. All three methods aim to maximize a specific function subject to some constraint. For PCA, we maximize $\frac{\mathbf{a^T \Sigma a}}{\mathbf{a^T a}}$; for CCA, we maximize $\frac{\mathbf{a^T \Sigma_{12} b}}{\sqrt{\mathbf{a^T \Sigma_{11} a}}\sqrt{\mathbf{b^T \Sigma_{22} b}}}$; for Fisher's multi-class LDA, we maximize $\frac{\hat{\mathbf{a}}^\mathbf{T}\mathbf{B}\hat{\mathbf{a}}}{\hat{\mathbf{a}}^\mathbf{T}\mathbf{W}\hat{\mathbf{a}}}$.
3. The optimal vectors of interest for all three methods are related to eigenvectors of a specific matrix. For PCA, the ith PC is given by $\mathbf{e_i^T X}$ where e_i is the eigenvector of the covariance matrix Σ; for CCA, $\mathbf{a} = \mathbf{\Sigma_{11}^{-1/2} e}$ and $\mathbf{b} = \mathbf{\Sigma_{22}^{-1/2} f}$ where $\mathbf{e}$ and $\mathbf{f}$ are the eigenvectors of $\Sigma_{11}^{-1/2}\Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}\Sigma_{11}^{-1/2}$ and $\Sigma_{22}^{-1/2}\Sigma_{21}\Sigma_{11}^{-1}\Sigma_{12}\Sigma_{22}^{-1/2}$ respectively; for Fisher's multi-class LDA, the ith discriminant is $\hat{\mathbf{e}}_\mathbf{i}^\mathbf{T}\mathbf{x}$ where $\hat{\mathbf{e}}_\mathbf{i}$ is the eigenvector of $\mathbf{W^{-1}B}$.

Question 4

(a) The foollowing R code was used to perform LDA:

```
# Divide dataset for training and testing
train.set <- as.logical(c(rep(1, 40), rep(0, 110)))
test.set <- dta[!train.set, ]

# Perform LDA using training set and then predict using test set
lda.fit <- lda(Class ~ ., data = dta, subset = train.set)
lda.pred <- predict(lda.fit, test.set)

# Calculate the accuracy
lda.class <- lda.pred$class
dta.class <- test.set[, "Class"]
cat("The classification accuracy rate for LDA is", round(mean(lda.class == dta.class)*100, 3), "%.")
```

```
## The classification accuracy rate for LDA is 69.091 %.
```

(b) The foollowing R code was used to perform QDA:

```
qda.fit <- qda(Class ~ ., data = dta, subset = train.set)
qda.pred <- predict(qda.fit, test.set)
qda.class <- qda.pred$class

cat("The classification accuracy rate for QDA is", round(mean(qda.class == dta.class)*100, 3), "%.")
```

```
## The classification accuracy rate for QDA is 83.636 %.
```

(c) From the result given above, we see that QDA performs better than LDA in this case. It might due to unequal population covariance matrices, that is, $\Sigma_1 \neq \Sigma_2$. The following tables are the covariance matrices for Class A and B respectively, and it is fairly clear that these matrices are different. For example, $Var(X3)$ is 4.39 for Class A but only 0.22 for Class B. We can apply a further CLX test (Cai, Liu, and Xia 2013) to test the equality of these two sample covariance matrices. From the results shown below we clearly reject the null hypothesis that the two classes have the same covariance matrices, which means $\Sigma_1 \neq \Sigma_2$. Therefore, using a pooled covariance matrix is not appropriate and QDA is preferred.

Table 1: Covariance Matrix for Class A

	X1	X2	X3	X4
X1	0.89	-0.09	1.79	0.74
X2	-0.09	0.17	-0.43	-0.18
X3	1.79	-0.43	4.39	1.87
X4	0.74	-0.18	1.87	0.84

Table 2: Covariance Matrix for Class B

	X1	X2	X3	X4
X1	0.27	0.09	0.18	0.06
X2	0.09	0.10	0.08	0.04
X3	0.18	0.08	0.22	0.07
X4	0.06	0.04	0.07	0.04

```
testCov(dta[dta$Class == 'A', 1:4], dta[dta$Class == 'B', 1:4], method = "CLX")
```

```
##
##  Two-Sample CLX test
##
## data:  dta[dta$Class == "A", 1:4] and dta[dta$Class == "B", 1:4]
## Statistic = 744.35, p-value < 2.2e-16
## alternative hypothesis: two.sided
```

Question 5

In this question we aim to improve the classification accurary of QDA. Since the division of test and training sets has been given in Question 4 (Though I was a bit confused why the size of training set 40 is way smaller than that of test set 110), in order to avoid invidious comparisons between QDA and its extension, we use the same division as above throughout Question 5 and we will not consider cross-validation where we have to redefine the division.

(a) We notice that in Question 4, we use all the explanatory variables as the features for the classification rule. To improve the accurary of QDA, we would like to take full advantage of the information given by the variables but ignore some trivial effect. We could try to reduce the dimension of the features by: Method 1 - using only a few principle components generated by PCR with large eigenvalue (Næs and Mevik 2001); Method 2 - using only a subset of the original variables.

For Method 1, we first retain the principal components with large eigenvalue as we did in Question 1-(b)-3, then we use the new variables (retained components) Y_1, Y_2 as the features to perform QDA. Here is the code for the procedures:

```
dta2 <- dta                                                    # Replicate dataset

dta2$Y1 <- (t(PCA$rotation[, 1:2]) %*% t(dta2[, 1:4]))[1, ]    # Calculate Y1 using PC1
dta2$Y2 <- (t(PCA$rotation[, 1:2]) %*% t(dta2[, 1:4]))[2, ]    # Calculate Y2 using PC2
test.set2 <- dta2[!train.set, ]                                # Define test set

# Calculate the accuracy
qda.fit3 <- qda(Class ~ Y1 + Y2, data = dta2, subset = train.set)
qda.pred3 <- predict(qda.fit3, test.set2)
qda.class3 <- qda.pred3$class
cat("The classification accuracy rate for QDA Method 1 is",
    round(mean(qda.class3 == dta.class)*100, 3), "%.")
```

```
## The classification accuracy rate for QDA Method 1 is 83.636 %.
```

The accuracy for QDA Method 1 here is the same as the above QDA in Question 4-(b). However, if we use another division, for example swap the training set and test set, the accurary is 70% for QDA whilst 97.5% for QDA Method 1, where the accurary for the modified QDA is 27.5% greater.

For Method 2, we write a loop to perform QDA using all possible subsets of the four original variables ($2^4 - 1 = 15 possibilities$), then we compute their classification accuracy and choose the best one. Here is the code for the procedures:

```
vars <- c("Class","X1","X2","X3","X4")                # Store variable names
N <- list(1, 2, 3, 4)                                 # Define element indices
COMB <- sapply(N, function(m) combn(x=vars[2:5], m))  # Generate all combinations of elements
COMB2 <- list()                                       # Store all possible formulas
k = 0                                                 # Iteration number

# Generate all possible formulas
for(i in seq(COMB)){
  tmp <- COMB[[i]]
  for(j in seq(ncol(tmp))){
    k <- k + 1
    COMB2[[k]] <- formula(paste("Class", "~", paste(tmp[,j], collapse=" + ")))
  }
}
```

```r
# Calculate the accuracy for all possible combinations
res <- vector(mode="list", length(COMB2))
res.pred <- vector(mode="list", length(COMB2))
res.class <- vector(mode="list", length(COMB2))
res.acc <- vector(mode="numeric", length(COMB2))

for(i in seq(COMB2)){
  res[[i]] <- qda(COMB2[[i]], data = dta, subset = train.set)
  res.pred[[i]] <- predict(res[[i]], test.set)
  res.class[[i]] <- res.pred[[i]]$class
  res.acc[i] <- mean(res.class[[i]] == dta.class)
}

opt <- which(res.acc == max(res.acc))           # The optimal subset with largest accuracy
res[opt]                                        # Group means of the selected subset
```

```
## [[1]]
## Call:
## qda(COMB2[[i]], data = dta, subset = train.set)
##
## Prior probabilities of groups:
##     A     B
## 0.725 0.275
##
## Group means:
##         X2       X4
## A 3.158621 1.403448
## B 2.681818 1.318182
```

```r
cat("The classification accuracy rate for QDA Method 2 is",
    round(mean(res.class[[opt]] == dta.class)*100, 3), "%.")
```

```
## The classification accuracy rate for QDA Method 2 is 91.818 %.
```

From the results shown above we see that the optimal subset for QDA Method 2 is $\{X_2, X_4\}$ with accuracy 91.8%, which is better than the original QDA (83.6%).

(b) From Lecture 5, we define the Quadratic discrimination score for the i^{th} class π_i as

$$d_i^Q(x) = ln(p_i) - \frac{1}{2}ln|\Sigma_i| - \frac{1}{2}(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)$$

which can also be written as

$$d_i^Q(x) = ln(p_i) - \frac{1}{2}\sum_{k=1}^{K}\frac{(t_{ik})^2}{\lambda_{ik}} - \frac{1}{2}ln|\sum_{k=1}^{K}(e_{ik})(\lambda_{ik})(e_{ik}^T)|$$

where e_{ik} is the k^{th} eigenvector of the covariance matrix Σ_i for group i, λ_{ik} is the corresponding eigenvalue, t_{ik} is the score/coordinate for the new sample along eigenvector k in group i.

For Method 1, since the eigenvalues are used as the denominators (reciprocal), a relatively small eigenvalue will have a great influence on $d_i^Q(x)$. If directions with no predictive power are estimated imprecisely, they will represent noise. Since the small eigenvectors and eigenvalues may be very imprecise estimates for their population analogues, the noise effect may be more serious for the smaller eigenvalues than for the larger.

The estimates are uncertain thus can weaken the classification power of $d_i^Q(x)$ and lead to very unstable classification rules. Therefore, Method 1 is expected to perform better when there exists small eigenvalues; however, it will perform wrose if necessary PCs with large eigenvalues are removed.

For Method 2, we remove unnecessary explanatory variables to reduce the effect of multicollinearity. Taking a look at the correlation matrix of the four variables, we see that $Corr(X_1, X_3) = 0.872$, $Corr(X_1, X_4) = 0.818$, $Corr(X_3, X_4) = 0.963$, which means X_1, X_2, X_3 are highly correlated thus probably only one of them should be retained. And the optimal result above coincides with our guess. Using this method, we select the optimal subset by looking at the accuracy of predicting the test set, so the subset must be the best option if using the same training and test sets. However, if the size of the test set used is small, then the difference in accuracy amongst all the combinations will not be large enough to distinguish, which means a bad combination would probably be chosen. When this combination is used to predict a new dataset, the accurary will not be as good as some others. Therefore, a large size of the test set is preferred for this method.

References

Cai, Tony, Weidong Liu, and Yin Xia. 2013. "Two-Sample Covariance Matrix Testing and Support Recovery in High-Dimensional and Sparse Settings." *Journal of the American Statistical Association* 108 (501). Taylor & Francis Group: 265–77.

Næs, Tormod, and Bjørn-Helge Mevik. 2001. "Understanding the Collinearity Problem in Regression and Discriminant Analysis." *Journal of Chemometrics* 15 (4). Chichester, UK: John Wiley & Sons, Ltd: 413–26.