For this homework, we will be working with a study from Gene Expression Omnibus (GEO) with the accession GDS2880. This is an Affymetrix microarray experiment (HGU133A array). The data researchers were investigating patient matched normal and stage 1 or stage 2 clear cell renal cell carcinoma (cRCC) tumors to provide insight into the molecular pathogenesis of cRCC. We will be conducting outlier analysis using various methods to identify aberrant samples, followed by missing value imputation to assess the accuracy of two different algorithms.

1.) Download and load the renal cell carcinoma data file into R. Make sure that the row names are in the correct location (Affymetrix fragment names). Look at the dimensions and verify that you have 22 arrays and 22,283 probesets.

```
>HW1 <- "C:/Users/fermi/Documents/Fall 2020/HW1/renal_cell_carcinoma.txt"
>HW1_anno <- "C:/Users/fermi/Documents/Fall
2020/HW1/renal_carcinoma_annotation.txt"

>data<-read.table(HW1, header=T)
>anno<-read.table(HW1_anno)

> dim(data)

Output:
[1] 22283    22
```

2.) Label the header columns of your data frame maintaining the GSM ID, but adding the Normal/Tumor identity.

```
>sub.data<-subset(data, select= anno[,1])

#Initially tried some more fancy coding using a loop, but didn't work so resorted to manual

> names(sub.data)[names(sub.data) == "GSM146778"] <- "GSM146778_Normal"
> names(sub.data)[names(sub.data) == "GSM146780"] <- "GSM146780_Normal"
> names(sub.data)[names(sub.data) == "GSM146782"] <- "GSM146782_Normal"
> names(sub.data)[names(sub.data) == "GSM146784"] <- "GSM146784_Normal"
> names(sub.data)[names(sub.data) == "GSM146786"] <- "GSM146786_Normal"
> names(sub.data)[names(sub.data) == "GSM146789"] <- "GSM146789_Normal"
> names(sub.data)[names(sub.data) == "GSM146790"] <- "GSM146790_Normal"
> names(sub.data)[names(sub.data) == "GSM146792"] <- "GSM146792_Normal"
> names(sub.data)[names(sub.data) == "GSM146794"] <- "GSM146794_Normal"
> names(sub.data)[names(sub.data) == "GSM146798"] <- "GSM146798_Normal"
> names(sub.data)[names(sub.data) == "GSM146796"] <- "GSM146796_Normal"
> View(sub.data)
> names(sub.data)[names(sub.data) == "GSM146779"] <- "GSM146779_Tumor"
> names(sub.data)[names(sub.data) == "GSM146781"] <- "GSM146781_Tumor"
> names(sub.data)[names(sub.data) == "GSM146783"] <- "GSM146783_Tumor"
> names(sub.data)[names(sub.data) == "GSM146785"] <- "GSM146785_Tumor"
> names(sub.data)[names(sub.data) == "GSM146787"] <- "GSM146787_Tumor"
> names(sub.data)[names(sub.data) == "GSM146788"] <- "GSM146788_Tumor"
> names(sub.data)[names(sub.data) == "GSM146791"] <- "GSM146791_Tumor"
> names(sub.data)[names(sub.data) == "GSM146799"] <- "GSM146799_Tumor"
> names(sub.data)[names(sub.data) == "GSM146793"] <- "GSM146793_Tumor"
> names(sub.data)[names(sub.data) == "GSM146795"] <- "GSM146795_Tumor"
```

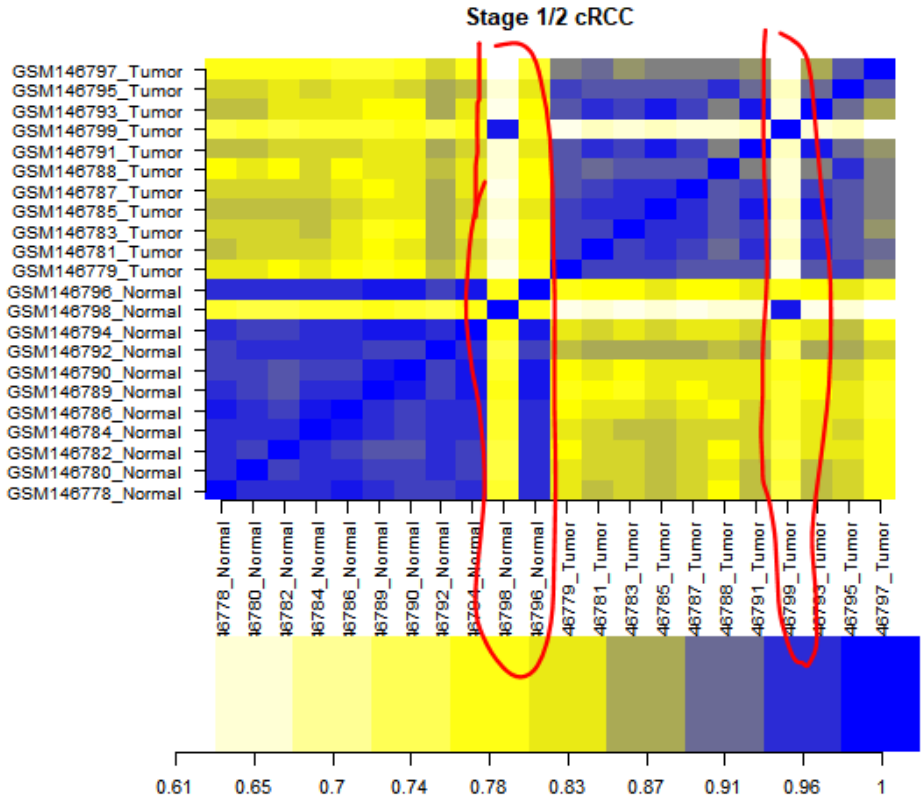
**> names(sub.data)[names(sub.data) == "GSM146797"] <- "GSM146797_Tumor"**

3.) Identify any outlier samples using the following visual plots:

**Outlier samples identified in red.**

4.) Correlation plot (heat map)

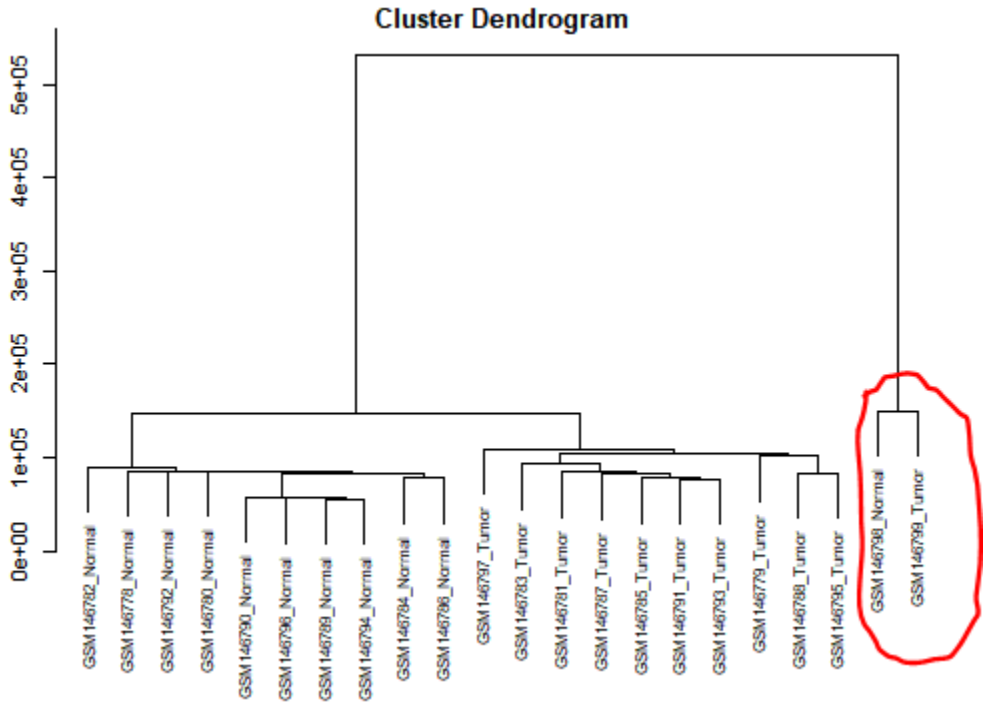
```
>library(gplots)
>dat.cor <- cor(sub.data,use="pairwise.complete.obs")

>layout(matrix(c(1,1,1,1,1,1,1,1,2,2), 5, 2, byrow = TRUE))
>par(oma=c(5,7,1,1))
>cx <- rev(colorpanel(25,"blue","yellow","white"))
>leg <- seq(min(dat.cor,na.rm=T),max(dat.cor,na.rm=T),length=10)
>image(dat.cor,main="Stage 1/2 cRCC",axes=F,col=cx)
>axis(1,at=seq(0,1,length=ncol(dat.cor)),label=dimnames(dat.cor)[[2]],cex.axis=0.9,las=2)
>axis(2,at=seq(0,1,length=ncol(dat.cor)),label=dimnames(dat.cor)[[2]],cex.axis=0.9,las=2)

>par(mar=c(1,1,1,1))
>image(as.matrix(leg),col=cx,axes=F)
>tmp <- round(leg,2)
>axis(1,at=seq(0,1,length=length(leg)),labels=tmp,cex.axis=1)
```




**#The outliers from the correlation plot are samples GSM146798_Normal, and GSM146799_Tumor.**

Hierarchical clustering dendrogram

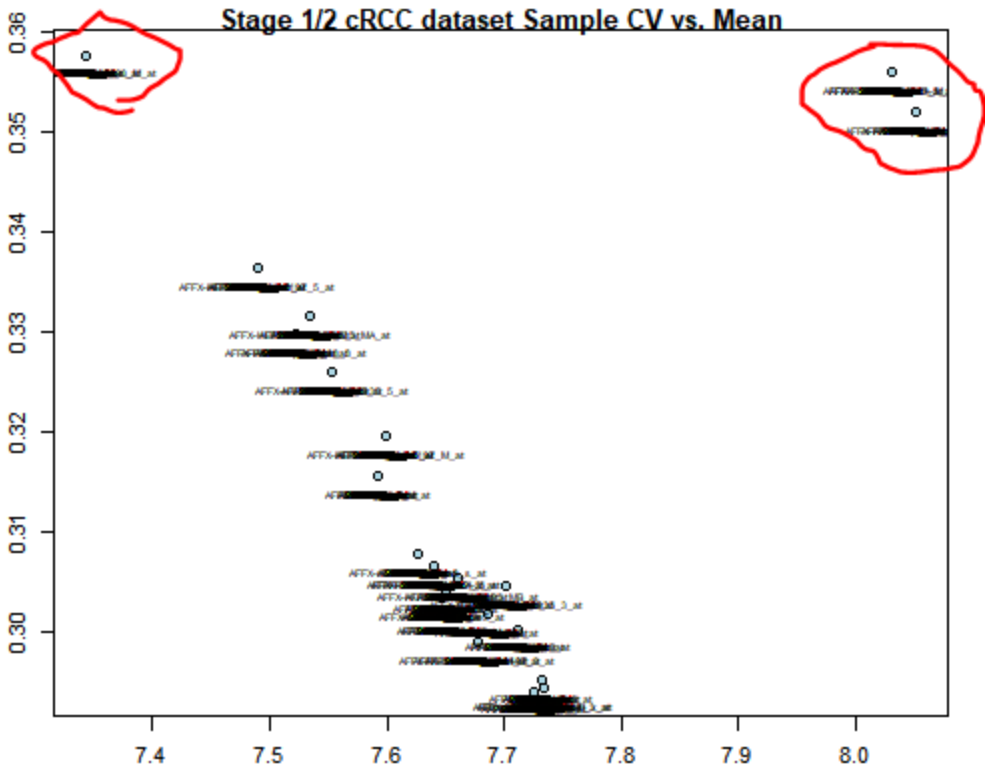
```
>dat<-t(sub.data)
>dat.dist<-dist(dat,method="euclidean")
>dat.clust<-hclust(dat.dist,method="single")
>plot(dat.clust,labels=names(dat),cex=.75)
```




#The outliers from the hierarchical clustering dendogram are samples GSM146798_Normal, and GSM146799_Tumor.

**#CV vs. mean plot**

```
>dat.mean <- apply(log2(sub.data),2,mean)
>dat.sd <- sqrt(apply(log2(sub.data),2,var))
>dat.cv <- dat.sd/dat.mean

>plot(dat.mean,dat.cv,main="Stage 1/2 cRCC dataset Sample CV vs.
Mean",xlab="Mean",ylab="CV",col='blue',cex=1.5,type="n")
>points(dat.mean,dat.cv,bg="lightblue",col=1,pch=21)
>text(dat.mean,dat.cv,label=dimnames(dat)[[2]],pos=1,cex=0.5)
```




**>sort(dat.mean)** **#unable to read crowded labels, so seeing values**

```
GSM146793_Tumor GSM146779_Tumor  GSM146791_Tumor  GSM146787_Tumor  GSM146785_Tumor GSM146778_Normal  GSM146781_Tumor GSM146789_Normal GSM146794_Normal
       7.343864        7.490723        7.522828        7.533817        7.553271        7.591612        7.599608        7.627258        7.639380
GSM146790_Normal GSM146780_Normal GSM146786_Normal  GSM146788_Tumor GSM146796_Normal  GSM146795_Tumor GSM146783_Tumor GSM146782_Normal GSM146784_Normal
       7.646792        7.650113        7.654387        7.659890        7.677633        7.685823        7.701110        7.711031        7.726157
 GSM146797_Tumor GSM146792_Normal GSM146798_Normal  GSM146799_Tumor
       7.731688        7.733859        8.031200        8.050734
```

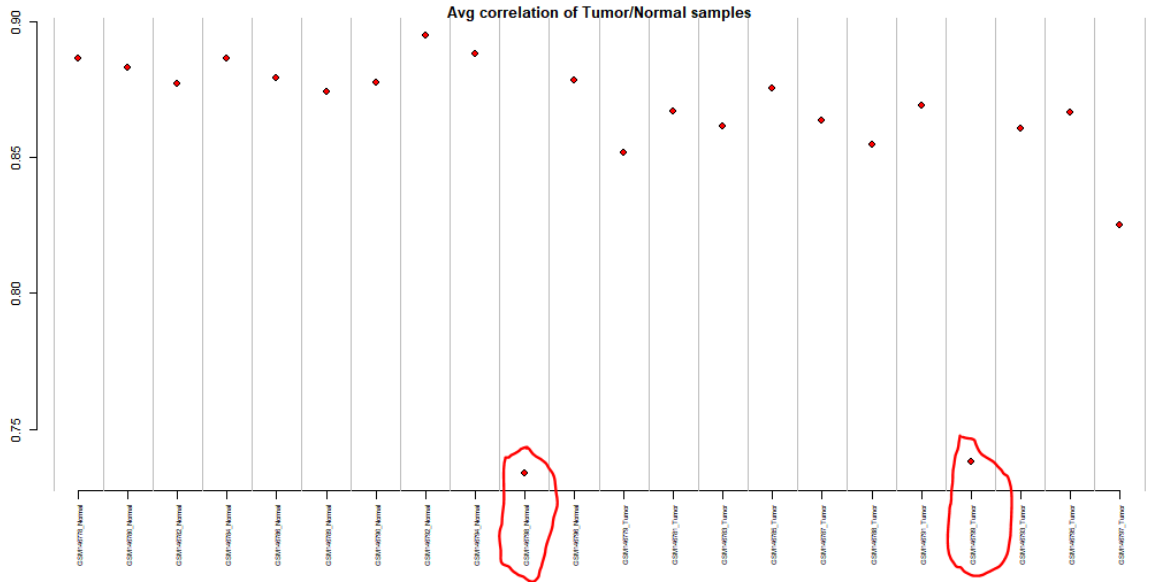
**#The outliers from the CV vs Mean plot is sample GSM146793_Tumor on the low end, and samples GSM146798_Normal, and GSM146799_Tumor on the higher end.**

Average correlation plot

**>dat.avg <- apply(dat.cor,1,mean)**
**>par(oma=c(3,0.1,0.1,0.1))**

**>png(filename="figure.png", width=900, bg="white") #y labels were off page without this**
**>par(mar=c(5,6,4,1)+.1)**
**>dev.off()**

**>plot(c(1,length(dat.avg)),range(dat.avg),type="n",xlab="",ylab="Avg r",main="Avg correlation of Tumor/Normal samples",axes=F)**
**>points(dat.avg,bg="red",col=1,pch=21,cex=1.25)**
**>axis(1,at=c(1:length(dat.avg)),labels=dimnames(sub.data)[[2]],las=2,cex.lab=0.4,cex.axis=0.6)**
**>axis(2)**
**>abline(v=seq(0.5,62.5,1),col="grey")**




```
> sort(dat.avg)
GSM146798_Normal   GSM146799_Tumor
       0.7337567          0.7379351
```

**#The outliers from the average correlation plot are samples GSM146798_Normal, and GSM146799_Tumor.**

For all plots, make sure you label the points appropriately, title plots, and label axes. You will also need to provide a legend for the correlation plot. You can use the gplots for a color gradient, or just use the default colors.

6.) Install and load the impute library.

```
>if (!requireNamespace("BiocManager", quietly = TRUE))
        install.packages("BiocManager")

>BiocManager::install("impute")
```

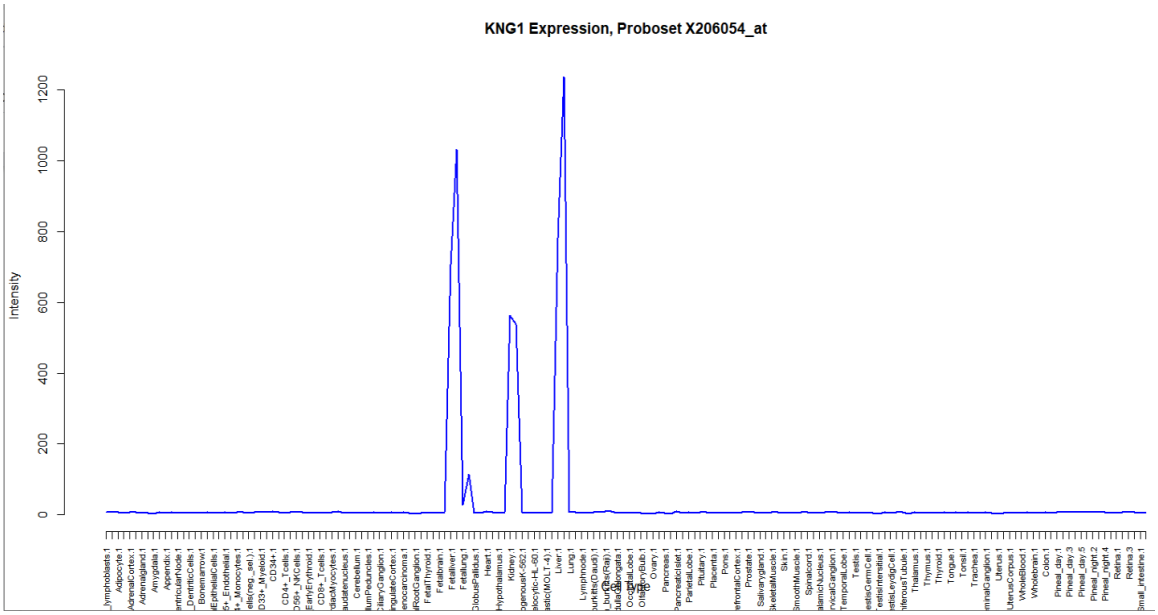
7.) Remove the outlier samples you identified in the first part of this assignment.

```
> sub.data<-sub.data[,-10] #GSM146798
> sub.data<-sub.data[,-18] #GSM146799
```

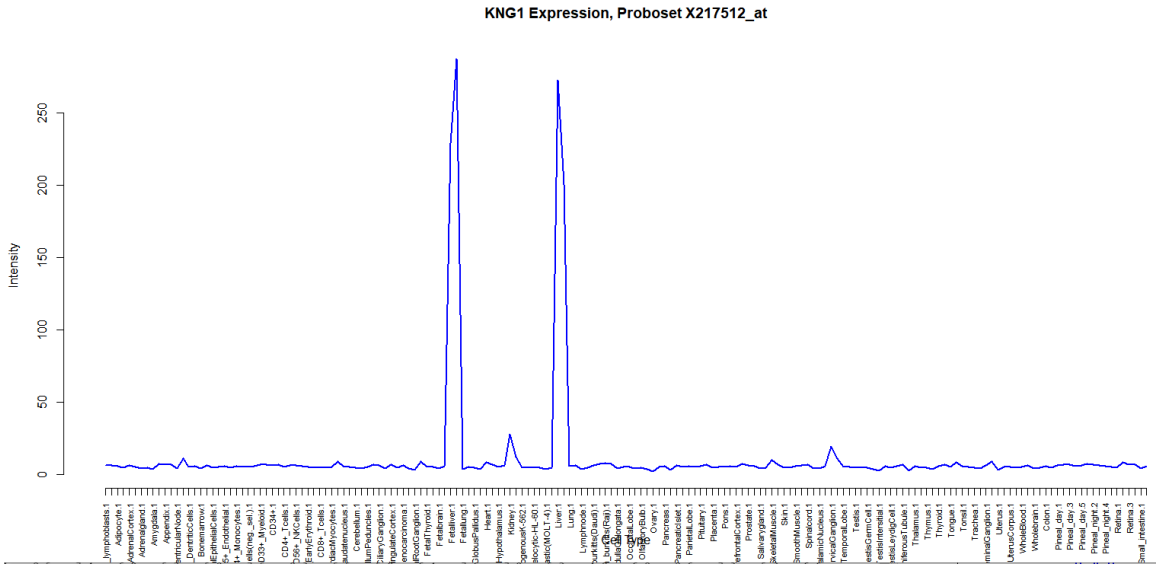
8.) Now we are going to use a couple of transcripts that were determined in this study to be indicative of normal renal function. The genes we will assess are kininogen 1 (KNG1) and aquaporin 2 (AQP2). Using either NetAffx or Gene Cards websites (or other resources, if you like), extract the probesets for these two genes. Hint: KNG1 has two while AQP2 has one. Then plot a profile plot (expression intensity vs. samples) for each probeset for these two genes. You may have to convert the data frame row to a vector to plot it. Do the plots of these genes seem to indicate normal renal function? Explain.

```
#******Files Downloaded from: http://biogps.org/#goto=genereport&id=359
>KNG1 <- "C:/Users/fermi/Documents/Fall 2020/HW1/KNG1.txt"
>AQ2 <- "C:/Users/fermi/Documents/Fall 2020/HW1/AQ2.txt"

>KNG1Data<-read.table(KNG1, header=T)
>AQ2Data<-read.table(AQ2, header=T)
```

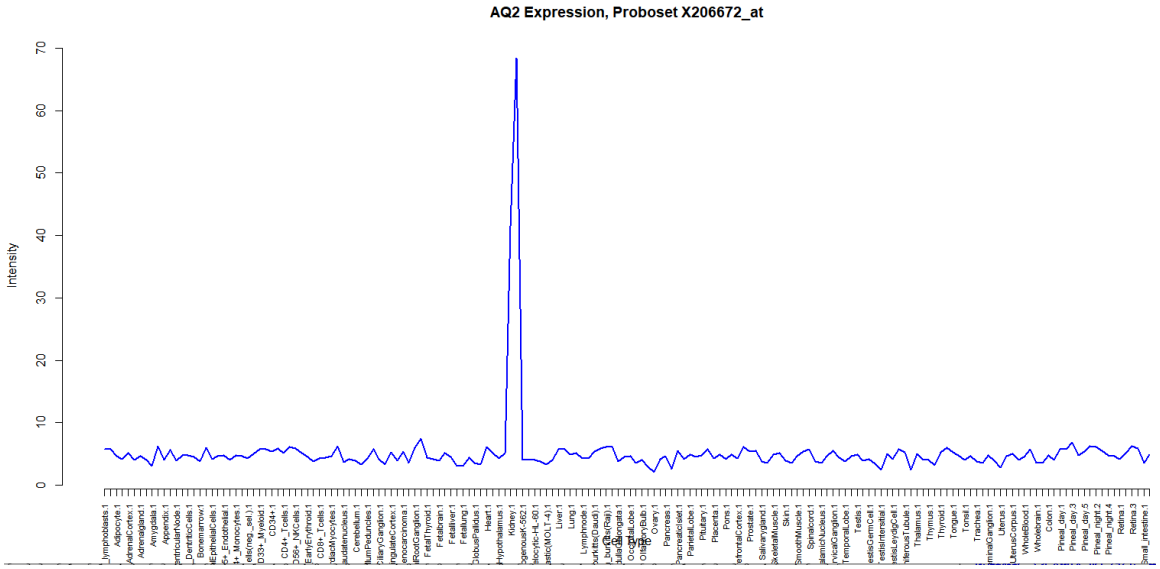
**#First KNG1 Proboset - X206054_at**
**#X= KNG1Data[,1]; Y = KNG1Data[,2]**
**>plot(as.numeric(KNG1Data[,2]),type='l',lwd=2,col='blue', main="KNG1 Expression, Proboset X206054_at" ,xlab="Cell Type", ylab="Intensity",axes=F)**
**>axis(1,at=c(1:length(KNG1Data[,1])),labels=as.vector(KNG1Data[,1]),las=2,cex.axis=0.7)**
**>axis(2)**




**#Second KNG1 Proboset - X217512_at**
**#X= KNG1Data[,1]; Y = KNG1Data[,3]**
**>plot(as.numeric(KNG1Data[,3]),type='l',lwd=2,col='blue', main="KNG1 Expression, Proboset X217512_at" ,xlab="Cell Type", ylab="Intensity",axes=F)**
**>axis(1,at=c(1:length(KNG1Data[,1])),labels=as.vector(KNG1Data[,1]),las=2,cex.axis=0.7)**
**axis(2)**




**#AQ2 Proboset X206672_at**
**#X= AQ2Data[,1]; Y= AQ2Data[,2]**
**>plot(as.numeric(AQ2Data[,2]),type='l',lwd=2,col='blue', main="AQ2 Expression, Proboset X206672_at" ,xlab="Cell Type", ylab="Intensity",axes=F)**
**>axis(1,at=c(1:length(AQ2Data[,1])),labels=as.vector(AQ2Data[,1]),las=2,cex.axis=0.7)**
**>axis(2)**




AQ2:
Kidney.1 41.5
Kidney.2 68.4

KNG1:
Fetallung.2 114.8 5.0
Kidney.2 537.5 12.2
Kidney.1 564.0 27.8
Fetalliver.1 705.9 229.0
Liver.1 781.7 273.0

Fetalliver.2 1032.0 287.6
Liver.2 1237.1 200.7

**#***Plots and Normal Renal Function Explanation***#**

**The plots show that KNG1 and AQ2 have among the highest expression on kidney cells compared to other cells. Since renal describes the function of kidneys, and these genes have high expression in kidneys, I would hypothesize that KNG1 and AQ2 are vital for kidney function. Thus, I would believe that their high expression is part of normal kidney function.**

9.) We want to assess the accuracy of missing value imputation. So assign the KNG1 probeset (206054_at) an NA value, only for array GSM146784. Be sure to first save the original value before replacing it with an NA. Also cast the data frame to a matrix to run this function.

```
>nine.val<-sub.data['206054_at','GSM146784_Normal'] #8385.3

> sub.data['206054_at','GSM146784_Normal']<-NA #confirming

>dataMatrix<-data.matrix(sub.data, rownames.force = NA)
```

10.) Now estimate the missing values in the array using 6 nearest neighbors and Euclidean distance with the impute.knn() function.

```
#installing impute package for function impute.knn()
>if (!requireNamespace("BiocManager", quietly = TRUE))
   install.packages("BiocManager")
>BiocManager::install("impute")
>library(impute)

>knn<-impute.knn(dataMatrix, k=6, rowmax=0.5, colmax=0.8, maxp=1500,
      rng.seed=362436069)

> knnData<-knn[["data"]]
```

11.) Look at the value that was imputed for your gene and calculate the relative error of this value using the actual value that you saved.

```
> knnData['206054_at','GSM146784_Normal']
[1] 7559.533

#relative error = (actual value - predicted)/actual
>(nine.val-knnData['206054_at','GSM146784_Normal'])/nine.val
[1] 0.09847789
```

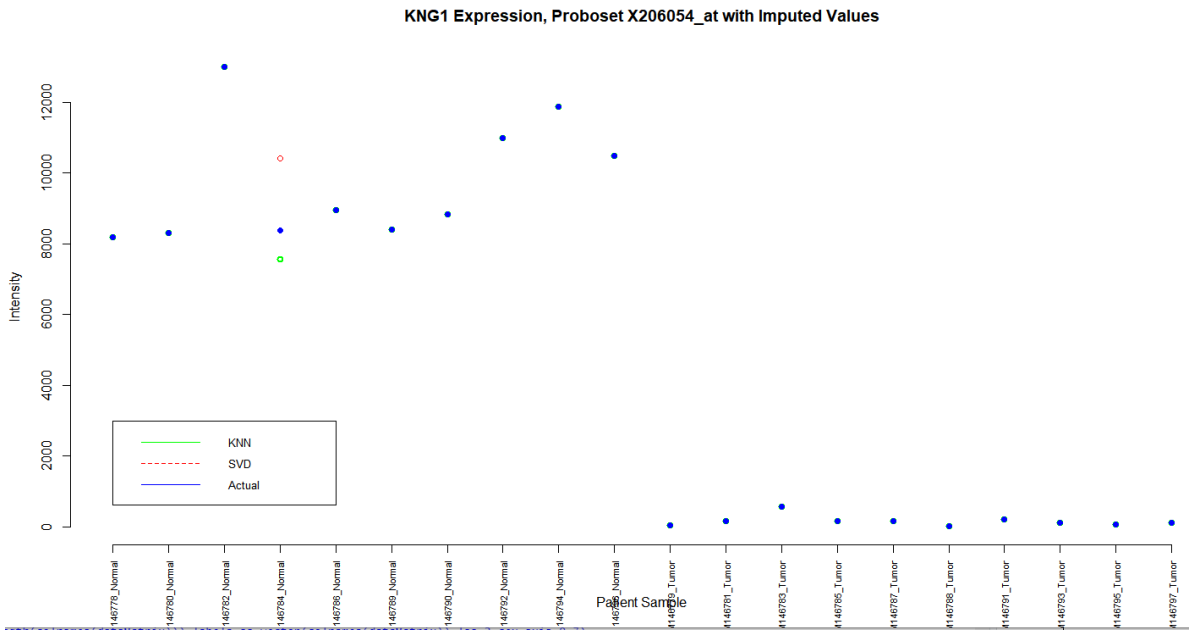
12.) Now impute the missing values using the SVD imputation method. This is in the pcaMethods package and the function is called pca ()with method svdImpute and set nPcs=9. To retrieve the output matrix, see the help file.

```
#installing proper package
>if (!requireNamespace("BiocManager", quietly = TRUE))
   install.packages("BiocManager")
>BiocManager::install("pcaMethods")
>library(pcaMethods)
```

```
> result<-pca(dataMatrix, method="svdImpute", nPcs=9)
> SVDData<-completeObs(result)

> SVDData['206054_at','GSM146784_Normal']
[1] 10418
```

13.) Finally, plot a gene profile plot of the probeset for this gene, where the two different imputed values are represented as different colored points and the actual value is a third point.

```
> dataMatrix['206054_at','GSM146784_Normal']<-nine.val

> plot(as.numeric(knnData['206054_at',]),lwd=2,col='green', main="KNG1 Expression, Proboset
    X206054_at with Imputed Values" ,xlab="Patient Sample", ylab="Intensity",axes=F)
>axis(1,at=c(1:length(colnames(dataMatrix))),labels=as.vector(colnames(dataMatrix)),las=2,cex.
    axis=0.7)
>axis(2)

>points(SVDData['206054_at',], col= 'red')
>points(dataMatrix['206054_at',], col='blue',pch=19)


>legend(1, 3000, legend=c("KNN", "SVD", "Actual"),
    col=c("green", "red", "blue"), lty=1:2, cex=0.8)
```




Generate the code and plots for each. Turn in the visuals, code, and an explanation of the questions asked. Paste all information into a PDF doc.