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.

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. (2pts.)

```
dat = read.table("E:\\renal_cell_carcinoma.txt", header = T, row.names = 1)
ann = read.table("E:\\renal_carcinoma_annotation.txt", header = F)
dim(dat)
```

- [1] 22283 22

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

```
cl = as.character(ann[,1])
dat = dat[,cl]
```

reindexed the columns of the dat table to match the annotation file. Columns 1-10 are the normal identity, 11-21 are tumor identity.

```
Normal = dat[,1:10]
Tumor = dat[,11:21]
```

3.) Identify any outlier samples using the following visual plots: ***MAKE SURE TO STATE WHAT THE OULIERS ARE IN WORDS***

a. The correlation heat map shows two potential outliers, being the probesets from GSM146798(normal) and GSM164799(tumor).
b. The Cluster dendrogram shows two potential outliers as GSM146798(Normal) and GSM146799(tumor).
c. The CV vs Mean graph showed two outliers, the same as the dendrogram with GSM146798(Normal) and GSM146799(tumor). I decided not to count GSM146793 as an outlier as it fit in with the trend of the graph even though it was separated from many other data points.
d. The average correlation plot again showed GSM146798(Normal) and GSM146799(tumor) as outliers.

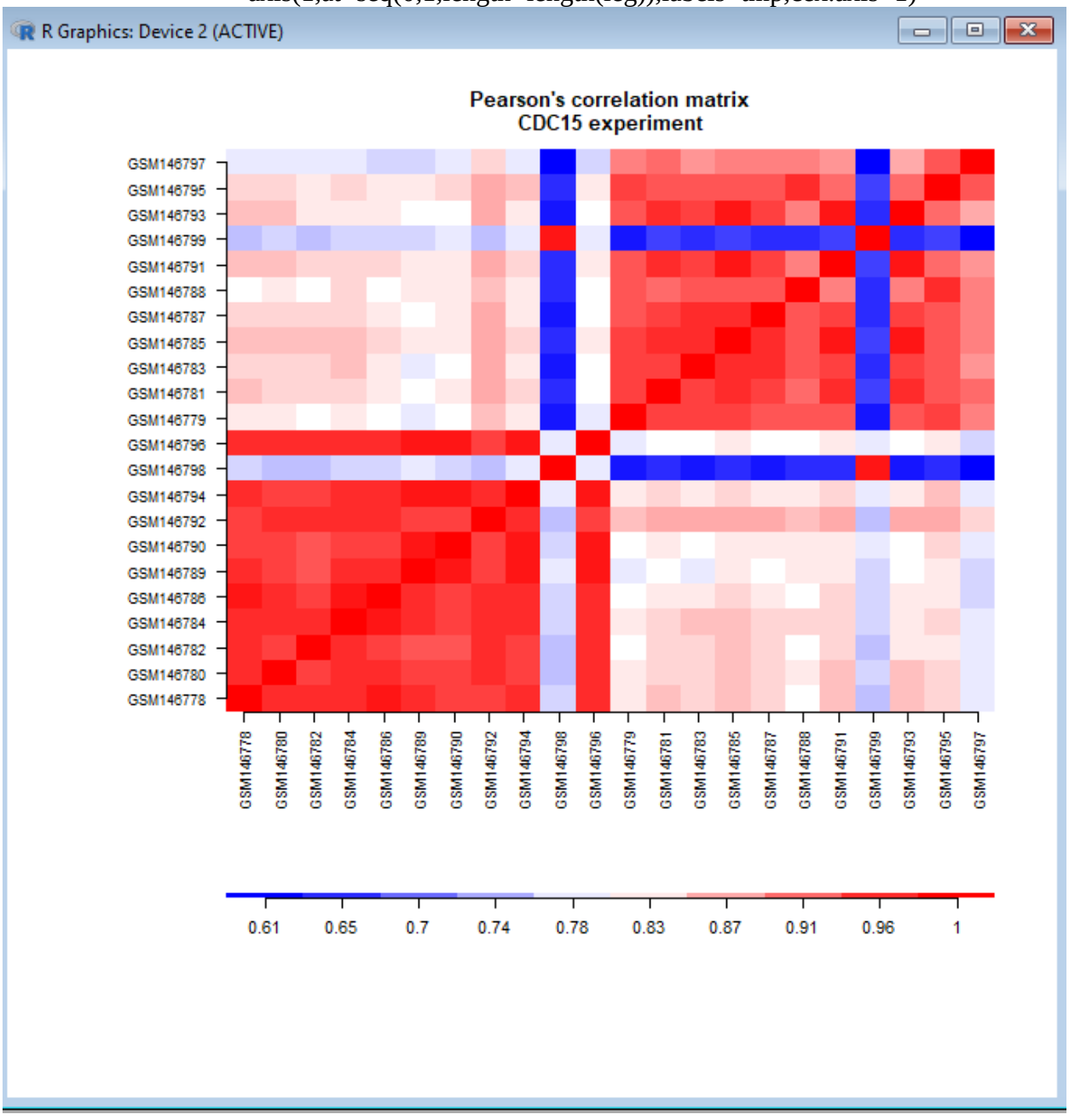
    i. Based on the plots/graphs, I would conclude that GSM146798(Normal) and GSM146799(tumor) are the Outliers in this dataset.

4.)

a. Correlation plot (heat map) (2pts.)

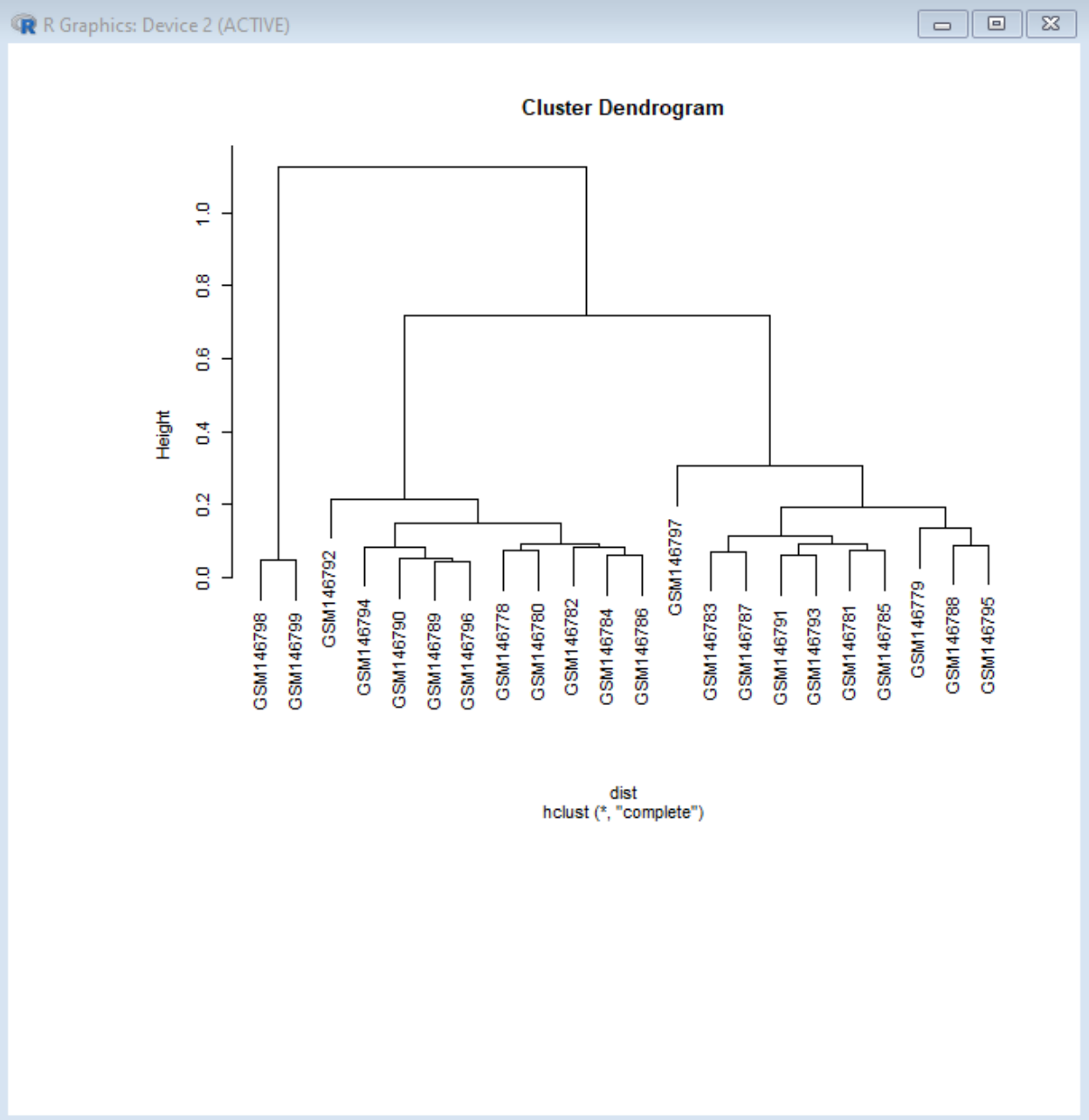
```
library(gplots)
dat.cor = cor(dat, use = 'pairwise.complete.obs')
layout(matrix(c(1,1,1,1,1,1,1,1,2,2), 5, 2, byrow = TRUE))
par(oma=c(5,7,1,1))
cx <- rev(colorpanel(25,"red","white","blue"))
leg <- seq(min(dat.cor,na.rm=T),max(dat.cor,na.rm=T),length=10)
```

```
image(dat.cor,main="Pearson's correlation matrix\nCDC15
experiment",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)
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)
```



b. Hierarchical clustering dendrogram(2pts.)

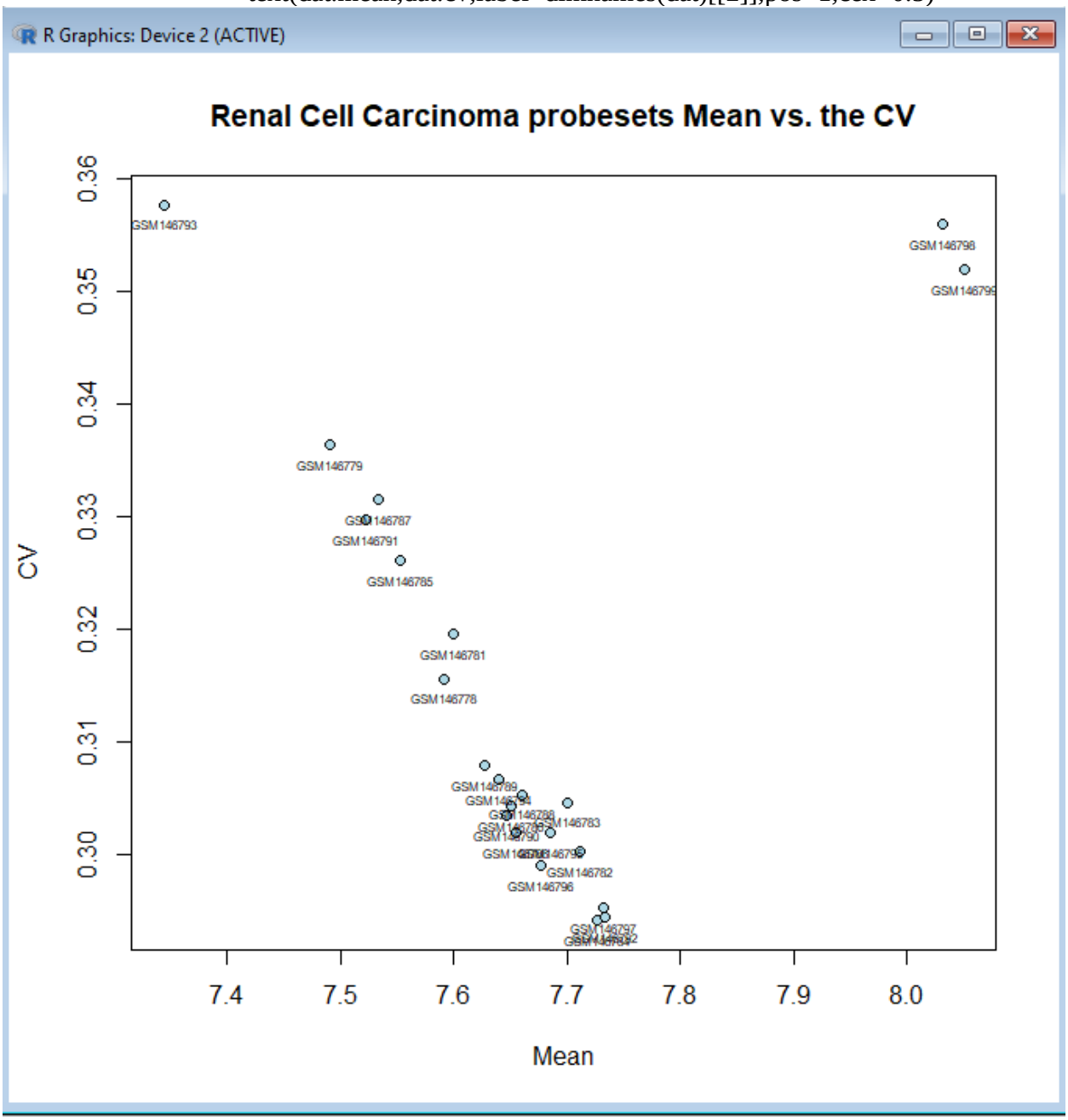
```
dist = dist(dat.cor, method = "euclidian", diag = TRUE)
hc = hclust(dist)
plot(hc)
```



c. CV vs. mean plot (2pts.)

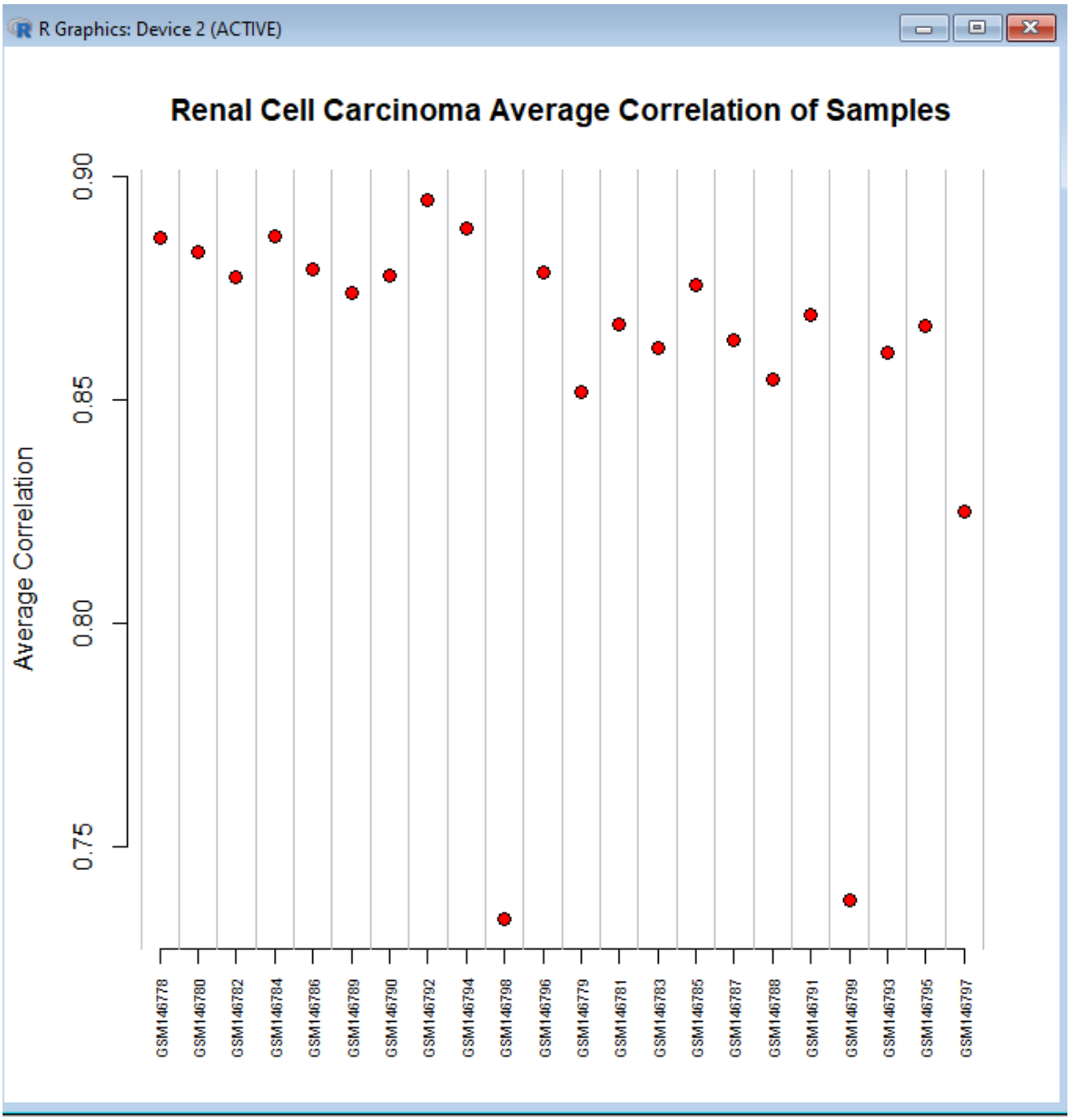
```
dat.mean = apply(log2(dat),2,mean)
dat.sd = sqrt(apply(log2(dat),2,var))
dat.cv = dat.sd/dat.mean
plot(dat.mean,dat.cv,main="Renal Cell Carcinoma probesets Mean vs. the
CV",xlab='Mean',ylab='CV')
points(dat.mean,dat.cv,bg="lightblue",col=1,pch=21)
```

text(dat.mean,dat.cv,label=dimnames(dat)[[2]],pos=1,cex=0.5)



d. Average correlation plot(2pts.)

```
dat.avg = apply(dat.cor,1,mean)
plot(c(1,length(dat.avg)),range(dat.avg),type="n",xlab="",ylab="Average
Correlation",main="Renal Cell Carcinoma Average Correlation of
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(dat)[[2]],las=2,cex.lab=0.
4,cex.axis=0.6)
axis(2)
abline(v=seq(0.5,62.5,1),col="grey")
```



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. (1pt.)

```
install.packages('VIM')
library(VIM)
```

7.) Remove the outlier samples you identified in the first part of this assignment. (2pts.)
   a. Need to remove GSM146798(Normal) and GSM146799(tumor) columns in the dataset.

```
dat[,'GSM146798'] = NULL
dat[,'GSM146799'] = NULL
dim(dat)
[1] 22283    20
```

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.** (6pts.)
   a. Probeset IDs KNG1: 206054_at AND 217512_at

```
kng1 = dat['206054_at',]
kng2 = dat['217512_at',]
```

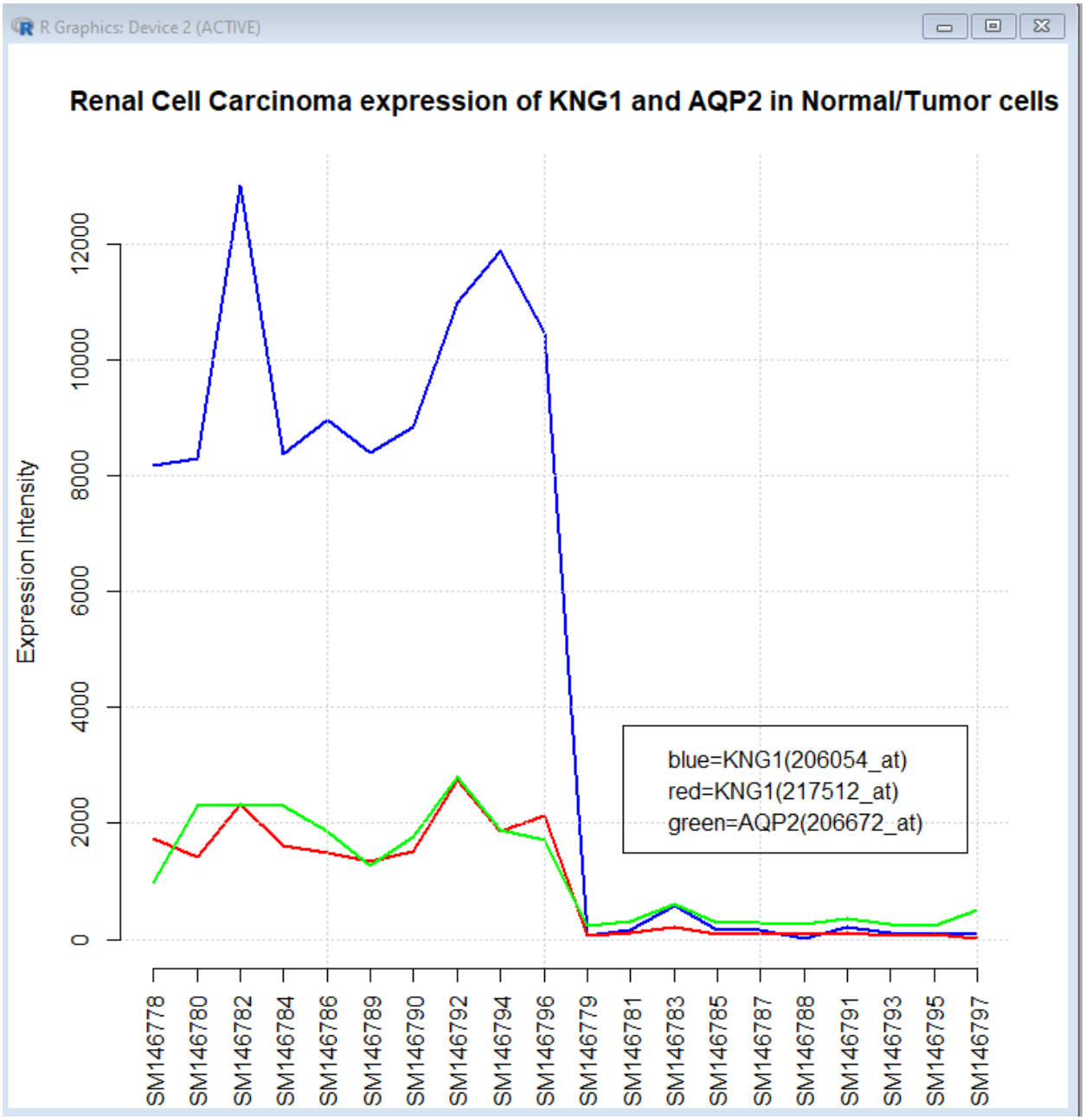
   b. Probeset ID AQP2: 206672_at

```
aqp2 = dat['206672_at',]
```

   c. Plotting the three probesets

```
plot(c(1,ncol(dat)),range(as.numeric(kng1)),type="n", xlab = ",ylab =
'Expression Intensity', main= 'Renal Cell Carcinoma expression of KNG1
and AQP2 in Normal/Tumor cells', axes=F)
lines(c(1:ncol(dat)),as.numeric(kng1), lwd=2,col='blue')
lines(c(1:ncol(dat)),as.numeric(kng2), lwd=2,col='red')
lines(c(1:ncol(dat)),as.numeric(aqp2), lwd=2,col='green')
grid()
axis(2)
axis(1,at=c(1:ncol(dat)),dimnames(dat)[[2]],las=2,cex.lab=0.3)
xmax = par("usr")[2]-1
ymin = 1500
legend(x=xmax, y=ymin, legend= strsplit(
paste("blue=KNG1(206054_at)", ",", "red=KNG1(217512_at)", ",",
"green=AQP2(206672_at)", sep="), "," )[[1]], xjust=1, yjust=0, cex=1.0)
```

   d. It seems as though the first KNG1 group from the probeset ID #206054_at has significantly different expression from the other KNG1 set in the group (#217512_at), which is around AQP2. These genes are all expressed at much lower levels in the tumor samples, all showing relatively the same expression intensity. KNG1 from probeset 217512_at and AQP2 seem to be showing normal renal function as they are similar in both normal and tumor, and the first KNG1 seems to not indicate normal renal 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**. <u>Be sure to first save the original value before replacing it with an NA</u>**. Also cast the data frame to a matrix to run this function. (2pts.)

```
dat['206054_at', 'GSM146784']
        [1] 8385.3
q9 = 8385.3
dat['206054_at', 'GSM146784'] = NA
```

```
dat.m = as.matrix(dat)
```

10.)Now estimate the missing values in the array using 6 nearest neighbors and Euclidean distance with the impute.knn() function. (2pts.)

```
dat.m1 = kNN(dat.m , variable = 'GSM146784', k = 6, metric =
euclidian)
dat.m1['206054_at', 'GSM146784']
[1] 6523.5
q10 = 6523.5
```

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. (2pts.)

Relative error: |(8385.3-6523.5)| / 8385.3 = 0.222 = 22.2% Relative error

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 retrive the output matrix, see the help file. (2pts.)

```
dat12 = pca(dat.m, nPcs = 9, completeObs = TRUE, svdImpute)
dat.12 = completeObs(dat12)
dat.12['206054_at', 'GSM146784']
[1] 9427.891
q12 = 9427.891
```

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. (6pts.)

```
line1 = dat['206054_at', ]
line2 = dat.m1['206054_at', ]
line3 = dat12['206054_at'. ]
plot(c(1,ncol(dat)),range(as.numeric(line1)),type='n',xlab='',ylab='
Expression Intensity', main='Imputation Evaluation for KNG1 in
Normal and Tumor Cells',axes=F)
lines(c(1:ncol(dat)),as.numeric(line2), lwd=2,col='red')
lines(c(1:ncol(dat)),as.numeric(line3), lwd=2,col='green')
lines(c(1:ncol(dat)),as.numeric(line1), lwd=2,col='blue')
axis(2)
axis(1,at=c(1:ncol(dat)),dimnames(dat)[[2]],las=2,cex.lab=0.3)
```

```
grid()
legend(x=xmax, y=ymin, legend= strsplit( paste("blue=Actual",
",", "red=KNN Impute", ",", "green=SVD Impute", sep="), ","
)[[1]], xjust=1, yjust=0, cex=1.0)
```