# Appendix 1

# Array metrics

>biocLite("simpleaffy")

>biocLite("affyQCReport")

>library(simpleaffy)

>qc.data<-qc(Data)

>slotNames(qc.data) # just to check it has worked, should get a list of variables beginning with scale.factors

>setwd("D:/Dropbox/PhD/Bioinformatics/Output data human")

>pdf(file="QCPlot\_all\_controls.pdf",height=8, width=12)#paper="a4r")

>plot(qc.data)

>dev.off()

>tiff(filename="qc\_humansod\_all\_controls.tiff",width=480,height=480,compression="none")

>plot(qc.data)

>dev.off()

>library(affyQCReport)

>QCReport(Data,file="affyQCReport.pdf")

# Raw chip images

>setwd("D:/Dropbox/PhD/Bioinformatics/Output data human")

>pdf(file="SOD astro raw images all con.pdf",height=4, width=4,paper="a4r")

>image((Data)[,1]) # change the number after the comma for the subsequent arrays

#Have a look at the chips on a per-chip level, nicer images than the raw chip images above

>library(affyPLM)

>pdf(file="all con SOD astro raw images colour.pdf",paper="a4r")

>celfiles.qc<-fitPLM(Data)

>image(celfiles.qc,which=1,add.legend=TRUE) # as above change number after “which=” for subsequent arrays

# Relative Log Expression plots should have values close to zero...

>pdf(file="all con SOD astro RLE plot.pdf",height=8, >width=12)#paper="a4r")

>RLE(celfiles.qc,main="Relative Log Expression")

>dev.off()

# Normalised Unscaled Standard Errors - median standard error should be 1 for most genes

>pdf(file="all con SOD astro NUSE plot.pdf",height=8, width=12)

>NUSE(celfiles.qc,main="Normalised Unscaled Standard Error")

dev.off()

Figure 1 Script used in Bioconductor for quality control analysis of mouse and human arrays. This script allows one to look at a QC report containing information on background, percentage genes present, actin and gapdh 3’-5’ ratios etc. It also allows one to look at raw images of the chips, and to produce graphs of the normalised unscaled standard error (NUSE) and the relative log expression (RLE).

>biocLite()

>biocLite("puma")

>library(puma)

>library(affy)

# extracting data from .CEL files

>file.names<-c("sodGD34\_92.CEL",

"sodGD49\_94.CEL",

"sodGD37\_04.CEL",

"confGD2JK4.CEL",

"confGD135.34.CEL",

"conGD20\_07.CEL",

"conGD2JK10.CEL")

>Data<-ReadAffy(filenames=file.names)

# Adding pheno data to the samples

>pData(Data)<-data.frame("Genotype"=c("SOD1","SOD1","SOD1",

"CTRL","CTRL","CTRL", "CTRL"), "SampleName"=c("34/92","49/94","37/04","JK4","135.34","20/07", "JK10"),row.names=rownames(pData(Data)))

>show(Data)

>pData(Data)

>eset\_cel\_mmgmos<-mmgmos(Data)

###################### Filter ############################

#Filter by present call (Mas5)

>biocLite("genefilter")

>library(genefilter)

>eset.mas5<-mas5calls(Data)

>mascallsfilter<-function(cutoff="A",number){

function(x){

sum((x)==(cutoff))!=number

}

}

>m5cfun<-mascallsfilter(number=dim(eset.mas5)[2])

>mcfun<-filterfun(m5cfun)

>mcsub<-genefilter(eset.mas5,mcfun)

>sum(mcsub)# genes left 35298

>X<-eset\_cel\_mmgmos[mcsub,] # filter these genes from current data set

>dim(X)

#### Identifying differentially expressed genes with the IPPLR method##########

>eset\_cel\_IPPLR<-pumaCombImproved(X)

>eset\_cel\_DE<-pumaDE(eset\_cel\_IPPLR)

>colnames(createContrastMatrix(eset\_cel\_IPPLR))

#Output data as p value table and FC value table

>t\_pv<-(pLikeValues(eset\_cel\_DE,contrast=1,direction="either"))

>a<-(length(which(t\_pv<=0.05)))

>t\_pv\_1<-sort(t\_pv)

>X\_disease\_control<-t\_pv\_1[1:a]

>write.table(X\_disease\_control,file="AstroPUMApvalues\_filt\_male\_v\_female.txt")

>t\_fc<-(FC(eset\_cel\_DE))

>t\_fc<-t\_fc[,1]

>test<-names(t\_fc)

>genes1<-names(X\_disease\_control)

>list<-which(test %in% genes1)

>X\_disease\_control\_FC<-(t\_fc[list])

>write.table(X\_disease\_control\_FC, >file="AstroPUMA\_FC\_filt\_male\_v\_female.txt")

################### Add gene names and symbols ##################

# Go on to David and use the ID conversion tools to get annotation data.

# Then import all tables back into R

>annot<-read.csv("annots.csv")

>FC<-read.csv("AstroPUMA\_FC\_filt\_male\_v\_female.csv")

>p<-read.csv("AstroPUMApvalues\_filt\_male\_v\_female.csv")

>combo <- merge(annot,FC,by="ID")

>combo2<-merge(combo,p,by="ID")

>write.csv(combo2,file="PUMA\_humanSOD\_male\_v\_female\_astro\_filt\_annotated.csv")

>pvalue<-p[,2]

>pdf(file="hist\_p\_value\_PUMAFIVEcon(3).pdf",paper="a4r")

>hist(pvalue, main="Histogram of p-values (FIVE-con)",xlab="p-value")

>dev.off()

############## Gender confound elimination ######################

>gender<read.csv("PUMA\_humanSOD\_male\_v\_female\_astro\_filt\_annotated.csv")

>goodcon<-read.csv("PUMA\_humanSOD\_v\_GOODcon\_astro\_filt\_annotated.csv")

>genconfounds<-which(goodcon[,1]%in%gender[,1])

>genelim<-goodcon[-c(genconfounds),]

>write.csv(genelim,file="PUMA\_humanSOD\_GOODCON\_FINAL\_gender\_elim.csv")

Figure Script used for analysis of human gene chips in R and bioconductor. The script starts with the input of the names for each chip and the disease status. This is then followed by a normalisation step using the MMGMOS algorithm from the puma package. A filter is then applied which removes probesets if they are called absent using the MAS5 algorithm for all genechips. Differential gene expression is then identified and results tables are produced, which are then combined and annotated with gene names and symbols using the DAVID online database. Finally, a removal of genes that are called significant due to the differing gender composition of the two groups was performed by subtracting those genes called significantly different in a male vs female analysis from the SOD1 vs control dataset.