AprilW3

Awais Choudhry

23/04/2017

20-4-2017

This code intends to clean up the chipSeq Data i downloaded from NCBI GEo as an expression set. I have called the Arx peaks.

## trying to import the chipseq Data  
  
library(Biobase)  
library(GEOquery)  
library(limma)  
library(magrittr)  
  
# load series and platform data from GEO  
gset <- getGEO("GSE29985", GSEMatrix =TRUE, AnnotGPL=TRUE)[[1]]  
chipDataSet<-cbind(as.data.frame(gset@featureData@data$`Chromosome annotation`), exprs(gset))  
colnames(chipDataSet)<- c("probe", "n2a1", "n2a2", "n2a3", "embyrobrain1", "embyrobrain2", "embyrobrain3")  
chipFilterNa<-chipDataSet[!chipDataSet$probe=="",]  
splitColoumnComma<-data.frame(do.call('rbind', strsplit(as.character(chipDataSet$probe),',')))  
splitColoumnSlash<-data.frame(do.call('rbind', strsplit(as.character(splitColoumnComma$X1),'///')))  
  
  
spiltColumnSpace<-data.frame(do.call('rbind', strsplit(as.character(splitColoumnComma$X2),' ',fixed=TRUE)))  
spiltColumnDotDot<-data.frame(do.call('rbind', strsplit(as.character(spiltColumnSpace$X3),'..',fixed=TRUE)))  
  
spiltColumnOpener<-data.frame(do.call('rbind', strsplit(as.character(spiltColumnDotDot$X1),'(',fixed=TRUE)))  
spiltColumnCloser<-data.frame(do.call('rbind', strsplit(as.character(spiltColumnDotDot$X2),')',fixed=TRUE)))  
  
probeAnnotationDataFrame<-cbind(splitColoumnSlash[1],spiltColumnOpener[2], spiltColumnCloser[1], spiltColumnSpace[2])  
colnames(probeAnnotationDataFrame)<- c("chromosome","start", "end", "probeID")  
dataFrameGrange<-cbind(probeAnnotationDataFrame, chipFilterNa[2:7])  
dataFrameGrange$start <- as.character(dataFrameGrange$start)%>%as.numeric()  
dataFrameGrange$end <- as.character(dataFrameGrange$end)%>%as.numeric()  
  
##The CHIPSEQDATA in a dataframe  
chiPSeqRawDataFrame<-makeGRangesFromDataFrame(na.omit(dataFrameGrange),  
 keep.extra.columns=TRUE,  
 ignore.strand=TRUE,  
 seqinfo=NULL,  
 seqnames.field=c("seqnames", "seqname",  
 "chromosome", "chrom",  
 "chr", "chromosome\_name",  
 "seqid",1),  
 start.field="start",  
 end.field=c("end", "stop"),  
 starts.in.df.are.0based=FALSE)  
  
  
### Checking the variability between samples!  
grangeRawN2a1<- chiPSeqRawDataFrame[,2]  
grangeRawN2a2<-chiPSeqRawDataFrame[,3]  
grangeRawN2a3<-chiPSeqRawDataFrame[,4]  
  
grangeRawBrain1<-chiPSeqRawDataFrame[,5]  
grangeRawBrain2<-chiPSeqRawDataFrame[,6]  
grangeRawBrain3<-chiPSeqRawDataFrame[,7]  
  
library(Gviz)  
library(magrittr)  
grangeRawN2a1Track<-grangeRawN2a1%>%DataTrack(type="l", name="N2a Sample 1")  
grangeRawN2a2Track<-grangeRawN2a2%>%DataTrack(type="l", name="N2a Sample 2")  
grangeRawN2a3Track<-grangeRawN2a3%>%DataTrack(type="l", name="N2a Sample 3")  
  
grangeRawBrain1Track<-grangeRawBrain1%>%DataTrack(type="l", name="Embyronic Brain 1")  
grangeRawBrain2Track<-grangeRawBrain2%>%DataTrack(type="l", name="Embyronic Brain 2")  
grangeRawBrain3Track<-grangeRawBrain3%>%DataTrack(type="l", name="Embyronic Brain 3")  
  
  
  
dataFrameN2aAverage<-cbind(dataFrameGrange[1:3], rowMeans(dataFrameGrange[5:7], na.rm = TRUE))  
DataFrameBrainAverage<-cbind(dataFrameGrange[1:3], rowMeans(dataFrameGrange[8:10], na.rm = TRUE))  
  
grangeAverageN2aRaw<-makeGRangesFromDataFrame(na.omit(dataFrameN2aAverage),  
 keep.extra.columns=TRUE,  
 ignore.strand=TRUE,  
 seqinfo=NULL,  
 seqnames.field=c("seqnames", "seqname",  
 "chromosome", "chrom",  
 "chr", "chromosome\_name",  
 "seqid",1),  
 start.field="start",  
 end.field=c("end", "stop"),  
 starts.in.df.are.0based=FALSE)  
  
  
grangeAverageBrainRaw<-makeGRangesFromDataFrame(na.omit(DataFrameBrainAverage),  
 keep.extra.columns=TRUE,  
 ignore.strand=TRUE,  
 seqinfo=NULL,  
 seqnames.field=c("seqnames", "seqname",  
 "chromosome", "chrom",  
 "chr", "chromosome\_name",  
 "seqid",1),  
 start.field="start",  
 end.field=c("end", "stop"),  
 starts.in.df.are.0based=FALSE)  
  
  
  
##Making the Averages into Data Tracks  
averageBrainRawTrack<-grangeAverageBrainRaw%>%DataTrack(type="histogram", name = "Average Brain")  
averageN2aRawTrack<-grangeAverageN2aRaw%>%DataTrack(type="histogram", name="Average N2a")  
  
plotTracks(c(averageN2aRawTrack, averageBrainRawTrack))  
  
##Checking to see where the plot falls.   
  
findOverlaps(grangeAverageBrainRaw, grangeBrainChipSeq)  
  
  
seqlevels(grangeBrainChipSeq)

The code really splits up the annotation data in the corresponding columns, hence allowing me to generate a grange object which i can then use to compare to the datasets. Unfortunately the seqlevels are incompantiable sytles hence i need to convert to the UCSC style. To do this i had to write a for loop that rewrites the "Chromosome 1" as "chr1". I can then plots these as tracks.

This code below generates perfectly matched Arx Motifs and produces the tables similar to those written regarding spacing and distance of Arx motifs. These however are done at 100% to see if Arx motif spacing does hold up at perfect spacing.

## Checking to see if the numbers are robust  
  
library(magrittr)  
library(GenomicRanges)  
library(ggplot2)  
library(magrittr)  
library(tibble)  
library(pander)  
library(reshape2)  
library(plyr)  
library(MotifDb)  
library(BSgenome.Mmusculus.UCSC.mm9)  
library(magrittr)  
library(reshape2)  
  
enhancerGrange <-  
 import(con = "~/DataFiles/Enhancer Tracks/Mouse/Enhanceresmm9.bed")  
UCSCgenes <- import("~/Scripts/March/mm9.bed")  
promoters <- promoters(UCSCgenes)  
  
ArxPlaindrmicMinus1<-rbind( A=c(0,1,1,0,0,1,1,0,0,1),   
 C=c(0,0,0,0,0,0,0,0,0,0),  
 G=c(0,0,0,0,0,0,0,0,0,0),  
 T=c(1,0,0,1,1,0,0,1,1,0))  
  
arx6MerPWMNospace<-rbind( A=c(0,1,1,0,0,1,0,1,1,0,0,1),   
 C=c(0,0,0,0,0,0,0),  
 G=c(0,0,0,0,0,0,0) ,  
 T=c(1,0,0,1,1,0,1,0,0,1,1,0))  
  
arx6MerPWM1space<-rbind( A=c(0,1,1,0,0,1,0.25,1,0,0,1,1,0),   
 C=c(0,0,0,0,0,0,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0),  
 T=c(1,0,0,1,1,0,0.25,0,1,1,0,0,1))  
  
arx6MerPWM2space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,1,0,0,1,1,0),   
 C=c(0,0,0,0,0,0,0.25,0.25),  
 G=c(0,0,0,0,0,0,0.25,0.25,0) ,  
 T=c(1,0,0,1,1,0,0.25,0.25,0,1,1,0,0,1))  
  
arx6MerPWM3space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,1,0,0,1,1,0),  
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0),  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,0,1,1,0,0,1))  
  
arx6MerPWM4space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,0.25,1,0,0,1,1,0),  
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25) ,  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,0.25,0,1,1,0,0,1))  
  
arx6MerPWM5space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,0.25,0.25,1,0,0,1,1,0),  
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0),  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,0.25,0.25,0,1,1,0,0,1))  
   
arx6MerPWM6space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,0.25,0.25,0.25,1,0,0,1,1,0),  
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0.25,0),  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,0.25,0.25,0.25,0,1,1,0,0,1))  
  
arx6MerPWM7space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,0.25,0.25,0.25,0.25,1,0,0,1,1,0),  
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0) ,  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0,1,1,0,0,1))  
  
### Tandeom Sites  
arxJolma<-rbind( A=c(0,1,1,0,0,0.25,1,1,0,0,1),   
 C=c(0,0,0,0,0,0.25,0,0,0,0,0),  
 G=c(0,0,0,0,0,0.25,0,0,0,0,0),  
 T=c(1,0,0,1,1,0.25,0,0,1,1,0))  
arxTandemNoSpace<-rbind( A=c(0,1,1,0,0,1,0,1,1,0,0,1),  
 C=c(0,0,0,0,0,0,0,0,0,0,0,0),  
 G=c(0,0,0,0,0,0,0,0,0,0,0,0) ,  
 T=c(1,0,0,1,1,0,1,0,0,1,1,0))  
  
arxTandem1Space<-rbind( A=c(0,1,1,0,0,1,0.25,0,1,1,0,0,1),  
 C=c(0,0,0,0,0,0,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0) ,  
 T=c(1,0,0,1,1,0,0.25,1,0,0,1,1,0))  
  
arxTandem2Space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0,1,1,0,0,1),  
 C=c(0,0,0,0,0,0,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0) ,  
 T=c(1,0,0,1,1,0,0.25,0.25,1,0,0,1,1,0))  
  
arxTandem3Space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,0,1,1,0,0,1),  
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0) ,  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,1,0,0,1,1,0))  
arxTandem4Space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,0.25,0,1,1,0,0,1),  
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0) ,  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,0.25,1,0,0,1,1,0))  
  
arxTandem5Space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,0.25,0.25,0,1,1,0,0,1),   
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0) ,  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,0.25,0.25,1,0,0,1,1,0))  
arxTandem6Space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,0.25,0.25,0.25,0,1,1,0,0,1),   
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0.25,0) ,  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,0.25,0.25,0.25,1,0,0,1,1,0))  
arxTandem7Space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0,1,1,0,0,1),   
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0.25,0.25,0.25,0) ,  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,0.25,0.25,0.25,0.25,1,0,0,1,1,0))  
arxTandem8Space<-rbind( A=c(0,1,1,0,0,1,0.25,0.25,0.25,0.25,0,1,1,0,0,1),   
 C=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0),  
 G=c(0,0,0,0,0,0,0.25,0.25,0.25,0.25,0) ,  
 T=c(1,0,0,1,1,0,0.25,0.25,0.25,0.25,1,0,0,1,1,0))  
##requires code from the 16-4-2017 to run  
grangeJolmaMinus<-  
 matchPWM(arxJolma, BSgenome.Mmusculus.UCSC.mm9, "100%")  
  
grangeplaindromicNospace <-  
 matchPWM(arx6MerPWMNospace, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeplaindromic1space <-  
 matchPWM(arx6MerPWM1space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeplaindromic2space <-  
 matchPWM(arx6MerPWM2space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeplaindromic3space <-  
 matchPWM(arx6MerPWM3space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeplaindromic4space <-  
 matchPWM(arx6MerPWM4space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeplaindromic5space <-  
 matchPWM(arx6MerPWM5space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeplaindromic6space <-  
 matchPWM(arx6MerPWM6space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeplaindromic7space <-  
 matchPWM(arx6MerPWM7space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
  
grangeMinusOne <-  
 matchPWM(ArxPlaindrmicMinus1, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeTandemNoSpace<-  
 matchPWM(arxTandemNoSpace, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeTandem1space <-  
 matchPWM(arxTandem1Space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeTandem2space <-  
 matchPWM(arxTandem2Space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeTandem3space <-  
 matchPWM(arxTandem3Space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeTandem4space <-  
 matchPWM(arxTandem4Space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeTandem5space <-  
 matchPWM(arxTandem5Space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeTandem6space <-  
 matchPWM(arxTandem6Space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grangeTandem7space <-  
 matchPWM(arxTandem7Space, BSgenome.Mmusculus.UCSC.mm9, "100%")  
  
#grangeplaindromic1space<-matchPWM(arx6MerPWM1space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeplaindromic2space<-matchPWM(arx6MerPWM2space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeplaindromic3space<-matchPWM(arx6MerPWM3space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeplaindromic4space<-matchPWM(arx6MerPWM4space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeplaindromic5space<-matchPWM(arx6MerPWM4space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeplaindromic6space<-matchPWM(arx6MerPWM4space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeplaindromic7space<-matchPWM(arx6MerPWM4space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeTandem1space<-matchPWM(arxTandem1Space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeTandem2space<-matchPWM(arxTandem2Space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeTandem3space<-matchPWM(arxTandem3Space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeTandem4space<-matchPWM(arxTandem4Space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeTandem5space<-matchPWM(arxTandem5Space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeTandem6space<-matchPWM(arxTandem6Space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
#grangeTandem7space<-matchPWM(arxTandem7Space, BSgenome.Mmusculus.UCSC.mm9, "90%")  
tandemDataTable <- rbind(  
 cbind(  
 length(grangeJolmaMinus),  
 sum(countOverlaps(grangeJolmaMinus, UCSCgenes)),  
 sum(countOverlaps(grangeJolmaMinus, promoters)),  
 sum(countOverlaps(grangeJolmaMinus, enhancerGrange)),  
 (length(grangeJolmaMinus)-sum(countOverlaps(grangeJolmaMinus, enhancerGrange))-  
 sum(countOverlaps(grangeJolmaMinus, promoters))- sum(countOverlaps(grangeJolmaMinus, UCSCgenes)))  
),  
 cbind(  
 numberofTandem <- length(grangeMinusOne),  
 dataTableNoGenesminus1 <-  
 sum(countOverlaps(grangeMinusOne, UCSCgenes)),  
 dataTableMinus1 <-  
 sum(countOverlaps(grangeMinusOne, promoters)),  
 dataTableMinus1r <-  
 sum(countOverlaps(grangeMinusOne, enhancerGrange)),  
 (length(grangeMinusOne)-sum(countOverlaps(grangeMinusOne, enhancerGrange))-  
 sum(countOverlaps(grangeMinusOne, promoters))- sum(countOverlaps(grangeMinusOne, UCSCgenes)))  
 ),  
 cbind(  
 numberofTandemNoSpaceSites <- length(grangeTandemNoSpace),  
 dataTableNoGenes <-  
 sum(countOverlaps(grangeTandemNoSpace, UCSCgenes)),  
 dataTableNoSpacePromoters <-  
 sum(countOverlaps(grangeTandemNoSpace, promoters)),  
 dataTableNoSpaceEnhancer <-  
 sum(countOverlaps(grangeTandemNoSpace, enhancerGrange)),  
 (length(grangeTandemNoSpace)-sum(countOverlaps(grangeTandemNoSpace, enhancerGrange))-  
 sum(countOverlaps(grangeTandemNoSpace, promoters))- sum(countOverlaps(grangeTandemNoSpace, UCSCgenes)))  
 ),  
 cbind(  
 numberofTandem1spaceSites <- length(grangeTandem1space),  
 dataTable1SpaceGenes <-  
 sum(countOverlaps(grangeTandem1space, UCSCgenes)),  
 dataTable1SpacePromoters <-  
 sum(countOverlaps(grangeTandem1space, promoters)),  
 dataTable1SpaceEnhancer <-  
 sum(countOverlaps(grangeTandem1space, enhancerGrange)),  
 (length(grangeTandem1space)-sum(countOverlaps(grangeTandem1space, enhancerGrange))-  
 sum(countOverlaps(grangeTandem1space, promoters))- sum(countOverlaps(grangeTandem1space, UCSCgenes)))  
 ),  
 cbind(  
 numberofTandem2spaceSites <- length(grangeTandem2space),  
 dataTable2SpaceGenes <-  
 sum(countOverlaps(grangeTandem2space, UCSCgenes)),  
 dataTable2SpacePromoters <-  
 sum(countOverlaps(grangeTandem2space, promoters)),  
 dataTable2SpaceEnhancer <-  
 sum(countOverlaps(grangeTandem2space, enhancerGrange)),  
 (length(grangeTandem2space)-sum(countOverlaps(grangeTandem2space, enhancerGrange))-  
 sum(countOverlaps(grangeTandem2space, promoters))- sum(countOverlaps(grangeTandem2space, UCSCgenes)))  
 ),  
 cbind(  
 numberofTandem3spaceSites <- length(grangeTandem3space),  
 dataTable3SpaceGenes <-  
 sum(countOverlaps(grangeTandem3space, UCSCgenes)),  
 dataTable3SpacePromoters <-  
 sum(countOverlaps(grangeTandem3space, promoters)),  
 dataTable3SpaceEnhancer <-  
 sum(countOverlaps(grangeTandem3space, enhancerGrange)),  
 (length(grangeTandem3space)-sum(countOverlaps(grangeTandem3space, enhancerGrange))-  
 sum(countOverlaps(grangeTandem3space, promoters))- sum(countOverlaps(grangeTandem3space, UCSCgenes)))  
 ),  
 cbind(  
 numberofTandem4spaceSites <- length(grangeTandem4space),  
 dataTable4SpaceGenes <-  
 sum(countOverlaps(grangeTandem4space, UCSCgenes)),  
 dataTable4SpacePromoters <-  
 sum(countOverlaps(grangeTandem4space, promoters)),  
 dataTable4SpaceEnhancer <-  
 sum(countOverlaps(grangeTandem4space, enhancerGrange)),  
 (length(grangeTandem4space)-sum(countOverlaps(grangeTandem4space, enhancerGrange))-  
 sum(countOverlaps(grangeTandem4space, promoters))- sum(countOverlaps(grangeTandem4space, UCSCgenes)))  
 ),   
cbind(  
 numberofTandem5spaceSites <- length(grangeTandem5space),  
 dataTable5SpaceGenes <-  
 sum(countOverlaps(grangeTandem5space, UCSCgenes)),  
 dataTable5SpacePromoters <-  
 sum(countOverlaps(grangeTandem5space, promoters)),  
 dataTable5SpaceEnhancer <-  
 sum(countOverlaps(grangeTandem5space, enhancerGrange)),  
 (length(grangeTandem5space)-sum(countOverlaps(grangeTandem5space, enhancerGrange))-  
 sum(countOverlaps(grangeTandem5space, promoters))- sum(countOverlaps(grangeTandem5space, UCSCgenes)))  
 ),  
 cbind(  
 numberofTandem6spaceSites <- length(grangeTandem6space),  
 dataTable6SpaceGenes <-  
 sum(countOverlaps(grangeTandem6space, UCSCgenes)),  
 dataTable6SpacePromoters <-  
 sum(countOverlaps(grangeTandem6space, promoters)),  
 dataTable6SpaceEnhancer <-  
 sum(countOverlaps(grangeTandem6space, enhancerGrange)),  
 (length(grangeTandem6space)-sum(countOverlaps(grangeTandem6space, enhancerGrange))-  
 sum(countOverlaps(grangeTandem6space, promoters))- sum(countOverlaps(grangeTandem6space, UCSCgenes)))  
 ),  
 cbind(  
 numberofTandem7spaceSites <- length(grangeTandem7space),  
 dataTable7SpaceGenes <-  
 sum(countOverlaps(grangeTandem7space, UCSCgenes)),  
 dataTable7SpacePromoters <-  
 sum(countOverlaps(grangeTandem7space, promoters)),  
 dataTable7SpaceEnhancer <-  
 sum(countOverlaps(grangeTandem7space, enhancerGrange)),  
 (length(grangeTandem7space)-sum(countOverlaps(grangeTandem7space, enhancerGrange))-  
 sum(countOverlaps(grangeTandem7space, promoters))- sum(countOverlaps(grangeTandem7space, UCSCgenes)))  
 )  
) %>% as.data.frame  
  
colnames(tandemDataTable) <- c("Total",  
 "Motifs in genes",  
 "Motifs in promoters",  
 "Motifs in enhancers",  
 "Other")  
  
  
rownames(tandemDataTable) <- c("Arx Jolma",  
 "Minus one",  
 "No Space",  
 "1 Space",  
 "2 Space",  
 "3 Space",  
 "4 Space",  
 "5 Space",  
 "6 Space",  
 "7 Space")  
tandemDataTable %>% pander()  
  
tandemDataTable <- rownames\_to\_column(tandemDataTable)  
reshapedTandemDataTable<-reshape(tandemDataTable,  
 varying = c( "Motifs in promoters", "Motifs in enhancers", "Other", "Motifs in genes"),  
 v.names = "Numbers of Motif",  
 timevar = "Location",  
 times = c( "Promoters", "Enhancers", "Other","Genes" ),  
 direction = "long")  
ggplot(reshapedTandemDataTable, aes(x = rowname, y = `Numbers of Motif`, fill = `Location`)) +  
 geom\_bar(stat = "identity") +  
 xlab(label= "Number of Nucleotides Between Motifs")+  
 ylab(label= "NUmber of Arx Motifs")+  
 guides(fill=guide\_legend(title="Genomic Location"))+  
 theme\_bw()+  
 theme(axis.text=element\_text(size=12),  
 axis.title=element\_text(size=14,face="bold"))+  
 scale\_color\_manual(values=c(`Enhancer`="#999999", `Genes`="#E69F00", `Non-coding`="#56B4E9", `Promoters`= "#56B4E9"))  
  
  
  
  
  
planindromicDataTable <- rbind(  
 cbind(  
 length(grangeJolmaMinus),  
 sum(countOverlaps(grangeJolmaMinus, UCSCgenes)),  
 sum(countOverlaps(grangeJolmaMinus, promoters)),  
 sum(countOverlaps(grangeJolmaMinus, enhancerGrange)),  
 (length(grangeJolmaMinus)-sum(countOverlaps(grangeJolmaMinus, enhancerGrange))-  
 sum(countOverlaps(grangeJolmaMinus, promoters))- sum(countOverlaps(grangeJolmaMinus, UCSCgenes)))  
 ),  
 cbind(  
 length(grangeMinusOne),  
 sum(countOverlaps(grangeMinusOne, UCSCgenes)),  
 sum(countOverlaps(grangeMinusOne, promoters)),  
 sum(countOverlaps(grangeMinusOne, enhancerGrange)),  
 (length(grangeMinusOne)-sum(countOverlaps(grangeMinusOne, enhancerGrange))-  
 sum(countOverlaps(grangeMinusOne, promoters))- sum(countOverlaps(grangeMinusOne, UCSCgenes)))  
 ),  
 cbind(  
 length(grangeplaindromicNospace),  
 sum(countOverlaps(grangeplaindromicNospace, UCSCgenes)),  
 sum(countOverlaps(grangeplaindromicNospace, promoters)),  
 sum(countOverlaps(grangeplaindromicNospace, enhancerGrange)),  
 (length(grangeplaindromicNospace)-sum(countOverlaps(grangeplaindromicNospace, enhancerGrange))-  
 sum(countOverlaps(grangeplaindromicNospace, promoters))- sum(countOverlaps(grangeplaindromicNospace, UCSCgenes)))  
 ),  
 cbind(  
 length(grangeplaindromic1space),  
 Arx6mer <- sum(countOverlaps(grangeplaindromic1space, UCSCgenes)),  
 sum(countOverlaps(grangeplaindromic1space, promoters)),  
 sum(countOverlaps(grangeplaindromic1space, enhancerGrange)),  
 (length(grangeplaindromic1space)-sum(countOverlaps(grangeplaindromic1space, enhancerGrange))-  
 sum(countOverlaps(grangeplaindromic1space, promoters))- sum(countOverlaps(grangeplaindromic1space, UCSCgenes)))  
 ),  
 cbind(  
 length(grangeplaindromic2space),  
 sum(countOverlaps(grangeplaindromic2space, UCSCgenes)),  
 sum(countOverlaps(grangeplaindromic2space, promoters)),  
 sum(countOverlaps(grangeplaindromic2space, enhancerGrange)),  
 (length(grangeplaindromic2space)-sum(countOverlaps(grangeplaindromic2space, enhancerGrange))-  
 sum(countOverlaps(grangeplaindromic2space, promoters))- sum(countOverlaps(grangeplaindromic2space, UCSCgenes)))  
 )  
 ,  
 cbind(  
 numberOfArxSitesPlaindromic3Space <- length(grangeplaindromic3space),  
 sum(countOverlaps(grangeplaindromic3space, UCSCgenes)),  
 sum(countOverlaps(grangeplaindromic3space, promoters)),  
 sum(countOverlaps(grangeplaindromic4space, enhancerGrange)),  
 (length(grangeplaindromic3space)-sum(countOverlaps(grangeplaindromic3space, enhancerGrange))-  
 sum(countOverlaps(grangeplaindromic3space, promoters))- sum(countOverlaps(grangeplaindromic3space, UCSCgenes)))  
 ),  
 cbind(  
 numberOfArxSitesPlaindromic4Space <- length(grangeplaindromic4space),  
 sum(countOverlaps(grangeplaindromic4space, UCSCgenes)),  
 sum(countOverlaps(grangeplaindromic4space, promoters)),  
 sum(countOverlaps(grangeplaindromic4space, enhancerGrange)),  
 (length(grangeplaindromic4space)-sum(countOverlaps(grangeplaindromic4space, enhancerGrange))-  
 sum(countOverlaps(grangeplaindromic4space, promoters))- sum(countOverlaps(grangeplaindromic4space, UCSCgenes)))  
 ),  
 cbind(  
 numberOfArxSitesPlaindromic5Space <- length(grangeplaindromic5space),  
 sum(countOverlaps(grangeplaindromic5space, UCSCgenes)),  
 sum(countOverlaps(grangeplaindromic5space, promoters)),  
 sum(countOverlaps(grangeplaindromic5space, enhancerGrange)),  
 (length(grangeplaindromic5space)-sum(countOverlaps(grangeplaindromic5space, enhancerGrange))-  
 sum(countOverlaps(grangeplaindromic5space, promoters))- sum(countOverlaps(grangeplaindromic5space, UCSCgenes)))  
 ),  
 cbind(  
 numberOfArxSitesPlaindromic6Space <- length(grangeplaindromic6space),  
 sum(countOverlaps(grangeplaindromic6space, UCSCgenes)),  
 sum(countOverlaps(grangeplaindromic6space, promoters)),  
 sum(countOverlaps(grangeplaindromic6space, enhancerGrange)),  
 (length(grangeplaindromic6space)-sum(countOverlaps(grangeplaindromic6space, enhancerGrange))-  
 sum(countOverlaps(grangeplaindromic6space, promoters))- sum(countOverlaps(grangeplaindromic6space, UCSCgenes)))  
 ),  
 cbind(  
 numberOfArxSitesPlaindromic7Space <- length(grangeplaindromic7space),  
 sum(countOverlaps(grangeplaindromic7space, UCSCgenes)),  
 sum(countOverlaps(grangeplaindromic7space, promoters)),  
 sum(countOverlaps(grangeplaindromic7space, enhancerGrange)),  
 (length(grangeplaindromic7space)-sum(countOverlaps(grangeplaindromic7space, enhancerGrange))-  
 sum(countOverlaps(grangeplaindromic7space, promoters))- sum(countOverlaps(grangeplaindromic7space, UCSCgenes)))  
 )  
) %>% as.data.frame()  
colnames(planindromicDataTable) <- c("Total",  
 "Motifs in genes",  
 "Motifs in Promoters",  
 "Motifs in Enhancers",  
 "Other")  
rownames(planindromicDataTable) <-c("Arx Jolma",  
 "Minus one",  
 "No Space",  
 "1 Space",  
 "2 Space",  
 "3 Space",  
 "4 Space",  
 "5 Space",  
 "6 Space",  
 "7 Space")  
  
  
planindromicDataTable %>% pander()  
planindromicDataTable<- rownames\_to\_column(planindromicDataTable)  
  
reshapedPlaindromicDataTable<-reshape(planindromicDataTable,  
 varying = c( "Motifs in Promoters", "Motifs in Enhancers", "Other", "Motifs in genes"),  
 v.names = "Numbers of Motif",  
 timevar = "Location",  
 times = c( "Promoters", "Enhancers", "Other","Genes" ),  
 direction = "long")  
ggplot(reshapedPlaindromicDataTable, aes(x = rowname, y = `Numbers of Motif`, fill = `Location`)) +  
 geom\_bar(stat = "identity") +  
 xlab(label= "Number of Nucleotides Between Motifs")+  
 ylab(label= "NUmber of Arx Motifs")+  
 guides(fill=guide\_legend(title="Genomic Location"))+  
 theme\_bw()+  
 theme(axis.text=element\_text(size=12),  
 axis.title=element\_text(size=14,face="bold"))+  
 scale\_color\_manual(values=c(`Enhancer`="#999999", `Genes`="#E69F00", `Non-coding`="#56B4E9", `Promoters`= "#56B4E9"))

Again i need to re-rerun an older-chunk of code for this one to work. This code requires code written on the 16th. It will then match arx motif at 100%. I then again called lengths to see if Arx is at the same spacing.

Further, we invesetigated weather Arx motifs are in cis or Trans. Cis meaning on chromosome X and Trans meaning across strands. In addition, i plotted to see if Arx motifs across strands vs Arx motifs on the same strand. This however proved to be fruitless as complementary sequence would be the Arx motif in plaindromic arrangement.

##New code stuff  
library(seqLogo)  
library(magrittr)  
library(GenomicRanges)  
library(ggplot2)  
library(magrittr)  
library(tibble)  
library(pander)  
library(reshape2)  
library(plyr)  
library(MotifDb)  
library(BSgenome.Mmusculus.UCSC.mm9)  
  
arx6Mer <-  
 rbind(  
 A = c(0, 1, 1, 0, 0, 1),  
 C = c(0, 0, 0, 0, 0, 0),  
 G = c(0, 0, 0, 0, 0, 0) ,  
 T = c(1, 0, 0, 1, 1, 0)  
 )  
  
arx6MerPlaindrome<-  
 rbind(  
 A = c(1, 0, 0, 1, 1, 0),  
 C = c(0, 0, 0, 0, 0, 0),  
 G = c(0, 0, 0, 0, 0, 0) ,  
 T = c(0, 1, 1, 0, 0, 1)  
 )  
  
grange6mer<-matchPWM(arx6Mer, BSgenome.Mmusculus.UCSC.mm9, "100%")  
grange6merplaindrome<-matchPWM(arx6MerPlaindrome, BSgenome.Mmusculus.UCSC.mm9, "100%")  
  
  
grange6merPlus<-subset(grange6mer, strand=="+")  
grange6merMinus<-subset(grange6mer, strand=="-")  
grange6merPlaindromePlus<-subset(grange6merplaindrome, strand =="+")  
grange6merPlainDromeMinus<-subset(grange6merplaindrome, strand == "-")  
  
TransArxMotif<-distanceToNearest(grange6merPlus, grange6merPlainDromeMinus, ignore.strand=TRUE)  
intergerforTrans<-subset(TransArxMotif, distance>0 & distance<= 200 )%>%countRnodeHits()  
nonOverlappingMinusTransClusters<-subset(grange6merPlainDromeMinus, intergerforTrans)  
  
cisArxMotif<-distanceToNearest(grange6merPlus)  
intergerforCis<-subset(cisArxMotif, distance>0 & distance<= 200 )%>%countRnodeHits()  
nonOverlappingMinusCisMotifs<-subset(grange6merPlainDromeMinus, intergerforCis)  
  
  
  
UCSCgenes <- import("~/Scripts/March/mm9.bed")  
promoters <- promoters(UCSCgenes)  
  
distanceToNearest(cisArxMotifs, promoters)  
distanceToNearest(nonOverlappingMinusCisMotifs, promoters)  
  
distanceToNearest(nonOverlappingCisClusters, promoters)  
distanceToNearest(nonOverlappingCisClusters, promoters)%>%as.data.frame()  
ggplotCisPromoters<-distanceToNearest(nonOverlappingMinusCisMotifs, promoters)%>%as.data.frame()  
ggplotTransPromoters<-distanceToNearest(nonOverlappingMinusTransClusters, promoters)%>%as.data.frame()  
  
ggplotCisTrans<-merge(ggplotCisPromoters[3], ggplotTransPromoters[3], by=0, all= TRUE)  
ggplotCisTransReshaped<-reshape(ggplotCisTrans,  
 varying = c("distance.x", "distance.y"),  
 v.names = "Distance",  
 timevar = "Orientation",  
 times = c("Cis", "Trans"),  
 direction = "long")  
  
ggplot(ggplotCisTransReshaped, aes(x=Distance, group=Orientation))+  
 geom\_freqpoly(binwidth= 1000)+  
 theme\_bw()

I then again, plotted cis and trans motifs with ggplot.

23-4-2017 This code below has isolated the DNA strings that do not contain the Arx 6mer which can now be further analysed outside of R for sequence enrichement. The idea here is to identify any cofactors or other transcription factors by Arx which I can suggest for further analysis such as protein binding microarrays if the protein is expressed in the same cells and at the same time as Arx. This would explain why we see Arx ChIP-confirmed sites without Arx motifs.

##Sequence enrichment SCript  
  
  
library(readxl)  
library(magrittr)  
library(GenomicRanges)  
  
n2aChipSeq <- read\_excel("~/DataFiles/ChIPseq/Mouse/ChIPseqDataQuille2011.xls",   
 sheet = "only N2a")%>%as.data.frame  
embyroChipSeq <- read\_excel("~/DataFiles/ChIPseq/Mouse/ChIPseqDataQuille2011.xls", sheet = "only emb brain")%>%as.data.frame  
commonChipSeq <- read\_excel("~/DataFiles/ChIPseq/Mouse/ChIPseqDataQuille2011.xls", sheet = "common genes")%>%as.data.frame  
  
  
  
chipSeqDataCleaner<-function(x){  
 splitColoumnMinus<-data.frame(do.call('rbind', strsplit(as.character(x$location),'-',fixed=TRUE)))  
 colnames(splitColoumnMinus)<- c("X1", "end")#re naming the coloumns  
 splitColoumnSemiColon<-data.frame(do.call('rbind', strsplit(as.character(splitColoumnMinus$X1),':',fixed=TRUE)))  
 colnames(splitColoumnSemiColon)<- c("chromosome", "start")#renaming those two  
 geneSymbolMetaDataFromOriginalData<-x[2:3]  
 dataFrameOfChipSeqData<- cbind(geneSymbolMetaDataFromOriginalData, splitColoumnSemiColon, splitColoumnMinus[2])%>%na.omit()  
 removingTheNegatives<- cbind(dataFrameOfChipSeqData, (as.data.frame(as.numeric(as.character(dataFrameOfChipSeqData$end)))-as.data.frame(as.numeric(as.character(dataFrameOfChipSeqData$start)))))  
   
   
 negativesRemoved<-subset(removingTheNegatives, removingTheNegatives$`as.numeric(as.character(dataFrameOfChipSeqData$end))`>0)  
 grangeChipSeq<-makeGRangesFromDataFrame(negativesRemoved,  
 keep.extra.columns=FALSE,  
 ignore.strand=FALSE,  
 seqinfo=NULL,  
 seqnames.field=c("seqnames", "seqname",  
 "chromosome", "chrom",  
 "chr", "chromosome\_name",  
 "seqid"),  
 start.field="start",  
 end.field=c("end", "stop"),  
 strand.field="strand",  
 starts.in.df.are.0based=FALSE)  
}  
  
  
  
grangeN2aChipSeq<-chipSeqDataCleaner(n2aChipSeq)  
grangeBrainChipSeq<-chipSeqDataCleaner(embyroChipSeq)  
grangeCommonChipSeq<-chipSeqDataCleaner(commonChipSeq)  
  
####sequence Enrichement  
library(PWMEnrich)  
library(BSgenome.Mmusculus.UCSC.mm9)  
library(JASPAR2016)  
  
  
##Strings for Sequence Enrichement WITHOUT ARX motifs! 6mer specically  
  
##Motifs in Common genes with no Arx motifs!  
overlaps<- distanceToNearest(grangeCommonChipSeq, grange6mer)  
nonOverLappingInterger<- subset(overlaps, !distance==0)%>%countLnodeHits()  
nonArxContainingCommon<- subset(grangeCommonChipSeq, nonOverLappingInterger)  
stringsCommon<-getSeq(BSgenome.Mmusculus.UCSC.mm9,nonArxContainingCommon)  
  
##Motifs in Brain genes with no Arx motifs!  
overlapsBrain<- distanceToNearest(grangeBrainChipSeq, grange6mer)  
nonOverLappingIntergerBrain<- subset(overlapsBrain, !distance==0)%>%countLnodeHits()  
nonArxContainingBrain<- subset(grangeBrainChipSeq, nonOverLappingIntergerBrain)  
stringsBrain<-getSeq(BSgenome.Mmusculus.UCSC.mm9,nonArxContainingBrain)  
  
##Motifs in N2a genes with no Arx motifs!  
overlapsN2a<- distanceToNearest(grangeN2aChipSeq, grange6mer)  
nonOverLappingInterger<- subset(overlapsN2a, !distance==0)%>%countLnodeHits()  
nonArxContainingN2a<- subset(grangeN2aChipSeq, nonOverLappingInterger)  
stringsN2a<-getSeq(BSgenome.Mmusculus.UCSC.mm9,nonArxContainingN2a)

Again importing the ChIP peaks, and subsetting for Peaks which do not contain an Arx motif. I then got the DNA sequences for these peaks and exported them into Fasta files. These were exported and processed using WEEDER2.0 in bash.