May W4

Awais Choudhry

2017-5-23

23/5/2017

Today i went through the sequence enrichement PDfs i generated a couple of weeks ago, and attempted to see if there was anything enriched in the non-ARX containing peaks. I generated an excel sheet in which i counted the number of apperances of each transcription factors motif appeared in the enriched sequences. For example, transcription factor X, motif appeared 3 times in the enriched results of N2a cells, but 0 in the embyronic or common peaks. THe only transcription factor that appears in all 3 groups (the N2a peaks, the embyronic mouse brain and the common peaks) is AZF1. A couple of the TFs appeared in multiple peaks, these are: STAT1, CUP2, hb, Squamosa, Abd-B, NRIH2\_RXRA and arid3a I wanted to investigate the number of instances in which these TF's motifs appear in the ChIP-chip peaks to potentially identify if there is/are any TF/TFS whose motifs regularly appear in the peaks

library(readxl)  
library(magrittr)  
library(ggplot2)  
library(tibble)  
library(JASPAR2016)  
library(TFBSTools)  
  
library(BSgenome.Mmusculus.UCSC.mm9)  
library(readxl)  
## loading in the counts of Sequences  
 allJasparDataBase <- read\_excel("~/Scripts/May/allJasparDataBase.xlsx",   
 col\_names = FALSE)%>%as.data.frame  
 ##removing bad data   
allJasparDataBase <- split(allJasparDataBase, seq(nrow(allJasparDataBase)))  
allJasparDataBase<-lapply(allJasparDataBase, as.character)%>%unique  
allJasparDataBase<-as.list(cbind(allJasparDataBase))[1:length(allJasparDataBase)-1]  
  
##Loading in the ChIP-chip peaks  
  
  
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)  
  
  
  
  
## Doing it as a list  
permantDataFrame=NULL  
names<-NULL  
i=1  
for(i in 1:length(allJasparDataBase)) {  
   
abdB<-list()  
abdB[["ID"]]<- allJasparDataBase[[i]]  
abdBJasparSet<- getMatrixByID(JASPAR2016, abdB)  
abdBMatrix<- abdBJasparSet@profileMatrix  
  
names<-rbind(names,abdBJasparSet@name)  
  
## get PWM  
  
object<-matchPWM(abdBMatrix, BSgenome.Mmusculus.UCSC.mm9, "100%")  
  
  
tempdataFrame<-cbind(  
n2aAzf1Overlaps<-length(subset(grangeN2aChipSeq, findOverlaps(object, grangeN2aChipSeq)%>%countRnodeHits())),  
brainAzf1Overlaps<-subset(grangeBrainChipSeq, findOverlaps(object, grangeBrainChipSeq)%>% countRnodeHits())%>%length(),  
commonAzf1Overlaps<-subset(grangeCommonChipSeq, findOverlaps(object, grangeBrainChipSeq)%>%countRnodeHits())%>%length()  
)  
permantDataFrame<-rbind(permantDataFrame, tempdataFrame)  
i<-i+1  
}  
  
allJasparDataBase<-as.list(cbind(allJasparDataBase))[1:length(allJasparDataBase)-1]  
rownames(permantDataFrame)<-as.list(namesList)  
  
library(pander)  
sink("DataTableAllMotifs")  
permantDataFrame%>%pander()  
sink()  
  
ggplotAllMotifDataFrame<-as.data.frame(permantDataFrame)  
  
library(ggplot2)  
library(reshape2)  
ggplotAllMotifDataFrame<-rownames\_to\_column(ggplotAllMotifDataFrame)  
ggplotAllMotifDataFrame2<-reshape(ggplotAllMotifDataFrame,  
 varying = c( "V1", "V2", "V3"),  
 v.names = "Number of Motifs",  
 timevar = "Cell type",  
 times=c("N2a", "Embyronic Brain", "Common"),  
 direction = "long")  
  
ggplot(ggplotAllMotifDataFrame2, aes(x=`rowname`, y= `Number of Motifs`))+  
 geom\_bar(stat="identity",aes( fill=`Cell type`))+  
 theme\_bw()+  
 xlab(label= "Transcription Factor")+  
 ylab(label = "Number of Motifs in   
 ChIP-chip Peaks")+  
 theme(axis.text.x=element\_text(size=12, angle = 90, vjust = 0.5),  
 axis.title=element\_text(size=14,face="bold"))

What i have done, is developed a function that will take the input of the JASPAR ID, get the PWM, match it the mm9 genome, identify how many of these motifs coincide with the ARX Peaks in each repsective group (n2a, brasin and common) returning a table.

From the results, it is clear that we do not see any TF(s) that stand out and show significnat enrichement in the peaks. The highest enriched TF is arid3a, showing 67 motifs in ARX peaks, with the rest, Stat1, azf1m RXRA::NRH2, Squasoma, Cup2 and Hb, showing less than 10. If there is an ARX-TF complex forming, this complex is not binding to the monoeric site of the TF or ARX.

In conjunction with the poorly enriched sequeneces, I think the Quille et al data from 2011 is going to prove inconculsive in providing additional evidence in how ARX binds.

## Re running our mouse spacing numbers in human data.

## 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)  
  
  
##Mouse Inputs  
enhancerGrange <-  
 import(con = "~/DataFiles/Enhancer Tracks/Mouse/Enhanceresmm9.bed")  
UCSCgenes <- import("~/Scripts/March/mm9.bed")  
promoters <- promoters(UCSCgenes)  
gtfUCSCexonscoding<-import("~/Scripts/March/FullMm9genome.GTF")  
  
  
##Human inputs  
enhancerGrangeHumans<-import(con="~/DataFiles/Enhancer Tracks/Human/human\_permissive\_enhancers\_phase\_1\_and\_2.bed")  
UCSCgenesHuman<-import(con= "~/DataFiles/Gene Tracks/Human/hg.bed")  
promotersHuman<-promoters(UCSCgenesHuman)  
gtfUCSChHumans<-import(con="~/DataFiles/Gene Tracks/Human/hg19.gtf")  
  
##Inputs  
enhancerGrange<-enhancerGrangeHumans  
UCSCgenes<-UCSCgenesHuman  
promoters<-promotersHuman  
gtfUCSCexonscoding<-gtfUCSChHumans  
  
genomeInput<-BSgenome.Hsapiens.UCSC.hg19  
ArxPlaindrmicMinus1<-rbind( A=c(0,1,1,0,0,0,0,1,1,0),   
 C=c(0,0,0,0,0,0,0),  
 G=c(0,0,0,0,0,0,0),  
 T=c(1,0,0,1,1,1,1,0,0,1))  
  
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  
arxtandemMinus1<-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))  
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, genomeInput, "100%")  
grangeplaindromicMinus1 <-  
 matchPWM(ArxPlaindrmicMinus1, genomeInput, "100%")  
grangeplaindromicNospace <-  
 matchPWM(arx6MerPWMNospace, genomeInput, "100%")  
grangeplaindromic1space <-  
 matchPWM(arx6MerPWM1space, genomeInput, "100%")  
grangeplaindromic2space <-  
 matchPWM(arx6MerPWM2space, genomeInput, "100%")  
grangeplaindromic3space <-  
 matchPWM(arx6MerPWM3space, genomeInput, "100%")  
grangeplaindromic4space <-  
 matchPWM(arx6MerPWM4space, genomeInput, "100%")  
grangeplaindromic5space <-  
 matchPWM(arx6MerPWM5space, genomeInput, "100%")  
grangeplaindromic6space <-  
 matchPWM(arx6MerPWM6space, genomeInput, "100%")  
grangeplaindromic7space <-  
 matchPWM(arx6MerPWM7space, genomeInput, "100%")  
  
grangeTandemMinusOne <-  
 matchPWM(arxtandemMinus1, genomeInput, "100%")  
grangeTandemNoSpace<-  
 matchPWM(arxTandemNoSpace, genomeInput, "100%")  
grangeTandem1space <-  
 matchPWM(arxTandem1Space, genomeInput, "100%")  
grangeTandem2space <-  
 matchPWM(arxTandem2Space, genomeInput, "100%")  
grangeTandem3space <-  
 matchPWM(arxTandem3Space, genomeInput, "100%")  
grangeTandem4space <-  
 matchPWM(arxTandem4Space, genomeInput, "100%")  
grangeTandem5space <-  
 matchPWM(arxTandem5Space, genomeInput, "100%")  
grangeTandem6space <-  
 matchPWM(arxTandem6Space, genomeInput, "100%")  
grangeTandem7space <-  
 matchPWM(arxTandem7Space, genomeInput, "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(grangeTandemMinusOne),  
 dataTableNoGenesminus1 <-  
 sum(countOverlaps(grangeTandemMinusOne, UCSCgenes)),  
 dataTableMinus1 <-  
 sum(countOverlaps(grangeTandemMinusOne, promoters)),  
 dataTableMinus1r <-  
 sum(countOverlaps(grangeTandemMinusOne, enhancerGrange)),  
 (length(grangeTandemMinusOne)-sum(countOverlaps(grangeTandemMinusOne, enhancerGrange))-  
 sum(countOverlaps(grangeTandemMinusOne, promoters))- sum(countOverlaps(grangeTandemMinusOne, 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",  
 "Non Coding")  
  
  
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", "Non Coding", "Motifs in genes"),  
 v.names = "Numbers of Motif",  
 timevar = "Location",  
 times = c( "Promoters", "Enhancers", "Non coding","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(grangeplaindromicMinus1),  
 sum(countOverlaps(grangeplaindromicMinus1, UCSCgenes)),  
 sum(countOverlaps(grangeplaindromicMinus1, promoters)),  
 sum(countOverlaps(grangeplaindromicMinus1, enhancerGrange)),  
 (length(grangeplaindromicMinus1)-sum(countOverlaps(grangeplaindromicMinus1, enhancerGrange))-  
 sum(countOverlaps(grangeplaindromicMinus1, promoters))- sum(countOverlaps(grangeplaindromicMinus1, 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",  
 "Non Coding")  
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", "Non Coding", "Motifs in genes"),  
 v.names = "Numbers of Motif",  
 timevar = "Location",  
 times = c( "Promoters", "Enhancers", "Non coding","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"))  
  
  
  
##taking older code and redoing it so that it presents number of motifs from the GENE start site.   
  
  
  
startsites<-subset(gtfUCSCexonscoding, type=="start\_codon")  
  
dataFrameDistance1SpacePromoter <-  
 distanceToNearest(grangeTandem1space, startsites) %>%   
 as.data.frame()  
dataFrameDistance2SpacePromoter <-  
 distanceToNearest(grangeTandem2space, startsites) %>%  
 as.data.frame()  
dataFrameDistance3SpacePromoter <-  
 distanceToNearest(grangeTandem3space, startsites) %>%  
 as.data.frame()  
dataFrameDistance6SpacePromoter <-  
 distanceToNearest(grangeTandem6space, startsites) %>%  
 as.data.frame()  
  
  
dataFrameMerger<-function(z,x,c,v){  
   
 test<-merge(z[3],x[3],by=0, all=TRUE, row.names=NULL)  
 test2<-merge(test, c[3], by=0, all=TRUE, row.names=NULL)  
 test3<- merge(test2, v[3], by=0,all=TRUE, row.names=NULL)  
 return(test3)  
}  
  
dataFrameDistanceofTandemMotifsFromPromoter<-dataFrameMerger(dataFrameDistance1SpacePromoter,  
 dataFrameDistance2SpacePromoter,   
 dataFrameDistance3SpacePromoter,   
 dataFrameDistance6SpacePromoter)  
dataFrameDistanceofTandemMotifsFromPromoter<- dataFrameDistanceofTandemMotifsFromPromoter[4:7]  
colnames(dataFrameDistanceofTandemMotifsFromPromoter)<- c("1 Space",  
 "2 Space",  
 "3 Space",  
 "6 Space")  
ggplotdataFrameDistanceofTandemicMotifsFromPromoter<-reshape(dataFrameDistanceofTandemMotifsFromPromoter,  
 varying = c("1 Space", "2 Space", "3 Space", "6 Space"),  
 v.names = "Distance",  
 timevar = "Space",  
 times = c("1 Nucleotide", "2 Nucleotide", "3 Nucleotide", "6 Nucleotide"),  
 direction = "long")  
  
ggplot(ggplotdataFrameDistanceofTandemicMotifsFromPromoter, aes(x=Distance, group=Space, fill=Space))+  
 geom\_freqpoly(bins = 500, aes(colour=Space))+  
 theme\_bw()+  
 xlab(label = "Distance To The Closest Transcription Start Site(Base Pairs)")+  
 ylab(label= "Number of Motifs")+  
 theme(text = element\_text(size=12))+  
 scale\_x\_continuous(limits = c(0, 200000))+  
 scale\_y\_continuous(limits = c(0, 100))  
  
  
  
##histogram of distances of Plaindromic Motifs  
  
  
dataFrameDistancePlandromic1SpacePromoter <-  
 distanceToNearest(grangeplaindromic1space, startsites) %>% as.data.frame  
dataFrameDistancePlandromic2SpacePromoter <-  
 distanceToNearest(grangeplaindromic2space, startsites) %>% as.data.frame  
dataFrameDistancePlandromic3SpacePromoter <-  
 distanceToNearest(grangeplaindromic3space, startsites) %>% as.data.frame  
dataFrameDistancePlandromic4SpacePromoter <-  
 distanceToNearest(grangeplaindromic4space, startsites) %>% as.data.frame  
head(dataFrameDistancePlandromic4SpacePromoter)  
dataFrameDistanceofPlandromicMotifsFromPromoter <- dataFrameMerger(dataFrameDistancePlandromic1SpacePromoter,  
 dataFrameDistancePlandromic2SpacePromoter,  
 dataFrameDistancePlandromic3SpacePromoter,  
 dataFrameDistancePlandromic4SpacePromoter)  
  
  
dataFrameDistanceofPlandromicMotifsFromPromoter<-dataFrameDistanceofPlandromicMotifsFromPromoter[4:7]  
colnames(dataFrameDistanceofPlandromicMotifsFromPromoter)<- c("1 Space",  
 "2 Space",  
 "3 Space",  
 "4 Space")  
ggplotdataFrameDistanceofPlaindromicMotifsFromPromoter<-reshape(dataFrameDistanceofPlandromicMotifsFromPromoter,  
 varying = c("1 Space", "2 Space", "3 Space", "4 Space"),  
 v.names = "Distance",  
 timevar = "Space",  
 times = c("1 Nucleotide", "2 Nucleotide", "3 Nucleotide", "4 Nucleotide"),  
 direction = "long")  
ggplot(ggplotdataFrameDistanceofPlaindromicMotifsFromPromoter, aes(x=Distance, group=Space))+  
 geom\_freqpoly(bins = 500, aes(colour=Space))+  
 theme\_bw()+  
 theme(text = element\_text(size=16))+  
 xlab(label = "Distance To Closest Transcription Start Site(Base Pairs)")+  
 ylab(label= "Number Of Motifs")+  
 scale\_x\_continuous(limits = c(0, 200000))+  
 scale\_y\_continuous(limits = c(0, 100))  
  
  
  
  
## Average distances  
NumericTandem<-apply(dataFrameDistanceofTandemMotifsFromPromoter, 2, as.numeric)  
NumericPlandrimoc<-apply(dataFrameDistanceofPlandromicMotifsFromPromoter, 2, as.numeric)  
Space1Av<-sum(na.omit(NumericTandem[,1]))/length(na.omit(NumericTandem[,1]))  
Space2Av<-sum(na.omit(NumericTandem[,2]))/length(na.omit(NumericTandem[,2]))  
Space3Av<-sum(na.omit(NumericTandem[,3]))/length(na.omit(NumericTandem[,3]))  
Space4Av<-sum(na.omit(NumericTandem[,4]))/length(na.omit(NumericTandem[,4]))  
  
  
Space1AvPlandromic<-sum(na.omit(NumericPlandrimoc[,1]))/length(na.omit(NumericPlandrimoc[,1]))  
Space2AvPlandromic<-sum(na.omit(NumericPlandrimoc[,2]))/length(na.omit(NumericPlandrimoc[,2]))  
Space3AvPlandromic<-sum(na.omit(NumericPlandrimoc[,3]))/length(na.omit(NumericPlandrimoc[,3]))  
Space4AvPlamdromic<-sum(na.omit(NumericPlandrimoc[,4]))/length(na.omit(NumericPlandrimoc[,4]))

This script above develops 4 new graphs of the Arx spacing and orientations in the human model. This is used to develop which

24-5-2017

## HiC data

I have done a long series of things outside of R to get to the bedpe file i am importing, as listed below 1. I downloaded a series of draft reads off NCBI geo and used them with the HiC Pro pipeline Downloaded Reference: Shen Y, Yue F, McCleary DF, Ye Z et al. A map of the cis-regulatory sequences in the mouse genome. Nature 2012 Aug 2;488(7409):116-20. PMID: 22763441

1. I utilised Ning's script to convert the HiC Pro output into a .bedpe file
2. I moved the file to my VM
3. I imported the file and am now following the genomicIntearactions to generate HiC dataInputs

library(GenomicInteractions)  
library(Gviz)  
library(BSgenome.Mmusculus.UCSC.mm9)  
library(magrittr)  
library(Biostrings)  
library(TxDb.Mmusculus.UCSC.mm9.knownGene)  
  
## genome inputs  
txdb <- TxDb.Mmusculus.UCSC.mm9.knownGene  
genomeInput<-BSgenome.Mmusculus.UCSC.mm9  
## Code that doesn't need to be re-run  
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))  
  
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))  
   
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))  
 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))  
   
  
arxTandem6SpaceTFBS<-matchPWM(arx6MerPWM6space,genomeInput, "100%")  
arxPlaindromic4SpaceTFBS<-matchPWM(arx6MerPWM4space, genomeInput, "100%")  
arx6mer2SpaceTFBS<-matchPWM(arxTandem2Space,genomeInput, "100%")  
arx6merTFBS<-matchPWM(arx6Mer, BSgenome.Mmusculus.UCSC.mm9,"100%")  
  
  
  
## IMPORTs  
#IntearctionFile  
interactionHiCMouse<-makeGenomicInteractionsFromFile("/home/a1649239/properReadCounts.bedpe",   
 type = "bedpe", experiment\_name = "Draft HiC Mouse Embyronic", description = "mouseBrain" )  
genemm9bed<-import("~/DataFiles/Gene Tracks/Mouse/mm9.bed")  
enhancerGrange<-import("~/DataFiles/Enhancer Tracks/Mouse/Enhanceresmm9.bed")  
  
  
 H3K27acTrack<- import(con= "~/DataFiles/ChIPseq/Mouse/HistoneModifications/first5col/H3K27ac2.bed")%>%DataTrack(type= "histogram", name= "H3k27ac", col.histogram="green")  
 H3k36meTrack<- import(con= "~/DataFiles/ChIPseq/Mouse/HistoneModifications/first5col/H3K36me2.bed")%>%DataTrack(type= "histogram", name= "H3k36me", col.histogram="red")  
 H3K9me32Track<- import(con= "~/DataFiles/ChIPseq/Mouse/HistoneModifications/first5col/H3K9me32.bed")%>%DataTrack(type= "histogram", name= "H3K9me32", col.histogram="blue")  
  
  
  
## Does need to be re run  
Ebf3:chr7:144185354-144706128  
Shox2:chr3:66577191-66985693  
Lmo1:chr7:116082086-116513822  
chr<-"chr7"  
FromM<-116112086  
ToM<- 116493822  
  
  
arx6merTandem2chr<-subset(arx6mer2SpaceTFBS, seqnames==chr)  
arx6merTandem6chr<-subset(arxTandem6SpaceTFBS, seqnames==chr)  
arx6merPlaindromic4chr<-subset(arxPlaindromic4SpaceTFBS, seqnames==chr)  
arx6merchr<-subset(arx6merTFBS, seqnames==chr)  
  
  
  
  
## annotationTracks for each Arx Model!  
arxTandem2SpaceTrack<-AnnotationTrack(arx6merTandem2chr, name= "Tandem 2 Space")  
arxTandem6SpaceTrack<- AnnotationTrack(arx6merTandem6chr, name= "Tandem 6 Space")  
arxPlaindromic4SpaceTrack<-AnnotationTrack(arx6merPlaindromic4chr, name= "Plaindromic 4 Space")  
  
  
  
## Gene names from shiny app  
knownGenes<- GeneRegionTrack(txdb, genome="mm9", chromosome=chr, showId=TRUE, geneSymbol=TRUE, name="UCSC", stacking = "pack")  
symbols <- unlist(mapIds(org.Mm.eg.db, gene(knownGenes), "SYMBOL", "ENTREZID", multiVals = "first"))  
symbol(knownGenes) <- symbols[gene(knownGenes)]  
  
  
##Getting Promoter sequneces for subsetting Interaction files as a Grange!  
promotermm9<-subset(promoters(genemm9bed), seqnames==chr)%>%GRanges  
  
  
## Genomic Features Interaction Track  
promoterTrack <- AnnotationTrack(promotermm9, genome="mm9", name="Promoters", featureAnnotation="id", chromosome = chr, stacking= "dense")  
enhTrack <- AnnotationTrack("~/DataFiles/Enhancer Tracks/Mouse/Enhanceresmm9.bed", genome="mm9", name="Enhancers", stacking = "dense") # doesnt need to be here   
  
  
  
  
## Annotating the interactions between promoters and ARX!!  
  
#We need to an ID column in each Grange! So lets do that first, just renamoign a random column to ID  
  
colnames(mcols(enhancerGrange))<-c("id", "score", "itmeRgb", "thick", "blacks")  
colnames(mcols(promotermm9))<-c("id" , "score", "itemRgb", "thick", "blocks" )  
colnames(mcols(arx6mer2SpaceTFBS))<-c("id", "string")  
colnames(mcols(arxTandem6SpaceTFBS))<-c("id", "string")  
colnames(mcols(arxPlaindromic4SpaceTFBS))<-c("id", "string")  
colnames(mcols(arx6merTFBS))<-c("id", "string")  
annotation.features <- list(enhancer = enhancerGrange,   
 promoters=promotermm9,  
 ArxIntearctions= arx6mer2SpaceTFBS,  
 ArxIntearctions=arxTandem6SpaceTFBS,  
 ArxIntearctions=arxPlaindromic4SpaceTFBS ,  
 Arx6MerInteractions= arx6merTFBS)  
annotateInteractions(interactionHiCMouse, annotation.features)  
interactionsWithCountsAbove1<-interactionHiCMouse[interactionHiCMouse$counts>0]  
##Plotting the proportions of interactions in a pie chart  
plotInteractionAnnotations(interactionsWithCountsAbove1, legend = TRUE)  
  
## Okay now lets select for a subset of Intearactions!   
#THis subsets for 2 spaced Arx Tandem intearctions with promoters!  
  
allarx6MerInteractions<-interactionHiCMouse[isInteractionType(interactionHiCMouse, "Arx6MerInteractions", "promoters")]  
validArxIntearctions<-allarx6MerInteractions[allarx6MerInteractions$counts>0]%>%InteractionTrack(name = "Arx Specific Intearctions")  
  
  
  
  
  
# All interactions Track  
#mouseBrainInteractionsTrack <- InteractionTrack(interactionHiCMouse, name = "brain Interactions",chromosome = chr)  
  
  
##Colouring Tracks  
displayPars(enhTrack) <- list(fill = "black", col = NA)  
displayPars(validArxIntearctions) = list(col.interactions="red",  
 col.anchors.line = "black",  
 interaction.dimension="height",   
 interaction.measure ="counts",  
 plot.trans=FALSE,  
 plot.outside = TRUE,   
 col.outside="0",   
 anchor.height = 0.1)  
  
plotTracks(list(validArxIntearctions,   
 promoterTrack, knownGenes,   
 enhTrack, H3K27acTrack,H3k36meTrack, H3K9me32Track),   
 sizes = c(0.7, 0.2, 0.4, 0.1, 0.2 ,0.2, 0.2),from = FromM,to = ToM,   
 chromosome=chr,  
 cex.title = 0.72,   
 rotation.title = 0,   
 showAxis = FALSE,   
 background.title = "white",  
 lwd.title = 2,   
 title.width = 2,   
 cex.main = 5,   
 col = NULL,   
 fontcolor.title = "black")

Above is the code i have written that is finalised for indetifying 20kb bins which interact with ARX genome. The intent here was to select for Arx motifs which interact with promoter sequences hence influencing gene expression directly as oppose to indirectly. There are an extremely large quantity of interactions, the majoriety of which are between distal sites and distal sites (see pie chart). Secondly, there is an a small subset of interactions in which ARX intearctions influence.

## Methylation Influencing Arx Binding!

Recently, like 2 weeks ago, Jolma released a paper where the results suggested that homeodomain TFs prefer to bind to TCGTTA as the mCpG appears quite similar to the AA sequence according the paper. In the supplementary data, they found that Arx motif did not change binding specificty in methylated SELEX and unmethylated SELEX henceforth, to confirm i am examing the ChIP-seq peaks to see how many times the TGCTTA motif appears.

arxMethylatedTFBS<-matchPWM(round(PWM("TCGTTA")\*7), BSgenome.Mmusculus.UCSC.mm9, "100%")  
  
##Loading in the ChIP-chip peaks  
  
library(BSgenome.Mmusculus.UCSC.mm9)  
library(readxl)  
  
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)  
  
grangeN2aChipSeq<-subset(grangeN2aChipSeq, width<10012)  
  
numberOfPotentialMethylatedSites<-cbind(  
sum(findOverlaps(arxMethylatedTFBS, grangeN2aChipSeq)%>%countRnodeHits()),  
sum(findOverlaps(arxMethylatedTFBS, grangeBrainChipSeq)%>%countRnodeHits()),  
sum(findOverlaps(arxMethylatedTFBS, grangeCommonChipSeq)%>%countRnodeHits())  
)  
colnames(numberOfPotentialMethylatedSites)<-c("N2a", "Embyronic Mouse Brain", "Common")  
rownames(numberOfPotentialMethylatedSites)<-c("Number Of motifs")  
library(pander)  
numberOfPotentialMethylatedSites%>%pander()

As indicated by the lack of motifs found in the Arx ChIP-chip peaks (data from Quille et al 2011 paper) it is clear that in invivo ARX does not bind to TmCGTTA as an alternative motif. As previously descrived according to: "Impact of cytosine methylation on DNA binding specificities of human transcription factors" by Yimeng Yin et al (2017) they described ARX as not being influenced by methylation. We can further back this claim as over-expression of Arx has not bound to only 4 TGCTTA sequences, none of which correlate with methylation sites.

Furthermore, today, after attaining the Hi-C data from HiC-Pro on the server, i redeveloped the shiny app from the ground up to use this HiC data inconjunction with a variety of ARX motifs. The ability to select for particular tracks is remains to be developed. A massive short comming is that the tracks can only be up to a certain size however, the amount of data greatly execeedes it hence, i constantly reloading the data each time the start, finish and Chromosome changes is required to make the app run. Ideally i'd like to have the data all loaded before hand however, this is not possible.

#  
# This is a Shiny web application. You can run the application by clicking  
# the 'Run App' button above.  
#  
# Find out more about building applications with Shiny here:  
#  
# http://shiny.rstudio.com/  
#  
  
library(shiny)  
  
# Define UI for application that draws a histogram  
ui <- fluidPage(  
   
 # Application title  
 titlePanel("Arx Mouse Model"),  
   
 # Sidebar with a slider input for number of bins   
 sidebarLayout(  
 sidebarPanel(  
 numericInput("fromM", "Starting Base",value = 90531985),  
 numericInput("toM", "Finishing Base", value = 90543694),  
   
 selectInput("chrM", label = h3("Select box"),   
 choices = list("Chromosome 1" = "chr1",  
 "Chromosome 2" = "chr2",   
 "Chromosome 3" = "chr3",  
 "Chromosome 4" = "chr4",  
 "Chromosome 5" = "chr5",  
 "Chromosome 6" = "chr6",  
 "Chromosome 7" = "chr7",  
 "Chromosome 8" = "chr8",  
 "Chromosome 9" = "chr9",  
 "Chromosome 10" = "chr10",  
 "Chromosome 11" = "chr11",  
 "Chromosome 12" = "chr12",  
 "Chromosome 13" = "chr13",  
 "Chromosome 14" = "chr14",  
 "Chromosome 15" = "chr15",  
 "Chromosome 16" = "chr16",  
 "Chromosome 17" = "chr17",  
 "Chromosome 18" = "chr18",  
 "Chromosome 19" = "chr19",  
 "Chromosome X" = "chrX",  
 "Chromosome Y" = "chrY"), selected = "chrX"),  
 hr(),  
 fluidRow(column(3, verbatimTextOutput("value"))  
 ),  
   
 # Copy the line below to make a checkbox  
 checkboxInput("cluster", label = "Clustered Sites", value = FALSE)  
   
 ),  
   
   
 # Show a plot of the generated distribution  
 mainPanel(  
 plotOutput("gvizPlot")  
   
 )  
 )  
)  
  
# Define server logic required to draw a histogram  
server <- function(input, output) {  
   
 output$gvizPlot <- renderPlot({  
 if(!exists("chr")){  
 library(GenomicInteractions)  
 library(Gviz)  
 library(BSgenome.Mmusculus.UCSC.mm9)  
 library(magrittr)  
 ## genome inputs  
 txdb <- TxDb.Mmusculus.UCSC.mm9.knownGene  
 assign("txdb", txdb, .GlobalEnv)  
 genomeInput<-BSgenome.Mmusculus.UCSC.mm9  
 ## Code that doesn't need to be re-run  
 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))  
   
 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))  
   
 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))  
 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))  
   
   
 arxTandem6SpaceTFBS<<-matchPWM(arx6MerPWM6space,genomeInput, "100%")  
 arxPlaindromic4SpaceTFBS<<-matchPWM(arx6MerPWM4space, genomeInput, "100%")  
 arx6mer2SpaceTFBS<<-matchPWM(arxTandem2Space,genomeInput, "100%")  
 arx6merTFBS<<-matchPWM(arx6Mer, BSgenome.Mmusculus.UCSC.mm9,"100%")  
   
 assign("arxTandem6SpaceTFBS", arxTandem6SpaceTFBS, .GlobalEnv)  
 assign("arxPlaindromic4SpaceTFBS", arxPlaindromic4SpaceTFBS, .GlobalEnv)  
 assign("arx6mer2SpaceTFBS", arx6mer2SpaceTFBS, .GlobalEnv)  
 assign("arx6merTFBS", arx6merTFBS, .GlobalEnv)  
   
 ## IMPORTs  
 #IntearctionFile  
 interactionHiCMouse<<-makeGenomicInteractionsFromFile("/home/a1649239/properReadCounts.bedpe",   
 type = "bedpe", experiment\_name = "Draft HiC Mouse Embyronic", description = "mouseBrain" )  
 genemm9bed<<-import("~/DataFiles/Gene Tracks/Mouse/mm9.bed")  
 enhancerGrange<<-import("~/DataFiles/Enhancer Tracks/Mouse/Enhanceresmm9.bed")  
   
   
 H3K27acTrack<<- import(con= "~/DataFiles/ChIPseq/Mouse/HistoneModifications/first5col/H3K27ac2.bed")%>%DataTrack(type= "histogram", name= "H3k27ac",   
 col.histogram="green")  
 H3k36meTrack<<- import(con= "~/DataFiles/ChIPseq/Mouse/HistoneModifications/first5col/H3K36me2.bed")%>%DataTrack(type= "histogram", name= "H3k36me",  
 col.histogram="red")  
 H3K9me32Track<<- import(con= "~/DataFiles/ChIPseq/Mouse/HistoneModifications/first5col/H3K9me32.bed")%>%DataTrack(type= "histogram", name= "H3K9me32",  
 col.histogram="blue")  
 assign("interactionHiCMouse", interactionHiCMouse, .GlobalEnv)  
 assign("genemm9bed", genemm9bed, .GlobalEnv)  
 assign("enhancerGrange", enhancerGrange, .GlobalEnv)  
 assign("H3K27acTrack", H3K27acTrack, .GlobalEnv)  
 assign("H3k36meTrack", H3k36meTrack, .GlobalEnv)  
 assign("H3K9me32Track", H3K9me32Track, .GlobalEnv)  
 assign("enhancerGrange", enhancerGrange, .GlobalEnv)  
 assign("H3K27acTrack", H3K27acTrack, .GlobalEnv)  
   
 ## Does need to be re run  
  
 chr<-input$chrM  
 FromM<-input$fromM  
 ToM<- input$toM  
 assign("chr", chr, .GlobalEnv)  
 assign("FromM", FromM, .GlobalEnv)  
 assign("ToM", ToM, .GlobalEnv)  
   
 arx6merTandem2chr<-subset(arx6mer2SpaceTFBS, seqnames==chr)  
 arx6merTandem6chr<-subset(arxTandem6SpaceTFBS, seqnames==chr)  
 arx6merPlaindromic4chr<-subset(arxPlaindromic4SpaceTFBS, seqnames==chr)  
 arx6merchr<-subset(arx6merTFBS, seqnames==chr)  
 arx6merchr1<-subset(arx6merchr, start>=FromM)  
 arx6merchr1<-subset(arx6merchr1, end<=ToM)  
 assign("arx6merTandem2chr", arx6merTandem2chr, .GlobalEnv)  
 assign("arx6merTandem6chr", arx6merTandem6chr, .GlobalEnv)  
 assign("arx6merPlaindromic4chr", arx6merPlaindromic4chr, .GlobalEnv)  
 assign("arx6merchr1", arx6merchr, .GlobalEnv)  
 arxMotifTrack<-AnnotationTrack(arx6merchr1, genome="mm9", name= "Arx 6 mer", stacking = "dense")  
   
   
 ## annotationTracks for each Arx Model!  
 arxTandem2SpaceTrack<-AnnotationTrack(arx6merTandem2chr, name= "Tandem 2 Space")  
 arxTandem6SpaceTrack<- AnnotationTrack(arx6merTandem6chr, name= "Tandem 6 Space")  
 arxPlaindromic4SpaceTrack<-AnnotationTrack(arx6merPlaindromic4chr, name= "Plaindromic 4 Space")  
 assign("arxTandem2SpaceTrack", arxTandem2SpaceTrack, .GlobalEnv)  
 assign("arxTandem6SpaceTrack", arxTandem6SpaceTrack, .GlobalEnv)  
 assign("arxPlaindromic4SpaceTrack", arxPlaindromic4SpaceTrack, .GlobalEnv)  
   
   
   
 ## Gene names from shiny app  
 knownGenes<- GeneRegionTrack(txdb, genome="mm9", chromosome=chr, showId=TRUE, geneSymbol=TRUE, name="UCSC", stacking = "pack")  
 symbols <- unlist(mapIds(org.Mm.eg.db, gene(knownGenes), "SYMBOL", "ENTREZID", multiVals = "first"))  
 symbol(knownGenes) <- symbols[gene(knownGenes)]  
   
 assign("knownGenes", knownGenes, .GlobalEnv)  
 assign("symbols", symbols, .GlobalEnv)  
   
 ##Getting Promoter sequneces for subsetting Interaction files as a Grange!  
 promotermm9<-subset(promoters(genemm9bed), seqnames==chr)%>%GRanges  
 assign("promotermm9", promotermm9, .GlobalEnv)  
  
   
 ## Genomic Features Interaction Track  
 promoterTrack <- AnnotationTrack(promotermm9, genome="mm9", name="Promoters", featureAnnotation="id", chromosome = chr, stacking= "dense")  
 enhTrack <- AnnotationTrack("~/DataFiles/Enhancer Tracks/Mouse/Enhanceresmm9.bed", genome="mm9", name="Enhancers", stacking = "dense") # doesnt need to be here   
 arxMotifTrack<-AnnotationTrack(arx6merchr, genome="mm9", name= "Arx 6 mer", stacking = "dense")  
 assign("promoterTrack", promoterTrack, .GlobalEnv)  
 assign("enhTrack", enhTrack, .GlobalEnv)  
 assign("arxMotifTrack", arxMotifTrack, .GlobalEnv)  
   
   
 ## Annotating the interactions between promoters and ARX!!  
   
 #We need to an ID column in each Grange! So lets do that first, just renamoign a random column to ID  
   
 colnames(mcols(enhancerGrange))<-c("id", "score", "itmeRgb", "thick", "blacks")  
 colnames(mcols(promotermm9))<-c("id" , "score", "itemRgb", "thick", "blocks" )  
 colnames(mcols(arx6mer2SpaceTFBS))<-c("id", "string")  
 colnames(mcols(arxTandem6SpaceTFBS))<-c("id", "string")  
 colnames(mcols(arxPlaindromic4SpaceTFBS))<-c("id", "string")  
 colnames(mcols(arx6merTFBS))<-c("id", "string")  
 annotation.features <- list(enhancer = enhancerGrange,   
 promoters=promotermm9,  
 ArxIntearctions= arx6mer2SpaceTFBS,  
 ArxIntearctions=arxTandem6SpaceTFBS,  
 ArxIntearctions=arxPlaindromic4SpaceTFBS ,  
 Arx6MerInteractions= arx6merTFBS)  
 annotateInteractions(interactionHiCMouse, annotation.features)  
  
   
 ## Okay now lets select for a subset of Intearactions!   
 #THis subsets for 2 spaced Arx Tandem intearctions with promoters!  
 interactionHiCMouseTrack<-interactionHiCMouse[interactionHiCMouse$counts>0]%>%InteractionTrack(name= "All Interactions")  
 allarx6MerInteractions<-interactionHiCMouse[isInteractionType(interactionHiCMouse, "Arx6MerInteractions", "promoters")]  
 validArxIntearctions<-allarx6MerInteractions[allarx6MerInteractions$counts>0]%>%InteractionTrack(name = "Arx Specific Intearctions")  
 assign("allarx6MerInteractions", allarx6MerInteractions, .GlobalEnv)  
 assign("validArxIntearctions", validArxIntearctions, .GlobalEnv)  
   
   
   
   
 # All interactions Track  
 #mouseBrainInteractionsTrack <- InteractionTrack(interactionHiCMouse, name = "brain Interactions",chromosome = chr)  
   
   
 ##Colouring Tracks  
 displayPars(enhTrack) <- list(fill = "black", col = NA)  
 displayPars(validArxIntearctions) = list(col.interactions="red",  
 col.anchors.line = "black",  
 interaction.dimension="height",   
 interaction.measure ="counts",  
 plot.trans=FALSE,  
 plot.outside = TRUE,   
 col.outside="0",   
 anchor.height = 0.1)  
   
 plotTracks(list(interactionHiCMouseTrack, arxMotifTrack, lncRNAmm9Track,  
 promoterTrack, knownGenes,   
 enhTrack, H3K27acTrack,H3k36meTrack, H3K9me32Track),   
 sizes = c(0.7, 0.2, 0.2, 0.2, 0.4, 0.1, 0.2 ,0.2, 0.2),from = FromM,to = ToM,   
 chromosome=chr,  
 cex.title = 0.72,   
 rotation.title = 0,   
 showAxis = FALSE,   
 background.title = "white",  
 lwd.title = 2,   
 title.width = 2,   
 cex.main = 5,   
 col = NULL,   
 fontcolor.title = "black")  
   
   
 }  
 else if(!input$chrM==chr){  
 ## Does need to be re run  
   
 chr<-input$chrM  
 FromM<-input$fromM  
 ToM<- input$toM  
 assign("chr", chr, .GlobalEnv)  
 assign("FromM", FromM, .GlobalEnv)  
 assign("ToM", ToM, .GlobalEnv)  
   
 arx6merTandem2chr<-subset(arx6mer2SpaceTFBS, seqnames==chr)  
 arx6merTandem6chr<-subset(arxTandem6SpaceTFBS, seqnames==chr)  
 arx6merPlaindromic4chr<-subset(arxPlaindromic4SpaceTFBS, seqnames==chr)  
 arx6merchr<-subset(arx6merTFBS, seqnames==chr)  
 arx6merchr1<-subset(arx6merchr, start>=FromM)  
 arx6merchr1<-subset(arx6merchr1, end<=ToM)  
 assign("arx6merTandem2chr", arx6merTandem2chr, .GlobalEnv)  
 assign("arx6merTandem6chr", arx6merTandem6chr, .GlobalEnv)  
 assign("arx6merPlaindromic4chr", arx6merPlaindromic4chr, .GlobalEnv)  
 assign("arx6merchr1", arx6merchr, .GlobalEnv)  
 arxMotifTrack<-AnnotationTrack(arx6merchr1, genome="mm9", name= "Arx 6 mer", stacking = "dense")  
   
   
 ## annotationTracks for each Arx Model!  
 arxTandem2SpaceTrack<-AnnotationTrack(arx6merTandem2chr, name= "Tandem 2 Space")  
 arxTandem6SpaceTrack<- AnnotationTrack(arx6merTandem6chr, name= "Tandem 6 Space")  
 arxPlaindromic4SpaceTrack<-AnnotationTrack(arx6merPlaindromic4chr, name= "Plaindromic 4 Space")  
 assign("arxTandem2SpaceTrack", arxTandem2SpaceTrack, .GlobalEnv)  
 assign("arxTandem6SpaceTrack", arxTandem6SpaceTrack, .GlobalEnv)  
 assign("arxPlaindromic4SpaceTrack", arxPlaindromic4SpaceTrack, .GlobalEnv)  
   
   
   
 ## Gene names from shiny app  
 knownGenes<- GeneRegionTrack(txdb, genome="mm9", chromosome=chr, showId=TRUE, geneSymbol=TRUE, name="UCSC", stacking = "pack")  
 symbols <- unlist(mapIds(org.Mm.eg.db, gene(knownGenes), "SYMBOL", "ENTREZID", multiVals = "first"))  
 symbol(knownGenes) <- symbols[gene(knownGenes)]  
   
 assign("knownGenes", knownGenes, .GlobalEnv)  
 assign("symbols", symbols, .GlobalEnv)  
   
 ##Getting Promoter sequneces for subsetting Interaction files as a Grange!  
 promotermm9<-subset(promoters(genemm9bed), seqnames==chr)%>%GRanges  
 assign("promotermm9", promotermm9, .GlobalEnv)  
   
   
 ## Genomic Features Interaction Track  
 promoterTrack <- AnnotationTrack(promotermm9, genome="mm9", name="Promoters", featureAnnotation="id", chromosome = chr, stacking= "dense")  
 enhTrack <- AnnotationTrack("~/DataFiles/Enhancer Tracks/Mouse/Enhanceresmm9.bed", genome="mm9", name="Enhancers", stacking = "dense") # doesnt need to be here   
 arxMotifTrack<-AnnotationTrack(arx6merchr1, genome="mm9", name= "Arx 6 mer", stacking = "dense")  
 assign("promoterTrack", promoterTrack, .GlobalEnv)  
 assign("enhTrack", enhTrack, .GlobalEnv)  
 assign("arxMotifTrack", arxMotifTrack, .GlobalEnv)  
   
   
 ## Annotating the interactions between promoters and ARX!!  
   
 #We need to an ID column in each Grange! So lets do that first, just renamoign a random column to ID  
   
 colnames(mcols(enhancerGrange))<-c("id", "score", "itmeRgb", "thick", "blacks")  
 colnames(mcols(promotermm9))<-c("id" , "score", "itemRgb", "thick", "blocks" )  
 colnames(mcols(arx6mer2SpaceTFBS))<-c("id", "string")  
 colnames(mcols(arxTandem6SpaceTFBS))<-c("id", "string")  
 colnames(mcols(arxPlaindromic4SpaceTFBS))<-c("id", "string")  
 colnames(mcols(arx6merTFBS))<-c("id", "string")  
 annotation.features <- list(enhancer = enhancerGrange,   
 promoters=promotermm9,  
 ArxIntearctions= arx6mer2SpaceTFBS,  
 ArxIntearctions=arxTandem6SpaceTFBS,  
 ArxIntearctions=arxPlaindromic4SpaceTFBS ,  
 Arx6MerInteractions= arx6merTFBS)  
 annotateInteractions(interactionHiCMouse, annotation.features)  
   
   
 ## Okay now lets select for a subset of Intearactions!   
 #THis subsets for 2 spaced Arx Tandem intearctions with promoters!  
   
 allarx6MerInteractions<-interactionHiCMouse[isInteractionType(interactionHiCMouse, "Arx6MerInteractions", "promoters")]  
 validArxIntearctions<-allarx6MerInteractions[allarx6MerInteractions$counts>0]%>%InteractionTrack(name = "Arx Specific Intearctions")  
 assign("allarx6MerInteractions", allarx6MerInteractions, .GlobalEnv)  
 assign("validArxIntearctions", validArxIntearctions, .GlobalEnv)  
   
   
   
   
 # All interactions Track  
 #mouseBrainInteractionsTrack <- InteractionTrack(interactionHiCMouse, name = "brain Interactions",chromosome = chr)  
   
   
 ##Colouring Tracks  
 displayPars(enhTrack) <- list(fill = "black", col = NA)  
 displayPars(validArxIntearctions) = list(col.interactions="red",  
 col.anchors.line = "black",  
 interaction.dimension="height",   
 interaction.measure ="counts",  
 plot.trans=FALSE,  
 plot.outside = TRUE,   
 col.outside="0",   
 anchor.height = 0.1)  
   
 plotTracks(list(validArxIntearctions, arxMotifTrack,   
 promoterTrack, knownGenes,   
 enhTrack, H3K27acTrack,H3k36meTrack, H3K9me32Track),   
 sizes = c(0.7, 0.2, 0.2, 0.4, 0.1, 0.2 ,0.2, 0.2),from = FromM,to = ToM,   
 chromosome=chr,  
 cex.title = 0.72,   
 rotation.title = 0,   
 showAxis = FALSE,   
 background.title = "white",  
 lwd.title = 2,   
 title.width = 2,   
 cex.main = 5,   
 col = NULL,   
 fontcolor.title = "black")  
   
 }  
 else{  
 chr<-input$chrM  
 FromM<-input$fromM  
 ToM<- input$toM  
   
 arx6merchr<-subset(arx6merTFBS, seqnames==chr)  
 arx6merchr1<-subset(arx6merchr, start>=FromM)  
 arx6merchr1<-subset(arx6merchr1, end<=ToM)  
 arxMotifTrack<-AnnotationTrack(arx6merchr1, genome="mm9", name= "Arx 6 mer", stacking = "dense")  
   
 plotTracks(list(validArxIntearctions, arxMotifTrack,   
 promoterTrack, knownGenes,   
 enhTrack, H3K27acTrack,H3k36meTrack, H3K9me32Track),   
 sizes = c(0.7, 0.2, 0.2, 0.4, 0.1, 0.2 ,0.2, 0.2),from = FromM,to = ToM,   
 chromosome=chr,  
 cex.title = 0.72,   
 rotation.title = 0,   
 showAxis = FALSE,   
 background.title = "white",  
 lwd.title = 2,   
 title.width = 2,   
 cex.main = 5,   
 col = NULL,   
 fontcolor.title = "black")  
   
 }  
 }, width= "auto", height=700)  
}  
  
# Run the application   
shinyApp(ui = ui, server = server)

26-5-2017 Identifying what the Actual ARX gene region interacts with to see if there is potential mutations in the gene influence the expression of other regulatory elements either long non-coding RNAs, promoters or other promoters as there is an enhancer in ARX gene. To do this I have imported the locations of each mapped raw read. These raw reads are the valid pairs, as determined by the HiC pro pipeline. These raw reads are subsetted based on which componment of the read i am looking at (R1 vs R2). So i have imported R1's that are above 90 000 000bp and below 91 000 000pb for R1 hence allowing R2 to bind to anything else on the x chromosome. I also imported R2 mapping between 90 000 000bp and 91 000 000bp allowing R1 to map anywhere.

I have further subsetted both of these reads for the reads mapping (either R1 or R2) to ARX gene and see where their counter part is mapped. In addition, i have taken 100kb upstream and 100kb down stream to examine where these reads are mapped repsectively. Any peaks we can investigate further on the UCSC genome browser.

library(readr)  
rawARXreads <- read\_delim("~/DataFiles/HiC/Mouse/chrX3",   
 "\t", escape\_double = FALSE, col\_names = FALSE,   
 trim\_ws = TRUE)%>%as.data.frame()  
  
rawARXreadsR2 <- read\_delim("~/DataFiles/HiC/Mouse/chrXR2",   
 "\t", escape\_double = FALSE, col\_names = FALSE,   
 trim\_ws = TRUE)%>%as.data.frame()  
  
  
rawARXreadsR2Intra<-rawARXreadsR2[rawARXreadsR2$X2==rawARXreadsR2$X5,]  
rawARXreadsIntra<-rawARXreads[rawARXreads$X2==rawARXreads$X5,]  
  
rawARXreadsR2Intra<-subset(rawARXreadsR2Intra, 90531985<X6)  
rawARXreadsR2Intra<-subset(rawARXreadsR2Intra, 90543694>X6)  
  
rawARXreadsIntra<-subset(rawARXreadsIntra, 90531985<X3)  
rawARXreadsIntra<-subset(rawARXreadsIntra, 90543694>X3)  
  
ggplot<-c(rawARXreadsIntra$X6,rawARXreadsR2Intra$X3 )%>%as.data.frame  
  
  
  
  
ggplot(ggplot, aes(x=.))+  
 geom\_freqpoly(bins=100)+  
 theme\_bw()+  
 geom\_vline(xintercept=90531985)+  
 geom\_vline(xintercept=90543694)  
  
## where does the 100kb upstream of ARX map to?!?!   
  
rawARXreadsR2Intra<-rawARXreadsR2[rawARXreadsR2$X2==rawARXreadsR2$X5,]  
rawARXreadsIntra<-rawARXreads[rawARXreads$X2==rawARXreads$X5,]  
rawARXreadsR2Intra<-subset(rawARXreadsR2Intra, 90131985<X6)  
rawARXreadsR2Intra<-subset(rawARXreadsR2Intra, 90531985>X6)  
  
rawARXreadsIntra<-subset(rawARXreadsIntra, 90131985<X3)  
rawARXreadsIntra<-subset(rawARXreadsIntra, 90531985>X3)  
  
ggplot<-c(rawARXreadsIntra$X6,rawARXreadsR2Intra$X3 )%>%as.data.frame  
  
  
ggplot(ggplot, aes(x=., fill=.))+  
 geom\_freqpoly(bins=100, aes(colour=.))+  
 theme\_bw()+  
 geom\_vline(xintercept=90531985)+  
 geom\_vline(xintercept=90543694)  
  
  
##Down stream which contains a in the gene desert!  
  
rawARXreadsR2Intra<-rawARXreadsR2[rawARXreadsR2$X2==rawARXreadsR2$X5,]  
rawARXreadsIntra<-rawARXreads[rawARXreads$X2==rawARXreads$X5,]  
rawARXreadsR2Intra<-subset(rawARXreadsR2Intra, 90543694<X6)  
rawARXreadsR2Intra<-subset(rawARXreadsR2Intra, 90643694>X6)  
  
rawARXreadsIntra<-subset(rawARXreadsIntra, 90543694<X3)  
rawARXreadsIntra<-subset(rawARXreadsIntra, 90643694>X3)  
  
ggplot<-c(rawARXreadsIntra$X6,rawARXreadsR2Intra$X3 )%>%as.data.frame  
  
  
ggplot(ggplot, aes(x=.))+  
 geom\_freqpoly(bins=100)+  
 theme\_bw()+  
 geom\_vline(xintercept=90531985)+  
 geom\_vline(xintercept=90543694)  
  
  
  
  
rawARXreadsR2Intra<-rawARXreadsR2[rawARXreadsR2$X2==rawARXreadsR2$X5,]  
rawARXreadsIntra<-rawARXreads[rawARXreads$X2==rawARXreads$X5,]  
rawARXreadsR2Intra<-subset(rawARXreadsR2Intra, 90043694<X6)  
rawARXreadsR2Intra<-subset(rawARXreadsR2Intra, 90943694>X6)  
  
rawARXreadsIntra<-subset(rawARXreadsIntra, 90043694<X3)  
rawARXreadsIntra<-subset(rawARXreadsIntra, 90943694>X3)  
  
ggplot<-c(rawARXreadsIntra$X6,rawARXreadsR2Intra$X3 )%>%as.data.frame  
  
  
ggplot(ggplot, aes(x=.))+  
 geom\_histogram(bins=100)+  
 theme\_bw()+  
 geom\_vline(xintercept=90531985)+  
 geom\_vline(xintercept=90543694)

As evident from all the results, almost all reads are mapped to to the immdeidate surroundings in all the plots. However, it is evident there is a very small increase in readings maping at 10 millionbps and 150 million data relative to the 20million bp closer. This is particular for the 20million. Furthermore, there is an increase in the number of reads at 80million compared to 100 million (both approximately 10million bps from Arx gene

### Differentially expressed Genes

This Code identifies the distance to ARX motifs relative to the closest transcritpion start site.

PA1DifferentiallyExpressedGenes<-import(con="/home/a1649239/DataFiles/Tessa differential Expressed/PoylATessa (1).gtf")  
  
startSitesPA1Deregulated<-subset(PA1DifferentiallyExpressedGenes, type=="start\_codon")  
  
mm9Gtf<-import(con="~/DataFiles/Gene Tracks/Mouse/FullMm9genome.GTF")  
mm9GtfStartSites<-subset(mm9Gtf, type=="start\_codon")  
nonderegulatedPoylA1<-mm9GtfStartSites[!mm9GtfStartSites %over% startSitesPA1Deregulated]  
  
  
dataFrameMerger<-function(z,x,c,v){  
   
 test<-merge(z,x,by=0, all=TRUE, row.names=NULL)  
 test2<-merge(test, c, by=0, all=TRUE, row.names=NULL)  
 test3<- merge(test2, v, by=0,all=TRUE, row.names=NULL)  
 return(test3)  
}  
  
  
#6mer  
DistanceToClosestArxMotifPoylA1<-distanceToNearest(startSitesPA1Deregulated, arx6merTFBS )%>%as.data.frame()  
DistanceToClosestArxMotifPoylA1<-DistanceToClosestArxMotifPoylA1$distance%>%as.character%>%as.numeric%>%as.data.frame  
  
#plaindromic Distance  
DistanceToClosestArxMotifPoylA14plaindromic<-distanceToNearest(startSitesPA1Deregulated, arx6merPlaindromic4chr )%>%as.data.frame()  
DistanceToClosestArxMotifPoylA14plaindromic<-DistanceToClosestArxMotifPoylA14plaindromic$distance%>%as.character%>%as.numeric%>%as.data.frame  
#tandem2Space  
DistanceToClosestArxMotifPoylA12Tandem<-distanceToNearest(startSitesPA1Deregulated, arx6mer2SpaceTFBS )%>%as.data.frame()  
DistanceToClosestArxMotifPoylA12Tandem<-DistanceToClosestArxMotifPoylA12Tandem$distance%>%as.character%>%as.numeric%>%as.data.frame  
#tandem6Space  
DistanceToClosestArxMotifPoylA16Tandem<-distanceToNearest(startSitesPA1Deregulated, arx6merTandem6chr )%>%as.data.frame()  
DistanceToClosestArxMotifPoylA16Tandem<-DistanceToClosestArxMotifPoylA16Tandem$distance%>%as.character%>%as.numeric%>%as.data.frame  
  
poylADistanceToMotifModels<-dataFrameMerger(DistanceToClosestArxMotifPoylA1,  
 DistanceToClosestArxMotifPoylA14plaindromic,  
 DistanceToClosestArxMotifPoylA12Tandem ,  
 DistanceToClosestArxMotifPoylA16Tandem)  
poylADistanceToMotifModels<-poylADistanceToMotifModels[4:7]  
colnames(poylADistanceToMotifModels)<-c("6mer", "4 Spaced Plaindrome", "2 Spaced Tandem", "6 Spaced Tandem")  
  
ggplotpoylADistanceToMotifModels<-reshape(poylADistanceToMotifModels,   
 varying= c("6mer", "4 Spaced Plaindrome", "2 Spaced Tandem", "6 Spaced Tandem"),  
 v.names= "Distance",  
 timevar = "Motif Model",  
 times= c("6mer", "4 Spaced Plaindrome", "2 Spaced Tandem", "6 Spaced Tandem"),  
 direction="long")  
  
  
  
#6mer  
DistanceToClosestArxMotifControl<-distanceToNearest(nonderegulatedPoylA1, arx6merTFBS )%>%as.data.frame()  
DistanceToClosestArxMotifControl<-DistanceToClosestArxMotifControl$distance%>%as.character%>%as.numeric%>%as.data.frame  
#plaindromic Distance  
DistanceToClosestArxMotifControl4plaindromic<-distanceToNearest(nonderegulatedPoylA1, arx6merPlaindromic4chr )%>%as.data.frame()  
DistanceToClosestArxMotifControl4plaindromic<-DistanceToClosestArxMotifControl4plaindromic$distance%>%as.character%>%as.numeric%>%as.data.frame  
#tandem2Space  
DistanceToClosestArxMotifControl2Tandem<-distanceToNearest(nonderegulatedPoylA1, arx6mer2SpaceTFBS )%>%as.data.frame()  
DistanceToClosestArxMotifControl2Tandem<-DistanceToClosestArxMotifControl2Tandem$distance%>%as.character%>%as.numeric%>%as.data.frame  
#tandem6Space  
DistanceToClosestArxMotifControl6Tandem<-distanceToNearest(nonderegulatedPoylA1, arx6merTandem6chr )%>%as.data.frame()  
DistanceToClosestArxMotifControl6Tandem<-DistanceToClosestArxMotifControl6Tandem$distance%>%as.character%>%as.numeric%>%as.data.frame  
  
  
controlDistanceToMotifModels<-dataFrameMerger(DistanceToClosestArxMotifControl,  
 DistanceToClosestArxMotifControl4plaindromic,  
 DistanceToClosestArxMotifControl2Tandem ,  
 DistanceToClosestArxMotifControl6Tandem)  
controlDistanceToMotifModels<-controlDistanceToMotifModels[4:7]  
colnames(controlDistanceToMotifModels)<-c("6mer", "4 Spaced Plaindrome", "2 Spaced Tandem", "6 Spaced Tandem")  
  
ggplotcontrolDistanceToMotifModels<-reshape(controlDistanceToMotifModels,   
 varying= c("6mer", "4 Spaced Plaindrome", "2 Spaced Tandem", "6 Spaced Tandem"),  
 v.names= "Distance",  
 timevar = "Motif Model",  
 times= c("6mer", "4 Spaced Plaindrome", "2 Spaced Tandem", "6 Spaced Tandem"),  
 direction="long")  
  
  
  
  
library(reshape2)  
ggplotPoylAandControlDistances<-reshape(PoylAandControlDistances,  
 varying = c("..x", "..y"),  
 v.names = "Distance",  
 timevar = "Mutation",  
 times = c("PoylA", "Control"),  
 direction = "long")  
  
  
  
library(ggplot2)  
ggplot(ggplotpoylADistanceToMotifModels, aes(x=Distance, group=`Motif Model`))+  
 geom\_histogram(aes(fill=`Motif Model`), bins=100)+  
 theme\_bw()+  
 xlab(label="Distance from transcription start site (bp)")+  
 ylab(label="Number of Arx Motifs")+  
 facet\_wrap(~`Motif Model`, scale="free")  
  
ggplot(ggplotcontrolDistanceToMotifModels, aes(x=Distance, group=`Motif Model`))+  
 geom\_histogram(aes(fill=`Motif Model`), bins=100)+  
 theme\_bw()+  
 xlab(label="Distance from transcription start site (bp)")+  
 ylab(label="Number of Arx Motifs")+  
 facet\_wrap(~`Motif Model`, scale="free")

Unfortunately, we do not see any corerlation between motif distance and to TSS and differnitally gene expression. Likely because genomic DNA-sequence is insufficent. Hence, integration of other genomic data types such as conservation and epigenomics would be useful in this instance. Also as this motif is degenerate amongst many HD Tfs, we can likely believe it is not

### 28-5-2017

Another rather important ARX question we have is the number of motifs correlating with the level of differenital expression.

PA1DifferentiallyExpressedGenes<-import(con="/home/a1649239/DataFiles/Tessa differential Expressed/PoylATessa (1).gtf")  
  
startSitesPA1Deregulated<-subset(PA1DifferentiallyExpressedGenes, type=="start\_codon")  
  
mm9Gtf<-import(con="~/DataFiles/Gene Tracks/Mouse/FullMm9genome.GTF")  
mm9GtfStartSites<-subset(mm9Gtf, type=="start\_codon")  
nonderegulatedPoylA1<-mm9GtfStartSites[!mm9GtfStartSites %over% startSitesPA1Deregulated]  
mm9Bed<-import(con="~/DataFiles/Gene Tracks/Mouse/mm9.bed")  
library(readxl)  
Supplementary\_tables\_1\_3 <- read\_excel("~/DataFiles/Tessa differential Expressed/Supplementary\_tables\_1-3.xls",   
 sheet = "Table 3 -PolyA pool dereg genes",   
 skip = 1)  
  
  
polyApool<-import("~/DataFiles/Gene Tracks/Mouse/PolyApool (1).bed")  
poylApoolPromoters<-promoters(polyApool)  
PoylA1<-import("~/DataFiles/Tessa differential Expressed/PA1.bed")%>%promoters()  
PoylA2<-import("~/DataFiles/Tessa differential Expressed/PolA2.bed")%>%promoters()  
  
promoterRegions<-promoters(mm9Bed)  
promoterRegionsPolyA<-promoters(PA1DifferentiallyExpressedGenes)  
  
  
promotersNonPoylADeregulated<-promoterRegions[!promoterRegions %over% promoterRegionsPolyA]  
  
## Getting Number of Motifs upstream  
NonPolyADeregulated<-findOverlaps(arx6mer2SpaceTFBS, promotersNonPoylADeregulated)%>%countRnodeHits()%>%as.data.frame()  
poylApoolNumberOfMotifs<-findOverlaps(arx6mer2SpaceTFBS, poylApoolPromoters)%>%countRnodeHits()%>%as.data.frame()  
PoylA1NUmberOfMotifs<-findOverlaps(arx6mer2SpaceTFBS, PoylA1)%>%countRnodeHits()%>%as.data.frame()  
PA2NumberOfMotifs<- findOverlaps(arx6mer2SpaceTFBS, PoylA2)%>%countRnodeHits()%>%as.data.frame()  
  
  
  
dataFrameMerger<-function(z,x,c,v){  
   
 test<-merge(z,x,by=0, all=TRUE, row.names=NULL)  
 test2<-merge(test, c, by=0, all=TRUE, row.names=NULL)  
 test3<- merge(test2, v, by=0,all=TRUE, row.names=NULL)  
 return(test3)  
}  
  
PoylAandControl<-dataFrameMerger(NonPolyADeregulated, poylApoolNumberOfMotifs, PoylA1NUmberOfMotifs, PA2NumberOfMotifs)  
PoylAandControl<-PoylAandControl[4:7]  
colnames(PoylAandControl)<-c("Control", "Poyl A Pool", " PoylA1 Deregulated Genes", "PolyA2 Deregulated Genes")  
  
  
ggplotReshapedpoylA<-reshape(PoylAandControl,  
varying =c("Control", "Poyl A Pool", " PoylA1 Deregulated Genes", "PolyA2 Deregulated Genes"),  
 v.names = "Number Of Motifs",  
 timevar = "Regulation",   
 times = c("Control", "Poyl A Pool"," PoylA1 Deregulated Genes", "PolyA2 Deregulated Genes"),  
 direction = "long")  
  
  
  
ggplot(ggplotReshapedpoylA, aes(x=`Number Of Motifs`, group = Regulation))+  
 geom\_histogram(aes(fill=Regulation))+  
 facet\_wrap( ~ Regulation, scales="free\_y")+  
 coord\_cartesian(ylim=c(0,100))

From this plot we can see there is no correlation in the average distance of genomic interactions.

## Bioinformatics investigating the promoter regions of Tessa's differential expressed genes

PA1DifferentiallyExpressedGenes<-import(con="/home/a1649239/DataFiles/Tessa differential Expressed/PoylATessa (1).gtf")  
  
startSitesPA1Deregulated<-subset(PA1DifferentiallyExpressedGenes, type=="start\_codon")  
  
mm9Gtf<-import(con="~/DataFiles/Gene Tracks/Mouse/FullMm9genome.GTF")  
mm9GtfStartSites<-subset(mm9Gtf, type=="start\_codon")  
nonderegulatedPoylA1<-mm9GtfStartSites[!mm9GtfStartSites %over% startSitesPA1Deregulated]  
mm9Bed<-import(con="~/DataFiles/Gene Tracks/Mouse/mm9.bed")  
library(readxl)  
Supplementary\_tables\_1\_3 <- read\_excel("~/DataFiles/Tessa differential Expressed/Supplementary\_tables\_1-3.xls",   
 sheet = "Table 3 -PolyA pool dereg genes",   
 skip = 1)  
  
  
polyApool<-import("~/DataFiles/Gene Tracks/Mouse/PolyApool (1).bed")  
poylApoolPromoters<-promoters(polyApool)  
PoylA1<-import("~/DataFiles/Tessa differential Expressed/PA1.bed")%>%promoters()  
PoylA2<-import("~/DataFiles/Tessa differential Expressed/PolA2.bed")%>%promoters()  
  
promoterRegions<-promoters(mm9Bed)  
promoterRegionsPolyA<-promoters(PA1DifferentiallyExpressedGenes)  
  
  
promotersNonPoylADeregulated<-promoterRegions[!promoterRegions %over% promoterRegionsPolyA]  
  
## Getting Number of Motifs upstream  
NonPolyADeregulated6mer<-findOverlaps(arx6merTFBS, promotersNonPoylADeregulated)%>%countRnodeHits()%>%as.data.frame()  
poylApoolNumberOfMotifs6mer<-findOverlaps(arx6merTFBS, poylApoolPromoters)%>%countRnodeHits()%>%as.data.frame()  
PoylA1NUmberOfMotifs6mer<-findOverlaps(arx6merTFBS, PoylA1)%>%countRnodeHits()%>%as.data.frame()  
PA2NumberOfMotifs6mer<- findOverlaps(arx6merTFBS, PoylA2)%>%countRnodeHits()%>%as.data.frame()  
  
  
  
dataFrameMerger<-function(z,x,c,v){  
   
 test<-merge(z,x,by=0, all=TRUE, row.names=NULL)  
 test2<-merge(test, c, by=0, all=TRUE, row.names=NULL)  
 test3<- merge(test2, v, by=0,all=TRUE, row.names=NULL)  
 return(test3)  
}  
  
PoylAandControl6mer<-dataFrameMerger(NonPolyADeregulated6mer, poylApoolNumberOfMotifs6mer, PoylA1NUmberOfMotifs6mer, PA2NumberOfMotifs6mer)  
PoylAandControl6mer<-PoylAandControl6mer[4:7]  
colnames(PoylAandControl6mer)<-c("Control", "Poyl A Pool", " PoylA1 Deregulated Genes", "PolyA2 Deregulated Genes")  
  
  
ggplotReshapedpoylA6mer<-reshape(PoylAandControl6mer,  
varying =c("Control", "Poyl A Pool", " PoylA1 Deregulated Genes", "PolyA2 Deregulated Genes"),  
 v.names = "Number Of Motifs",  
 timevar = "Regulation",   
 times = c("Control", "Poyl A Pool"," PoylA1 Deregulated Genes", "PolyA2 Deregulated Genes"),  
 direction = "long")  
  
  
  
ggplot(ggplotReshapedpoylA6mer, aes(x=`Number Of Motifs`, group = Regulation))+  
 geom\_histogram(aes(fill=Regulation))+  
 facet\_wrap( ~ Regulation, scales="free\_y")

Examining this promoter region we can see only a few of the genes actually have an Arx motif (6mer or anothe rmotif model) present in the promoter region indicaiting either long range enhancer interacitons are occuring or Arx is binding to a different motif!

### Investigating genes with differential expression of Quille and Colanstance

Previous datasets have identified genes with differenital expression using microarrays however, the number of genes they report as differentially expressed varieis a lot. Hence we are comparing how many of these genes have an Arx motif and how many do not. Because this is microarray, it is likely missing a lot of genes however, it is the only publically available data.

library(rtracklayer)  
genomeInput<-BSgenome.Mmusculus.UCSC.mm9  
## Code that doesn't need to be re-run  
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))  
  
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))  
   
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))  
 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))  
   
  
arxTandem6SpaceTFBS<-matchPWM(arx6MerPWM6space,genomeInput, "100%")  
arxPlaindromic4SpaceTFBS<-matchPWM(arx6MerPWM4space, genomeInput, "100%")  
arx6mer2SpaceTFBS<-matchPWM(arxTandem2Space,genomeInput, "100%")  
arx6merTFBS<-matchPWM(arx6Mer, BSgenome.Mmusculus.UCSC.mm9,"100%")  
  
library(rtracklayer)  
library(ggplot2)  
library(reshape2)  
library(magrittr)  
library(Biostrings)  
colsantae2008<-import(con="~/DataFiles/Tessa differential Expressed/GaiaColasante.bed")%>%promoters()  
quille2011<-import(con="~/DataFiles/Tessa differential Expressed/DifferentiallyExpressedQuille.bed")%>%promoters()  
  
  
  
  
colsantae6mer<-findOverlaps(arx6merTFBS, colsantae2008)%>%countRnodeHits()%>%as.data.frame()  
quille6mer<-findOverlaps(arx6merTFBS, quille2011)%>%countRnodeHits()%>%as.data.frame()  
  
quileEtColstane6mer<-merge(colsantae6mer, quille6mer, by=0, all=TRUE)  
colnames(quileEtColstane6mer)<-c("rownames", "colstane", "quille")  
reshape(quileEtColstane6mer,  
 times = "Paper",   
 varying = c("colstane", "quille"),v.names = "Number Of Motifs",  
 timevar = c("colstane", "quille"), direction = "long" )

29-5-2017 #Comparing the number of ARX interactions

Here we are playing with our Hi-C Dataset. We are trying to annoate interactions between genoic regions that contain an Arx motif and

validArxIntearctions  
 annotation.features <- list(Arx6MerInteractions= arx6merTFBS)  
 annotateInteractions(interactionHiCMouse, annotation.features)  
 allarx6MerInteractions<-interactionHiCMouse[isInteractionType(interactionHiCMouse, "Arx6MerInteractions", "promoters")]

whilest we are able to identify these interactions, we are clearly seeing that basically every genomic bin contains an Arx motif. Hence we will need to reduce the number of motifs further and the size of the HI-C data further.