Lab Book March

22/3/2017

Today i set out to acheive a series of small goals as follows: 1. Gviz plots/tracks instead the base functions

I spent a lot of my day reading through the Gviz manual attempting to trouble shoot many issues relating to file size, and formatting to importing into a format that will allow plotting with Gviz. Altering the server script only changed the single render plot. This is because altering the script is only part of single reactive renderplot function.

library(Gviz)  
library(rtracklayer)  
#HUMAN data tracks as i will be using both mouse and human data tracks.   
  
  
### setting up which region to import from the files as size is an issue  
##from, to and chr will be used for all tracks.  
from <- 28000000  
to <- 29000000  
chr <- "chrX"  
##human Tracks  
  
setwd("~/Research Proposal/")  
#importing the GC content track from UCSC  
gcContent <- UcscTrack(genome = "hg19",  
 chromosome = chr,  
 track = "GC Percent",  
 table = "gc5Base",  
 trackType = "DataTrack",  
 start = "start",  
 end = "end",   
 data = "score",   
 type = "hist",   
 window = -1,  
 windowSize = 1500,  
 fill.histogram = "black",  
 col.histogram = "black",  
 ylim = c(30, 70),  
 name = "GC Percent",  
 from = from,  
 to = to)  
  
#importing the UCSC gene Track for the hg19 genome  
  
knownGenes <- UcscTrack(genome = "hg19",   
 chromosome = chr,  
 track = "knownGene",   
 trackType = "GeneRegionTrack",  
 rstarts = "exonStarts",  
 rends = "exonEnds",  
 gene = "name",  
 symbol = "name",  
 transcript = "name",  
 strand = "strand",  
 fill = "#8282d2",  
 name = "UCSC Genes",  
 stacking = "dense",  
 showID = TRUE)  
  
  
## importing TF chipConfirmed tracks  
ChIpconfirmedbed<- import.bed("5col.bed")  
chipConfirmedSitesV3<-DataTrack(ChIpconfirmedbed,  
 genome = "hg19",  
 chromosome = chr,  
 to = to,   
 from = from,   
 name = "ChIPSeqV3Bed",  
 names = "name",  
 type = "histogram",  
 fill = "green",  
 color = "green")  
  
  
##Ideogram Track  
chromosomeBandTrack<- IdeogramTrack(genome = "hg19",   
 chromosome = chr)

##MYmade GC content Track  
#This was because the UCSC gene track above wont plot due to size.  
library(BSgenome.Hsapiens.UCSC.hg19)  
library(TxDb.Hsapiens.UCSC.hg19.knownGene)  
library(magrittr)  
library(biomaRt)  
library(ggplot2)  
library(dplyr)  
library(GenomicRanges)  
library(zoo)  
library(magrittr)  
library(parallel)  
library(biomaRt)  
library(MotifDb)  
library(seqLogo)  
  
mdb <- MotifDb  
matrices.human <- MotifDb::query(mdb, 'hsapiens')  
  
pfm.arx\_jolma2013.jaspar <- MotifDb::query(mdb, 'Hsapiens-jolma2013-ARX')[[1]]  
pcm.arx\_jolma2013.jaspar <- round(100 \* pfm.arx\_jolma2013.jaspar)  
  
arxMotifs <- matchPWM(pcm.arx\_jolma2013.jaspar,BSgenome.Hsapiens.UCSC.hg19[["chrX"]], "90%")  
ranges<-cbind.data.frame(as.data.frame(ranges(arxMotifs)), as.data.frame(arxMotifs), seqnames = c(chr))  
  
ARXMotifsGRangeForChrX<- makeGRangesFromDataFrame(ranges,  
 keep.extra.columns=TRUE,  
 ignore.strand=FALSE,  
 seqinfo=NULL,  
 seqnames.field="seqnames",  
 start.field="start",  
 end.field=c("end", "stop"),  
 strand.field="strand",  
 starts.in.df.are.0based=FALSE)  
  
arxMotifsTrack <- AnnotationTrack(ARXMotifsGRangeForChrX,  
 genome= "hg19",  
 chromosome = chr,  
 name= "ARX binding sites",  
 stacking = "dense",  
 colour = "red")  
  
  
### DNase Sec Track from Encode from John Stamatoyannopoulos, UW, link: https://www.encodeproject.org/experiments/ENCSR344FLH/  
  
dnaseSeq <- import("ENCFF752YMC.bigWig", which =  
 GRanges("chrX", IRanges(1, end = .Machine$integer.max - 1)))  
#converting to dataTrack  
dnaseSeqTrack <- DataTrack(dnaseSeq,  
 from = from,  
 to = to,  
 colour = "green",  
 name = "DnaseSeq105brainmale")  
  
  
## Enhancer Tracks from phantom 5 website for the hg19 genome  
  
enhancerTrack <- import("human\_permissive\_enhancers\_phase\_1\_and\_2.bed") %>% DataTrack( to = to, from = from, chromosome = chr, type = "histogram")  
  
  
###plotTracks  
  
plotTracks(list(chromosomeBandTrack, knownGenes, arxMotifsTrack, chipConfirmedSitesV3 ,dnaseSeqTrack, enhancerTrack), to= to, from = from )

This code above generates a gviz plot that plots 5 tracks; an ideogram, knowngenes track, Arx tf binding sites. chipseq confirmed transcription factor binding sites, a DNA occupancy track and an enhancer track. All of this data is from public sources: the ENCODE project for the dnaseseq track, the ChIPseq track came from UCSC track developed by researchers at standford and Yale and the enhancer track is from fantom5. It worth noting that due to lack of RAM currently i cannot play the GC content Track. The basic outlie of this data is that at the top the to, from and chromsome can be the reactive outputs hence changing them will change the outputs. Unfortunately due to lack of memory we cannot preload all the data hence changes in them will take a while to process and regenerate the data as it is downloaded off the UCSC hub.

#### 23-3-2017

Today i re-wrote the code i wrote above but in this instance i used mouse data tracks. The code is almost identicla in structure however, as mouse genome has a lot less data i made use of what is available. Some challenges i faced included finding data for specific tissue and the correct version of the mm9 genome.

## Mouse Tracks: they're necessary as i'll be ultising mouse models  
library(rtracklayer)  
library(Gviz)  
fromM<- 90531985  
toM<- 90543694  
chrM<- "chrX"  
  
setwd("~/Research Proposal/")  
##Ideogram  
  
ideogramTrackMouse <- IdeogramTrack(genome = "mm9",  
 chrM)  
##Genes  
  
knownGenesMouse <- UcscTrack(genome = "mm9",   
 chromosome = chrM,  
 track = "knownGene",   
 trackType = "GeneRegionTrack",  
 rstarts = "exonStarts",  
 rends = "exonEnds",  
 gene = "name",  
 symbol = "name",  
 transcript = "name",  
 strand = "strand",  
 fill = "#8282d2",  
 name = "UCSC Genes",  
 stacking = "pack")  
  
##ChIP confirmed TFBS  
chipConfirmedSitesMouse <- UcscTrack(genome = "mm9",  
 chromosome = chrM,  
 track = "PSU TFBS",  
 trackType = "DataTrack",  
 gene = "name",  
 symbol = "name",  
 name = "PSU TFBS")  
  
##GC content  
  
gcContentMouse <- UcscTrack(genome = "mm9",  
 chromosome = chrM,  
 track = "GC Percent",  
 table = "gc5Base",  
 trackType = "DataTrack",  
 start = "start",  
 end = "end",  
 data = "score",   
 type = "hist",   
 window = -1,  
 windowSize = 1500,  
 fill.histogram = "black",  
 col.histogram = "black",  
 ylim = c(30, 70),   
 name = "GC Percent")  
## Mouse Arx Motifs  
  
library(BSgenome.Mmusculus.UCSC.mm9)  
library(Gviz)  
library(magrittr)  
mdb <- MotifDb  
mouseArx <- MotifDb::query(mdb, 'Mmusculus-jolma2013-Arx')[[1]]  
mouseArxPWM <- round(mouseArx\*100)  
mouseArxDataFrame<- matchPWM(mouseArxPWM, BSgenome.Mmusculus.UCSC.mm9[["chrX"]], "90%")%>%  
 ranges()%>%  
 as.data.frame()%>% cbind(seqnames=("chrX"))  
grangesMouseArx<- makeGRangesFromDataFrame(mouseArxDataFrame,  
 keep.extra.columns=FALSE,  
 ignore.strand=FALSE,  
 seqinfo=NULL,  
 seqnames.field="seqnames",  
 start.field="start",  
 end.field=c("end", "stop"),  
 strand.field="strand",  
 starts.in.df.are.0based=FALSE)  
##Arx mouse track  
MouseARXmotifs<-AnnotationTrack(grangesMouseArx,  
 genome= "hg19",  
 chromosome = "chrX",  
 name= "ARX binding sites",  
 stacking = "dense",  
 colour = "red")  
  
##DNase seq track  
  
dnaseSeqMouse <- import(con = "~/DataFiles/ChIPseq/Mouse/cerebrummousednaseseq.bigWig", which =  
 GRanges(chrM, IRanges(1, end = .Machine$integer.max - 1)))%>% DataTrack( to = toM, from = fromM)  
  
  
##Replace Chipconfirmed sites  
plotTracks(list(ideogramTrackMouse, knownGenesMouse, MouseARXmotifs, dnaseSeqMouse),  
 to = toM,   
 from= fromM)

Above is the code i have written for the mouse genome. It will plot the ARX motifs, known mouse genes, ideogram track and the DNase seq and It requires the ChIPseq Data from cheryl's lab in addition to any other data i can get my hands on. The code is very simple, it takes the 3 variables at the top and then queries each of the respective track's datafiles for the data for that region. For instance the gene range track takes the object "chrM" which as stated above it is currently called "chrX" and queries the UCSC database for the genes in this region. This structure will make it readily trasnferable to the shiny app as the shiny app can use interactive inputs to give new values to chrM. This will then re-render the plot to the corresponding values.

29-3-2017

Today i attempted to generate a conservation track which i can utilise to add a conservation score to Arx motifs to potentially identify "important" motifs vs not important motifs. In addition, this may help identify motifs which appear simply by chance whereas other motifs which are actively selected for/conserved.

##Conservations!  
library(rtracklayer)  
library(Gviz)  
library(magrittr)  
   
 session <- browserSession()  
 genome(session) <- "mm9"  
 trackNames(session) ## list the track names  
 ## choose the Conservation track for a portion of mm9 chrM  
 query <- ucscTableQuery(session, "Conservation",  
 GRangesForUCSCGenome("mm9", chrM,  
 IRanges(1, .Machine$integer.max - 1)))  
 ## list the table names  
 tableNames(query)  
 ## get the phastCons30way track  
 tableName(query) <- "phastConsElements30way" ## retrieve the track data  
 conservationTrack<-track(query) # a GRanges object  
   
 tableName(query) <- "multiz30way"## get a data.frame summarizing the multiple alignment  
 object<-getTable(query)  
 dataFrameGrange<- as.data.frame(cbind(object[2:4], object[7]))  
 GrangeConservation<- makeGRangesFromDataFrame(dataFrameGrange,  
 keep.extra.columns=TRUE,  
 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)%>% DataTrack(type = "l",  
 name = "Conservation Track")

Above is the data Track i used for applying a conservation score. I also plotted the track ultising Gviz however, the grange can be used to identify the conservation score of the motif.

30-3-2017

We believe protein dosage may play an influence on which genes Arx can regulate. For example, genes that only require a single Arx protein to repress or active would only have a single Arx motif hence would require a lower dosage of the Arx protein for repression. Genes with multiple Arx motifs may require multiple Arx proteins binding to these sites resulting in order for proper regulation to occur hence a lower dosage of Arx would only occupy a proportion of the sites resulting in deregulation of Arx. So in order to regulate Arx.

To do this i wrote a piece of code that would identify Arx motifs that are located within 200bp of one another.

q<- 2  
w<-1  
distances <- NA  
for(i in 1:dim(mouseArxDataFrame)[1]){  
distances<-cbind(distances ,mouseArxDataFrame[q,1]-mouseArxDataFrame[w,2])  
print(w)  
q<- q+1  
w<- w+1  
}  
  
dataFrameDistances<-list(t(distances))%>%as.data.frame()  
combinedDataFrameDistances<- cbind(mouseArxDataFrame, dataFrameDistances[1:dim(dataFrameDistances)[1]-1,] )  
names(combinedDataFrameDistances) <- c("start", "end", "width", "seqnames", "distancebetween")  
  
grangeDataFrame<-combinedDataFrameDistances[combinedDataFrameDistances$distancebetween<=200,]  
grangeDataFrame2<- grangeDataFrame[grangeDataFrame$distance>=0,]  
clustereDataTrack<-makeGRangesFromDataFrame(grangeDataFrame2[-1,],  
 seqinfo=NULL,  
 seqnames.field="seqnames",  
 start.field="start",  
 end.field=c("end", "stop"),  
 strand.field="strand",  
 starts.in.df.are.0based=FALSE) %>%AnnotationTrack(genome= "mm9",  
 chromosome = chrM,  
 name= "ARX binding sites",  
 stacking = "dense",  
 colour = "red")

As i am still unfamiliar with GRanges, i convereted everything to a dataframe first. I then wrote a for loop which will substract the end of one Arx motif to the start of the second Arx motif. These values will then be assigned to the Arx motif from which they start with. I then subset the dataframe for Arx motifs with a distance equal to or less than 200bp and convert back to a GRange which is then used to plot Arx motifs.

31-3-2017

Below is a small segement of code that develops a table of differentially expressed Arx genes confirmed by Tessa's paper. There are some glitches i have noticed, when transfering from gene symbols to UCSC gene IDs. My though process was rather simple i wanted to create a script that can iterate through the long list of genes giving the; closest Arx TFBS, cluster and the conservation score of the respective sites.

##Table developer  
  
##First thing is that we need to convert all relevant data to dataframes for easy queries  
  
  
library(readr)  
library(dplyr)  
#importing the relevant Data  
  
geneSymbols <- read\_tsv("~/DataFiles/Tessa differential Expressed/2colmultiplenames.txt")  
pointer<-import(con= "~/DataFiles/Tessa differential Expressed/PA16col.bed")%>%as.data.frame()  
enhancerDataFrame<- import( con = "~/DataFiles/Enhancer Tracks/Mouse/Enhanceresmm9.bed", which =  
 GRanges(chrM, IRanges(1, end = .Machine$integer.max - 1)))%>%ranges()%>%as.data.frame()  
  
differentiallyExpressed <- left\_join(pointer, geneSymbols, by = c("name"="#kgID"))  
geneNames<- as.data.frame(unique(geneSymbols[2]))  
x<-1  
  
for(i in 1:dim(geneNames[1])){  
  
  
selectedGene<-differentiallyExpressed[differentiallyExpressed$geneSymbol==as.character(geneNames[x,1]),]  
toM<-max(selectedGene[3])+10000  
fromM<-min(selectedGene[2])-10000  
chrM<-selectedGene[1,1]  
  
  
##closet ARX motif, done via subtraction of gene start site - ARX data frame and locating the closest TFBS  
  
##distance to ARX motif, Select for motif starts and the start site of gene find the lowest value  
tableDistanceToArxTFBS<- min(abs(mouseArxDataFrame[2]-min(selectedGene[2])))  
  
#remove the NA from the Cluster Data frame ##i shoudl rename this ;')   
##find distance between the start site of gene and the lo  
grangeDataFrame3<-grangeDataFrame2[-1,]  
tableDistanceToArxCluster<-min(abs(grangeDataFrame3[1]-min(selectedGene[2])),  
 na.rm = FALSE)  
##distance of motifs to enhancer?  
tableDistanceToEnhancer<- min(abs(enhancerDataFrame-(min(selectedGene[2])+tableDistanceToArxTFBS)))  
   
   
tableConservationScoreOfTFBS<-dataFrameConversation$score[abs(min(selectedGene[2]+tableDistanceToArxTFBS)-dataFrameConversation[2])==min(abs(min(selectedGene[2]+tableDistanceToArxTFBS)-dataFrameConversation[2]))]  
tableConservationScoreOfCluster<-dataFrameConversation$score[abs(min(selectedGene[2]+tableDistanceToArxCluster)-dataFrameConversation[2])==min(abs(min(selectedGene[2]+tableDistanceToArxCluster)-dataFrameConversation[2]))]  
  
if(!exists("ArxTable3"))  
 {  
  
ArxTable3<- cbind("Gene"=selectedGene[1,8],  
 "Chromosome" = as.character(selectedGene[1,1]),  
 "Start"= min(selectedGene[2]),  
 "End"=max(selectedGene[3]),   
 "Distance to closest Arx TFBS"=tableDistanceToArxTFBS,  
 "Conservation score of TFBS"= tableConservationScoreOfTFBS,  
 "Distance to closest cluster"=tableDistanceToArxCluster,  
 "Conservation score of ARX cluster"=tableConservationScoreOfCluster)  
} else {  
 ArxTable3<- rbind(ArxTable3,cbind(  
 "Gene"=selectedGene[1,8],  
 "Chromosome" = as.character(selectedGene[1,1]),  
 "Start"= min(selectedGene[2]),  
 "End"=max(selectedGene[3]),   
 "Distance to closest Arx TFBS"=tableDistanceToArxTFBS,  
 "Conservation score of TFBS"= tableConservationScoreOfTFBS,  
 "Distance to closest cluster"=tableDistanceToArxCluster,  
 "Conservation score of ARX cluster"=tableConservationScoreOfCluster))  
  
}  
x<<- x+1  
print(x)   
}

Intially i loaded all the data from my shiny app including the conservation scores, the Dataframes for the ARX motifs converting everything to a dataframe because i prefer to work with them as they are more readily manipulatible. In addition to this, i have gotten the gene ids for the each UCSC gene id and added the coloumn in giving me a dataframe for each differentially expressed gene with its gene symbol name. This has shown me that the UCSC table includded trancriptome variants hence i have to develop a method for identifying each gene's longest transcript. This is where i devleoped a dataframe that is called "geneNames" which i use to select for all transcript variants with the same gene ID, called "selectedGene". From this table of a specific gene, all variants are shown hence i select for the longest variant to identify the closest ARX transcription Factor motif, cluster and conservation scores. To attain the closest Arx Motif to a gene, i simply substract the starting motif from the starting sites of eac ARX motif and take the absoulte vlaues and select for the samllest value. I did this aswwell for the Arx Clusters. I utilised this to attain the conservation scores for these sites by taking the coordinates for the closest Arx motif and substracting it from the coordinates for the conservation scores and taking the conservation score for the closest site. All of this data was then combined into a table. If the table did not exist the table would be intilaised by cbind giving the collumn names. If it was already made, hence the first row is created the second chunk of command after "else" would be executed adding a new row for a new gene. This would be repeated for each gene ID until all of them have been iterated through.