Analysis

January 9, 2019

```
In [2]: library(Homo.sapiens)
        library(taRifx) ## Removes factors
        library(scales)
        library(SchramekLOH)
        library(gplots)
        library(IdeoViz)
        library(reshape)
```

```
In [917]: detach("package:SchramekLOH", unload=TRUE)
          require(SchramekLOH)
```

```
Loading required package: SchramekLOH

Attaching package: 'SchramekLOH'

The following objects are masked _by_ '.GlobalEnv':

    compTwoGenes, df.ex, doTheChi, generateIgvAttributes, getSegIQR,
    mapIds, plotIdeoGene, seg, snp6

The following object is masked from 'package:OrganismDbi':

    mapIds

The following object is masked from 'package:AnnotationDbi':

    mapIds
```

1 Setup

Loading in all the precomputed data files

```
In [3]: data("birdseed")  # df.bs, mapping.cov, mapping, ref.probe.ord
        data("expr.mapping")  # expr.mapping
        data("gaf")       # gaf
```

```
data("geneExpr")   # df.ex
data("mapping")    # mapping
data("snp6")       # snp6
data("Affyseg")    # affyseg
data("TCGAseg")    # seg
```

Setting up some of the paths

```
In [4]: #### Load in Mappings ####
        # Bed file obtained form http://www.affymetrix.com/Auth/analysis/downloads/
        pdir <- '/Users/rquevedo/Onedrive/PughLab/Schramek_notch/IGV_segs/TCGA_hnsc
        #outdir <- '/Users/rquevedo/Onedrive/PughLab/Schramek_notch/IGV_segs/TCGA_h
        outdir <- '/Users/rquevedo/Onedrive/PughLab/Schramek_notch/loh_analysis'
        tmpdir <- file.path(outdir, "tmp")
        plotsdir <- file.path(outdir, "plots")
        dir.create(tmpdir, recursive = TRUE, showWarnings = FALSE)
        dir.create(plotsdir, recursive = TRUE, showWarnings = FALSE)

        ref <- file.path(pdir, "ref")


        goi <- c("NOTCH1", "NOTCH2", "NOTCH3", "NOTCH4",
                 "JAG1", "JAG2", "ADAM10", "AJUBA")
```

Determine whether to use the TCGA Segs or the Affymetrix SNP6 Segs (same that the bird-seeds originate from)

```
In [5]: #### Read in Birdseed + Segs ####
        use.affy <- FALSE
        if(use.affy) seg <- affyseg
```

```
In [6]: head(seg)
```

ID	chrom	loc.start	loc.end	num.mark	seg.mean
TCGA-CN-6010-01	1	3218610	70988682	38435	-0.0543
TCGA-CN-6010-01	1	70990192	71001138	11	0.1953
TCGA-CN-6010-01	1	71002192	104005432	19689	-0.2485
TCGA-CN-6010-01	1	104009909	104613056	160	-0.0488
TCGA-CN-6010-01	1	104613622	149881398	9504	-0.2554
TCGA-CN-6010-01	1	149882014	247813706	61340	0.1959

2 Preprocess

2.1 Ordering all the data structures

```
In [7]: #### Chromosome order datasets ####
        seg$chrom <- gsub("(chr).*\\1", "\\1", paste0("chr", seg$chrom))
        seg.ids <- split(seg, f=seg$ID)
        seg.chr <- lapply(seg.ids, function(seg.i){
```

```
        seg.tmp <- split(seg.i, f=seg.i$chrom)
        chrom.ord <- match(paste0("chr", c(1:22)), names(seg.tmp))
        seg.tmp[chrom.ord]
      })
```

```
In [8]: head(seg.chr[['BALMS_p_TCGAb54and67_SNP_N_GenomeWideSNP_6_A03_730402']][['c
```

```
NULL
```

```
In [9]: #### Map Probesets to Genomic Loci ####
        if(exists("ref.probe.ord")){
            snp6 <- snp6[match(ref.probe.ord, snp6$V4),]
            snp6.ord <- snp6[,c(4, 1:3)]
            colnames(snp6.ord) <- c("probeset_id", "chrom", "start", "end")

            snp6.chr <- split(snp6.ord, f=snp6.ord$chrom)
            bs.chr <- split(as.data.frame(df.bs), snp6.ord$chrom)
            goi.df <- getGeneLoci(goi)
            goi.chr <- split(goi.df, f=goi.df$chr)

            chrom.ord <- match(paste0("chr", c(1:22, "X", "Y")), names(snp6.chr))
            snp6.chr <- snp6.chr[chrom.ord]
            bs.chr <- bs.chr[chrom.ord]
        }
```

2.2 Formatting the Gene expression data

```
In [10]: #### Expression analysis
         if(exists("df.ex")){
             ## Generate z-score per gene
             z <- function(x){ (x - mean(x, na.rm=TRUE)) / sd(x, na.rm=TRUE)}
             z.ex <- data.frame(t(apply(df.ex, 1, z)), stringsAsFactors=FALSE)

             ## Map Genes to Genomic Loci ##
             ord <- match(rownames(df.ex), gaf$V2)
             gaf.ord <- gaf[ord, c("V2", "V17")]
             gaf.ord$chr <- gsub(":.*", "", gaf.ord$V17)
             gaf.ord$start <- as.numeric(gsub("^.*:", "", gsub("-.*", "", gaf.ord$V
             gaf.ord$end <- as.numeric(gsub(":.*", "", gsub("^.*?-", "", gaf.ord$V1

             ## Reorder all the matrices into genomic loci numerical order
             chr.ord <- paste0("chr", c(1:22, "X", "Y"))
             gaf.ord <- gaf.ord[order(gaf.ord$start),]
             gaf.ord <- gaf.ord[order(match(gaf.ord$chr, chr.ord)), ]

             ord <- match(gaf.ord$V2, rownames(df.ex))
             df.ex <- df.ex[ord,]
             z.ex <- z.ex[ord,]
```

```
    ## Order the list by chromosomes
    gaf.chr <- split(gaf.ord, f=gaf.ord$chr)
    z.chr <- split(z.ex, f=gaf.ord$chr)

    chrom.ord <- match(paste0("chr", c(1:22, "X", "Y")), names(gaf.chr))
    gaf.chr <- gaf.chr[chrom.ord]
    z.chr <- z.chr[chrom.ord]
}
```

3 Analysis

3.1 Visualization and generate StdRes

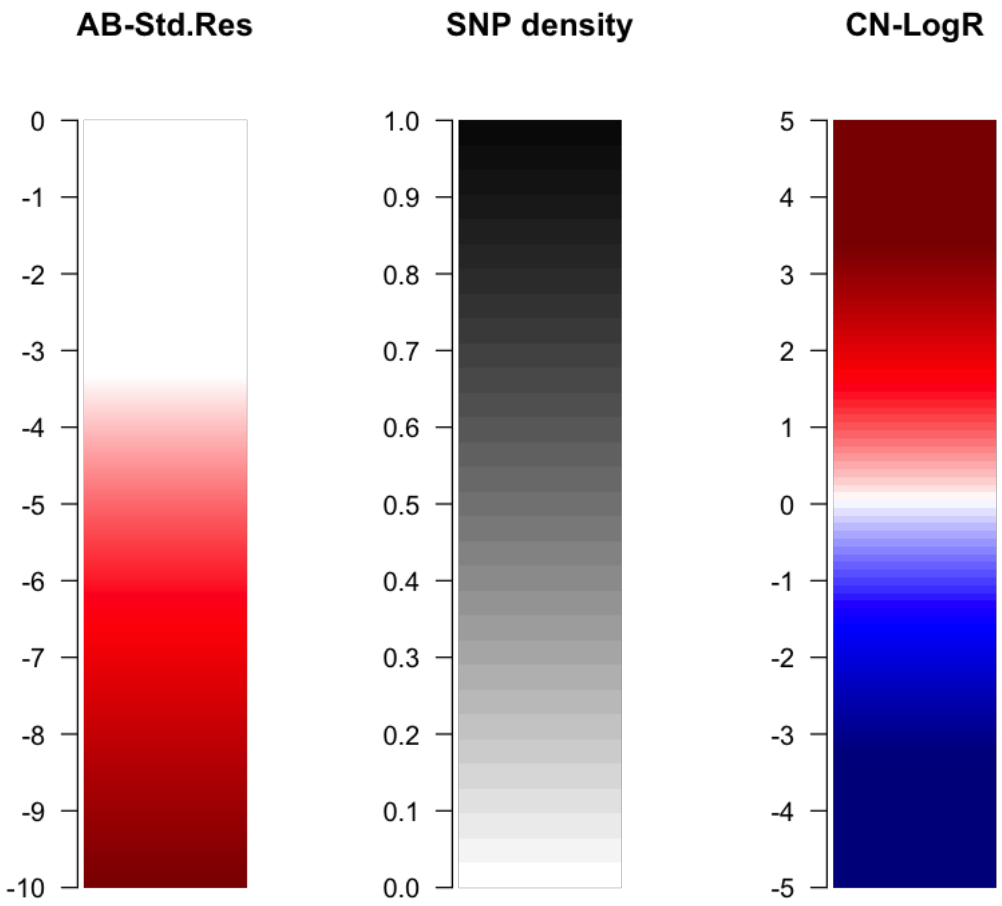
```
In [11]: library(gplots)
         library(RColorBrewer)
```

Setting up the colours that will be used for all visualizations

```
In [12]: rf <- colorRampPalette(c("white", "black"))
         pf <- colorRampPalette(c("white", "white", "red", "darkred"))
         cf <- colorRampPalette(c("blue4", "blue4", "blue", "white", "red", "darkre
         r <- rf(32); p <- pf(1000); cn <- cf(100)
         names(cn) <- round(seq(-4.9, 5.0, by=0.1),1)
```

Visualization for the colour bars and ranges used

```
In [52]: #pdf(file.path(plotsdir, "legend.pdf"), width=6)
         null <- split.screen(c(1,3))
         screen(1); color.bar(p, min=0, max=-10, title="AB-Std.Res")
         screen(2); color.bar(r, min=0, max=1, title="SNP density")
         screen(3); color.bar(cn, min=-5, max=5, title="CN-LogR")
         close.screen(all.screens=TRUE)
         #dev.off()
```




```
In [14]: bin.size <- 1000000
         spacer.param <- 20
```

```
In [743]: sample.stdres <- suppressWarnings(lapply(colnames(bs.chr[[1]]), mapAndPlo
                                  mapping.cov=mapping.cov, use.aff=use.affy,
                                  plotsdir=plotsdir, snp6.chr=snp6.chr,
                                  seg.chr=seg.chr, bs.chr=bs.chr, z.chr=z.chr,
                                  r=r, p=p, cn=cn, gen.plot=TRUE))
```

Summarize the standardized residuals data into data frames

```
In [744]: all.stdres <- lapply(sample.stdres, function(i) i[['all']])
          names(all.stdres) <- colnames(bs.chr[[1]])
```

```
## Reduce the gene to a single segment
sample.stdres.bkup <- sample.stdres
sample.stdres <- lapply(sample.stdres, function(i) {
  single.j <- sapply(split(i[['genes']], f=i[['genes']]$gene), function(j
    uniq.j <- apply(j, 2, unique)
    if(any(sapply(uniq.j[c('seg.start', 'seg.end', 'seg.mean')], length)
      uniq.j[['seg.start']] <- min(uniq.j[['seg.start']])
      uniq.j[['seg.end']] <- max(uniq.j[['seg.end']])
      uniq.j[['seg.mean']] <- mean(uniq.j[['seg.mean']])
    }

    sapply(uniq.j, function(x) x)
  })
  remove.factors(data.frame(t(single.j)))
})
names(sample.stdres) <- colnames(bs.chr[[1]])
```

In [16]:
```
head(all.stdres[['FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_D08_777884']
```

	chrom	start	end	stdres
[0,1e+06]	chr1	0	1000797	-2.5980762
(1e+06,2e+06]	chr1	1000797	2001595	-0.9878292
(2e+06,3e+06]	chr1	2001595	3002393	-0.3333333

In [17]:
```
head(sample.stdres[['FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_D08_77788
```

	gene	chr	gene.start	gene.end	bin.start	bin.end	seg.start	seg.end	seg.m
ADAM10	ADAM10	chr15	58888510	59042177	58814431	59621572	54814864	74252408	0.086
AJUBA	AJUBA	chr14	23440410	23451848	22652964	23466738	20501368	35724239	-0.264
JAG1	JAG1	chr20	10618332	10654694	10038090	11035720	455764	25990441	-0.284

In [15]:
```
#save(all.stdres, sample.stdres, sample.stdres.bkup, file=file.path(tmpdi
load(file.path(tmpdir, paste0("tmp", use.affy, ".RData")))
```

3.2 Generate 'Attributes' files for use in IGV

Summarize all the Standardized Residuals into LOH/Het value annotations for IGV visualization

In [18]:
```
stdres.thresh <- -5
attributes <- lapply(seq_along(sample.stdres), generateIgvAttributes,
                     sample.stdres=sample.stdres, mapping.cov=mapping.cov,
                     stdres.thresh=-5)
attributes <- do.call("rbind", attributes)
head(attributes, 5)
```

	TRACK_ID	ADAM10	AJUBA	JAG1	JAG2	NOTCH1	NOTCH2	NOTCH3	N
loh.val	TCGA-CN-6011-01	Het	Het	Het	LOH	Het	LOH	Het	H
loh.val1	TCGA-CN-6012-01	Het	LOH	LOH	LOH	LOH	LOH	Het	H
loh.val2	TCGA-CN-6016-01	Het	Het	LOH	LOH	LOH	Het	LOH	L
loh.val3	TCGA-CN-6018-01	Het	Het	LOH	LOH	LOH	Het	Het	L
loh.val4	TCGA-CN-6019-01	LOH	LOH	LOH	LOH	Het	Het	LOH	L

Write and save the data structures

```
In [749]: write.table(attributes, file.path(outdir, "LOH_attributes.txt"),
                      sep="\t", quote=FALSE, col.names=TRUE, row.names=FALSE)
          save(sample.stdres, attributes, mapping.cov, file=file.path(outdir, "gene
```

3.3 Generate contigency tables and test for significance

```
In [19]: segdir <- '/Users/rquevedo/Onedrive/PughLab/Schramek_notch/IGV_segs/TCGA_h
```

```
In [20]: mut.attr <- read.table(file.path(segdir, "hnsc_tcga_attributes.txt"),
                                sep="\t", header=TRUE, stringsAsFactors = FALSE,
                                check.names = FALSE)
```

Initialize the contigency table to be used for quick reference later

```
In [21]: all.ctbl <- doTheChi(ctbl=NULL, attributes, mut.attr,
                               tbl.idx=c(1,3), gene='ADAM10', mut='ADAM10_CNA')
         ctbl <- all.ctbl[['ctbl']]
```

Available mutations to compare for LOH, where "ADAM10" actually means "ADAM10_LOH"

```
In [53]: print(names(ctbl))
```

```
 [1] "NOTCH1_CNA"    "NOTCH1_MUT"    "NOTCH1_FUSION" "NOTCH2_CNA"
 [5] "NOTCH2_MUT"    "NOTCH2_FUSION" "NOTCH3_CNA"    "NOTCH3_MUT"
 [9] "NOTCH3_FUSION" "NOTCH4_CNA"    "NOTCH4_MUT"    "NOTCH4_FUSION"
[13] "JAG1_CNA"      "JAG1_MUT"      "JAG1_FUSION"   "JAG2_CNA"
[17] "JAG2_MUT"      "JAG2_FUSION"   "DLL1_CNA"      "DLL1_MUT"
[21] "DLL1_FUSION"   "ADAM10_CNA"    "ADAM10_MUT"    "ADAM10_FUSION"
[25] "AJUBA_CNA"     "AJUBA_MUT"     "AJUBA_FUSION"  "ADAM10"
[29] "AJUBA"         "JAG1"          "JAG2"          "NOTCH1"
[33] "NOTCH2"        "NOTCH3"        "NOTCH4"
```

Run the chi-squared analysis on the samples that are of interest

```
In [55]: print(doTheChi(ctbl=ctbl, tbl.idx=c(1,3), gene='ADAM10', mut='ADAM10_CNA'))
         print(doTheChi(ctbl=ctbl, tbl.idx=c(1,2), gene='NOTCH1', mut='NOTCH1_CNA'))
         print(doTheChi(ctbl=ctbl, tbl.idx=c(1,3), gene='AJUBA', mut='AJUBA_CNA'))
```

```
$Contigency
     j
i     HETLOSS HOMDEL no_alteration
  Het      53      0           183
  LOH      58      1           211

$p
[1] 0.8926257
```

```
$Std.Res
     j
i         HETLOSS no_alteration
  Het  0.2426701    -0.2426701
  LOH -0.2426701     0.2426701

$Contigency
     j
i     HOMDEL no_alteration
  Het      4           245
  LOH      7           250

$p
[1] 0.5777054

$Std.Res
     j
i          HOMDEL no_alteration
  Het -0.8616207     0.8616207
  LOH  0.8616207    -0.8616207

$Contigency
     j
i     HETLOSS HOMDEL no_alteration
  Het      24      1           126
  LOH      51      0           304

$p
[1] 0.7377338

$Std.Res
     j
i         HETLOSS no_alteration
  Het  0.4717808    -0.4717808
  LOH -0.4717808     0.4717808
```

3.4 StdRes plots

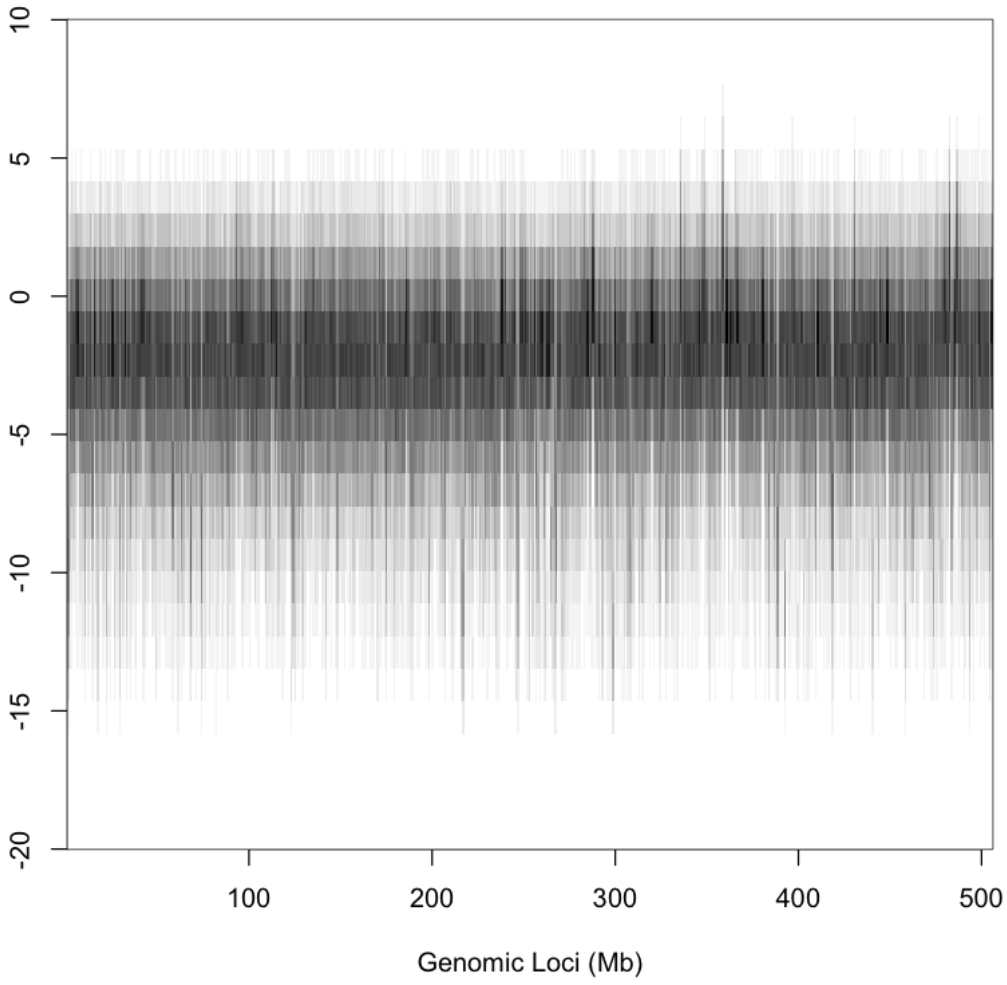
```
In [24]: #pdf(file.path(outdir, "stdres.pdf"), width=20)
         m.stdres <- do.call("rbind", all.stdres)
         m.stdres$ID <- gsub("\\..*", "", rownames(m.stdres))
         m.stdres <- melt(m.stdres, id.vars = "ID", measure.vars = "stdres")
         m.stdres$IDidx <- match(m.stdres$ID, unique(m.stdres$ID))
         h2 <- hist2d(m.stdres[,c("IDidx", "value")],
                      nbins=c(length(unique(m.stdres$ID)), 30),
                      col=r, ylim=c(-20,10),
```

```
                    xlab = "Genomic Loci (Mb)")
        #dev.off()
```

```
Attaching package: 'reshape'

The following objects are masked from 'package:S4Vectors':

    expand, rename
```



3.5 Ideogram of HetLoss over Gene of Interest

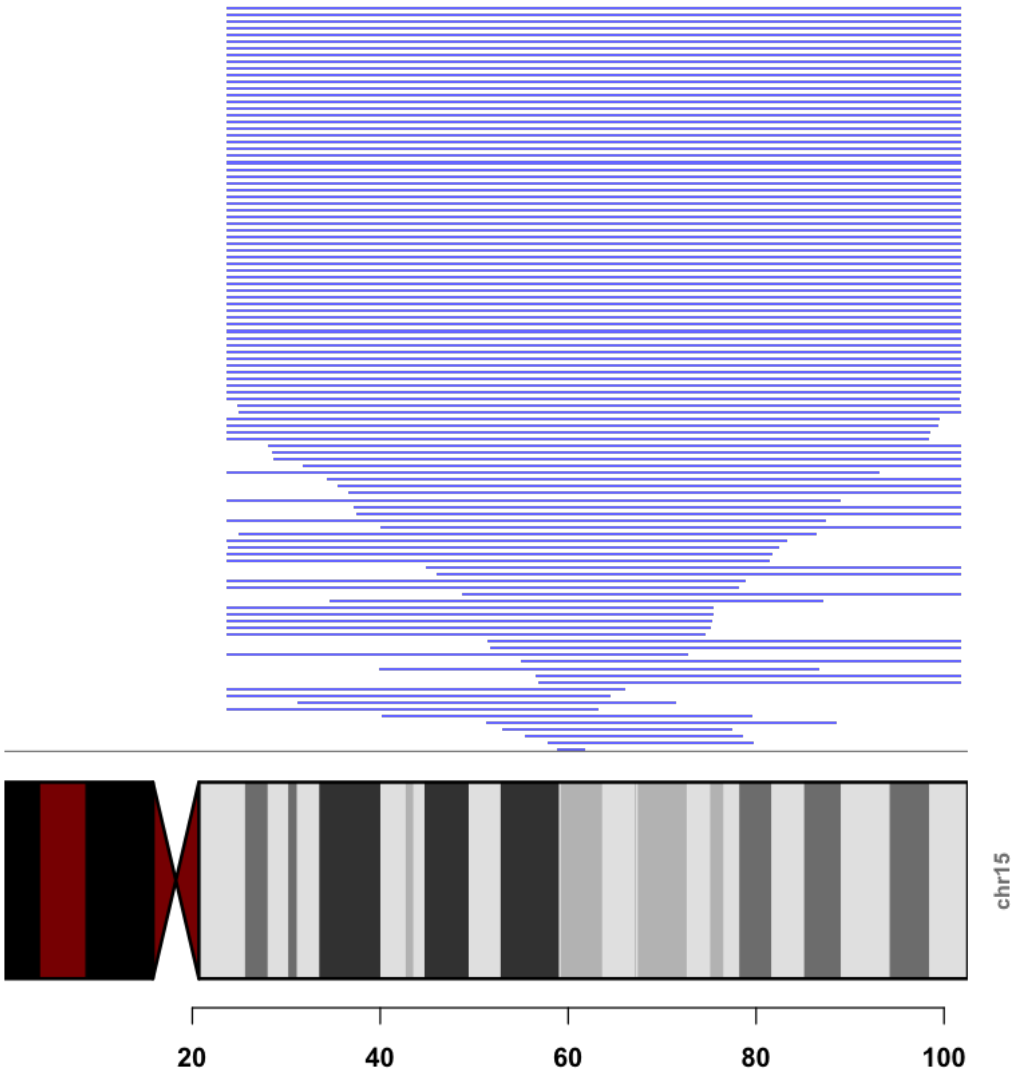
Set up the Ideogram hg19 reference

```
In [27]: ideo_hg19 <- getIdeo("hg19")
```

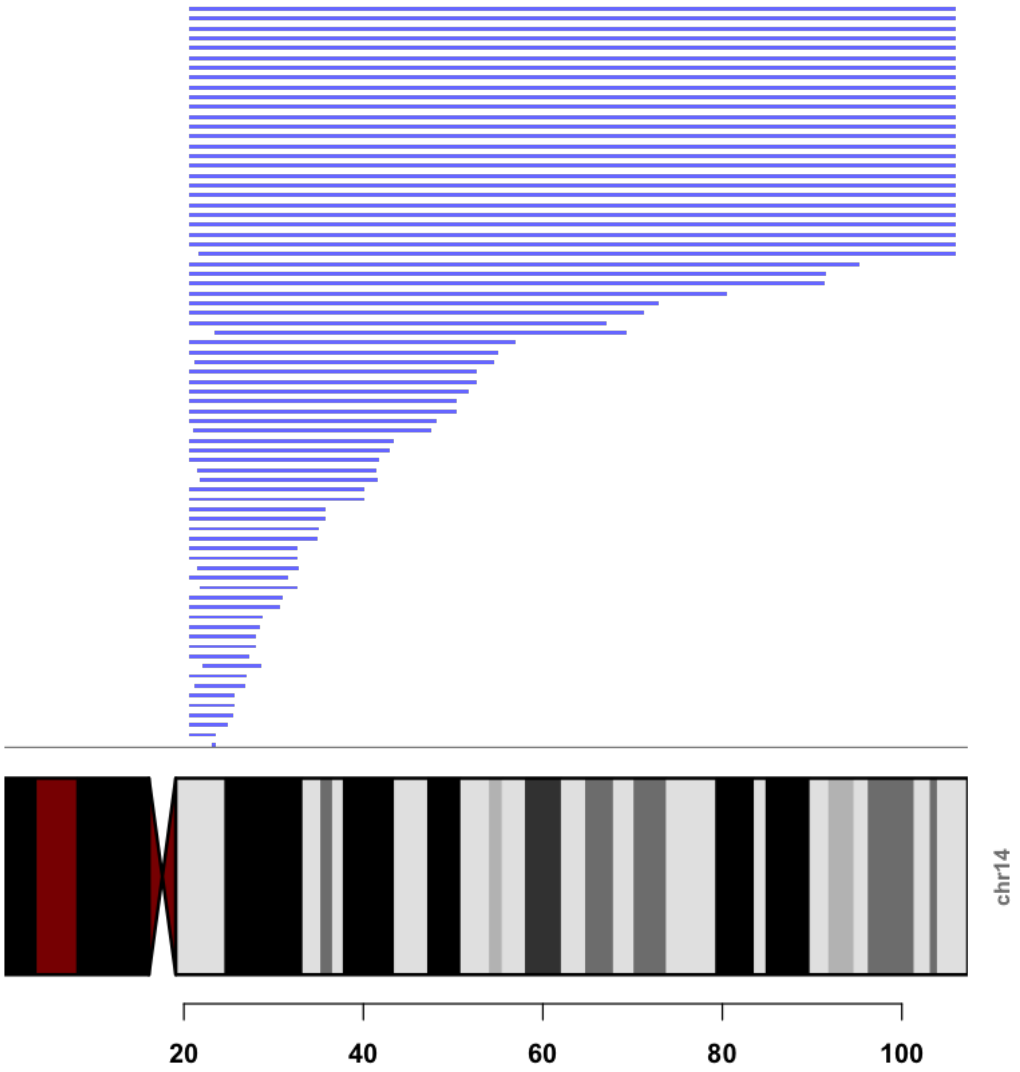
Visualize each gene of interest in their ideogram

```
In [30]: for(gene in rownames(sample.stdres[[1]])){
           pdf(file.path(outdir, paste0("ideo_", gene, ".pdf")), height=20)
           plotIdeoGene(ideo_hg19, sample.stdres, gene, thresh=-0.1)
           dev.off()
         }
```

```
In [28]: plotIdeoGene(ideo_hg19, sample.stdres, 'ADAM10', thresh=-0.1)
```



```
In [29]: plotIdeoGene(ideo_hg19, sample.stdres, 'AJUBA', thresh=-0.1)
```



3.6 Expression of AJUBA Mutants

```
In [31]: gene <- 'AJUBA'
         gene.alt <- 'JUB' # Alternate name of AJUBA
```

Grab gene expression for all samples and assign them to their proper data structures. Then, separate the samples based on their mutations

```
In [37]: .mutBoxplot <- function(ex.by.mut){
             ex.id <- colnames(ex.by.mut[[1]])[3]
```

```
            boxplot(lapply(ex.by.mut, function(x) x[,ex.id ]), outline=FALSE)

            x <- sapply(seq_along(ex.by.mut), function(x){
                ex <- ex.by.mut[[x]]
                points(x=rep(x, nrow(ex)), y=ex[, ex.id],
                       pch=16, col=alpha("black", rescale(as.numeric(ex[,4]), to=c
                       cex=rescale(as.numeric(ex[,4]), to=c(1,2)))
            })
            NULL
        }
```

```
In [56]: ex.by.mut <- parseIdsByMutation('AJUBA', getGeneExp('JUB'),
                                          seg.ids=seg.ids, lo.q=0.1, hi.q=0.9)
         null <- .mutBoxplot(ex.by.mut)
         lapply(ex.by.mut, head, 3)
```

```
Warning message in min(x):
"no non-missing arguments to min; returning Inf"Warning message in max(x):
"no non-missing arguments to max; returning -Inf"Warning message in min(x):
"no non-missing arguments to min; returning Inf"Warning message in max(x):
"no non-missing arguments to max; returning -Inf"
```

$'NA'

TRACK_ID	Alt	JUB.84962	IQR.90.
TCGA-BA-4074-01	no_alteration	-1.0212289	0.4705
TCGA-BA-4076-01	no_alteration	-0.1099163	1.2144
TCGA-BA-4077-01	no_alteration	-0.3082486	1.0026

$mut

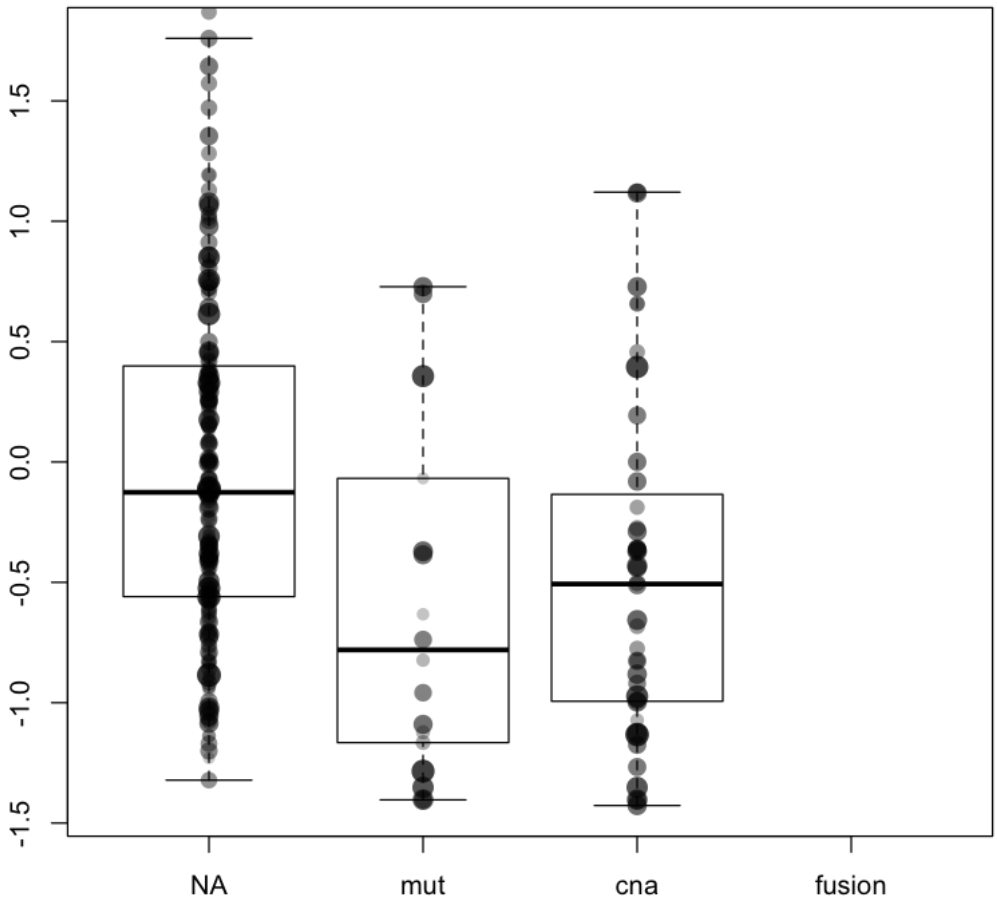
TRACK_ID	AJUBA_MUT	JUB.84962	IQR.90.
TCGA-BA-5556-01	R428Q I304Dfs*2	-0.6325205	0.1064
TCGA-BB-A5HY-01	S230Ffs*76	NA	0.68
TCGA-BB-A6UO-01	V264Lfs*2	NA	0.71

$cna

TRACK_ID	AJUBA_CNA	JUB.84962	IQR.90.
TCGA-4P-AA8J-01	HETLOSS	NA	0.2877
TCGA-BA-4075-01	HETLOSS	-0.2885436	0.6765
TCGA-BA-4078-01	HETLOSS	-0.6562828	0.7544

$fusion

TRACK_ID	AJUBA_FUSION	JUB.84962	IQR.90.




```
In [39]: ord <- order(ex.by.mut[['mut']][,3], decreasing=TRUE)
         ex.by.mut[['mut']][ord,]
```

	TRACK_ID	AJUBA_MUT	JUB.84962	IQR.90.
13	TCGA-CV-6003-01	T337_C341del	2.65706121	0.7254
11	TCGA-CR-6493-01	C406S	0.72805934	0.6963
25	TCGA-CX-7082-01	H360Y	0.69803062	0.6209
6	TCGA-CN-6997-01	R50Efs*192	0.35677236	0.9341
15	TCGA-CV-7099-01	H423Y	-0.06802904	0.068
10	TCGA-CR-6491-01	Q103*	-0.37074236	0.7302
18	TCGA-CV-7424-01	D108Rfs*16	-0.38532999	0.6718
1	TCGA-BA-5556-01	R428Q I304Dfs*2	-0.63252046	0.1064
14	TCGA-CV-6950-01	Q353*	-0.73796721	0.5662
4	TCGA-CN-4738-01	A351Qfs*39	-0.82308846	0.1903
30	TCGA-HD-7753-01	R324Gfs*84	-0.95843514	0.5462
29	TCGA-DQ-7588-01	R428*	-1.08991576	0.6774
7	TCGA-CN-6998-01	N433I	-1.12249891	0.2421
5	TCGA-CN-6018-01	C270Wfs*10	-1.16581195	0.2961
17	TCGA-CV-7418-01	E279*	-1.28425163	1.0081
12	TCGA-CV-5435-01	L280Afs*26	-1.35174099	0.8639
16	TCGA-CV-7177-01	E305Sfs*105	-1.40261590	0.8049
19	TCGA-CV-7432-01	R293*	-1.40352314	0.5513
2	TCGA-BB-A5HY-01	S230Ffs*76	NA	0.68
3	TCGA-BB-A6UO-01	V264Lfs*2	NA	0.71
8	TCGA-CN-A63V-01	C426Y	NA	0.2171
9	TCGA-CQ-5327-01	Q76*	NA	0.0558
20	TCGA-CV-A45Q-01	Q103* S302Wfs*108	NA	0.2981
21	TCGA-CV-A460-01	R293Lfs*13	NA	0.3748
22	TCGA-CV-A463-01	R371*	NA	0.3251
23	TCGA-CV-A6JM-01	E253Gfs*53	NA	0.6672
24	TCGA-CV-A6K2-01	G370R T361Sfs*48	NA	0.0437
26	TCGA-D6-A6EN-01	I339Nfs*50	NA	0.043
27	TCGA-D6-A6EP-01	X414_splice	NA	1.0883
28	TCGA-D6-A74Q-01	Q362*	NA	0.6643
31	TCGA-P3-A5QF-01	E507Kfs*54	NA	0.8036
32	TCGA-UF-A71D-01	X414_splice	NA	0.6532
33	TCGA-UF-A7JF-01	R77Pfs*164	NA	0.6197

3.7 Linear regression of HETLOSS Regions

```
In [40]: #gene <- 'ADAM10'
         gene <- 'AJUBA'
         mut <- paste0(gene, "_CNA")
         rge <- 5
```

Combines the StdRes of surrounding 1Mb bins to look for stable LOH regions rather than just artifactual regions that only span 1Mb

```
In [41]: range.stdres <- sapply(names(sample.stdres), aggregateStdRes,
                                range=rge, gene=gene, sample.stdres=sample.stdres,
         range.stdres <- data.frame(t(range.stdres), stringsAsFactors=FALSE)
```

Identifying the HETLOSS samples

```
In [42]: hetloss.idx <- which(mut.attr[,mut]  == 'HETLOSS')
         homloss.idx <- which(mut.attr[,mut]  == 'HOMDEL')
         mut.ids <- mut.attr[c(hetloss.idx, homloss.idx), 'TRACK_ID', drop=TRUE]
         mapped.ids <- mapIds(mut.ids, mapping.cov, in.type='tcga', out.type='affy'
```

Setting a boolean tag for those samples annotated as "HETLOSS"

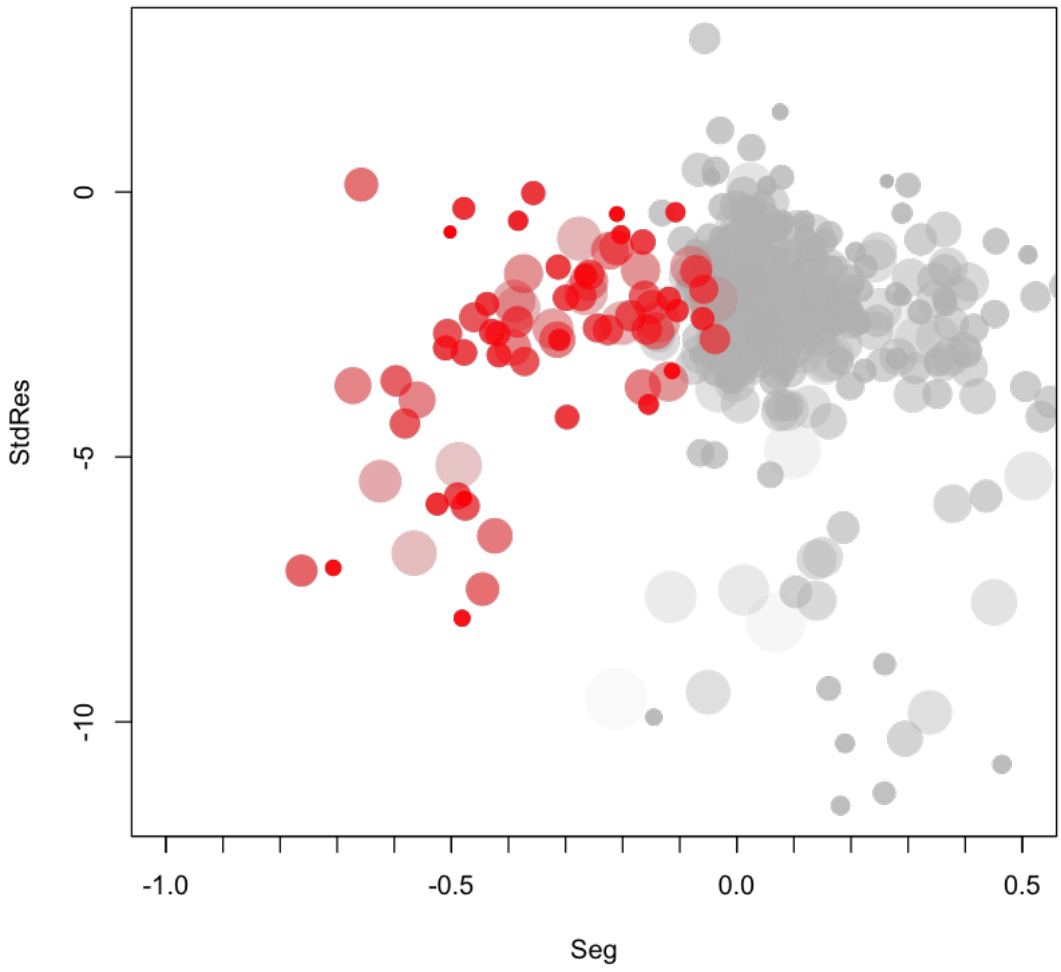
```
In [43]: range.stdres$HETLOSS <- FALSE
         range.stdres[which(rownames(range.stdres) %in% mapped.ids),]$HETLOSS <- TR

In [44]: head(range.stdres)
```

	mean	sd	seg	H
FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_D08_777884	-1.9134714	3.643712	-0.2648	TR
FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_E01_777860	-4.0880991	3.115277	0.7224	FA
FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_E08_777904	-1.8242195	1.922516	0.0006	FA
FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_C04_777934	-2.5143447	2.839284	0.0766	FA
FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_F02_777880	-1.4897747	2.670505	-0.0713	TR
FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_D03_777770	-0.8075069	1.608582	-0.2031	TR

```
In [45]: with(range.stdres, plot(mean~seg, cex=sd, xlim=c(-1, 0.5),
                                 col=alpha('grey', 1-rescale(sd, to=c(0,1))),
                                 pch=16, ylab="StdRes", xlab="Seg"))
         axis(side=1, at = seq(-5, 4, by=0.1),
              labels = rep("", length(seq(-5, 4, by=0.1))), tick = TRUE)

         with(range.stdres[which(range.stdres$HETLOSS),],
              points(mean~seg, cex=sd,
                     col=alpha('red', 1-rescale(sd, to=c(0,1))),
                     pch=16))
```



For interest, there was one AJUBA case that had Homozygous Deletion, and one AJUBA case where the logRRatio was EXTREMELY low. I wanted to see the details of these cases

3.7.1 ——

```
In [46]: # Sample with a Homozygous Deletion
         print("Homozygous Deletion case")
         print(paste(c(gsub("01[AB]-.*", "01", mut.attr[homloss.idx,'TRACK_ID']),
                       mapIds(mut.attr[homloss.idx,'TRACK_ID'], mapping.cov)), coll
         range.stdres[mapIds(mut.attr[homloss.idx,'TRACK_ID'], mapping.cov),]



         # Sample with a -3.0 segment
```

```
        print("Low LRR case")
        id.x <- mapIds(rownames(range.stdres[which(range.stdres$seg < -1),]), mapp
                       in.type='affy', out.type='tcga')
        print(paste(c(gsub("01[AB]-.*", "01", id.x),
                      rownames(range.stdres[which(range.stdres$seg < -1),])), coll

        mut.attr[mut.attr$TRACK_ID == gsub("01[AB]-.*", "01", id.x), , drop=FALSE]
        range.stdres[mapIds(id.x , mapping.cov, in.type="tcga", out.type="affy"),]
```

```
[1] "Homozygous Deletion case"
[1] "TCGA-CR-7399-01, BISON_p_TCGA_164_173_175_SNP_N_GenomeWideSNP_6_E04_863392"
```

	mean	sd	seg
BISON_p_TCGA_164_173_175_SNP_N_GenomeWideSNP_6_E04_863392	0.1386529	2.814682	-0.6578

```
[1] "Low LRR case"
[1] "TCGA-CN-6988-01, MIRES_p_TCGA_151_SNP_N_GenomeWideSNP_6_G03_831548"
```

	TRACK_ID	NOTCH1_CNA	NOTCH1_MUT	NOTCH1_FUSION	NOTCH2_CNA	NO
323	TCGA-CN-6988-01	no_alteration	no_alteration	no_alteration	no_alteration	no

	mean	sd	seg	HETLOS
MIRES_p_TCGA_151_SNP_N_GenomeWideSNP_6_G03_831548	-10.39918	2.153658	-2.944	FALSE

3.8 Comparison of Seg/LOH between two genes

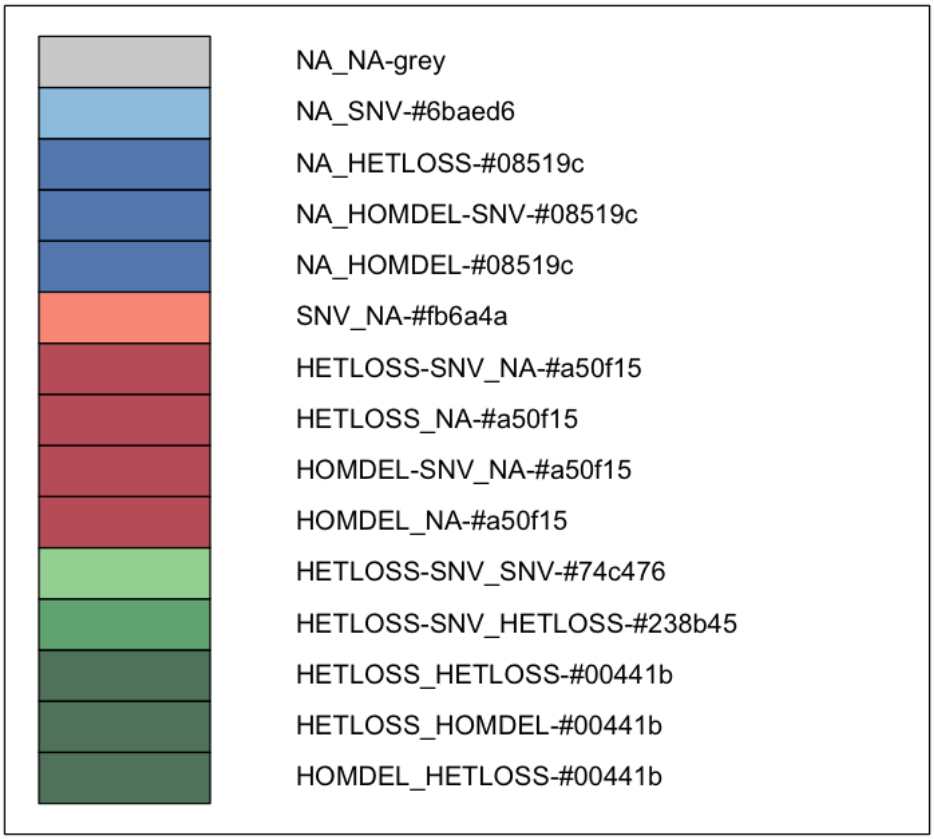
```
In [47]: gene.y <- 'ADAM10'
         gene.x <- 'AJUBA'
         rge <- 5
```

Create a colour schema for the unique ids (UID)

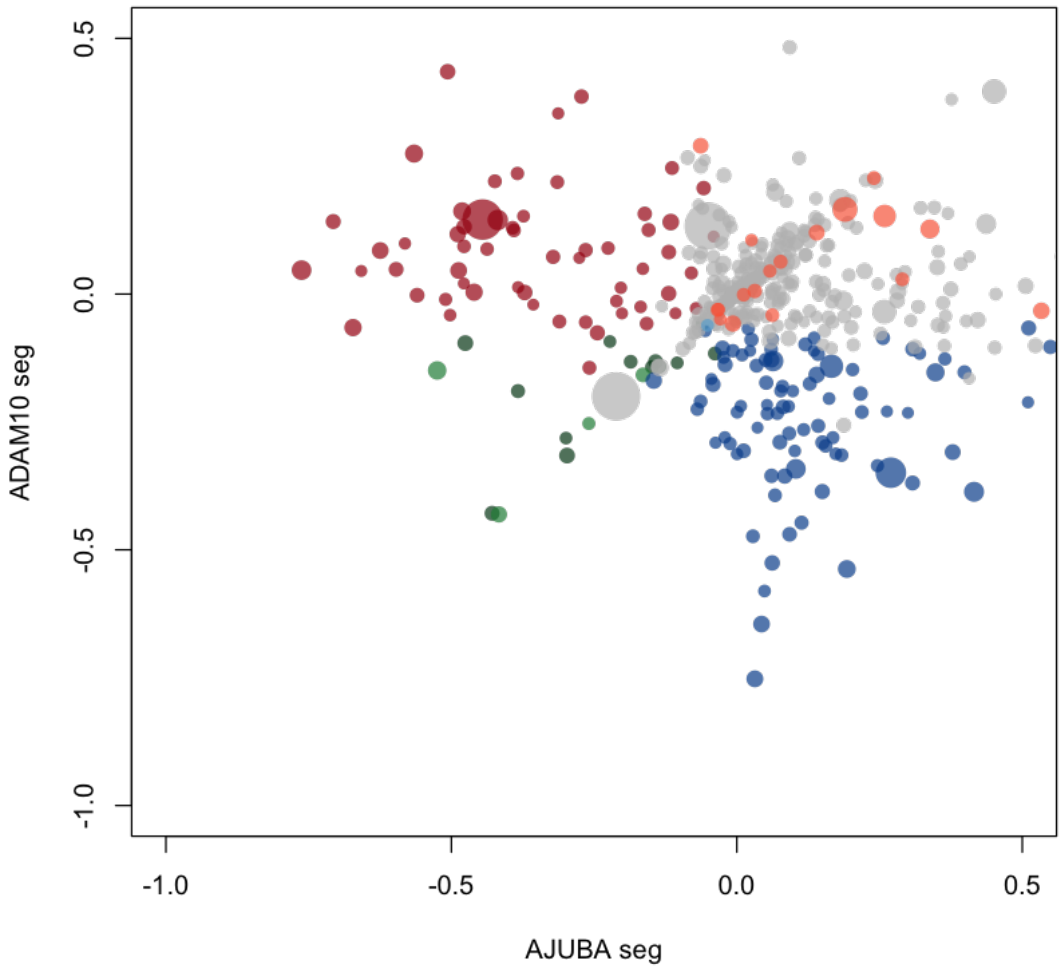
```
In [48]: col.df <- data.frame("UID"=c("HOMDEL_HETLOSS", "HETLOSS_HOMDEL", "HETLOSS_
                                     "HOMDEL_NA", "HOMDEL-SNV_NA", "HETLOSS_NA", "
                                     "NA_HOMDEL", "NA_HOMDEL-SNV", "NA_HETLOSS", "

                           "col"=c("#00441b", "#00441b", "#00441b", "#238b45", "
                                   "#a50f15", "#a50f15", "#a50f15", "#a50f15", "#
                                   "#08519c", "#08519c", "#08519c", "#6baed6", "c

In [49]: range.stdres.xyz <- compTwoGenes(gene.x, gene.y, col.df)
         range.stdres.xyz[which(range.stdres.xyz$UID == 'HETLOSS_HETLOSS'),]
```

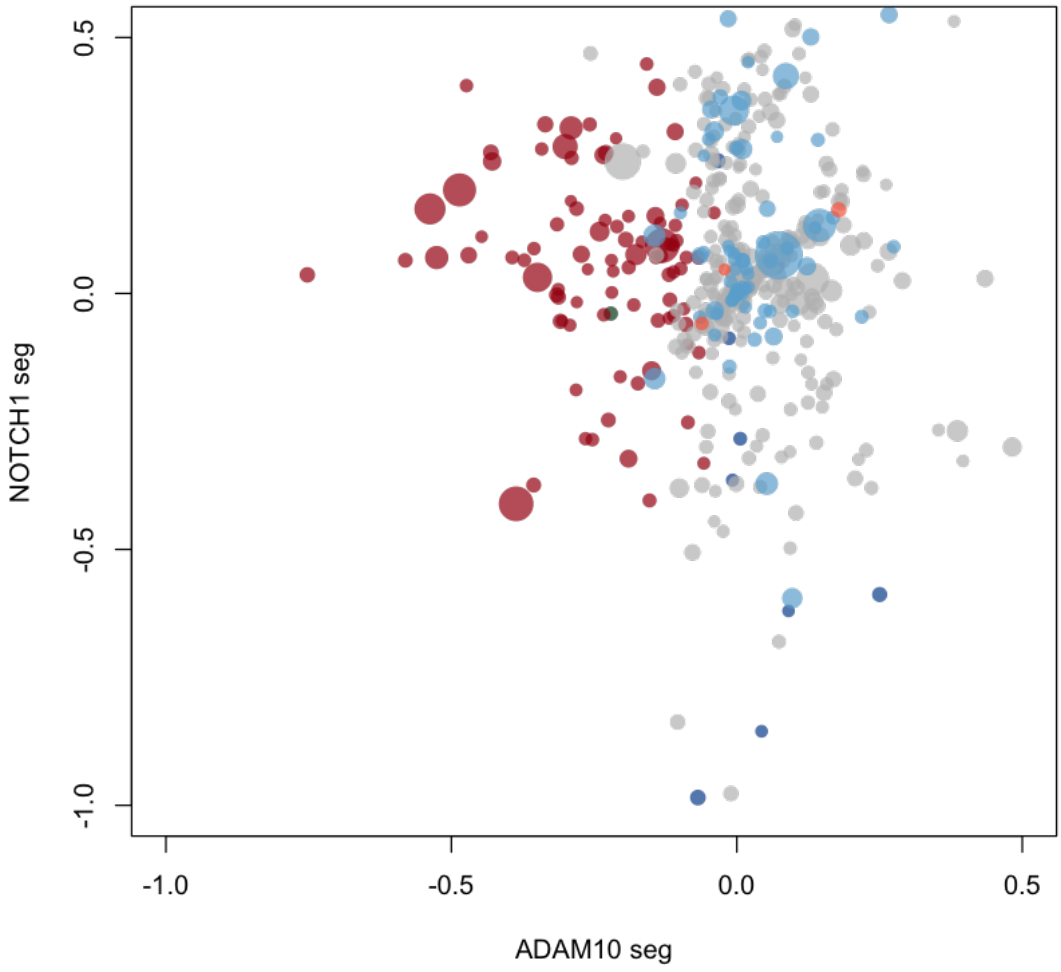


UID	TCGA_ID	TRACK_ID
HETLOSS_HETLOSS	TCGA-CQ-5332-01	FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_B0
HETLOSS_HETLOSS	TCGA-CV-5966-01	FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_A1
HETLOSS_HETLOSS	TCGA-H7-A76A-01	UNDID_p_TCGA_353_354_355_37_NSP_GenomeWideSNP_
HETLOSS_HETLOSS	TCGA-CV-7435-01	MAULS_p_TCGA_189_190_SNP_N_GenomeWideSNP_6_G
HETLOSS_HETLOSS	TCGA-MT-A7BN-01	UNDID_p_TCGA_353_354_355_37_NSP_GenomeWideSNP_
HETLOSS_HETLOSS	TCGA-D6-6826-01	MIRES_p_TCGA_151_SNP_N_GenomeWideSNP_6_D04_83
HETLOSS_HETLOSS	TCGA-UF-A7JA-01	CUTCH_p_TCGAb_355_37_52_NSP_GenomeWideSNP_6_C
HETLOSS_HETLOSS	TCGA-CQ-6221-01	CLUBS_p_TCGA_186_188_SNP_N_GenomeWideSNP_6_B0
HETLOSS_HETLOSS	TCGA-UF-A719-01	CUTCH_p_TCGAb_355_37_52_NSP_GenomeWideSNP_6_C
HETLOSS_HETLOSS	TCGA-CV-6940-01	MIRES_p_TCGA_151_SNP_N_GenomeWideSNP_6_F12_831
HETLOSS_HETLOSS	TCGA-CN-4725-01	BALMS_p_TCGAb54and67_SNP_N_GenomeWideSNP_6_G
HETLOSS_HETLOSS	TCGA-BA-5152-01	GROVE_p_TCGA_b145_153_SNP_N_GenomeWideSNP_6_C
HETLOSS_HETLOSS	TCGA-CQ-7072-01	RICES_p_TCGA_Batch_310_311_NSP_GenomeWideSNP_6_



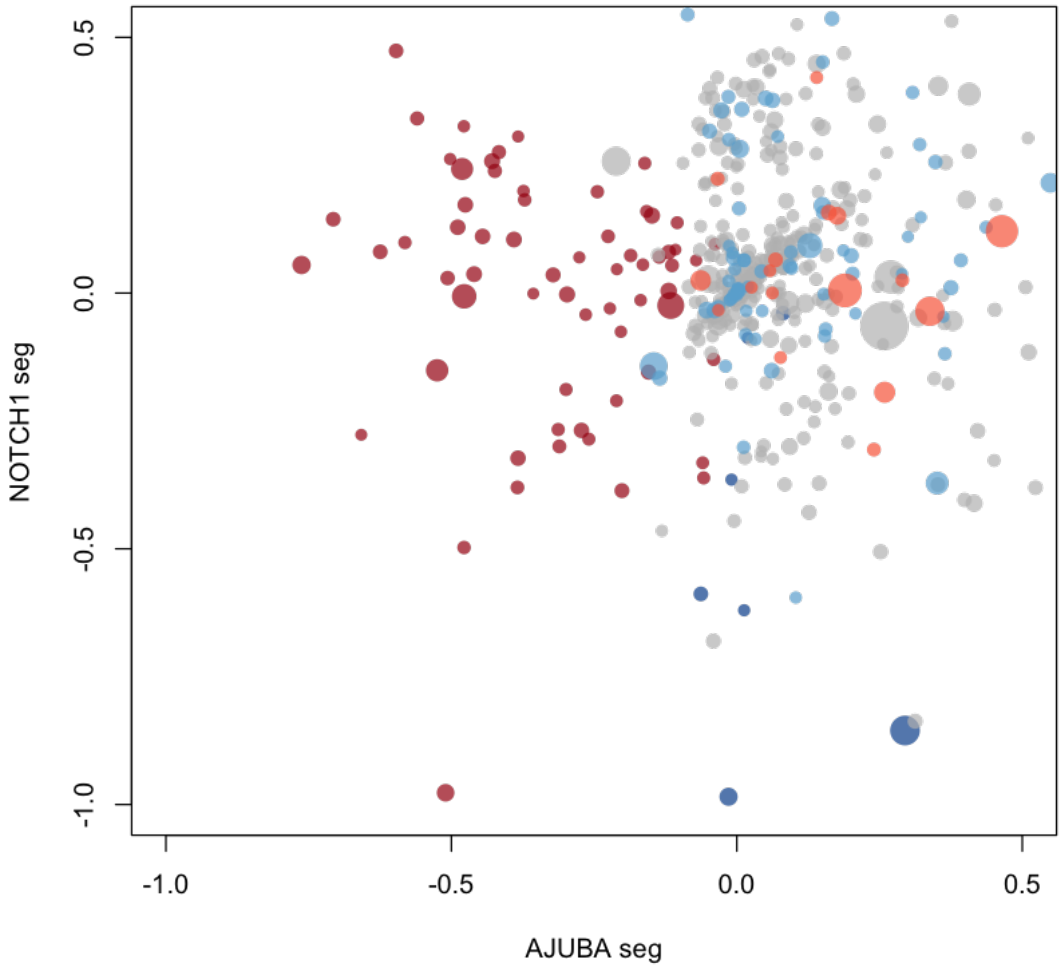

```
In [50]: range.stdres.xyz <- compTwoGenes('ADAM10', 'NOTCH1', col.df, cex.type = 'x
         head(range.stdres.xyz[which(range.stdres.xyz$seg.y < -0.3),], 10)
         #range.stdres.xyz[which(range.stdres.xyz$UID == 'HETLOSS_HETLOSS'),]
```

	UID	TCGA_ID	TRACK_ID
2	HETLOSS_HOMDEL-SNV	TCGA-QK-A8ZA-01	MESNE_p_TCGAb_401_02_03_04_05_N_Genom
10	HETLOSS_NA	TCGA-CV-6441-01	FISTS_p_TCGA_b107_121_SNP_N_GenomeWid
28	HETLOSS_NA	TCGA-KU-A66T-01	RICES_p_TCGA_Batch_310_311_NSP_GenomeW
33	HETLOSS_NA	TCGA-CV-7104-01	PEAKY_p_TCGA_b164_SNP_N_GenomeWideSN
50	HETLOSS_NA	TCGA-P3-A6T3-01	UNDID_p_TCGA_353_354_355_37_NSP_Genom
60	HETLOSS_NA	TCGA-CV-7435-01	MAULS_p_TCGA_189_190_SNP_N_GenomeWi
85	HETLOSS_NA	TCGA-UF-A719-01	CUTCH_p_TCGAb_355_37_52_NSP_GenomeWi
105	HETLOSS_SNV	TCGA-DQ-5631-01	GROVE_p_TCGA_b145_153_SNP_N_GenomeW
114	NA_HOMDEL	TCGA-BB-4224-01	BALMS_p_TCGAb54and67_SNP_N_GenomeWi
115	NA_HOMDEL	TCGA-P3-A5QA-01	LEGIT_p_TCGA_300_301_302_N_GenomeWideS




```
In [51]: range.stdres.xyz <- compTwoGenes('AJUBA', 'NOTCH1', col.df, cex.type = 'xy
         head(range.stdres.xyz[which(range.stdres.xyz$seg.y < -0.3),], 10)
```

	UID	TCGA_ID	TRACK_ID
3	HETLOSS_NA	TCGA-CR-5247-01	PEAKY_p_TCGA_b164_SNP_N_GenomeWideSNP_6_B09_863
4	HETLOSS_NA	TCGA-F7-7848-01	MAULS_p_TCGA_189_190_SNP_N_GenomeWideSNP_6_G10
7	HETLOSS_NA	TCGA-CN-4735-01	BALMS_p_TCGAb54and67_SNP_N_GenomeWideSNP_6_A03
17	HETLOSS_NA	TCGA-CV-7435-01	MAULS_p_TCGA_189_190_SNP_N_GenomeWideSNP_6_G12
18	HETLOSS_NA	TCGA-CN-A63U-01	RICES_p_TCGA_Batch_310_311_NSP_GenomeWideSNP_6_C
31	HETLOSS_NA	TCGA-UF-A719-01	CUTCH_p_TCGAb_355_37_52_NSP_GenomeWideSNP_6_G0
50	HETLOSS_NA	TCGA-HD-7229-01	PEAKY_p_TCGA_b164_SNP_N_GenomeWideSNP_6_B12_863
77	NA_HOMDEL	TCGA-BB-4224-01	BALMS_p_TCGAb54and67_SNP_N_GenomeWideSNP_6_A01
78	NA_HOMDEL	TCGA-CV-5434-01	FISTS_p_TCGA_b107_121_SNP_N_GenomeWideSNP_6_F03_
81	NA_HOMDEL	TCGA-P3-A5QA-01	LEGIT_p_TCGA_300_301_302_N_GenomeWideSNP_6_F06_13



In [ ]: