Using the DNaseI hypersensitivity data from encode in R

VJ Carey

November 1, 2011

1 Introduction

Annotation tracks from UCSC hg18 can be used with Bioconductor to help establish genomic contexts of events or alterations. The CD4-based hypersensitivity assays are collected in the structure rawCD4 in package encoDnaseI:

```
> library(encoDnaseI)
> data(rawCD4)
> rawCD4

hg18track (storageMode: lockedEnvironment)
assayData: 382713 features, 1 samples
  element names: dataVals
protocolData: none
phenoData: none
featureData
  featureNames: 1 2 ... 382713 (382713 total)
  fvarLabels: bin chrom chromStart chromEnd
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
  pubMedIds: 16791207
Annotation:
```

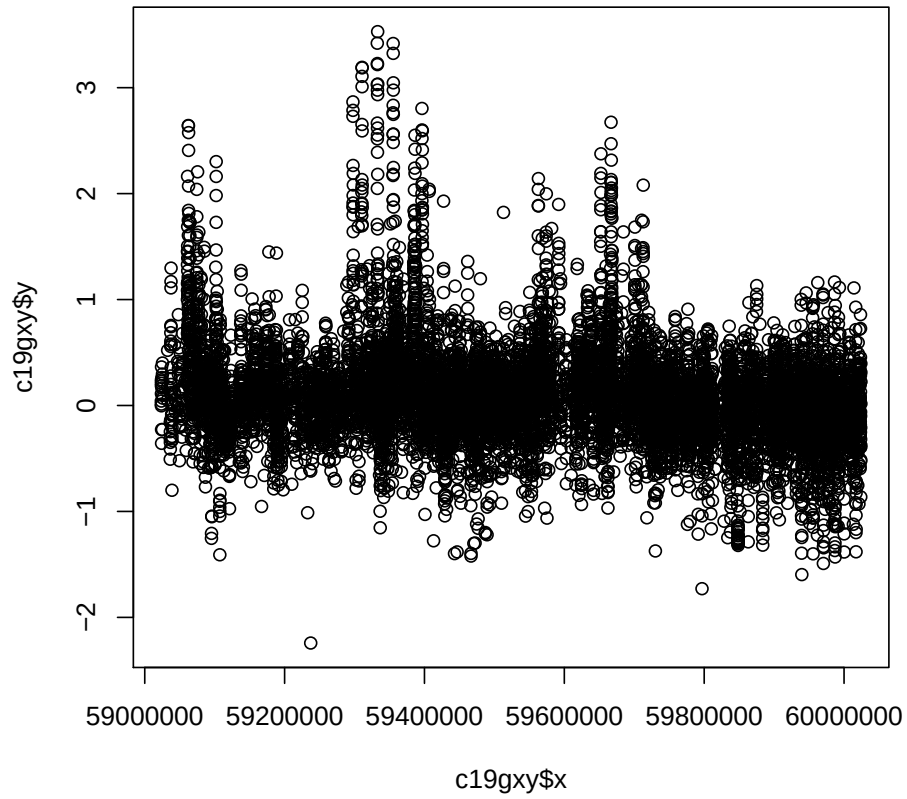
At present, we can subset the data by casting a chromosome number:

```
> c19g = rawCD4[ chrnum(19) ]
> c19g

hg18track (storageMode: lockedEnvironment)
assayData: 11158 features, 1 samples
  element names: dataVals
```

```
protocolData: none
phenoData: none
featureData
  featureNames: 129572 129573 ... 140729 (11158 total)
  fvarLabels: bin chrom chromStart chromEnd
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
  pubMedIds: 16791207
Annotation:
```

And we can get a trace of values along the chromosome:

```
> c19gxy = getTrkXY(c19g)
> plot(c19gxy)
```



2 Coupling the DnaseI series to genetics of gene expression

We would like to subset a racExSet from GGdata and look at snps that are in regions of high DNaseI sensitivity. Some infrastructure to help with this is:

```
> clipSnps = function( sms, chrn, lo, hi ) {
+  allp = getSnpLocs(sms)
+  allp = allp-allp[1] # relative
+  ok = allp >= lo & allp <= hi
+  thesm = smList(sms)[[1]]
+  rsn = colnames(thesm)
+  rid = rsn[which(ok)]
+  thesm = thesm[, rid, drop=FALSE]
+  nn = new.env()
+  tmp = list(thesm)
+  names(tmp) = as.character(chrn)
+  assign("smList", tmp, nn)
+  sms@smlEnv = nn
+  sms@activeSnpInds=which(ok)
+  sms
+ }
> rangeX = function(htrk) {
+  range(getTrkXY(htrk)$x)
+ }
```

So we get the information on expression and SNPs in chr19g and filter:

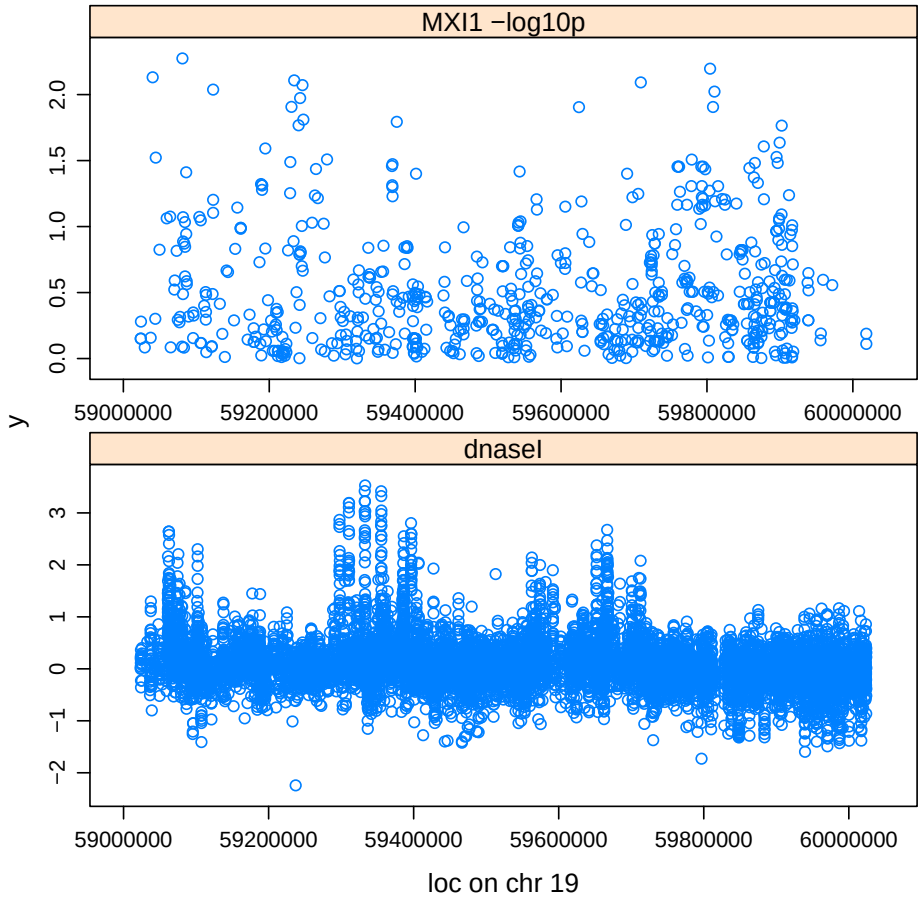
```
> library(GGtools)
> library(GGdata)
> h19 = getSS("GGdata", "19")
> rs19g = rangeX(c19g)
> library(SNPlocs.Hsapiens.dbSNP.20090506)
> c19l = getSNPlocs("chr19")
> h19locs = rbind(rsid=as.numeric(c19l[,"RefSNP_id"]), loc=as.numeric(c19l[,"loc"]))
> #h19locs = getSnpLocs(hmceuB36[chrnum(19),])[[1]]
> goodlocs = which(h19locs[2,] >= rs19g[1] & h19locs[2,] <= rs19g[2])
> h19rsn = paste("rs", h19locs[1,goodlocs], sep="")
> h19trim = h19[rsid(h19rsn),]
> #ok
> #c19gf = clipSnps( hmceuB36[chrnum(19),], chrnum(19), rs19g[1], rs19g[2] )
> #c19gf
```

A gene-specific screen can be computed as follows:

```
> oo = options()  # don't take warnings on multiple probes... caveat emptor
> options(warn=0)
> library(GGtools)
> showMethods("gwSnpTests")

Function: gwSnpTests (package GGtools)
sym="formula", sms="smlSet", cnum="cnumOrMissing", cs="missing"
sym="formula", sms="smlSet", cnum="snpdepth", cs="missing"

> smxi1 = gwSnpTests(genesym("MXI1")~1-1, h19trim, chrnum(19))

[1] "GI_18641367-A" "GI_18641367-I" "GI_18641369-I"

> smxi1

gwSnpScreenResult for gene  MXI1  [probe  GI_18641367-A ]

> #plot(smxi1)
> options(oo)
```

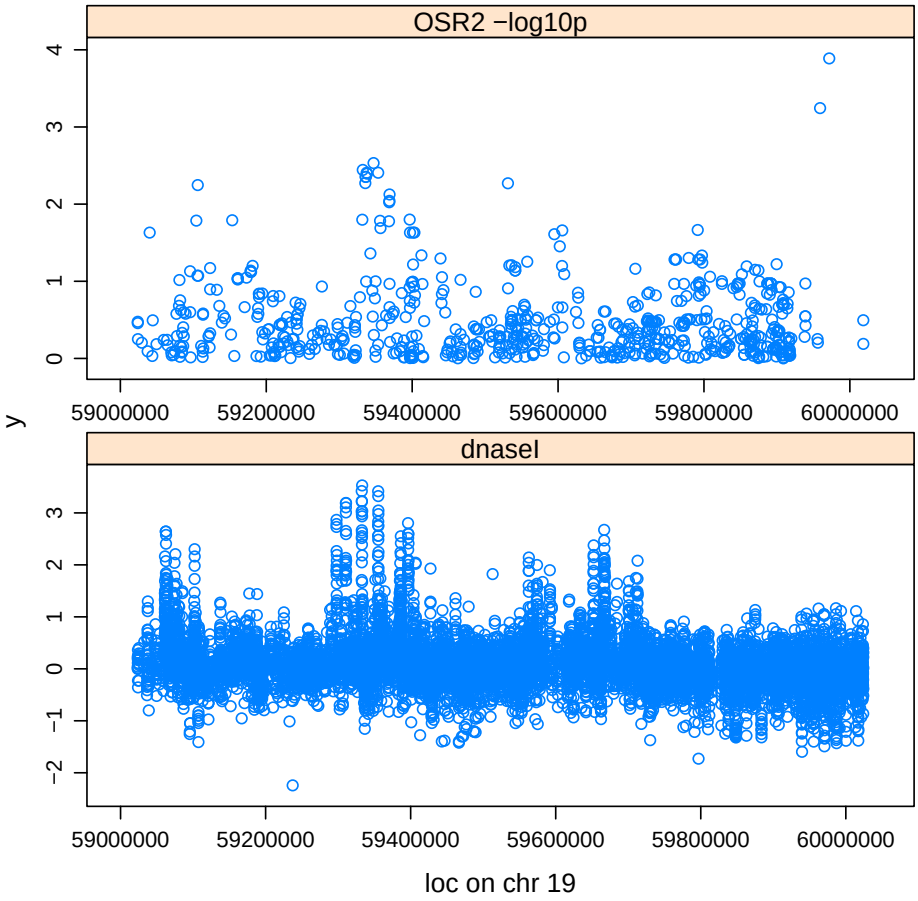
We'd like to look at the SNP screen results juxtaposed with the DnaseI results.

```
> print(juxtaPlot( c19g, smxi1, h19locs ))
```



Another example:

```
> oo = options()
> options(warn=0)
> sOSR2 = gwSnpTests(genesym("OSR2")~1-1, h19trim, chrnum(19))
> print(juxtaPlot( c19g, sOSR2, h19locs ))
> options(oo)
```



With these scores, we can find gene-snp combinations for which association is at least partly synchronized with DHS. Algorithms for systematically assessing synchronicity are in development.