

# Using the DNaseI hypersensitivity data from encode in R

VJ Carey

October 13, 2009

## 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
phenoData
  sampleNames: 1
  varLabels and varMetadata description: none
featureData
  featureNames: 1, 2, ..., 382713 (382713 total)
  fvarLabels and fvarMetadata description:
    bin: given bin
    chrom: chr..
    chromStart: numeric origin
    chromEnd: numeric close
experimentData: use 'experimentData(object)'
pubMedIds: 16791207
Annotation:
```

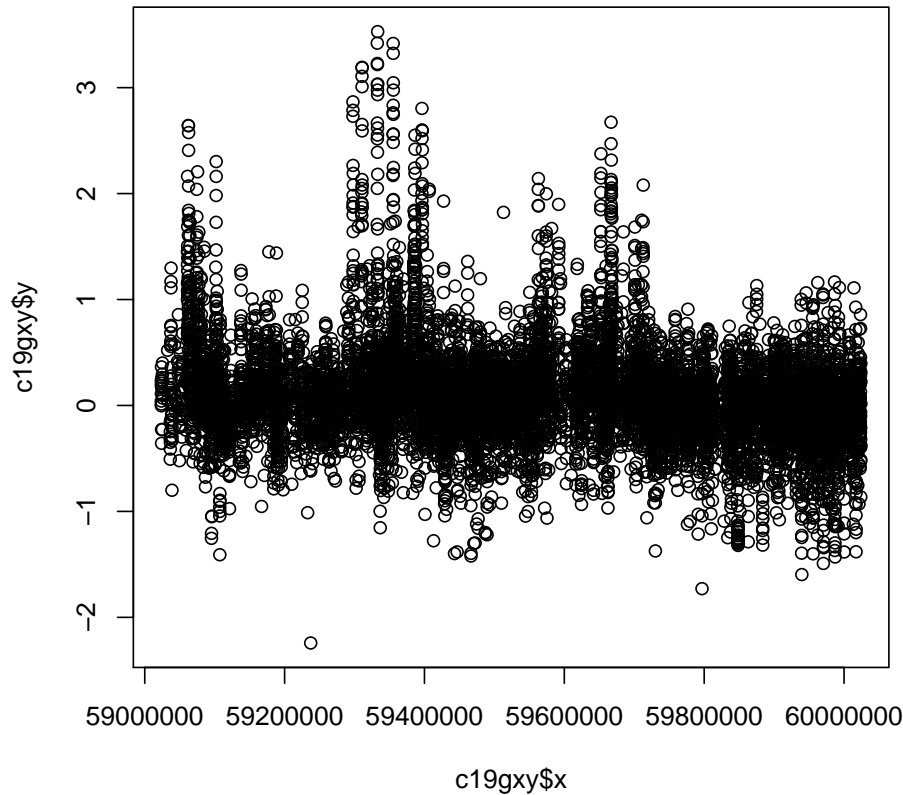
At present, we can subset the data by casting a chromosome number:

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

```
hg18track (storageMode: lockedEnvironment)
assayData: 11158 features, 1 samples
  element names: dataVals
phenoData
  sampleNames: 1
  varLabels and varMetadata description: none
featureData
  featureNames: 129572, 129573, ..., 140729 (11158 total)
  fvarLabels and fvarMetadata description:
    bin: given bin
    chrom: chr..
    chromStart: numeric origin
    chromEnd: numeric close
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]
+   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("smlList", 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)
> if (!exists("hmceuB36")) data(hmceuB36)
> rs19g = rangeX(c19g)
> h19 = hmceuB36[chrnum(19), ]
> 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), ]

```

A gene-specific screen can be computed as follows:

```

> oo = options()
> 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="chunksize"

sym="formula", sms="smlSet", cnum="snpdepth", cs="missing"

```

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

```

```

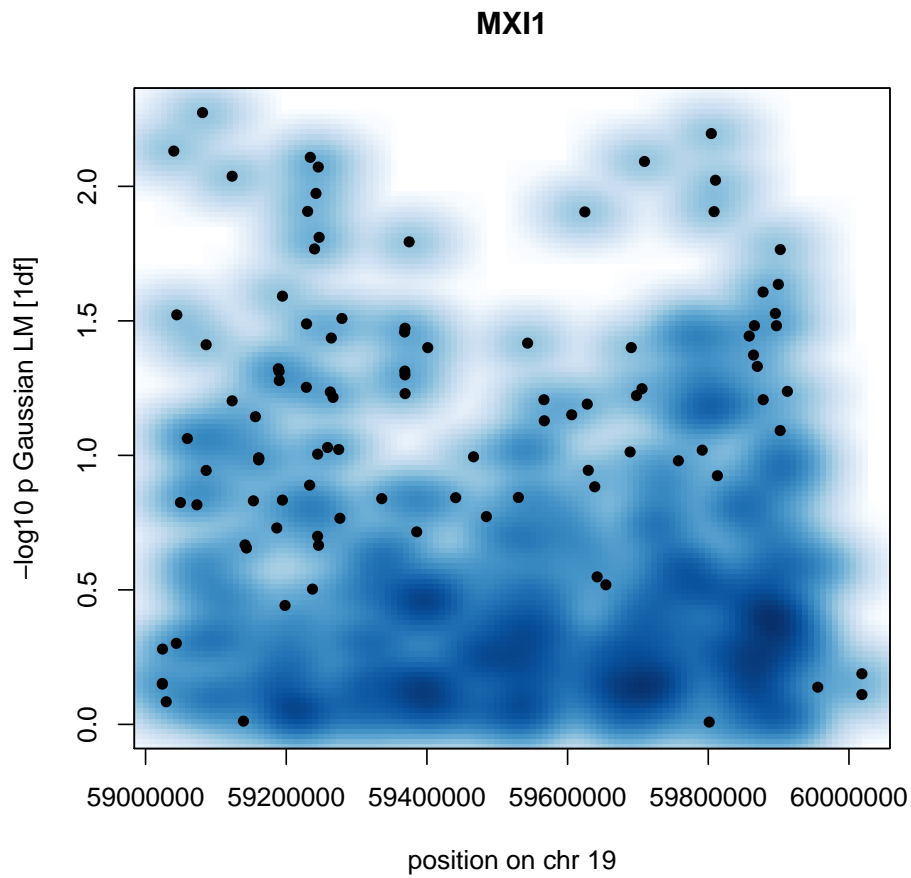
[1] "GI_18641367-A" "GI_18641367-I" "GI_18641369-I"

```

```

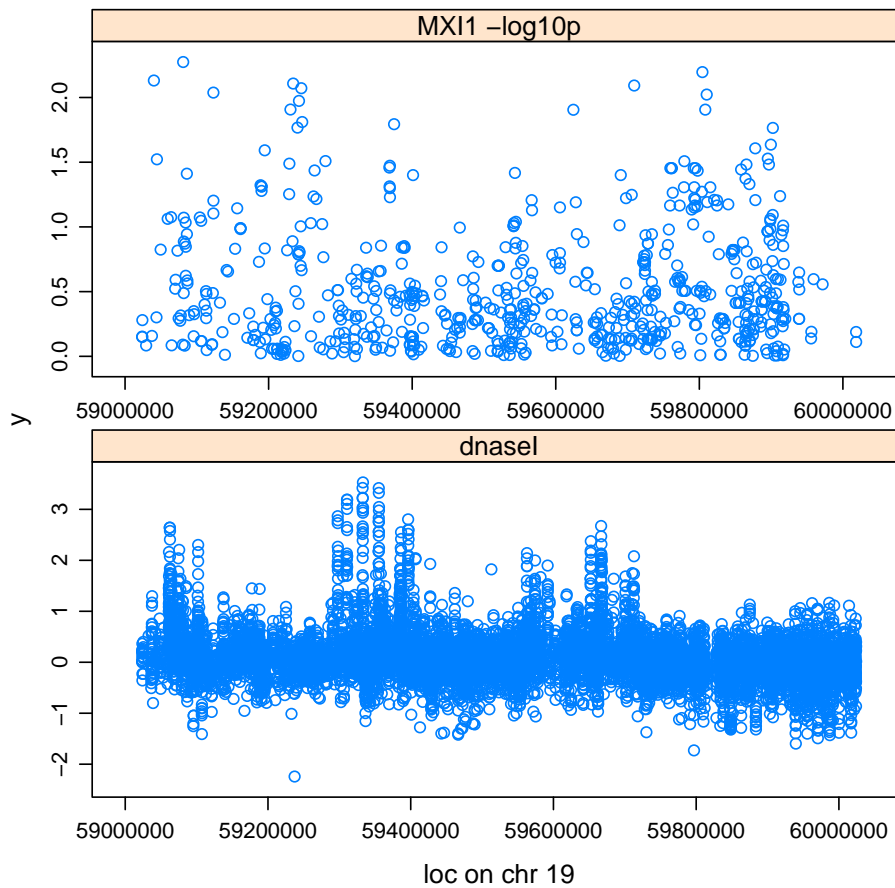
> plot(smxi1)
> options(oo)

```



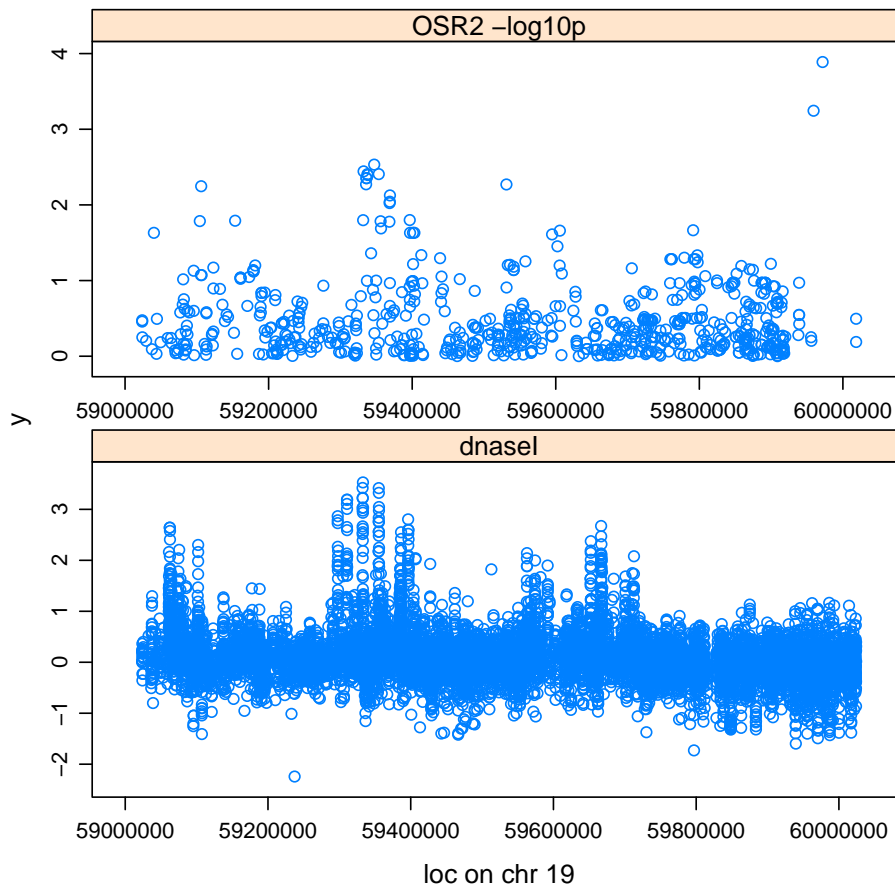
We'd like to look at the SNP screen results juxtaposed with the DnaseI results.

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



Another example:

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



We can score the highly associated snps for closeness to a highly DnaseI sensitive region using ALICOR:

```
> ALICOR(sOSR2, c19g)
```

```
[1] 0.2678520
```

```
> ALICOR(smx11, c19g)
```

```
[1] -0.01268991
```

```
> if (interactive()) {
+   if (!exists("mads"))
+     mads = apply(exprs(c19gf), 1, mad)
+   if (interactive())
+     fn = featureNames(c19gf)[which(mads > quantile(mads,
+     0.6))]
+   if (!interactive())
```

```

+       fn = featureNames(c19gf)[which(mads > quantile(mads,
+       0.97))]]
+   n19g = c19gf[exFeatID(fn), ]
+   if (file.exists("tw19g.rda"))
+       load("tw19g.rda")
+   if (!exists("tw19g"))
+       tw19g = twSnpScreen(n19g, chr19gmeta, ~., fastAGMfitter)
+   if (!file.exists("tw19g.rda"))
+       save(tw19g, file = "tw19g.rda")
+   if (file.exists("allscor.rda"))
+       load("allscor.rda")
+   if (!exists("allscor"))
+       allscor = sapply(tw19g, function(x) {
+           if (inherits(x, "try-error"))
+               return(NA)
+           else return(ALICOR(x, c19g))
+       })
+   if (!file.exists("allscor.rda"))
+       save(allscor, file = "allscor.rda")
+ }

```

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.