Figure panels 4b and 4d from the ENCODE (2012) paper

Chaocheng, Wolfgang Huber

January 9, 2012

Contents

1 Data import and preparations

version for 2011-11-20

```
> library("ENCODEFig4Band4D")
> library("randomForest")
> library("earth")
> data("TF-Model")
```

2 Main Figure 4B, aggregation plot of 4 selected TFs

Some explanatory text.

```
> data = TF_binding_profile_160bin
> cnum = nrow(data)/3
> dat1 = data[(1:cnum)*3-1,]
> dat2 = data[(1:cnum)*3,]
> dim(dat1)
```

```
[1] 80 84
```

```
> dat1 = dat1[21:60,]                ## select 40 bins around TSS
> dat2 = dat2[21:60,]
> se = c("SYDHTFBS_K562B_YY1", "SYDHTFBS_K562_CJUN",
+        "SYDHTFBS_K562_STAT1.1", "SYDHTFBS_K562_USF2")
> dat1 = dat1[,se]
> dat2 = dat2[,se]
> cnum = ncol(dat1)
```

To construct the labels that will be used for the x-axis, we first use the column names of `dat1`, split them at underscores ("_"), and extract the third term.

```
> mynam = sapply(strsplit(colnames(dat1), split = "_"), "[", 3)
> mynam

[1] "YY1"     "CJUN"    "STAT1.1" "USF2"
```

However, now we change our mind and instead hard-code them.

```
> mynam = c("YY1", "JUN", "STAT1", "USF2")
```

```
> par(mfrow=c(2,2))
> par(mar=c(2, 2, 0.5,0.5))
> par(mgp=c(1.0, 0.2, 0))
> par(tcl=-0.2)
> par(lend=2)
> for(k in 1:cnum)
+ {
+   tmp = (-20:19)*100+50
+   maxy = max(dat1[,k], dat2[,k])
+   miny = min(dat1[,k], dat2[,k])
+   plot(tmp, as.numeric(dat1[,k]), log="y", ylim=c(miny, maxy),
+        xlab=mynam[k], ylab="Average Signal",
+        type="l", pch=20, lwd=2, cex.axis=0.7, cex.lab=0.9, col="red")
+   lines(tmp, as.numeric(dat2[,k]),
+         type="l", pch=20, lwd=2, col="green")
+   abline(v=0, lty=2)
+   if(k==1)
+     {
+       mytext = c("HCP", "LCP")
+       legend(500, 1.31, cex=0.6, legend=mytext, lwd=2, col=c("red", "green"), bty="n")
+     }
+
+ }
```

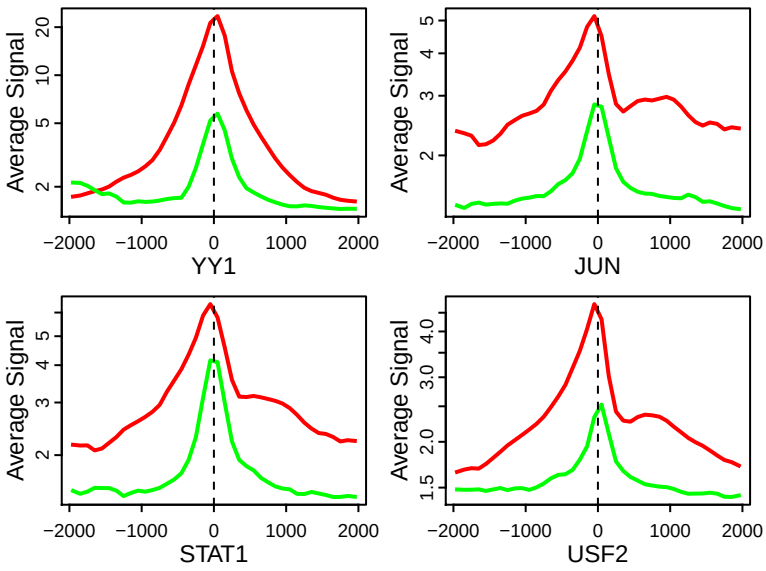

Figure 1: Figure 4B.

3 Main Figure 4D, an example result from TF model

```
> rawdata = TF_model_data
```

3.1 Classification model using RF

```
> data = rawdata
> dat1 = data[data[,1]==0,]
> dat2 = data[data[,1]>0,]
> dim(dat1)

[1] 29131    41

> dim(dat2)

[1] 55157    41

> dat1[,1] = "No"
> dat2[,1] = "Yes"
> se1 = sample(1:nrow(dat1), 1000)
> se2 = sample(1:nrow(dat2), 1000)
> tr = rbind(dat1[se1,], dat2[se2,])
```

```
> te = rbind(dat1[-se1,], dat2[-se2,])
> ##
> class.gen= row.names(tr)
> tr[,1] = as.factor(tr[,1])
> mm1 = randomForest(tr[,-1], tr[,1])
> pre= predict(mm1,te[,-1])
> res = table(pre, te[,1])
> (res[1,2]+res[2,1])/sum(res)

[1] 0.2057894

> pre= predict(mm1,te[,-1], type="prob")
> res = pre
> thr = (1:99)*0.01
> yy =  xx =  rep(0, length(thr))
> for(i in 1:length(thr))
+ {
+     aa = sum(res[,1]>=thr[i] & te[,1]=="No")
+     bb = sum(res[,1]<thr[i] & te[,1]=="No")
+     cc = sum(res[,1]>=thr[i] & te[,1]=="Yes")
+     dd = sum(res[,1]<thr[i] & te[,1]=="Yes")
+     yy[i] = aa/(aa+bb)
+     xx[i] = cc/(cc+dd)
+ }
> xx = c(1, xx, 0)
> yy = c(1, yy, 0)
> tmp1 = tmp2 = rep(0,100)
> for(i in 1:100)
+ {
+       tmp1[i] = xx[i]-xx[i+1]
+       tmp2[i] = (yy[i+1]+yy[i])/2
+ }
> myauc = sum(tmp1*tmp2)
```

3.2 Test expression model using predicted expressed genes

```
> data = rawdata
> data = data[!row.names(data)%in%class.gen, ]
> pre= predict(mm1,data[,-1])
> sum(pre=="Yes")                #

[1] 45120
```

```
> my0 = data[pre=="No", 1]
> data = data[pre=="Yes", ]
> dim(data)

[1] 45120    41

> data[,1] = log2(data[,1]+0.03)
> se = sample(1:nrow(data), 2000)
> tr = data[se,]
> te = data[-se,]
> mm2 = randomForest(tr[,-1], tr[,1])
> pre= predict(mm2,te[,-1])
> corr1 = cor(pre, te[,1])
> rmse1 = sqrt(sum((pre-te[,1])^2)/length(pre))
> cod1 = 1- sum((te[,1]-pre)^2)/sum((te[,1]-mean(te[,1]))^2)
> ############################
> xx= c(pre, rep(log2(0.03), length(my0)))
> yy = c(te[,1], log2(my0+0.03) )
> corr2 = cor(xx, yy)
> rmse2 = sqrt(sum((xx-yy)^2)/length(xx))
> cod2 = 1- sum((yy-xx)^2)/sum((yy-mean(yy))^2)
> rxx = xx
> ryy = yy
> xx= rxx
> yy = ryy
> xnum = length(xx)*0.10
> se = sample(1:length(xx), xnum)
> xx = xx[se]
> yy = yy[se]
```

3.3 Make the figure

```
> myFig = "Main_Fig4D.pdf"
> pdf(file=myFig, height =8, width = 16, pointsize=9)
> split.screen(c(1,2))
> screen(1)
> par(mar=c(5, 5, 5, 2) + 0.1)
> plot(xx, yy, xlab="predicted expression (log2)", ylab="measured expression (log2)", main=
> mylm = lm(yy~xx)
> abline(mylm, col="red")
> maxy = max(yy)
> miny = min(yy)
```

```
> maxx = max(xx)
> minx = min(xx)
> posx = minx + 0*(maxx-minx)/20
> posy = maxy - (maxy-miny)/20
> text(posx, posy, pos=4,  "Pearson's r=0.81; RMSE=2.57", cex=2)
> posx = minx + 0*(maxx-minx)/20
> posy = maxy- 1.9*(maxy-miny)/20
> text(posx, posy,  pos=4, "Classification: AUC = 0.89", cex=1.4)
> posx = minx + 0*(maxx-minx)/20
> posy = maxy- 2.6*(maxy-miny)/20
> text(posx, posy,  pos=4, "Rrgression: r = 0.62; RMSE = 3.06", cex=1.4)
> split.screen( figs = c( 2, 1), screen = 2 )
> screen(3)
> par(mar=c(5, 6, 5, 2) + 0.1, lwd=2)
> tmp = mm1$importance
> tmp = tmp[,1]
> tmp = sort(tmp, decreasing=T)
> barplot(tmp, ylab="Classification\n(Mean Decreased GINI)", names.arg="",  main="Relative
> par(xpd=T)
> for(s in 1:length(tmp))
+ {
+         posx = 1.5 + 1.2*(s-1)
+         posy = -max(tmp)/40
+         text(posx, posy, pos=2,  names(tmp)[s], srt=45, cex=0.8)
+ }
> screen(4)
> par(mar=c(5, 6, 1, 2) + 0.1, lwd=2)
> tmp = mm2$importance
> tmp = tmp[,1]
> tmp = sort(tmp, decreasing=T)
> barplot(tmp, ylab="Regression\n(Increase of Node Purity)", names.arg="", cex.lab =1.5)
> par(xpd=T)
> for(s in 1:length(tmp))
+ {
+         posx = 1.5 + 1.2*(s-1)
+         posy = -max(tmp)/40
+         text(posx, posy, pos=2,  names(tmp)[s], srt=45, cex=0.8)
+ }
> close.screen(all = TRUE)
> dev.off()
```

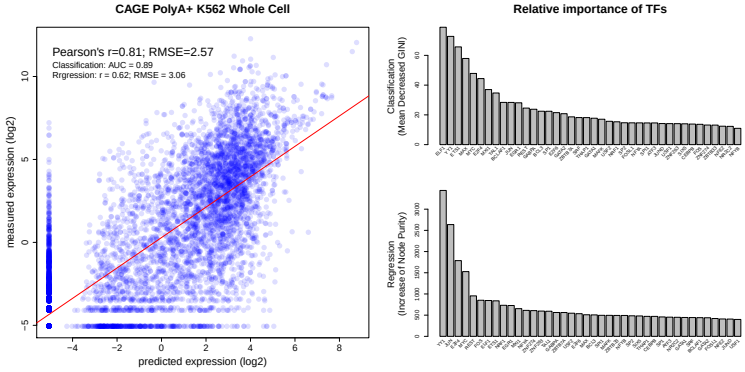

Figure 2: Figure 4D.