Supplementary Figure 2

Contents

Input Data

```
##load the library
library(motifStack)

##read pcms
pcmpath <- "pcmsDatasetDM"
pcms <- readPCM(pcmpath)
##convert to pfms
pfms<-lapply(pcms,pcm2pfm)
```

Supplementary Figure 2A-D. basic linear and radial trees

motifStack is designed to show multiple motifs in same canvas and it is simple. Users can use plotMotifLogoStack, plotMotifLogoStackWithTree or plotMotifStackWithRadialPhylog function to plot sequence logo stack in different layout. Also motifStack has one-step workflow to do distance calculation (depend on MotIV) and alignment, as well as to generate figure.

Using aligned motifs can help us to compare grouped motifs (default in motifStack function). DNAmotifAlignment can align DNA motifs for plotting based on their phylogeny. The motifs will be trimmed by Kullback-Leibler divergency to background (information content, IC) at both ends before the alignment without gap by comparison of the average log-likelihood ratio (ALLR) for each position.

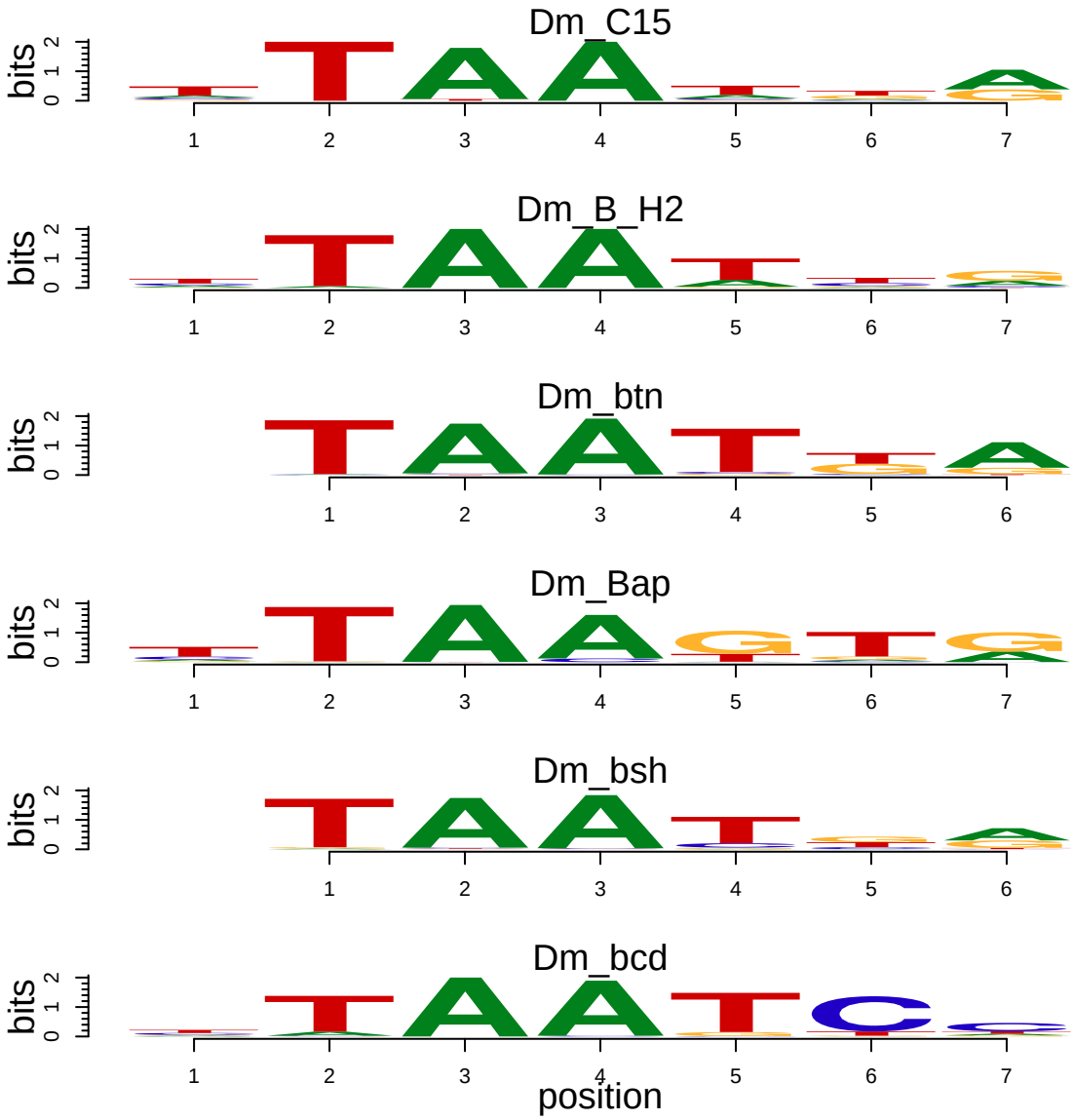
Supplementary Figure 2A

```
subset_pfms <- pfms[10:15]
motifStack(subset_pfms, layout="stack")
```

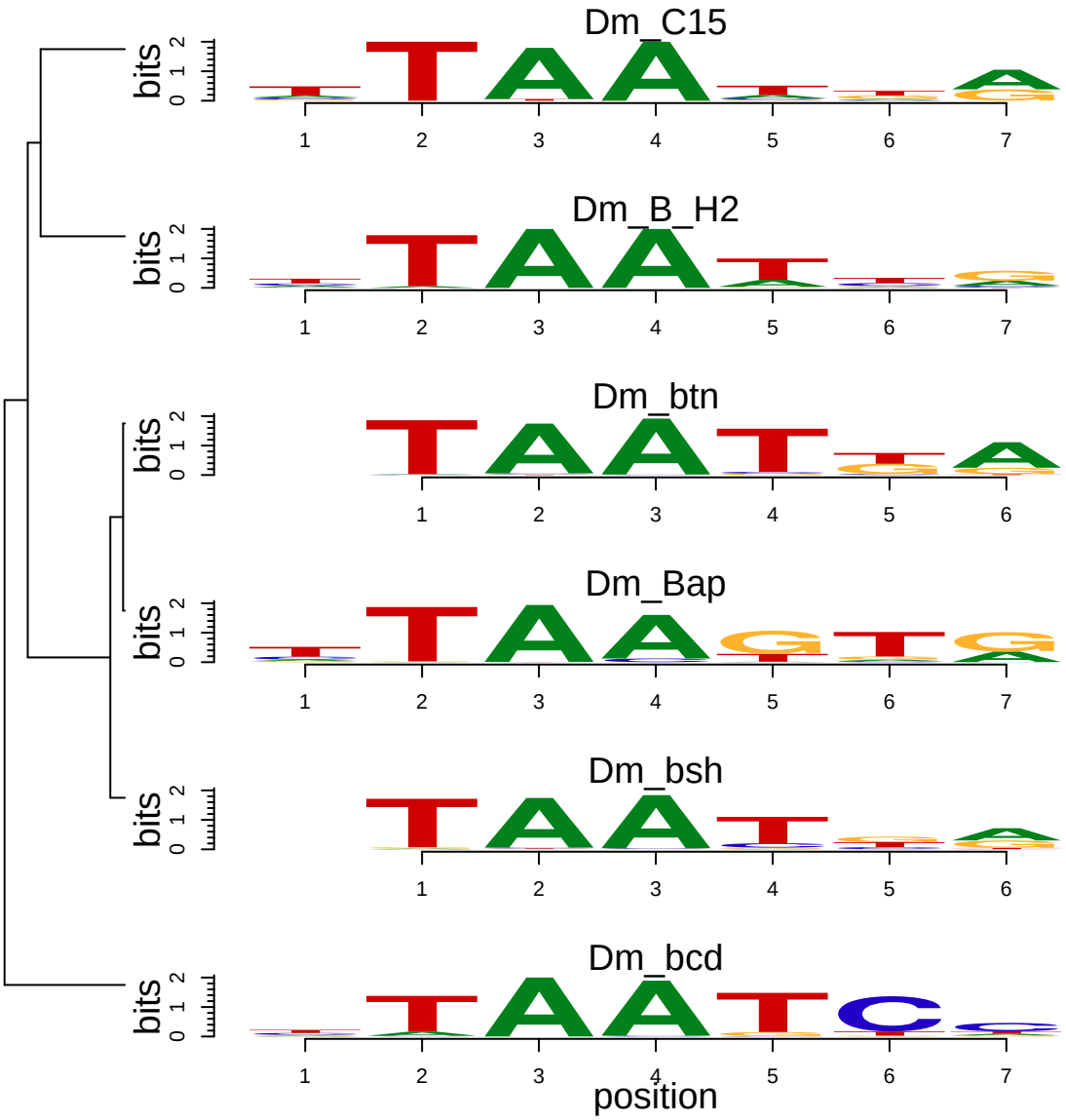
Supplementary Figure 2B

The distance tree could show the relationship among motifs. motifStack calculate the distance of motifs in background by MotIV:::motifDistances, which implemented STAMP. We can simply change the style of layout to show the distance among motifs.

```
##using default setting of MotIV to do cluster and show results
motifStack(subset_pfms, layout="tree", trueDist=TRUE)
```



Supplementary Figure 2A: stack layout of motifStack

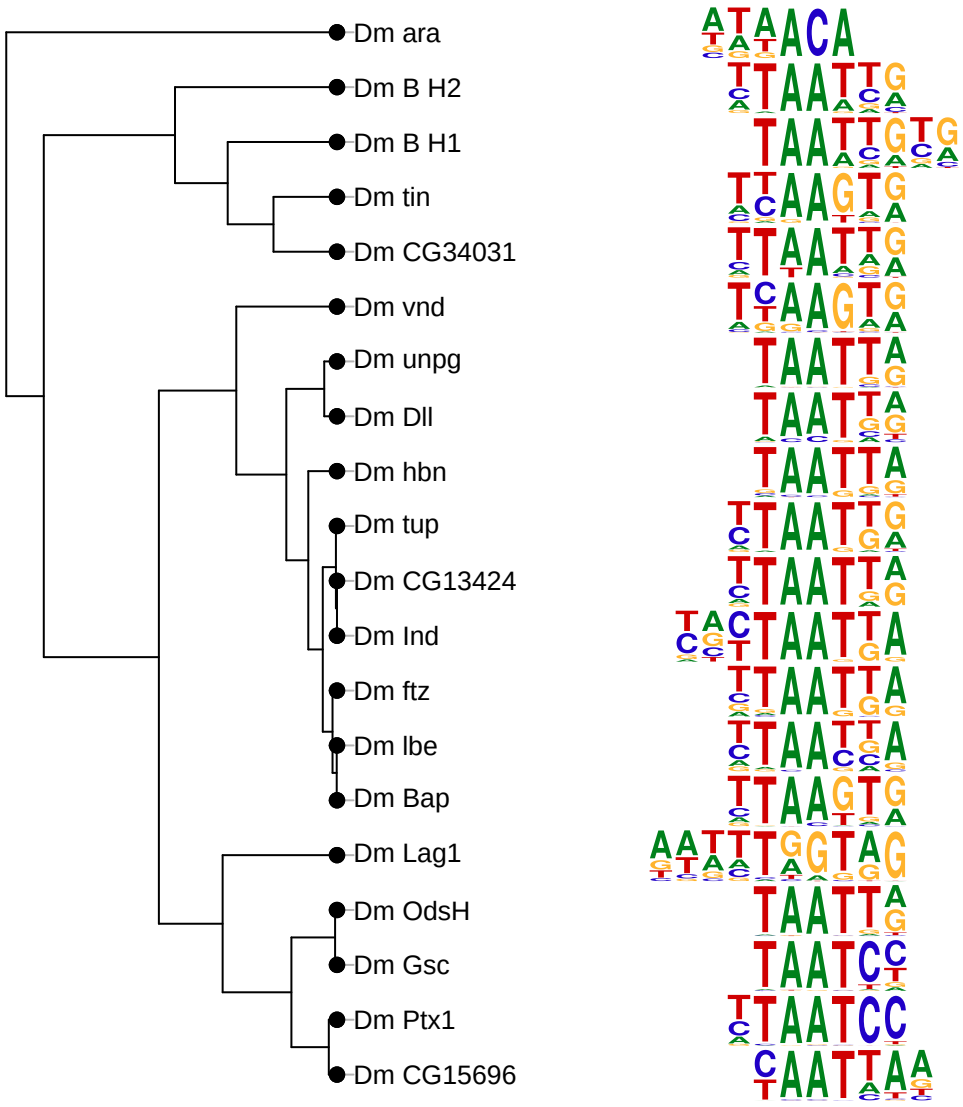


Supplementary Figure 2B: tree layout of motifStack

Supplementary Figure 2C

The tree-view style needs enough margin to show the logo name and logo position. If you want to show more motifs, you would want to try phylogeny style.

```
##try different style of sequence logo
motifStack(pfms[sample(1:length(pfms), 20)], layout="phylog", clabel.leaves=.8,
           f.logo=0.6, ic.scale=FALSE)
```



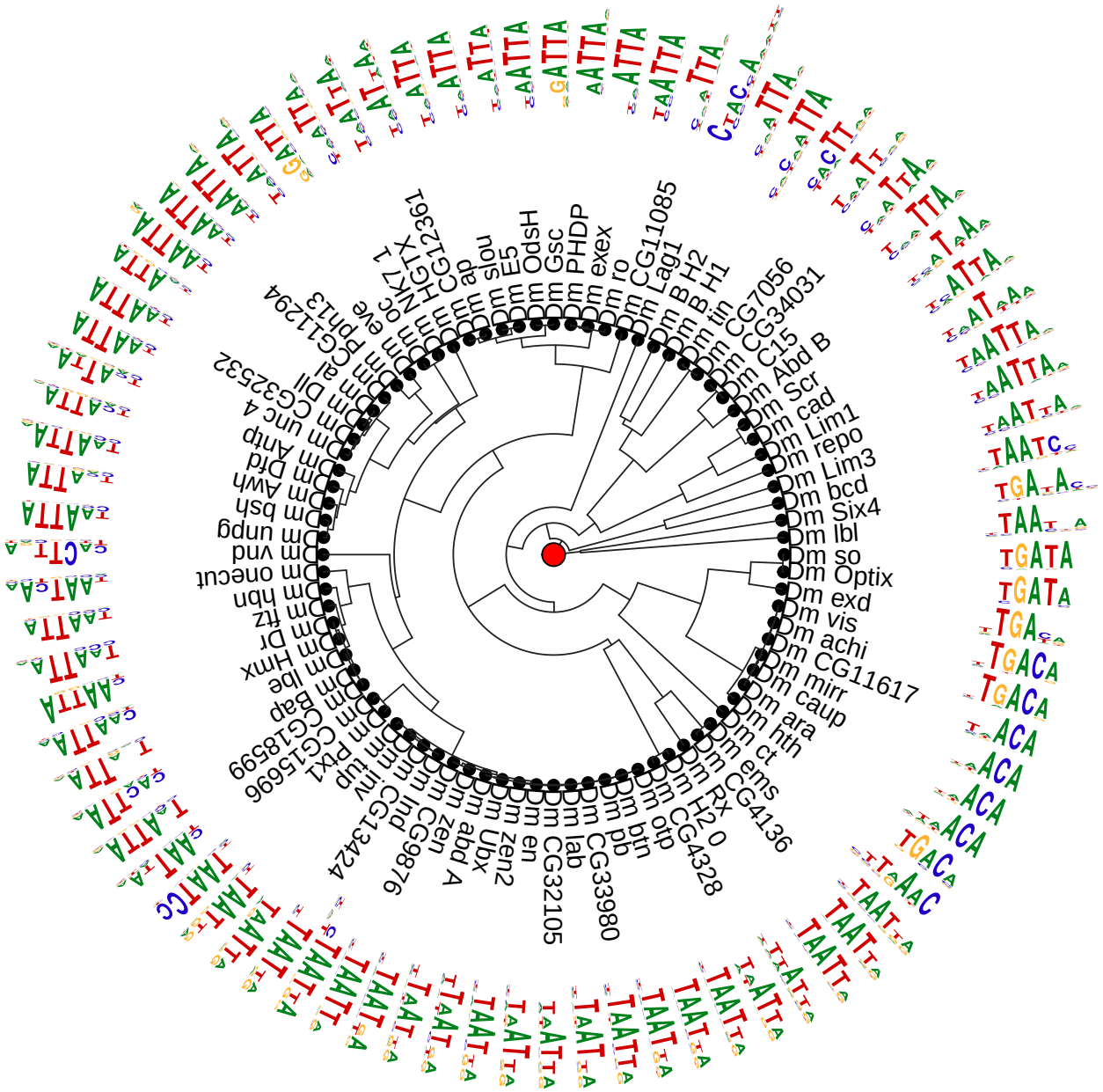
Supplementary Figure 2C: phylog layout of motifStack

Supplementary Figure 2D

If you want to show even more motifs in one figure, you would want to try radialPhylog layout.

```
##using Pearson Correleation Coefficient as Column comparison metric and Ungapped Smith-Waterman
motifStack(pfms, layout="radialPhylog")
```

As it is shown above, draw multiple motifs can be done very easily. Combined with vertical phylogenic tree



Supplementary Figure 2D: radialPhylog layout of motifStack

to show the distance of each motif, the sequence logos were aligned for each other. The tree style motif stack can easily impress people with the TF specificities and their phylogeny for small number of motifs. While radial phylogenic style motif stack are more suitable to show bunch of sequence logos with their phylogeny.

Supplementary Figure 2E-G. merge motifs by different distance

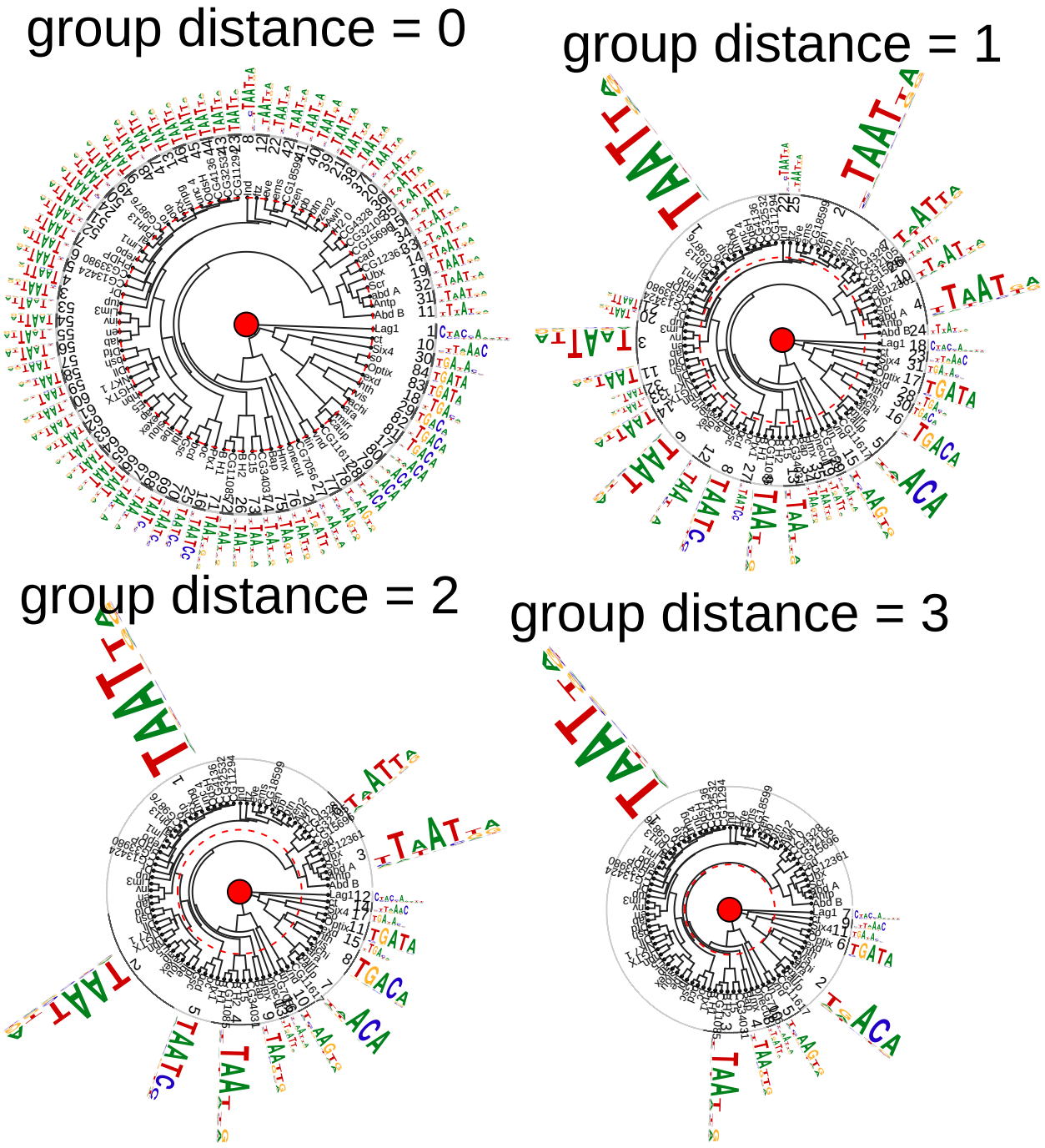
If several motifs have similar DNA-binding specificities in same species, they will compete to each other or work as pioneer factors. If the close motifs are in different species, this will provide a clue for evolution analysis. The close motifs may be shown by one sequence logo, called motif signature to emphaze the identical DNA-binding specificities.

To summarize motifs with signatures, the distances of the motifs should be calculated by STAMP, MovIV or MatAlign first. And then use motifSignature to extract the signatures. All the motifs are grouped by distance cutoff. For each grouped motifs, one signature will be generated. Here we show how the group distance will affect the results of merging. The red dotted line indicates the cutoff distance.

Supplementary Figure 2E

```
outpath <- "output"
matalign_path <- "./app/matalign-v4a"
neighbor_path <- "./app/neighbor.app/Contents/MacOS/neighbor"
pcmpath <- "pcmsDatasetDM"
pcms <- readPCM(pcmpath)
pfms<-lapply(pcms,pcm2pfm)
system(paste("perl MatAlign2tree.pl --in . --pcmpath", pcmpath,
             "--out", outpath,
             "--matalign", matalign_path,
             "--neighbor", neighbor_path,
             "--tree","UPGMA"))
newickstrUPGMA <- readLines(con=file.path(outpath, "NJ.matalign.distMX.nwk"))
phylogUPGMAmatAlign <- newick2phylog(newickstrUPGMA, FALSE)
##get the leaves of phylog to reorder the pfms
matAlignLeaveNames <- names(phylogUPGMAmatAlign$leaves)
this_motifs <- pfms[matAlignLeaveNames]
matAlignLeaveNames <- gsub("^Dm_", "", matAlignLeaveNames)

for(groupDistance in c(0, 1, 2, 3)){
    motifSig <- motifSignature(this_motifs,
                               phylogUPGMAmatAlign,
                               groupDistance=groupDistance,
                               min.freq=1)
    this_sig <- signatures(motifSig)
    ## get color set for signature groups
    this_gpCol <- sigColor(motifSig)
    plotMotifStackWithRadialPhylog(phylog=phylogUPGMAmatAlign,
                                   pfms=this_sig,
                                   col.inner.label.circle=this_gpCol,
                                   inner.label.circle.width=0.02,
                                   labels.leaves=matAlignLeaveNames,
                                   cleaves=.2, circle=1, circle.motif=1.5,
                                   clabel.leaves=.4, motifScale="logarithmic",
                                   angle=358, plotIndex=TRUE, IndexCex=.6,
```

```
                                groupDistance=groupDistance)
    text(0, 2.3, label=paste("group distance =", groupDistance), cex=2)
}
```



Once the signatures were generated, motifStack can show the signatures with multiple styles, rectangles, cloud or phylogenetic trees. As shown in the following figure, motif signatures can be shown as word cloud in a rectangle or a circle. The grey number in the right top corner indicates the number of motifs for this signature.

The signature cloud provides a concise visual summary of the sequence logos, which is helpful for evolution analysis and interpretation. However, the signature cloud loses the source motif names and their distance

information.

Supplementary Figure 2F

```
## get signature
gpDist <- c(.5, 1)
motifSig <- lapply(gpDist, motifSignature,
                   pfms=this_motifs,
                   phylog=phylogUPGMAmatAlign,
                   min.freq=1)
## motif cloud, cloud style
for(i in 1:2){
    motifCloud(motifSig[[i]], layout="cloud", scale=c(9, .75))
    op <- par(mar = c(0, 0, 0, 0))
    text(.5, .95, label=paste("group distance =", gpDist[i]))
    par(op)
}
```

Supplementary Figure 2G

```
## motif cloud, rectangle style
for(i in 1:2){
    motifCloud(motifSig[[i]], layout="rectangles", ic.scale=FALSE)
    message(paste("group distance =", gpDist[i]))
}
```

```
## group distance = 0.5
```

Supplementary Figure 2G: motifCloud

```
## group distance = 1
```

Supplementary Figure 2H-M. color sets

The radial phylogenic style motif stack can be divided into three parts: 1. radial phylogenic tree in inner circle with colorful background, 2. motif names in middle circle with color label ring, and 3. sequence logos in outer circle with color label ring. Users can use four groups of different color sets to fill the background of inner circle, the motif names, the inner label ring and the outer label ring. In following figure, we use background color of inner circle to show the motifs from the same PBM, background color of middle circle to show data source of motifs, and color of motif name label to describe the algorithm in generating the motifs.

```
##color sets
getPairColor <- function(n=10L){
    if(n %% 2 != 0) n <- n+1
    n <- n/2
    n <- rainbow(n)#as.character(t(matrix(rainbow(n=n), ncol=2, byrow=FALSE)))
    n2 <- highlightCol(n, .5)
    as.character(t(cbind(n, n2)))
}

pairColor <- getPairColor(22)

pfmList2matrixList <- function(pfms){
        m <- lapply(pfms, function(.ele) as(.ele, "matrix"))
        names(m) <- unlist(lapply(pfms, function(.ele) .ele@name))
        m
}
getMotIVOut <- function(pfms, cc, align){
```

Supplementary Figure 2G: motifCloud

```
    jaspar.scores <-
        MotIV::readDBScores(
            file.path(".", "app", "scores",
                      paste("JaspRand_", cc, "_", align, ".scores", sep="")))
    d <- MotIV::motifDistances(pfmList2matrixList(pfms), cc=cc, align=align)
    hc <- MotIV::motifHclust(d, method="average")
    phylog <- hclust2phylog(hc)
    pfms <- pfms[hc$order]
    aligned.pfms <- DNAmotifAlignment(pfms)
    leaveNames <-names(phylog$leaves)
    ## data source
    dataSource <- factor(grepl("_M", leaveNames))
    levels(dataSource) <- c("yellow", "blue") ##c("Uniprobe", "CIS-BP")
    dataSource <- as.character(dataSource)
    ## algorithm
    algorithm <-
        factor((!grepl("_M", leaveNames)) + grepl("_bml", leaveNames))
    levels(algorithm) <-
        c("blue", "brown", "orange") ##("DREAM5", "Seed-And-Wobble", "BEEML")
    algorithm <- as.character(algorithm)
    ## motifs from same PBM data
    motifGroup <- factor(gsub("(.*?)_.*$", "\\1", leaveNames))
    levels.motifGroup <- levels(motifGroup)
    levels(motifGroup) <- pairColor[1:length(levels(motifGroup))]
    colors.motifGroup <- levels(motifGroup)
    motifGroup <- as.character(motifGroup)
    return(list(aligned.pfms=aligned.pfms, unaligned.pfms=pfms, phylog=phylog,
```

```
                leaveNames=leaveNames,
                dataSource=dataSource,
                algorithm=algorithm,
                motifGroup=motifGroup,
                levels.motifGroup=levels.motifGroup,
                colors.motifGroup=colors.motifGroup))
}
##read pcms
pcmpath <- "pcmsDatasetAlgorithm"
pcms <- readPCM(pcmpath)
##convert to pfms
pfms<-lapply(pcms,pcm2pfm)
```

Supplementary Figure 2H

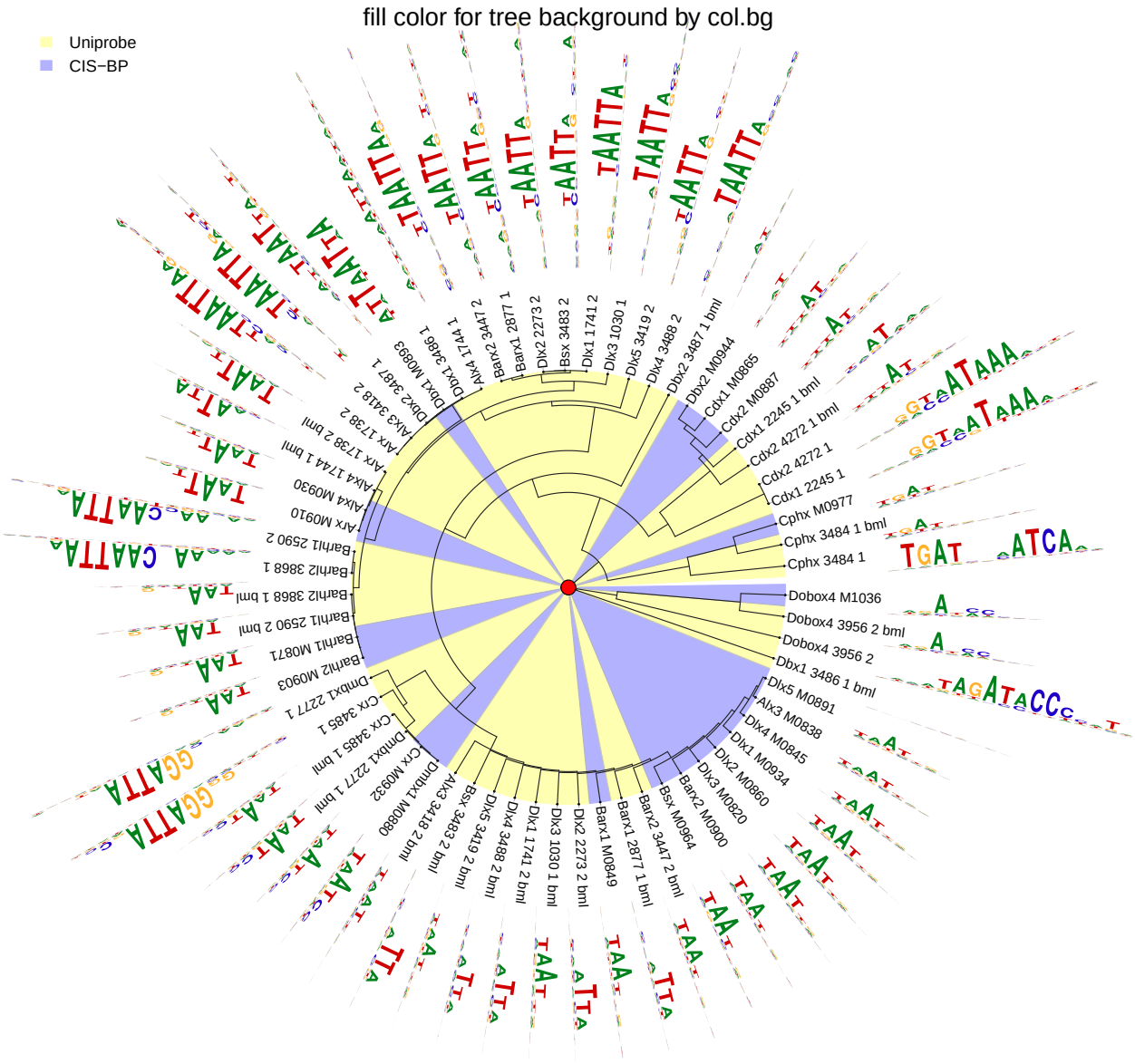
```
motIVout <- getMotIVOut(pfms, "PCC", "SWU")
attach(motIVout)
plotMotifStackWithRadialPhylog(phylog=phylog, pfms=unaligned.pfms,
                            labels.leaves=leaveNames,
                            col.bg=dataSource, col.bg.alpha=.3,
                            cleaves=.2, circle=1.1, circle.motif=1.6,
                            clabel.leaves=.8, angle=358)
legend(-2.3, 2.4, legend=c("Uniprobe", "CIS-BP"),
       fill= highlightCol(c("yellow", "blue"), alpha=.3),
       border="white", lty=NULL, bty = "n", cex=1)
text(0, 2.4, "fill color for tree background by col.bg", cex=1.5)
```

Supplementary Figure 2I

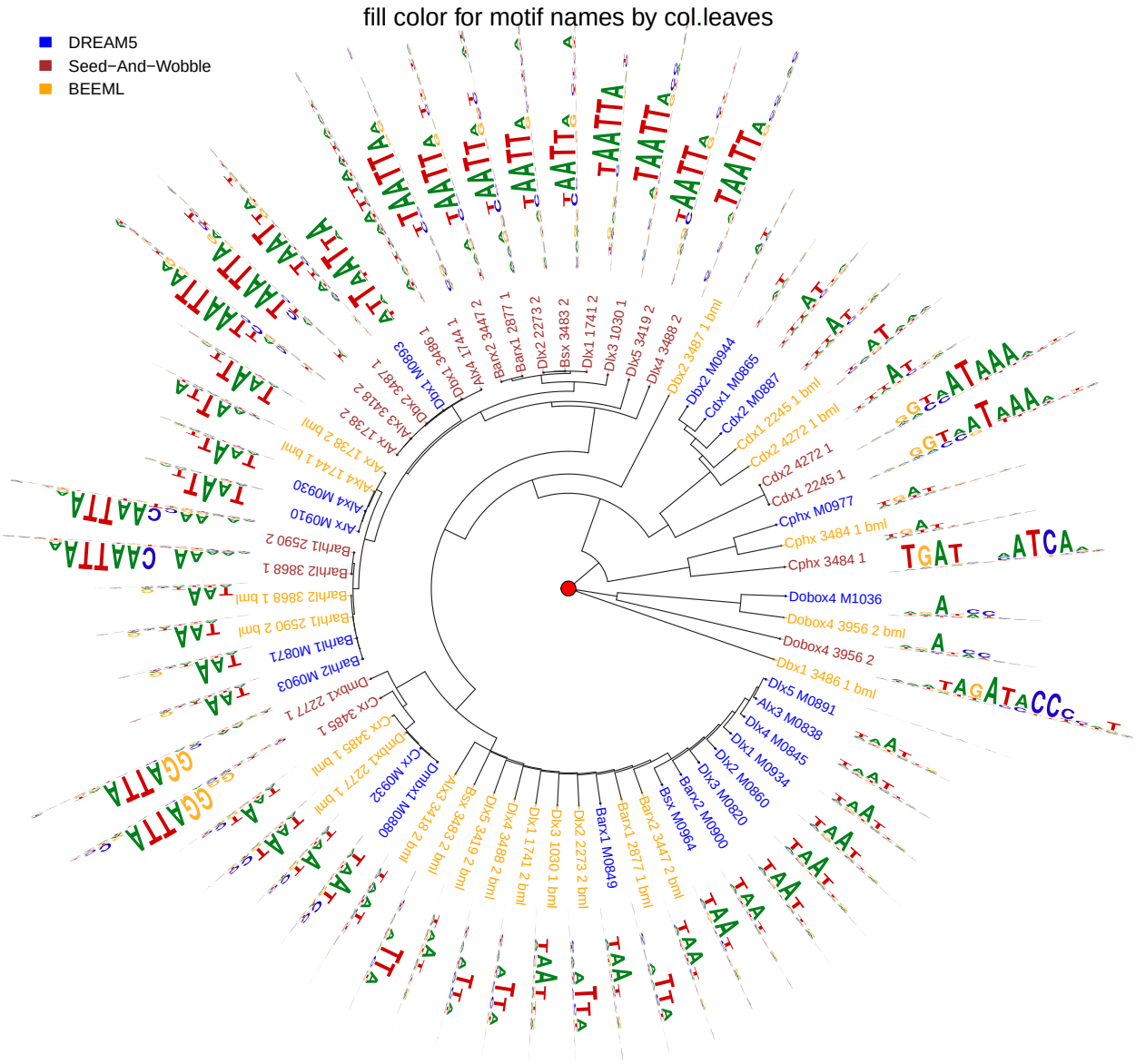
```
plotMotifStackWithRadialPhylog(phylog=phylog, pfms=unaligned.pfms,
                            labels.leaves=leaveNames,
                            col.leaves=algorithm,
                            cleaves=.2, circle=1.1, circle.motif=1.6,
                            clabel.leaves=.8, angle=358)
legend(-2.3, 2.4, legend=c("DREAM5", "Seed-And-Wobble", "BEEML"),
       fill= c("blue", "brown", "orange"),
       border="white", lty=NULL, bty = "n", cex=1)
text(0, 2.4, "fill color for motif names by col.leaves", cex=1.5)
```

Supplementary Figure 2J

```
icgp <- ifelse(sapply(sapply(unaligned.pfms, getIC), mean) > 0.6,
               "lightgray", "black")
plotMotifStackWithRadialPhylog(phylog=phylog, pfms=unaligned.pfms,
                            labels.leaves=leaveNames,
                            col.leaves.bg=icgp,
                            col.leaves.bg.alpha=.3,
                            cleaves=.2, circle=1.1, circle.motif=1.6,
                            clabel.leaves=.8, angle=358)
```

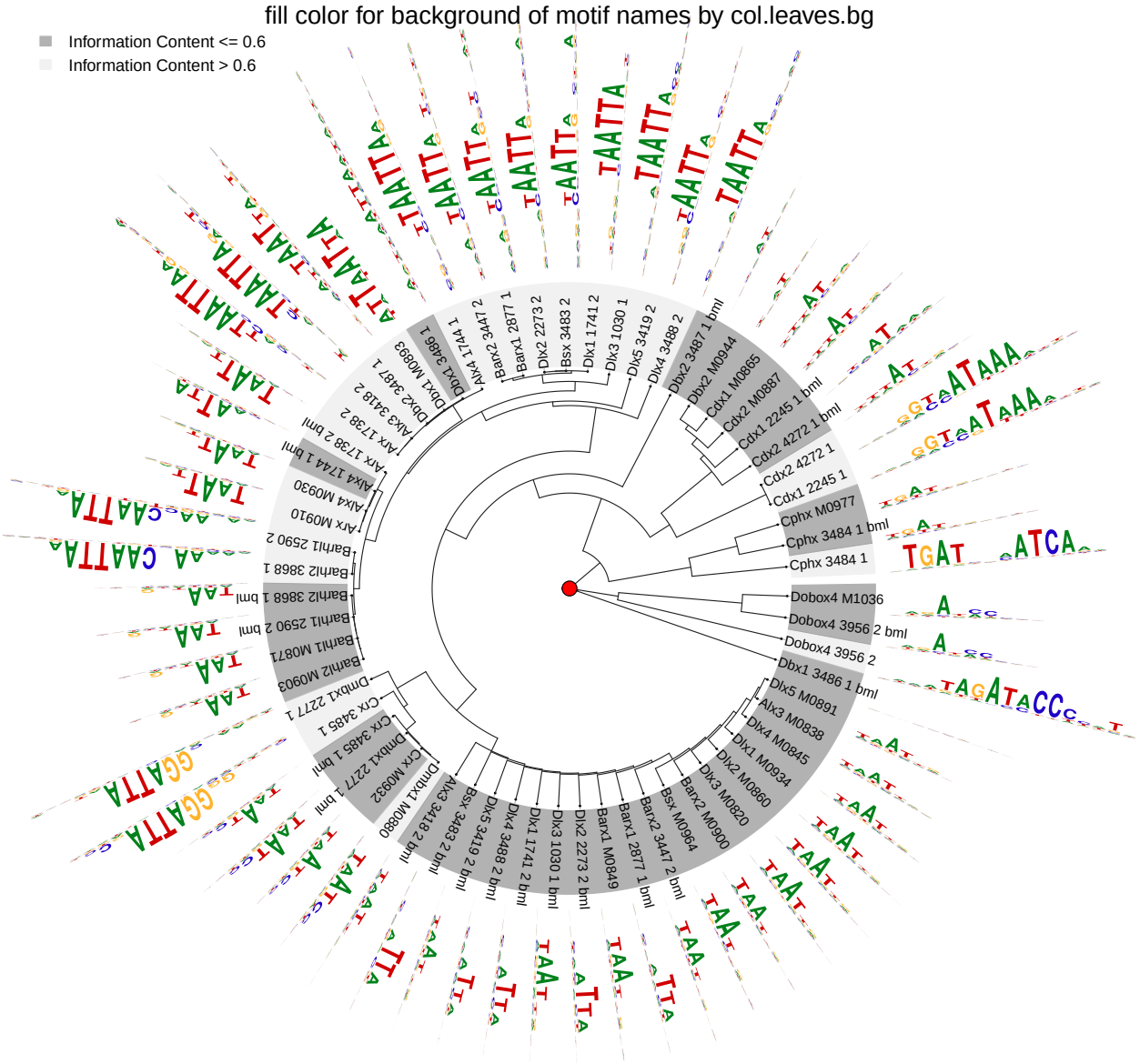


Supplementary Figure 2H: color setting, tree background



Supplementary Figure 2I: color setting, leaves

```
legend(-2.3, 2.4,
       legend=c("Information Content <= 0.6", "Information Content > 0.6"),
       fill= highlightCol(c("black", "lightgray"), alpha=.3),
       border="white", lty=NULL, bty = "n", cex=1)
text(0, 2.4,
     "fill color for background of motif names by col.leaves.bg",
     cex=1.5)
```



Supplementary Figure 2J: color setting, leave background

Supplementary Figure 2K

```
plotMotifStackWithRadialPhylog(phylog=phylog, pfms=unaligned.pfms,
                               labels.leaves=leaveNames,
```

```
                            col.inner.label.circle=motifGroup,
                            inner.label.circle.width=0.1,
                            cleaves=.2, circle=1.1, circle.motif=1.6,
                            clabel.leaves=.8, angle=358)
legend(1.5, 2.4, legend=levels.motifGroup[1:12],
       fill= colors.motifGroup[1:12],
       border="black", lty=NULL, bty = "n", ncol=2, cex=.8)
legend(1.5, -2, legend=levels.motifGroup[13:21],
       fill= colors.motifGroup[13:21],
       border="black", lty=NULL, bty = "n", ncol=2, cex=.8)
text(0, 2.4,
     "fill color for inner ring of the motif circle by col.inner.label.circle",
     cex=1.5)
```

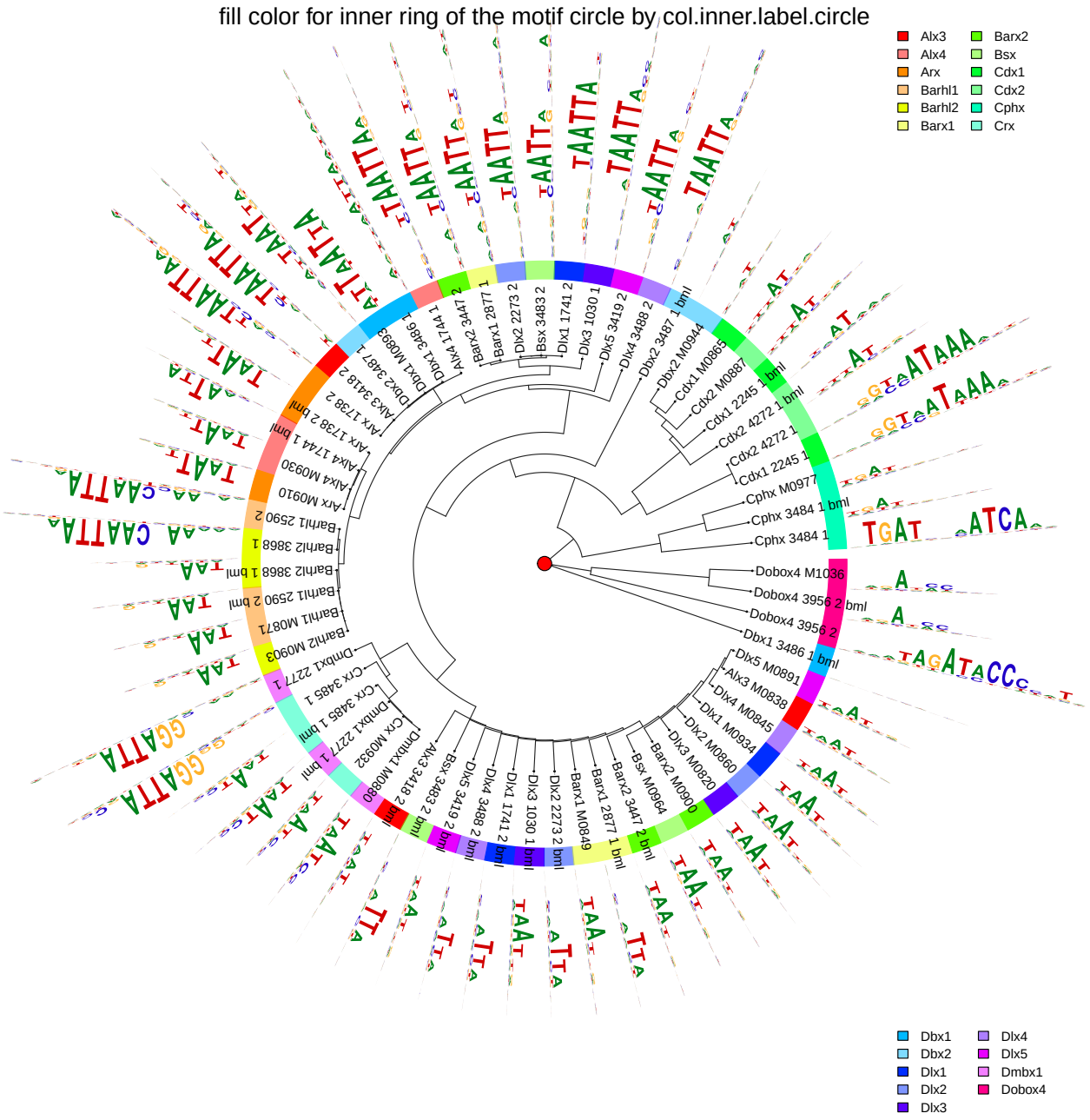
Supplementary Figure 2L

```
plotMotifStackWithRadialPhylog(phylog=phylog, pfms=unaligned.pfms,
                            labels.leaves=leaveNames,
                            col.outer.label.circle=motifGroup,
                            outer.label.circle.width=0.1,
                            cleaves=.2, circle=1.1, circle.motif=1.6,
                            clabel.leaves=.8, angle=358)
legend(1.5, 2.4, legend=levels.motifGroup[1:12],
       fill= colors.motifGroup[1:12],
       border="black", lty=NULL, bty = "n", ncol=2, cex=.8)
legend(1.5, -2, legend=levels.motifGroup[13:21],
       fill= colors.motifGroup[13:21],
       border="black", lty=NULL, bty = "n", ncol=2, cex=.8)
text(0, 2.4,
     "fill color for outer ring of the motif circle by col.outer.label.circle",
     cex=1.5)
```

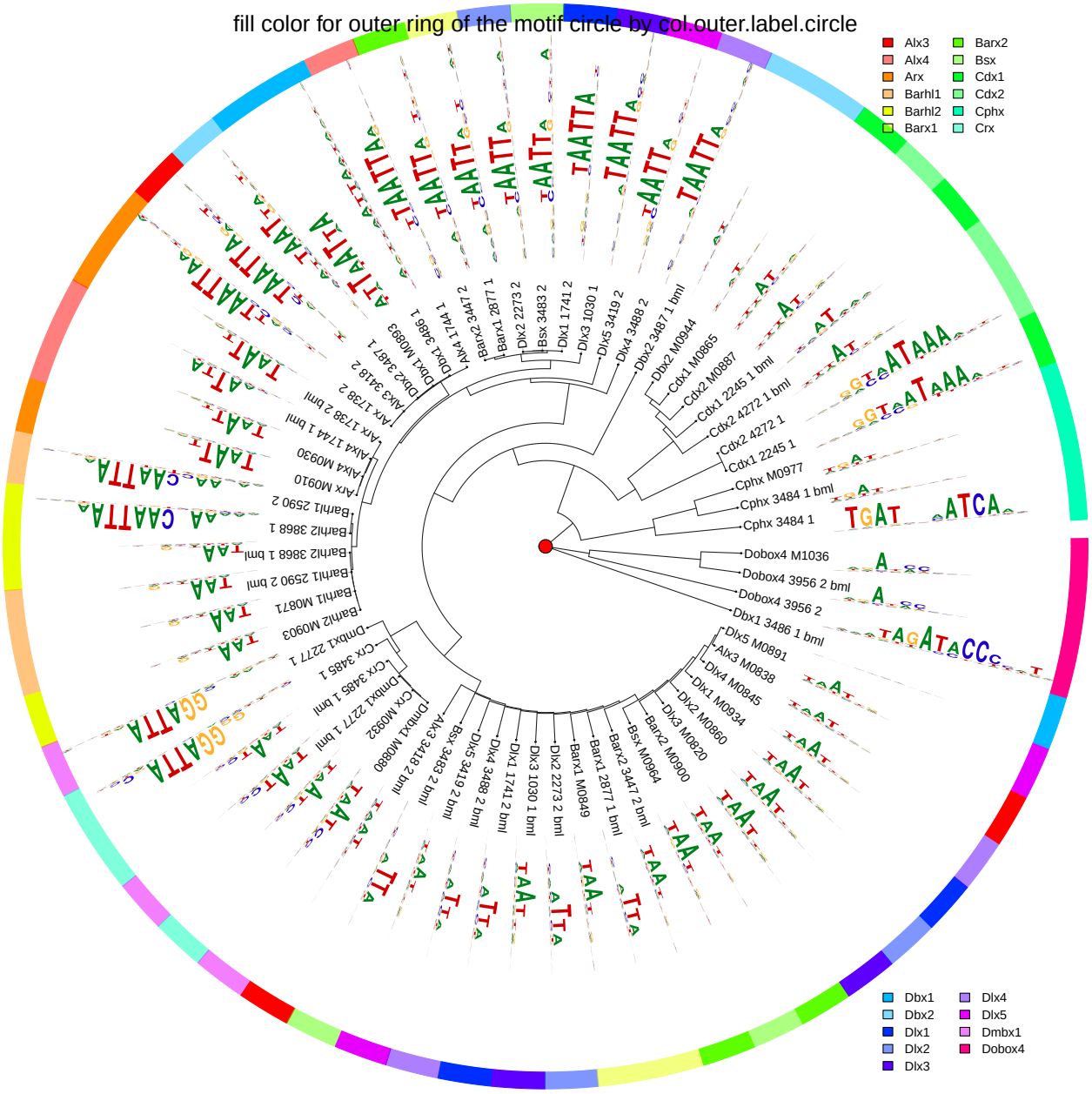
Supplementary Figure 2M

```
plotMotifStackWithRadialPhylog(phylog=phylog, pfms=aligned.pfms,
                               labels.leaves=leaveNames,
                               col.bg=algorithm, col.bg.alpha=.3,
                               col.leaves.bg=icgp,
                               col.leaves.bg.alpha=.3,
                               cleaves=.2, circle=1.1, circle.motif=1.6,
                               clabel.leaves=.8, angle=358)
legend(-2.3, 2.4,
       legend=c("algorithm", "CIS-BP/DREAM5", "Uniprobe/Seed-And-Wobble",
                "Uniprobe/BEEML", "Information Content", "<= 0.6", "> 0.6"),
       fill= c("white",
               highlightCol(c("blue", "brown", "orange"), alpha=.3),
               "white", highlightCol(c("black", "lightgray"), alpha=.3)),
       border="white", lty=NULL, bty = "n", cex=.8)
```

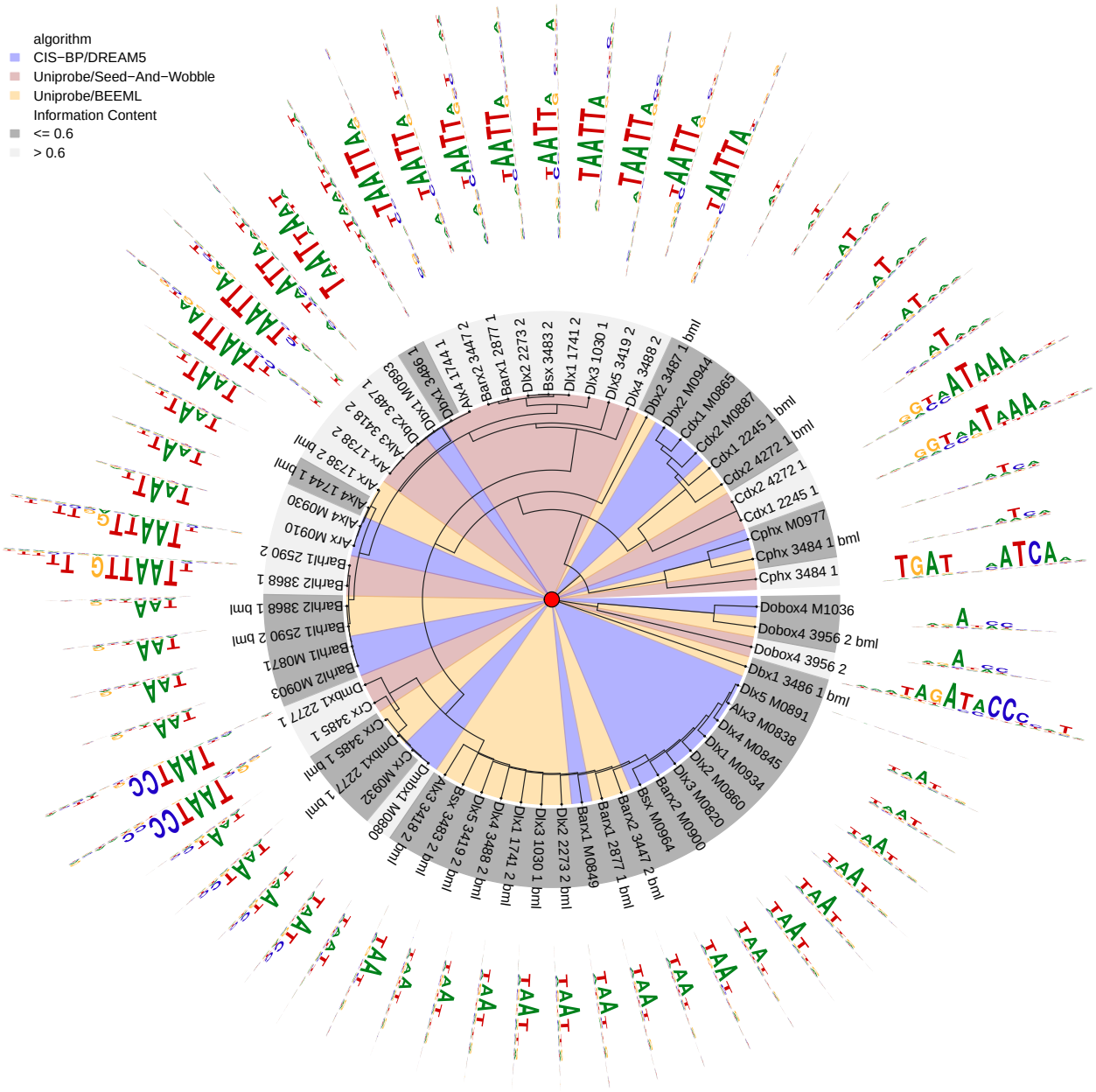
```
detach(motIVout)
```



Supplementary Figure 2K: color setting, inner circle



Supplementary Figure 2L: color setting, outer circle



Supplementary Figure 2M: color setting, combination

Supplementary Figure 2N-O. color sets applied to different alignment methods

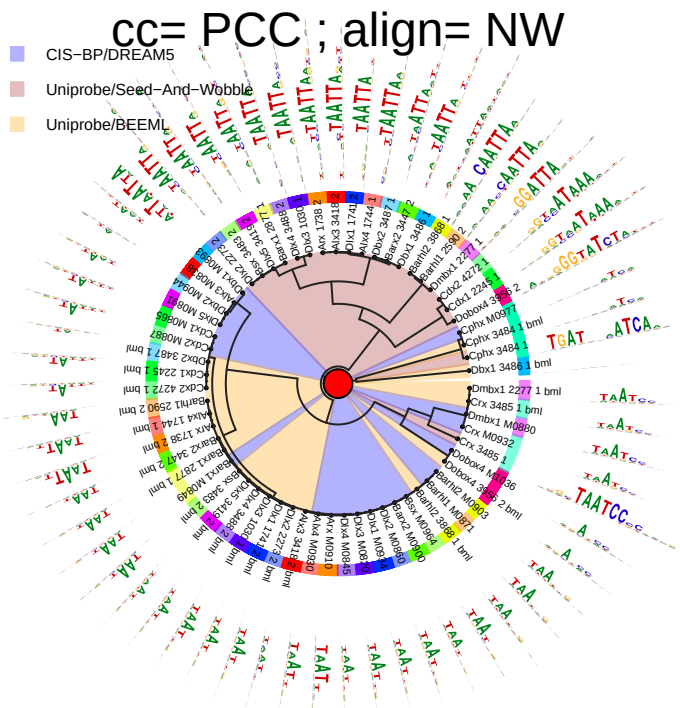
cluster by MovIV

Because the MotIV is a package in Bioconductor, it is very easy to draw motif stacks using MotIV. There are several choices to cluster motifs by MovIV.

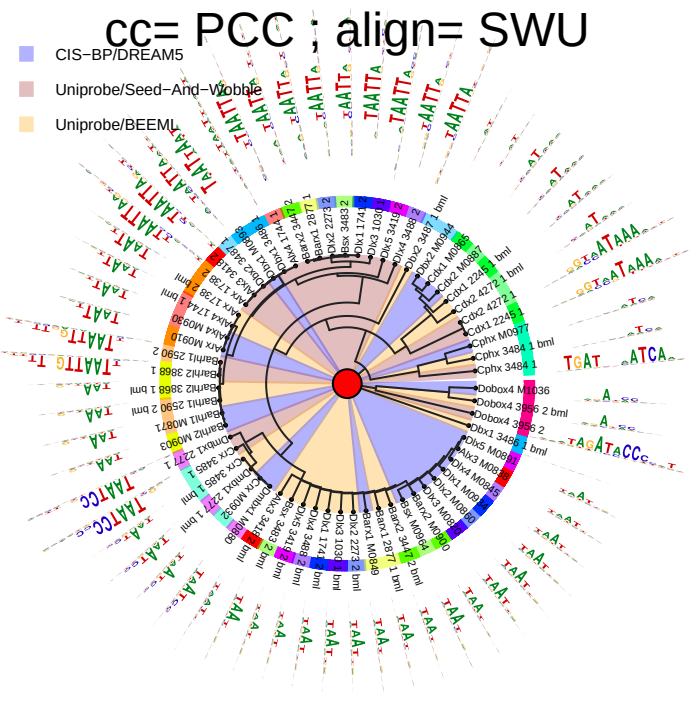
We can also try different parameters but not default setting of MotIV. Here we try to set column comparison metrics as "PCC" or "ALLR" and alignment method as Needleman-Wunsch or Ungapped Smith-Waterman.

Supplementary Figure 2N

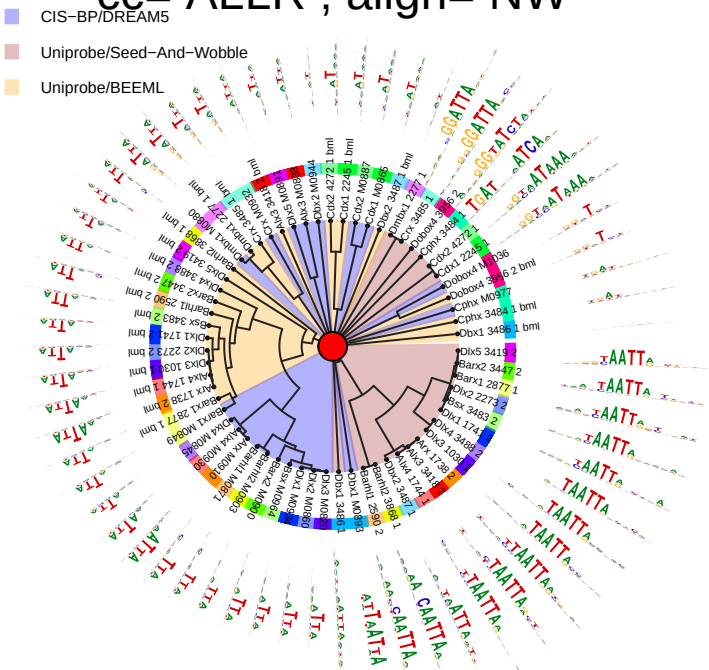
```
##using PCC as Column comparison metric and Ungapped Smith-Waterman as alignment method
##using PCC as Column comparison metric and Needleman-Wunsch as alignment method
##using ALLR as Column comparison metric and Ungapped Smith-Waterman as alignment method
##using ALLR as Column comparison metric and Needleman-Wunsch as alignment method
sta <- mapply(function(cc, align){
    motIVout <- getMotIVOut(pfms, cc, align)
    attach(motIVout)
    plotMotifStackWithRadialPhylog(phylog=phylog, pfms=aligned.pfms,
                                labels.leaves=leaveNames,
                                col.bg=algorithm, col.bg.alpha=.3,
                                col.inner.label.circle=motifGroup,
                                inner.label.circle.width=0.1,
                                cleaves=.2, circle=1.1, circle.motif=1.6,
                                clabel.leaves=.3, angle=358)

    legend(-2.3, 2.4,
           legend=c("CIS-BP/DREAM5", "Uniprobe/Seed-And-Wobble",
                    "Uniprobe/BEEML"),
           fill= highlightCol(c("blue", "brown", "orange"), alpha=.3),
           border="white", lty=NULL, bty = "n", cex=.5)
    text(0, 2.3, label=paste("cc=", cc, "; align=", align), cex=1.5)
    cnt <- rle(motifGroup)
    cnt <- split(algorithm, rep(1:length(cnt$lengths), cnt$lengths))
    cnt <- table(sapply(cnt, function(.ele) length(unique(.ele))))
    cnt.1 <- vector("integer", 3)
    names(cnt.1) <- 1:3
    cnt.1[names(cnt)] <- cnt
    cnt.1["1"] <- (cnt.1["1"]+cnt.1["2"]*2+cnt.1["3"]*3)/3 - cnt.1["2"] - cnt.1["3"]
    text(0, -2.3,
         label=paste(names(cnt.1), cnt.1, sep=":", collapse="; "), cex=1.5)
    detach(motIVout)
}, c("PCC", "PCC", "ALLR", "ALLR"), c("NW", "SWU", "NW", "SWU"))
```



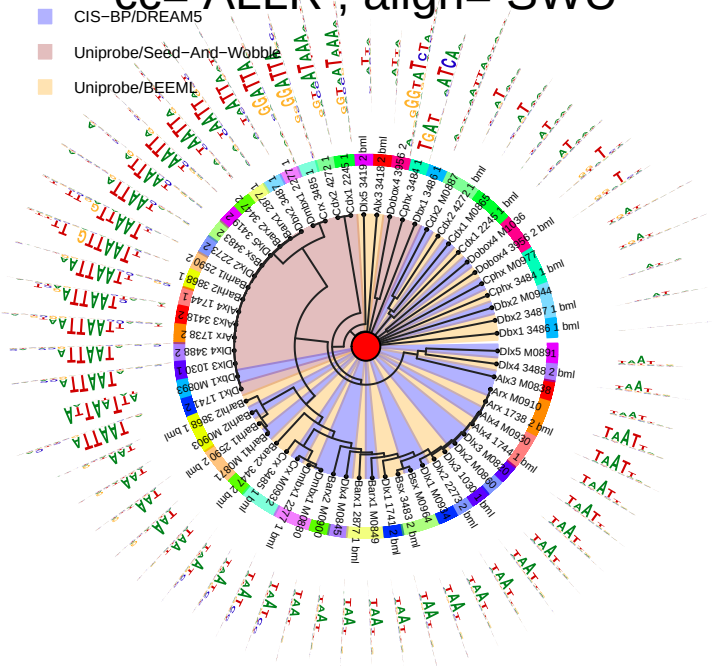
1:17; 2:3; 3:1



1:10; 2:9; 3:2



1:15; 2:6; 3:0



1:8; 2:13; 3:0

cluster by matAlign

MatAlign is another excellent software to produce motif distance. We could use MatAlign to produce a distance matrix and run Neighbor tree estimation program on this distance matrix to generate a tree. As it shown in following figure, algorithm to generating motifs from PBM data play a key role in motif clustering. DREAM5 and BEEML are more close compare to Seed-And-Wobble. MatAlign did well in seperate closer motifs. It is very clear that most of the motif which come from same PBM data and are genertaed by

DREAM5 and BEEML, are grouped together (background color of inner circle). And the tree generated by matAlign is very similar to the one by MotIV with cc=ALLR and align=SWU, because they are using same alignment methods.

The number (1:xx; 2:yy; 3:zz) in the bottom of the figure above and this figure is the count number of motifs by different methods in the same cluster. 1 means in that cluster, there is only one motif; 2 means 2 motifs are from 2 different algorithm; 3 means motifs are from 3 different algorithm. And xx, yy, zz is the count number in each category. By those numbers, it shows that SWU is better than NW and ALLR works better than PCC. And MatAlign works even better in this sample comparing to motIV.

Although this requires that phylip and matalign-v4a be installed, its performance is great in separating very close motifs such as homeodomain motifs. This document using homeodomain as example dataset, so all the analysis are done by MatAlign.

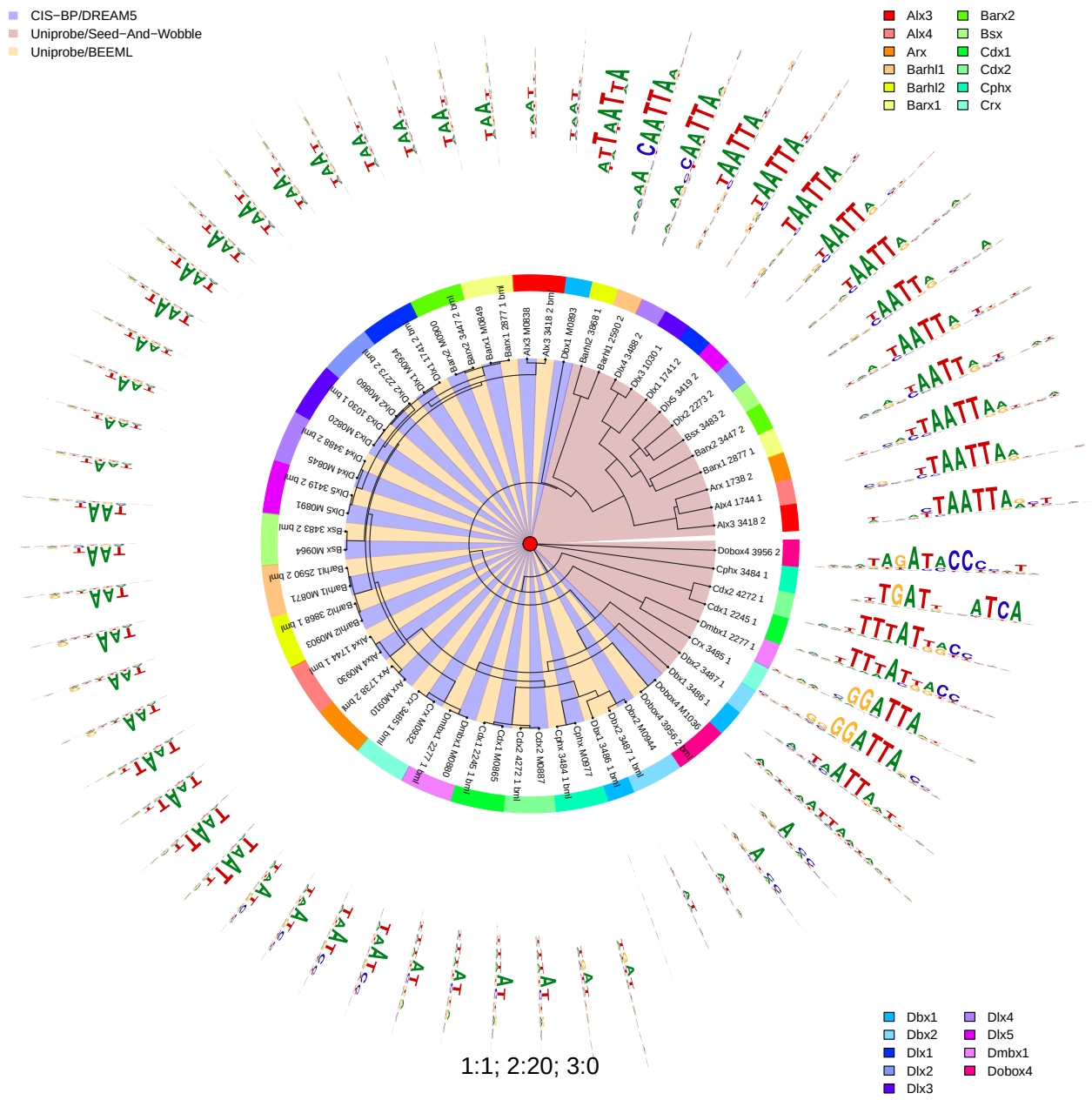
Supplementary Figure 2O

```
##read newick tree. Alignment is done by MatAlign (see http://stormo.wustl.edu/MatAlign/)
##and the newick tree is generated by Neighbor, which is a part of phylip (see
##http://evolution.genetics.washington.edu/phylip/progs.data.dist.html)
outpath <- "output"
matalign_path <- "./app/matalign-v4a"
neighbor_path <- "./app/neighbor.app/Contents/MacOS/neighbor"
system(paste("perl MatAlign2tree.pl --in . --pcmpath", pcmpath,
             "--out", outpath,
             "--matalign", matalign_path,
             "--neighbor", neighbor_path,
             "--tree","UPGMA"))
newickstrUPGMA <- readLines(con=file.path(outpath, "NJ.matalign.distMX.nwk"))
##convert to phylog object
phylogUPGMAmatAlign <- newick2phylog(newickstrUPGMA, FALSE)

##get the leaves of phylog to reorder the pfms
leaveNames <- names(phylogUPGMAmatAlign$leaves)
this_motifs <- pfms[leaveNames]

## data source
dataSource <- factor(grepl("_M", leaveNames))
levels(dataSource) <- c("yellow", "blue") ##c("Uniprobe", "CIS-BP")
dataSource <- as.character(dataSource)
## algorithm
algorithm <- factor((!grepl("_M", leaveNames)) + grepl("_bml", leaveNames))
levels(algorithm) <- c("blue", "brown", "orange") ##("DREAM5", "Seed-And-Wobble", "BEEML")
algorithm <- as.character(algorithm)
## motifs from same PBM data
motifGroup <- factor(gsub("(.*?)_.*$", "\\1", leaveNames))
levels.motifGroup <- levels(motifGroup)
levels(motifGroup) <- pairColor[1:length(levels(motifGroup))]
colors.motifGroup <- levels(motifGroup)
motifGroup <- as.character(motifGroup)

## draw the motifs
plotMotifStackWithRadialPhylog(phylog=phylogUPGMAmatAlign,
                               pfms=DNAmotifAlignment(this_motifs),
                               labels.leaves=leaveNames,
```

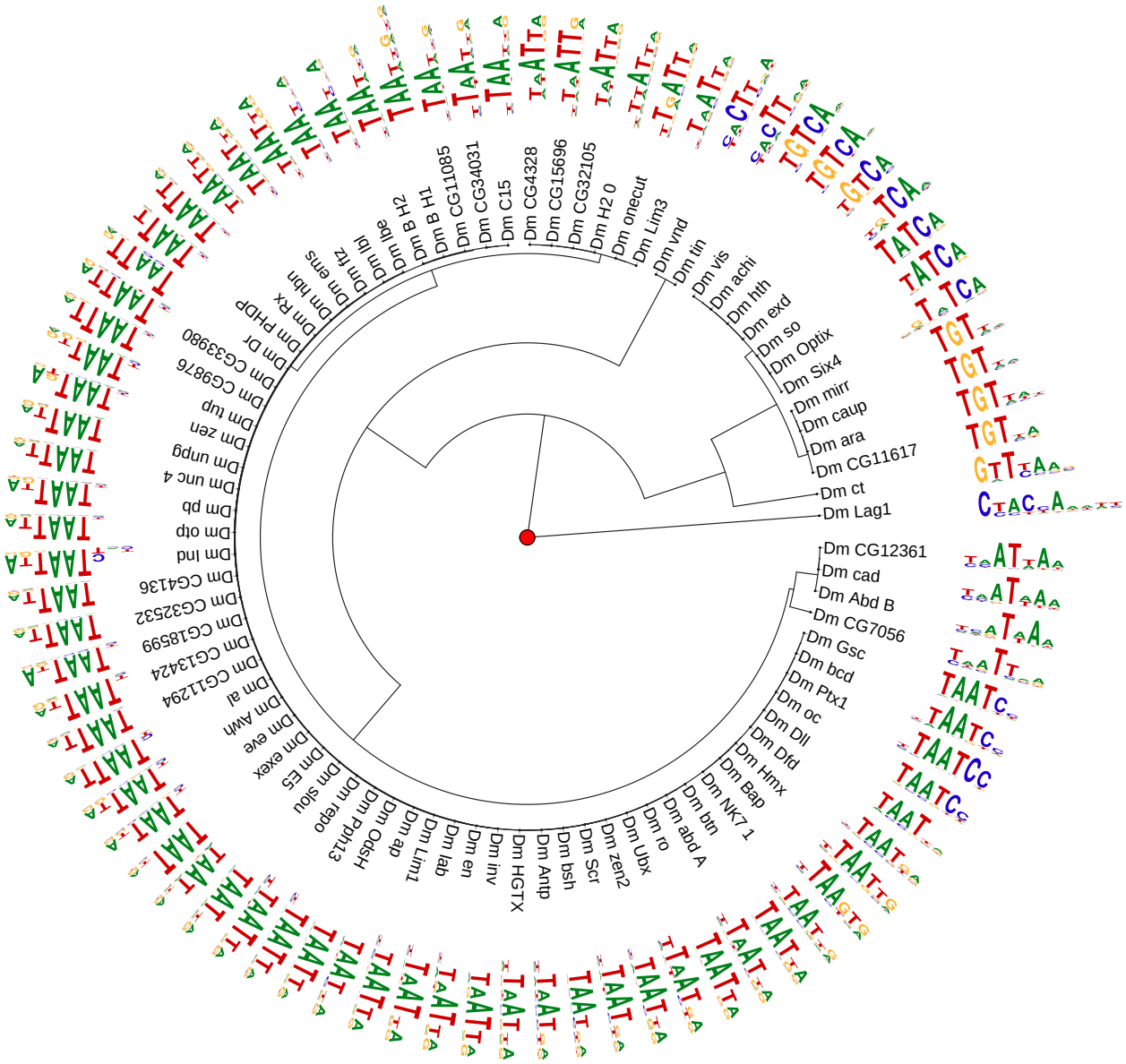
```
                                col.bg=algorithm,
                                col.bg.alpha=.3,
                                col.inner.label.circle=motifGroup,
                                inner.label.circle.width=.1,
                                cleaves=.2, circle=1.1,
                                circle.motif=1.6,
                                clabel.leaves=.6, angle=358)

legend(1.5, 2.4, legend=levels.motifGroup[1:12],
       fill= colors.motifGroup[1:12],
       border="black", lty=NULL, bty = "n", ncol=2, cex=.8)
legend(1.5, -2, legend=levels.motifGroup[13:21],
       fill= colors.motifGroup[13:21],
       border="black", lty=NULL, bty = "n", ncol=2, cex=.8)
legend(-2.35, 2.4,
       legend=c("CIS-BP/DREAM5", "Uniprobe/Seed-And-Wobble", "Uniprobe/BEEML"),
       fill= highlightCol(c("blue", "brown", "orange"), alpha=.3),
       border="white", lty=NULL, bty = "n", cex=.8)
cnt <- rle(motifGroup)
cnt <- split(algorithm, rep(1:length(cnt$lengths), cnt$lengths))
cnt <- table(sapply(cnt, function(.ele) length(unique(.ele))))
cnt.1 <- vector("integer", 3)
names(cnt.1) <- 1:3
cnt.1[names(cnt)] <- cnt
cnt.1["1"] <- (cnt.1["1"]+cnt.1["2"]*2+cnt.1["3"]*3)/3 - cnt.1["2"] - cnt.1["3"]
text(0, -2.3, label=paste(names(cnt.1), cnt.1, sep=":", collapse="; "), cex=1.5)
```

Supplementary Figure 2P. different distance resolution with different alignment methods

```
pcmpath <- "pcmsDatasetDM"
pcms <- readPCM(pcmpath)
pfms<-lapply(pcms,pcm2pfm)
motIVout <- getMotIVOut(pfms, "ALLR", "SWU")
plotMotifStackWithRadialPhylog(phylog=motIVout$phylog,
                                pfms=motIVout$aligned.pfms,
                                labels.leaves=motIVout$leaveNames,
                                cleaves=.2, circle=1.2, circle.motif=1.6,
                                clabel.leaves=1,
                                motifScale="logarithmic",
                                angle=358,
                                plotIndex=FALSE)
text(0, 2.4, label="motIV: cc=ALLR; align=SWU", cex=1.5)
```

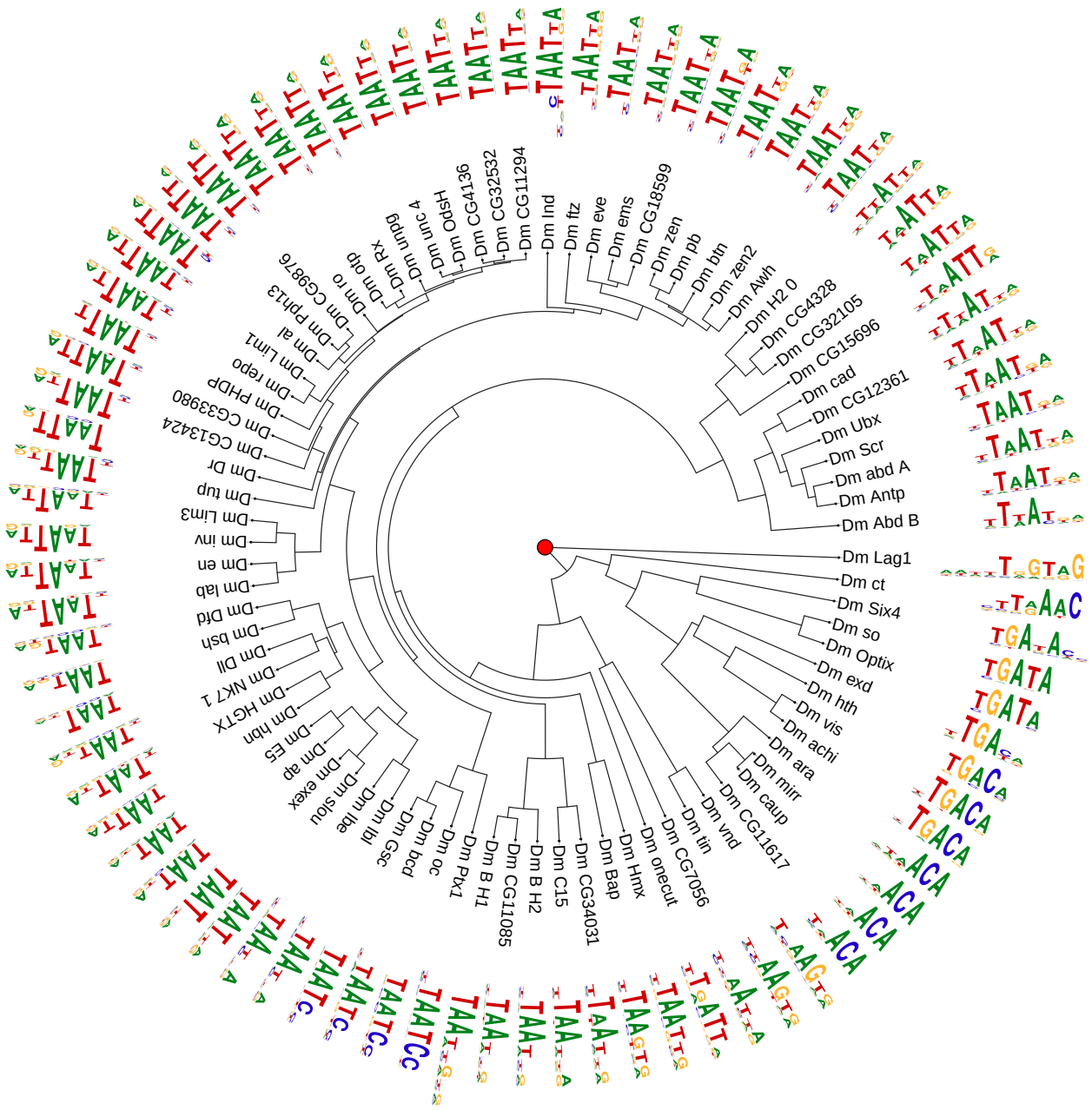


Supplementary Figure 2O: UPGMA tree by matAlign+PHYLIP




```
## function to read example data
getMatAlignOut <- function(pcmpath, outpath="output",
                           groupDistance=NA, trim=0.2){
    pcms <- readPCM(pcmpath)
    pfms<-lapply(pcms,pcm2pfm)
    matalign_path <- "./app/matalign-v4a"
    neighbor_path <- "./app/neighbor.app/Contents/MacOS/neighbor"
    system(paste("perl MatAlign2tree.pl --in . --pcmpath", pcmpath,
                 "--out", outpath,
                 "--matalign", matalign_path,
                 "--neighbor", neighbor_path,
                 "--tree","UPGMA"))
    newickstrUPGMA <-
        readLines(con=file.path(outpath, "NJ.matalign.distMX.nwk"))
```

```
    phylog <- newick2phylog(newickstrUPGMA, FALSE)
    leaves <- names(phylog$leaves)
    motifs <- pfms[leaves]
    if(!is.na(groupDistance)){
        motifSig <-
            motifSignature(motifs, phylog,
                           groupDistance=groupDistance,
                           min.freq=1, trim=trim)
        sig <- signatures(motifSig)
        gpCol <- sigColor(motifSig)
    }else{
        motifSig <- NA
        sig <- NA
        gpCol <- NA
    }

    return(list(phylog=phylog, sig=sig, gpCol=gpCol,
                motifs=DNAmotifAlignment(motifs),
                leaves=leaves,
                unaligned.pfms=motifs))
}

matAlignOut <- getMatAlignOut(pcmpath)
plotMotifStackWithRadialPhylog(phylog=matAlignOut$phylog,
                               pfms=matAlignOut$motifs,
                               labels.leaves=matAlignOut$leaves,
                               cleaves=.2, circle=1.2, circle.motif=1.6,
                               clabel.leaves=1, motifScale="logarithmic",
                               angle=358, plotIndex=FALSE)
text(0, 2.4, label="MatAlign", cex=1.5)
```




```
sessionInfo()
```

```
## R version 3.4.1 (2017-06-30)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: OS X El Capitan 10.11.6
##
## Matrix products: default
## BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
```

```
##
## attached base packages:
##  [1] stats4    parallel  grid      stats     graphics  grDevices utils
##  [8] datasets  methods   base
##
## other attached packages:
##  [1] motifStack_1.20.1   Biostrings_2.44.2   XVector_0.16.0
##  [4] IRanges_2.10.2      S4Vectors_0.14.3    ade4_1.7-6
##  [7] MotIV_1.32.0        BiocGenerics_0.22.0 grImport_0.9-0
## [10] XML_3.98-1.9
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.12               highr_0.6
##  [3] plyr_1.8.4                 compiler_3.4.1
##  [5] GenomeInfoDb_1.12.2        bitops_1.0-6
##  [7] tools_3.4.1                zlibbioc_1.22.0
##  [9] digest_0.6.12              evaluate_0.10.1
## [11] lattice_0.20-35            BSgenome_1.44.0
## [13] Matrix_1.2-10              DelayedArray_0.2.7
## [15] yaml_2.1.14                seqLogo_1.42.0
## [17] GenomeInfoDbData_0.99.0    rtracklayer_1.36.4
## [19] stringr_1.2.0              knitr_1.16
## [21] htmlwidgets_0.9            rprojroot_1.2
## [23] Biobase_2.36.2             BiocParallel_1.10.1
## [25] rGADEM_2.24.0              rmarkdown_1.6
## [27] magrittr_1.5               scales_0.4.1
## [29] backports_1.1.0            Rsamtools_1.28.0
## [31] htmltools_0.3.6            matrixStats_0.52.2
## [33] GenomicRanges_1.28.4       GenomicAlignments_1.12.1
## [35] SummarizedExperiment_1.6.3 colorspace_1.3-2
## [37] stringi_1.1.5              munsell_0.4.3
## [39] RCurl_1.95-4.8
```