Vingette: Bayesian network-based clustering with the R-package bnClustOmics

Simulated data

First we will look at how the package works with the simulated data example. The data was generated from 3 Bayesian networks. First, we create an abject of class 'bnInfo' that includes omics names and types.

```
library(bnClustOmics)
library(BiDAG)

bnnames<-bnInfo(simdata,c("b","c"),c("M","T"))
```

We proceed with runnig the function bnclustOmics that implements the EM algorithm. At each M-step of the algorithm the Bayesian network structures and parameters are learned using the Bayesian approach. We run the function bnclustOmics for different values of the number of clusters k. This code will take a while to run, ca.50 minutes.

```
bnres<-list()
for(k in 2:4) {
print(paste("running clustering for k=",k,sep=""))
bnres[[k]]<-bnclustOmics(simdata,bnnames,maxEM=4, kclust=k,
                                   startpoint = "mclustPCA")
}
```

Since, we know the ground truth, we can compare different clusterings to the true assignments using the function checkmembership.

```
#clustering accuracy
checkmembership(clusters(bnres[[2]]),simclusters)[c("ARI","precision","recall")]
```

```
## $ARI
## [1] 0.6122004
##
## $precision
## [1] 0.6371882
##
## $recall
## [1] 1
```

```
checkmembership(clusters(bnres[[3]]),simclusters)[c("ARI","precision","recall")]
```

```
## $ARI
## [1] 1
##
## $precision
## [1] 1
##
## $recall
## [1] 1
```

```
checkmembership(clusters(bnres[[4]]),simclusters)[c("ARI","precision","recall")]
```

```
## $ARI
## [1] 0.984027
##
## $precision
## [1] 1
##
## $recall
## [1] 0.9793594
```

In the absence of ground truth we can pick the number of clusters using either AIC or BIC. In the simulated examples both scores provide the correct estimated for k.

```
#the optimal number of clusters is 3 according to both AIC and BIC
chooseK(bnres,fun="BIC")$k
```

```
## [1] 3
```

```
chooseK(bnres,fun="AIC")$k
```

```
## [1] 3
```

To compare the discovered graphs to the ground truth, we need to relabel discovered labels, such that they align with exact cluster indices.

```
#to check the structure fit we first need to relabel according to
#corresponance between clustering labels
bnres[[3]]<-relabelSimulation(bnres[[3]],simclusters)

#compare MAP estimates to ground truth
compareDAGs(dags(bnres[[3]])[[1]],simdags[[1]])[c("TPR","FDR","SHD")]
```

```
##   TPR   FDR   SHD
##  0.78  0.55 71.00
```

```
compareDAGs(dags(bnres[[3]])[[2]],simdags[[2]])[c("TPR","FDR","SHD")]
```

```
##   TPR   FDR   SHD
##  0.68  0.61 97.00
```

```
compareDAGs(dags(bnres[[3]])[[3]],simdags[[3]])[c("TPR","FDR","SHD")]
```

```
##    TPR    FDR    SHD
##   0.63   0.67 106.00
```

The comparison above compared the MAP graphs to the ground truth. However, the output also includes posterior probabilities of all edges. Based on these probabilities, we can derive consensus models. When the data size is limited consensus models provide better fit (less false positive edges).

```
#threshold of 0.5
cons05<-getModels(bnres[[3]],p=0.5)

#compare consensus estimates (p=0.5) to ground truth
#TPR is better, SHD is better than for MAP graphs
compareDAGs(cons05[[1]],simdags[[1]])[c("TPR","FDR","SHD")]
```

```
##   TPR   FDR   SHD
##  0.78  0.43 50.00
```

```
compareDAGs(cons05[[2]],simdags[[2]])[c("TPR","FDR","SHD")]
```

```
##   TPR   FDR   SHD
##  0.73  0.45 61.00
```

```
compareDAGs(cons05[[3]],simdags[[3]])[c("TPR","FDR","SHD")]
```

```
##   TPR   FDR   SHD
##  0.69  0.55 77.00
```

```
#threshold of 0.9
cons09<-getModels(bnres[[3]],0.9)

#compare consensus estimates (p=0.9) to ground truth
#TPR is worse, but SHD is better than for MAP graphs and consensus graphs with p=0.5
compareDAGs(cons09[[1]],simdags[[1]])[c("TPR","FDR","SHD")]
```

```
##   TPR   FDR   SHD
##  0.57  0.13 32.00
```

```
compareDAGs(cons09[[2]],simdags[[2]])[c("TPR","FDR","SHD")]
```

```
##   TPR   FDR   SHD
##  0.47  0.22 45.00
```

```
compareDAGs(cons09[[3]],simdags[[3]])[c("TPR","FDR","SHD")]
```

```
##   TPR   FDR   SHD
##  0.42  0.32 50.00
```

Thresholding can inflate the differences between DAG representing discovered clusters. We can use a flexible threshold to compare the presence and abscence of certain edges in the networks representing the discovered clusters. The function annotateEdges can be used to make a list of annotated interactions, while the set of parameters sump, minp and minkp defines a set of flexible thresholds:

```
allInteractions<-annotateEdges(bnres[[3]],bnnames,sump=1.2,minp=0.5,minkp=0.9,dblist=simint)
```

```
## [1] 100
```

```
head(allInteractions)
```

```
##   from  to type1 type2 gene1 gene2    db        pcl1      pcl2      pcl3
## 1   M1  T4     M     T    M1    T4 FALSE 0.249687890 0.8027466 0.4681648
## 2   M1  T7     M     T    M1    T7 FALSE 0.815230961 0.2983770 0.1947566
## 3   M1 T12     M     T    M1   T12 FALSE 0.007490637 0.9575531 0.3945069
## 4   M1 T24     M     T    M1   T24 FALSE 0.347066167 0.5730337 0.3395755
## 5  M16 T43     M     T   M16   T43  TRUE 1.000000000 0.8039950 0.0000000
## 6  M20 T13     M     T   M20   T13 FALSE 0.000000000 0.0000000 0.9500624
```

We can visualize neighbourhoods of specific nodes in all clusters using the function plotNode.

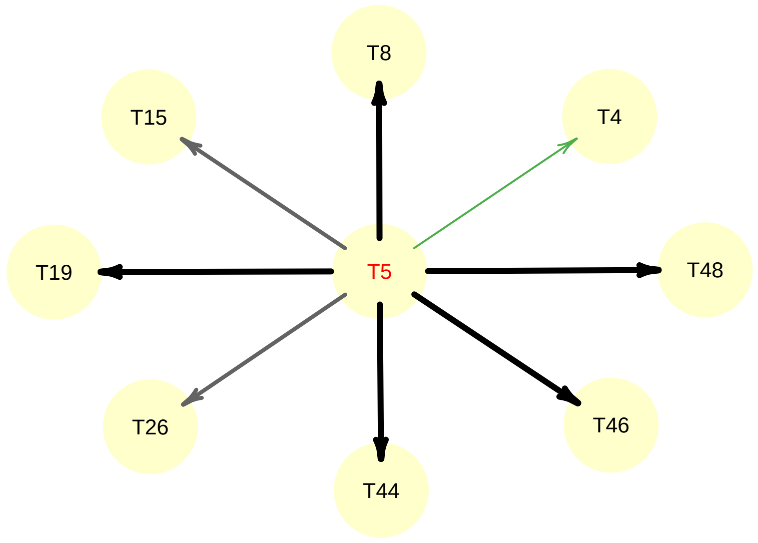
```
node1<-names(table(allInteractions$from)[order(table(allInteractions$from),decreasing = TRUE)[1]])
node2<-names(table(allInteractions$to)[order(table(allInteractions$to),decreasing = TRUE)[1]])

#number of all annotated interactions
nrow(allInteractions)
```

```
## [1] 100
```

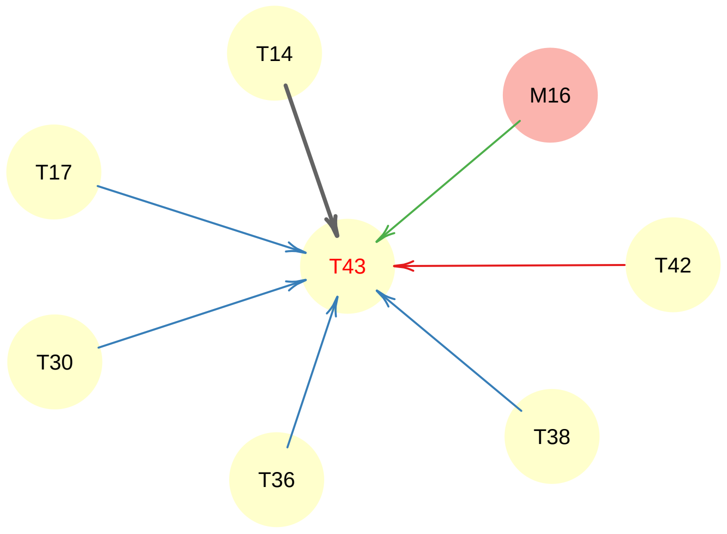
```
#number of true positives
length(which(allInteractions$db==TRUE))
```

```
## [1] 62
```

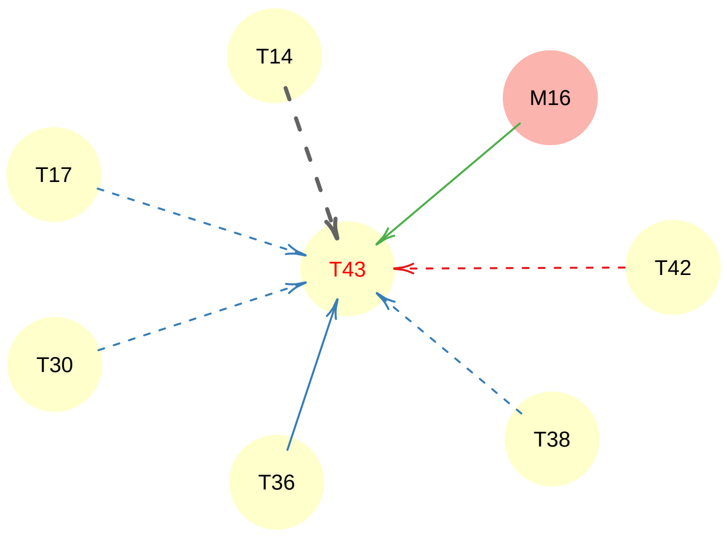
```
#plotting neighborhoods of node "T5"
plotNode(allInteractions,node1,p=0.5,cex=0.7,r=7,dbcheck=FALSE)
#check if interaction is in the ground truth graphs
#dashed lines for interactions not found in DB
plotNode(allInteractions,node1,p=0.5,cex=0.7,r=7,dbcheck=TRUE)
```



```
#plotting neighborhoods of node "T43"
plotNode(allInteractions,node2,p=0.5,cex=0.7,dbcheck=FALSE)
```



```
#check if interaction is in the ground truth graphs
#dashed lines for interactions not found in DB
plotNode(allInteractions,node2,p=0.5,cex=0.7,dbcheck=TRUE)
```



Biological data example.

Next we consider an example containing real biological data from 5 omics views from 45 patients with hepatocellular carcinoma. For each omics type, only a small subset of features was chosen for testing examples. The available omics types include: mutations (M), copy number changes (VN), transcriptome (T), proteome (P) and phosphoproteome (PP).

```
data(toydata)
dim(toydata$M)
```

```
## [1] 45 20
```

```
head(toydata$M)
```

```
##    CTNNB1 TP53 TTN PCLO ARID1A MUC16 OBSCN FLG CSMD3 SYNE1 ALB APOB HMCN1 MUC4 LRP1B
## S1      1    1   0    0      0     0     0   0     1     0   0    0     0    0     0
## S2      1    0   0    1      0     0     1   0     1     0   0    0     0    0     0
## S3      0    1   1    0      0     0     0   1     0     0   0    0     0    0     1
## S4      0    1   0    0      0     1     0   0     1     0   0    0     0    0     0
## S5      1    0   0    1      0     0     0   0     0     0   0    1     0    0     0
## S6      0    0   0    0      0     0     0   1     0     0   0    0     0    0     0
##    XIRP2 GPR98 HYDIN CSMD2 SDK1
## S1     0     0     0     0    0
## S2     0     0     0     1    0
## S3     0     0     0     0    1
## S4     0     0     0     0    0
## S5     0     0     0     0    0
## S6     0     0     0     0    0
```

```
dim(toydata$P)
```

```
## [1] 45 15
```

```
head(toydata$P)
```

```
##         P32754     P15088    P17677    P00326    Q93088    P08319     O75452
## S1   0.4063348 -0.9394057 0.2296135 -1.525890 -1.223183 -2.060413 -0.2513989
## S2  -1.1271807 -2.3592322 0.1663576 -1.321487 -5.852683 -1.341005 -1.4410522
## S3  -1.8876228 -2.9731484 -1.1454142 -3.491029  -3.476569  -3.570379 -2.2744578
## S4 -10.0483785 -9.1801867 -7.2548215 -8.719993 -11.864747 -10.274290 -7.7360363
## S5   0.2741213 -2.8256565 -0.6674648 -3.288038  -2.837974  -1.875802 -2.3672726
## S6  -6.6748416 -5.7811617 -3.8081823 -6.947549  -6.196801  -9.018637 -5.9268258
##        P11712      O14756     O95954     Q02928     Q9BTE3     Q9UKU0    Q9NVS2
## S1 -0.6738931  0.07931667  -2.339037 -1.2333577  0.2936188 -0.6796365  1.800419
## S2  0.2610725  0.32332941  -1.738512 -0.7675631 -0.6776561 -2.7786333  1.840714
## S3 -6.1387883 -0.61125299  -3.863853 -3.1894577 -6.0161427 -2.6590989 -2.795154
## S4 -8.5906561 -6.80205070 -10.797287 -7.3892539 -1.6437140 -6.9274677  1.755358
## S5 -0.4944224 -1.16190261  -3.196476 -0.7212430 -0.1363086 -0.5084004  2.369478
## S6 -8.5070996 -6.22982326  -5.819780 -8.8933665  0.2477952 -7.6375869  2.036001
##        Q86TV6
## S1  1.9877599
## S2 -2.6121088
## S3 -0.5506275
## S4 -0.1478331
## S5  1.1663397
## S6  3.3419922
```

```
#...etc
```

For constructing prior correctly, we will need to map feature ID from each omics type to a unique identifier, e.g. a gene symbol. For example ENSEMBLE is often use for transcriptome and UNIPROT for proteome for co structing blacklists and penalization matrices it is important to pass mappings of all IDs to gene names just one column is needed "gene", the rownames have to be similar to gene IDs within each omics type. We pass a list of matrices containing such mappings for each omics type to function bnInfo.

```
data(mappings)
head(mappings[["M"]])
```

```
##        genenu   gene
## HMCN1  "HMCN1"  "HMCN1"
## OBSCN  "OBSCN"  "OBSCN"
## HYDIN  "HYDIN"  "HYDIN"
## CTNNB1 "CTNNB1" "CTNNB1"
## TP53   "TP53"   "TP53"
## MUC4   "MUC4"   "MUC4"
```

```
head(mappings[["PP"]])
```

```
##             uniprot_site    gene site
## O75643_S225  O75643_S225 SNRNP20 S225
## Q99442_T375  Q99442_T375   SEC62 T375
## O95218_S153  O95218_S153  ZRANB2 S153
## P69905_S132  P69905_S132    HBA1 S132
## Q9NYF8_S397  Q9NYF8_S397  BCLAF1 S397
## Q7Z417_S214  Q7Z417_S214  NUFIP2 S214
```

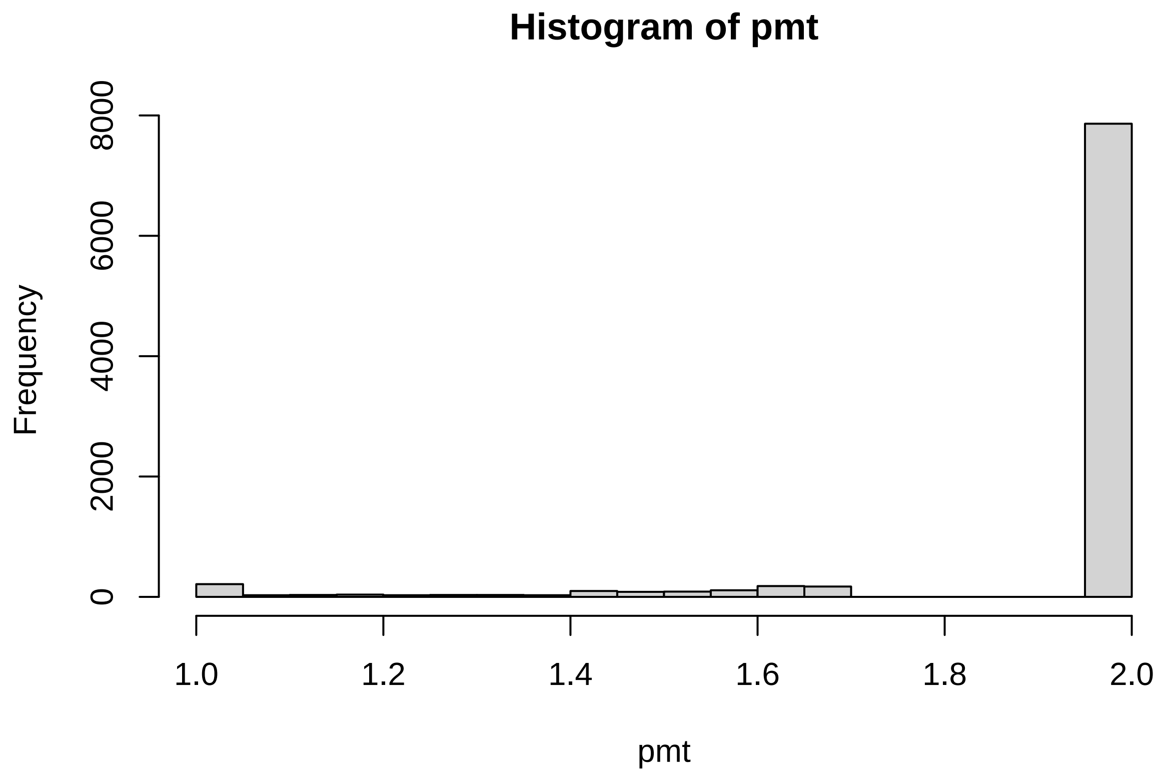
```
bnnames<-bnInfo(toydata,c("b","o","c","c","c"),c("M","CN","T","P","PP"),
                 mappings)
```

We proceed with constructing blackist and penalization matrices. The latest defines a graphical prior which can be used to give advantage to structures containing the edges corresponding to interaction found in the interaction databases (prior knowledge). The parameters pfbase, intpf and intsame define how different edges will be penalized; intpf=1, means we do not penalize interactions that are found in the databas; intsame=1, we do not penalize interactions between nodes representing the same gene but different omics types; pfbase=2, we penalize edges by a factor of 2 if th are not found in the database.If we want to use interaction scores to inform penalization factors, we will need to defind the upper bound of intpf, say 2, then the penalization factor is defines as 2-2*interactions_score:

```
#read the file containing interactions from the string database (prior information)
data(stringint)
head(stringint)
```

```
##   gene1 gene2 score
## 1  A1BG  PON1 0.173
## 2  A1BG   AFM 0.398
## 3  A1BG APOC3 0.330
## 4  A1BG  APOB 0.225
## 5  A1BG   TTR 0.459
## 6  A1BG  TP53 0.150
```

```
pmt<-penInit(bnnames,pfbase=2,intpf=2,intlist=stringint,intsame = 1, usescore=TRUE)
hist(pmt)
```



```
#blt<-pmt-1 #blacklist all non-string interactions


#initialize blacklist (optional)
#we blacklist the edges between variables of type "T" (transcriptome): intra=c("T")

#we blacklist the edges from variables representing gene X of type "P"
#to variables of type "T" representing the same gene: interXX=list(from=c("P"),to=c("T"))
#note that the edges in the other directions are allowed

#we blacklist the edges from variables representing gene X of type "CN"
#to variables of type "T", "P" and "PP" representing the gene Y:
#interXY=list(from=c("CN","CN","CN"),to=c("T","P","PP"))
```

```
blt<-blInit(bnnames,intra=c("T"),interXX=list(from=c("P"),to=c("T")),
            interXY=list(from=c("CN","CN","CN"),to=c("T","P","PP")))
```

We run the clustering using penalization and blacklist matrices:

We can further inspect MAP and consensus models, annotate interactions and plot neighborhoods of the nodes of interest.

```
#look at consensus networks for different threshold.
cons01<-getModels(bnres,0.1)
cons05<-getModels(bnres,0.5)

compareDAGs(cons01[[1]],cons01[[2]])
```

```
##     TP     FP     FN    TPR    FPR   FPRn    FDR    SHD
##  59.00  96.00 123.00   0.32   0.02   0.53   0.62 239.00
```

```
compareDAGs(cons05[[1]],cons05[[2]])
```

```
##     TP     FP     FN    TPR    FPR   FPRn    FDR    SHD
##  17.00  72.00  86.00   0.17   0.02   0.70   0.81 167.00
```

```
#annotate all edges
allInteractions<-annotateEdges(bnres,bnnames,sump=1.2,minp=0.4,minkp=0.9,dblist=stringint)
```

```
## [1] 79
```

```
head(allInteractions)
```

```
##     from          to type1 type2  gene1  gene2    db       pcl1       pcl2
## 1 CTNNB1      O75452     M     P CTNNB1  RDH16 FALSE 0.91885144 0.17103620
## 2    TTN      O75452     M     P    TTN  RDH16 FALSE 0.94007491 0.08489388
## 3   PCLO P35659_S244     M    PP   PCLO    DEK FALSE 0.49563046 0.91885144
## 4  MUC16      P11712     M     P  MUC16 CYP2C9 FALSE 0.19850187 0.99001248
## 5  OBSCN      Q93088     M     P  OBSCN   BHMT FALSE 0.07490637 0.97003745
## 6    FLG      Q9BTE3     M     P    FLG  MCMBP FALSE 0.91385768 0.07990012
```

```
nrow(allInteractions)
```

```
## [1] 79
```

```
#number of interactions for in the database
length(which(allInteractions$db))
```

```
## [1] 16
```

```
allInteractions[allInteractions$db,]
```

```
##                 from              to type1 type2   gene1   gene2   db        pcl1
## 20 ENSG00000198650          O95954     T     P     TAT    FTCD TRUE 0.002496879
## 22          P32754          P08319     P     P     HPD    ADH4 TRUE 0.903870162
## 37          P00326          O75452     P     P   ADH1C   RDH16 TRUE 0.210986267
## 38          P00326          O95954     P     P   ADH1C    FTCD TRUE 0.963795256
## 43          O95954          Q02928     P     P    FTCD CYP4A11 TRUE 0.012484395
## 45          Q02928 ENSG00000138109     P     T CYP4A11  CYP2C9 TRUE 0.732833958
## 46          Q02928          P11712     P     P CYP4A11  CYP2C9 TRUE 0.327091136
## 47          Q02928          Q9UKU0     P     P CYP4A11   ACSL6 TRUE 0.957553059
## 60     P35659_S244     O43719_S642    PP    PP     DEK HTATSF1 TRUE 0.962546816
## 61     P35659_S244     O14647_S208    PP    PP     DEK    CHD2 TRUE 0.013732834
## 66     P69905_S132    P11277_S2060    PP    PP    HBA1     SPTB TRUE 0.023720350
## 67       O95400_S49     Q9Y5J1_S124    PP    PP  CD2BP2    UTP18 TRUE 0.082397004
## 69     P11277_S2060      P68871_Y36    PP    PP    SPTB      HBB TRUE 0.976279650
## 70     P11277_S2060     P69905_S132    PP    PP    SPTB     HBA1 TRUE 0.965043695
## 71      O14647_S208     P35659_S244    PP    PP    CHD2      DEK TRUE 0.986267166
## 76      O95218_S153     Q9NYF8_S397    PP    PP  ZRANB2   BCLAF1 TRUE 0.908863920
##             pcl2
## 20 0.976279650
## 22 0.424469413
## 37 0.990012484
## 38 0.998751561
## 43 0.997503121
## 45 0.943820225
## 46 0.925093633
## 47 0.097378277
## 60 0.034956305
## 61 0.970037453
## 66 0.990012484
## 67 0.990012484
## 69 0.007490637
## 70 0.009987516
## 71 0.029962547
## 76 0.911360799
```

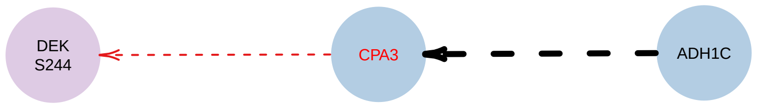
```
#number of interactions between nodes representing the same genes
length(which(allInteractions$gene1==allInteractions$gene2))
```

```
## [1] 5
```

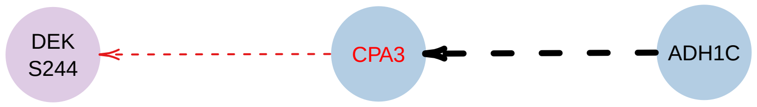
```
allInteractions[allInteractions$gene1==allInteractions$gene2,]
```

```
##                 from          to type1 type2  gene1  gene2    db       pcl1      pcl2
## 14 ENSG00000158104      P32754     T     P    HPD    HPD FALSE 0.05493134 1.0000000
## 19 ENSG00000198099      P08319     T     P   ADH4   ADH4 FALSE 0.28589263 1.0000000
## 21 ENSG00000138109      P11712     T     P CYP2C9 CYP2C9 FALSE 0.98751561 0.9912609
## 63       P68871_T88 P68871_T13    PP    PP    HBB    HBB FALSE 0.06741573 0.9650437
## 64       P68871_T88 P68871_Y36    PP    PP    HBB    HBB FALSE 0.03245943 0.9887640
```

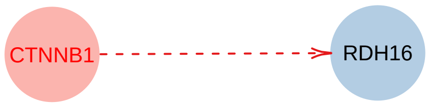
```
#plot node neighborhood
plotNode(allInteractions,"P15088")
```



```
#different threshold and font size
plotNode(allInteractions,"P15088",p=0.3,cex=0.7)
```



```
#different node
node<-"CTNNB1"
plotNode(allInteractions,node,p=0.3,cex=0.7)
```


CTNNB1
RDH16