Epistatic Nested Effects Models
Inferring mixed epistatis from indirect measurements of knockout screens.

Madeline Diekmann & Martin Pirkl

December 16, 2016

This package is an extension of the classic Nested Effects Models provided in package *nem*. Nested Effects Models is a pathway reconstruction method, which takes into account effects of downstream genes. Those effects are observed for every knockout of a pathway gene, and the nested structure of observed effects can then be used to reconstruct the pathway structure. However, classic Nested Effects Models do not account for double knockouts. In this package *epiNEM*, one additional layer of complexity is added. For every two genes, acting on one gene together, the relationship is evaluated and added to the model as a logic gate. Genetic relationships are represented by the logics OR (no relationship), AND (functional overlap), NOT (masking or inhibiting) and XOR (mutual prevention from acting on gene C).

Loading epiNEM

```
## install.packages("devtools", verbose = F, quiet = T)

## library(devtools)

## install_github("cbg-ethz/epiNEM", quiet = T)

library(epiNEM)
```

Simulations

We compare epiNEM to several network inference methods.

```
library(bnem, quietly = T, verbose = F) #  install_github("MartinFXP/B-NEM/package")

##
## Attaching package:  'BiocGenerics'
## The following objects are masked from 'package:parallel':
##
##    clusterApply, clusterApplyLB, clusterCall, clusterEvalQ, clusterExport,
##    clusterMap, parApply, parCapply, parLapply, parLapplyLB, parRapply,
##    parSapply, parSapplyLB
## The following objects are masked from 'package:igraph':
##
##    normalize, union
```

```
## The following objects are masked from 'package:stats':
##
##    IQR, mad, xtabs
## The following objects are masked from 'package:base':
##
##    Filter, Find, Map, Position, Reduce, anyDuplicated, append, as.data.frame,
##    cbind, colnames, do.call, duplicated, eval, evalq, get, grep, grepl,
##    intersect, is.unsorted, lapply, lengths, mapply, match, mget, order, paste,
##    pmax, pmax.int, pmin, pmin.int, rank, rbind, rownames, sapply, setdiff,
##    sort, table, tapply, union, unique, unsplit, which, which.max, which.min
##
## Attaching package:  'graph'
## The following objects are masked from 'package:igraph':
##
##    degree, edges, intersection
##
## Attaching package:  'RBGL'
## The following objects are masked from 'package:igraph':
##
##    bfs, dfs, transitivity
## The following object is masked from 'package:e1071':
##
##    extractPath
## hash-2.2.6 provided by Decision Patterns
## Warning:  package 'ggplot2' was built under R version 3.3.2
## Warning:  package 'XML' was built under R version 3.3.2
##
## Attaching package:  'XML'
## The following object is masked from 'package:graph':
##
##    addNode
##
## Attaching package:  'nem'
## The following object is masked from 'package:RBGL':
##
##    transitive.closure
##
## Attaching package:  'snow'
## The following objects are masked from 'package:BiocGenerics':
##
##    clusterApply, clusterApplyLB, clusterCall, clusterEvalQ, clusterExport,
##    clusterMap, clusterSplit, parApply, parCapply, parLapply, parRapply,
##    parSapply
## The following objects are masked from 'package:parallel':
##
##    clusterApply, clusterApplyLB, clusterCall, clusterEvalQ, clusterExport,
##    clusterMap, clusterSplit, makeCluster, parApply, parCapply, parLapply,
##    parRapply, parSapply, splitIndices, stopCluster
##
## Attaching package:  'latticeExtra'
## The following object is masked from 'package:ggplot2':
##
##    layer
```

```
library(nem)

library(minet)

library(pcalg)
```

```
runs <- 100

noiselvls <- c(0.01, 0.025, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5)

random <- list(FPrate = 0.1, FNrate = noiselvls,
               single = 4, double = 1, reporters = 100, replicates = 3)

spec <- sens  <- logics <- array(0, dim = c(2, runs, length(noiselvls)))

sens2 <- spec2 <- time <- array(0, dim = c(5, runs, length(noiselvls)))

do <- c("n", "p", "a")

do <- c("e", "b", do)

popSize <- 100

maxTime <- F

forcelogic <- T

epinemsearch <- "greedy"

nIterations <- 3

bnemsearch <- "genetic"

parallel <- NULL

logicgate <- matrix("", runs, length(noiselvls))

edgenr <- matrix(0, runs, length(noiselvls))

## for (i in 1:runs) {

##     print(paste("run ", i, sep = ""))

##     for (j in 1:length(noiselvls)) {

##         print(paste("noiselvl ", j, sep = ""))

##         topology <- CreateTopology(random$single, random$double, force = forcelogic)

##         topology <- unlist(unique(topology), recursive = FALSE)

##         extTopology <- ExtendTopology(topology$model, random$reporters)
```

```
##         sortedData <- GenerateData(topology$model, extTopology,
##                                    random$FPrate, random$FNrate[j], random$replicates)

##         logicgate[i, j] <- paste(topology$logics, collapse = "_")

##         edgenr[i, j] <- sum(topology$origModel == 1)

##         if ("e" %in% do) {
##             print("epiNEM")

##             start <- Sys.time()
##             TriplModel <- epiNEM(filename = sortedData,
##                       method = epinemsearch, nIterations = nIterations)
##             time[1, i, j] <- difftime(Sys.time(), start, units = "secs")
##             print(time[1, i, j])

##             tp <- sum(topology$model == 1 & TriplModel$model == 1)
##             tn <- sum(topology$model == 0 & TriplModel$model == 0)
##             fp <- sum(topology$model == 0 & TriplModel$model == 1)
##             fn <- sum(topology$model == 1 & TriplModel$model == 0)
##             sens[1, i, j] <- tp/(tp+fn)
##             spec[1, i, j] <- tn/(tn+fp)
##             tp <- sum(topology$origModel == 1 & TriplModel$origModel == 1)
##             tn <- sum(topology$origModel == 0 & TriplModel$origModel == 0)
##             fp <- sum(topology$origModel == 0 & TriplModel$origModel == 1)
##             fn <- sum(topology$origModel == 1 & TriplModel$origModel == 0)
##             sens2[1, i, j] <- tp/(tp+fn)
##             spec2[1, i, j] <- tn/(tn+fp)
##             tp <- 0
##             for (k in 1:length(topology$column)) {
##                 for (l in 1:length(TriplModel$column)) {
##                     if (topology$column[k] == TriplModel$column[l]) {
##                         if (topology$logics[k] %in% TriplModel$logics[l]) {
##                             tp <- tp + 1
##                         }
##                     }
##                 }
##             }
##             logics[1, i, j] <- tp/(length(topology$logics) +
##                                        length(TriplModel$logics) - tp)
##             print(sens[1, i, j])
##             print(spec[1, i, j])
##             print(sens2[1, i, j])
##             print(spec2[1, i, j])
##             print(logics[1, i, j])

##         }

##         if ("b" %in% do) {
##             print("B-NEM")

##             gtn <- epi2bg(topology)
```

```
##             fc <- cbind(Ctrl_vs_S = -1, epi2bg(sortedData))*(-1)

##             bnemnoise <- sample(1:nrow(fc), floor(nrow(fc)*random$FNrate[j]))

##             fc[bnemnoise, 1] <- 0

##             ers <- t(topology$model)*(-1)
##             colnames(ers) <- paste("S_vs_S_",
##                                    gsub("\\.", "_", colnames(ers)), sep = "")
##             ers <- cbind(Ctrl_vs_S = 1, ers)
##             ers <- ers[, order(colnames(ers))]

##             CNOlist <- dummyCNOlist(stimuli = "S",
##                                     inhibitors = LETTERS[1:random$single],
##                                     maxStim = 1, maxInhibit = 2,
##                                     signals = LETTERS[1:random$single])

##             parents <- unique(unlist(strsplit(colnames(sortedData)[grep("\\.",
##                                     colnames(sortedData))], "\\.")))

##             nodes <- unique(colnames(sortedData)[-grep("\\.", colnames(sortedData))])

##             child <- nodes[-which(nodes %in% parents)]

##             sifMatrix <- NULL
##             for (k in LETTERS[1:random$single]) {
##                sifMatrix <- rbind(sifMatrix, c("S", "1", k))#, c("S", "-1", k))
##                 for (l in LETTERS[1:random$single]) {
##                     if (k %in% l) { next() }
##                     if (k %in% parents) {
##                         sifMatrix <- rbind(sifMatrix, c(k, "1", l), c(k, "-1", l))
##                     } else {
##                         sifMatrix <- rbind(sifMatrix, c(k, "1", l))
##                     }

##             randfile <- paste("pkn_", as.numeric(Sys.time()), sep = "")
##             write.table(sifMatrix, file = randfile, sep = "\t",
##                         row.names = FALSE, col.names = FALSE, quote = FALSE)
##             PKN <- readSIF(randfile)
##             unlink(randfile)

##             model <- preprocessing(CNOlist, PKN)

##             initBstring <- absorption(rep(1, length(model$reacID)), model)

##             if (maxTime) { maxTime2 <- time[1, i, j] } else { maxTime2 <- Inf }

##             start <- Sys.time()
##             bga <- bnem(search = bnemsearch,
##                         fc=fc,
##                         CNOlist=CNOlist,
##                         model=model,
##                         initBstring=initBstring,
```

```
##                             draw = F,
##                             verbose = F,
##                             popSize = popSize,
##                             maxTime = maxTime2,
##                             parallel = parallel
##                             )
##             time[2, i, j] <- difftime(Sys.time(), start, units = "secs")
##             print(time[2, i, j])

##             ers2 <- computeFc(CNOlist, t(simulateStatesRecursive(CNOlist,
##                                                                  model, bga$bString)))
##             ers2 <- ers2[, unique(colnames(fc))]
##             ers2 <- ers2[, order(colnames(ers2))]

##             tp <- sum(ers == -1 & ers2 == -1)
##             tn <- sum(ers == 0 & ers2 == 0)
##             fn <- sum(ers == -1 & ers2 == 0)
##             fp <- sum(ers == 0 & ers2 == -1)
##             sens[2, i, j] <- tp/(tp+fn)
##             spec[2, i, j] <- tn/(tn+fp)
##             gtn2 <- abs(dnf2adj(gtn))
##             if (length(grep("S", rownames(gtn2))) > 0) {
##                 gtn2 <- gtn2[-grep("S", rownames(gtn2)), -grep("S", colnames(gtn2))]
##             }
##             gtn2 <- gtn2[order(rownames(gtn2)), order(colnames(gtn2))]
##             res <- abs(dnf2adj(bga$graph))
##             if (length(grep("S", rownames(res))) > 0) {
##                 res <- as.matrix(res[-grep("S", rownames(res)),
##                                                  -grep("S", colnames(res))])
##             }
##             if (dim(res)[1] == 1) {
##                 colnames(res) <- rownames(res) <- gsub(".*=", "", bga$graph)
##             } else {
##                 res <- res[order(rownames(res)), order(colnames(res))]
##             }
##             if (nrow(res) < nrow(gtn2)) {
##                 res2 <- rbind(cbind(res, matrix(0, nrow(res), nrow(gtn2) - nrow(res))),
##                               matrix(0, nrow(gtn2) - nrow(res), ncol(gtn2)))
##                 colnames(res2)[(ncol(res)+1):ncol(res2)] <-
##                                         colnames(gtn2)[which(!(colnames(gtn2)
##                                         %in% colnames(res)))]
##                 rownames(res2)[(nrow(res)+1):nrow(res2)] <-
##                                         rownames(gtn2)[which(!(rownames(gtn2)
##                                         %in% rownames(res)))]
##                 res2 <- res2[order(rownames(res2)), order(colnames(res2))]
##                 res <- res2
##             }
##             diag(gtn2) <- diag(res) <- 0
##             tp <- sum(gtn2 == 1 & res == 1)
##             tn <- sum(gtn2 == 0 & res == 0)
##             fn <- sum(gtn2 == 1 & res == 0)
##             fp <- sum(gtn2 == 0 & res == 1)
##             sens2[2, i, j] <- tp/(tp+fn)
##             spec2[2, i, j] <- tn/(tn+fp)
```

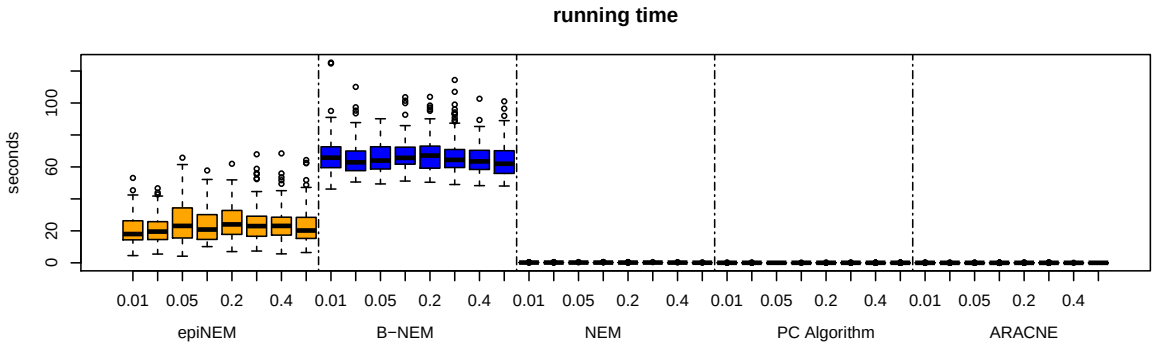
```
##             tp <- sum(bga$graph %in% gtn)
##             logics[2, i, j] <- tp/(length(gtn) + length(bga$graph) - tp)
##             print(sens[2, i, j])
##             print(spec[2, i, j])
##             print(sens2[2, i, j])
##             print(spec2[2, i, j])
##             print(logics[2, i, j])

##             print(bga$graph)
##             print(gtn)

##         }

##         if (any(c("n", "p", "a") %in% do)) {

##             reddata <- sortedData[, -grep("\\.", colnames(sortedData))]
##             gtnadj <- topology$origModel
##             gtnadj <- gtnadj[order(apply(gtnadj, 1, sum), decreasing = T),
##                                     order(apply(gtnadj, 2, sum), decreasing = F)]
##             gtnadj[lower.tri(gtnadj)] <- gtnadj[upper.tri(gtnadj)]
##             gtnadj <- gtnadj[order(rownames(gtnadj)), order(colnames(gtnadj))]
##             eadj <- topology$origModel
##             eadj <- eadj[order(rownames(eadj)), order(colnames(eadj))]
##             reddata2 <- matrix(0, nrow(reddata)*random$replicates,
##                                      length(unique(colnames(reddata))))
##             for (k in 1:length(unique(colnames(reddata)))) {
##                 reddata2[, k] <- as.vector(reddata[, which(colnames(reddata) %in%
##                                                    unique(colnames(reddata))[k])])
##             }
##             colnames(reddata2) <- unique(colnames(reddata))

##         }

##         if ("n" %in% do) {
##             print("NEM")

##             start <- Sys.time()
##             if (epinemsearch %in% "greedy") {
##                 nemres <- nem(reddata, inference = "nem.greedy")
##             } else {
##                 nemres <- nem(reddata, inference = "search")
##             }
##             nadj <- transitive.reduction(graph2adj(nemres$graph))
##             time[3, i, j] <- difftime(Sys.time(), start, units = "secs")
##             print(time[3, i, j])

##             tp <- sum(eadj  == 1 & nadj == 1)
##             tn <- sum(eadj == 0 & nadj == 0)
##             fp <- sum(eadj == 0 & nadj == 1)
##             fn <- sum(eadj == 1 & nadj == 0)
##             sens2[3, i, j] <- tp/(tp+fn)
##             spec2[3, i, j] <- tn/(tn+fp)
##             print(sens2[3, i, j])
```

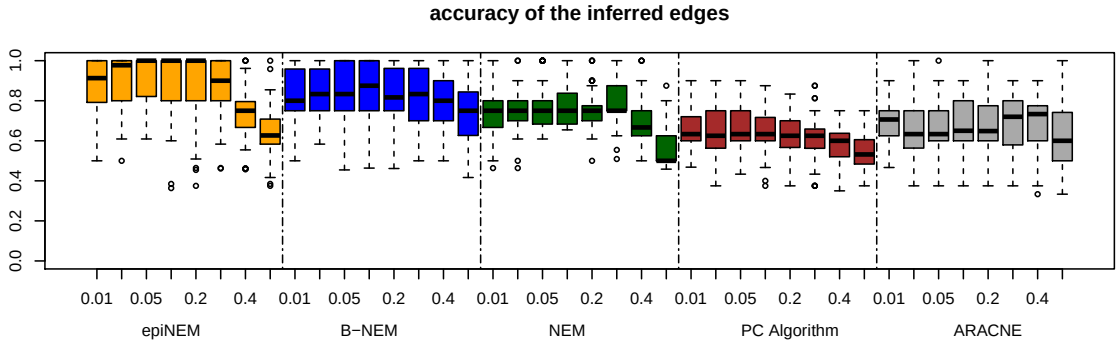
```
##             print(spec2[3, i, j])

##         }

##         if ("p" %in% do) {
##             print("PCalg")

##             start <- Sys.time()
##             pc.fit <- pc(suffStat = list(C = cor(reddata2), n = nrow(reddata2)),
##                   indepTest = gaussCItest, ## indep.test: partial correlations
##                   alpha=0.05, labels = colnames(reddata2), verbose = F)
##             pcadj <- graph2adj(pc.fit@graph)
##             time[4, i, j] <- difftime(Sys.time(), start, units = "secs")
##             print(time[4, i, j])

##             tp <- sum(gtnadj == 1 & pcadj == 1)
##             tn <- sum(gtnadj  == 0 & pcadj == 0)
##             fp <- sum(gtnadj == 0 & pcadj == 1)
##             fn <- sum(gtnadj == 1 & pcadj == 0)
##             sens2[4, i, j] <- tp/(tp+fn)
##             spec2[4, i, j] <- tn/(tn+fp)
##             print(sens2[4, i, j])
##             print(spec2[4, i, j])

##         }

##         if ("a" %in% do) {
##             print("Aracne")

##             start <- Sys.time()
##             ares <- build.mim(reddata2)
##             ares <- aracne(ares)
##             ares <- disc(ares, 0)
##             ares <- ares[order(rownames(ares)), order(colnames(ares))]
##             nas <- which(is.na(ares) == T)
##             ares[nas] <- 0
##             diag(ares) <- 0
##             time[5, i, j] <- difftime(Sys.time(), start, units = "secs")
##             print(time[5, i, j])

##             tp <- sum(gtnadj == 1 & ares == 1)
##             tn <- sum(gtnadj == 0 & ares == 0)
##             fp <- sum(gtnadj == 0 & ares == 1)
##             fn <- sum(gtnadj == 1 & ares == 0)
##             sens2[5, i, j] <- tp/(tp+fn)
##             spec2[5, i, j] <- tn/(tn+fp)
##             print(sens2[5, i, j])
##             print(spec2[5, i, j])

##         }

##     }
```

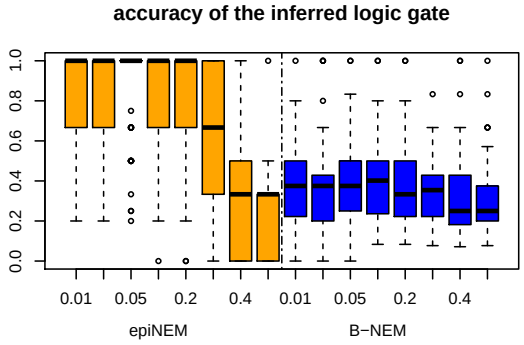
```
## }
```

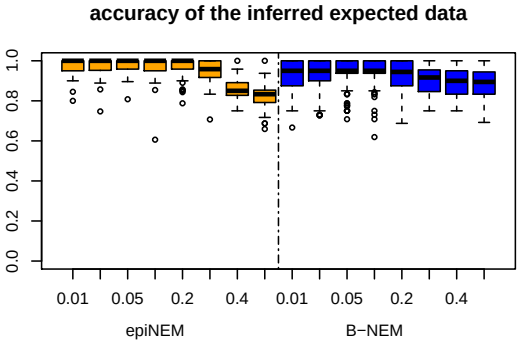
```
data(sim)

colvec <- c(rep("orange", length(noiselvls)), rep("blue", length(noiselvls)),
            rep("darkgreen", length(noiselvls)), rep("brown", length(noiselvls)),
            rep("darkgrey", length(noiselvls)))

acc <- (sens + spec)/2

acc2 <- (sens2 + spec2)/2

m <- rbind(c(1,1), c(2,2), c(3,4))

layout(m)

timeframe <- as.data.frame(
    cbind(data.frame(epiNEM = time[1,,]),
          data.frame(BNEM = time[2,,]), data.frame(NEM = time[3,,]),
          data.frame(Cor = time[4,,]), data.frame(MI = time[5,,])))

colnames(timeframe) <- rep(noiselvls, 5)

boxplot(timeframe, col = colvec, main = "running time", ylab = "seconds")

abline(v=(1:(length(do)-1)*length(noiselvls) + 0.5), col = "black", lty = 6)

axis(1, c(3, 11, 19, 28, 36)+1, c("epiNEM", "B-NEM", "NEM", "PC Algorithm", "ARACNE"),
     tick = F, pos = -25)

accframe2 <- as.data.frame(
    cbind(data.frame(epiNEM = acc2[1,,]),
          data.frame(BNEM = acc2[2,,]), data.frame(NEM = acc2[3,,]),
          data.frame(Cor = acc2[4,,]), data.frame(MI = acc2[5,,])))

colnames(accframe2) <- rep(noiselvls, 5)

boxplot(accframe2, col = colvec, main = "accuracy of the inferred edges", ylim = c(0,1))

abline(v=(1:(length(do)-1)*length(noiselvls) + 0.5), col = "black", lty = 6)

axis(1, c(3, 11, 19, 28, 36)+1, c("epiNEM", "B-NEM", "NEM", "PC Algorithm", "ARACNE"),
     tick = F, pos = -0.2)

## logical nems:

colvec2 <- c(rep("orange", length(noiselvls)), rep("blue", length(noiselvls)))

logicsframe <- as.data.frame(cbind(data.frame(epiNEM = logics[1,,]),
                                   data.frame(BNEM = logics[2,,])))

colnames(logicsframe) <- rep(noiselvls, 2)
```

```
boxplot(logicsframe, col = colvec2, main = "accuracy of the inferred logic gate",
        ylim = c(0,1))

abline(v=length(noiselvls)+0.5, col = "black", lty = 6)

axis(1, c(3, 11, 19, 28, 36)+1, c("epiNEM", "B-NEM", "NEM", "PC Algorithm", "ARACNE"),
     tick = F, pos = -0.2)

accframe <- as.data.frame(cbind(data.frame(epiNEM = acc[1,,]),
                                data.frame(BNEM = acc[2,,])))

colnames(accframe) <- rep(noiselvls, 2)

boxplot(accframe, col = colvec2, main = "accuracy of the inferred expected data",
        ylim = c(0,1))

abline(v=length(noiselvls)+0.5, col = "black", lty = 6)

axis(1, c(3, 11, 19, 28, 36)+1, c("epiNEM", "B-NEM", "NEM", "PC Algorithm", "ARACNE"),
     tick = F, pos = -0.2)
```









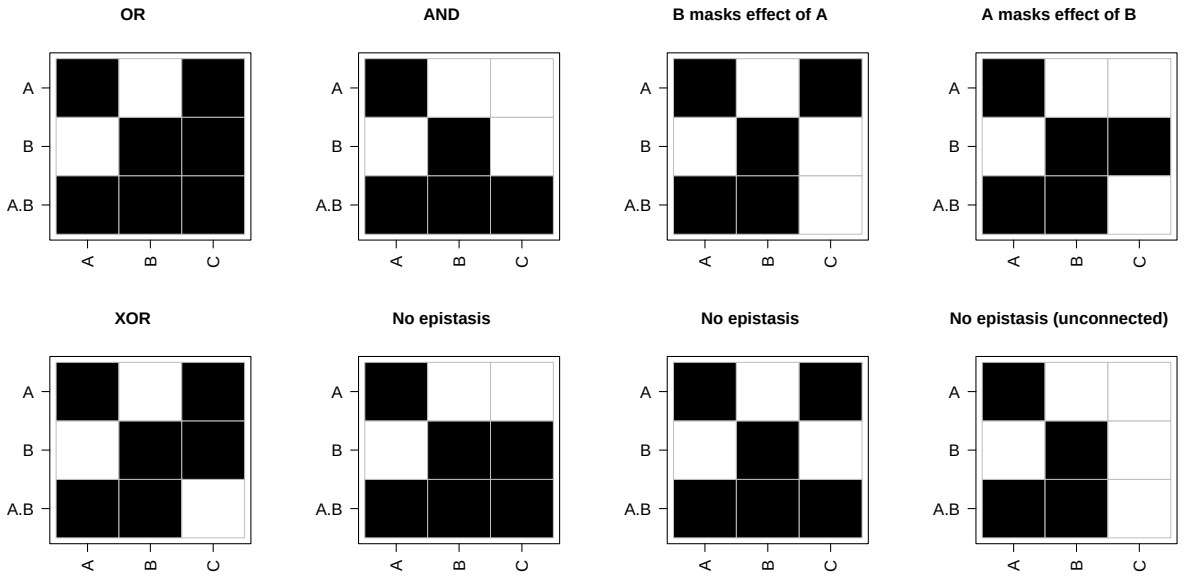
Yeast knockout screens

In this section we analyse previously published yeast knockout screens. The screens consist of gene expression data derived from double and single knockout mutants. We use epiNEM on each double mutant combined with each single mutant.

The results of the knockout screens have been annotated according to the following legend:

```
options(warn=-1)
heatmapOP(matrix(c(1,-1,1,-1,1,1, 1, 1, 1), 3, 3,
                 dimnames = list(c("A", "B", "A.B"), LETTERS[1:3])), Colv = F, Rowv = F,
          main = "OR", col = "Greys", sub = "", colorkey = NULL)
heatmapOP(matrix(c(1,-1,1,-1,1,1, -1, -1, 1), 3, 3,
                 dimnames = list(c("A", "B", "A.B"), LETTERS[1:3])), Colv = F, Rowv = F,
          main = "AND", col = "Greys", sub = "", colorkey = NULL)
```

```
heatmapOP(matrix(c(1,-1,1,-1,1,1, 1, -1, -1), 3, 3,
                 dimnames = list(c("A", "B", "A.B"), LETTERS[1:3])), Colv = F, Rowv = F,
          main = "B masks effect of A", col = "Greys", sub = "", colorkey = NULL)
heatmapOP(matrix(c(1,-1,1,-1,1,1, -1, 1, -1), 3, 3,
                 dimnames = list(c("A", "B", "A.B"), LETTERS[1:3])), Colv = F, Rowv = F,
          main = "A masks effect of B", col = "Greys", sub = "", colorkey = NULL)
heatmapOP(matrix(c(1,-1,1,-1,1,1, 1, 1, -1), 3, 3,
                 dimnames = list(c("A", "B", "A.B"), LETTERS[1:3])), Colv = F, Rowv = F,
          main = "XOR", col = "Greys", sub = "", colorkey = NULL)
heatmapOP(matrix(c(1,-1,1,-1,1,1, -1, 1, 1), 3, 3,
                 dimnames = list(c("A", "B", "A.B"), LETTERS[1:3])), Colv = F, Rowv = F,
          main = "No epistasis", col = "Greys", sub = "", colorkey = NULL)
heatmapOP(matrix(c(1,-1,1,-1,1,1, 1, -1, 1), 3, 3,
                 dimnames = list(c("A", "B", "A.B"), LETTERS[1:3])), Colv = F, Rowv = F,
          main = "No epistasis", col = "Greys", sub = "", colorkey = NULL)
heatmapOP(matrix(c(1,-1,1,-1,1,1, -1, -1, -1), 3, 3,
                 dimnames = list(c("A", "B", "A.B"), LETTERS[1:3])), Colv = F, Rowv = F,
          main = "No epistasis (unconnected)", col = "Greys", sub = "", colorkey = NULL)
options(warn=0)
```



Wageningen et al., 2010

```
file <-
    "http://www.holstegelab.nl/publications/sv/signaling_redundancy/downloads/DataS1.txt"

data <- read.delim(file)

dataM <- data[-(1:2), (1+(1:(324/2))*2)]

dataP <- data[-(1:2), (2+(1:(324/2))*2)]

dataM <- dataM[-1, ]
```

```r
dataP <- dataP[-1, ]

dataM <- apply(dataM, c(1,2), as.numeric)

dataP <- apply(dataP, c(1,2), as.numeric)

dataBin <- dataM

sig <- 0.05

cutoff <- log2(1.7)

dataBin[which(dataP < sig & dataP > 0 & abs(dataM) >= cutoff)] <- 1

dataBin[which(dataP >= sig | dataP == 0 | abs(dataM) < cutoff)] <- 0

dataBin <- dataBin[-which(apply(dataBin, 1, max) == 0), ]

dataBinWag <- dataBin

genelist <- toupper(c('hsl1', 'cla4', 'gin4', 'swe1', 'hsl1.cla4'))

colnames(dataBin) <- gsub(".del.vs..wt", "", colnames(dataBin))

colnames(dataBin) <- gsub(".del", "", colnames(dataBin))

doubles <- colnames(dataBin)[grep("\\.", colnames(dataBin))]

doubles <- sort(doubles[-grep("vs", doubles)])

doubles.genes <- unique(unlist(strsplit(doubles, "\\.")))

singles <- colnames(dataBin)[-grep("\\.", colnames(dataBin))]

singles <- unique(sort(singles))

llmat <- logicmat <- matrix(0, length(singles), length(doubles))

rownames(llmat) <- rownames(logicmat) <- singles

colnames(llmat) <- colnames(logicmat) <- doubles

globalgenes <- which(apply(dataBin, 1, max) == 1)

## for (i in doubles[set]) {
##     if (which(doubles %in% i) == 8) { next() }
##     print(i)
##     doubles.singles <- unlist(strsplit(i, "\\."))
##     egenes <- which(apply(dataBin[, which(colnames(dataBin) %in%
##             c(i, doubles.singles))], 1, max) == 1)
##     for (j in singles) {
##         print(j)
##         if (j %in% doubles.singles) { next() }
```

```
##         dataTmp <- dataBin[, grep(paste(
##              paste("^", c(i, j, doubles.singles), "$", sep = ""), collapse = "|"),
##                                                                 colnames(dataBin))]

##         if (path %in% "fixed_set") {
##             dataTmp <- dataTmp[egenes, ]
##         }
##         if (path %in% "global") {
##             dataTmp <- dataTmp[globalgenes, ]
##         }
##         if (path %in% "") {
##             dataTmp <- dataTmp[which(apply(dataTmp, 1, max) == 1), ]
##         }

##         i1 <- which(singles %in% j)
##         i2 <- which(doubles %in% i)

##         if (!(is.null(dim(dataTmp)))) {

##             if (any(dataTmp[, j] != 0)) {

##                 epires <- epiNEM(dataTmp, method = "exhaustive")

##                 tmp <- epires$logics
##                 if ("OR" %in% tmp) {
##                     if (sum(epires$origModel[, j]) != 2) {
##                         tmp <- "NOEPI"
##                     } else {
##                         if (all(tmp %in% "OR")) {
##                             tmp <- "OR"
##                         } else {
##                             tmp <- tmp[which(!(tmp %in% "OR"))]
##                         }
##                     }
##                 }

##                 logicmat[i1, i2] <- tmp
##                 llmat[i1, i2] <- epires$score

##             } else {

##                 logicmat[i1, i2] <- "UNCON"
##                 llmat[i1, i2] <- -Inf

##             }

##         } else {

##             logicmat[i1, i2] <- "UNCON"
##             llmat[i1, i2] <- -Inf

##         }
```

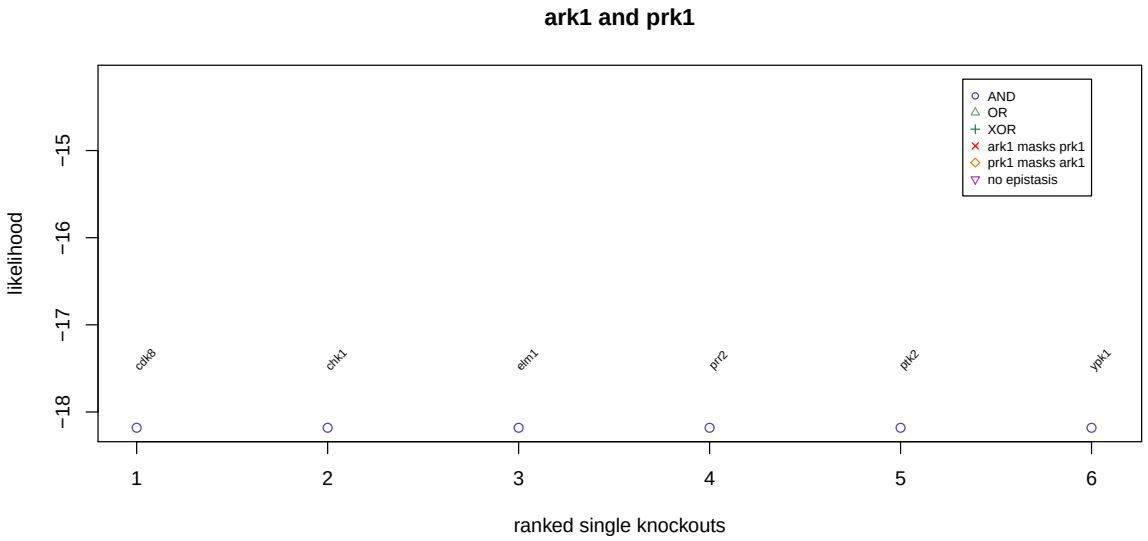
```
##     }

## }
```

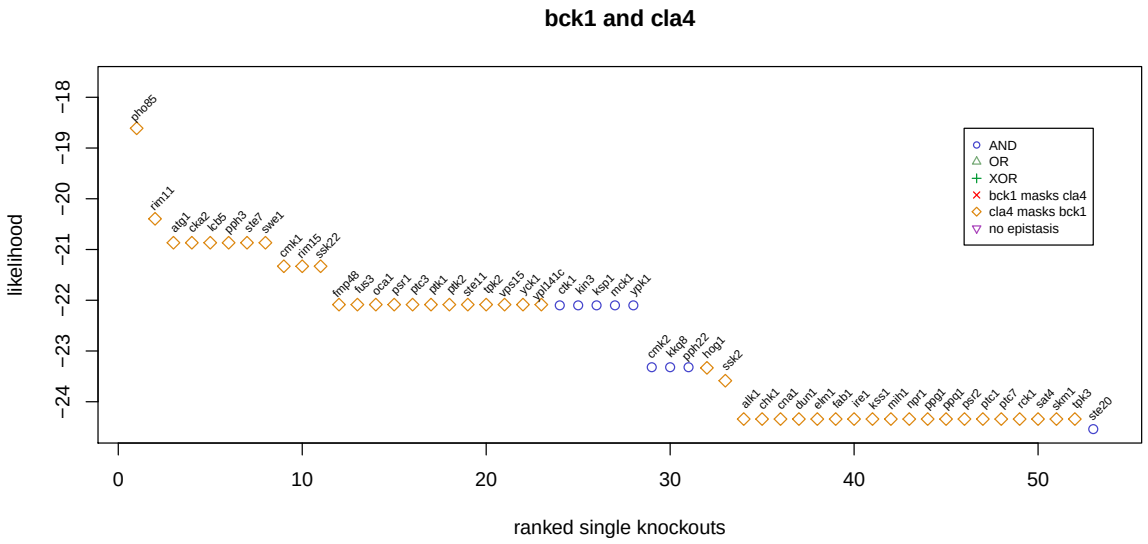
Plot results.

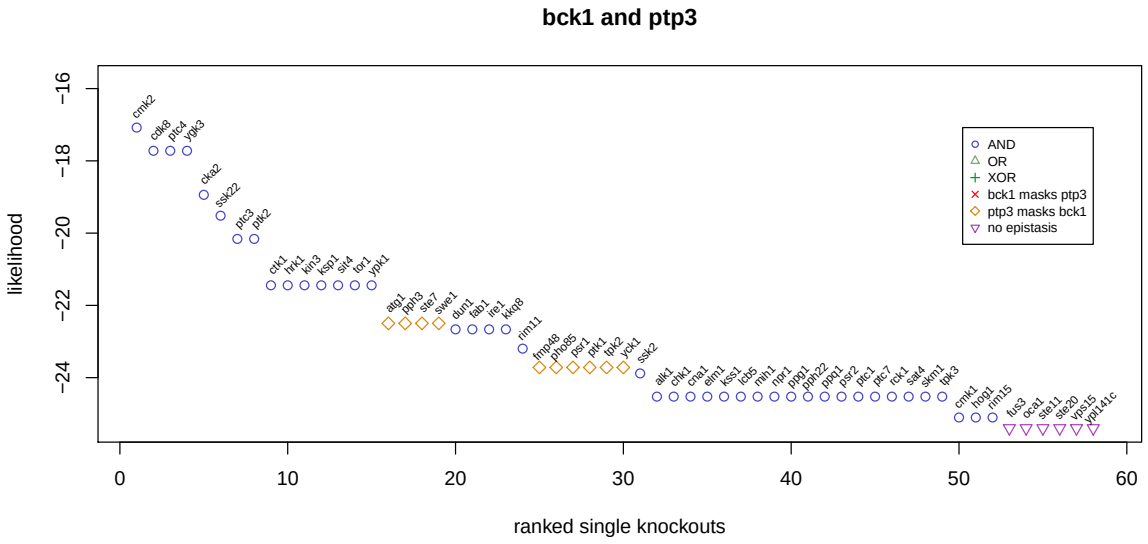
```
palette(c("#4444cc", "#77aa77", "#009933", "#ff0000", "#dd8811", "#aa44bb", "#999900"))

data(wageningen_res)

llmat0 <- wageningen$ll

logicmat0 <- wageningen$logic

for (i in 1:length(doubles)) {

    if (i %in% 8) { next() }

    logicvec <- logicmat0[, i]

    llvec <- llmat0[, i]

    logicvec <- logicvec[order(llvec, decreasing = T)]

    llvec <- llvec[order(llvec, decreasing = T)]

    parents <- unlist(strsplit(doubles[i], "\\."))

    pchvec <- numeric(length(llvec))

    pchvec[which(logicvec %in% "AND")] <- 1
    pchvec[which(logicvec %in% "OR")] <- 2
    pchvec[which(logicvec %in% "XOR")] <- 3
    pchvec[grep(paste("^", parents[1], sep = ""), logicvec)] <- 4
    pchvec[grep(paste("^", parents[2], sep = ""), logicvec)] <- 5
    pchvec[which(logicvec %in% "NOEPI")] <- 6
    pchvec[which(logicvec %in% c("NOINFO", "NOINF"))] <- 7

    logicvec <- logicvec[-which(logicvec %in% "0")]
    pchvec <- pchvec[-which(pchvec == 0)]
    llvec <- llvec[-which(llvec == 0)]

    colvec <- pchvec

    if (all(is.infinite(llvec) == T)) {

        llvec[1:length(llvec)] <- -1000

        margin <- 100

        donames <- 30

    } else {
```

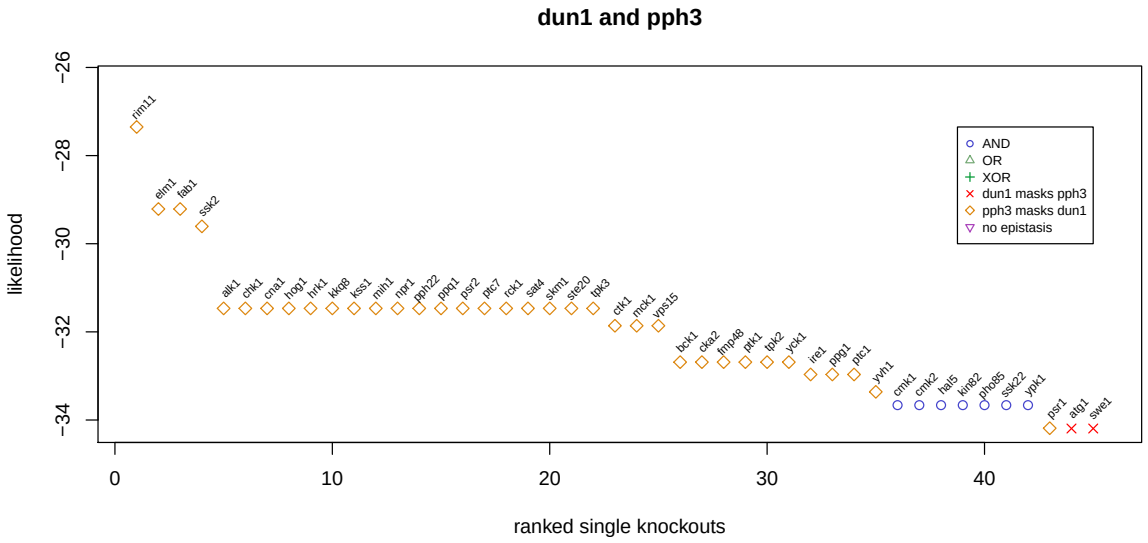
```
        llvec[which(is.infinite(llvec) == T)] <- NA

        ## llvec[which(is.infinite(llvec) == T)] <- min(llvec) - 100

        margin <- abs(max(llvec[1:30], na.rm = T) - min(llvec[1:30], na.rm = T))

        offset <- 0.075

        if (margin == 0) { margin <- 10; offset <- 0.0375 }

        donames <- 30 - sum(is.na(llvec[1:30]) == T)

        if (any(is.na(llvec[1:30]) == T)) { margin2 <- margin*2
        } else { margin2 <- margin }

        llvec[which(is.na(llvec) == T)] <- min(llvec, na.rm = T) - margin

        margin <- margin2

    }

    if (all(llvec[-(1:30)] - min(llvec[-(1:30)]) == 0)) {

        p2max <- max(llvec[-(1:30)]) + margin

    } else {

        p2max <- max(llvec[-(1:30)])

    }

    mark <- ""
    thetop <- sum(!(logicvec %in% c("NOINFO", "NOINF")))
    legendx <- length(llvec[1:thetop])
    p2max <- max(llvec[1:thetop])
    if (p2max == min(llvec[1:thetop])) {
        p2max <- p2max+margin*0.2
    }
    legendtext <- c("AND", "OR", "XOR", paste(parents[1]," masks ", parents[2], sep = ""),
                    paste(parents[2], " masks ", parents[1], sep = ""), "no epistasis")
    if (thetop == 0) { next() }
    plot = plot(llvec[1:thetop], pch = pchvec[1:thetop], col = colvec[1:thetop],
                ylab = "likelihood", xlab = "ranked single knockouts",
                ylim = c(min(llvec[1:thetop]), max(llvec[1:thetop])+margin*0.2),
                xlim = c(1, thetop+(thetop/100)),
                main = paste(unlist(strsplit(doubles[i], "\\.")), collapse = " and "))
    text = text((1:thetop)+(thetop/100), llvec[1:thetop]+(margin*offset),
                labels = names(llvec)[1:thetop], cex = 0.6, srt = 45, pos = 3,
                offset = 0)
    mtext = mtext(mark, side = 3, line = 1, outer = F, cex = 4, adj = 0)
    legend = legend(legendx, p2max,
                    legend = legendtext,
                    col = 1:6, pch = 1:6, xjust = 1, yjust = 1, cex = 0.7)
```

```
}
```

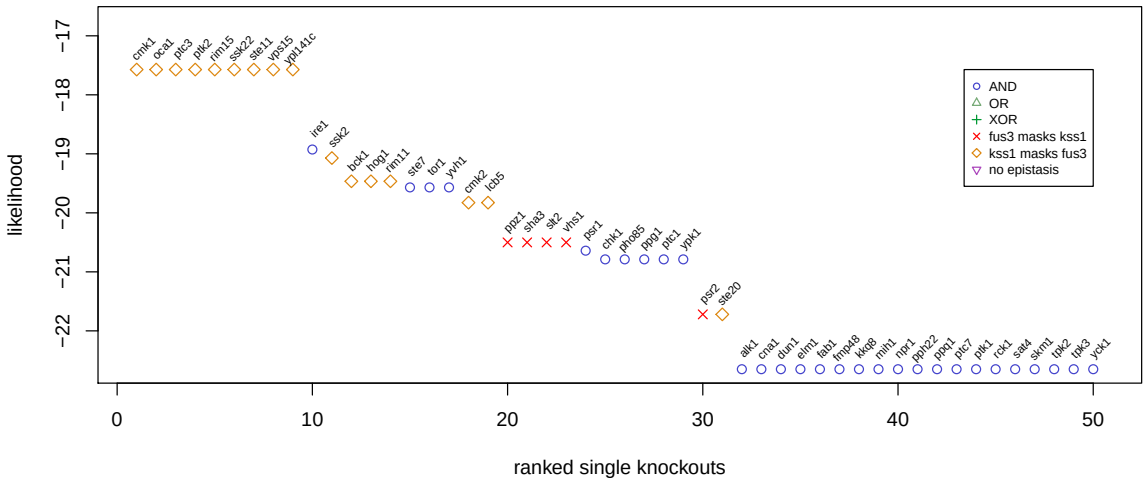

ark1 and prk1
likelihood
−15
−16
−17
−18
1
2
3
4
5
6
ranked single knockouts
AND
OR
XOR
ark1 masks prk1
prk1 masks ark1
no epistasis
cdk8
chk1
elm1
prr2
ptk2
ypk1


bck1 and cla4
likelihood
−18
−19
−20
−21
−22
−23
−24
0
10
20
30
40
50
ranked single knockouts
AND
OR
XOR
bck1 masks cla4
cla4 masks bck1
no epistasis
pho85
rim11
atg1
cka2
kcb5
pph3
ste7
swe1
cmk1
rim15
ssk22
fmp48
tus3
oca1
psr1
ptc3
ptk1
ptk2
ste11
tpk2
vps15
yck1
gpi141c
ctk1
kin3
ksp1
mck1
ypk1
cmk2
kkq8
pph22
hog1
ssk2
atk1
chk1
cna1
dun1
elm1
fab1
ire1
kss1
mih1
npr1
ppg1
ppn1
psr2
ptc1
ptc7
rck1
sat4
skm1
tpk3
ste20

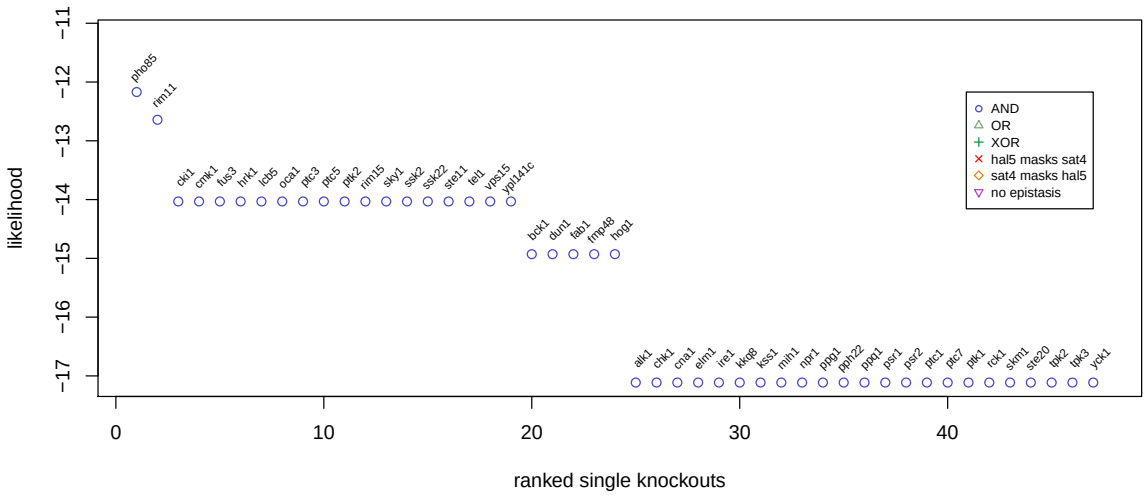

bck1 and ptp3
likelihood
ranked single knockouts
AND
OR
XOR
bck1 masks ptp3
ptp3 masks bck1
no epistasis

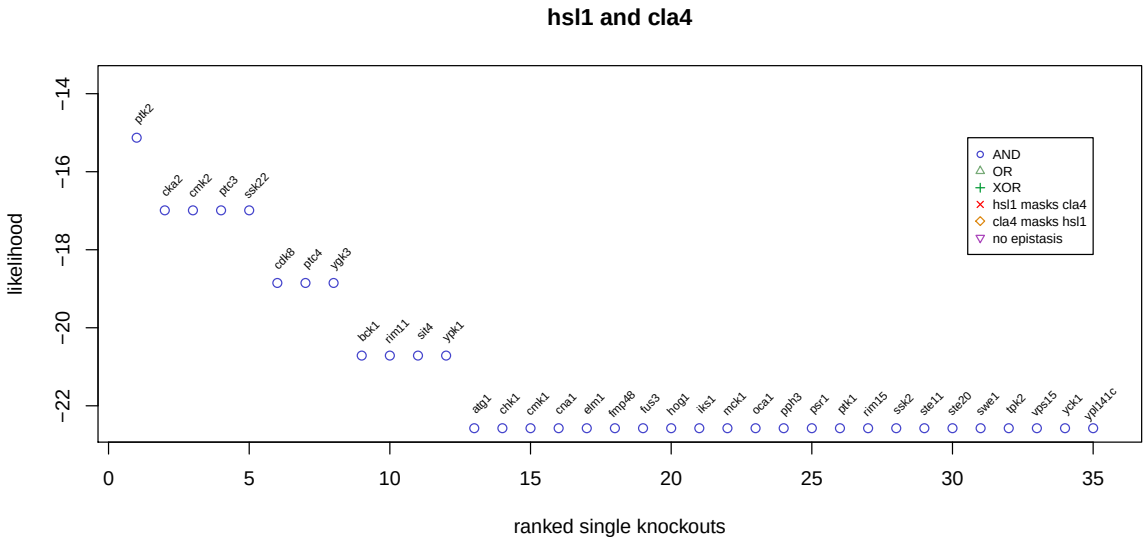

dun1 and pph3
likelihood
ranked single knockouts
AND
OR
XOR
dun1 masks pph3
pph3 masks dun1
no epistasis

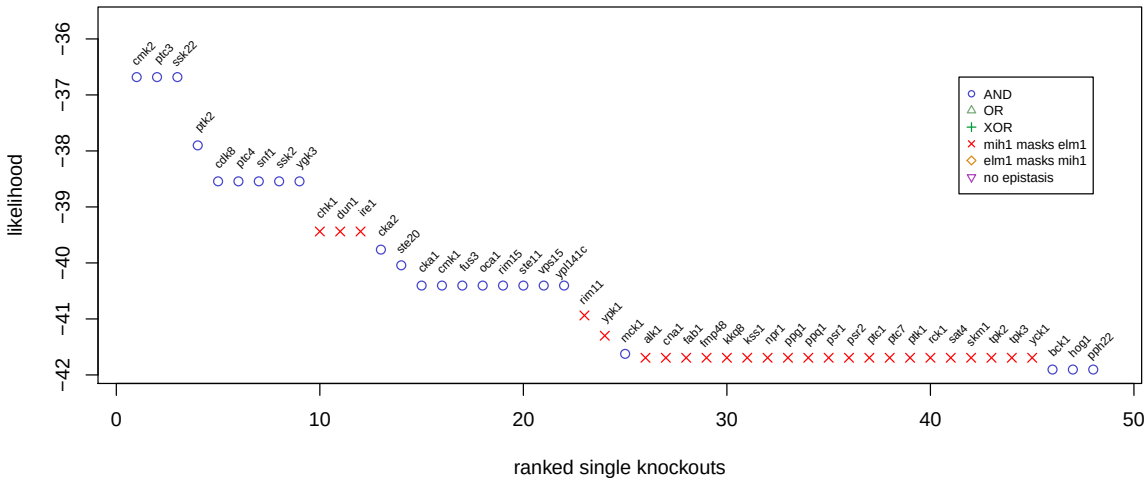
fus3 and kss1

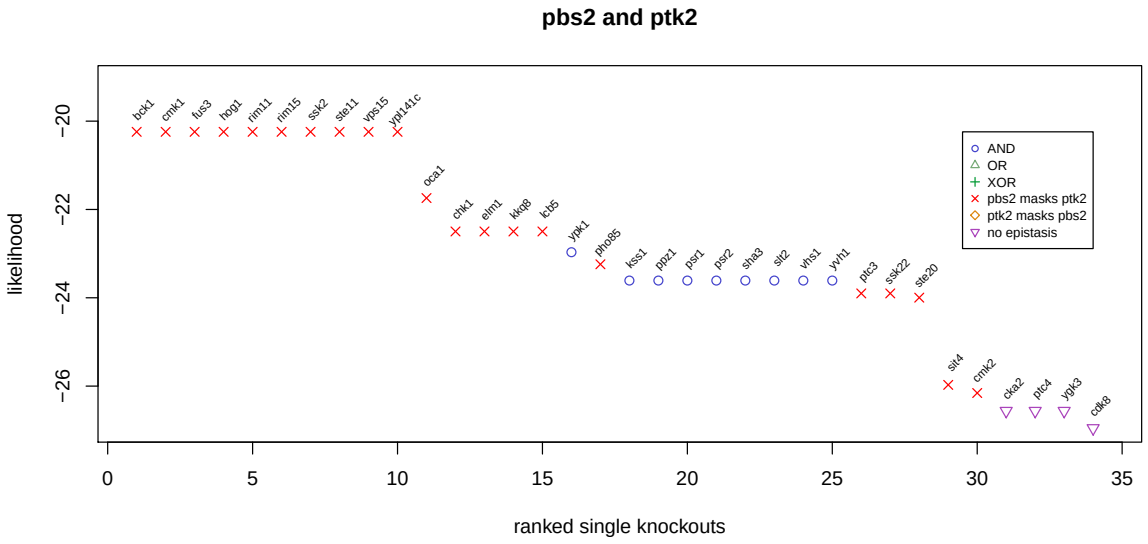


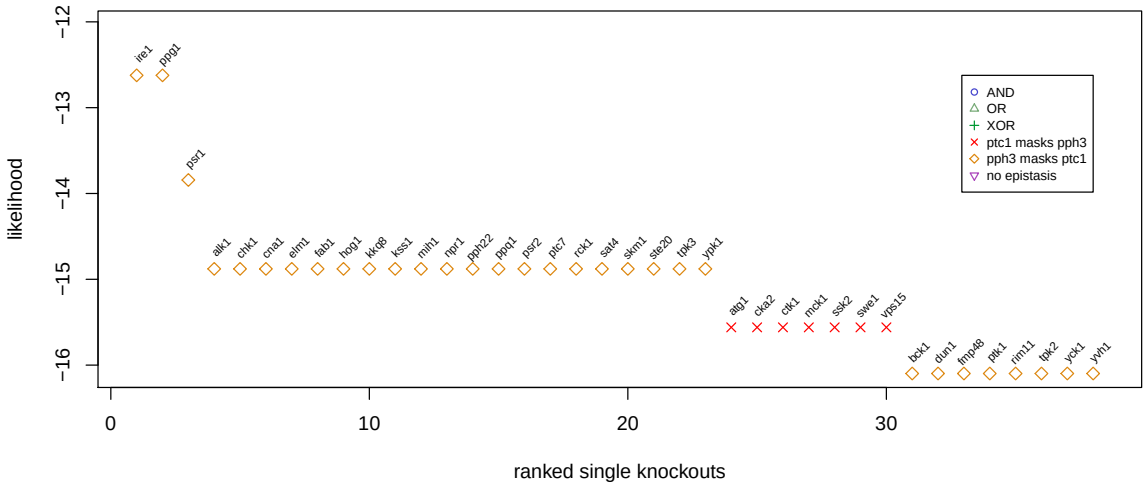
hal5 and sat4



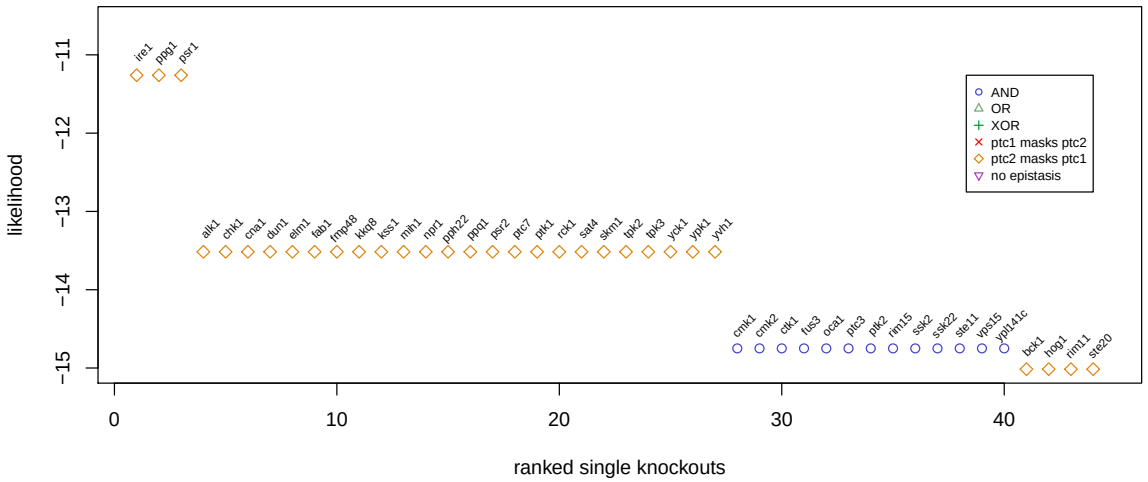

hsl1 and cla4
likelihood
ranked single knockouts
AND
OR
XOR
hsl1 masks cla4
cla4 masks hsl1
no epistasis
ptk2
cka2
cmk2
ptc3
ssk22
cdk8
ptc4
ygk3
bck1
rim11
sit4
ypk1
atg1
chk1
cmk1
cna1
elm1
fmp48
fus3
hog1
iks1
mck1
oca1
pph3
psr1
ptk1
rim15
ssk2
ste11
ste20
swe1
tpk2
vps15
yck1
ydr141c


mih1 and elm1
likelihood
ranked single knockouts
AND
OR
XOR
mih1 masks elm1
elm1 masks mih1
no epistasis
cmk2
ptc3
ssk22
ptk2
cdk8
ptc4
snf1
ssk2
ygk3
chk1
dun1
ire1
cka2
ste20
cka1
cmk1
fus3
oca1
rim15
ste11
vps15
ydl141c
rim11
ypk1
mck1
atk1
cna1
fab1
fmp48
kkq8
kss1
npr1
ppg1
ppq1
psr1
psr2
ptc1
ptc7
ptk1
rck1
sat4
skm1
tpk2
tpk3
yck1
bck1
hog1
pph22

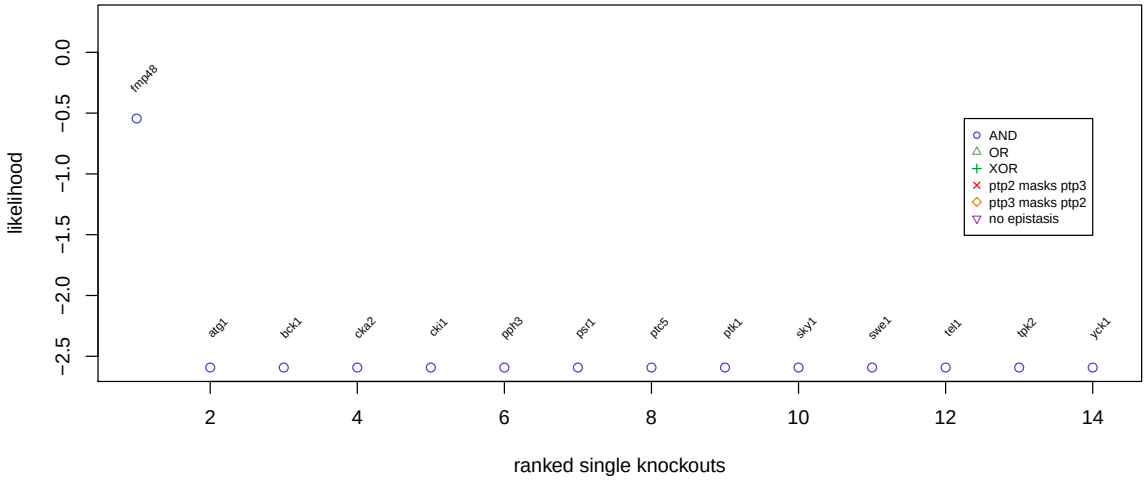

pbs2 and ptk2
likelihood
−20
−22
−24
−26
0
5
10
15
20
25
30
35
ranked single knockouts
bck1
cmk1
lus3
hog1
rim11
rim15
ssk2
ste11
vps15
ypl141c
oca1
chk1
elm1
kkq8
lcb5
ypk1
pho85
kss1
pbz1
psr1
psr2
sha3
slt2
vhs1
yvh1
ptc3
ssk22
ste20
sit4
cmk2
cka2
ptc4
ygk3
ctk3
AND
OR
XOR
pbs2 masks ptk2
ptk2 masks pbs2
no epistasis


ptc1 and pph3
likelihood
−12
−13
−14
−15
−16
0
10
20
30
ranked single knockouts
ire1
ppg1
psr1
alk1
chk1
cna1
elm1
fab1
hog1
kkq8
kss1
mih1
npr1
pph22
ppq1
psr2
ptc7
rck1
sat4
skm1
ste20
tpk3
ypk1
atg1
cka2
ctk1
mck1
ssk2
swe1
vps15
bck1
dun1
fmp48
ptk1
rim11
tpk2
yck1
yvh1
AND
OR
XOR
ptc1 masks pph3
pph3 masks ptc1
no epistasis

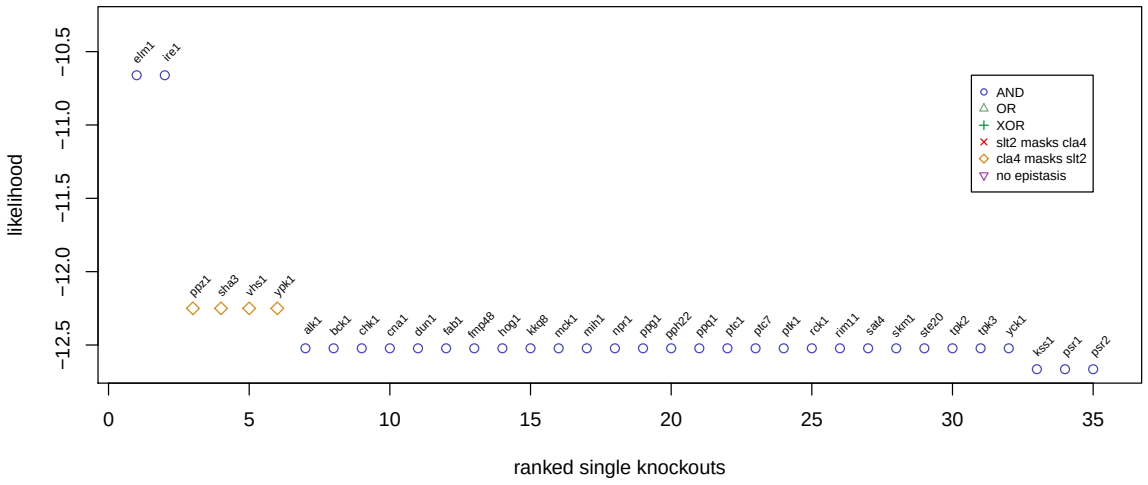
ptc1 and ptc2



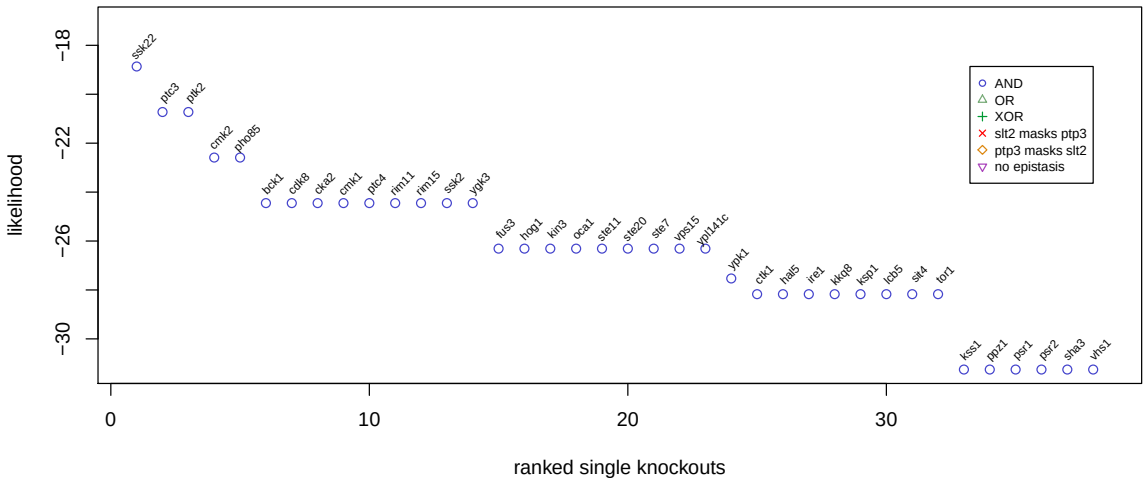
ptp2 and ptp3

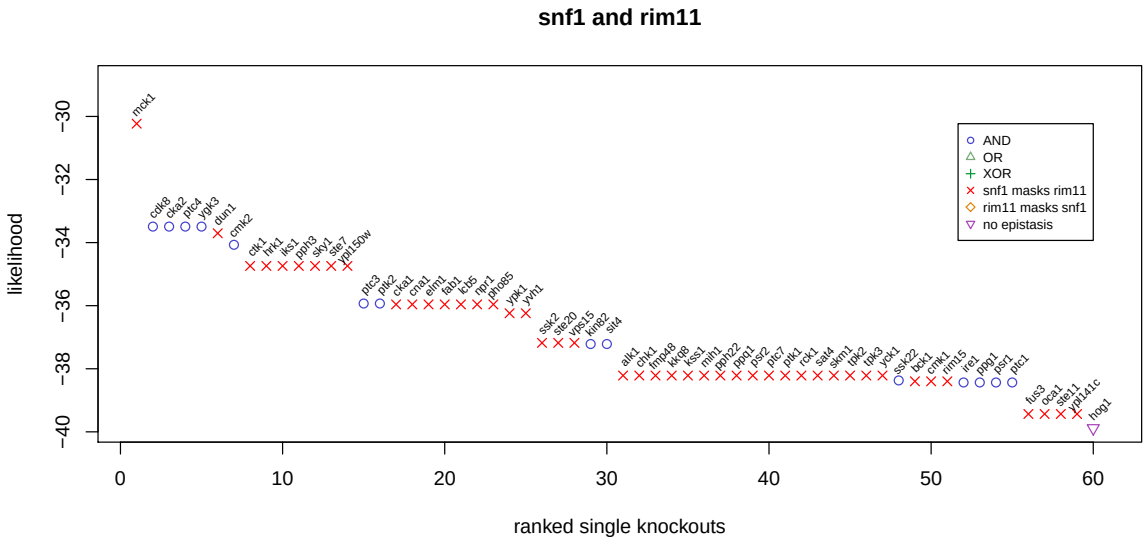


slt2 and cla4



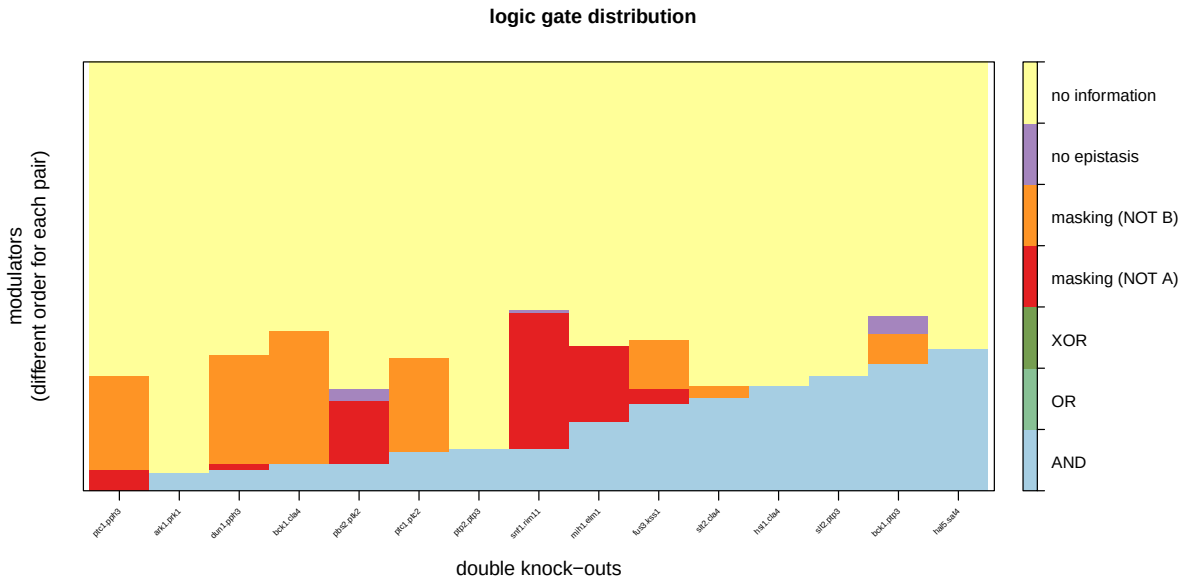
slt2 and ptp3






```
distmat <- wageningen$logic

distmat[which(distmat %in% "AND")] <- 1
distmat[which(distmat %in% "OR")] <- 2
distmat[which(distmat %in% "XOR")] <- 3
distmat[which(distmat %in% "NOEPI")] <- 6
distmat[which(distmat %in% c("NOINFO", "NOINF"))] <- 7

for (i in 1:ncol(distmat)) {

    genes <- unlist(strsplit(colnames(distmat)[i], "\\."))

    distmat[which(distmat[, i] %in%
                  paste(genes[1], " masks the effect of ", genes[2], sep = "")), i] <- 4


    distmat[which(distmat[, i] %in%
                  paste(genes[2], " masks the effect of ", genes[1], sep = "")), i] <- 5

}

distmat <- apply(distmat, c(1,2), as.numeric)

for (i in 1:ncol(distmat)) {
    distmat[, i] <- rev(sort(distmat[, i]))
}

distmat <- distmat[-which(apply(distmat, 1, sum) == 0), ]

distmat <- distmat[, -which(apply(distmat, 2, max) == 0 | apply(distmat, 2, min) == 7)]

y <- distmat

distmat <- distmat[, order(apply(distmat, 2, function(x) { return(sum(x == 1)) }))]
```

```
y[which(y == 5)] <- 4

rownames(distmat) <- NULL

labeltext <- c("", "no information\n\n\n", "no epistasis\n\n\n","masking (NOT B)\n\n\n",
               "masking (NOT A)\n\n\n", "XOR\n\n\n", "OR\n\n\n", "AND\n\n\n")

heatmapOP(distmat, Colv = F, Rowv = F, main = "logic gate distribution",
          sub = "", col = "Paired", breaks = seq(0.5,7.5, length.out = 8),
          cexRow = 0, cexCol = 0.4, aspect = "fill",
          colorkey = list(space = "right",
                          labels = rev(labeltext), width = 1,
                          at = seq(1.5,7.5, length.out = 8)),
          xlab = "double knock-outs",
          ylab = "modulators\n(different order for each pair)",
          xrot = 45, bordercol = "transparent")
```



Sameith et al., 2015

```
file <- paste("http://www.holstegelab.nl/publications/GSTF_geneticinteractions/",
              "downloads/del_mutants_limma.txt", sep = "")

data <- read.delim(file)

data <- apply(data, c(1,2), as.character)

dataM <- data[-1, which(data[1, ] %in% "M")]

dataM <- apply(dataM, c(1,2), as.numeric)

dataP <- data[-1, which(data[1, ] %in% "p.value")]

dataP <- apply(dataP, c(1,2), as.numeric)
```

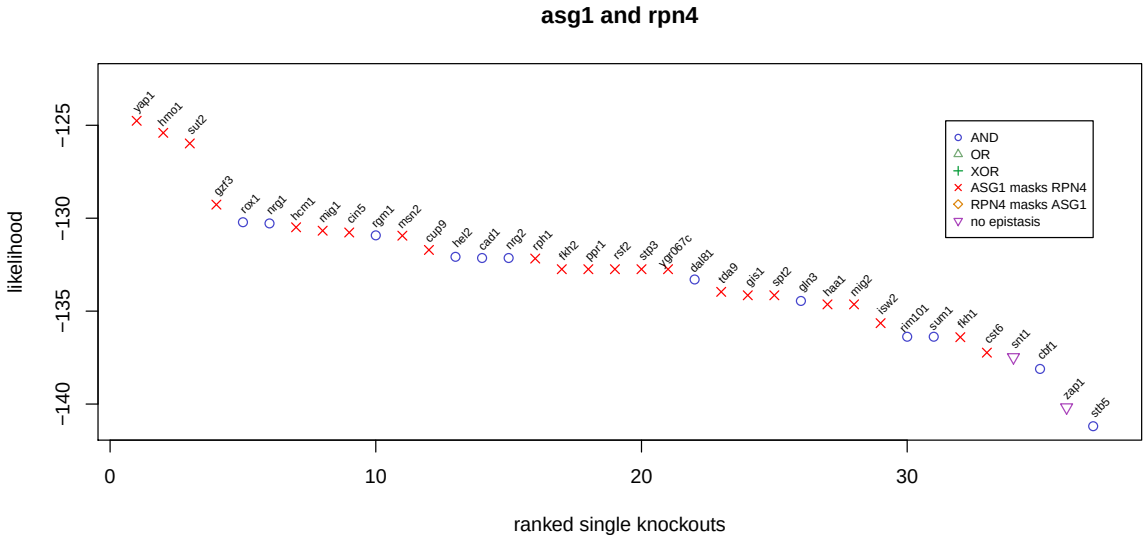
```
dataBin <- dataM

sig <- 0.01

cutoff <- log2(1.5)

dataBin[which(dataP < sig & dataP > 0 & abs(dataM) >= cutoff)] <- 1

dataBin[which(dataP >= sig | dataP == 0 | abs(dataM) < cutoff)] <- 0

dataBin <- dataBin[-which(apply(dataBin, 1, max) == 0), ]

colnames(dataBin) <- gsub("\\.\\.\\.", "\\.", colnames(dataBin))

## big screen:

doubles <- colnames(dataBin)[grep("\\.", colnames(dataBin))]

doubles.genes <- unique(unlist(strsplit(doubles, "\\.")))

singles <- colnames(dataBin)[-grep("\\.", colnames(dataBin))]

singles <- unique(sort(singles))

llmat <- logicmat <- matrix(0, length(singles), length(doubles))

rownames(llmat) <- rownames(logicmat) <- singles

colnames(llmat) <- colnames(logicmat) <- doubles

globalgenes <- which(apply(dataBin, 1, max) == 1)

## for (i in doubles[set]) {
##     print(i)
##     doubles.singles <- unlist(strsplit(i, "\\."))
##     egenes <- which(apply(dataBin[,
##                   which(colnames(dataBin) %in% c(i, doubles.singles))], 1, max) == 1)
##     for (j in singles) {
##         print(j)
##         if (j %in% doubles.singles) { next() }

##         dataTmp <- dataBin[, grep(paste(paste("^", c(i, j, doubles.singles), "$", sep
##                       = ""), collapse = "|"), colnames(dataBin))]

##         if (path %in% "fixed_set") {
##             dataTmp <- dataTmp[egenes, ]
##         }
##         if (path %in% "global") {
##             dataTmp <- dataTmp[globalgenes, ]
##         }
##         if (path %in% "") {
##             dataTmp <- dataTmp[which(apply(dataTmp, 1, max) == 1), ]
##         }
```

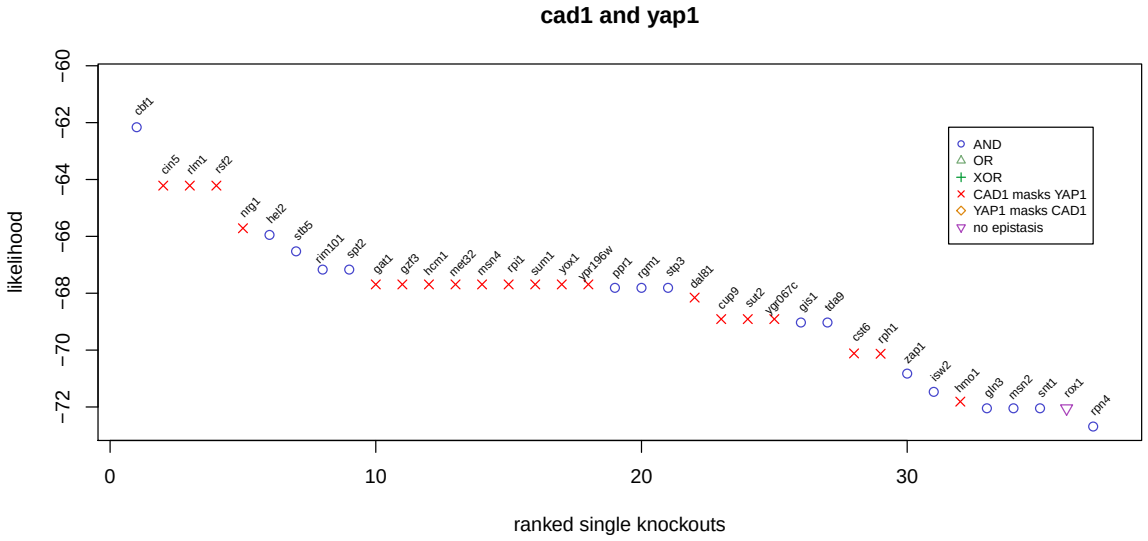
```
##          i1 <- which(singles %in% j)
##          i2 <- which(doubles %in% i)

##          if (!(is.null(dim(dataTmp)))) {

##              if (any(dataTmp[, j] != 0)) {

##                  epires <- epiNEM(dataTmp, method = "exhaustive")

##                  tmp <- epires$logics
##                  if ("OR" %in% tmp) {
##                      if (sum(epires$origModel[, j]) != 2) {
##                          tmp <- "NOEPI"
##                      } else {
##                          if (all(tmp %in% "OR")) {
##                              tmp <- "OR"
##                          } else {
##                              tmp <- tmp[which(!(tmp %in% "OR"))]
##                          }
##                      }
##                  }

##                  logicmat[i1, i2] <- tmp
##                  llmat[i1, i2] <- epires$score

##              } else {

##                  logicmat[i1, i2] <- "UNCON"
##                  llmat[i1, i2] <- -Inf

##              }

##          } else {

##              logicmat[i1, i2] <- "UNCON"
##              llmat[i1, i2] <- -Inf

##          }

##      }

## }
```

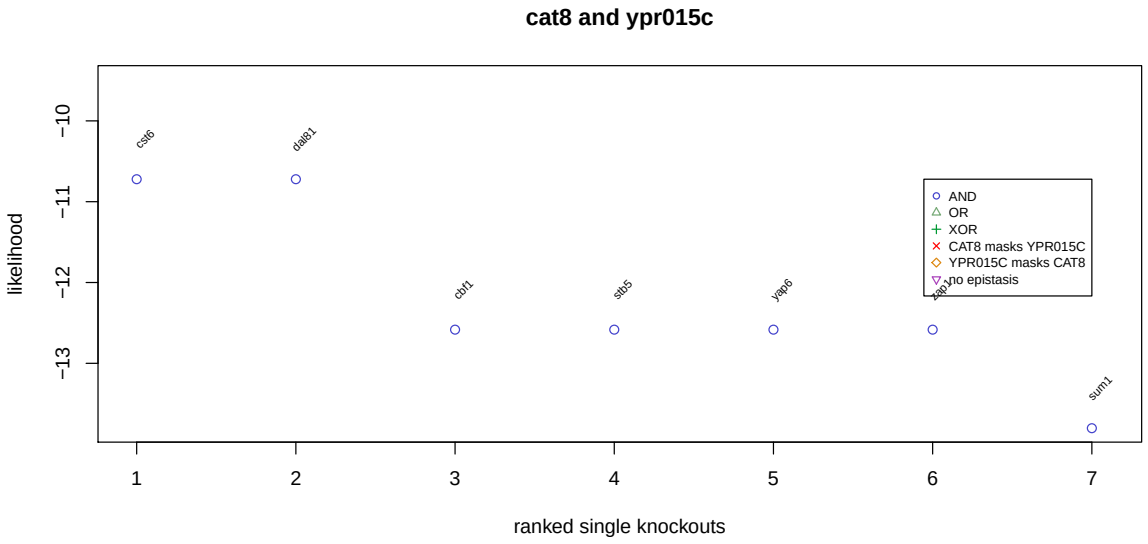
```
data(sameith_res)

llmat0 <- sameith$ll

logicmat0 <- sameith$logic

for (i in 1:length(doubles)) {

  logicvec <- logicmat0[, i]
```

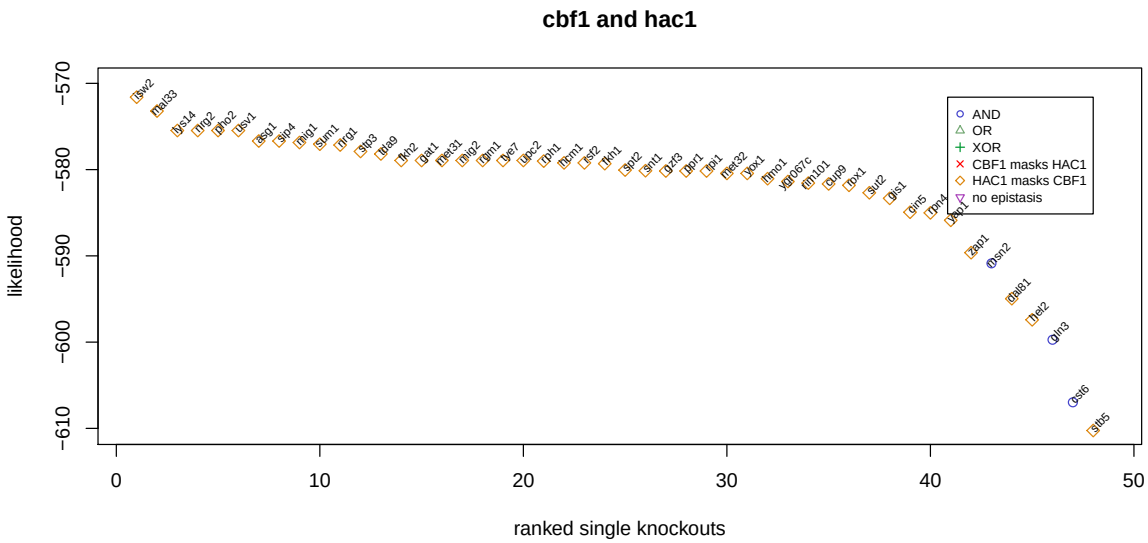
```
llvec <- llmat0[, i]

logicvec <- logicvec[order(llvec, decreasing = T)]

llvec <- llvec[order(llvec, decreasing = T)]

parents <- unlist(strsplit(doubles[i], "\\."))

pchvec <- numeric(length(llvec))

  pchvec[which(logicvec %in% "AND")] <- 1
  pchvec[which(logicvec %in% "OR")] <- 2
  pchvec[which(logicvec %in% "XOR")] <- 3
  pchvec[grep(paste("^", parents[1], sep = ""), logicvec)] <- 4
  pchvec[grep(paste("^", parents[2], sep = ""), logicvec)] <- 5
  pchvec[which(logicvec %in% "NOEPI")] <- 6
  pchvec[which(logicvec %in% c("NOINFO", "NOINF"))] <- 7

  logicvec <- logicvec[-which(logicvec %in% "0")]
  pchvec <- pchvec[-which(pchvec == 0)]
  llvec <- llvec[-which(llvec == 0)]

  colvec <- pchvec

  if (all(is.infinite(llvec) == T)) {

      llvec[1:length(llvec)] <- -1000

      margin <- 100

      donames <- 30

  } else {

      llvec[which(is.infinite(llvec) == T)] <- NA

      margin <- abs(max(llvec[1:30], na.rm = T) - min(llvec[1:30], na.rm = T))

      if (margin == 0) { margin <- 10 }

      donames <- 30 - sum(is.na(llvec[1:30]) == T)

      if (any(is.na(llvec[1:30]) == T)) { margin2 <- margin*2
      } else { margin2 <- margin }

      llvec[which(is.na(llvec) == T)] <- min(llvec, na.rm = T) - margin

      margin <- margin2

  }

  if (all(llvec[-(1:30)] - min(llvec[-(1:30)]) == 0)) {
```

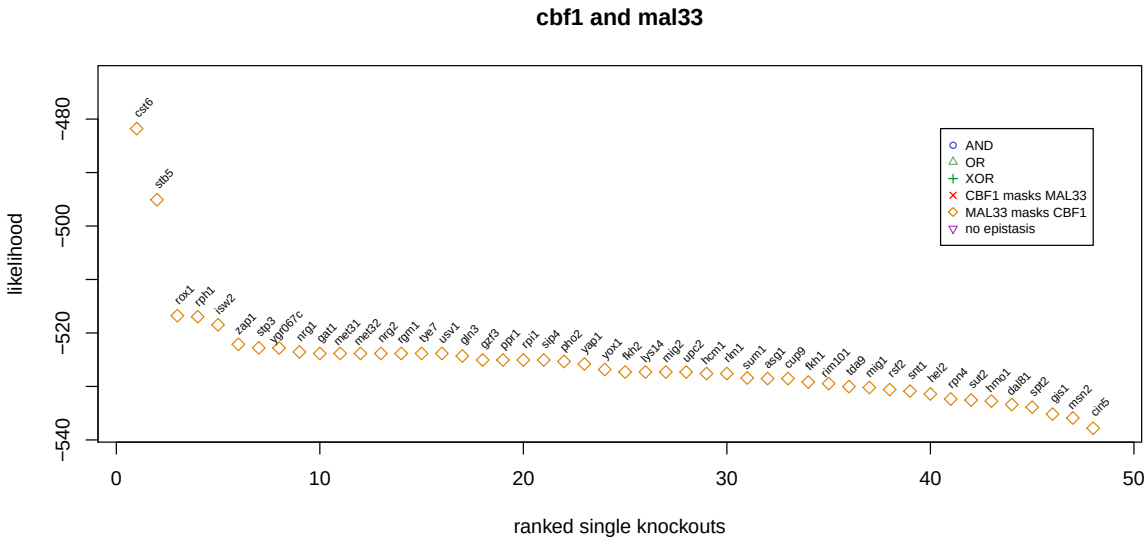
```
        p2max <- max(llvec[-(1:30)]) + margin

    } else {

        p2max <- max(llvec[-(1:30)])

    }

    labeltext <- c("AND", "OR", "XOR", paste(parents[1], " masks ", parents[2], sep = ""),
                   paste(parents[2], " masks ", parents[1], sep = ""), "no epistasis")

    mark <- ""
    pointx <- 10000
    thetop <- sum(!(logicvec %in% c("NOINFO", "NOINF")))
    legendx <- length(llvec[1:thetop])
    p2max <- max(llvec[1:thetop])
    if (p2max == min(llvec[1:thetop])) {
        p2max <- p2max+margin*0.2
    }
    if (thetop == 0) { next() }
    plot = plot(llvec[1:thetop], pch = pchvec[1:thetop], col = colvec[1:thetop],
                ylab = "likelihood", xlab = "ranked single knockouts",
                ylim = c(min(llvec[1:thetop]), max(llvec[1:thetop])+margin*0.2),
                xlim = c(1, thetop+(thetop/100)),
                main = paste(tolower(unlist(strsplit(doubles[i], "\\."))),
                             collapse = " and "))
    text = text((1:thetop)+(thetop/100), llvec[1:thetop]+(margin*offset),
                labels = tolower(names(llvec)[1:thetop]), cex = 0.6, srt = 45, pos = 3,
                offset = 0)
    mtext = mtext(mark, side = 3, line = 1, outer = F, cex = 4, adj = 0)
    legend = legend(legendx, p2max,
                    legend = labeltext, col = 1:6, pch = 1:6, xjust = 1, yjust = 1,
                    cex = 0.7)

}
```

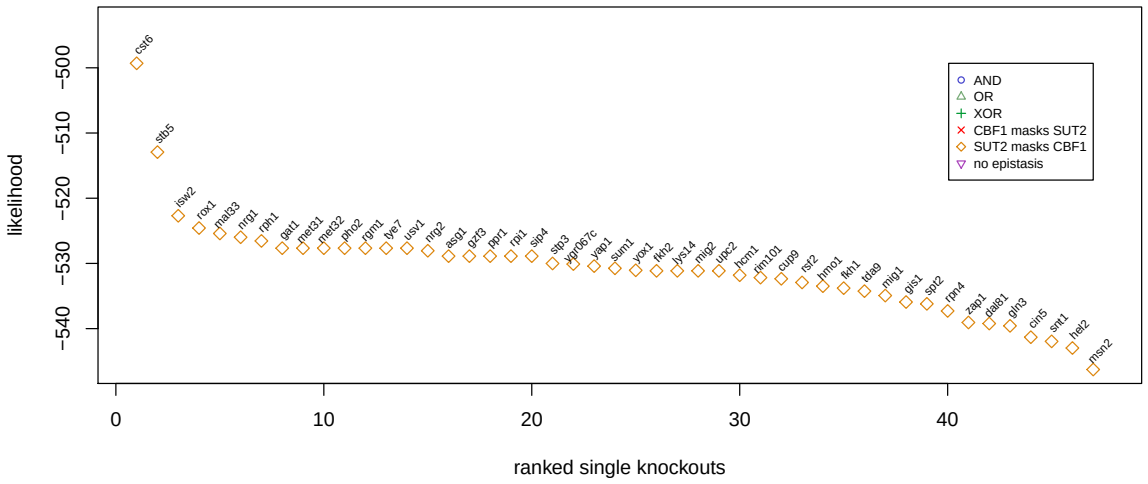

asg1 and rpn4
likelihood
−125
−130
−135
−140
0
10
20
30
ranked single knockouts
AND
OR
XOR
ASG1 masks RPN4
RPN4 masks ASG1
no epistasis
yap1
hmo1
sut2
gzf3
rox1
nrg1
hcm1
mig1
cin5
rpn1
msn2
cup9
hel2
cad1
mig2
rph1
tkh2
ppr1
rsf2
stp3
ygr067c
dal81
tda9
gis1
spt2
gln3
haa1
mig2
isw2
rim101
sum1
fkh1
cst6
snf1
cbf1
zap1
stb5

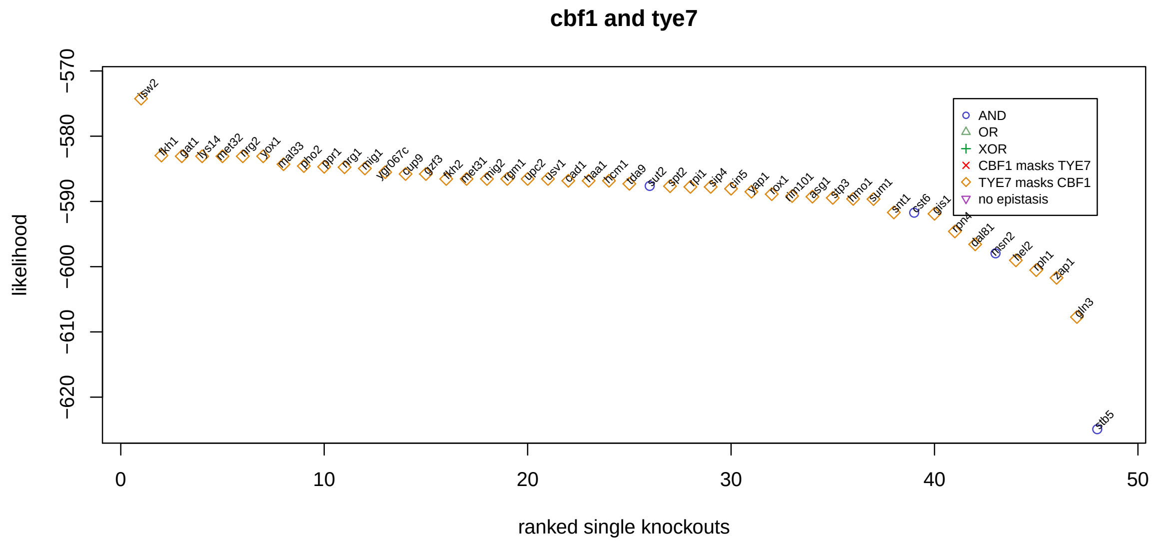

cad1 and yap1
likelihood
−60
−62
−64
−66
−68
−70
−72
0
10
20
30
ranked single knockouts
AND
OR
XOR
CAD1 masks YAP1
YAP1 masks CAD1
no epistasis
cbf1
cin5
rlm1
rsf2
nrg1
hel2
stb5
rim101
spt2
gat1
gzf3
hcm1
mes2
msn4
rpi1
sum1
vox1
ypr196w
ppr1
rgm1
stp3
dal81
cup9
sut2
ygr067c
gis1
tda9
cst6
rph1
zap1
isw2
hmo1
gln3
msn2
snf1
rox1
rpn4

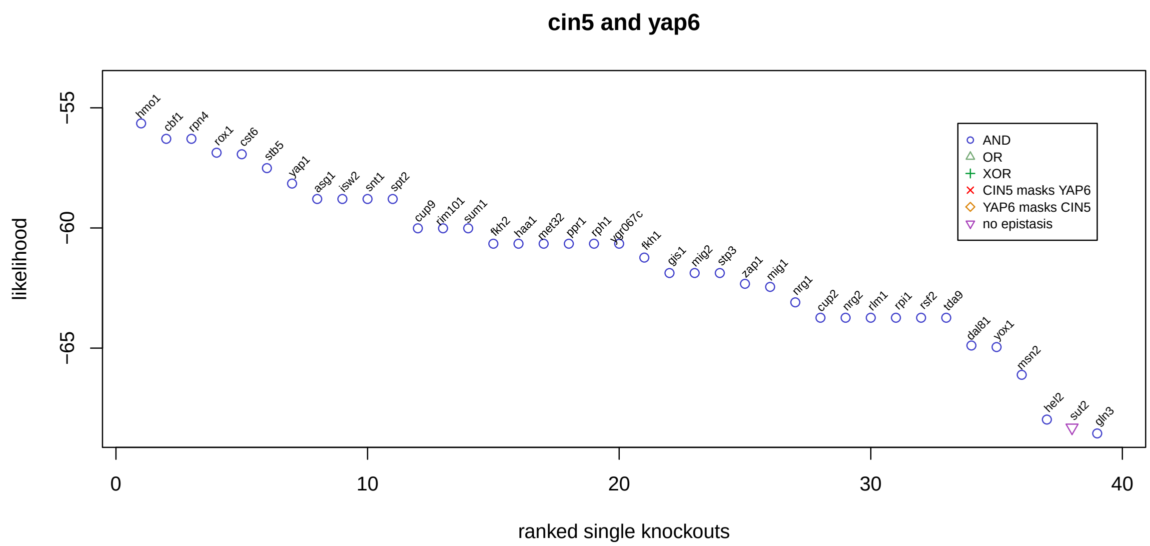

cat8 and ypr015c
likelihood
−10
−11
−12
−13
cat6
dal81
cbf1
stb5
yap6
zap1
sum1
AND
OR
XOR
CAT8 masks YPR015C
YPR015C masks CAT8
no epistasis
1
2
3
4
5
6
7
ranked single knockouts

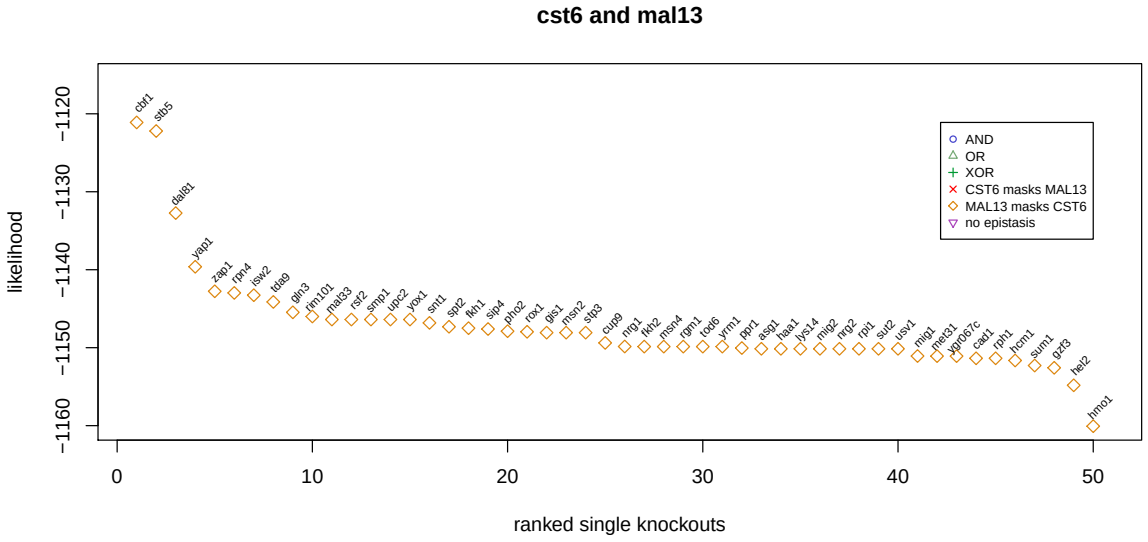

cbf1 and hac1
likelihood
−570
−580
−590
−600
−610
AND
OR
XOR
CBF1 masks HAC1
HAC1 masks CBF1
no epistasis
0
10
20
30
40
50
ranked single knockouts

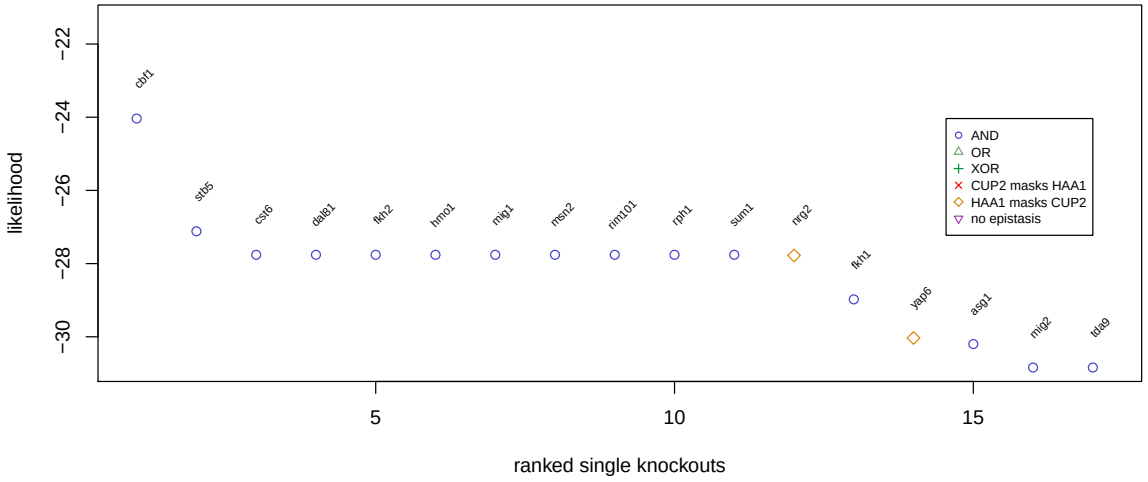

cbf1 and mal33
likelihood
−480
−500
−520
−540
0
10
20
30
40
50
ranked single knockouts
AND
OR
XOR
CBF1 masks MAL33
MAL33 masks CBF1
no epistasis
csr6
stb5
rox1
rph1
isw2
zap1
stp3
ygr067c
nrg1
gat1
met31
met32
nrg2
rgm1
tye7
usv1
gln3
gzf3
ppr1
rpi1
stp4
pho2
yap1
yox1
fkh2
lys14
mig2
upc2
hcm1
rlm1
sum1
asg1
cup9
fkh1
rim101
tda9
mig1
rsf2
snt1
hel2
rpn4
sut2
hmo1
dal81
spt2
gis1
msn2
cin5


cbf1 and sut2
likelihood
−500
−510
−520
−530
−540
0
10
20
30
40
ranked single knockouts
AND
OR
XOR
CBF1 masks SUT2
SUT2 masks CBF1
no epistasis
csr6
stb5
isw2
rox1
mal33
nrg1
rph1
gat1
met31
met32
pho2
rgm1
tye7
usv1
nrg2
asg1
gzf3
ppr1
rpi1
stp4
stp3
ygr067c
yap1
sum1
yox1
fkh2
lys14
mig2
upc2
hcm1
rim101
cup9
rsf2
hmo1
fkh1
tda9
mig1
gis1
spt2
rpn4
zap1
dal81
gln3
cin5
snt1
hel2
msn2

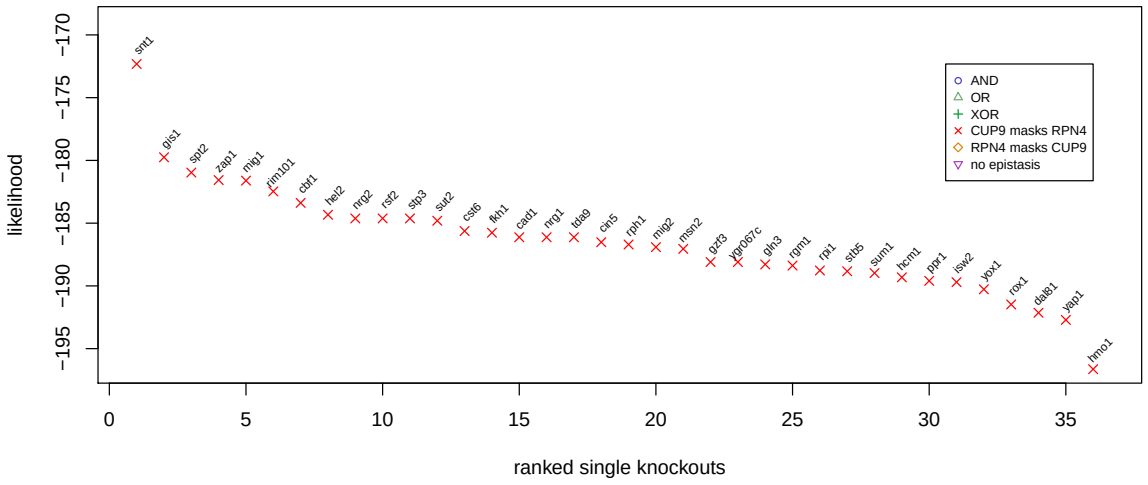

cbf1 and tye7
likelihood
−570
−580
−590
−600
−610
−620
0
10
20
30
40
50
ranked single knockouts
AND
OR
XOR
CBF1 masks TYE7
TYE7 masks CBF1
no epistasis


cin5 and yap6
likelihood
−55
−60
−65
0
10
20
30
40
ranked single knockouts
AND
OR
XOR
CIN5 masks YAP6
YAP6 masks CIN5
no epistasis

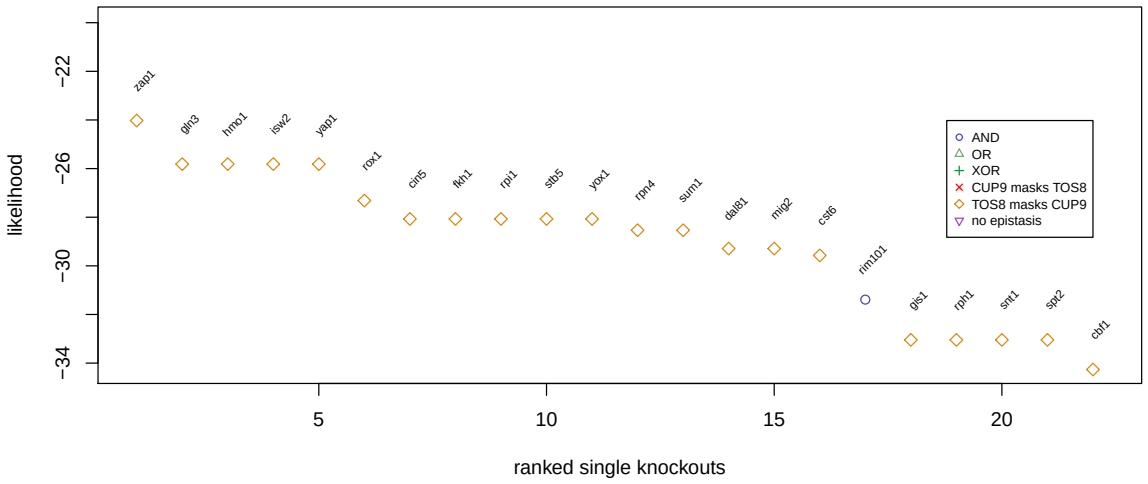

cst6 and mal13
likelihood
−1120
−1130
−1140
−1150
−1160
0
10
20
30
40
50
ranked single knockouts
AND
OR
XOR
CST6 masks MAL13
MAL13 masks CST6
no epistasis
cbf1
stb5
dal81
yap1
zap1
rpn4
isw2
tda9
gln3
rim101
mal33
rsf2
smp1
upc2
yox1
snt1
spt2
fkh1
stp4
pho2
rox1
gis1
msn2
stp3
cup9
nrg1
fkh2
msn4
rgm1
tod6
yrm1
ppr1
asg1
haa1
hys1
mig2
arg2
rpi1
sut2
usv1
mig1
met31
ygr067c
cad1
rph1
hcm1
sum1
gzf3
hel2
hmo1

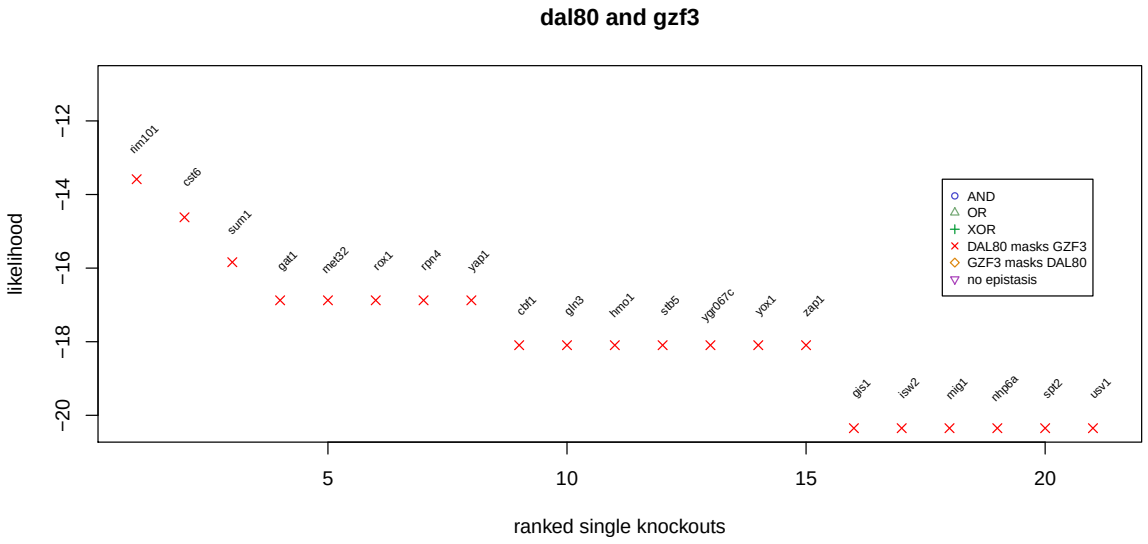

cup2 and haa1
likelihood
−22
−24
−26
−28
−30
5
10
15
ranked single knockouts
AND
OR
XOR
CUP2 masks HAA1
HAA1 masks CUP2
no epistasis
cbf1
stb5
cst6
dal81
fkh2
hmo1
mig1
msn2
rim101
rph1
sum1
nrg2
fkh1
yap6
asg1
mig2
tda9

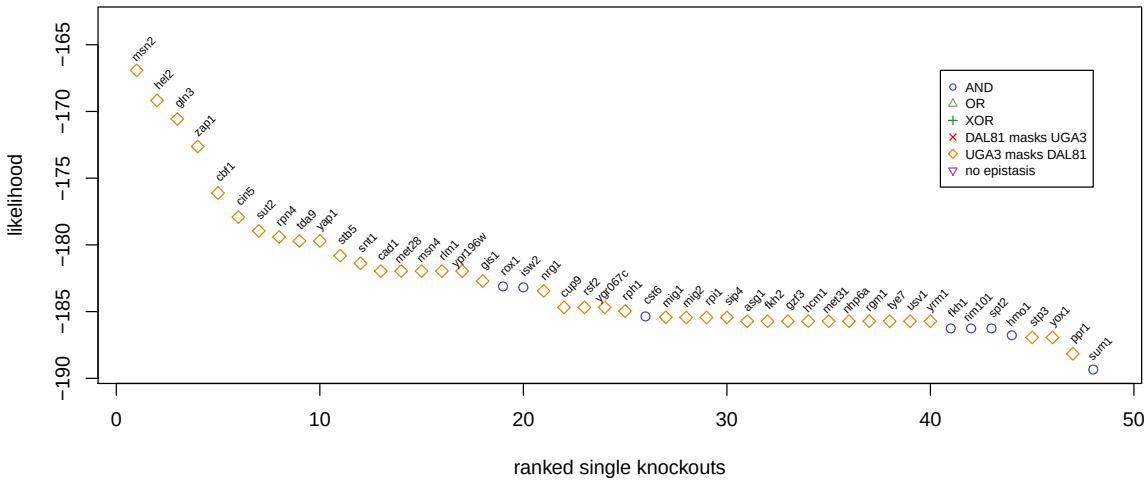
cup9 and rpn4

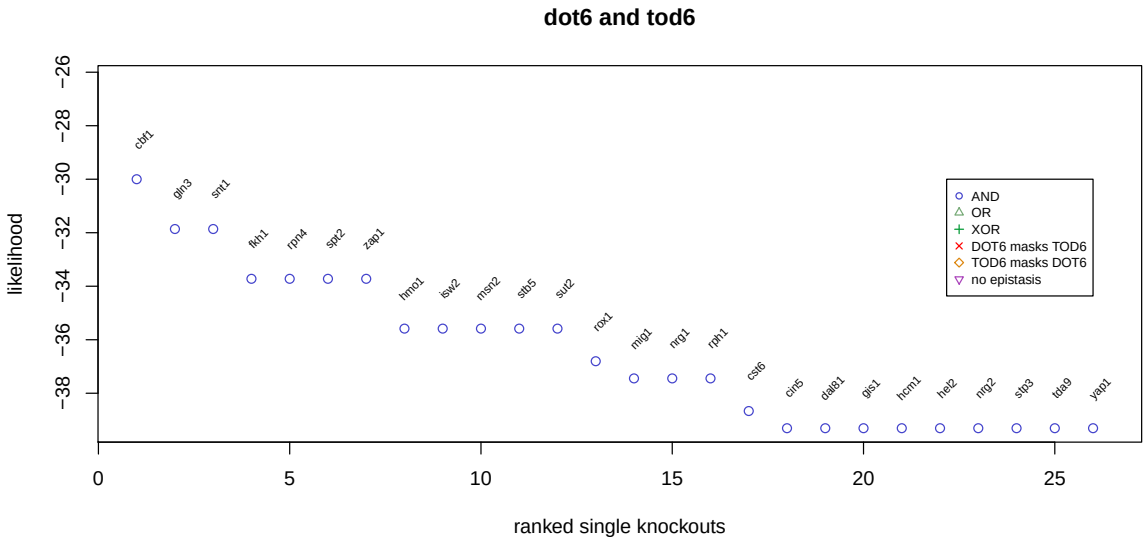


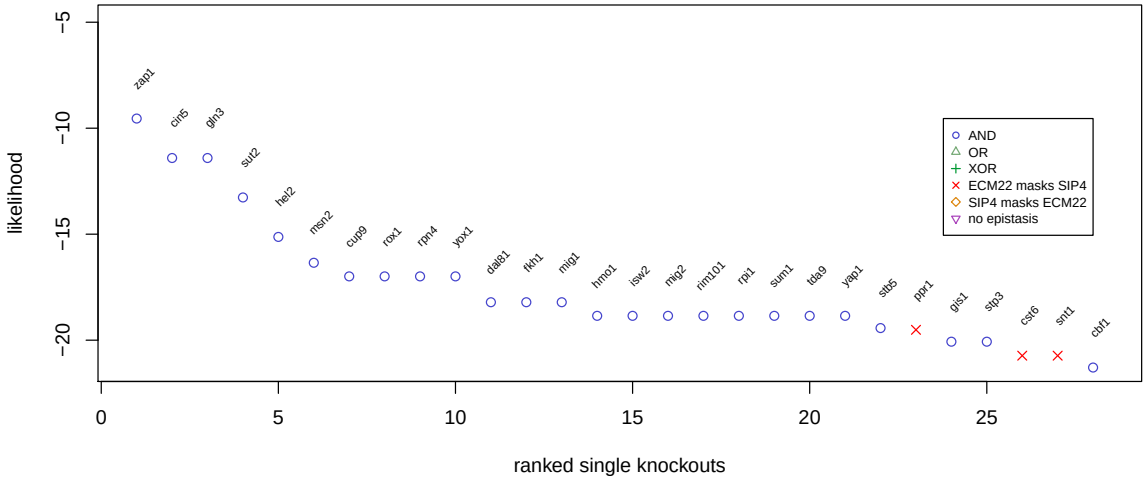
cup9 and tos8

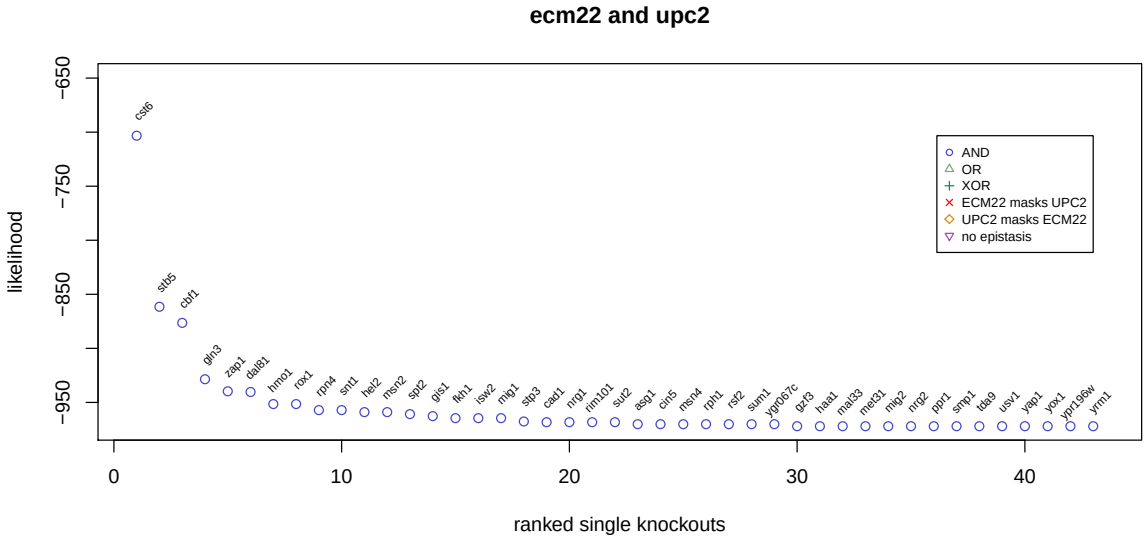


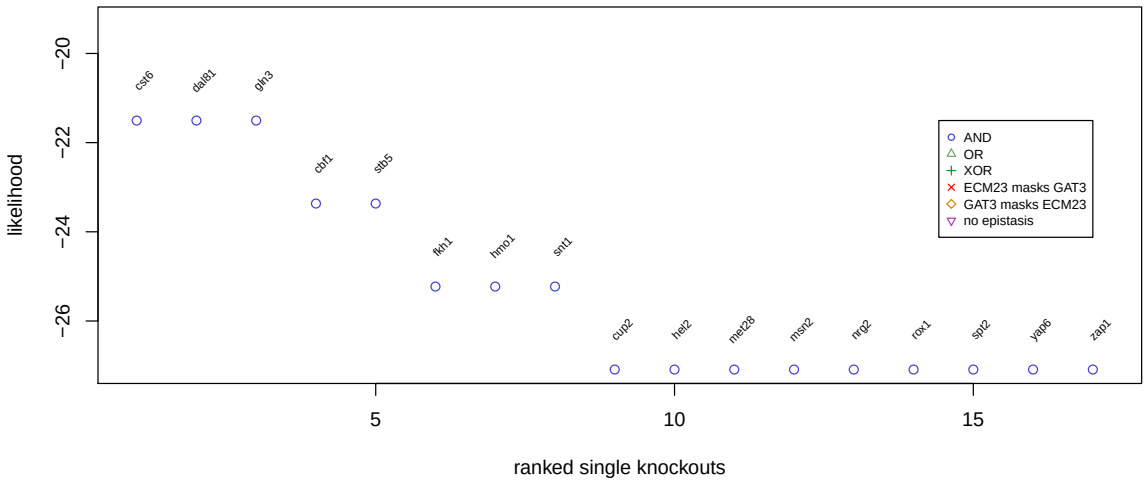

dal80 and gzf3
likelihood
−12
−14
−16
−18
−20
5
10
15
20
ranked single knockouts
rim101
csl6
sum1
gat1
met32
rox1
rpn4
yap1
cbf1
gln3
hmo1
stb5
ygr067c
yox1
zap1
gis1
isw2
mig1
nhp6a
spt2
usv1
AND
OR
XOR
DAL80 masks GZF3
GZF3 masks DAL80
no epistasis

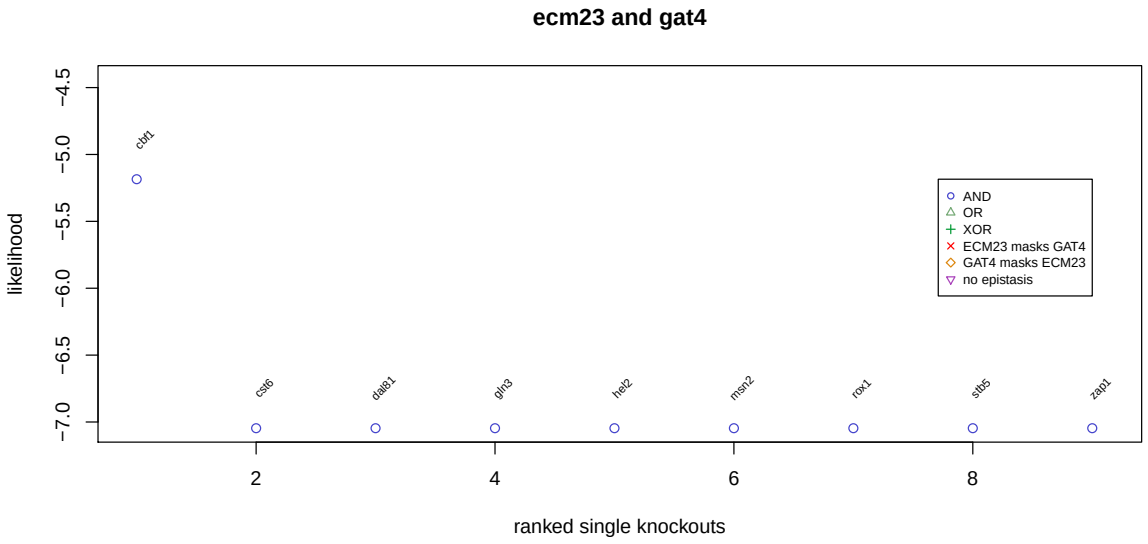

dal81 and uga3
likelihood
−165
−170
−175
−180
−185
−190
0
10
20
30
40
50
ranked single knockouts
msn2
het2
gln3
zap1
cbf1
ctn5
sut2
rpn4
tda9
yap1
stb5
sfl1
cad1
met28
msn4
rlm1
ygr196w
gis1
rox1
isw2
nrg1
cup9
rsf2
ygr067c
rph1
csl6
mig1
mig2
rpi1
skn7
asg1
fkh2
gzf3
hcm1
met31
nhp6a
rgm1
tye7
usv1
yrm1
fkh1
rim101
spt2
hmo1
stb3
yox1
ppr1
sum1
AND
OR
XOR
DAL81 masks UGA3
UGA3 masks DAL81
no epistasis

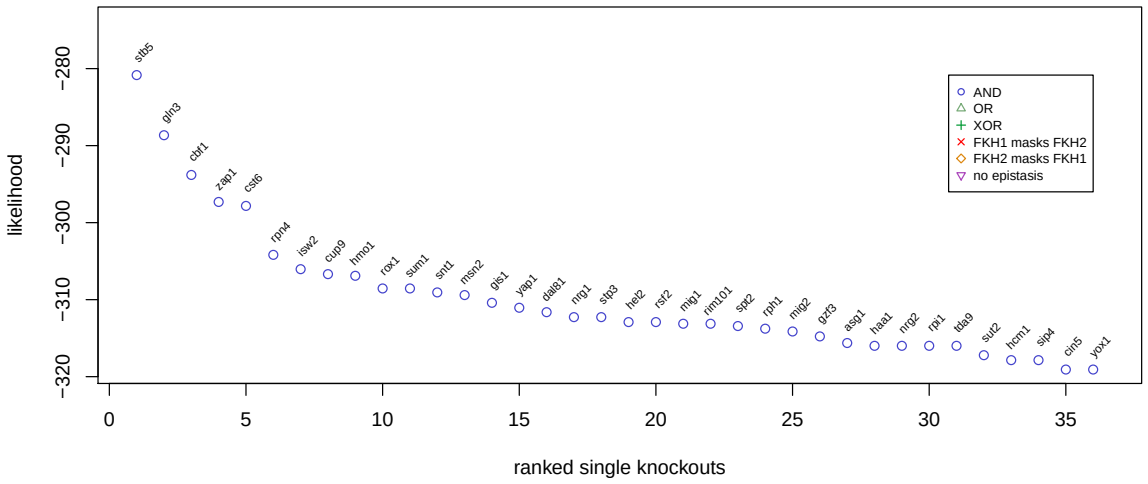

dot6 and tod6
likelihood
−26
−28
−30
−32
−34
−36
−38
0
5
10
15
20
25
ranked single knockouts
cbf1
gln3
snt1
fkh1
rpn4
spt2
zap1
hmo1
isw2
msn2
stb5
sut2
rox1
mig1
nrg1
rph1
cst6
cin5
dal81
gis1
hcm1
hel2
nrg2
stp3
tda9
yap1
AND
OR
XOR
DOT6 masks TOD6
TOD6 masks DOT6
no epistasis

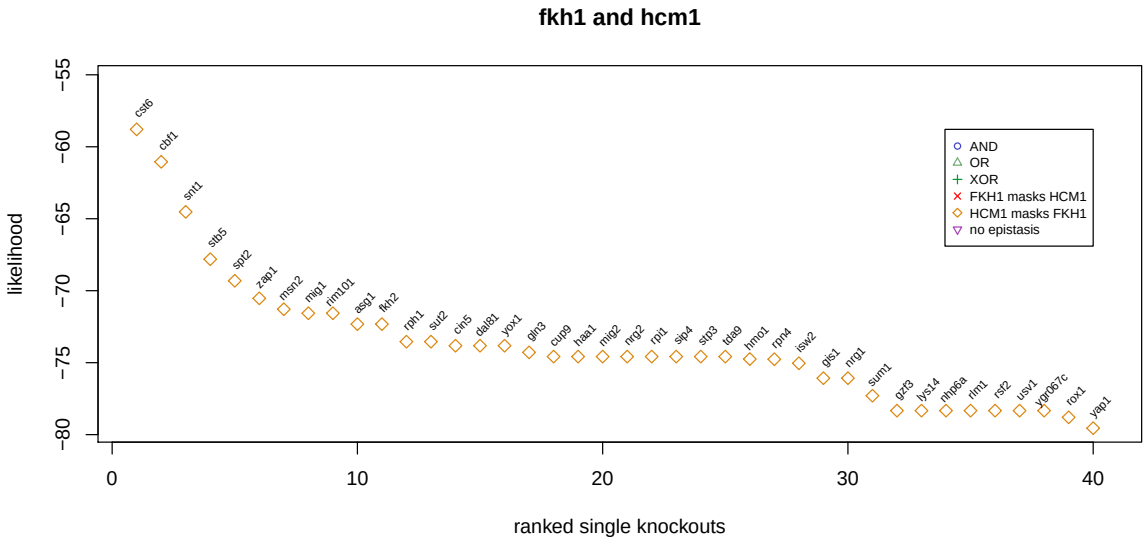

ecm22 and sip4
likelihood
−5
−10
−15
−20
0
5
10
15
20
25
ranked single knockouts
zap1
cin5
gln3
sut2
hel2
msn2
cup9
rox1
rpn4
yox1
dal81
fkh1
mig1
hmo1
isw2
nrg2
rim101
rpi1
sum1
tda9
yap1
stb5
ppr1
gis1
stp3
cst6
snt1
cbf1
AND
OR
XOR
ECM22 masks SIP4
SIP4 masks ECM22
no epistasis

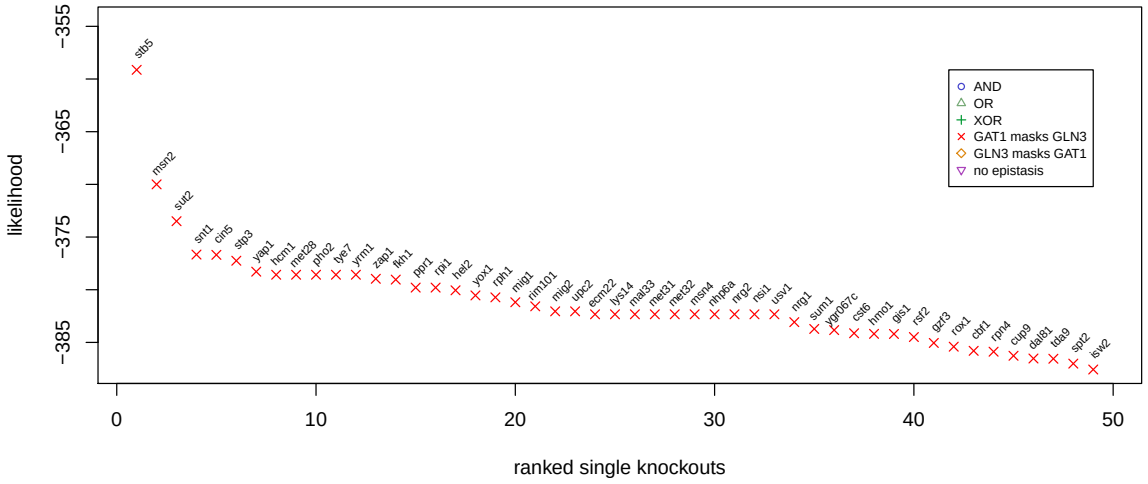

ecm22 and upc2
likelihood
−650
−750
−850
−950
0
10
20
30
40
ranked single knockouts
cst6
stb5
cbf1
gln3
zap1
dal81
hmo1
rox1
rpn4
snt1
hel2
msn2
spt2
gis1
fkh1
isw2
mig1
stp3
cad1
nrg1
rim101
sut2
asg1
cin5
msn4
rph1
rsf2
sum1
ygl067c
gat3
ttaa1
mal33
met31
mig2
nrg2
ppr1
smp1
tda9
usv1
yap1
yox1
ypr196w
yrm1
AND
OR
XOR
ECM22 masks UPC2
UPC2 masks ECM22
no epistasis

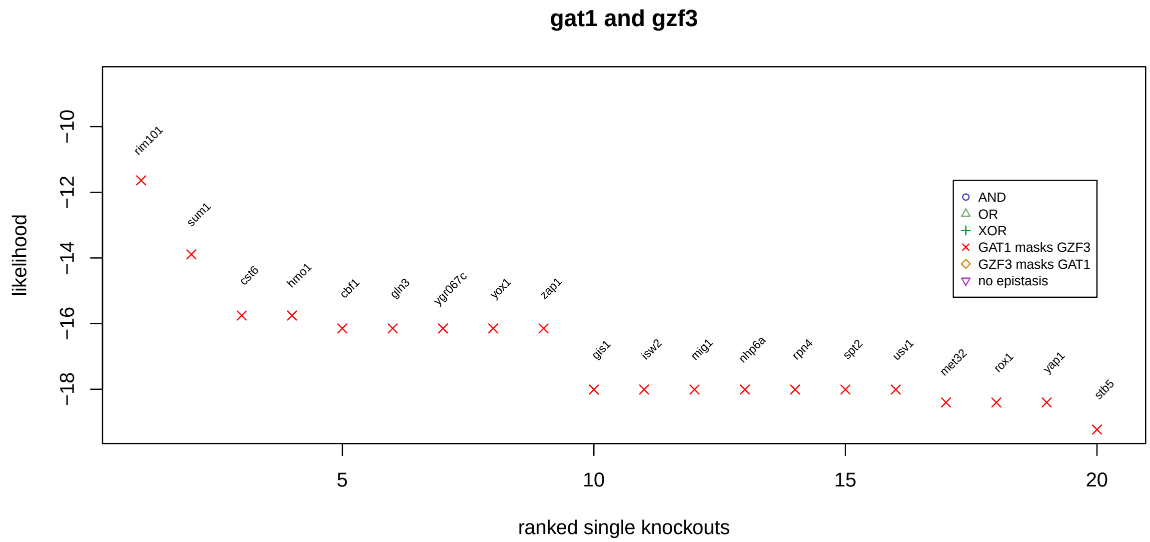

ecm23 and gat3
likelihood
−20
−22
−24
−26
5
10
15
ranked single knockouts
cst6
dal81
gln3
cbf1
stb5
fkh1
hmo1
snt1
cup2
hel2
met28
msn2
nrg2
rox1
spt2
yap6
zap1
AND
OR
XOR
ECM23 masks GAT3
GAT3 masks ECM23
no epistasis


ecm23 and gat4
likelihood
−4.5
−5.0
−5.5
−6.0
−6.5
−7.0
cbf1
csf6
dal81
gln3
hel2
msn2
rox1
stb5
zap1
AND
OR
XOR
ECM23 masks GAT4
GAT4 masks ECM23
no epistasis
2
4
6
8
ranked single knockouts

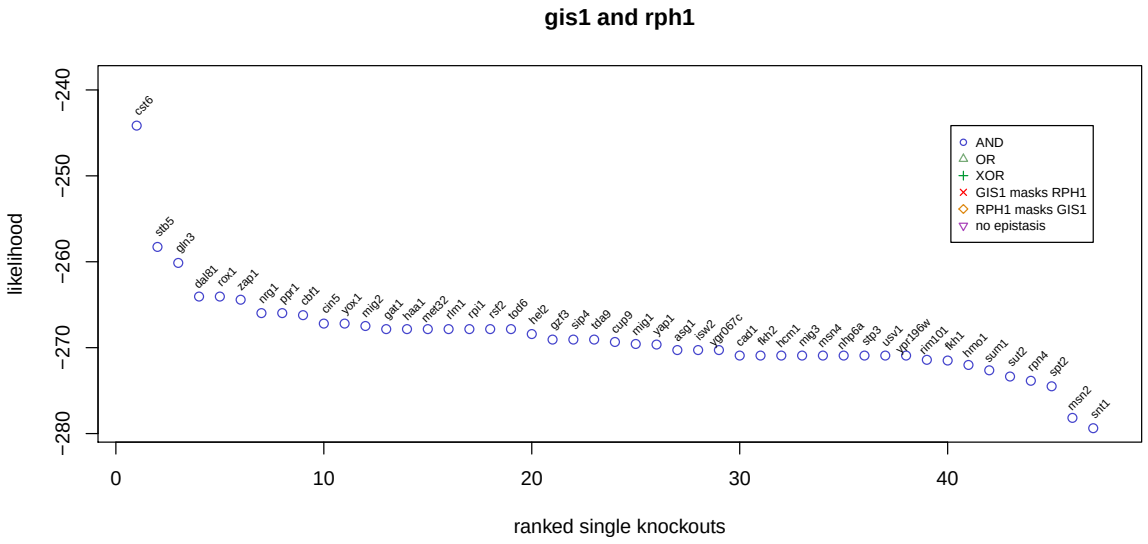

fkh1 and fkh2
likelihood
−280
−290
−300
−310
−320
stb5
gln3
cbf1
zap1
csf6
rpn4
isw2
cup9
hmo1
rox1
sum1
snt1
msn2
gis1
yap1
dal81
mgi1
stp3
hel2
rsr2
mig1
rim101
spt2
rph1
mig2
gzf3
asg1
haa1
mrp2
rpi1
tda9
sul2
hcm1
stp4
cin5
yox1
AND
OR
XOR
FKH1 masks FKH2
FKH2 masks FKH1
no epistasis
0
5
10
15
20
25
30
35
ranked single knockouts

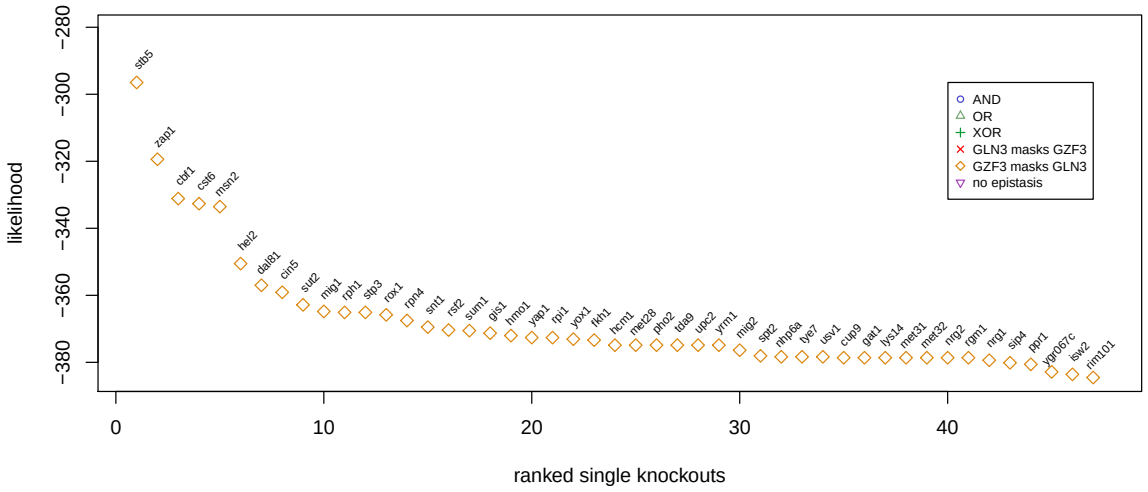

fkh1 and hcm1
likelihood
−55
−60
−65
−70
−75
−80
0
10
20
30
40
ranked single knockouts
AND
OR
XOR
FKH1 masks HCM1
HCM1 masks FKH1
no epistasis
csi6
cbf1
snt1
stb5
spi2
zap1
msn2
mig1
rim101
ssg1
fkh2
rph1
sut2
cin5
dal81
yox1
gln3
cup9
haa1
mig2
nrg2
rpi1
stp4
stp3
tda9
hmo1
rpn4
isw2
gis1
nrg1
sum1
grf3
lys14
rhp6a
rlm1
rsf2
usv1
ygr067c
rox1
yap1


gat1 and gln3
likelihood
−355
−365
−375
−385
0
10
20
30
40
50
ranked single knockouts
AND
OR
XOR
GAT1 masks GLN3
GLN3 masks GAT1
no epistasis
stb5
msn2
sut2
snt1
cin5
stp3
yap1
hcm1
met28
pho2
bye7
ynm1
zap1
fkh1
ppr1
rpi1
hel2
yox1
rph1
mig1
rim101
mig2
upc2
ecm22
lys14
mal33
met31
met32
msn4
rhp6a
nrg2
rsi1
usv1
nrg1
sum1
ygr067c
csi6
hmo1
gis1
rsf2
grf3
rox1
cbf1
rpn4
cup9
dal81
tda9
spi2
isw2

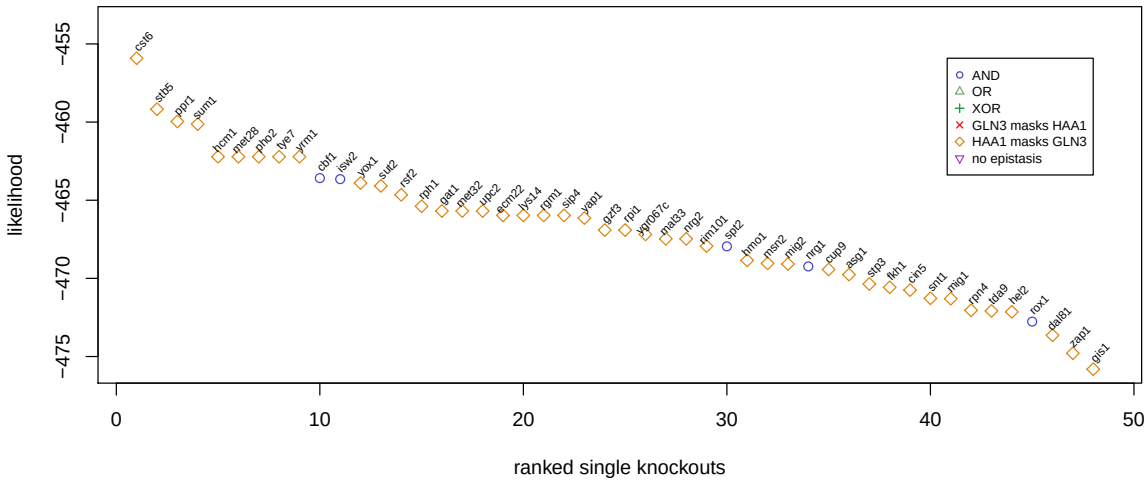

gat1 and gzf3
likelihood
−10
−12
−14
−16
−18
rim101
sum1
cst6
hmo1
cbf1
gln3
ygr067c
yox1
zap1
gis1
isw2
mig1
nhp6a
rpn4
spt2
usv1
met32
rox1
yap1
stb5
AND
OR
XOR
GAT1 masks GZF3
GZF3 masks GAT1
no epistasis
5
10
15
20
ranked single knockouts


gat4 and hmo1
likelihood
−255
−260
−265
−270
cbf1
rim101
spt2
stb5
isw2
rpn4
gis1
ygr067c
gzf3
mal33
nhp6a
ppr1
rlm1
rsf2
usv1
sum1
cst6
cin5
hcm1
rph1
stp3
yox1
msn2
rox1
dal81
asg1
cad1
fkh2
met28
msn4
ygr196w
nrg1
mig2
pho2
rpi1
tye7
mig1
fkh1
sut2
snt1
yap1
lys14
cup9
zap1
gln3
AND
OR
XOR
GAT4 masks HMO1
HMO1 masks GAT4
no epistasis
0
10
20
30
40
ranked single knockouts

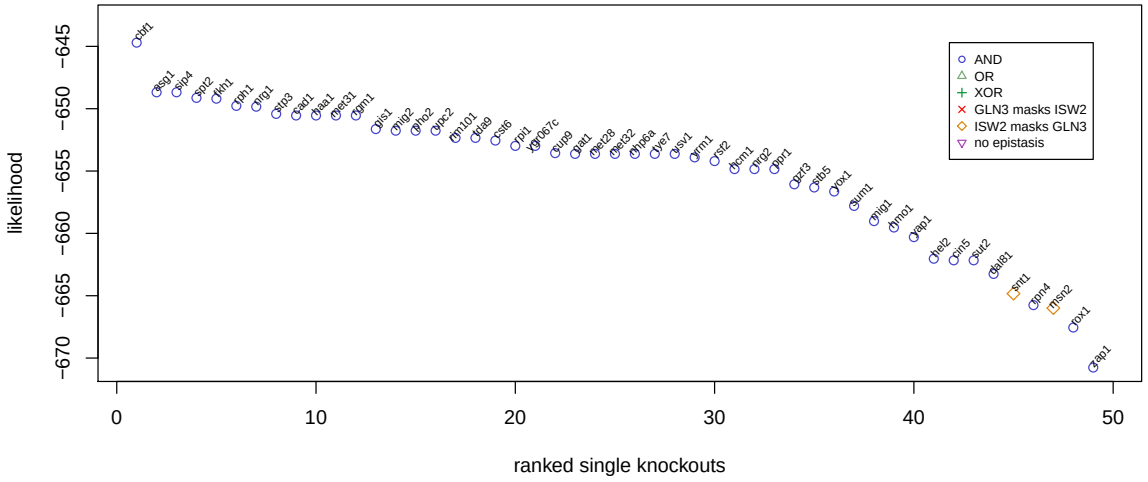

gis1 and rph1
likelihood
−240
−250
−260
−270
−280
0
10
20
30
40
ranked single knockouts
AND
OR
XOR
GIS1 masks RPH1
RPH1 masks GIS1
no epistasis
cst6
stb5
gln3
dal81
rox1
zap1
nrg1
pox1
cbf1
cin5
yox1
mig2
gat1
haa1
met32
rim1
rpi1
rsf2
tod6
hel2
gzf3
sip4
tda9
cup9
mig1
yap1
asg1
isw2
ygr067c
cad1
fkh2
hcm1
mig3
msn4
nhp6a
stp3
usv1
ygr196w
rim101
fkh1
hmo1
sum1
sul2
rpn4
spt2
msn2
snt1


gln3 and gzf3
likelihood
−280
−300
−320
−340
−360
−380
0
10
20
30
40
ranked single knockouts
AND
OR
XOR
GLN3 masks GZF3
GZF3 masks GLN3
no epistasis
stb5
zap1
cbf1
cst6
msn2
hel2
dal81
cin5
sut2
mig1
rph1
stp3
rox1
rpn4
snt1
rsf2
sum1
gis1
hmo1
yap1
rpi1
yox1
fkh1
hcm1
met28
pho2
tda9
upc2
yrm1
mig2
spt2
nhp6a
tye7
usv1
cup9
gat1
lys14
met31
met32
nrg2
rgm1
nrg1
sip4
ppr1
ygr067c
isw2
rim101

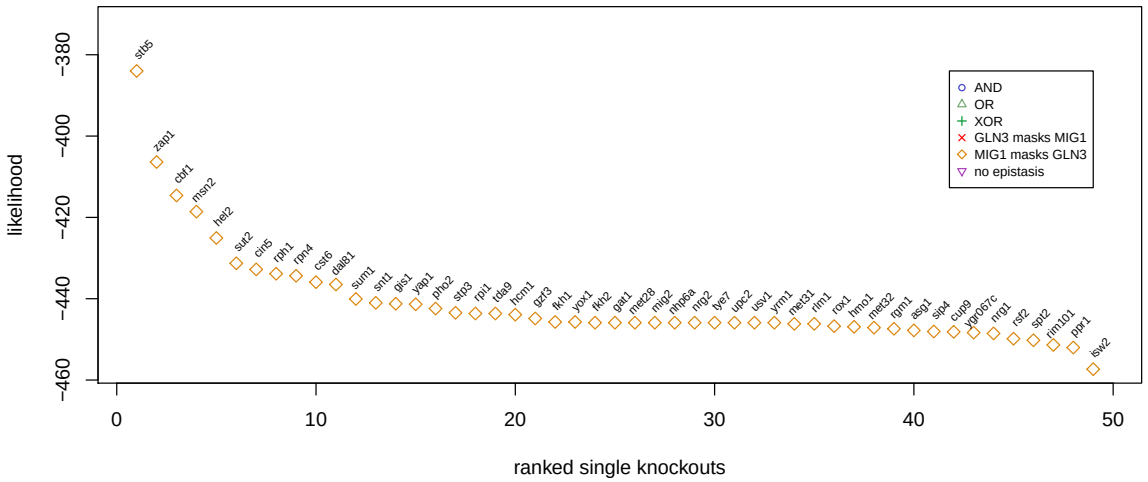
gln3 and haa1



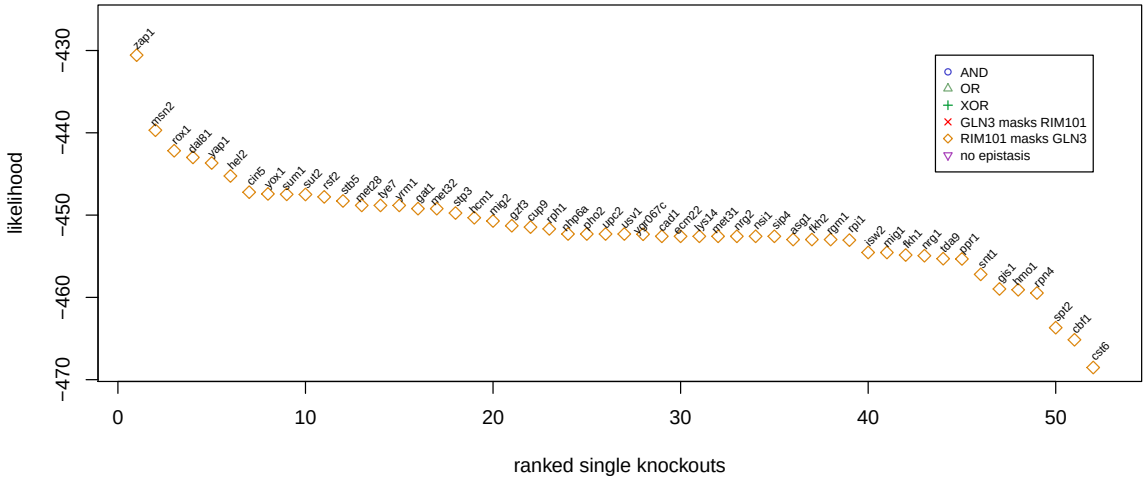
gln3 and isw2

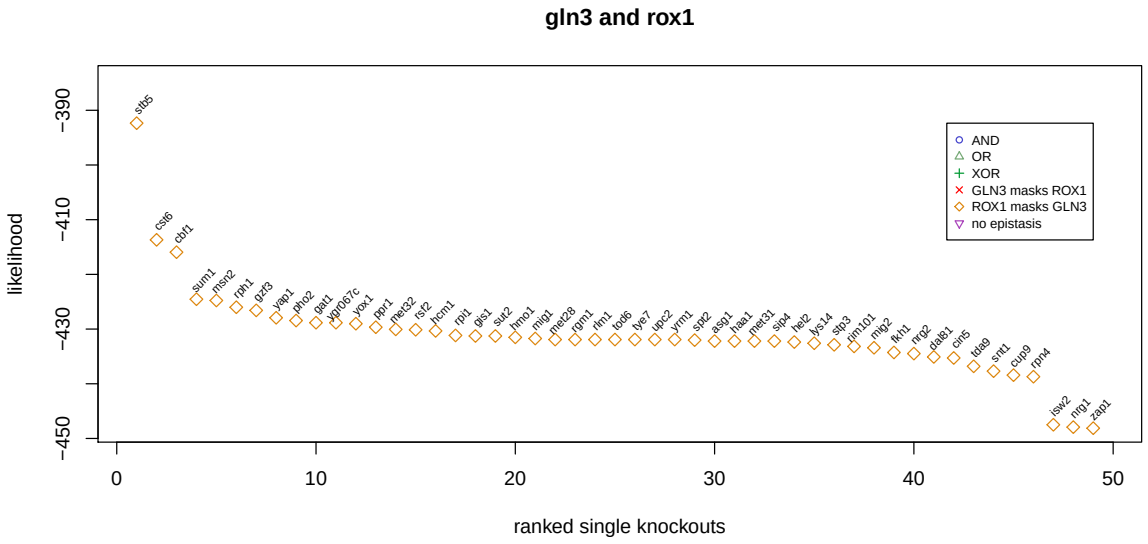


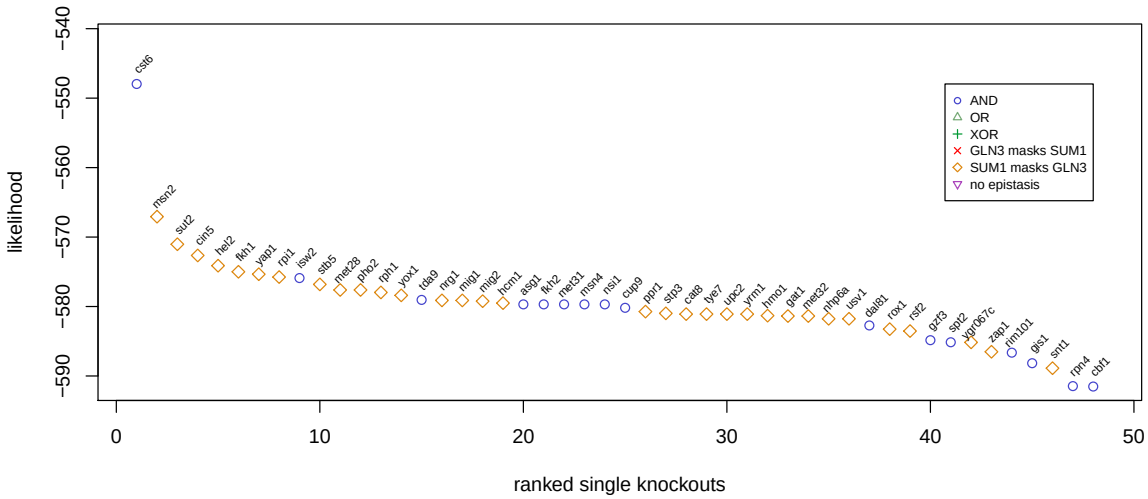
gln3 and mig1



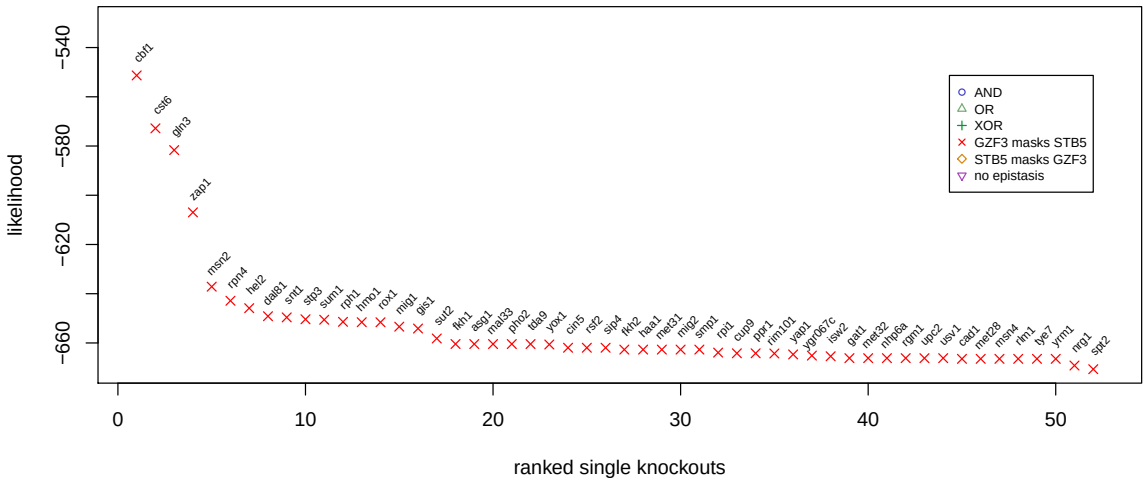
gln3 and rim101



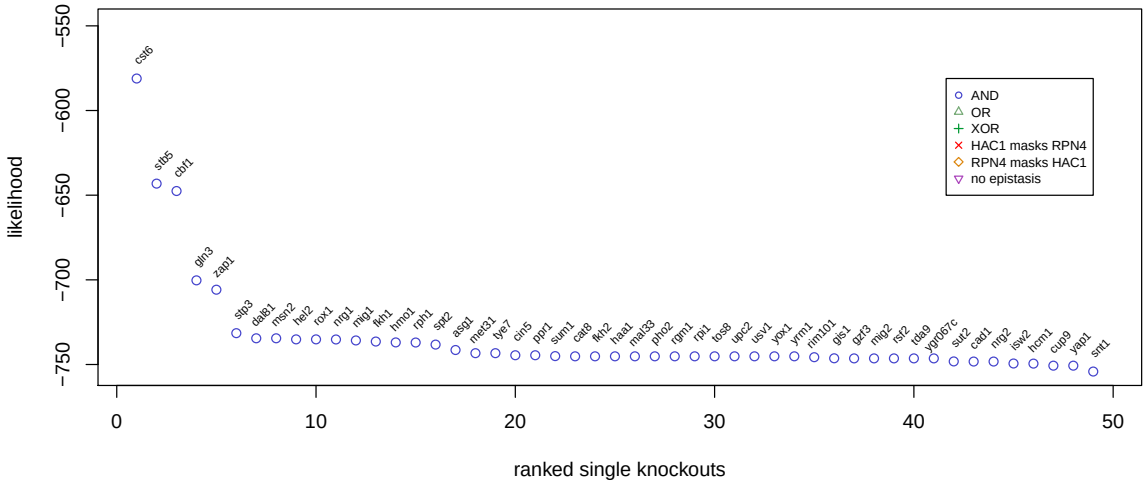

gln3 and rox1
likelihood
−390
−410
−430
−450
0
10
20
30
40
50
ranked single knockouts
AND
OR
XOR
GLN3 masks ROX1
ROX1 masks GLN3
no epistasis
stb5
csi6
cbf1
sum1
msn2
rph1
pzf3
yap1
pho2
gat1
ygr067c
yox1
ppr1
met32
rsf2
hcm1
rpi1
gis1
sut2
hmo1
mig1
met28
rgm1
tfn1
tod6
tye7
upc2
yrm1
stp2
asg1
haa1
met31
stp4
hel2
lys14
stp3
rim101
mig2
fkh1
nrg2
dal81
cin5
tda9
snt1
cup9
rpn4
isw2
nrg1
zap1



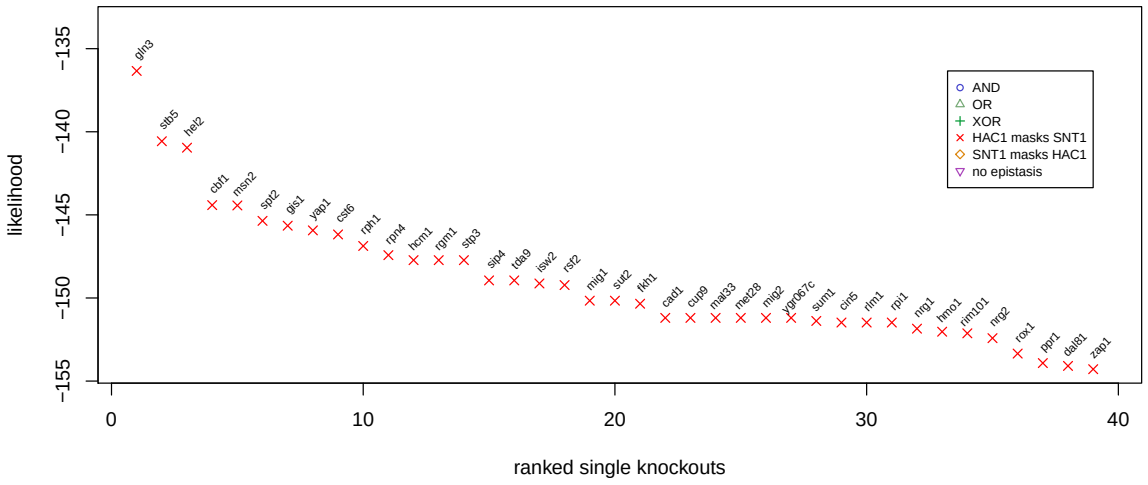
gzf3 and stb5



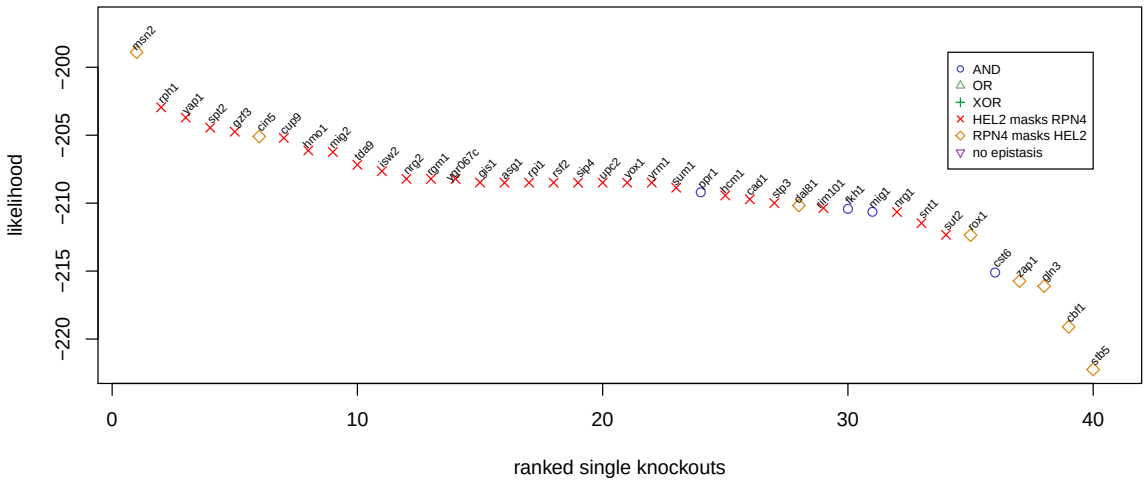
hac1 and rpn4



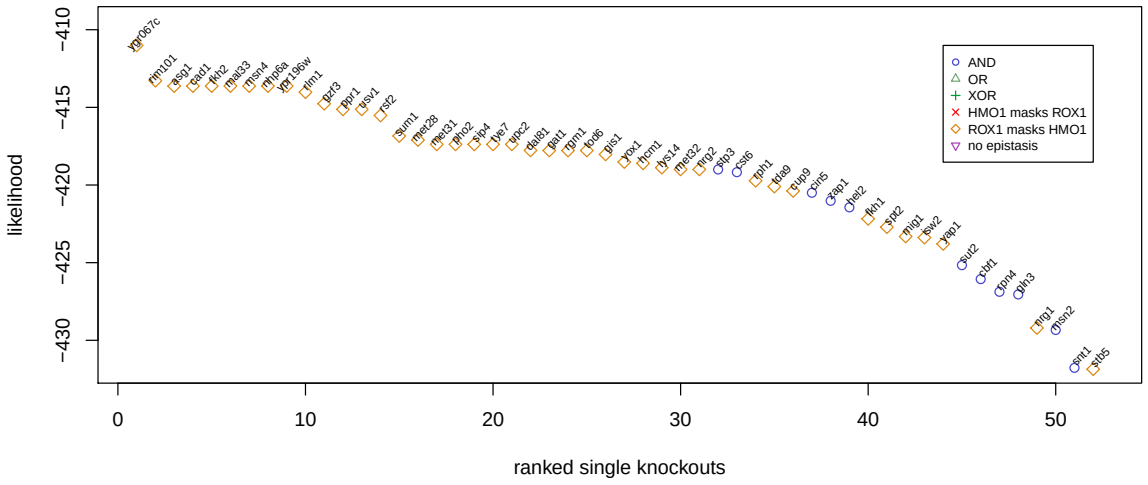
hac1 and snt1



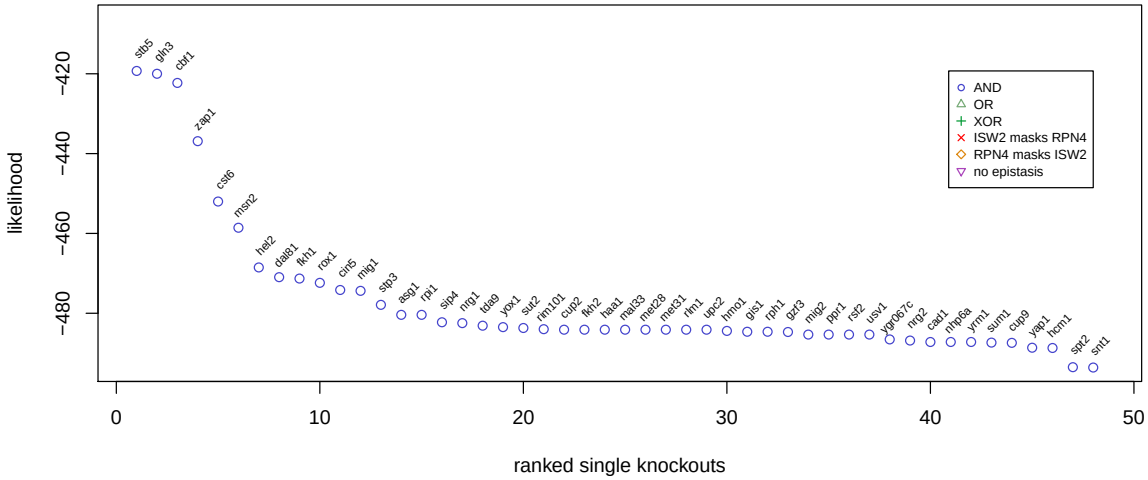
hel2 and rpn4

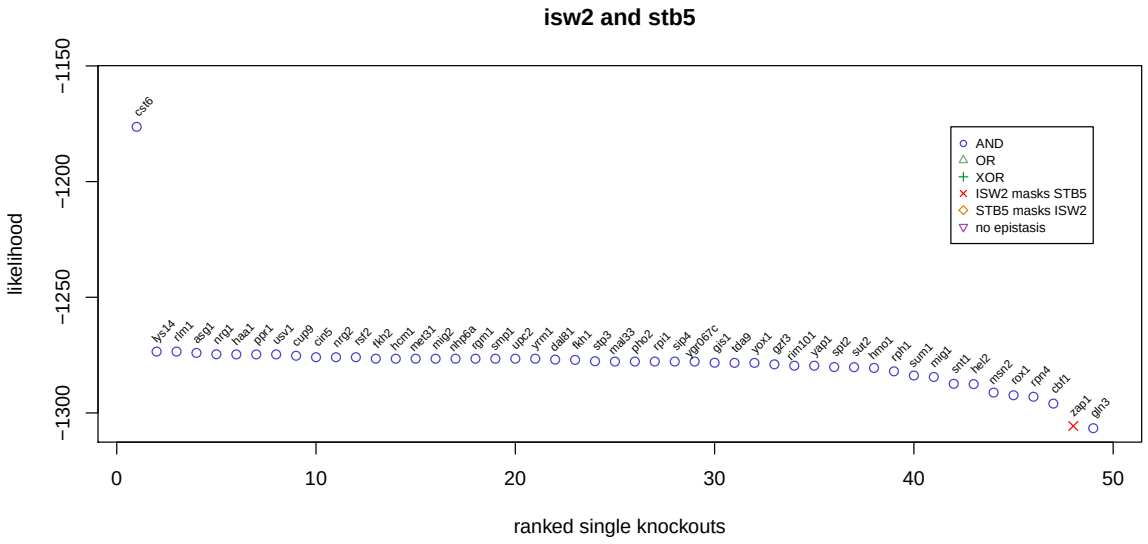


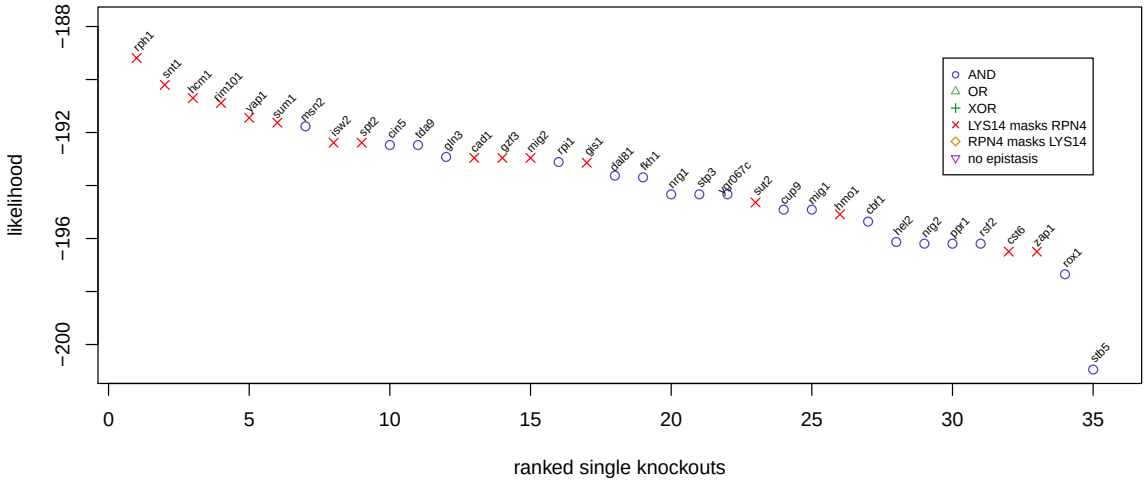
hmo1 and rox1

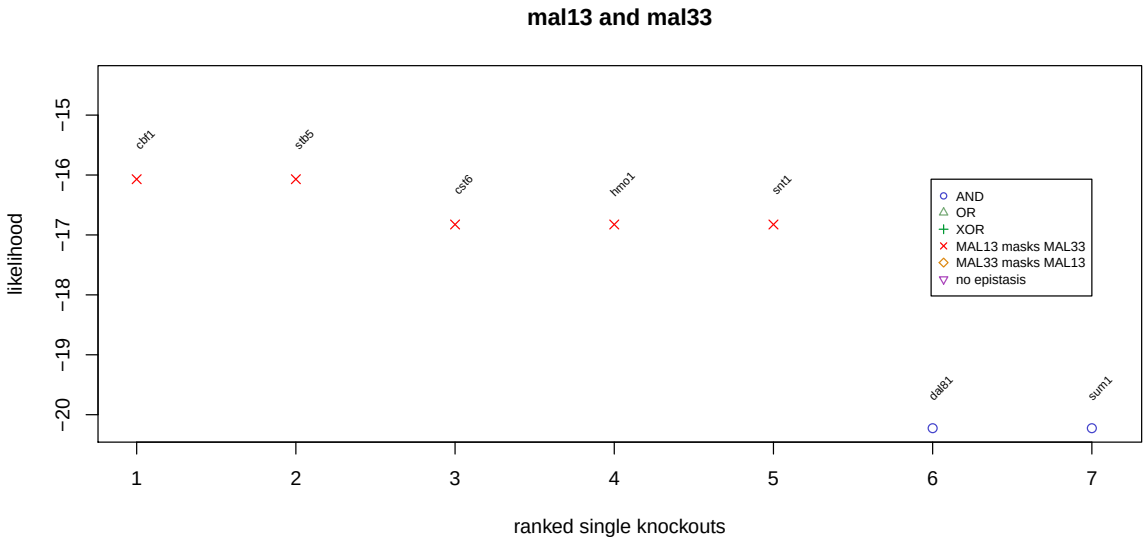


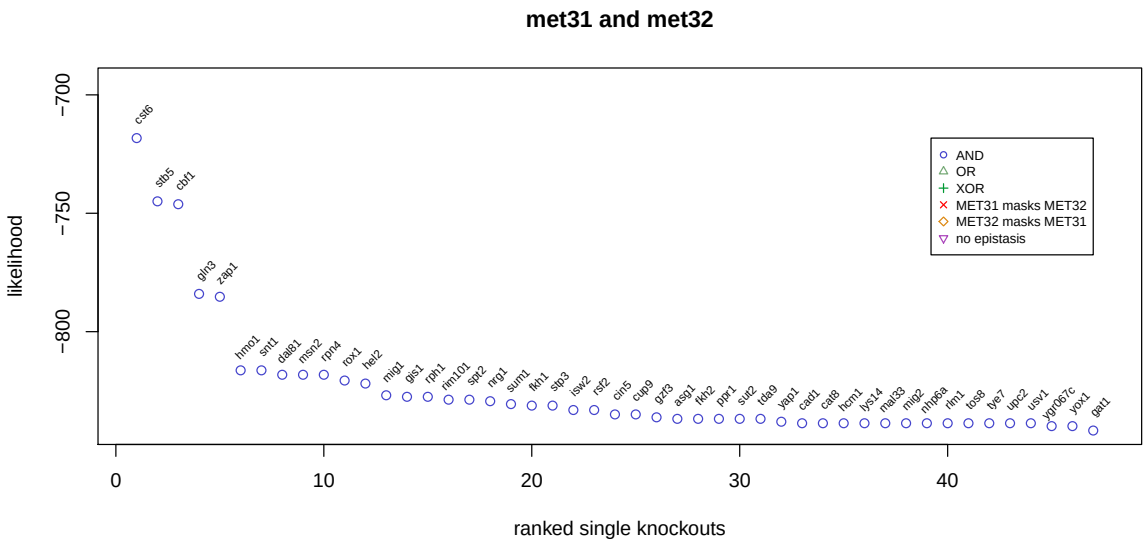
isw2 and rpn4



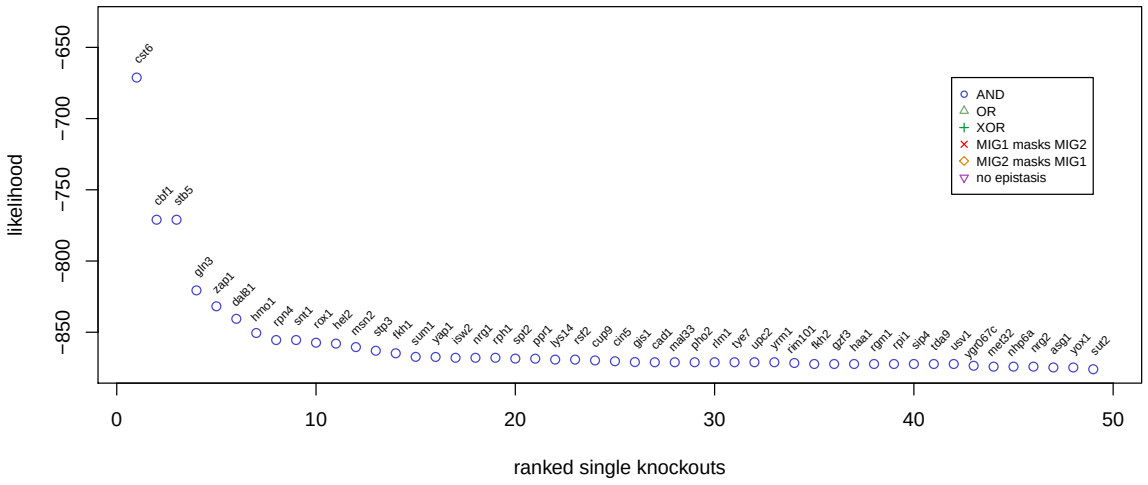

isw2 and stb5
likelihood
−1150
−1200
−1250
−1300
0
10
20
30
40
50
ranked single knockouts
AND
OR
XOR
ISW2 masks STB5
STB5 masks ISW2
no epistasis
csi6
lys14
rim1
asg1
nrg1
haa1
ppr1
usv1
cup9
cin5
nrg2
rsf2
fkh2
hcm1
met31
mig2
nhp6a
rgm1
smp1
upc2
yrm1
dal81
fkh1
stp3
ma33
pho2
rpi1
stp4
ygr067c
gis1
tda9
yox1
pzf3
rim101
yap1
spt2
suf2
hmo1
rph1
sum1
mig1
snt1
hel2
msn2
rox1
rpn4
cbf1
zap1
gln3


lys14 and rpn4
likelihood
−188
−192
−196
−200
0
5
10
15
20
25
30
35
ranked single knockouts
AND
OR
XOR
LYS14 masks RPN4
RPN4 masks LYS14
no epistasis
rpn1
snt1
hcm1
rim101
yap1
sum1
msn2
isw2
spt2
cin5
tda9
gln3
cad1
pzf3
mig2
rpi1
gis1
dal81
fkh1
nrg1
stp3
ygr067c
suf2
cup9
mig1
hmo1
cbf1
hel2
nrg2
ppr1
rsf2
csi6
zap1
rox1
stb5

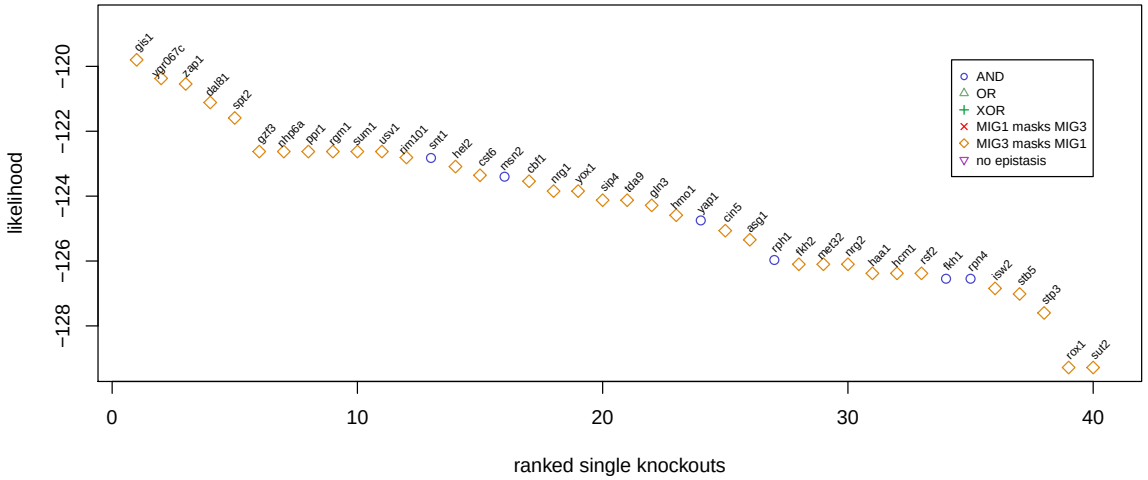

mal13 and mal33
likelihood
−15
−16
−17
−18
−19
−20
cbf1
stb5
csf6
hmo1
snt1
dal81
sum1
AND
OR
XOR
MAL13 masks MAL33
MAL33 masks MAL13
no epistasis
1
2
3
4
5
6
7
ranked single knockouts


met31 and met32
likelihood
−700
−750
−800
csf6
stb5
cbf1
gln3
zap1
hmo1
snt1
dal81
msn2
rpn4
rox1
hsf2
mig1
gis1
rph1
rim101
spt2
nrg1
sum1
fkh1
stp3
isw2
rsf2
cin5
cup9
prr3
asg1
fkh2
ppr1
sut2
tda9
yap1
cad1
cat8
hcm1
lys14
mal33
mig2
nhp6a
rfm1
tos8
bye7
ups2
usv1
ygr067c
yox1
gat1
AND
OR
XOR
MET31 masks MET32
MET32 masks MET31
no epistasis
0
10
20
30
40
ranked single knockouts

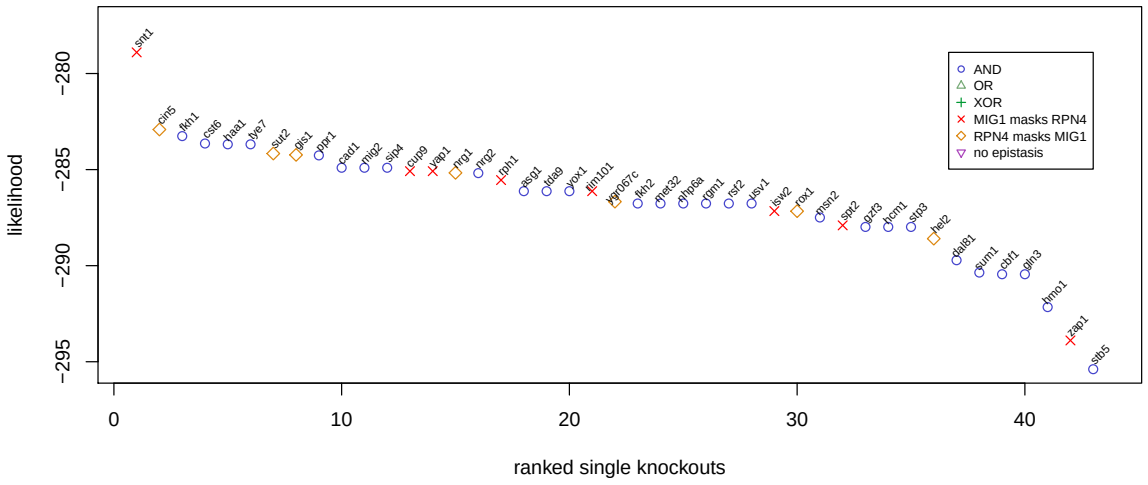
mig1 and mig2



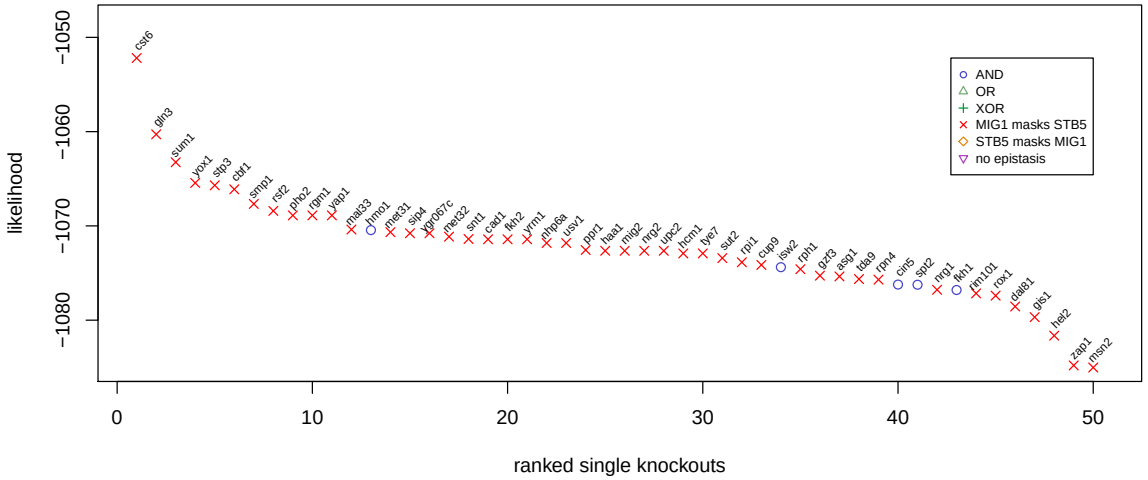
mig1 and mig3



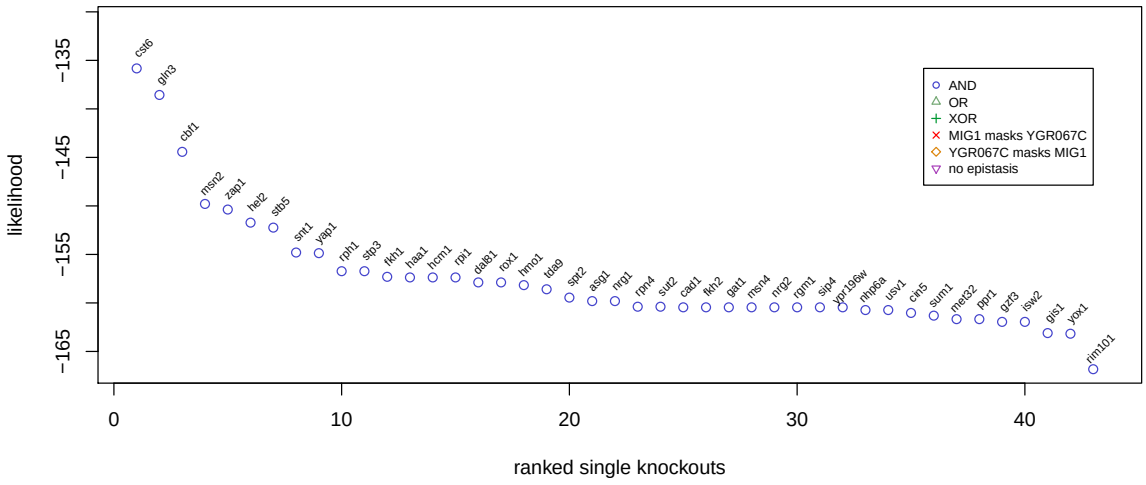
mig1 and rpn4



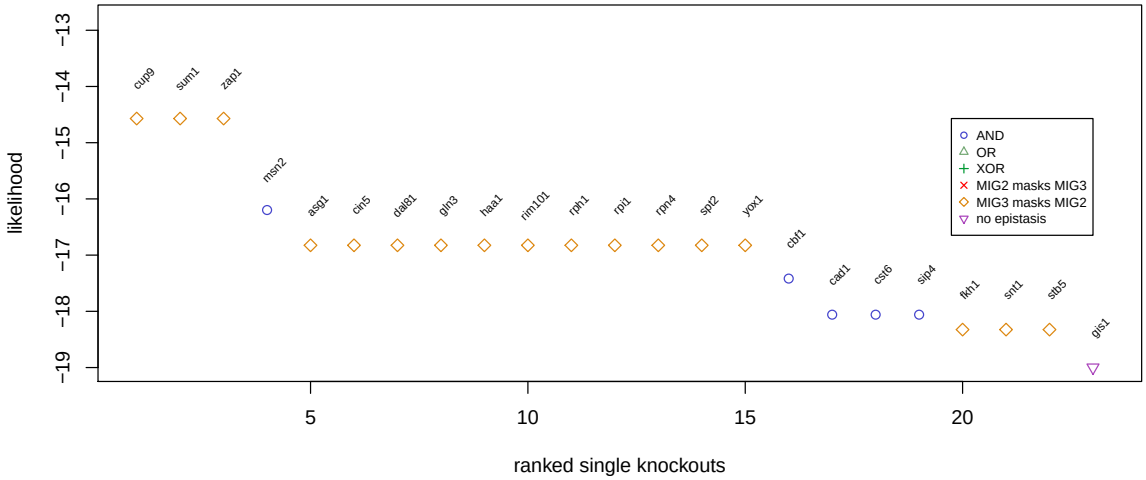
mig1 and stb5

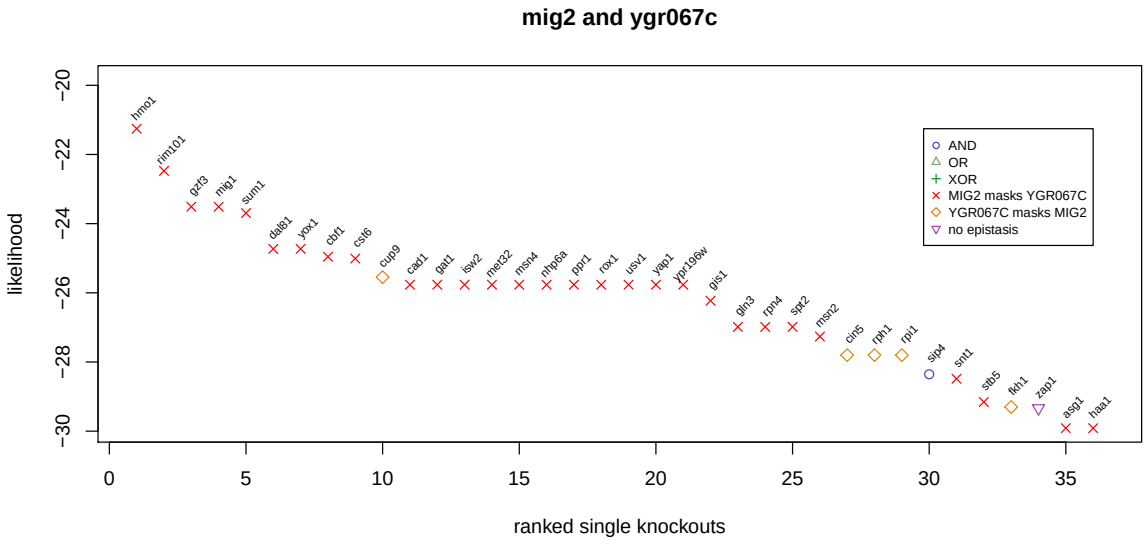


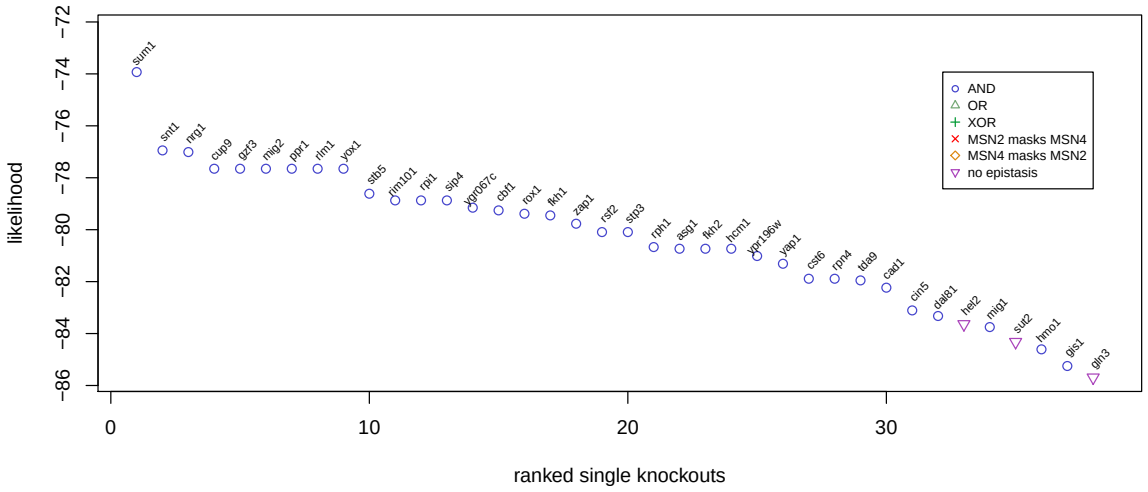
mig1 and ygr067c

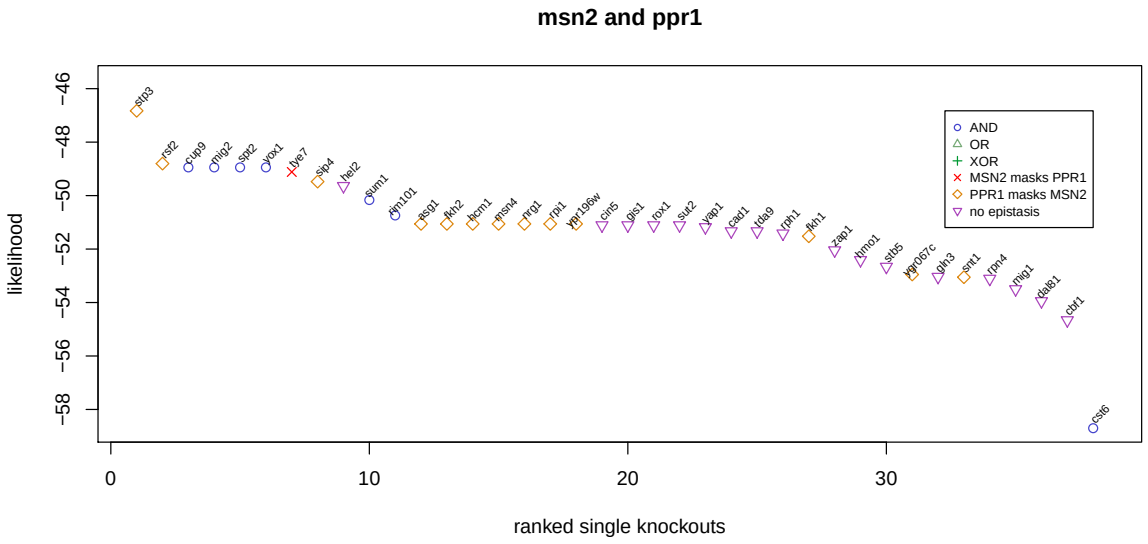


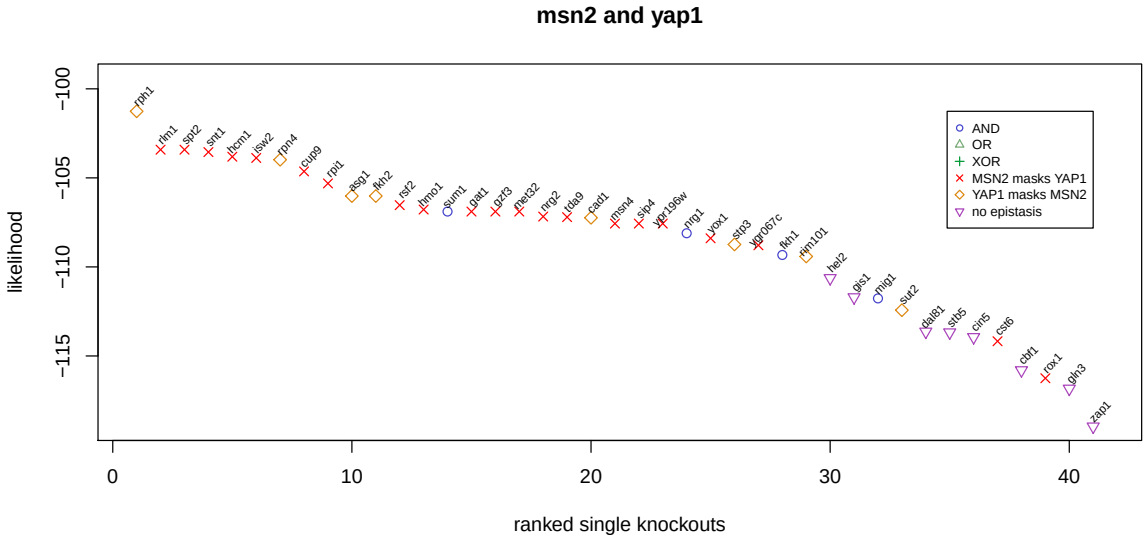
mig2 and mig3



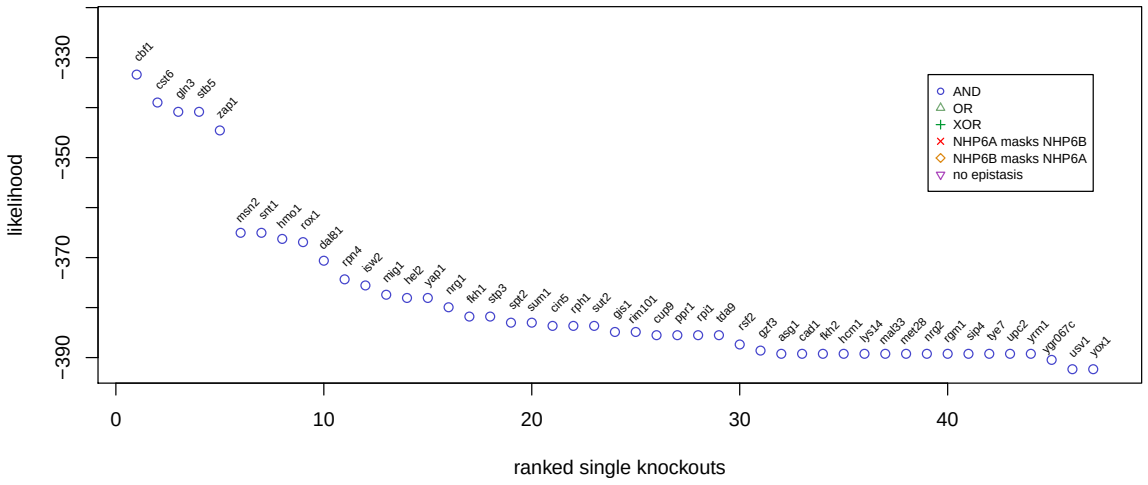

mig2 and ygr067c
likelihood
−20
−22
−24
−26
−28
−30
0
5
10
15
20
25
30
35
ranked single knockouts
hmo1
rim101
grf3
mig1
sum1
dal81
yox1
cbf1
cst6
cup9
cad1
gat1
isw2
met32
msn4
nhp6a
ppr1
rox1
usv1
yap1
ygr196w
gis1
gln3
rpn4
spt2
msn2
cin5
rph1
rpi1
stp4
snt1
stb5
fkh1
zap1
asg1
haa1
AND
OR
XOR
MIG2 masks YGR067C
YGR067C masks MIG2
no epistasis


msn2 and msn4
likelihood
−72
−74
−76
−78
−80
−82
−84
−86
0
10
20
30
ranked single knockouts
sum1
snt1
nrg1
cup9
grf3
mig2
ppr1
rim1
yox1
stb5
rim101
rpi1
sip4
ygr067c
cbf1
rox1
fkh1
zap1
rsf2
stp3
rpn1
asg1
fkh2
hcm1
ygr196w
yap1
cst6
rpn4
tda9
caf1
cin5
dal81
hel2
mig1
sut2
hmo1
gis1
gln3
AND
OR
XOR
MSN2 masks MSN4
MSN4 masks MSN2
no epistasis

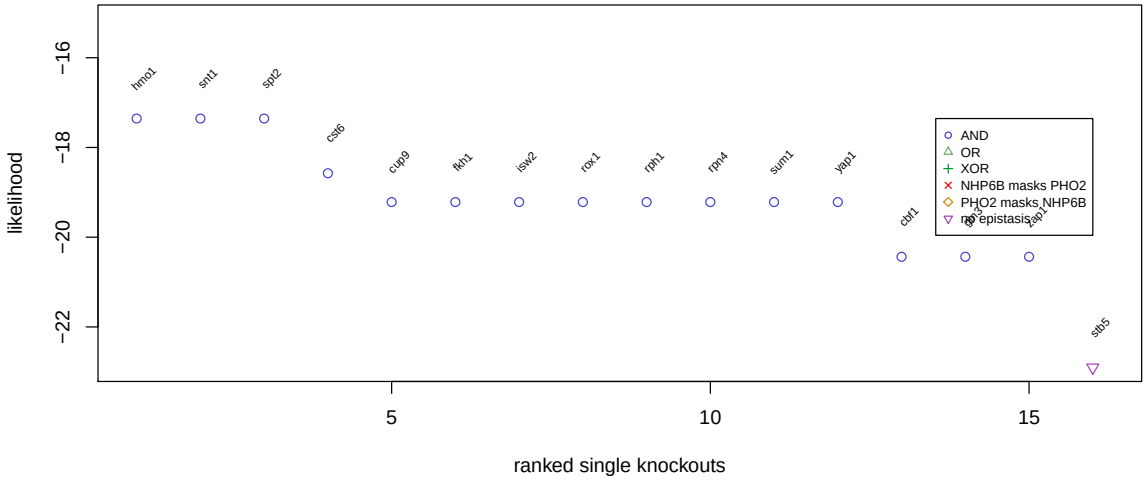

msn2 and ppr1
likelihood
−46
−48
−50
−52
−54
−56
−58
0
10
20
30
ranked single knockouts
AND
OR
XOR
MSN2 masks PPR1
PPR1 masks MSN2
no epistasis
stp3
rsf2
cup9
mig2
spt2
yox1
tye7
sip4
hst2
sum1
dim101
nsg1
fkh2
bcm1
msn4
nrg1
rpn1
ygr196w
cin5
ois1
rox1
sut2
yap1
cad1
tda9
rpn1
fkh1
zap1
ahc1
stb5
ygr067c
dln3
snt1
rpn4
mig1
dal81
ctf1
csi6


msn2 and yap1
likelihood
−100
−105
−110
−115
0
10
20
30
40
ranked single knockouts
AND
OR
XOR
MSN2 masks YAP1
YAP1 masks MSN2
no epistasis
rpn1
rlm1
spt2
snt1
bcm1
tsw2
rpn4
cup9
rpn1
nsg1
fkh2
rsf2
ahc1
sum1
gat1
prz3
het32
nrg2
tda9
cad1
msn4
sip4
ygr196w
mig1
rox1
stp3
ygr067c
fkh1
dim101
hst2
ois1
mig1
sut2
dal81
stb5
cin5
csi6
ctf1
rox1
dln3
zap1

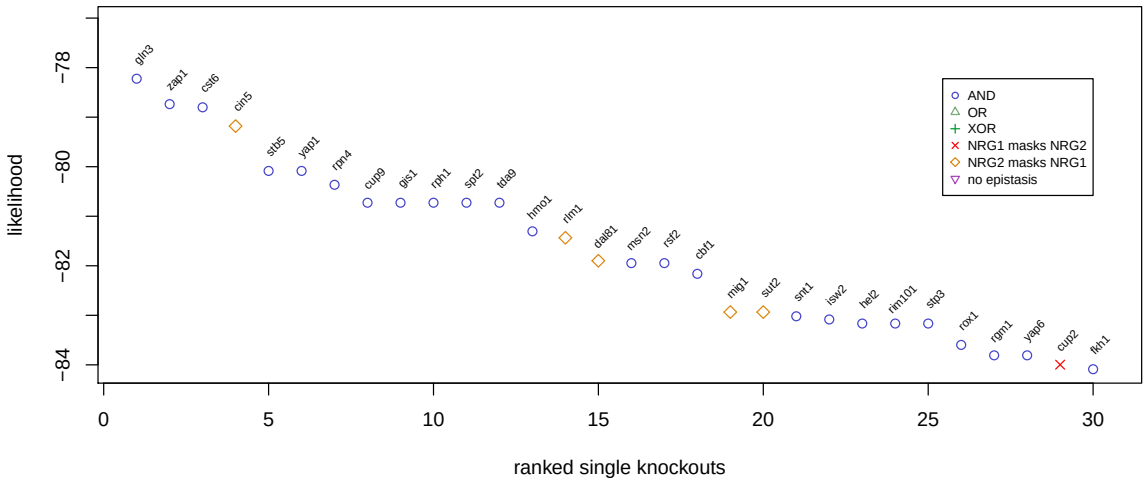
nhp6a and nhp6b



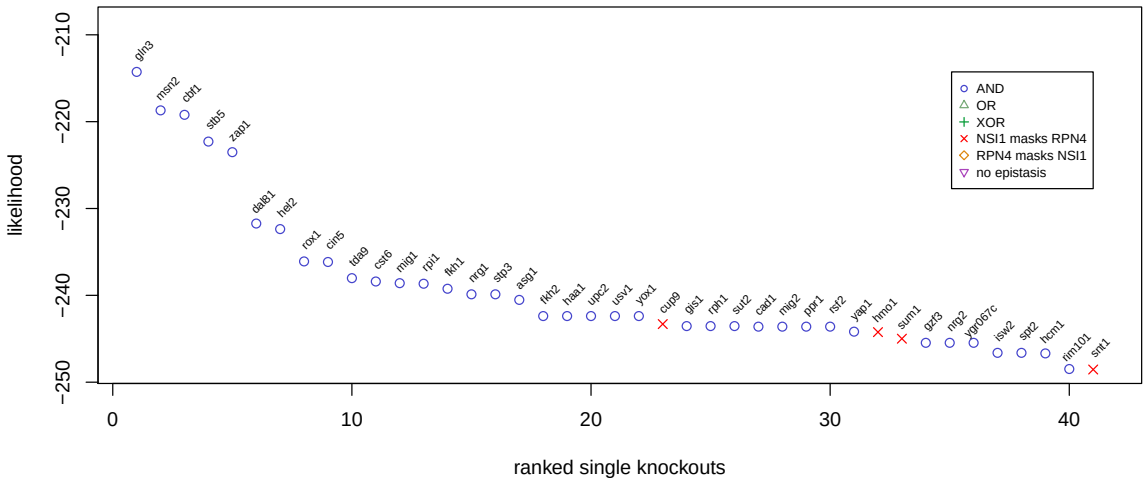
nhp6b and pho2



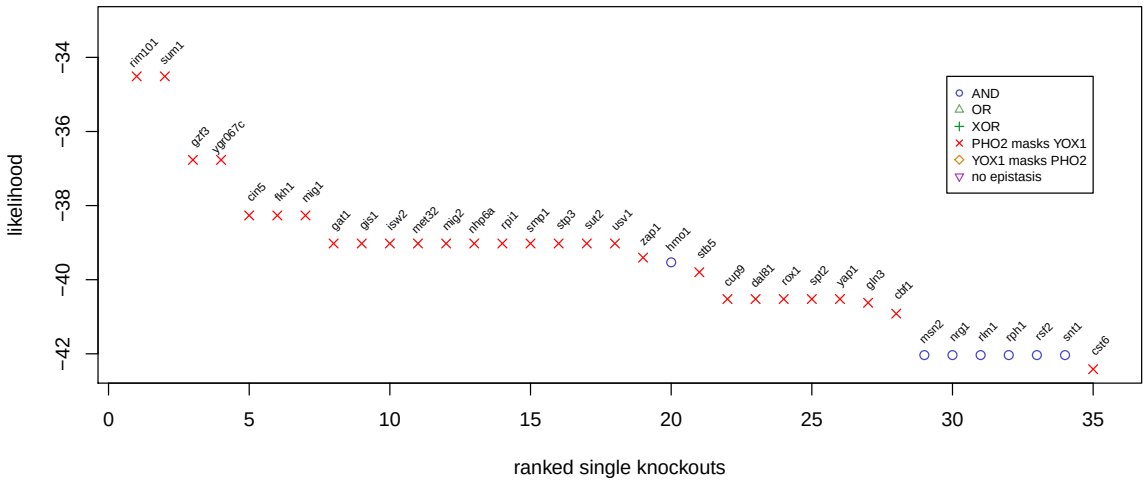
nrg1 and nrg2



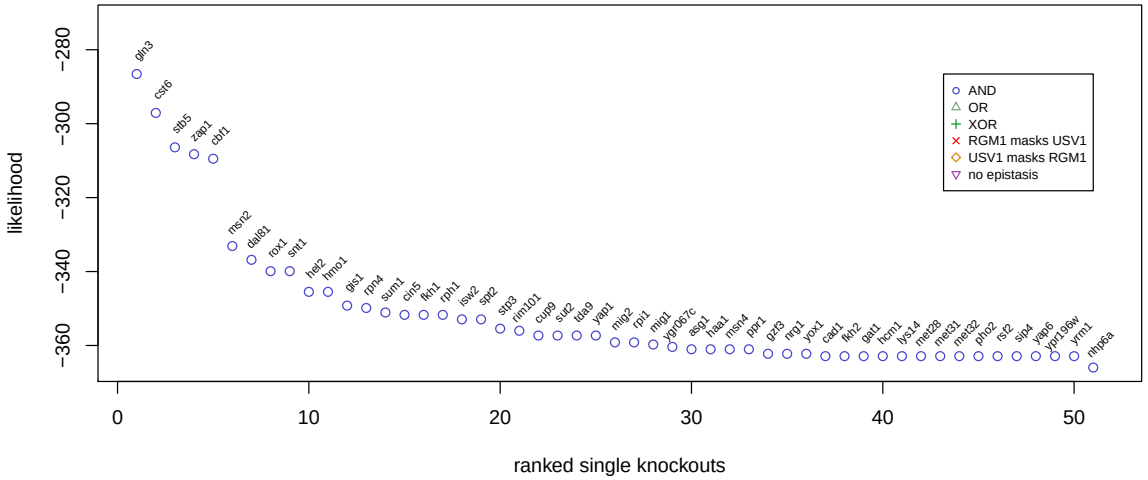
nsi1 and rpn4

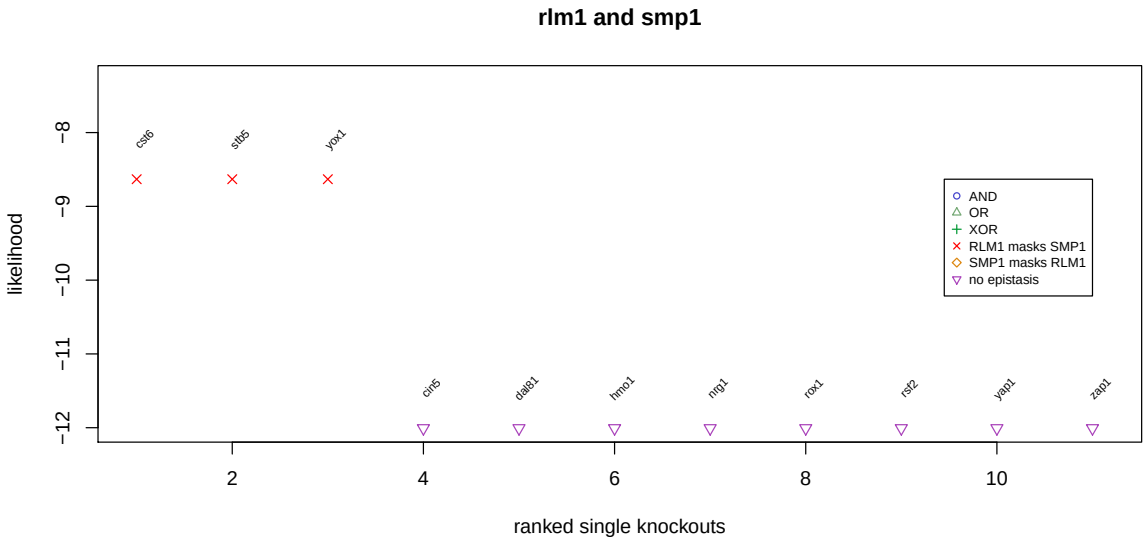


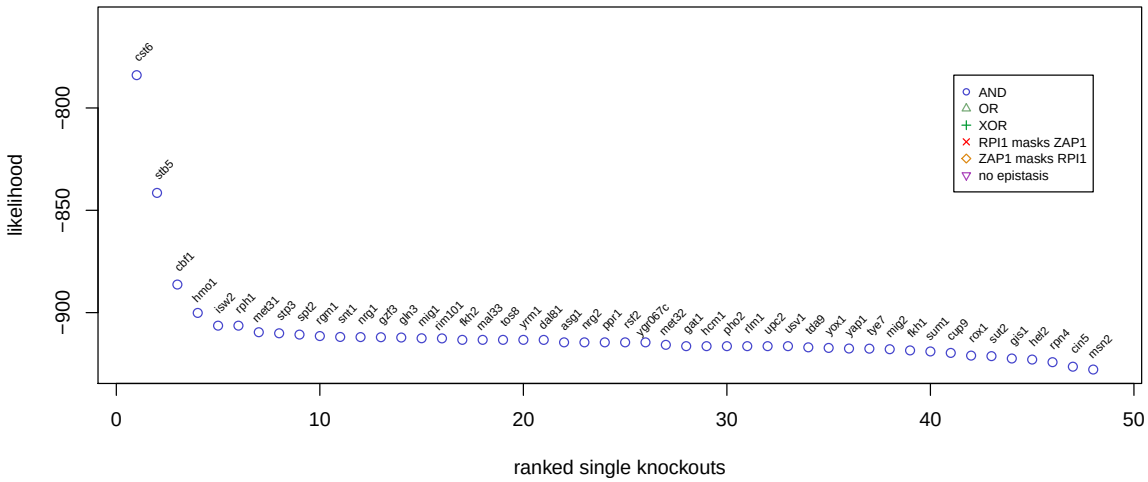
pho2 and yox1

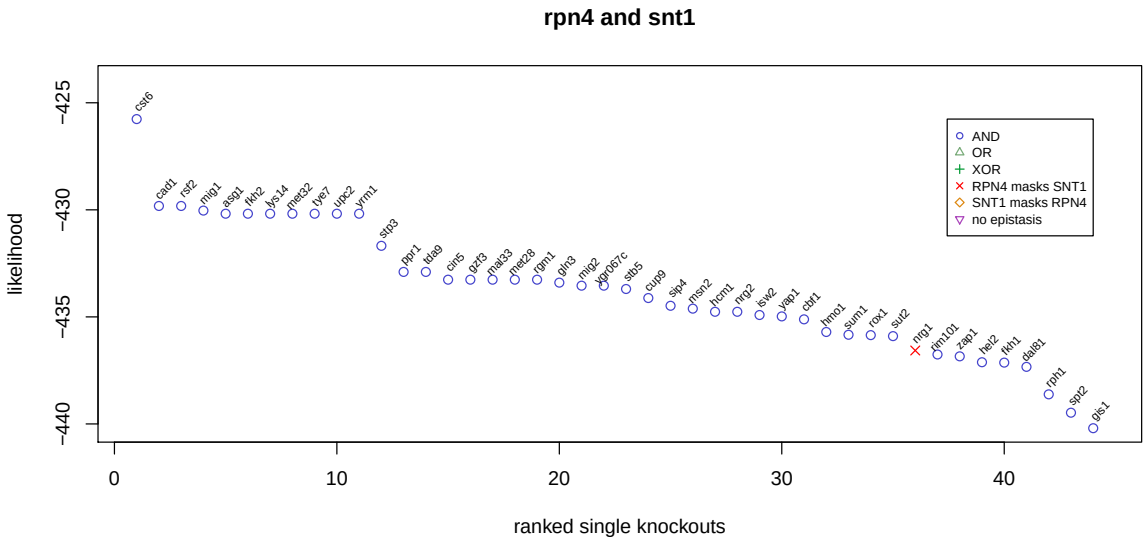


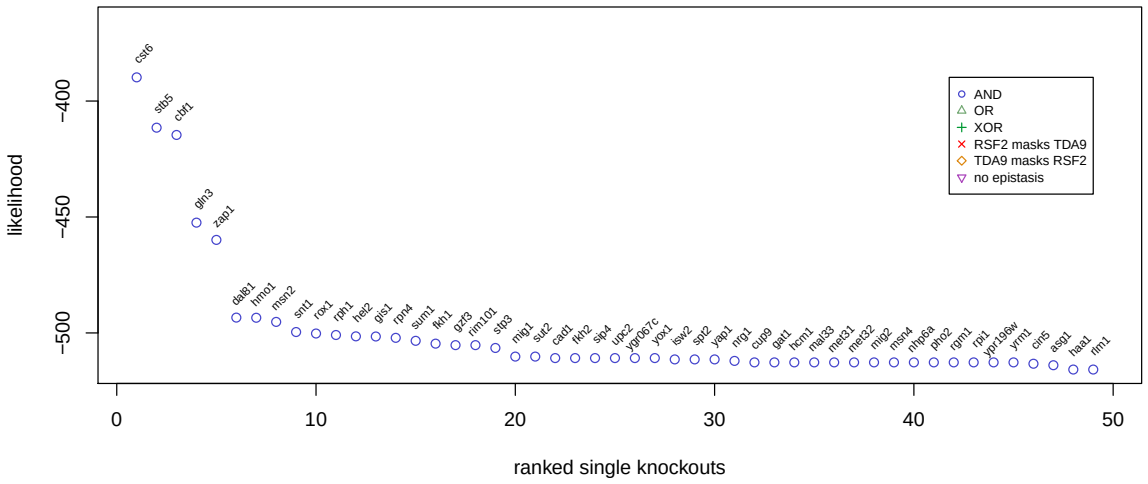
rgm1 and usv1

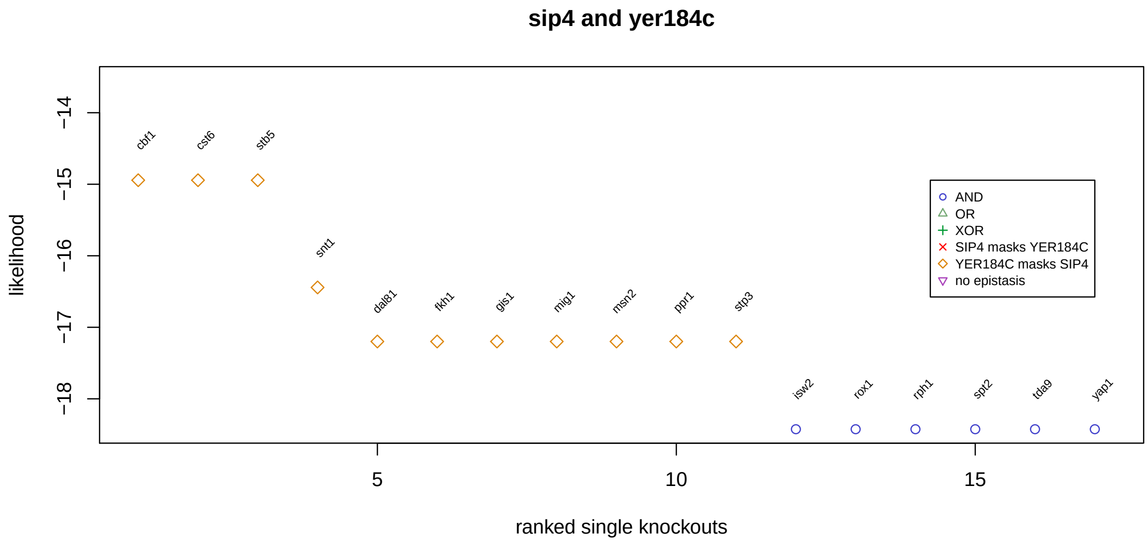


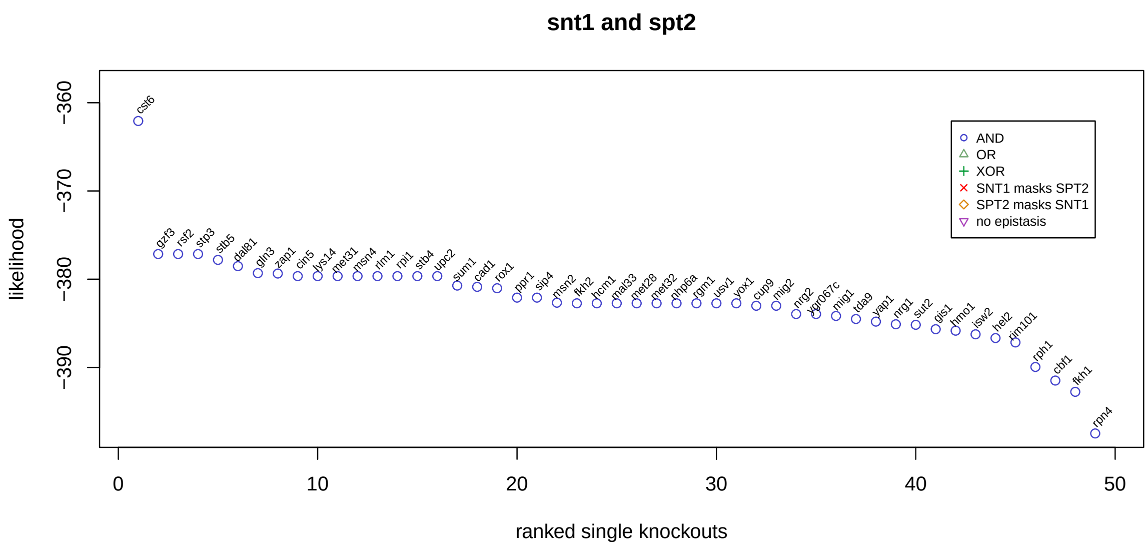

rlm1 and smp1
likelihood
−8
−9
−10
−11
−12
2
4
6
8
10
ranked single knockouts
cst6
stb5
yox1
cin5
dal81
hmo1
nrg1
rox1
rsf2
yap1
zap1
AND
OR
XOR
RLM1 masks SMP1
SMP1 masks RLM1
no epistasis

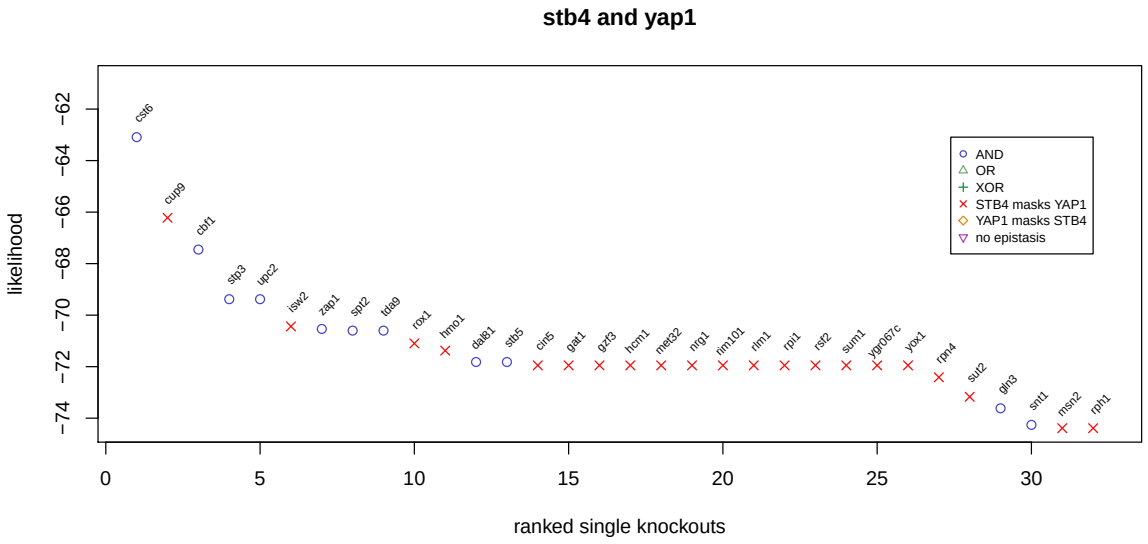

rpi1 and zap1
likelihood
−800
−850
−900
0
10
20
30
40
50
ranked single knockouts
cst6
stb5
cbf1
hmo1
isw2
rph1
met31
stp3
spt2
rgm1
snf1
nrg1
gzf3
glr3
mig1
rim101
fkh2
mal33
tos8
yrm1
dal81
asg1
nrg2
ppr1
rsf2
ygr067c
met32
gat1
hcm1
pho2
rlm1
upc2
usv1
tda9
yox1
yap1
tye7
mig2
fkh1
sum1
cup9
rox1
sut2
gis1
hel2
rpn4
cin5
msn2
AND
OR
XOR
RPI1 masks ZAP1
ZAP1 masks RPI1
no epistasis

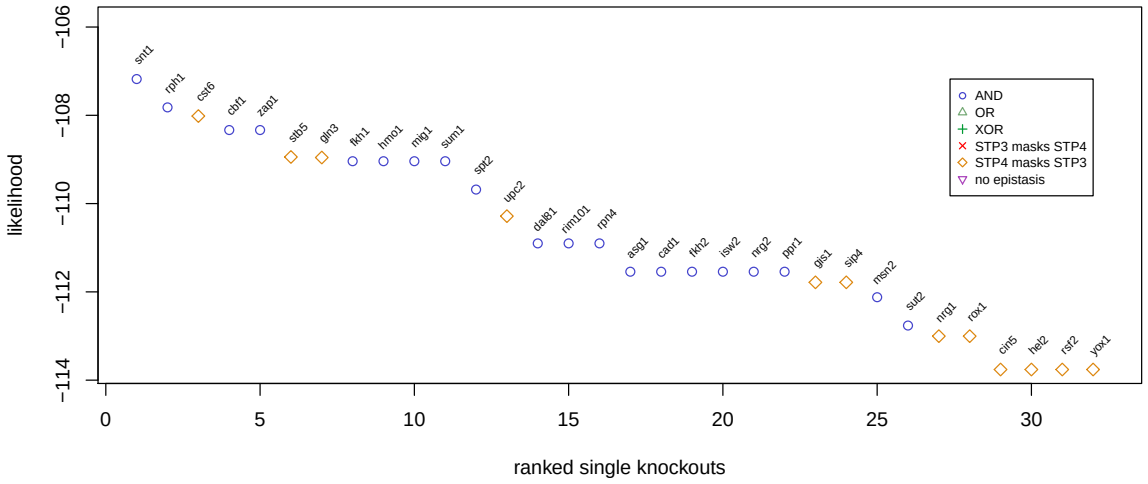

rpn4 and snt1
likelihood
−425
−430
−435
−440
0
10
20
30
40
ranked single knockouts
AND
OR
XOR
RPN4 masks SNT1
SNT1 masks RPN4
no epistasis
csi6
cad1
rsf2
mig1
asg1
fkh2
lys14
met32
tye7
upc2
yrm1
stp3
ppr1
tda9
cin5
gzf3
rtg3
met28
rgm1
gln3
mig2
ygr067c
stb5
cup9
skn7
msn2
hcm1
nrg2
isw2
yap1
cbf1
hmo1
sum1
rox1
sut2
nrg1
rim101
zap1
hel2
fkh1
dal81
rph1
spt2
gis1

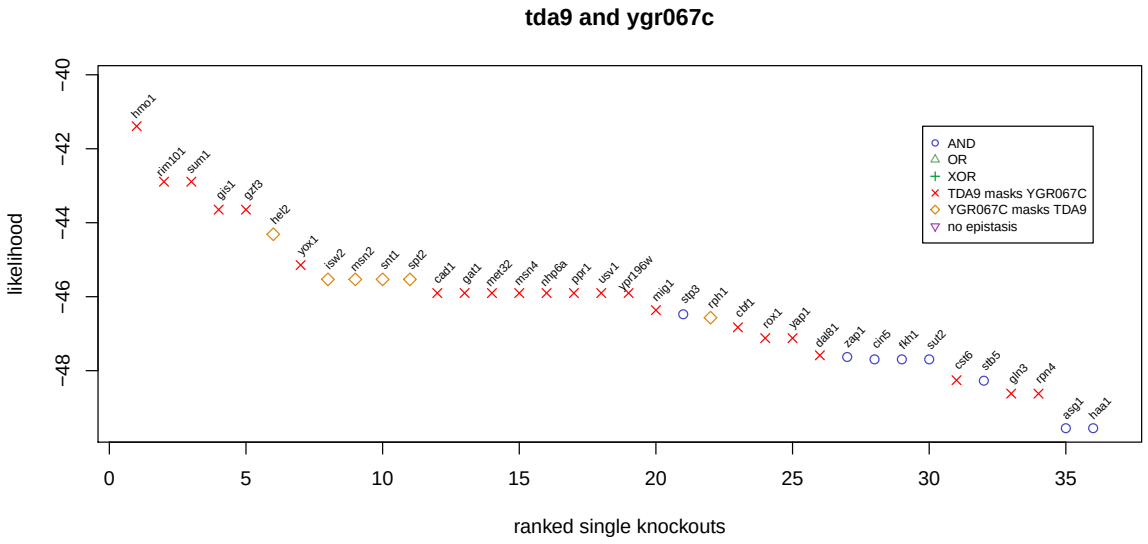

rsf2 and tda9
likelihood
−400
−450
−500
0
10
20
30
40
50
ranked single knockouts
AND
OR
XOR
RSF2 masks TDA9
TDA9 masks RSF2
no epistasis
csi6
stb5
cbf1
gln3
zap1
dal81
hmo1
msn2
snt1
rox1
rph1
hel2
gis1
rpn4
sum1
fkh1
gzf3
rim101
stp3
mig1
sut2
cad1
fkh2
skn7
upc2
ygr067c
yox1
isw2
spt2
yap1
nrg1
cup9
gat1
hcm1
mal33
met31
met32
mig2
msn4
nhp6a
pho2
rgm1
rpi1
ypr196w
yrm1
cin5
asg1
haa1
ttm1

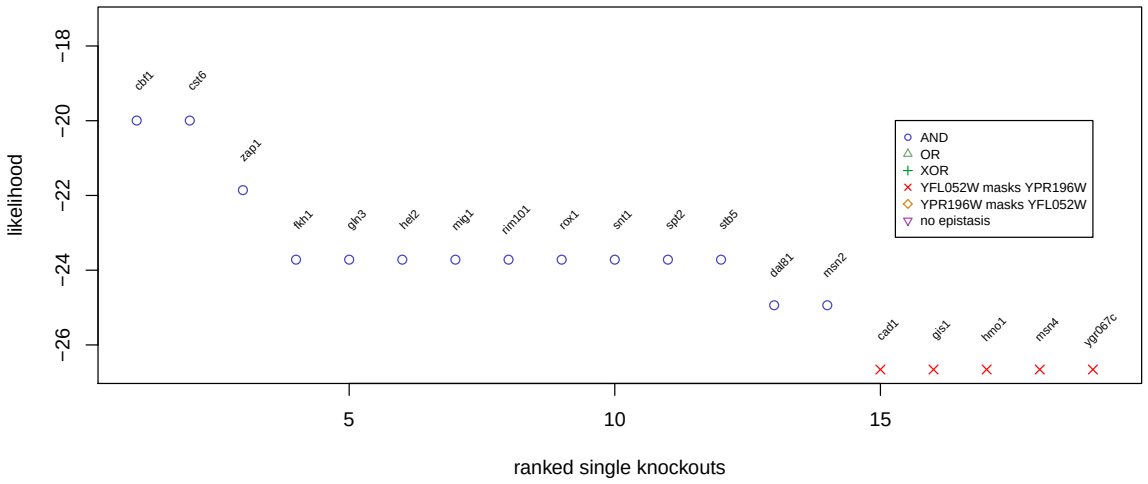

sip4 and yer184c
likelihood
−14
−15
−16
−17
−18
cbf1
cst6
stb5
snt1
dal81
fkh1
gis1
mig1
msn2
ppr1
stp3
isw2
rox1
rph1
spt2
tda9
yap1
AND
OR
XOR
SIP4 masks YER184C
YER184C masks SIP4
no epistasis
5
10
15
ranked single knockouts

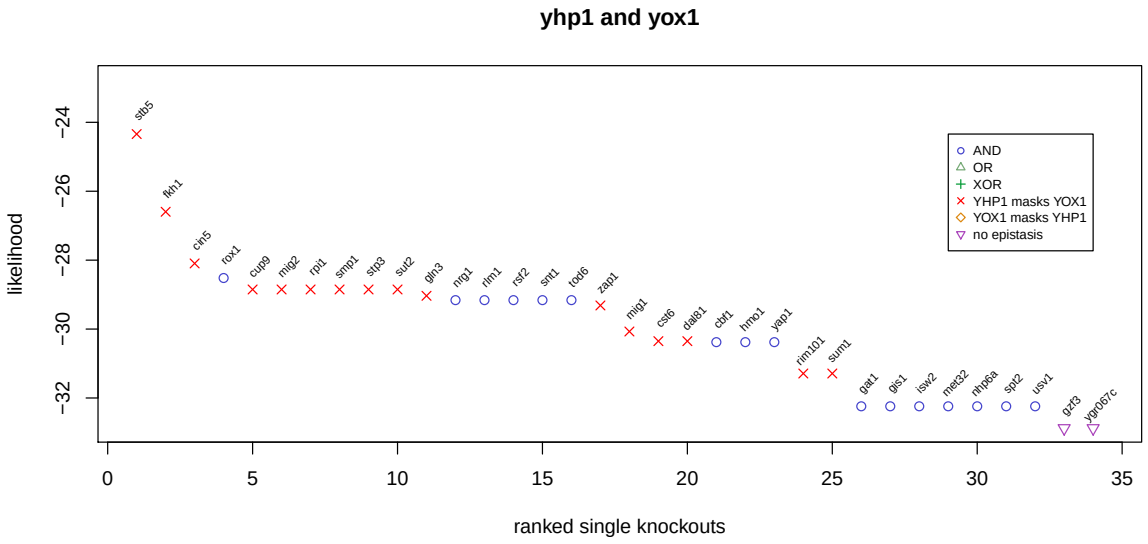

snt1 and spt2
likelihood
−360
−370
−380
−390
cst6
gzf3
rsf2
stp3
stb5
dal81
gln3
zap1
cin5
lys14
met31
msn4
rlm1
rpi1
stb4
upc2
sum1
cad1
rox1
ppr1
sip4
msn2
fkh2
hcm1
mal33
met28
met32
nhp6a
rgm1
usv1
yox1
cup9
mig2
nrg2
ygr067c
mig1
tda9
yap1
nrg1
sut2
gis1
hmo1
isw2
hel2
rim101
rph1
cbf1
fkh1
rpn4
AND
OR
XOR
SNT1 masks SPT2
SPT2 masks SNT1
no epistasis
0
10
20
30
40
50
ranked single knockouts

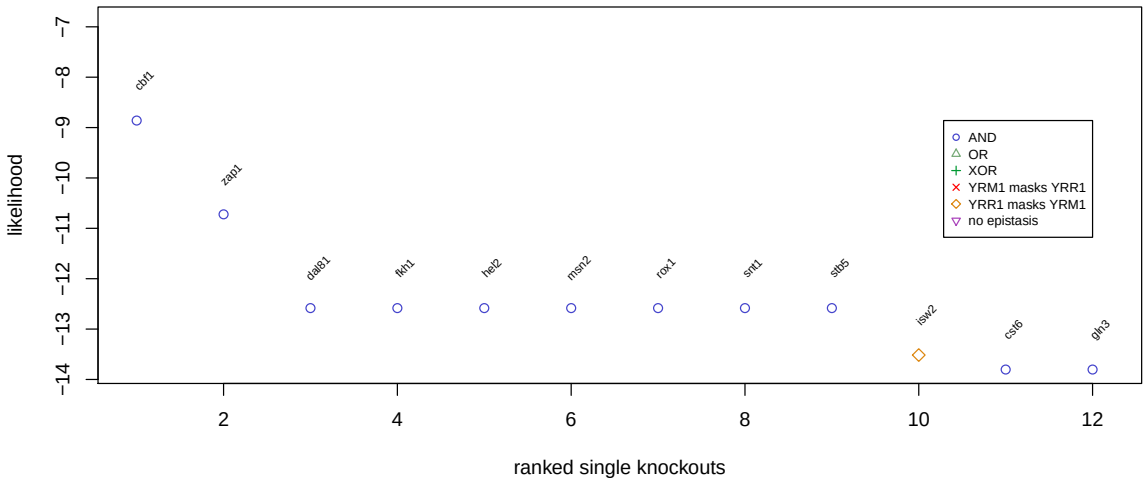

stb4 and yap1
likelihood
−62
−64
−66
−68
−70
−72
−74
0
5
10
15
20
25
30
ranked single knockouts
cst6
cup9
cbf1
stp3
upc2
isw2
zap1
spt2
tda9
rox1
hmo1
dal81
stb5
cin5
gat1
gzf3
hcm1
met32
nrg1
rim101
rlm1
rpi1
rsf2
sum1
ygr067c
yox1
rpn4
sut2
glr3
snt1
msn2
rph1
AND
OR
XOR
STB4 masks YAP1
YAP1 masks STB4
no epistasis


stp3 and stp4
likelihood
−106
−108
−110
−112
−114
0
5
10
15
20
25
30
ranked single knockouts
snt1
rph1
cst6
cbf1
zap1
stb5
glr3
fkh1
hmo1
mig1
sum1
spt2
upc2
dal81
rim101
rpn4
asg1
cad1
fkh2
isw2
nrg2
ppr1
gis1
sip4
msn2
sut2
nrg1
rox1
cin5
hal2
rsf2
yox1
AND
OR
XOR
STP3 masks STP4
STP4 masks STP3
no epistasis


tda9 and ygr067c
likelihood
−40
−42
−44
−46
−48
0
5
10
15
20
25
30
35
ranked single knockouts
hmo1
rim101
sum1
gis1
gzf3
hel2
yox1
isw2
msn2
snf1
spt2
cad1
gat1
met32
msn4
nhp6a
ppr1
usv1
ypr196w
mig1
stp3
rph1
cbf1
rox1
yap1
dal81
zap1
cin5
fkh1
sut2
cst6
stb5
gln3
rpn4
asg1
haa1
AND
OR
XOR
TDA9 masks YGR067C
YGR067C masks TDA9
no epistasis

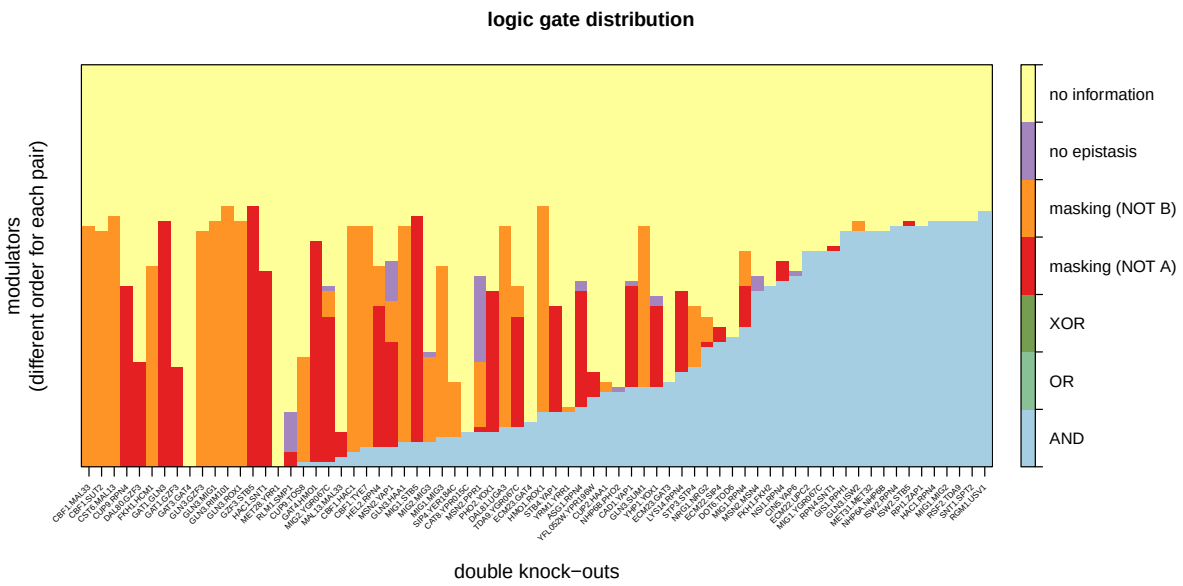

yfl052w and ypr196w
likelihood
−18
−20
−22
−24
−26
5
10
15
ranked single knockouts
cbf1
cst6
zap1
fkh1
gln3
hel2
mig1
rim101
rox1
snf1
spt2
stb5
dal81
msn2
cad1
gis1
hmo1
msn4
ygr067c
AND
OR
XOR
YFL052W masks YPR196W
YPR196W masks YFL052W
no epistasis






```
distmat <- sameith$logic

distmat[which(distmat %in% "AND")] <- 1
distmat[which(distmat %in% "OR")] <- 2
distmat[which(distmat %in% "XOR")] <- 3
distmat[which(distmat %in% "NOEPI")] <- 6
distmat[which(distmat %in% c("NOINFO", "NOINF"))] <- 7

for (i in 1:ncol(distmat)) {

    genes <- unlist(strsplit(colnames(distmat)[i], "\\."))

    distmat[which(distmat[, i] %in% paste(genes[1], " masks the effect of ",
                                          genes[2], sep = "")), i] <- 4
```

```
    distmat[which(distmat[, i] %in% paste(genes[2], " masks the effect of ",
                                          genes[1], sep = "")), i] <- 5

}

distmat <- apply(distmat, c(1,2), as.numeric)

for (i in 1:ncol(distmat)) {
    distmat[, i] <- rev(sort(distmat[, i]))
}

distmat <- distmat[-which(apply(distmat, 1, sum) == 0), ]

library(bnem)

y <- distmat

distmat <- distmat[, order(apply(distmat, 2, function(x) { return(sum(x == 1)) }))]

y[which(y == 5)] <- 4

rownames(distmat) <- NULL

labeltext <- c("", "no information\n\n\n", "no epistasis\n\n\n",
               "masking (NOT B)\n\n\n", "masking (NOT A)\n\n\n",
               "XOR\n\n\n", "OR\n\n\n", "AND\n\n\n")

heatmapOP(distmat, Colv = F, Rowv = F, main = "logic gate distribution", sub = "",
          col = "Paired", breaks = seq(0.5,7.5, length.out = 8), cexRow = 0,
          cexCol = 0.4, aspect = "fill",
          colorkey = list(space = "right", labels = rev(labeltext), width = 1,
                          at = seq(1.5,7.5, length.out = 8)),
          xlab = "double knock-outs",
          ylab = "modulators\n(different order for each pair)",
          xrot = 45, bordercol = "transparent")
```

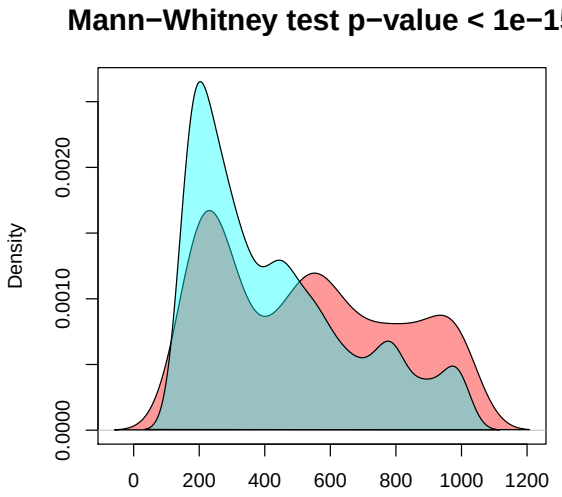


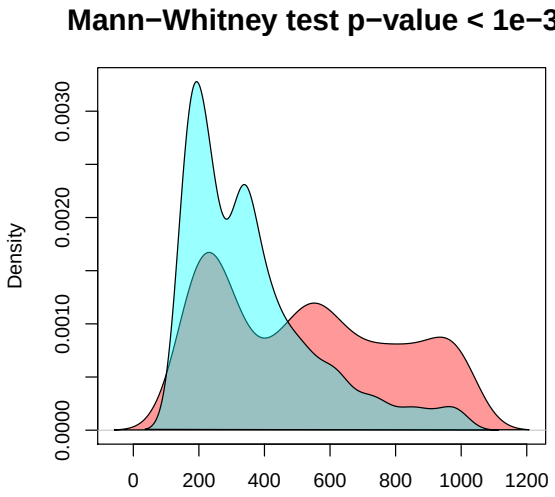
Now we plot the densities of the string-db interaction scores of our identified modulators and a random draw.

```
par(mfrow=c(1,2))

library(STRINGdb)

get_STRING_species(version="10", species_name=NULL)[26, ] # 4932

##    species_id            official_name             compact_name   kingdom type
## 26       4932 Saccharomyces cerevisiae Saccharomyces cerevisiae eukaryota core

string_db <- STRINGdb$new( version="10", species=4932, score_threshold=0,
                           input_directory="~/")

llmat <- wageningen$ll

logicmat <- wageningen$logic

string.scores <- list()

string.names <- character()

## for (i in 1:ncol(llmat)) {

##     if (sum(!(llmat[, i] %in% c(0,-Inf))) > 0) {
##         top30 <- llmat[, i]
##         top30[which(top30 == 0)] <- -Inf
##         top30 <- top30[which(!(llmat[, i] %in% c(0,-Inf)))]
##         top30 <- top30[order(top30,decreasing = T)[1:min(30, sum(!(llmat[, i]
##             %in% c(0,-Inf))))]]

##         doubles <- unlist(strsplit(colnames(llmat)[i], "\\."))

##         for (j in names(top30)) {
##             tmp <- string_db$get_interactions(string_db$mp(c(doubles[1], j)))
```

```
##             string.scores <- c(string.scores, tmp$combined_score)
##             string.names <- c(string.names, paste(sort(c(doubles[1], j)), collapse = "_"))
##             tmp <- string_db$get_interactions(string_db$mp(c(doubles[2], j)))
##             string.scores <- c(string.scores, tmp$combined_score)
##             string.names <- c(string.names, paste(sort(c(doubles[2], j)), collapse = "_"))
##         }

##     } else {
##         next()
##     }

## }

data(wageningen_string)

tmp <- string_db$get_interactions(string_db$mp(unique(unlist(strsplit(colnames(dataBinWag)
                                                                      , "\\."))))))

stsc <- unlist(string.scores)

denspval <- wilcox.test(stsc, unlist(tmp$combined_score), alternative = "greater")$p.value

for (i in 100:1) {

    if (denspval < 10^(-i)) {

        denspval <- paste("< ", 10^(-i), sep = "")

    }

}

plot(density(stsc), col = "#00000000",
     ylim = c(0, max(c(max(density(stsc)$y),max(density(unlist(tmp$combined_score))$y)))),
     main = paste("Mann-Whitney test p-value ", denspval, sep = ""), xlab = "",
     cex.main = 1.5)
polygon(density(stsc), col = "#ff000066")

lines(density(unlist(tmp$combined_score)), col = "#00000000")
polygon(density(unlist(tmp$combined_score)), col = "#00ffff66")

llmat <- sameith$ll

logicmat <- sameith$logic

string.scores <- list()

string.names <- character()

## for (i in 1:ncol(llmat)) {

##     if (sum(!(llmat[, i] %in% c(0,-Inf))) > 0) {
##         top30 <- llmat[, i]
```

```
##          top30[which(top30 == 0)] <- -Inf
##          top30 <- top30[which(!(llmat[, i] %in% c(0,-Inf)))]
##          top30 <- top30[order(top30, decreasing = T)[1:min(30, sum(!(llmat[, i]
##              %in% c(0,-Inf))))]]

##          doubles <- unlist(strsplit(colnames(llmat)[i], "\\."))

##          for (j in names(top30)) {
##              tmp <- string_db$get_interactions(string_db$mp(c(doubles[1], j)))
##              string.scores <- c(string.scores, tmp$combined_score)
##              string.names <- c(string.names, paste(sort(c(doubles[1], j)), collapse = "_"))
##              tmp <- string_db$get_interactions(string_db$mp(c(doubles[2], j)))
##              string.scores <- c(string.scores, tmp$combined_score)
##              string.names <- c(string.names, paste(sort(c(doubles[2], j)), collapse = "_"))
##          }

##      } else {
##          next()
##      }

## }

data(sameith_string)

tmp <- string_db$get_interactions(string_db$mp(unique(unlist(strsplit(colnames(dataBin)
                                                                      , "\\."))))))

stsc <- unlist(string.scores)

denspval <- wilcox.test(stsc, unlist(tmp$combined_score), alternative = "greater")$p.value

for (i in 100:1) {

    if (denspval < 10^(-i)) {

        denspval <- paste("< ", 10^(-i), sep = "")

    }

}

plot(density(stsc), col = "#00000000",
     ylim = c(0, max(c(max(density(stsc)$y), max(density(unlist(tmp$combined_score))$y)))),
     main = paste("Mann-Whitney test p-value ", denspval, sep = ""), xlab = "",
     cex.main = 1.5)
polygon(density(stsc), col = "#ff000066")

lines(density(unlist(tmp$combined_score)), col = "#00000000")
polygon(density(unlist(tmp$combined_score)), col = "#00ffff66")
```






```
sessionInfo()
```

```
## R version 3.3.1 (2016-06-21)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: OS X 10.11.5 (El Capitan)
##
## locale:
## [1] C/UTF-8/C/C/C/C
##
## attached base packages:
## [1] grid      parallel  stats     graphics  grDevices utils     datasets  methods
## [9] base
##
## other attached packages:
##  [1] STRINGdb_1.14.0     pcalg_2.4-3         minet_3.32.0        bnem_0.99.0
##  [5] latticeExtra_0.6-28 RColorBrewer_1.1-2  lattice_0.20-34     snowfall_1.84-6.1
##  [9] snow_0.4-2          matrixStats_0.51.0  nem_2.48.0          CellNOptR_1.20.0
## [13] XML_3.98-1.5        Rgraphviz_2.18.0    RCurl_1.95-4.8      bitops_1.0-6
## [17] ggplot2_2.2.0       hash_2.2.6          RBGL_1.50.0         graph_1.52.0
## [21] BiocGenerics_0.20.0 devtools_1.12.0     epiNEM_0.99.0       knitr_1.15.1
## [25] igraph_1.0.1        gtools_3.5.0        e1071_1.6-7         BoolNet_2.1.3
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.8          bdsmatrix_1.3-2      corpcor_1.6.8        png_0.1-7
##  [5] class_7.3-14         assertthat_0.1       digest_0.6.10        gmp_0.5-12
##  [9] chron_2.3-47         plyr_1.8.4           stats4_3.3.1         RSQLite_1.0.0
## [13] evaluate_0.10        sqldf_0.4-10         highr_0.6            BiocInstaller_1.24.0
## [17] gplots_3.0.1         lazyeval_0.2.0       gdata_2.17.0         gsubfn_0.6-6
## [21] proto_1.0.0          statmod_1.4.26       stringr_1.1.0        munsell_0.4.3
## [25] tibble_1.2           withr_1.0.2          DBI_0.5-1            ggm_2.3
## [29] gtable_0.2.0         magrittr_1.5         scales_0.4.1         KernSmooth_2.23-15
## [33] stringi_1.1.2        limma_3.30.4         robustbase_0.92-6    boot_1.3-18
## [37] fastICA_1.2-0        tools_3.3.1          DEoptimR_1.0-6       sfsmisc_1.1-0
## [41] abind_1.4-5          plotrix_3.6-3        clue_0.3-51          colorspace_1.3-0
## [45] cluster_2.0.5        caTools_1.17.1       memoise_1.0.0
```

References:

Martin Pirkl, Madeline Diekmann, Marlies van der Wees, Niko Beerenwinkel, Holger Fröhlich, Florian Markowetz. Inferring Modulators of Genetic Interactions with Epistatic Nested Effects Models. submitted.