Hotspots and Causal Inference For Yeast Data

Elias Chaibub Neto*and Brian S Yandell†

October 22, 2012

Here we reproduce the analysis of the budding yeast genetical genomics data-set presented in Chaibub Neto et al. (2012). The data represents a cross of a standard yeast laboratory strain, and a wild isolate from a California vineyard (Brem and Kruglyak 2005). It consists of expression measurements on 5,740 transcripts measured on 112 segregant strains with dense genotype data on 2,956 markers. Processing of the expression measurements raw data was done as described in Brem and Kruglyak (2005), with an additional step of converting the processed measurements to normal quantiles by the transformation $\Phi^{-1}[(r_i - 0.5)/112]$, where Φ is the standard normal cumulative density function, and the r_i are the ranks.

The data were provided by Rachel Brem and further edited by Jun Zhu and Bin Zhang (formerly of Sage Bionetworks). Elias Chaibub Neto and Brian Yandell have organized the data and analysis into this R statistical package.

We first load the yeast cross object (`yeast.orf`), and compute the conditional genotype probabilities using Haldane's map function, genotype error rate of 0.0001, and setting the maximum distance between positions at which genotype probabilities were calculated to 2cM.

```
> library(qtlhot)
> library(qtlyeast)
> ## data(yeast.orf)
> yeast.orf <- calc.genoprob(yeast.orf, step = 2)
```

The following command does an genome scan for QTL using R/qtl for all the traits using Haley-Knott regression (Haley and Knott 1992).

```
> scan.orf <- scanone(yeast.orf, pheno.col = seq(nphe(yeast.orf)), method = "hk")
```

To save space, we work with only the genome regions that are above the single trait LOD threshold and within 1.5 LOD of the maximum per chromosome. We do this after we determine the permutation LOD threshold below.

*Department of Computational Biology, Sage Bionetworks, Seattle WA
†Department of Statistics, University of Wisconsin-Madison, Madison WI

1 Hotspot Inference

Plan of action: 1. Find Churchill-Doerge 5% LOD threshold 2. Determine hotspot counts relative to LOD threshold (Jansen method) 3. conduct permutation test (using CHTC) 4. report Jansen and Chaibub-Neto results 5. identify hotspots.

1.1 Churchill-Doerge LOD threshold

Since we are using normal scores on the traits, we need only conduct permutation threshold calculation with a normal response. Here we create one trait and then do 1000 permutations. We have saved this as `perm.orf`.

```
> cross <- yeast.orf
> cross$pheno <- data.frame(norm = rnorm(nind(cross)))

> set.seed(12345)
> perm.orf <- scanone(cross, method = "hk", n.perm = 1000)

> ## data(perm.orf)
> summary(perm.orf)

LOD thresholds (1000 permutations)
     lod
5%  3.48
10% 3.08

> lod.thr <- c(summary(perm.orf, alpha = 0.05))
```

Now we save only the high lods of the `scan.orf` object to save space.

```
> highlod.orf <- highlod(scan.orf, lod.thr = lod.thr, drop.lod = 1.5)
```

This takes considerable time, so we have actually saved the completed scans as object `scan.orf`. However, the `scan.orf` object is 203Mb, so we don't keep it in the package. Instead we have saved `highlod.orf`.

```
> ## data(highlod.orf)
```

1.2 Hotspots for Yeast Data above LOD threshold

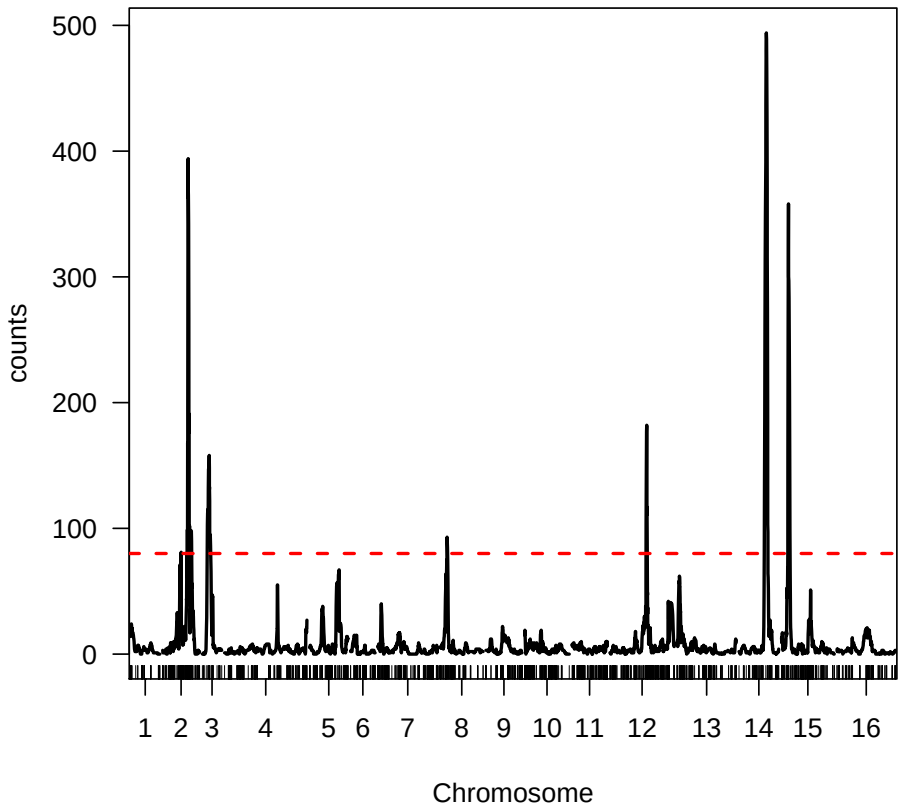
Now we show the hotspots. We can get summary and plot from `highlod.orf`, but it is sometimes more helpful to first turn it into a `hotsize` object. We use an arbitrary threshold of 80 traits per hotspot, which is passed along to `scanone` summary and plot methods, to get some handle on hotspots.

```
> hotsize.orf <- hotsize(highlod.orf, lod.thr = lod.thr)
> summary(hotsize.orf, threshold = 80)
```

```
hotsize elements:  chr pos max.N
LOD threshold: 3.475609

                       chr   pos max.N
YBR154C_chr2@548401      2 224.1   394
c3.loc60                 3  60.0   158
NHR001C.3_chr8@111683    8  58.5    93
YLR258W_chr12@662627    12 323.6   182
c14.loc240              14 240.0   494
gOL02.1_chr15@174364    15  61.1   358

> plot(hotsize.orf)
> abline(h = 80, lwd = 2, col = "red", lty = 2)
```



This shows hotspots, but there is no way yet to assess their significance. To do that, we must run some further permutations across all the traits together, preserving their correlation structure. This takes even more time, so we will do it offline and show the results.

2 Causal Inference

Current efforts in systems genetics have focused on the development of statistical approaches that aim to disentangle causal relationships among molecular phenotypes in segregating populations. Model selection criterions, such as the AIC and BIC, have been widely used for this purpose, in spite of being unable to quantify the uncertainty associated with the model selection call. We illustrate analysis of the Brem and Kruglyak (2005 PNAS) data using software implemented in R/qtlhot.

In order to evaluate the precision of the causal predictions made by the methods we used validated causal relationships extracted from a data-base of 247 knock-out experiments in yeast (Hughes et al. 2000, Zhu et al. 2008). In each of these experiments, one gene was knocked-out, and the expression levels of the remainder genes in control and knocked-out strains were interrogated for differential expression. The set of differentially expressed genes form the knock-out signature (ko-signature) of the knocked-out gene (ko-gene), and show direct evidence of a causal effect of the ko-gene on the ko-signature genes.

Next, we load a yeast annotation data.frame, `yeast.annot`, that provides the orf, gene symbol, and chromosome location (in both Mb and cM) of each one of the 5,740 transcripts. (This information will be needed to determine which ko-genes show significant QTLs.) Next, we load a yeast annotation (derived from the YEAST R package) data.frame, `yeast.annot`, that provides ORFs, gene names, chromosome, and position in Mb and cM.

```
> ## data(yeast.annot)
> head(yeast.annot)

         orf  gene chr   Mb.pos   cM.pos
3952 YAL001C  TFC3   1 0.151168 102.4066
3951 YAL002W  VPS8   1 0.143709 101.3745
3950 YAL003W  EFB1   1 0.142176 101.1623
1330 YAL005C  SSA1   1 0.141433 101.0595
3934 YAL007C  ERP2   1 0.138347 100.1245
3933 YAL008W FUN14   1 0.136916 100.1245
```

Next, we load the list of ko-signatures derived from the knock-out experiments in Hughes et al. (2000) and Zhu et al. (2008). We show below the first knock-out signature.

```
> ## data(ko.list)
> length(ko.list)

[1] 247

> ko.list[[1]]

 [1] "YAR073W"   "YBL013W"   "YBL032W"   "YBL042C"   "YBL054W"   "YBL064C"
 [7] "YBR013C"   "YBR054W"   "YBR072W"   "YBR126C"   "YBR155W"   "YBR186W"
[13] "YCL030C"   "YDL038C"   "YDL234C"   "YDL244W"   "YDR001C"   "YDR018C"
```

```
[19] "YDR055W"   "YDR077W"   "YDR085C"   "YDR399W"   "YDR518W"   "YDR533C"
[25] "YDR534C"   "YER055C"   "YER062C"   "YFL014W"   "YFL030W"   "YFL058W"
[31] "YGL156W"   "YGL162W"   "YGL187C"   "YGL234W"   "YGR032W"   "YGR043C"
[37] "YGR138C"   "YGR161C"   "YGR171C"   "YGR213C"   "YGR250C"   "YHL040C"
[43] "YHR087W"   "YHR096C"   "YHR104W"   "YHR216W"   "YIL125W"   "YJL034W"
[49] "YJL054W"   "YJL116C"   "YJR151C"   "YKL029C"   "YKL090W"   "YKL097W.A"
[55] "YKL163W"   "YKL165C"   "YKR061W"   "YLL019C"   "YLL060C"   "YLR120C"
[61] "YLR121C"   "YLR142W"   "YLR178C"   "YLR194C"   "YLR350W"   "YLR359W"
[67] "YML130C"   "YML131W"   "YMR040W"   "YMR090W"   "YMR173W"   "YMR181C"
[73] "YMR300C"   "YNL112W"   "YNL134C"   "YNL160W"   "YNL220W"   "YOL151W"
[79] "YOL031C"   "YOR173W"   "YOR289W"   "YOR338W"   "YOR382W"   "YPL088W"
[85] "YPL277C"   "YPR156C"   "YAR075W"   "YDR243C"   "YFR024C.A" "YOL053C.A"
```

Next, we determine which of the 247 ko-genes also showed a significant QTL in our data set, according to a permutation test (Churchill and Doerge 1994) aiming to control GWER < 0.05. For each one of the ko-genes with a significant QTL, that is, with LOD score above `lod.thr` $= 3.48$, the function `GetCandReg` returns the ko-gene's chromosome (`phys.chr`) and physical position in cM (`phys.pos`), as well as, the LOD score (`peak.lod`) at the peak position (`peak.pos`), and the chromosome where the peak is located (`peak.chr`). In total, we observed 135 ko-genes with significant QTLs. These ko-genes are our candidate regulators. We show below the information on the first 10 candidate regulators. Note that some ko-genes map to the same chromosome where they are physically located, while other map to different chromosomes.

```
> cand.reg <- GetCandReg(highlod.orf, yeast.annot, names(ko.list))
> dim(cand.reg)

[1] 135   6

> head(cand.reg)

      gene phys.chr phys.pos peak.chr    peak.pos peak.lod
2   YMR282C       13 473.2316       14 236.0138450 3.692560
3   YER017C        5 152.3216       14 238.0138450 6.597231
7   YER069W        5 211.7280        3  54.0140660 3.975861
9   YOR058C       15 188.5460        8   0.9067482 3.569372
10  YGL017W        7 227.0394        7 221.6439074 5.894020
14  YMR055C       13 235.2625       13 246.0276440 5.578000
```

Genes that map to positions close to their physical locations are said to map in *cis* (local-linkages). Genes that map to positions away from their physical locations are said to map in *trans* (distal-linkages). There is no unambiguous way the determine how close a gene needs to map to its physical location in order to be classified as cis. Our choice is to classify a gene as cis if the 1.5-LOD support interval (Manichaikul et al. 2006) around the LOD peak contains the gene's physical location, and if the LOD score at its physical location is higher the the LOD threshold. The function `GetCisCandReg` determines which of the candidate regulators map in cis.

```
> cis.cand.reg <- GetCisCandReg(highlod.orf, cand.reg)
> dim(cis.cand.reg)

[1] 28  7

> head(cis.cand.reg)

      gene phys.chr phys.pos peak.pos peak.lod peak.pos.lower peak.pos.upper
10 YGL017W        7 227.0394 221.6439 5.894020       210.0037       230.7461
14 YMR055C       13 235.2625 246.0276 5.578000       144.0276       260.0276
16 YMR275C       13 467.3183 460.0276 5.508846       420.0276       472.0276
48 YLR342W       12 402.5087 402.5087 8.666742       400.0196       410.0196
61 YNL021W       14 278.5199 242.0138 4.359721       222.0138       280.0138
63 YOR038C       15 179.1709 174.0012 5.060326       156.0012       184.0012
```

We see that only 28, out of the 135 candidate regulators, show cis-linkages. (The additional columns `peak.pos.lower` and `peak.pos.upper` show, respectively, the lower and upper bounds of the 1.5-LOD support interval around `peak.pos`.)

For each one of the 135 candidate ko-genes, we determined which other genes also co-mapped to the same QTL of the ko-gene. The co-mapping genes represent the putative targets of a ko-gene. The function `GetCoMappingTraits` returns a list with the putative targets of each ko-gene. A gene is included in the putative target list of a ko-gene when its LOD peak is greater than `lod.thr` and the 1.5 LOD support interval around the peak contains the location of the ko-gene's QTL.

```
> comap.targets <- GetCoMappingTraits(highlod.orf, cand.reg)
> summary(sapply(comap.targets, length))

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    1.0    63.5   188.0   236.9   479.0   569.0

> comap.targets[[7]]

[1] "YDL013W" "YGL254W" "YML069W" "YMR247C"

> length(unlist(comap.targets))

[1] 31975
```

The number of targets vary from ko-gene to ko-gene (from 1 to 569). We illustrate with the putative targets of one ko-gene (YMR275C) with 4 putative targets. In total, the 135 candidate regulators have 31975 targets.

Next, we use the function `FitAllTests` to fit the causality tests of each candidate regulator ko-gene (`pheno1`) to its putative targets (`pheno2`). We use the candidate regulator's QTL (`Q.chr` and `Q.pos`) as a causal anchor. This function fits: the AIC and BIC model selection criterions (Schadt et al. 2005); the AIC- and BIC-based versions of the joint, parametric and

non-parametric CMST tests (Chaibub Neto et al. 2012); and the CIT test (Millstein et al. 2009). We do not run it here because this step can take a few hours, as we perform a total of 31975 tests for each of the 9 approaches. The function `JoinTestOutputs` joins together the outputs of the 135 separate fits of the `FitAllTests` function.

```
> set.seed(123456789) # we fix a seed because cit uses bootstrap
> for (k in 1 : 135) {
>   cat("trait=", k, "\n")
>   out <- FitAllTests(cross = yeast.orf,
+                      pheno1 = cand.reg[k, 1],
+                      pheno2 = comap.targets[[k]],
+                      Q.chr = cand.reg[k, 4],
+                      Q.pos = cand.reg[k, 5])
>   save(out, file=paste("output_ko_validation", cand.reg[k, 1], "RData",
+        sep = "."), compress = TRUE)
> }
> ko.tests <- JoinTestOutputs(comap.targets)
```

We are now using the Benjamini-Hochberg adjustment for the non-parametric CMST tests. Therefore to get the adjusted values, we do the following:

```
> ## data(ko.tests)
> adj.ko.tests <- p.adjust.np(ko.tests)
```

After loading the joined results we use the function `PrecTpFpMatrix` to summarize the performance of the different methods in terms of "biologically validated" true positives, false positives and precision, of the inferred causal relations. Since we already have the results of the knock-out experiments (recall that `ko.list` holds the ko-signatures of the ko-genes), we define a true positive as a statistically significant causal relation between a ko-gene and a putative target gene, when the putative target gene belongs to the ko-signature of the ko-gene. Similarly, we define a false positive as a statistically significant causal relation between a ko-gene and a putative target gene when the target gene doesn't belong to the ko-signature. (For the AIC and BIC methods, that do not provide a p-value measuring the significance of the causal call, we simply use the detected causal relations in the computation of true and false positives). The "validated precision", is computed as the ratio of true positives by the sum of true and false positives. The `PrecTpFpMatrix` computes these measures to both all ko-genes, and to cis ko-genes only. The argument `alpha` sets the significant levels at each the summaries are computed. Since this takes awhile, we have also stored `roc.aux` as a data object in the package.

```
> roc.aux <- PrecTpFpMatrix(alpha = seq(0.01, 0.10, by = 0.01),
+                           nms = cand.reg[, 1],
+                           val.targets = ko.list,
+                           all.orfs = highlod.orf$names,
+                           tests = adj.ko.tests,
+                           cis.index = cis.cand.reg[[2]])
```

```
> ## data(roc.aux)
```

Before we show plots, here are some preliminary plot settings in a simple plot routine that will be used repeatedly for figures.

```
> plots <- function(roc.aux, elements) {
+   par(mfrow = c(1,3))
+   par(mar=c(5, 4.1, 4, 2) + 0.1)
+   myplot(roc.aux[[elements[1]]], "Number of true positives", "(a)")
+   myplot(roc.aux[[elements[2]]], "Number of false positives", "(b)")
+   myplot(roc.aux[[elements[3]]], "Precision", "(c)")
+ }
> myplot <- function(sum.type, sum.label, main = "") {
+   ymax <- max(sum.type)
+   my.pch <- c(1, 21, 24, 23, 25, 2, 5, 6, 8)
+   xaxis <- seq(0.01, 0.10, by=0.01)
+   yaxis <- seq(0, ymax,length.out = length(xaxis))
+   plot(xaxis, yaxis, type = "n", ylab = sum.label, cex = 1.5,
+        xlab = "target level", cex.axis = 1.5,
+        cex.lab = 1.7, main = main, cex.main = 2)
+   for (k in 1 : 9) {
+     lines(xaxis, sum.type[k,], type="b", lwd=2, pch=my.pch[k], cex=1.5,
+           col = "black", bg = "black")
+   }
+ }
```

Below we reproduce Figure 5 of Chaibub Neto et al. (2012). This figure presents the number of inferred true positives, number of inferred false positives and the prediction precision across varying significance levels for each one of the methods. The results were computed using all 135 ko-gene/putative target lists.

Next, we reproduce Figure 6 of Chaibub Neto et al. (2012). This figure was generated using the results of the 27 cis ko-gene/putative targets lists.

3 References

1. Brem R., L. Kruglyak, 2005 The landscape of genetic complexity across 5,700 gene expression trait in yeast. PNAS **102:** 1572-1577.

2. Broman K., H. Wu, S. Sen, G. A. Churchill, 2003 R/qtl: QTL mapping in experimental crosses. Bioinformatics **19**: 889-890.

3. Chaibub Neto et al. (2012) Causal model selection hypothesis tests in systems genetics. Genetics (under review)

4. Churchill G. A., R. W. Doerge, 1994 Empirical threshold values for quantitative trait mapping. Genetics **138**: 963-971.

5. Haley C., S. Knott, 1992 A simple regression method for mapping quantitative trait loci in line crosses using flanking markers. Heredity **69**: 315-324.

6. Hughes T. R., M. J. Marton, A. R. Jones, C. J. Roberts, R. Stoughton, et al, 2000 Functional discovery via a compendium of expression profiles. Cell **102:** 109-116.

7. Manichaikul A., J. Dupuis, S. Sen, and K. W. Broman, 2006 Poor performance of bootstrap confidence intervals for the location of a quantitative trait locus. Genetics **174:** 481-489.

8. Schadt E. E., J. Lamb, X. Yang, J. Zhu, S. Edwards, et al., 2005 An integrative genomics approach to infer causal associations between gene expression and disease. Nature Genetics **37**: 710-717.

9. Zhu J., B. Zhang, E. N. Smith, B. Drees, R. B. Brem, L. Kruglyak, R. E. Bumgarner, E. E. Schadt, 2008 Integrating large-scale functional genomic data to dissect the complexity of yeast regulatory networks. Nature Genetics **40**: 854-861.

```
> plots(roc.aux, c("Tp1","Fp1","Prec1"))
```

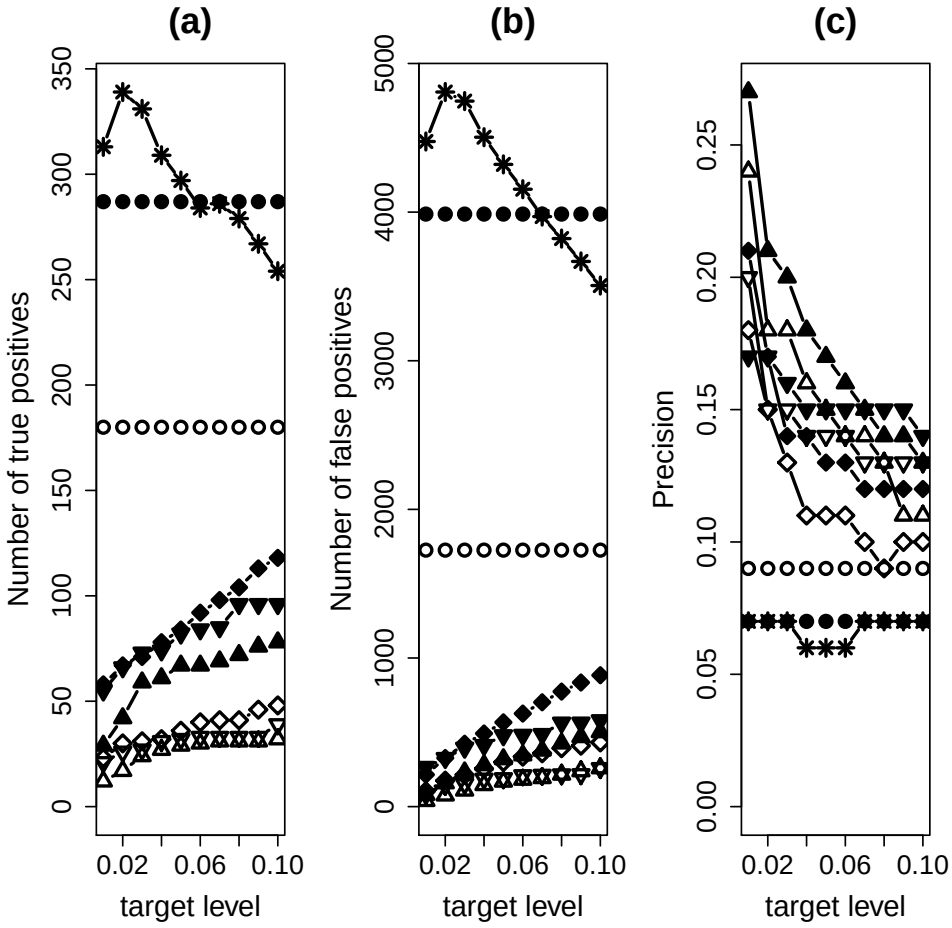


Figure 1: Reproduction of Figure 5 on Chaibub Neto et al. 2012. Target significance level by overall (a) number of true positives, (b) number of false positives and (c) precision across all 135 ko-gene/putative target lists. Asterisk represents the CIT. Empty and filled symbols represent, respectively, AIC- and BIC-based methods. Diamonds: parametric CMST. Point-down triangles: non-parametric CMST. Point-up triangles: joint-parametric CMST. Circles: AIC and BIC.

```
> plots(roc.aux, c("Tp2","Fp2","Prec2"))
```

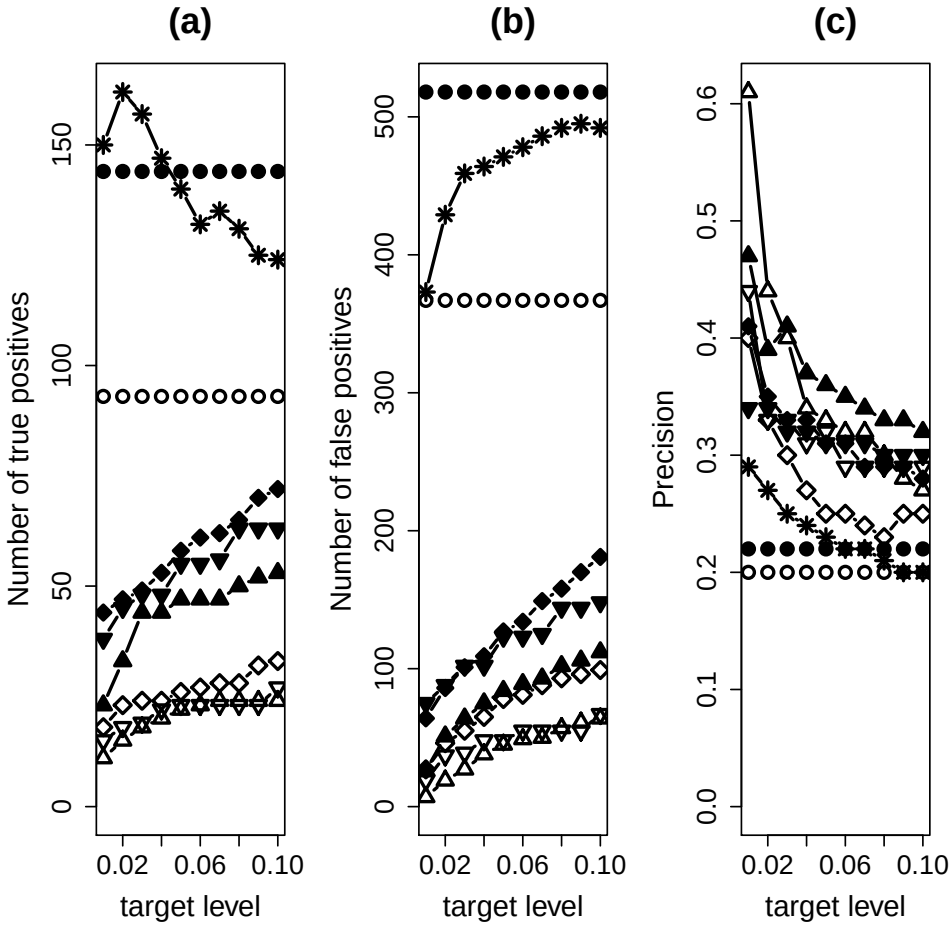


Figure 2: Reproduction of Figure 6 on Chaibub Neto et al. 2012. Target significance level by overall (a) number of true positives, (b) number of false positives and (c) precision restricted to 27 cis ko-gene/putative target lists. Asterisk represents the CIT. Empty and filled symbols represent, respectively, AIC- and BIC-based methods. Diamonds: parametric CMST. Point-down triangles: non-parametric CMST. Point-up triangles: joint-parametric CMST. Circles: AIC and BIC.