Practical session: Reconstruction of regulatory network

Jean-Philippe Vert

1 Data preparation

1.1 Download E. coli expression and regulation data

Download the data of Faith et al. from the course webpage. For example, from a command window, type:

```
wget http://cbio.ensmp.fr/~jvert/svn/tutorials/practical/reginference/data.tar.gz
tar xvzf data.tar.gz
```

This will create in your folder three files containing respectively the list of E. coli gene names, a matrix of expression data, and the list of know regulations.

1.2 Read the data in R and reduce the network

```
# Load expression data
exp <- as.matrix(read.table('expression.txt'))

# Load regulation data
reg <- read.table('regulation.txt')

# Load gene names
gname <- as.matrix(read.table('gene_names.txt'))

# List of tf and targets
tf <- sort(unique(reg[,1]))
targets <- sort(unique(reg[,2]))

## For this practical session, we reduce the number of targets to avoid long
## computation time. In general this is not required.

# Take only 200 targets randomly among all targets
ntargets <- 200
targets <- targets[sample(length(targets),200)]

# Restrict regulations to those targets
reg <- reg[!is.na(match(reg[,2],targets)),]
nreg <- dim(reg)[1]

# Restrict tf to those targeting those targets
tf <- sort(unique(reg[,1]))
ntf <- length(tf)
```

1.3 Visualization of the network in Cytoscape

From R, we prepare two data files that can be read by Cytoscape to visualize the graph: a file for the edges, and a file for the gene names

```
# Prepare network files for visualization in Cytoscape ##
write.table(reg,file="myedges.txt",row.names=FALSE,col.names=FALSE)
write.table(cbind(netgenes,gname[netgenes]),"mygenenames.txt",quote=FALSE,...
row.names=FALSE,col.names=FALSE)
```

Then open Cytoscape. To load the edges, go to the `File->Import->Network from Table` menu. There, select your file containing the edges, declare column 1 as ”source interaction” and column 2 as ”target interaction”, and import. For the gene names, go to the `File->Import->Attributes from Table` menu, select your file, check that the text input options recognize the space as column delimiter, and import.

You should now see you network in a window. You can play with it, zoom, choose a different layout as well as properties of the edges and vertices.

Question 1 Visualize the network. Which genes seem to play a central role in E. coli regulation?

2 De novo network reconstruction

We first consider the problem: given the matrix expression, can we reconstruct automatically the gene regulatory network?

2.1 Assessing the performance of a network reconstruction method

We will test different methods to associate a score to all TFxTarget pairs, from the most likely to have a regulation (large score) to the less likely. We assess their performance by checking where in the ranked list they put the known regulations. We quantify this with the ROC and the precision/recall curves. It will therefore be useful to prepare a function to automatically assess the performance of a reconstruction method from its scoring matrix.

```
## Assessing the performance

library(ROCR)

# We make the ntf*ntargets binary matrix of known regulations
truereg <- matrix(0,ntf,ntargets)
for (i in seq(nreg)) {
truereg[match(reg[i,1],tf),match(reg[i,2],targets)]=1
}

# Fonction to assess the performance of a similarity matrix
allind <- ntf * ntargets
assessperf <- function(simscore) {
# simscore should be a ntf*ntargets matrix of score
pred <- prediction(simscore[seq(allind)],truereg[seq(allind)])
roc <- performance(pred, measure = "tpr", x.measure = "fpr")
pre <- performance(pred, measure = "prec", x.measure = "rec")
return(list(roc=roc,pre=pre))
}
```

Question 2 Create a random prediction (eg, a matrix of scores randomly generated according to a normal distribution). How look its ROC and precision-recall curves?

2.2 Reconstruction based on similarity between TF and targets

In this section we consider methods that score a candidate TF-target pair by a measure of "similarity" between the expression profiles of the TF and the genes. We test three measures of similarity:

- Pearson correlation coefficient
- Spearman correlation coefficient
- Mutual information

These measures are implemented in the package `bioDist`, through the functions `cor.dist`, `spearman.dist` and `MIdist`.

Question 3 Install the `bioDist` package and check the help in `R` to understand how these functions work. How do you compute the different measures of similarity between two vectors?

Question 4 Compute the three measures of similarity between all TF and all targets. Compute their performance (ROC and precision-recall curves) in predicting the gene regulatory network. Which measure is the best?

2.3 Reconstruction based on regression

In this section we consider methods that express the problem as a regression problem: which TF are sufficient to predict the expression level of a given target? More precisely, for a given target, we try to predict its expression levels across the experiments as a function of the expression levels of just a few transcription factors. Ideally, only a few TF are needed, namely, the one that regulate it. In practice, we assign a score to each TF to quantify its strength to predict the expression of the target.

To apply this strategy, we therefore need a function of the form `scorefeatures(x,y)`, for any $n \times p$ matrix `x` and $n \times 1$ vector `y`, which returns a $p \times 1$ vector of scores for each feature, quantifying how important the features are to predict `y` from `x`. We provide on the webpage two such functions using either random forests, or Lasso regression with stability selection.

Question 5 For each of the two measures of feature importance, score each TF on each target (except self-regulation), and assess the quality of the score to predict interaction. Which one is the best?

2.4 Comparions

Question 6 Compare all de novo methods in terms of ROC curves and precision/recall curves.

Question 7 Visualize the predicted network in Cytoscape. How similar is it from the true network?

3 Supervised methods

In this section we test a supervised inference method. The setting is different: we assume that we already know some edges, and our goal is to predict missing edges with the help of gene expression. We folllow the methodology of SIRENE, which uses a SVM to discriminate between known targets and all other genes for each TF. The score of a candidate target is then the score of the SVM.

We first randomly split all regulations in two: those assumed to be known (train), and those to be discovered (test).

```
trainregindex <- sample(nreg,nreg/2)
testregindex <- seq(nreg)[-trainregindex]
```

We then make a binary matrix containing a positive label for the training regulations. This will be used by the supervised methods.

```
# We make the ntf*ntargets binary matrix of known regulations
trainreg <- matrix(-1,ntf,ntargets)
for (i in seq(length(trainregindex))) {
trainreg[match(reg[trainregindex[i],1],tf),match(reg[trainregindex[i],2],targets)]=1
}
```

We are now ready to compute a score for each TF/target pair not in the training set, by learning a SVM to discriminate the known targets from other targets for each TF in turn.

Question 8 Implement this function. Compute the ROC and precision-recall curves of this procedure.