Finding Metabolic Pathways in Decision Forests

André Flöter[1], Joachim Selbig[2], and Torsten Schaub[1]

[1] Institut für Informatik, Universität Potsdam, August-Bebel-Str. 89/Hs.4, D-14482 Potsdam, Germany
[2] Max-Planck-Institut für molekulare Pflanzenphysiologie, D-14424 Potsdam, Germany

Abstract. Data of metabolite concentrations in tissue samples is a powerful source of information about metabolic activity in organisms. Several methods have already been reported that allow for inferences about genetic regulatory networks using expression profiling data. Here we adapted these techniques for metabolic concentration data. While somewhat accurate in predicting certain properties these methods encounter problems when dealing with networks containing a high number of vertices (> 50). To circumvent these difficulties, larger data sets are usually reduced à priori by means of preprocessing techniques. Our approach allows to make network inferences using ensembles of decision trees that can handle almost any amount of vertices, and thus to avoid time consuming preprocessing steps. The technique works on a bootstrap principle and heuristically searches for partially correlated relations between all objects. We applied this approach to synthetically generated data as well as on data taken from real experiments.

1 Introduction

The reconstruction of valid generative models from data has been subject of several scientific disciplines. Some of the developed techniques have now been used and enhanced to derive models of genetic networks from gene expression data. Most notable are the works of Dana Pe'er (2001) and Horimoto Katsuhisa (2001). Generated models are hoped to disclose yet unknown coherences, in particular quantifiable dependencies between the expression levels of genes. Such a dependency could be preconceived in future biological research in order to find explanations for the functionality of genes.

With a similar intention we considered those methods for constructing generative models from data of metabolic concentrations produced by gaschromatographic mass spectrometry (GC/MS) (Fiehn (2000)). The derivation of dependencies between the metabolites of the given data set is a desirable capability in the process of finding new or validating known metabolic pathways respectively. Newly discovered dependencies can be used as feasible hypothesis to be examined in further experiments.

Up to now, the reported techniques all have a high computational complexity, which makes it difficult to use them on large data sets, such as expression profiling data. Also, if it is required to include these algorithms in

a loop, e.g. in an automatic analysis, they are unfit for smaller data sets like GC/MS data. This is why we developed a pragmatic approach that adopts some of the heuristics of the decision tree algorithm C4.5 (Quinlan (1993)) and thereby reduces the complexity of the needed calculations. This approach and a sample result will be discussed below.

2 Background and related work

2.1 Visualizing correlation graphs

A very simple approach to extract information from given metabolic concentration data is the visualisation of correlations within a data set (Kose (2001)). This method computes Pearson correlation coefficients (Statistica (2002)) between all measured metabolites. The visualisation is then given by a graph made of one node for each metabolite. For each correlation coefficient that exceeds a given threshold the respective nodes are linked with an edge. Once all edges are established it is possible to group those metabolites which are interconnected to one another. Groups, having edges between all of their members, are called cliques. It is presumed that cliques contain metabolites which appear in related metabolic pathways.

This approach is based on the Pearson correlation coefficient. It is therefore only possible to detect undirected linear and pairwise correlations with it, thus no information can be drawn about a cause and an effect of a computed correlation. Our approach extends this work by using and calculating a different type of correlation.

2.2 Usefulness of partial correlation

Some illustrative examples of standard statistical textbooks (Statistica (2002)) show that more information can be discovered while searching for dependencies between variables, if partial correlations are computed instead of simple correlation coefficients.

Partial correlation reveals correlations between more than two variables. It is sometimes also called "conditional" correlation, because it searches for correlations between two variables under a certain fixed condition of other variables. With this property, it is potentially possible to establish a cause from an effect (Shipley (2002)). Hence, we can use this feature to attribute directions to the edges of the above graph.

However, there are two drawbacks concerning the computability of partial correlations: First, the calculation has a high complexity, because pairwise correlations have to be calculated of all variables for any considered condition of the other variables. Second, small data sets lead to unreliable results, because there are very few samples left, once the data possessing a certain fixed condition is removed.

Therefore, it would be desirable to have a means of reducing the computational cost and circumvent the problems with sparse data sets.

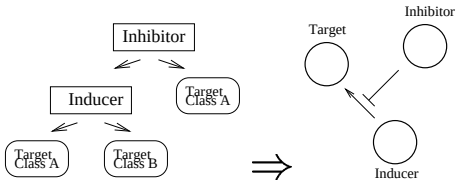

Fig. 1. A two-node tree with a possible interpretation as a hypothesis for a partial correlation.

2.3 Partial correlation and decision trees

Decision trees are a representation of discrete-valued functions. The parameters of the function can either be discrete-valued or real-valued. In practice, these trees are often learned automatically by one of several induction algorithms in order to approximate a target function over a given data set (Mitchell (1997)). The goal of such a function is always to classify a data object (a vector of attributes) into one of several possible target classes. Classifying an object is sometimes also referred to as predicting the value of the object's target variable.

At first glance, inducing a decision tree and calculating (partial) correlation are two completely different things. However, there is a relation between them. A decision tree can be used to predict a discrete target variable. For decision-making the tree uses the values of some other variables of the data set. So, a tree predicts the target variable by the value of other variables. If a correlation between two variables is known with all parameters, it is also possible to predict the value of the one variable by the value of the other. The difference is that a real valued variable can be predicted with the knowledge of a correlation whilst a decision tree can only predict discrete-valued variables.

If we discretize a real-valued target variable in an appropriate way, e.g. intervals of the target variable's domain, the above observation suggests that a decision tree would use those variables for a prediction which are correlating to the target variable. In fact, if there is one strongly correlating variable to the target variable, the induction algorithms will choose it as the only node in a tree for predicting the target, because the entropy regarding the target classes will drop most, if the data is split according to the value of a correlating variable.

Further, if a variable correlates only to the target variable under a certain condition of another variable, that is a partial correlation, the induction algorithms are likely to reflect this by generating a tree with two levels, which tests for the conditional variable in the top node and for the partially correlating variable in the subsequent node(s) as illustrated in Figure 1.

Thus, a decision tree offers a feasible hypothesis for one existing (partial) correlation between the target variable and other variables of the data. Since

decision trees can be grown at a lower computational complexity than direct calculation of partial correlations, they could be used to find them in a faster manner.

3 Algorithm for growing and evaluating trees

3.1 Growing a decision forest

Due to an unlimited number of possible discretizations of a real-valued domain there is potentially a vast number of possible decision trees that can be grown over a real-valued target variable. It would therefore be convenient to have a way of growing a limited set of trees with little redundancy in their indicated correlations. Sets of decision trees are also referred to as decision forests.

We propose an iterative algorithm for the growth of a decision forest that removes each variable that appears as top node of an induced tree from the data set and then inducts the next one on the same target. Thereby the maximum number of produced trees is limited to the number of variables in the data set. Since each variable can appear only once as top node in a tree, that is a key variable for the proposed correlation, there will additionally be little redundancy in the set of suggested hypothesis.

Due to the heuristics utilised by C4.5 the algorithm tends to reveal simple correlations in simple trees[1] as long as directly correlating variables are still in the data set. Once those are taken out of the data, C4.5 tries to find combinations of (foremost two) variables that still allow for a precise prediction of the target. These two-node trees primarily test for partially correlating variables. The property of delivering simple trees at first and more complex ones on later iterations will later be used to terminate the computation early.

To run this algorithm we still need to identify a target variable and discretize it in an appropriate way. We suggest to use a variable that indicates the presence of two separate states, because the instances of such a variable can easily be discretized into its two states, leaving only 0 and 1 as classes for the target. To find an eligible target it is helpful to look for bimodally distributed variables as illustrated in Figure 2. This can be done either with statistical means (Silverman (1981), Statistica (2002)), or visually, or through expert knowledge about a specific variable that reflects two states.

A very effective way to reduce the amount of computation in this step is to limit the depth of the decision trees in advance. If the depth is limited to 2 levels, hypothesis for partial correlations will still be found, but the complexity of the calculation is reduced by an entire factor.

[1] normally one-node trees

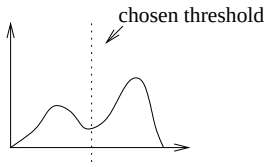

Fig. 2. A bimodally distributed variable being discretized visually.

3.2 Evaluating trees from the forest

To reduce the number of trees to be taken as potential hypothesis for (partial) correlations we introduce an evaluation function. Intuitively, it is desirable to get trees which classify most data objects correctly only with a few decisions. This would mean that the target variable can be predicted on the basis of only a few other variables, proposing that these are in strong (partial) correlation with the target. Whereas a tree needing many decisions for prediction supposedly tests variables that are only weakly (partially) correlated to the target.

With C_i being the set of correctly classified objects in leafs at depth i of a decision tree the function

$$value(T) := \sum_{i=1}^{d_{max}} |C_i| \cdot \frac{1}{e^i}$$

delivers a value between 0 and $\frac{1}{e}$ times the number of all data objects. A tree correctly classifying all objects close to the top receives a value close to or above 1, and a value close to 0 is given for trees that need many decisions to make their prediction.

This function can also be used to decrease the number of tree inductions, thus further reducing the computational cost of the growth of the forest. The first trees in the induction process tend to put strongly correlating variables at the top of the tree, because they split the data with least entropy. If there are no strongly correlating variables left, the objects can only be classified after a combination of tests on other variables. The more complicated the tree, the less probably it proposes an existing correlation, and the less is the value of the evaluation function. For this matter the growth of the forest could be stopped when the value of the evaluation function falls short of a certain threshold[2].

3.3 Assembling a network

At this point we have shown how to efficiently grow a forest and how to select potentially good hypothesis with the help of an evaluation function. We now fit the selected hypothesis into a graph according to a few simple rules:

[2] Empirically the value 0.5 showed to be a reasonable choice for a threshold.

1. A node is put into the graph for the target attribute.
2. For each one-node[3] tree, we insert a node for the test attribute of the tree into the graph and an undirected edge between this node and the node of the target attribute.
3. For each two-node tree, we insert a node into the graph for each node of the tree and a directed edge from the node of the second level of the tree which classifies most objects to the node of the target attribute. A second directed edge is inserted from the node referring to the top node of the tree to the first directed edge. The second edge should have a different styled pointer, because it indicates an inhibition or activation of the influence of the first edge.
4. All nodes in the graph referring to the same variable are merged.

The complete procedure (selecting target, growing forest, assembling graph) can now be rerun with another target variable until all suitable target variables have been used. The graph will grow with each iteration of the procedure.

The algorithm for generating a network needs a data matrix and delivers a hypothesis for a network in form of a graph. Here is a summarized description of it:

1. While suitable target variables are present do
   (a) Choose a target variable and discretize it.
   (b) Grow and store a decision tree to predict the target variable.
   (c) Take the top variable of the generated tree out of the data set.
2. Assemble the network from the stored decision trees.

4 Application on data of metabolic concentrations

We tested the algorithm on metabolic concentration data of potatoes. The potatoes had been grown under two distinct conditions, so some of them could respond to one condition and some to the other. Because of these two situations a few of the concentrations of the metabolites were clearly bimodally distributed, indicating a direct reaction to the two environments.

To examine the output of the algorithm in a biological context, we chose one experimentally unidentified[4], bimodally distributed metabolite as target, a maximum tree depth of two, and an evaluation threshold of 0.5 for the algorithm. The resulting graph is shown in Figure 3. The numbers in front of each name are the trace and the retention index from the GC/MS experiment. Here they serve exclusively as identifiers; for metabolites with unidentified names, as is the target metabolite in the centre, they are the only reference.

[3] We also consider trees that classify a vast majority of all objects correctly with just one node as one-node trees.

[4] Sometimes, in GC/MS experiments metabolites are clearly detectable and measureable, but they cannot reliably be identified.

The numbers above or below the names are the values of the evaluation function for the corresponding decision trees.

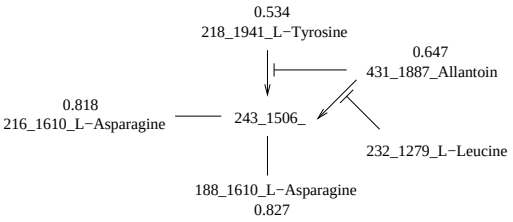

Fig. 3. The output of an iteration of the algorithm for an example metabolite.

Fig. 4. A pathway of the KEGG database leading from Leucine on the left to Asparagine on the right.

By consulting KEGG we found that all displayed metabolites of the output graph are involved in the amino acid metabolism. The KEGG database (http://www.kegg.com/) is a resource comprising large-scale data in Genomics, Proteomics and Metabolomics. In particular, we retrieved the pathway of Figure 4 from Leucine to Asparagine approximately matching an indicated correspondence in the graph. It perambulates the same metabolites as the direct path from Leucine to Asparagine in the graph of Figure 3. The notation and numbers in the figure are taken directly from KEGG. The numbers below the structural depictions are identifiers for the metabolites, the numbers above the arrows indicate the involved enzyme for the reaction. They are discussed in more detail on the KEGG site.

These results confirm an interrelation of the algorithm's output and biological functionality.

5 Discussion

We have introduced an algorithm that produces a partially directed graph visualizing (partial) correlations from data. The output graph can be interpreted as a hypothesis for causalities between the variables of the data set. We have shown in an example that the delivered hypothesis is also consistent with biological knowledge, specifically with metabolisms.

Our approach is driven by the demand to produce hypothesis for metabolic (sub-) networks at a reduced computational cost. The obtained results are encouraging in view of this goal. This allows us to run the algorithm in a loop or a computationally demanding environment. Hence, we hope that the use of our algorithm on metabolic concentration data will allow for quicker and more complex results.

We can only compare our work in terms of computational cost to other works with a similar objective, because the other approaches have only been applied on expression profiling data, yet. But due to the lower complexity of a heuristical search as opposed to an exhaustive search our algorithm does offer an advantage on the computational part.

There are several ideas on how to further improve the algorithm. First, it would be a helpful feature to be able to evaluate the induced hypothesis not only on a statistical measure, but also on a function based on prior knowledge. Second, the discretization of the target variable(s) is mostly done manually at this time. General statistical approaches for finding bimodalities (K-Means-Clustering, Silverman-Test) fail to deliver usable results on sparse data sets. For a fully automatic data analysis, however, it is necessary to automatically find and reliably discretize bimodalities in a biological context.

6 Acknowledgements

We thank Joachim Kopka, Ralf Steuer, and Jacques Nicolas for their helpful collaboration and comments.

References

Fiehn O., Kopka J., Dörmann P., Altmann T., Trethewey R.N., and Wilmitzer L. (2000): Metabolite profiling for plant functional genomics. In: Proc. Natl. Acad. Sci. USA, 95, 14863-14868.

Horimoto K. and Toh H. (2001): Automatic System for Inferring a Network from Gene Expression Profiles. *Genome Informatics, Vol. 12 2001, p270-271.*

Kose F., Weckwerth W., Linke T., Fiehn O. (2001): Visualising plant metabolomic correlation networks using clique-metabolite matrices. *Bioinformatics, Vol. 17, p1198-1208.*

Mitchell, Tom (1997): *Machine Learning.* McGraw-Hill, Boston, USA.

Pe'er D., Regev A., Elidan G. and Friedman N. (2001): Inferring subnetworks from perturbed expression profiles. *Bioinformatics, Vol.17 Suppl.1 2001, S215-S224.*

Shipley, Bill (2002): *Cause and Correlation in Biology.* Cambridge University Press, Cambridge, UK.

Silverman, B.W. (1981): Using Kernel Density Estimates to investigate Multimodality. *Journal of Royal Statistical Society, Series B, 43,97-99.*

Statistica (2002), *Electronic Textbook StatSoft.*
http://www.statsoftinc.com/textbook/stathome.html

Quinlan, J.R. (1993): *C4.5: Programs for Machine Learning.* Morgan Kaufmann, San Mateo, USA.