# Chapter 3

# Exploring molecular characteristics of disease

**— differential coexpression of groups of genes —**

***Synopsis:*** *This chapter is on exploring molecular characteristics of disease. I describe the* `dcoex` *algorithm designed to highlight groups of differentially coexpressed genes. That is it looks for groups of genes that are coherently expressed in one (predefined) group of samples but lose this coherence in another group. After motivating the objective I develop the* `dcoex` *algorithm. Application to childhood leukemia data yields biologically plausible results. Statistical significance, robustness and novelty are assessed.*

## 3.1 Motivation

Gene expression is a tightly regulated process, crucial for the proper functioning of a cell. In microarray data, coregulation of genes is reflected by strong correlations between expression levels [45]. Molecular disease mechanisms typically constitute abnormalities in the (co)regulation of genes [55]. Resulting changes in expression profiles help identifying disease related genes and in several cases facilitate improved diagnosis and prognosis of disease outcome [65, 112, 128, 147, 151, 152, 157]. Alteration of gene regulation often results in up or down regulated genes and common analysis strategies look for these differentially expressed genes.

Not all relevant changes in gene expression need to be manifested by up or down regulation of individual genes. It is well conceivable that changes in the coregulation structure of genes lead to a diseased phenotype. While in healthy cells a group of genes may be under a common regulatory control, this control might be lost for diseased cells. In that

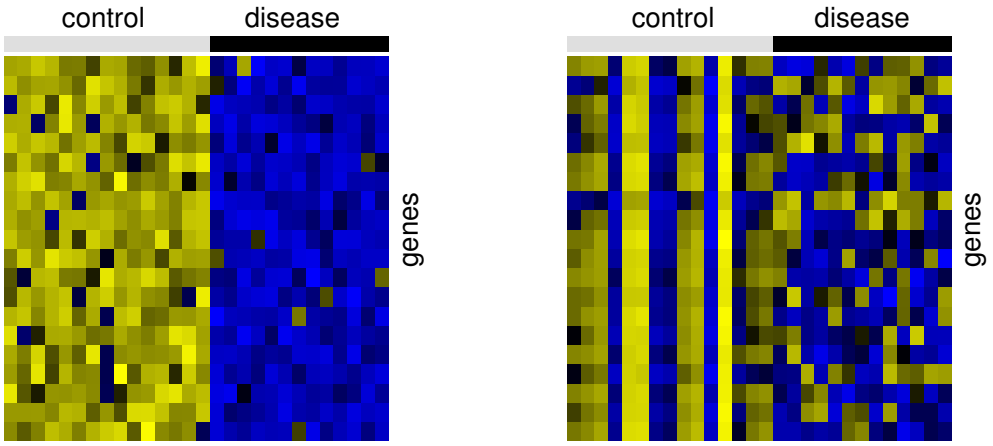


***Figure 3.1: Different types of structure in microarray data.*** *The plots show simulated heatmaps for two prototypical situations. We compare of a group of control patients with a disease group. The plot on the left side shows single differentially expressed genes. The right plot shows a group of differentially coexpressed genes, displaying a pattern as a group. Both situations lead to characteristic but distinct patterns.*

case, coexpression of the genes involved would disappear. We call such a group of genes *differentially coexpressed* between the healthy and the disease phenotype. We do not hypothesize that regulatory control as such is lost, but merely that the *common* regulation of a group of genes breaks down.

We can expect that differential coexpression of a group of genes leads to characteristic patterns in the expression data. To illustrate the patterns, we contrast differential expression with differential coexpression in a simple simulation experiment: To simulate the differentially expressed genes, we sample each expression value from a normal distribution with its mean dependent on the phenotype. For the differential coexpression pattern we employ an additive model regarding gene and patient effects, which is described in detail in Section 3.2.

The result is shown in Figure 3.1, showing two heatmaps as a color representations of the simulated expression matrices. Each column corresponds to a sample and each row to a gene. Samples are ordered corresponding to membership in control or disease group, while the genes are hierarchically clustered. The left plot displays differentially expressed genes, which have a (significantly) different mean expression in each group. The genes in right plot are differentially coexpressed: they display a coherent expression pattern in the control group which is lost in the disease group. None of the genes in the right plot is differentially expressed, the mean expression of all genes is almost identical in both groups, control and disease.

From a more formal perspective, differential gene expression aims for genes with changes in first order moments (mean expression). Our approach attempts to identify groups of

genes with different dependency structure in the two phenotypical groups: When the genes are coregulated the expression values should be highly dependent, while they become independent when the common regulation is lost. Assuming e.g. a normal model for (generalized-log transformed [39, 68]) gene expression values, this translates to finding gene groups with phenotype-conditional changes in the correlation structure (second order moments).

In contrast to differential expression, differential coexpression cannot be analyzed gene by gene. Since we are looking for groups of genes that exhibit changes in its members' interplay, we need to take into account all possible groups. In typical microarray experiments the number of genes is in the order of tens of thousands, implying about $2^{10000}$ subsets. The challenge is to efficiently screen the astronomically large number of subsets. In Section 3.2 we derive an algorithm for doing so. We start by defining a quantitative concept of coexpression, leading to a score for differential coexpression that allows for an efficient evaluation of candidate sets of differentially coexpressed genes. In Section 3.3 we apply the algorithm to simulated and real data. We discuss the biological plausibility as well as the robustness and novelty of our findings.

## 3.2 The dcoex algorithm

In this Section we introduce an algorithm for finding groups of differentially coexpressed genes. Starting with quantifying the concept of differential coexpression, we take up a score for the coexpression of genes previously suggested by Cheng et al. [24]. We adopt it to our situation and generalize it to account for *differential* coexpression, where it enables an efficient screening for candidate sets of differentially coexpressed genes. This makes a local search heuristic feasible for large gene expression data sets.

### 3.2.1 Scoring differential coexpression

We now turn to deriving a score for differential coexpression of a set $I$ of genes. We start with quantifying the coexpression of a set of genes and discuss the resulting score. Scoring differential coexpression will be a straight forward generalization of the coexpression case.

#### Scoring coexpression

The intention behind assigning a coexpression score to a group of genes is to single out low scoring gene sets as candidates for coregulation. Coregulation cannot be directly observed on (non-interventional) microarray data, and we take coexpressed groups as candidates for coregulated groups. This is the same rationale as the one behind popular clustering and biclustering approaches, e.g. [5, 45, 93]. While coexpression of two genes

may be quantified by e.g. the Pearson correlation coefficient, it is not clear how to generalize this to a group of genes. We choose to call a group of genes $I$ to be coexpressed across a group of samples $J$, if the expression values of all the genes in each sample are highly dependent.

We assume that for a group $I$ of coexpressed genes all the expression levels of all the genes are approximately the same for a fixed patient. Expression value estimates do not come on an absolute scale [67], therefore we allow each gene its specific offset (on the generalized-log transformed data). Further on the common expression level of the genes may differ between patients, as the samples might be from patients in different conditions and taken under different circumstances. Formalizing the above, we use the following model: If $\tilde{\mathbf{C}}$ is a $p \times n$ expression matrix (e.g. of a control group of patients) on generalized-log scale, we consider a group $I$ of genes coexpressed in a group of samples $J$, if

$$\tilde{c}_{ij} = a_i + b_j + \varepsilon_{ij} \quad \text{for } i \in I \text{ and } j \in J \tag{3.1}$$

with small residuals $\varepsilon_{ij}$. In Equation (3.1) the $a_i$ compensate for the gene specific scales, while the $b_j$ adjust for the patient heterogeneity. How well this additive model fits the data can be quantified by the mean squared residual $\frac{1}{|I||J|}\sum_{i\in I\, j\in J}\varepsilon_{ij}^2$. A small mean squared residual corresponds to a coexpressed group of genes, while a large value implies a group of not coexpressed genes.

In our case, the group of samples $J$ is either the control or the disease group and predetermined by the data. We do not optimize over this parameter, as is the case in biclustering settings [24, 93, 156]. Our focus is on the group of genes $I$ alone. An estimate for the gene specific scale factors $a_i$ in Equation (3.1) is given by the mean of the expression values across the samples. For this reason we consider the *row centered* expression matrix $\mathbf{C} = (\boldsymbol{c}^{(1)}, \ldots, \boldsymbol{c}^{(p)})^T$, with $\boldsymbol{c}^{(1)}, \ldots \boldsymbol{c}^{(p)} \in \mathbb{R}^{n_C}$. The mean squared residual score for a group of genes is then defined by

$$\begin{aligned} S_C(I) &= \frac{1}{|I| n_C} \sum_{i\in I} ||\boldsymbol{c}^{(i)} - \boldsymbol{\mu}^C||_2^2 \\ \boldsymbol{\mu}^C = \boldsymbol{\mu}^C(I) &= \frac{1}{|I|} \sum_{i\in I} \boldsymbol{c}^{(i)} \quad , \end{aligned} \tag{3.2}$$

as was proposed before for biclustering [24]. This is the mean squared residual of the additive model in Equation (3.1), where the parameters $a_i$ and $b_j$ have been estimated by row and column means. If $S_C(I)$ is small, the genes in $I$ are coexpressed. If it is large, they are not. A low scoring gene set implies that expression values are well approximated by their mean value in each sample. In fact, a group of genes with a small $S_C(I)$ corresponds to a group of $\boldsymbol{c}^{(i)}, i \in I$ gathered tightly around their mean. The score is the same as the mean squared Euclidean distance of the group members from their mean.

***Relation of $S_C$ to ANOVA*** We shortly relate the coexpression score with the ANOVA

framework. For convenience, let **C** now denote the row-centered submatrix defined by selecting the genes in $I$ and the samples from the control group $J$. The model of Equation (3.1), leading to the coexpression score $S_C$, is also used in the context of *tow way analysis of variance (ANOVA)*. There, one considers a random variable $Y_{ij} = \mu_{ij} + \epsilon_{ij}$ with $\epsilon_{ij} \sim N(0, \sigma^2)$. For convenience, assume $1 \leq i \leq |I|$ and $i \leq j \leq |J|$. This model implies the assumptions of independence, normality and homoscedasticity. In the case of a single observation per cell, factor interactions cannot be identified and one tries to fit a model of additive factor effects [89]:

$$\mu_{ij} = \mu + \phi_i + \theta_j \quad ,$$

where $\sum_i \phi_i = 0$ and $\sum_j \theta_j = 0$ for identifiability. Maximum likelihood estimators of the model components are known to be $\hat{\mu} = \bar{Y}$, $\hat{\phi}_i = \bar{Y}_{i\cdot} - \bar{Y}$, $\hat{\theta}_j = \bar{Y}_{\cdot j} - \bar{Y}$. Identifying the $Y_{ij}$ with the $\mathbf{C}_{ij}$, we see that in our case $\hat{\mu} = 0 = \hat{\phi}_i$ (for all $i$) and the $\hat{\theta}_j$ equal the column-means. Therefore, the coexpression score is related to the *error sum of squares (SSE)* of a corresponding two way anova model:

$$S_C = \frac{SSE}{|I||J|}.$$

The ANOVA framework provides rigorous statistical tests to decide if one of the factor effects, say $\phi_l$ or $\theta_m$, are different from zero. The theoretical basis forms Cochran's theorem, from which for example follows that $SSE/\sigma^2 \sim \chi^2((|I|-1)(|J|-1))$. We, on the other hand, simply employ $S_C$ as a measure of coherence amongst the row-vectors of **C**.

***Relation of $S_C$ to the Pearson correlation coefficient*** Here we relate the coexpression score $S_C$ to the Pearson correlation coefficient between the row-vectors (gene-vectors) of **C**. As the correlation coefficient is scale invariant, we may assume the rows of **C** to be *scaled* as well. That is, we have $||\boldsymbol{c}^{(i)}||_2^2 = 1$, with $\boldsymbol{c}^{(i)}$ the $i$-th row-vector of **C**. In that case, the correlation coefficient of the $i$-th and the $k$-th gene-vector, $r_{ik} = \boldsymbol{c}^{(i)T}\boldsymbol{c}^{(k)}$, equals the cosine between the two row-vectors. Let us further define the group-size $m := |I|$, the vector of column-means $\boldsymbol{\mu}^C := \sum_i \boldsymbol{c}^{(i)}/m$ (as before) and the residual vectors $\boldsymbol{\epsilon}^{(i)} := \boldsymbol{c}^{(i)} - \boldsymbol{\mu}^C$. This implies $\sum_i \boldsymbol{\epsilon}^{(i)} = \mathbf{0}$ and $\boldsymbol{\mu}^C = \boldsymbol{c}^{(k)} - \boldsymbol{\epsilon}^{(k)}$ for any $1 \leq k \leq m$. As a measure of coherence of the $m$ gene-vectors in **C**, the *average* correlation $\bar{r} := \frac{1}{m(m-1)} \sum_{i \neq j} r_{ij}$ may be of interest. For this quantity we get:

$$\begin{aligned}
\sum_{i,j} \boldsymbol{c}^{(i)T}\boldsymbol{c}^{(j)} &= m(m-1)\bar{r} + m \\
&= m^2 \boldsymbol{\mu}^{CT}\boldsymbol{\mu}^C = m^2\left(\boldsymbol{c}^{(k)} - \boldsymbol{\epsilon}^{(k)}\right)^T\left(\boldsymbol{c}^{(k)} - \boldsymbol{\epsilon}^{(k)}\right) \\
&= m\sum_k \left(||\boldsymbol{c}^{(k)}||_2^2 + ||\boldsymbol{\epsilon}^{(k)}||_2^2 - 2||\boldsymbol{\epsilon}^{(k)}||_2^2 + \mathbf{0}\right) \\
&= m^2(1 - |J|S_C)
\end{aligned}$$

and therefore

$$\bar{r} = 1 - \frac{m}{m-1}|J|S_C \quad .$$

For the case of $m = 2$, where we have only two genes in the group $I$, it holds that $\bar{r} = r$ and

$$r = 1 - 2|J|S_C \quad .$$

In summary we have that in case the residuals are small, all the gene vectors $\boldsymbol{c}^{(i)}$ are highly correlated (on average). Also, this notion of coexpression accounts for heterogeneity amongst the patients: Genes can be highly expressed in one patient and less so in another, still leading to a small score in case the expression varies coherently.

***Further properties of*** $S_C$ Finding a group of coexpressed genes can now be done by finding a group of genes with a low score $S_C$. Note a few properties of $S_C$:

- Single genes are perfectly coexpressed. This in in accordance with $S_C(I) = 0$ for all $I$ with $|I| = 1$ and $S_C(I) > 0$ for all $I$ with $|I| > 1$ (assuming we have no identical gene vectors $\boldsymbol{c}^{(i)}$).
- For every $I$ we can take out a gene to decrease $S_C(I)$ [24]: For all $I$ there exists a gene $m \in I$ such that $S(I \setminus m) > S(I)$.
- $S_C(I)$ is a *pseudo Boolean function* [17]: $S_C(I)$ constitutes a mapping $\mathscr{P}(\mathscr{I}) \mapsto \mathbb{R}_+$ from the power set of the set of all genes $\mathscr{I}$ into the positive reals.

From the first two properties we know that it does not suffice to look for a group of coexpressed genes as $I^\star = \min_I S_C(I)$, as this will always be solved by any $I$ with $|I| = 1$. But we can look for groups of $k$ coexpressed genes as $I^\star = \min_I S_C(I)$ subject to $|I| = k$. We will have the same situation for the case of differential coexpression.

It is useful to discuss the effect of including or excluding a group $M$ of genes from $I$ has on the resulting score. We can (see Appendix) express the new score in terms of the old score and the old means $\boldsymbol{\mu}^C$ via:

$$\begin{aligned} S_C(I \setminus M) &= \frac{k}{k-|M|} S_C(I) - \frac{1}{n_C(k-|M|)} \sum_{i \in M} ||\boldsymbol{c}^{(i)} - \boldsymbol{\mu}^C||_2^2 \\ &\quad - \frac{1}{n_C(k-|M|)^2} \sum_{i,j \in M \times M} (\boldsymbol{c}^{(i)} - \boldsymbol{\mu}^C)^T (\boldsymbol{c}^{(j)} - \boldsymbol{\mu}^C) \\ S_C(I \cup M) &= \frac{k}{k+|M|} S_C(I) + \frac{1}{n_C(k+|M|)} \sum_{i \in M} ||\boldsymbol{c}^{(i)} - \boldsymbol{\mu}^C||_2^2 \\ &\quad - \frac{1}{n_C(k+|M|)^2} \sum_{i,j \in M \times M} (\boldsymbol{c}^{(i)} - \boldsymbol{\mu}^C)^T (\boldsymbol{c}^{(j)} - \boldsymbol{\mu}^C). \end{aligned} \tag{3.3}$$

Note that the last of the terms on the right hand sides of Equation (3.3) will *always* reduce the score. This can be seen by considering the Matrix $A = \{a_{ij}\}$ with $a_{ij} = (\boldsymbol{c}^{(i)} - \boldsymbol{\mu}^C)^T (\boldsymbol{c}^{(j)} - \boldsymbol{\mu}^C)$. The last term is then proportional to $\mathbf{1}^T A \mathbf{1}$, with $\mathbf{1}$ a vector of all ones of

dimension $|M|$. As $A$ is a Gram matrix [124] by construction it is positive semi-definite and therefore $\mathbf{1}^T A\mathbf{1} \geq 0$. Keeping that in mind, Equation (3.3) provides a straight forward criterion whether exclusion or inclusion of a gene $m$ into $I$ will decrease the score: Inclusion will be beneficial if $||\boldsymbol{c}^{(m)} - \boldsymbol{\mu}^C||_2^2 < n_C S(I)$, and exclusion when $||\boldsymbol{c}^{(m)} - \boldsymbol{\mu}^C||_2^2 > n_C S(I)$. The above derivation of these criteria is simpler than that in [24]. We will utilize the results to derive corresponding criteria for the case of differential coexpression.

#### Scoring differential coexpression

In the previous section we introduced a scoring scheme for the coexpression of a group of genes. In this section we utilize the results to score the *differential* coexpression of a set of genes. Let $S_C(I)$ score the coexpression of the genes in $I$ in the control group and $S_D(I)$ in the diseased group. We define a score for differential coexpression via of the genes in $I$ (with respect to $D$ and $C$):

$$S(I) := \frac{n_D}{n_C}\frac{S_C(I)}{S_D(I)} \quad , \tag{3.4}$$

where $n_C$ and $n_D$ code for the number of samples in the control and disease groups. That is, we are *simultaneously* looking for good coexpression in one group and low coherence in the other group: Gene groups with a low score are coexpressed in the control group, but in the disease group coherence is lost. This is illustrated in Figure 3.2. To find the group of the $k$ most differentially coexpressed genes we have to solve the problem

$$\min_I \frac{n_D}{n_C}\frac{S_C(I)}{S_D(I)} \quad \text{subject to} \quad |I| = k \quad . \tag{3.5}$$

At this point it is not obvious that we have to keep the constraint $|I| = k$, but in the next section we see that we are in the same situation as before: We can always find a gene in $I$ which is favorable to exclude. Equation (3.5) is a binary (or 0-1) polynomial fractional program [143]. Chang [23] provides a procedure to approach such problems by transformation to a (linear) mixed integer program. Since such programs are generally NP-hard, we resort to a heuristic search strategy.

### 3.2.2 A stochastic search algorithm for low scoring gene sets

Having formulated a score for differential coexpression, we are left with the objective to find one (or possibly many) low-scoring groups of genes. That is, we set out to minimize $S(I)$ over all possible candidate groups $I$. Since there are usually tens of thousands of genes involved in a microarray experiment, an exhaustive search is clearly infeasible. Since rigorous minimization of $S(I)$ is hard, we now describe a heuristic algorithm for finding low scoring sets of genes.

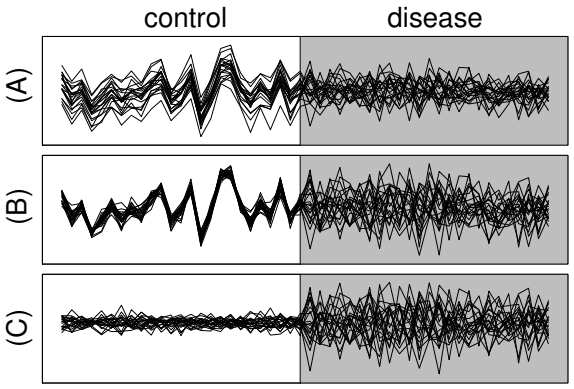


***Figure 3.2: Expression profiles of differentially coexpressed genes.*** *We show simulated expression values (y-axis) versus patients (x-axis). In the patients to the left (control group) the genes are coexpressed, but not so in the patients in the disease group. Plot (A) shows the expression values, in Plot (B) the genes have been centered. Plot (C) shows the residuals $\varepsilon_{ij}$ that make up the score. One can see that the mean squared residual is a measure of the co-expression.*

The general framework is that of a stochastic greedy downhill procedure. We begin by equipping the set of all possible subsets of genes on the array with a neighborhood structure. We define two sets of genes as neighbors, if and only if they can be converted into each other via inclusion or exclusion of a single gene: $\mathcal{N}(I) := \{\tilde{I} : |I \triangle \tilde{I}| = 1\}$, where $\triangle$ denotes the symmetric difference operator: $A \triangle B = (A \cup B) \setminus (A \cap B)$. The strategy of the downhill procedure is as follows: Starting with a random set of genes $I$, we move to neighboring sets taking downhill directions. We include or exclude genes such that the score decreases. This is continued until no more downhill directions are found or a predefined number of iterations is exceeded. Two points need to be addressed: One is related to computational efficiency, and a second one is related to the size of the target gene sets $I$.

### Efficient screening of candidate gene sets

Evaluating the score for a candidate gene set requires calculating means in the control and disease group as well as the sum of the squared residuals. For data sets with tens of thousands of genes this can be computationally demanding, rendering a naive implementation of the downhill procedure impractical.

We present an algorithm that screens neighborhoods more efficiently. The key ingredient is an efficient to calculate criterion whether the inclusion or exclusion of a candidate gene $m$ will reduce $S(I)$. Assume we are performing the downhill search and arrive at a set $I$ of genes. To address the question whether inclusion or exclusion of a gene is beneficial, recall the definition of $S$:

$$S(I) = \frac{n_D S_C(I)}{n_C S_D(I)} = \frac{n_D}{n_C} \frac{\sum_{i \in I} ||\boldsymbol{c}^{(i)} - \boldsymbol{\mu}^C||_2^2}{\sum_{i \in I} ||\boldsymbol{d}^{(i)} - \boldsymbol{\mu}^D||_2^2} \quad . \tag{3.6}$$

Reshaping Equation (3.3) for the inclusion or exclusion of a group of genes of size one, we arrive at:

$$\begin{aligned} S_C(I \setminus m) &= c_-^1 \, S_C(I) - c_-^2 \, ||\boldsymbol{c}^{(m)} - \boldsymbol{\mu}^C||_2^2 \\ S_C(I \cup m') &= c_+^1 \, S_C(I) + c_+^2 \, ||\boldsymbol{c}^{(m')} - \boldsymbol{\mu}^C||_2^2 \quad , \end{aligned}$$

where the group $I$ was of size $k$, $c_+^1 = k/(k+1)$, $c_+^2 = k/((k+1)^2 n_C)$, $c_-^1 = k/(k-1)$ and $c_+^2 = k/((k-1)^2 n_C)$. With that, we can write $S(I \setminus m)$ in the following form:

$$S(I \setminus m) = \frac{c_1 \, a - c_2 \alpha}{c_1 \, b - c_2 \beta}$$

with appropriately substituted variables. For the inclusion of gene $m'$ we get an analogous representation with the plus sign exchanged to a minus sign. The criterion for a reduced score we get from the following Lemma:

**Lemma 1.** *Let $s_1 = \frac{a}{b}$,$s_2 = \frac{a+\alpha}{b+\beta}$ and $s_3 = \frac{a-\alpha}{b-\beta}$ with $a$, $\alpha$, $b$ and $\beta$ all $\in \mathbb{R}_+$. Then:*

*(i)* $\frac{\alpha}{\beta} < s_1 \quad \Leftrightarrow \quad s_2 < s_1$

*(ii)* $\frac{\alpha}{\beta} > s_1 \quad \Leftrightarrow \quad s_3 < s_1$

**Proof** By rewriting:

$(i)$ $s_2 < s_1$ implies $a + \alpha < a/b\,(b+\beta)$ hence we have $\alpha/\beta < a/b = s_1$

$(ii)$ $s_2 > s_1$ implies $a + \alpha > a/b\,(b+\beta)$ hence we have $\alpha/\beta > a/b = s_1$ □

Re-substituting variables we arrive for the inclusion case at:

$$S(I \setminus m) < S(I) \quad \text{if and only if} \quad \frac{||\boldsymbol{c}^{(m)} - \boldsymbol{\mu}^C||_2^2}{||\boldsymbol{d}^{(m)} - \boldsymbol{\mu}^D||_2^2} > \frac{S_C(I)}{S_D(I)} \tag{3.7}$$

which makes a useful criterion when considering removing gene $m$. It is very fast to evaluate, since all quantities have already been calculated to compute $S(I)$. For the inclusion of gene $m'$ we arrive at:

$$S(I \cup m') < S(I) \quad \text{if and only if} \quad \frac{||\boldsymbol{c}^{(m')} - \boldsymbol{\mu}^C||_2^2}{||\boldsymbol{d}^{(m')} - \boldsymbol{\mu}^D||_2^2} < \frac{S_C(I)}{S_D(I)} \quad . \tag{3.8}$$

Using the criteria above, we can explore the neighborhood $\mathcal{N}(I)$ of $I$ by looking up the corresponding $||\boldsymbol{d}^{(m)} - \boldsymbol{\mu}^C||_2^2$ and $||\boldsymbol{d}^{(m)} - \boldsymbol{\mu}^D||_2^2$ and a simple division. This is faster than calculating every $S(\tilde{I})$ for all $\tilde{I} \in \mathcal{N}(I)$, leading to the applicability of the algorithm to large scale expression studies. Further speedup is achieved by going several steps at a time. We collect all genes in the neighborhood of $I$ that meet the criterion for a diminished score. From these genes we randomly choose a predefined fraction $\beta$ and ex- or include them at once. We will discuss the influence of $\beta$ in more detail at the end of this section. The randomization of both, the start set and the choice of downhill directions,

in principle results in different identified gene sets belonging to different local minima of $S$. This feature is highly useful for exploratory analysis, since in large data sets we expect more than one relevant pattern.

### Tuning the size of the target set

In this Section we show that the differential coexpression score has the same property as the coexpression score discussed before:

- There always exists a gene $m \in I$, such that $S(I \setminus m) < S(I)$.
- There does not always exists a gene $m \notin I$, such that $S(I \cup m) < S(I)$.

These two properties imply that iterative application of the criteria for a reduced score would eventually exclude all genes from the solution $I$. To see that we can always exclude genes from $I$, write the score in the following form:

$$S(I) = \frac{n_D}{n_C} \frac{\sum_i \alpha_i}{\sum_i \beta_i} \quad ,$$

where we have written out the sums in Equation (3.6). Note that $\alpha_i \geq 0$ and $\beta_i \geq 0$. For a gene $m$ which is beneficial to exclude, the criterion (3.7) reads in the above notation $\alpha_m/\beta_m > S(I)$. There will always be a gene meeting this requirement:

**Lemma 2.** *Let $s = \sum_i^n \alpha_i / \sum_i^n \beta_i$ with $\alpha_i$ and $\beta_i \in \mathbb{R}_+$. Then either there exists an $m$ with $\alpha_m/\beta_m > s$ $(1 \leq m \leq n)$ or $\alpha_i/\beta_i = s$ for all $1 \leq i \leq n$.*

**Proof** Rewrite $s$ as $s = \sum_i (\alpha_i/\beta_i)\beta_i / \sum_i \beta_i$ and assume there exists no $m$ with $\alpha_m/\beta_m > s$. Then we have $s \geq \sum_i s\beta_i / \sum_i \beta_i = s$ where equality holds if and only if $\alpha_i/\beta_i = s$ for all $1 \leq i \leq n$. □

The above argument shows that there will always be genes in $I$ that can be excluded to improve the score. That we do not necessarily find genes to include shows the following consideration: Assume we arrive at the optimal solution $I^\star$ of size $k$. Then we can always reduce the score by excluding a suitable gene, as shown before. After excluding a gene and reducing the score, no gene can fulfill the criterion for being included again, because the current score (with $I$ of size $k-1$) is lower than the optimal score for groups of size $k$. The score will increase with every gene we include. For this reason, an iterative application of the criteria for a reduced score would eventually exclude all genes from the solution $I$.

To compensate, we utilize modified criteria: We exclude gene $m$ from $I$ only if

$$C_m^-(\alpha) = \alpha \Big[ \frac{||\boldsymbol{c}^{(m)} - \boldsymbol{\mu}^C||_2^2}{||\boldsymbol{d}^{(m)} - \boldsymbol{\mu}^D||_2^2} - S(I) \Big] - (1-\alpha)\frac{1}{|I|} > 0 \quad , \tag{3.9}$$

and we include gene $m'$ into $I$ if

$$C^{+}_{m'}(\alpha) = \alpha\Big[S(I) - \frac{||\boldsymbol{c}^{(m')} - \boldsymbol{\mu}^{C}||_2^2}{||\boldsymbol{d}^{(m')} - \boldsymbol{\mu}^{D}||_2^2}\Big] + (1-\alpha)\frac{1}{|I|} > 0 \quad . \tag{3.10}$$

In the criteria above we introduced a tuning parameter $\alpha$ in $[0,1]$. It weights a penalty term for excluding too many genes and therefore influences the size of the final set of genes. Larger values of $\alpha$ yield smaller groups of genes and vice versa.

The introduction of $\alpha$ can be viewed as modifying the original optimization problem $I^{\star} = \text{argmin}_I\, S(I)$ into:

$$I^{\star} = \underset{I}{\text{argmin}}\, \left\{\alpha S(I) - (1-\alpha)\log|I|\right\} \quad . \tag{3.11}$$

The log-term in the objective function counteracts the property of $S(I)$ to exclude too many genes. This kind of penalty term also leads to the second terms in the right hand side of Equations (3.9) and (3.10):

$$\begin{aligned} \log(|I|+1) - \log|I| &= +1/|I| + \mathcal{O}(1/|I|^2) \\ \log|I| - \log(|I|+1) &= -1/|I| + \mathcal{O}(1/|I|^2) \quad . \end{aligned}$$

The criteria (3.9) and (3.10) trade-off two opposing contributions to an objective function. To be able to perform such a comparison, we should in principle characterize the effect of including or excluding a gene on the residual score $S(I)$ quantitatively. As it is, $S(I) - ||\boldsymbol{c}^{(m')} - \boldsymbol{\mu}^{C}||_2^2 / ||\boldsymbol{d}^{(m')} - \boldsymbol{\mu}^{D}||_2^2$ is only *proportional* to the benefit of including gene $m'$. This flaw can be fixed by considering the proportionality constant, namely $c_{m'}/(S_D(I) + c_{m'})$ with $c_{m'} = ||\boldsymbol{d}^{(m')} - \boldsymbol{\mu}^{D}||_2^2$. For the exclusion case there is a similar constant, $c_m/(S_D(I) - c_m)$, which can be shown to be positive. This also complies with intuition, as we expect a gene closer to the mean than the group average ($c_m < S_D(I)$) in the samples where we are looking for decoherence to be beneficial to exclude. But overall, we found the criteria (3.9) and (3.10) to work well in practice as they are. Algorithm 1 (Figure 3.3) summarizes the overall procedure.

**Runtime experiment**

We now briefly examine how the algorithm performs with respect to a naive implementation of a downhill search. Naive in the sense that for each candidate set the complete score is calculated in contrast to considering the criteria 3.7) and (3.8). We generate random data ($iid \sim N(0,1)$) for a variable number of genes and run both downhill searches. Setting $\alpha = 1$ fixes the size of the final group to one. In this experiment $\beta$ is disregarded and we in- or exclude only one gene per downhill step. Results are shown in the left plot of Figure 3.4. We show the quotient of the naive and our running time, and improvements can be seen to be substantial. The naive running time is shown below the

**ALGORITHM 1**

indent=2em

initialize $I$ randomly
$G \leftarrow \emptyset$
**while** counter < maxiter **do**
  **for** all $\tilde{I} \in \mathcal{N}(I)$ **do** {*screen neighborhood*}
    $m \leftarrow \tilde{I} \triangle I$
    **if** $C_m^{(+,-)}(\alpha) > 0$ **then** {*select favorable genes*}
      $G \leftarrow G \cup \{m\}$
    **end if**
  **end for**
  **if** $G = \emptyset$ **then** {*exit for a locally optimal solution*}
    return $I$
  **else**
    $n \leftarrow \max\{\lfloor \beta \cdot |G| \rfloor, 1\}$
    $g \leftarrow$ uniform sample of size $n$ from $G$ {*select a subset of favorable genes*}
    $I \leftarrow I \triangle g$ {*update I*}
  **end if**
  counter $\leftarrow$ counter + 1
**end while**
return $I$

***Figure 3.3: A greedy downhill approach for finding local minima of the score*** *$S$. A random starting point $I$ is chosen. The neighborhood $\mathcal{N}(I)$ is explored using the criteria $C_m^{(+,-)}(\alpha)$ defined in Equations (3.9) and (3.10). Downhill steps are taken, possibly several at a time, before $S(I)$ is recalculated. This is repeated until no further downhill directions are found, or until a predefined number of iterations is exceeded. The operator $\triangle$ denotes the symmetric difference of two sets: $A \triangle B = (A \cup B) \setminus (A \cap B)$.*

speedup factor; we see that the naive algorithm needs about 21 minutes for 1000 genes whereas the improved version finishes in less than one minute.

The right plot of Figure 3.4 shows the effect of $\beta$ on the running time. The data are again random, and $\alpha$ has been fixed to 1/2. The running time improves with increasing $\beta$. As increasing $\beta$ also increases the chance of missing high quality minima, we choose $\beta = 0.01$, making computations with several thousands of genes feasible.

## 3.3 Application to data

In this section we describe the application of the algorithm to two types of data, simulated and real. By applying the algorithm to simulated data we are able to objectively

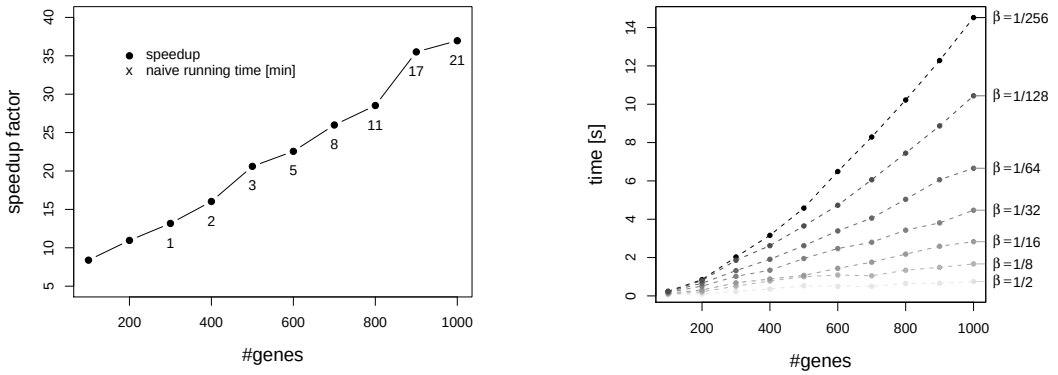


***Figure 3.4: Accelerating the downhill search**. The left plot shows the speedup achieved by implementing Equations (3.7) and (3.8) compared to naively exploring neighborhoods. The naive running-time is shown below the speedup factor. Improvements are substantial, even when considering only about thousand genes. The right plot shows the influence of $\beta$ on the running time. The running time improves with increasing $\beta$. Since increasing $\beta$ also increases the possibility of overlooking high quality minima, we choose $\beta = 0.01$, making computations with several thousands of genes feasible. The runtime experiments were performed on an `AMD Athlon(tm) 64 Processor` running at 2000 MHz (QuantispeedRating: 3200+) and on random data (see text).*

evaluate its performance and study the influence of the tuning parameter $\alpha$. While the application to real life clinical data is straight forward, interpretation of results is not. We give an example of a group of differentially coexpressed genes we find in childhood leukemia. Further on, we assess the possibility that our finding is a chance artifact, its robustness to variation of the starting point of the search procedure and the sensitivity to perturbation of the input data.

### 3.3.1 Simulated data

To test the algorithm we choose a supervised setting where the truth is known. We simulate data to generate a control group of $n_C = 10$ "samples" and a disease group of another $n_D = 10$ "samples" for $p = 120$ genes. For the control group, we draw 20 coexpressed genes directly from the additive model:

$$c_{ij} = a_i + b_j + \varepsilon_{ij}.$$

We take $a_i \sim N(0,\sigma)$, $b_i \sim N(0,\sigma)$ and $\varepsilon_{ij} \sim N(0,\sigma/s)$, and multiply the $c_{ij}$ by a factor of $\sqrt{\sigma^2(2+1/s^2)}$ to ensure an expected standard deviation of $\sigma$. For the remaining 100 genes we use $c_{ij} = N(0,\sigma)$. Concerning the disease group we sample data only via $d_{ij} \sim N(0,\sigma)$. We choose $\sigma = 1$ and use $s$ to tune the signal to noise ratio. This simple simulation experiment cannot be expected to resemble real data, but we use it to evaluate the algorithm in a controlled setting. We run it on the simulated data and check

how many of the 20 differentially coexpressed genes we recover. We do this for different values for the signal to noise ratio: $s = 0.5, 1, 1.5, 2, 3$ and 7. We choose different $\alpha$ between zero and one to observe the effect on the solution. To guard for sampling effects we repeat the procedure 1000 times and report the median performance. Results are shown in Figure 3.5. The upper plot shows the median number of genes in the pattern the algorithm retrieved. The lower plot reports the true positive rate, i.e. the fraction of the genes in the pattern made up from the 20 differentially simulated genes. Lighter gray values correspond to a lower $s$, darker gray values stand for a more clear signal.

We observe that for reasonable signal strength ($s > 1$) the algorithm recovers the majority of the 20 genes. Further on, $\alpha$ can be used to tune the size of the pattern recovered. Finally, the influence of $\alpha$ on the size of the solution is less prominent for a clear signal (large $\beta$) as compared to noisy signals (small $s$). This implies that for a clear signal the algorithm is less sensitive towards the effects of the user-tunable parameter.

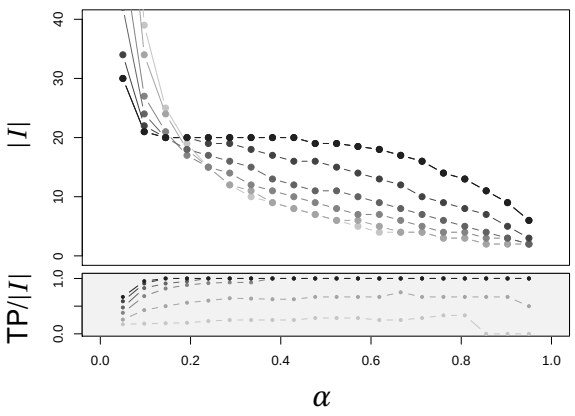


***Figure 3.5: Performance of the algorithm on simulated data.*** *The upper plot shows the group size k of the solution depending on tuning parameter $\alpha$. The lower plot shows the according true positive rate, i.e. the fraction of genes in the solution made up from the twenty hidden genes. The different gray shades represent different signal to noise ratios from $s = 1/2$ (lightest) to $s = 7$ (darkest). See text for a more detailed description of the setup.*

## 3.3.2 Clinical data

### The data set

As a real world application we choose a dataset from a study on acute lymphoblastic leukemia [157]. Affymetrix HG-U95Av2 chips were used to measure the gene expression levels in bone marrow from children with acute leukemia. The data set consists of 327 samples, which are divided into several subgroups according to characteristic cytogenetic aberrations. This includes a normal group of leukemia patients where no such aberrations are found. We compare all cytogenetic positive groups (disease groups) to leukemia without cytogenetic alterations (control).

**Results**

First, we normalize and variance stabilize [68] the expression data. For computational efficiency, we calculate the absolute deviation from the median for all genes and discard the lower 50% of them. Note that the score $S$ is not scale invariant. We rescale the genes separately for each phenotypical group to standard deviation one. Otherwise, the algorithm fails to detect differential coexpression patterns; instead high scoring gene sets show differential variance of single genes.

We found patterns reflecting differential coexpression in almost all analyses of cytogenetically characterized leukemia. A particularly nice example showed up comparing Philadelphia positive (t(9;22)+, BCR–ABL+) leukemia to the cytogenetically normal group, choosing $\alpha = 0.5$. A prominent local minimum of $S$ is shown in Figure 3.6.

The left plot shows a heatmap of the expression matrix. It clearly identifies the stripe-pattern we expect for differentially coexpressed genes (compare Figure 3.1). The right plot depicts the expression profiles of the genes as lines. We see the genes are more coherently coexpressed in the control group, as we expect for differentially coexpressed genes (compare Figure 3.2). One can see in all the plots that we have found a group of genes displaying the characteristic pattern we are looking for.

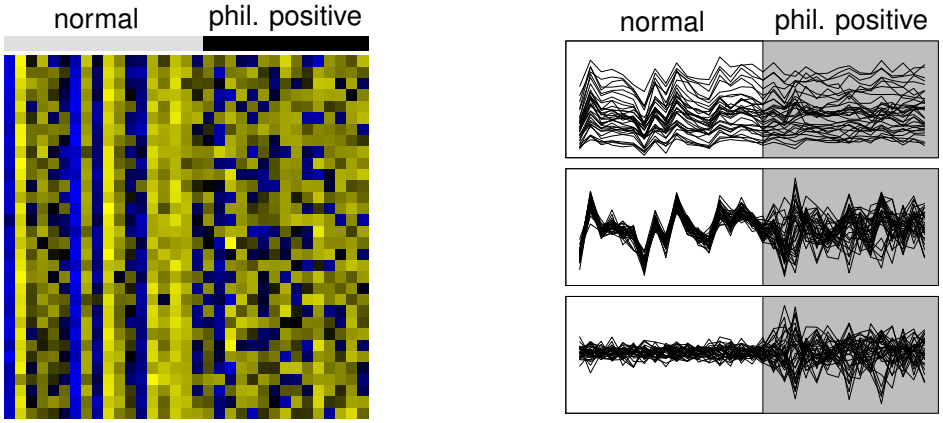


***Figure 3.6:*** ***A group of differentially coexpressed genes in Philadelphia positive leukemia****. The left plot shows a heatmap of the expression matrix. It clearly identifies the stripe-pattern we expect for differentially coexpressed genes (compare Figure 3.1). The right plot depicts the expression profiles of the genes as lines. We see the genes are more coherently coexpressed in the control group, as we expect for differentially coexpressed genes (compare Figure 3.2).*

Excluding the genes that form our pattern from the data and applying the algorithm again, we end up with a cluster of genes reflecting a different local minimum. The list of genes corresponding to the two patterns is shown in Table 3.1.

| **Pattern 1** (34 genes) | | | |
|---|---|---|---|
| HINT1 (3094) | PSMA2 (5683) | PSMB1 (5689) | FNTA (2339) |
| CG018 (90634) | RHEB (6009) | ARHGDIB (397) | NCL (4691) |
| EXO70 (23265) | KARS (3735) | KIAA1579 (55225) | SRP19 (6728) |
| RNF4 (6047) | HCP15 (157317) | HNRPA2B1 (3181) | RNP24 (10959) |
| ECHS1 (1892) | C9orf10 (23196) | REA (11331) | UGP2 (7360) |
| GNAS (2778) | GNAS (2778) | OTOR (56914) | COPE (11316) |
| ATP5A1 (498) | ATP5J2 (9551) | PTOV1 (53635) | YWHAQ (10971) |
| CBFB (865) | HLA-Z (267017) | COX6A1 (1337) | STARD7 (56910) |
| CALM2 (805) | PPP2R1A (5518) | | |
| **Pattern 2** (21 genes) | | | |
| PSMB2 (5690) | UBC (7316) | ARHA (387) | PSMA4 (5685) |
| ARPC2 (10109) | ACTB (60) | ENG (2022) | GG2-1 (25816) |
| ACTR3 (10096) | PITPNB (23760) | CAPNS1 (826) | PFN1 (5216) |
| SF3B2 (10992) | GNAI2 (2771) | ARHA (387) | PPP1CC (5501) |
| PSMC5 (5705) | ARPC5 (10092) | MRLC2 (103910) | AES (166) |
| GDI2 (2665) | | | |

***Table 3.1: List of differentially coexpressed genes in philadelphia positive leukemia.*** *In brackets we give the LocusLink ID. Genes that are underlined are either annotated to be members of the proteasome-ubiquitin pathway (PSMB1, PSMA2, PSMB2, PSMA4, PSMC5 and UBC), are known to interact with it (YWHAQ), or they are associated with protein transport / cellular location (SRP19, RNP24, COPE, FNTA and GDI2).*

### Biological plausibility

Our method is exploratory in the sense that we can neither guarantee to find the optimal scoring set of genes, nor can we derive rigorous statistical tests for the significance of score levels. This is due to the fact that the scores on the randomized data also result from a heuristic search. Nevertheless, our procedure can be used as a tool for the exploratory analysis of microarray data, complementing the frequently used clustering procedures. Its purpose is hypothesis generation, not hypothesis verification.

Any interpretation of exploratory analyses is speculative, and further biological experiments are needed for verification. In our analysis of Philadelphia positive leukemia, we detected two differential coexpression patterns, containing several genes of the proteasome-ubiquitin pathway, namely *PSMB1*, *PSMA2*, *PSMB2*, *PSMA4*, *PSMC5* and *UBC*. The proteasome plays a central role in regulation of proteins that control cell-cycle progression and apoptosis [3]. For various cancer types it has been shown that inhibition of proteasome activity results in programmed cell death, and it has therefore become an important target for anti cancer therapy [3]. Particularly, a study on chronic lymphoblastic leukemia (CLL) shows, that CLL derived lymphocytes are hyper sensitive to proteasome inhibition (by lactacystin) as compared to normal human lymphocytes [95–97]. Another interesting gene in the pattern is *YWHAQ*, coding for a 14-3-3 theta protein. Fujita et al. [50, 81] show that the protein Akt phosphorylates p27, a prognostic tumor marker, and thereby promotes binding of 14-3-3 theta allowing degradation of p27 by

the proteasome. Increased degradation of p27 is associated to decreased overall survival in mantle cell lymphoma, another B-cell malignancy [27].

Recalling our method, it is important to note that genes in differential coexpression patterns can only be interpreted in the context of the other genes in the set that form the pattern. In our case, one functional group of genes is associated to intracellular protein transport and cellular localization, including *SRP19*, *RNP24*, *COPE*, *FNTA* and *GDI2*. This group of genes is functionally more heterogeneous than the proteasome associated proteins. But altogether one expects protein synthesis, protein transport and protein degradation to be highly coordinated processes.

### 3.3.3 Significance, robustness and novelty of results

***Significance*** Naturally, the question arises whether our findings are artifacts of the high dimensionality of the data. To asses this question, we apply a permutation procedure: We repeatedly shuffle patient labels ($a$) for all genes at once and ($b$) for each gene separately. We run the algorithm on the permuted data and the frequency of scores smaller than or equal to the biological score then yields an empirical $p$-value. While ($a$) assesses the impact of non-phenotype related grouping, it might be prone to the effect of confounding variables. Version ($b$) is less conservative by removing all correlation structure from the null hypothesis.

Figure 3.7 displays the results for the pattern from the leukemia example. The $x$-axis denotes the score $S$, the $y$-axis the group size $|I|$ of the retrieved patterns. The dashed lines represent a constant value of $S(I) - log(|I|)$, as motivated by Equation (3.11). We see that in both sampling schemes the leukemia result is unlikely to appear by chance. We only observe one random instance of $S(I) - log(I)$ smaller than the biological result using sampling scheme ($a$). This corresponds to an empirical p-value of 0.001. Utilizing sampling scheme ($b$), the realization of the original result becomes even less likely. Hence, it is unlikely that the observed differential coexpression is a chance artifact.

***Robustness*** To assess the stability of the patterns, we consider two sources of variation on the data. First, we explore how varying the starting point of the search influences the result. After that we assess how subsampling the two groups affects the patterns found.

Assessment of sensitivity with regard to the startingpoint: We start the algorithm from 10,000 different random points and compare the output. The random starting points are calculated by flipping a coin for each gene and including it with probability 0.3 into a start pattern. Doing so, we discover six different local minima, all made up of different subsets of the same 41 genes. Three of the minima are prominent, as they make up the vast majority of all the 10,000 runs (see Figure 3.8). Only two of the 41 genes are not present in one of the three minima. The barplot on the right of Figure 3.8 displays the frequency how often a gene contribute to a minimum. Genes present in patten 1 of

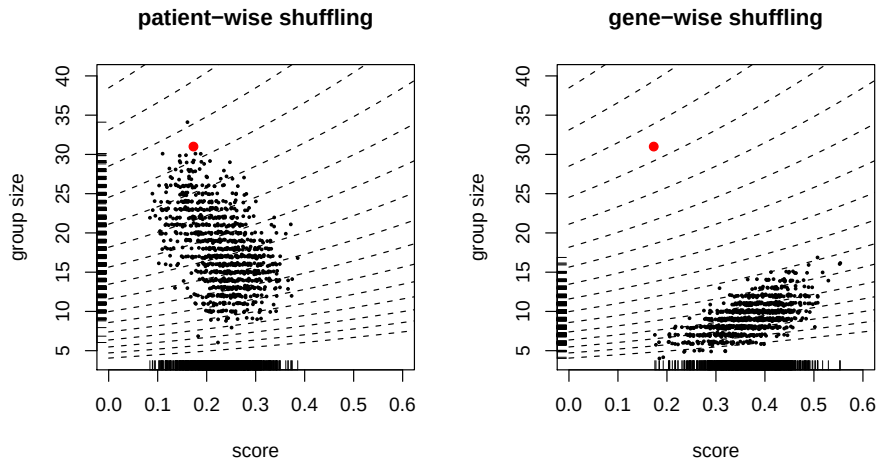


***Figure 3.7: Permutation based significance assessment.*** *The figure displays the results of repeatedly (1000 times) shuffling the patient labels and running the algorithm. The $x$-axis denotes the differential coexpression score $S$, the $y$-axis the group size $|I|$ of the patterns to which we added jitter. In the left plot the labels are shuffled "by patient", whereas in the right plot they are shuffled for each gene separately. The dashed lines denote a constant value of $S(I) - log(|I|)$, as motivated by Equation (3.11). The red dot corresponds to the leukemia example. We see that in both sampling schemes the original result is unlikely to appear by chance.*

Table 3.1 are in gray, the others are in white. Most genes contribute to all six minima, and all of them are contained in our previous results. This, combined with the fact that only 41 different genes were found contributing to minima, hints at remarkable stability of our algorithm with respect to variation of the starting point.

To assess the effect of perturbing the data, we subsample from the BCR–ABL+ and the normal group. We generate about 100 such subsampled data sets, randomly excluding two of the patients from each group. This corresponds to discarding more than 10% of the available data. Then we run the algorithm 100 times on each data set with different starting points chosen in the same manner as before. We find 468 unique patterns, consisting of different subsets on 329 genes. In spite of the seemingly large number of patterns found, the low number of genes involved supports the robustness of our algorithm. Averaging over all runs, each of the 329 genes is present in more than 900 resulting gene sets. Also, some genes contribute to more of the 468 patterns than others. This is illustrated in Figure 3.9 which shows an histogram of the frequencies with which the 329 genes contribute to a differential coexpression pattern (DCEP). We find that only few genes contribute to many of the DCEPs. We also notice that the genes contributing to the DCEPs in the complete data set are all high frequency genes. This is indicated by the black bars in the figure which mark bins containing at least one of these 41 genes. That is, we find the structure of the patterns of the full data set conserved in the results obtained on the subsampled data sets. In this sense the algorithm exhibits robustness

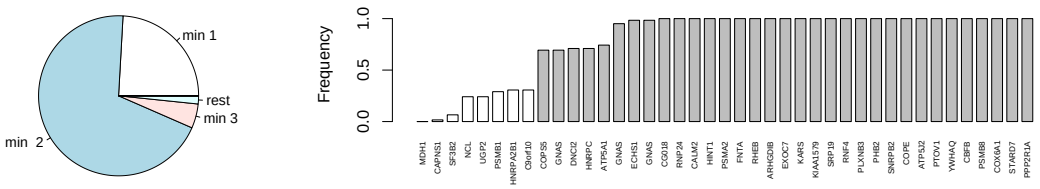


***Figure 3.8: Varying the starting point of the downhill search.*** *On the left side a pie-chart shows the number of times each minimum was recovered in 10,000 runs with random starting points. We see that three of the six minima we encounter are dominant, whereas others appear rarely. The bar plot on the right side shows for each gene the frequency of contributing to a minimum. Many genes contribute to all minima. Genes present in Pattern 1 of Table 3.1 are in gray, the others are in white.*

against data perturbation.

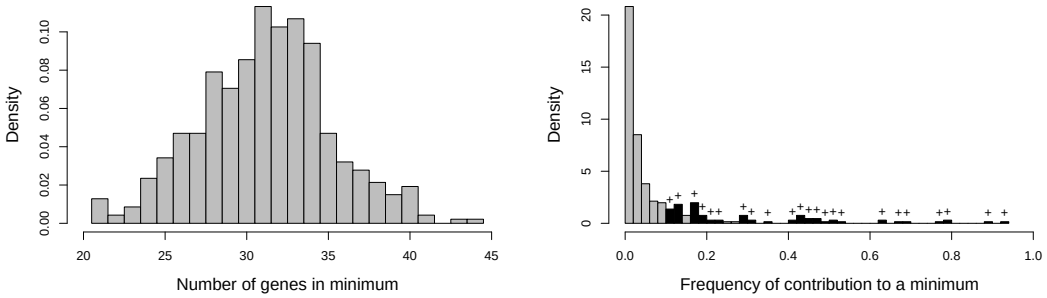


***Figure 3.9: Perturbing the data through subsampling.*** *The left histogram shows the distribution of the size of minima we recover from 100 random starting points on 100 subsampled data sets (see text). The right histogram is on the frequency with which a gene contributes to a minimum. We find only 329 genes contributing to all the 10,000 minima, and some of them with high frequency. Bins of the histogram containing genes from Pattern 1 of Table 3.1 are marked in black. Most of them are high frequency genes.*

***Novelty*** Next, we show that our algorithm is complementary to established analysis tools. It aims for specific structure in microarray data that other methods fail to identify. Of course, the literature on microarray data analysis is huge and we cannot discuss all of it. We confine ourselves to showing that the most widely used approaches, namely screening for differential expression and hierarchical clustering, do not identify differential coexpression structure. To address differential expression we sort all genes according to their individual t-scores. This shows the genes from our first pattern to be ranked from 106 to 6114 with a mean of 2340. Clearly, this list would not draw attention to the genes in the pattern.

We also calculate a hierarchical clustering of the expression data using Euclidean distance and average linkage [45] and Euclidean distance. The complete results are too complex to display. Therefore, we aggregate genes in a first clustering step choosing 100 representatives for clusters. These we cluster again hierarchically resulting in Figure 3.10. The black dots indicate clusters that contain genes from Pattern 1 in Table 3.1. The right plot shows the results for unscaled data, the left plot for scaled data. In both cases, one can clearly see how the genes are distributed all over the first half of the dendrogram. Hierarchical clustering would not have identified Pattern 1.

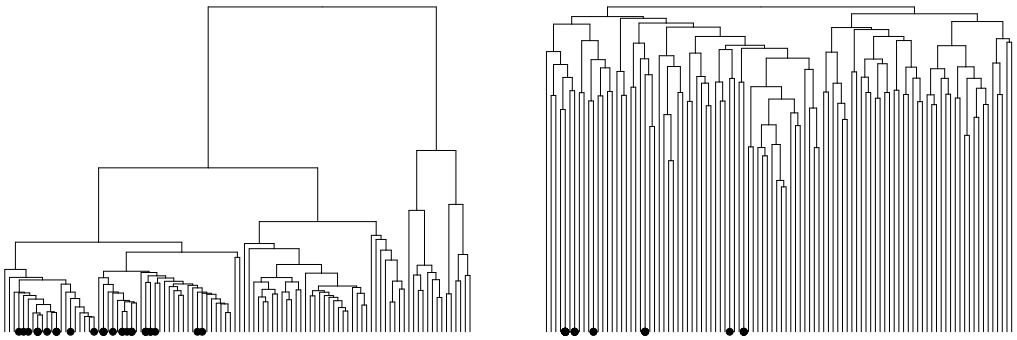

***Figure 3.10: Comparison to hierarchical clustering****. We applied hierarchical clustering to unscaled (left) and scaled data (right). Leaves which represent clusters that contain at least one of the genes from Pattern 1 in Table 3.1 are marked with a black dot. We see that hierarchical clustering does not identify the differential coexpression pattern we found.*

## 3.4 Discussion and chapter summary

In this chapter we introduced the concept of *differential coexpression* of groups of genes. We motivate our approach and develop the `dcoex` algorithm designed to find groups of differentially coexpressed genes. As a proof of concept we apply the algorithm to simulated data. We also present results on a data set on childhood leukemia, finding a group of differentially coexpressed genes in which ubiquitin- ubiquitylation- and proteasome-associated genes are overrepresented. We discuss biological plausibility, statistical significance, robustness and novelty of our findings.

While, to the best of our knowledge, we are the first to propose this type of analysis strategy the approach has been taken up by other researchers, e.g. [108]. The `dcoex` algorithm is an exploratory analysis tool, and its results are meant to complement those of other such tools like clustering or biclustering in order to help generate new hypotheses concerning e.g. the biology of neoplasms. This chapter also includes several improvements over our original publication [85], most notably a thorough derivation and discussion of the gene-inclusion and -exclusion criteria that enable a fast neighborhood

search (Equations (3.10) and (3.9)). A runtime experiment has been added, comparing our algorithm to a naive approach. Along with adding a section on robustness the assessment of significance has been extended.

Overall, this chapter introduces the concept of differential coexpression of groups of genes—a useful concept that complements clustering and biclustering in exploring the molecular characteristics of disease. We provide the `dcoex` algorithm in form of an R-language [70] add-on package that we briefly describe in the Appendix.