Heuristic Bayesian segmentation for discovery of co-expressed genes within genomic regions

Petri Pehkonen, Garry Wong, and Petri Törönen

Abstract—Segmentation aims to separate homogeneous areas from the sequential data, and plays a central role in data mining. It has applications ranging from finance to molecular biology, where bioinformatics tasks, such as genome data analysis, are active application fields. In this paper, we present a novel application of segmentation in locating genomic regions with co-expressed genes. We aim at automated discovery of such regions without requirement for user given parameters. In order to perform the segmentation within a reasonable time, we use heuristics. Most of the heuristic segmentation algorithms require some decision on the number of segments. This is usually accomplished by using asymptotic model selection methods like Bayesian information criterion. Such methods are based on some simplification which can limit their usage. In this paper, we propose a Bayesian model selection to choose the most proper result from heuristic segmentation. Our Bayesian model presents a simple prior for the number of segments. This is usually accomplished by using asymptotic model selection methods like Bayesian information criterion.

Index Terms— Biology and genetics; clustering, classification, and association rules; segmentation.

1 INTRODUCTION

GENE-EXPRESSION profiling using microarrays facilitates simultaneous profiling of thousands of genes in a single experiment. A common analysis for such data is clustering, which can reveal co-expressed genes in a set of biological conditions or time points. Such genes can be regulated in concert, for example due to the same transcription regulators. Particularly, some biological problems concern transcriptional mechanisms that are chromosome region specific. These include chromatin remodelling in eukaryotes [1, 2], function of operon like structures in bacteria [3], regulation in the duplicated regions [4], and epigenetic transcriptional mechanisms [5]. In such studies, the gene expression data is often compared with the chromosomal locations of genes. This is often performed by measuring correlations of expression profiles between neighbouring genes [6] or calculating the frequencies of co-expressed genes within a sliding window [7].

In this study, we aim at finding the borders of different sized regions with co-expressed genes, and separating such regions from each other and surrounding areas. This is pursued without setting predefined parameters, such as the size of a sliding window [7]. We address this problem by a novel application of a technique called segmentation. Its objective is to find borders or change-points between homogeneous local regions in a data stream or sequence. This has been applied to a wide range of areas previously, such as analysis of financial stock time point data [8], as well as many approaches of DNA sequence analysis [9, 10]. The locations of change-points, and in most applications their number, are unknown and are estimated from data. In this work, we consider the borders between the chromosome regions with co-expressed genes as the change-points, and seek their locations and number.

The optimal solution to the segmentation problem, for $N$ sized data with $k$ segments $(n=k-1$ change-points), can be obtained in time $O(kN^2)$ with dynamic programming based methods (e.g. [11]). These methods have been advantageous, as they facilitate a fully Bayesian approach by enumerating over all parameter values of a segmentation model (see e.g. [12, 13]). Due to large running times, approximate and heuristic algorithms are widely used with large datasets [14]. Approximate algorithms typically approximate the posterior distribution for segmentation solutions with fixed $k$, and recent methods concern also the estimation of $k$ [15, 16, 17]. In turn, the heuristic methods try typically to exclude suboptimal solutions from the search. These methods are popular as they are often very fast and perform well in many practical applications [17].

Commonly used heuristic segmentations are the greedy Top-Down (TD) and Bottom-Up (BU) methods [18, 19, 20], and their various modifications [21, 14]. These methods are popular as they facilitate rapid production of segmentations for different numbers of change-points, and are easy to implement. In this work, we have chosen to use the TD algorithm, as we aim finally at creating an application web server for large datasets. The algorithm runs in time $O(kN)$.

• P. Pehkonen is with the Department of Neurobiology, A.I. Virtanen-Institute, and Department of Biosciences, University of Kuopio P.O. Box 1627, FIN-70211 Kuopio, Finland. petri.pehkonen@uku.fi. E-mail: petri.pehkonen@uku.fi.

• G. Wong is with the Department of Neurobiology, A.I. Virtanen Institute, and Department of Biosciences, University of Kuopio P.O. Box 1627, FIN-70211 Kuopio, Finland. E-mail: garry.wong@uku.fi.

• P. Törönen is with the Bioinformatics Group, Institute of Biotechnology, P.O. Box 56, 00014 University of Helsinki, Finland. E-mail: petri.toronen@helsinki.fi.

Manuscript received (insert date of submission if desired). Please note that all acknowledgments should be placed at the end of the paper, before the bibliography.
Many heuristic segmentation algorithms produce a large set of potential solutions with a varying number of change points. As the change-point number is often unknown, a criterion for selecting the most proper solution is required. This is addressed in statistical model selection, where alternative models are evaluated in order to find the most suitable one. Typically, heuristic methods use model selection (see e.g. [9, 10, 22, 23]), such as Akaike Information Criterion (AIC) [24] and Bayesian Information Criterion (BIC) [25]. There are two weaknesses with these measures that are also considered in this paper. First, they typically use a maximum likelihood point estimation for data probabilities (e.g. [24, 25, 23]). Secondly, they are based on some simplifying assumptions, such as uniform prior distribution (e.g. [25, 23]) or independence of model parameters (e.g. [24]). This limits the usage of these measures to only a particular range of applications.

The work here presents a novel application of segmentation in gene expression analysis. At the same time, it improves the evaluation of the results from heuristic segmentations. We show several interconnected results. First, we present a Bayesian model for evaluating segmentations of multinomial data, which have different numbers of segments. As an input, the model requires one or more pre-evaluated segmentation solutions per each number of segments, and the observed data. Thus, it is suitable for heuristic methods, which often give such information as a preliminary result, and an alternative to measures like AIC and BIC. The model contains a simple, proper prior for the obtained change-point solution (similar than used earlier in [12]), and a novel objective prior for multinomial data. Second, although the developed Bayesian model is useful for selecting the number of change points, we show that with the TD method, it is better to use ML than the actual Bayes score for locating the next added change point. Third, we construct a framework for empirical comparison of different model selection methods with simulated data. This comparison shows that the developed Bayesian method outperforms the evaluated ML based methods. Also, the novel prior for multinomial data outperforms the other Bayes priors. Fourth, we apply our segmentation scheme in search of chromosomal regions with association to the gene co-expression clusters. The aim is to test whether the observed expression profiles (clusters) are localized to certain active regions of the genome.

2 HEURISTIC BAYESIAN SEGMENTATION

In this chapter, we describe the proposed segmentation scheme. First, the representation of the data is discussed in section 2.1. In section 2.2, we describe the TD algorithm. In the remaining sections, we describe the measures used in the change-point selection in the TD algorithm (2.3) and model selection for choosing a suitable solution (2.4 and 2.5).

2.1 Data Representation

We represent \( N \) genes of a chromosome as sequential data \( D_t \), where \( t = 1..N \). Each gene is represented with a data point \( D_t \), where the number \( t \) indicates the order of that gene in the chromosome (see figure 1 for details). As data, we use memberships of genes in initially obtained co-expression clusters. These clusters have been obtained from the preceding application of partitional clustering to gene expression data. In our approach, this clustering is performed multiple times with different numbers of clusters. This has two rationales. First, the lower cluster number facilitates searching more broader, and the higher cluster number more specific co-expression. Secondly, this decreases the effect of discretization that we make with data clustering. Each data point is a \( V \) dimensional vector \( D_t = (D_t^{(0)},...,D_t^{(v)},...,D_t^{(v)}) \in \{0,...,I_v\}^V \) indicating the memberships of the \( t \)th gene of the chromosome in \( V \) different clustering solutions. Here \( D_t^{(v)} \in \{0,...,I_v\} \) is the gene's cluster membership in the clustering solution \( v \), and \( I_v \) is the number of co-expression clusters in the clustering solution \( v \).

Figure 1 shows an illustration of the data mapping using two different gene expression clusterings with 3 and 4 clusters. The application is discussed in more detail in chapter 4, where we present the segmentation for finding co-expressed regions from yeast cell-cycle related gene expression data.

2.2 Segmentation Algorithm

The following steps are performed in the TD algorithm. For a given data sequence, a single change-point estimate is first searched. Then a 1-to-2 segmentation is performed in the estimated location (initial step). This creates two new sub-sequences, each of which is searched through for a new change-point estimate (step A). These two estimates are added to a priority queue sorted according to the global cost of an imaginary 1-to-2 segmentation in the location of each estimate (step B). The change-point estimate, that is the first in the priority queue, is chosen as a location for the next 1-to-2 segmentation (step C). The steps A-C are repeated until each data point is surrounded by change-points, the needed number of segments is obtained, or some stopping criterion is fulfilled. We execute 100 steps as we create artificial data with at most 10 segments, and as we expect many fewer segments from the biological datasets. This 1st repetition produces a solution with \( i+1 \) segments. This creates a hierarchy of nested solutions, including one solution for each number of change-points. Finally, the solution in the hierarchy that has the smallest cost, according to a model selection criterion, is chosen as a representative segmentation solution (step D).

An evaluating criterion is used in 3 different steps of the algorithm: in step A (and also in the initial step) for evaluating each position within a segment, in step B to sort the estimates in the priority queue, and in step D to choose the representative solution. We use a ML based measure (JS divergence) for steps A and B (and also for initial step). The motivation for this is given in section 2.3. In turn, for the model selection of step D, we compare performance of several different model selection methods. Three of these methods are based on ML and a penalization term: AIC, BIC, and modified BIC (see appendix I, equations A1, A2 and A3, respectively), denoted here with BIC2. Other methods are based on the developed Bayes model
presented in section 2.4, with various priors for multinomial data presented in section 2.5, including a novel modified prior. We use each model selection method for choosing a single solution from the hierarchy of all solutions.

### 2.3 Change-point Selection

Despite the use of a Bayesian approach (see 2.4) for evaluating different segmentation solutions in step D, we still decided to use a ML based measure Jensen-Shannon divergence ($D_{JS}$) [26] for locating individual change-points in step A (and in initial step) and sorting the priority queue in step B, in a similar way as in [27]. This was due to our observation, that when the Bayes Factor (BF) was used, the TD method created too many small segments while leaving larger segments intact.

In order to study the characteristics of ML and BF in change-point estimation, we used a one segment random model to create artificial data. This should not bias the methods toward any preferred position. Results are shown in supplementary file 1. ML scores are placed more or less randomly and evenly. However, the BF score with the FLAT prior (see definition in section 2.5) shows an average curvature that encourages the placement of the change-point to the ends of the data sequence. The same behaviour was observed with various priors.

The stronger negative outcome at the middle region with BF results potentially from having more certainty to the analysis than in the near border regions. This is due to observing more data on both sides of the change-point. This represses the effect of noise and provides a clearer decision with BF. In turn, at the ends, one of the generated segments will be small and the uncertainty increases, although the prior smoothes the signal. With uniformly distributed data, this feature of BF dominates the decision for placing a single change-point. Even so, the log-BF is always negative, favouring the selection of one segment model against the addition of the change-point. This is different to the ML method, which does not make similarly the decision for or against the splitting. Thus, with random data, the change-points are also more uniformly placed.

We found that the observed curve has an impact when using BF in change-point estimation with the TD method. With this algorithm, some borders that increase the cost of the complete segmentation model are often created in preceding steps, before finding the solution with the lowest cost. Thus, the algorithm confronts situations where it is required to choose among a set of several candidate positions (in the priority queue) that increase the cost. In such cases, the BF scoring favours the near end positions at the expense of the positions in the middle. The same effect would probably not be observed with the dynamic programming based methods or non-greedy heuristic methods using simultaneous placement of several change-points. Also, with very drastic cluster borders, this might not be a problem. However, due to use of a greedy algorithm, we prefer using ML scoring for placing change-points.

### 2.4 Bayesian Model Selection

The described segmentation produces one solution for each number of change-points. For model selection purpose (step D), we consider each solution as a model $M$ with two groups of parameters, denoted with $\Theta$ and $\Psi$. Group $\Theta$ includes parameters associated with representation of the data (class probabilities) within each segment and $\Psi$ includes parameters associated with division of data with change-points (segmentation solutions).

A typical method of Bayesian model selection is using BF that compares the integrated likelihoods of different models. For a segmentation model $M$, this would be implemented as follows:
where \( P(D | M, \Theta, \Psi) \) is the likelihood, \( P(\Theta | M) \) is the prior probability of data classes within the segments and \( P(\Psi | M) \) is the prior probability of individual segmentation solution, given a particular segmentation model \( M \). Parameters \( \Psi \) do not have an impact on the likelihood and therefore we can write \( P(D | M, \Theta, \Psi) = P(D | M, \Theta) \).

In order to reduce the computational complexity and facilitate evaluation of results from TD algorithm, we use certain simplifications for equation 1. In the first simplification, we assume that segments are not affected by the other segments, and they are therefore independent from each other. This allows us to combine the segment scores by simple multiplication. In the second simplification, we assume that different dimensions are similarly independent (when the data is multidimensional binary or multinomial). This assumption was somewhat violated in our application, as we represent the clustering solutions for the same data as different dimensions. This simplification was still considered to be important, as it enabled the combination of different dimensions using simply a multiplication.

Due to the previous two simplifications, we can use a model for data that is based on a multinomial likelihood function and its conjugate prior, Dirichlet prior distribution. For a change-point model with \( m + 1 \) segments, the resulting posterior has a following closed form after integration:

\[
F(x | \alpha) \propto \int_{\Theta} P(D | M, \Theta) P(\Theta | M, \Psi) d\Theta \propto \\
\prod_{c=1}^{m+1} \frac{N \Gamma \left( \sum_{i=1}^{l_c} \alpha_{cvi} \right)}{\Gamma \left( \sum_{i=1}^{l_c} x_{cvi} + \sum_{i=1}^{l_c} \alpha_{cvi} \right)} \prod_{i=1}^{l_c} \frac{\Gamma(x_{cvi} + \alpha_{cvi})}{\Gamma(\alpha_{cvi})} ,
\]

where \( x \) represents the observed data for data class \( i \) in dimension \( v \), and in the segment \( c \) segments (see 2.1 for further symbol definition). The obtained cluster specific prior to be somewhat objective, so that the user is not required to give additive information to the algorithm.

We tested various options for priors in our analysis, with the represented benchmarking system (see chapter 3 for a detailed description). The priors can be divided mainly to two categories: A) Priors with no information about the frequency of various classes in the dataset \( (\alpha_i = \alpha_j \text{ for all } i \text{ and } j) \) B) Priors that are being scaled according to the frequency of the class in the whole dataset \( (\alpha_i = ps \times P(D \in i) \text{ where } ps \text{ is the sum of the prior weights, also referred to as pseudo count, pre-observation information etc. and } P(D \in i) \text{ is the proportion of class } i \text{ in the whole data}) \). These priors have been used for example in text data mining [32]. They direct the results towards the average frequencies when we have a small number of posteriori (MAP) estimate with a flat prior, making the MAP solution the same as a maximum likelihood solution. We define the flat prior by dividing the probability one (proper prior) equally over all the possible splicing solutions that exist with the given size of the data and selected number of segments. The resulting prior is simple to calculate and is:

\[
P(\Psi | M) = \frac{1}{\binom{N-1}{m}},
\]

where \( N \) stands for the size of the dataset and \( m = \lvert \Psi \rvert \) stands for the number of change-points. Note that \( N - 1 \) represents the potential places for these change-points. The divisor is the number of unique solutions to segmentation of the dataset with a predefined number of change-points. Notice that if we have \( m = 0 \) and also if we have \( m = N - 1 \) the prior is 1. In both of these cases we have only one possible way for placing the required number of change points. The penalty given by this prior is maximal when the number of change-points \( m \sim (N - 1) / 2 \). This differs from BIC/AIC measures that automatically penalize the result more when the cluster number is larger.

From the previous equations 2 and 3, the final form of our criterion becomes:

\[
F(x | m) = \frac{1}{\binom{N-1}{m}} \times F(x | \alpha) ,
\]

The same prior has been used in the piecewise regression [51]. Also a similar model has been proposed earlier for a fully Bayesian analysis [12], where the model is evaluated over all potential segmentations with a selected number of change points. The only difference seems to be that the penalty for segmentation is \( (m!(N - m)!)/N! \) and not \( (m!(N - 1 - m)!)/(N -1)! \) (see [12] pp. 45), which does not go to 1 when we have a maximum number of change-points.

### 2.5 Dirichlet Prior Selection for Multinomial Model

We still have an open question: what prior to use with multinomial distributions in the segments? As we use the Bayes factors to compare various segmentation solutions, we need to have a proper prior. We also want the used prior to be somewhat objective, so that the user is not required to give additive information to the algorithm.

We tested various options for priors in our analysis, with the represented benchmarking system (see chapter 3 for a detailed description). The priors can be divided mainly to two categories: A) Priors with no information about the frequency of various classes in the dataset \( (\alpha_i = \alpha_j \text{ for all } i \text{ and } j) \) B) Priors that are being scaled according to the frequency of the class in the whole dataset \( (\alpha_i = ps \times P(D \in i) \text{ where } ps \text{ is the sum of the prior weights, also referred to as pseudo count, pre-observation information etc. and } P(D \in i) \text{ is the proportion of class } i \text{ in the whole data}) \). These priors have been used for example in text data mining [32]. They direct the results towards the average frequencies when we have a small number of
observations (see chapter 3. in [33]), instead of the equal probabilities. This seems more natural, than the equal prior probabilities.

Although the priors A and B define the relative sizes between the prior weights for each class, we still need to set up sum \( ps \), often requiring multiple prior sums for testing [36]. This tunes the bias vs. variance trade-off (see chapter 4 and app. B. in [33]). For priors A, we test \( ps = 1 \) and \( ps = I_c \). These are referred to as FLAT1 (also known as Perk's prior in the literature) and FLAT (popular flat prior), respectively. Earlier work ([12], [35]) found on similar Bayesian segmentation has used only the FLAT prior. For priors B, we first test \( ps = I_c \) and \( ps = I \), referred to as CSP (Class Specific Prior) and CSP1, respectively. Secondly, we test the square root of the segment size as the prior sum (leading to weights \( \alpha_i = \sqrt{n_c \times P(D \in i)} \), where \( n_c \) is the segment size) as proposed in [33]. This is referred to as Empirical Bayes Prior (EBP). It has been also used outside the pure Bayesian modeling as a pseudo count for Position Specific Scoring Matrices [34].

Our novel, further modification on the EBP distribution was that we used the \( \sqrt{P(D \in i)} \) instead of \( P(D \in i) \) in the EBP. This is referred to as Modified Empirical Bayes Prior (MEBP):

\[
\alpha_{MEBP} = \sqrt{n_c \times P(D \in i)}.
\] (5)

Our aim with this modification was to flatten the prior distribution by taking the square root of \( P(D \in i) \) and this way increase the size of the smallest prior weights. This was due to the unwanted behavior observed originally with the biological data we use in our application (see chapter 4) when segment size and probability of the class are small, causing the prior weight \( \alpha_i \) to approach zero (making the gamma function in the last division in equation 2 approach infinity).

Although originally designed to fit our analysis purposes, we still noticed that MEBP is linked to the Chi-square test, as the prior weight in MEBP is the square root of variance estimate used in the Chi square test. This makes the prior weight equal to the standard deviation estimate of the difference between the observed and expected number of class members. Although we evaluate this prior here with various datasets, we address the need for further analysis when using the prior for other multinomial data analysis tasks.

3 COMPARISON OF MODEL SELECTION METHODS

We constructed an evaluation framework for comparing various model selection methods. Altogether 9 different methods were compared. These included the three ML based measures: AIC, BIC and BIC2; and Bayesian methods with 6 different Dirichlet priors: CSP, CSP1, FLAT, FLAT1, EBP, and MEBP proposed in this paper. The framework has 5 adjacent steps, as shown in figure 2:

1. Building a data generating model (DGM).
2. Generation of artificial data from DGM.
3. Creating one solution for each number of changepoints with the TD method.
4. Selection of one solution using each of the compared model selection methods. Representation of each chosen solution as a Data Explaining Model (DEM).
5. Result evaluation by measuring similarity between DGM and each DEM.

Steps 1 and 2 are discussed in detail in section 3.1 and steps 4 and 5 in section 3.2. The results of method comparison are given in section 3.3.

3.1 Generation of Artificial Data

In order to compare the methods' performance, we produced artificial data with different parameter values. First, we varied the average magnitude of segments between large and small. This was motivated by our prior observations on poor performance of some methods with the biological data sets where the signal seemed to be scattered into rather small regions. Secondly, we tested a different number of parameters needed for data representation in segments, \(|\theta|\), by varying the number of classes in DGM. This was motivated by the fact that the penalization of ML based methods AIC and BIC is based on the assumption of independence of these parameters, as BF based methods do not use similar penalization. Also, the increase of data classes produces smaller class proportions in data which we observed as a possible difficulty with some Bayes priors. Third, we varied the size of the data as we wanted to observe the performance of data dependent Bayes priors when the prior information becomes more uncertain. To create class probabilities in DGMs, we used random numbers from a uniform distribution as weights for each data class.

According to the above criteria, we produced three different types of artificial data, each including three different configurations for the number of classes \((I_v = 2, I_v = 10 \text{ and } I_v = 30)\):

i. Several large segments. The segment number was chosen randomly between 1 and 10, and the segment sizes between 30 and 300.
ii. Few large segments. The segment number was chosen between 1 and 4, and the sizes between 30 and 100.
iii. Several small segments. Segment sizes were limited between 15 and 60 data points. The number of segments varied between 1 and 10.

For each of these 9 configurations, the whole process presented in figure 2 was repeated 100 times. In all configurations, data with 3-dimensions was used, although higher and lower (including one) numbers of dimensions were tested. We did not detect significant dependence between number of data dimensions and the performance of the methods (data not shown).
3.2 Evaluation of a Segmentation Solution

The first choice for evaluating an obtained segmentation would be to compare the number of segments in DGM with the number of segments in the obtained segmentation result. An even more detailed view could be obtained by comparing how well the segment borders or memberships of datapoints in DGM and the obtained segmentation solution correspond to each other. This has been performed using measures such as the Rand Index (RI) [37], Mutual Information (MI) [38], or several other methods (see for example [39]). The drawback of these methods is that they consider the solution defined by the borders of DGM as the optimal solution, omitting the uncertainty related to the data probabilities within the segments. With lesser data, with many very small segments, and/or when the neighboring segments are very similar, we actually hope to see a smaller number of segments than what is in the DGM (the Occam’s razor principle). Also, MI and related measures could not be used in a situation with one segment either in the DGM or in the obtained segmentation solution.

Instead of comparing borders or memberships, our evaluation starts with the comparison of DGM with the statistical model defined by an obtained segmentation solution. Such a model, referred to as DEM as mentioned above, is comprised of the probabilities of the various data classes (in each dimension) for each data-point in the obtained solution. The idea of comparison is to ask: How well the DEM, selected by each method, explains the future data that would come from the same DGM. We could address this by generating more test data sets from the same DGM, and then testing how well these new datasets are explained by the DEM, using cross-validation. If we were to repeat the generation of the test data infinite times, we would eventually approach the true distribution X:

\[ D_{KL} (X \| Y) = E_X \left( \log \left( \frac{X}{Y} \right) \right) \]

\[ = \sum_{i=1}^{N} \sum_{v=1}^{l} \sum_{j=1}^{k} P(X_i = i) \log \left( \frac{P(X_i = i)}{P(Y_i = i)} \right) \]  

where the calculation goes over each datapoint (t), dimension (v), and class (i). In our situation, it is natural to set X = DGM and Y = DEM. However, we found this problematic, as occasionally some of the segments in DEM had \( P(X = j) = 0 \). Within these segment areas, we would have \( D_{KL} = p(j) \times \log (p(j)/0) \) which is not solvable and approaches infinity. Typical ad hoc solutions to this problem would be to add a small value to all probabilities, or to fix the outcome of these cases to some constant. We considered this to be too significant a drawback. An alternative option would be to consider DEM as the true model and DGM as the approximate model (inverse \( D_{KL} \)). We found that this measure favoured too complex models (results not shown), as the distance is not increasing when \( P(X = j) = 0 \). Therefore, we decided to use a symmetrical version of \( D_{KL} \), Jensen-Shannon divergence [26]:

\[ D_{JS} (X \| Y) = \frac{1}{2} D_{KL} (X \| \frac{X+Y}{2}) + \frac{1}{2} D_{KL} (Y \| \frac{X+Y}{2}) \].  

Fig. 2. Benchmark for comparing the model selection methods. 1. A data generating model (DGM) is created. 2. Artificial data from DGM is generated. 3. TD-segmentation is performed for the data. 4a and 4b. One solution is chosen from the hierarchy using each model selection method (here only two, denoted with A and B, are shown). Each such solution is represented as a data explaining model (DEM). 5. The similarity between each DEM and the DGM is measured using the Jensen-Shannon divergence based measure. We use this measure with several different priors (see 3.2 for details), and at the end report a consensus of these measures. It should be noted that the developed MEBP is not included among these priors, so that the method evaluation would not favour that prior.
Currently, it seems to be a standard that the model evaluation uses the same prior information as the model optimization [31]. In our situation, the Bayes model selection optimizes the biased estimate (model with prior information), and this bias (prior) could be added to DEM as pseudo-counts in the $D_{JS}$ calculation. This would mean using a different prior in the evaluation of each Bayesian method. In turn, the ML-based methods (BIC and AIC) try to optimize the unbiased model, and thus the prior would not be used. In our opinion, using a different prior for evaluating each method would be a drawback, as it would lead to non-comparable measures. It should be more logical to use the same prior for evaluating each method. However, using a particular prior in evaluation might favor a method using the same or similar prior. Thus, we use a consensus of several different priors for the method evaluation. Among these, we exclude the developed MEBP prior. This way we are actually biasing the test slightly against MEBP. Thus, we use the following 6 commonly used priors: $\alpha_i = 0.1 P(d_j \in i) \times l_j$ (CSP), $\alpha_i = P(d_j \in i) \times \sqrt{l_j}$ (CSP5), $\alpha_i = P(d_j \in i) \times 2$ (CSP2), $\alpha_i = P(d_j \in i) \times \sqrt{n}$ (EBP), $\alpha_i = 0$ (NOP). The first 5 priors are actually the same or similar to those we use in the Bayesian model selection (CSP, EBP, FLAT). In turn, NOP corresponds to the ML model selection. Each prior is added in DEM to the probability of each data point $X_i$, in the following way:

$$P(X_i = j) = \frac{x_j + \alpha_i}{n + ps} \times \alpha_i,$$  

where $x_j$ represents the observed members of class $i$ for single DEM segment, and the prior sum $ps = \sum_j \alpha_j$. Note that $X$ represents only DEM here.

For each type of data and number of classes (9 all together), we report a difference in performance between our method and each competing method in segmentations of 100 replicated datasets as follows. First, for each dataset, we calculate the difference in $D_{JS}$ obtained with our model selection method (Bayes with MEBP prior) and each competing method. This is calculated using $D_{JS}$ score with each type of prior, producing 6 different difference measures. The same is repeated for each of the 100 replicated datasets. Secondly, we calculate a Z-score from the set of these 100 difference measures, one for each of the 6 priors used in the evaluation. A positive Z-score indicates that the developed Bayes method with MEBP prior performs better. A negative score indicates a better performance of the competing method. Next, we use this set of Z-scores as consensus for method performance by reporting its average (table 1) and percentiles (figure in supplementary file 2).

### 3.3 Results

The differences between ML-based methods and our Bayesian method are very strong: AIC and the two BIC methods show much weaker performance (table 1). These Z-scores would get extremely small p-values from Student or normal distribution. Indeed, AIC performs poorly with any 2 class data when BIC and BIC2 have quite competitive performance (with the exception of BIC2 in ii and BIC in iii). On the other hand, the BIC methods have poor performance with data containing 10 or 30 classes. This was expected, as these methods have a dependence on the number of classes in the data. The number of segments selected by each method (data not shown) shows the tendencies of AIC and BIC to select too complex and too simple models, respectively. Nonetheless, BIC2 performs better than BIC, especially with a larger number of classes. BIC2 with data i and 2 classes represents also only one case when the ML method has better performance over our method. However, the Z-score is quite insignificant in this case (Z-score = -0.8).

When compared to the other Bayes priors, the developed MEBP performs overall the best. The performance is the most equal when the data contains several large segments (data i). This is natural as the priors do not have so great an impact. The larger differences appear with the data ii and iii. The differences in prior performance become also more apparent with a larger number of classes as the prior weights get larger.

Among Bayes methods, the most similar results with the developed MEBP were given by EBP. This was expected as MEBP was derived from this prior. Also, with all 2-class datasets it has slightly better performance (Z-scores: -1.5, -0.8 and 0.9), making it the best performing prior in our comparison for 2-class datasets. Still, with a larger number of classes, where the priors have the clearest difference, MEBP starts to show much better performance. Especially with 30 classes, the Z-scores of EBP were 1.6, 3.2 and 3.8 with data i, ii and iii. This is due to its tendency to select a too complex model (data not shown). Thus, the clearest difference in the performance, in favor to our prior, is with small clusters and many classes. This supports the

### Table 1

<table>
<thead>
<tr>
<th>Data</th>
<th>AIC</th>
<th>BIC</th>
<th>BIC2</th>
<th>CSP</th>
<th>EBP</th>
<th>CSP1</th>
<th>Flat</th>
<th>Flat2</th>
</tr>
</thead>
<tbody>
<tr>
<td>i.</td>
<td>2</td>
<td>12.9</td>
<td>-0.8</td>
<td>-1.4</td>
<td>-1.5</td>
<td>-1.2</td>
<td>0.8</td>
<td>-1.4</td>
</tr>
<tr>
<td>i.</td>
<td>1.0</td>
<td>6.9</td>
<td>3.2</td>
<td>-0.7</td>
<td>-1.0</td>
<td>2.5</td>
<td>1.9</td>
<td>2.5</td>
</tr>
<tr>
<td>i.</td>
<td>1.5</td>
<td>14.5</td>
<td>9.8</td>
<td>3.0</td>
<td>1.6</td>
<td>3.8</td>
<td>0.7</td>
<td>3.3</td>
</tr>
<tr>
<td>ii.</td>
<td>2</td>
<td>10.3</td>
<td>0.8</td>
<td>1.4</td>
<td>0.0</td>
<td>-0.8</td>
<td>1.0</td>
<td>1.6</td>
</tr>
<tr>
<td>ii.</td>
<td>1.0</td>
<td>7.2</td>
<td>3.1</td>
<td>4.3</td>
<td>2.1</td>
<td>2.7</td>
<td>1.4</td>
<td>1.9</td>
</tr>
<tr>
<td>ii.</td>
<td>3.7</td>
<td>17.4</td>
<td>15.9</td>
<td>8.5</td>
<td>3.2</td>
<td>8.8</td>
<td>1.9</td>
<td>5.6</td>
</tr>
<tr>
<td>ii.</td>
<td>2</td>
<td>6.5</td>
<td>2.5</td>
<td>1.9</td>
<td>0.5</td>
<td>0.9</td>
<td>0.9</td>
<td>1.5</td>
</tr>
<tr>
<td>ii.</td>
<td>10</td>
<td>15.4</td>
<td>8.6</td>
<td>2.3</td>
<td>2.1</td>
<td>4.1</td>
<td>3.8</td>
<td>3.9</td>
</tr>
<tr>
<td>iii.</td>
<td>3.0</td>
<td>8.1</td>
<td>11.4</td>
<td>11.4</td>
<td>4.6</td>
<td>3.8</td>
<td>12.4</td>
<td>3.4</td>
</tr>
<tr>
<td>Average</td>
<td>4.9</td>
<td>8.6</td>
<td>6.1</td>
<td>2.3</td>
<td>1.2</td>
<td>3.9</td>
<td>1.6</td>
<td>9.7</td>
</tr>
</tbody>
</table>

Difference between the developed Bayes method with MEBP prior and each other compared model selection method (AIC, BIC, BIC2, CSP, EBP, CSP1, FLAT and FLAT2) in model selection performance. Each number indicates average of Z-scores of $D_{JS}$ differences (MEBP vs. a competing method) calculated using different priors added to DEM over 100 datasets of a specific type of data (column "Data" shows the data type, column "I" shows the number of classes). We highlight the results strongly in favor of our method (average of Z-scores ≥ 3.0) with bold font and results showing even slightest signal against our method (negative mean Z-score) with underlined font.
idea, given in chapter 2.5, that our prior is able to correct some of the instability with EBP, when the prior weights go very small.

The second most competitive results among Bayes methods were given by FLAT and CSP. With 10 and 30 classes, CSP gives worse results than MEBP, as it starts to select too complex models like EBP. However, this can be observed also with larger segments, and not only small segments, as with EBP. This is natural, as EBP is dependent on both the segment size and class proportion, whereas CSP depends only on the latter. Another observation is that the FLAT prior seems to perform much better than CSP with data ii. This is not surprising, as the reliability of prior information used by CSP (the class probabilities) is diminished due to the small data size. On the other hand, it can be seen that FLAT performs better than CSP also with large sized data (i), when there are 30 data classes. Therefore, the better performance could be also explained by the larger prior weights than in CSP.

From all Bayesian priors, the weakest performance was observed with the methods with the smaller prior sum (CSPI, FLATI). They were also exceptionally bad with data i where all others were the most equal. Still, they showed similar performance with other Bayes methods in 2 class datasets as the priors differ little with such data.

4 APPLICATION TO GENE EXPRESSION DATA

4.1 Data Representation

We analyzed the chromosomal locations of yeast genes classified to different groups according to gene expression levels during the cell cycle. Our aim here was to first create representative groups (clusters) for various expression profiles in the dataset, and then to see if they were over-represented in some potentially active genomic regions. We did not use the original measurements as most expression changes in the cell cycle dataset were quite small, with the signal being distributed to a large number of variables. Therefore we wanted to rather map the data with clustering to smaller dimensionality combining information on expression profiles over multiple dimensions. Note that methods like principal component analysis or non-negative matrix factorization could be used here. Similar work has been done also before in [6] but the emphasis was in the analysis of consecutive genes and in their similar expression.

The data contained 77 cell-cycle related DNA-microarray derived time points and other treatments published in [43]. 4017 of 6178 genes that had the lowest variance in expression levels were first separated as an unregulated set for the analysis. The rest of the genes were clustered according to expression levels by using k-means clustering using Euclidean distance (this would be probably similar to correlation as median centering normalization was used). Clustering was performed for several cluster numbers $k$ starting with $k=3$ and increasing to $k=6$. The basis of selecting such numbers was to obtain gene expression clusters corresponding to different phases of cell cycle $G_1$, $S$, $G_2$ and $M$, where phase $M$ can be further divided to mitosis and cytokinesis. Varying the number of clusters also aims to find consensus clusters that are repetitively observed over the changing clustering parameters (see [44] previous similar use of varying cluster number).

For each yeast chromosome, a dataset was created from the results of clustering in a similar way as explained in chapter 2.1 and shown in figure 1. Note that the actual distance in base pairs between the genes is omitted during this analysis as well as the length of each gene. This way a sequence of 6-dimensional (one dimension for each clustering result) multinomial observations with 4 to 7 classes (one class for each gene expression cluster and a class including the unregulated genes) for each dimension was created in order to find the chromosomal structure for cell-cycle related gene expression.

4.2 Results Evaluation

All 16 yeast chromosomes were analyzed. We further compared the selected segmentation solution for each chromosome against segmentation results with 100 different repeated randomly shuffled variations of the same data set (results shown in table 2). These results show that obtained results could not occur by random ordering in most of the chromosomes. We limited the further analysis to chromosomes 2, 3, 4, 6, 8, 10, 12 and 14 as they showed the best results over the randomized data sets. Furthermore, the associations between 800 genes and cell-cycle phases, obtained via detection of gene expression peaks during cell cycle time course in [43], were mapped over the obtained segments. Based on the mapping, we selected the chromosomes 2, 4, 6 and 12 for further analysis. We also validated some chosen segments in terms of the known interactions between yeast transcription regulators and genes in each segment. For this, we produced graph

<table>
<thead>
<tr>
<th>YEAST CHROMOSOME SEGMENTATION SCORES</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHR</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
</tbody>
</table>

Comparison of obtained log-BF scores ($\text{Log}_{10}\text{BF}$ column) for segmentations of yeast chromosomes against scores obtained from randomized data sets. *Rand. mean* is the average of the log$_{10}$BF for the segmentations from 100 runs with randomized chromosome. *Rand. std* is the standard deviation of log$_{10}$BF scores from the same runs. Z-scores and the corresponding logarithmic p-values (Log$_{10}$p) from normal distribution represent the significance of difference between real data and its randomizations.
visualizations showing the regulatory relationships using the YEASTRACT database tool [52]. These graphs are based on the evidence in scientific literature. Segmentation results, annotations and graph figures are available in supplementary files 3, 4, 5 and 6.

4.3 Biological Interpretation

Inspection of segmentation results shows that the method is able to find regions with strong coherence of data, i.e. similar gene expression clusters. Furthermore, we were able to find regions where a relationship to some phase or phases of cell cycle could be identified (marked with yellow in supplementary files 3, 4, 5 and 6). The cell cycle consists of four distinct phases: G1, S, G2 and M. The precise timing and coordination of available cellular components in each phase is essential since activation of each phase is dependent on the progression and completion of the previous phase. One cellular approach to efficient production of components is expansion of transcription units to local regions of the chromosome. Our results, shown using our approach, appears to support this idea. For example, closer inspection of chromosome IV segment number 6, region 152-164 (see table 3), showed a large number of G1 phase genes (7/13). This is very likely a transcriptional unit and suggests that other genes in the segment not previously attributed to G1 function (6/13), including hypothetical proteins, might actually be involved in this process. This is likely at least with gene YDL094C which is related to the same expression clusters with most of the G1 genes of the segment. Of the genes attributed to G1, nuclear proteins and DNA polymerase subunits are present as expected for this phase, but so are two protein O-mannosyltransferases whose function in the nucleus are not yet well understood. These enzymes may be involved in post translational modification of nuclear proteins. Nuclear proteins are known to be post translationally modified as part of a mechanism for regulating their activity. In the same chromosome, segment number 11, region 244-266, also show an exclusive region with G1 proteins. Their close proximity in the genome also suggests a coregulatory function.

In chromosome II, segment 1 showed 3/5 hypothetic G1 phase genes located together. Segment number 5, region 222-235, shows 3 G1 and then 3 G2/M genes which suggests an order of gene expression that coincides with the cell cycle.

In chromosome VI segment number 1, the localization of 6 hypothetical proteins 4 of which are with G1 annotation, is interesting as all 6 genes are hypothetical proteins with unknown functional annotation. Still all 6 genes, also the two with no mapping to cell cycle, are related to almost the same expression clusters. This might be another example of a transcriptional unit.

Regions of G1 specific genes could also be found in chromosome XII segment number 13, region 322-329 (5/8 genes). Another region in XII, segment number 23, region 570-584 shows enrichment of G1 genes (8/15) but with some mixture of other cell cycle phase genes. The large number of G1 genes found overall might indicate the need for larger transcriptional units to start the biosynthetic activities of the cell during this phase. Interestingly, also here a large number of these genes are hypothetical proteins. Many other G1, S, G2 and M proteins could be found in chromosome XII, but their order appears more random than the other chromosomes. This suggests that transcriptional sites on yeast chromosomes follow different orders depending upon the biological process. The genes within many segments that were found have poor or no annotation. These genes may indeed play a part in the cell cycle and the segmentation method demonstrating common chromosomal localization provides further evidence to support their role in the cell cycle.

Analysis of the previously chosen segments (marked with yellow in the supplementary files 3, 4, 5 and 6) in the light of transcription regulators with YEASTRACT database [52] shows that most of the genes within each segment are directly or indirectly regulated by the same proteins. For example, the four genes in segment number 1 in chromosome II are reported to be directly regulated by the same regulators ROX1, GAL4, YAP5, UGA3, GZF3 and GAT3. In turn, segment number 5 in the same chromosome contains mainly genes that can be divided into two sets by
regulators: the first genes are directly regulated by the transcription factor YAP1 (genes IST2, PHO3, PHO3, PBY1 and the gene VPS15). In turn, the other set of genes (genes RFC5, MMS4 and YBR089W) are regulated directly by MET4 transcription factor. Interestingly, MET4 protein is reported to interact with YAP1, and thus the first set of genes (regulated by YAP1) is also indirectly regulated by MET4. In chromosome IV, segment 6 contains genes regulated by the same regulators, but these relationships are indirect. In segment 11, in the same chromosome, the genes are regulated mainly in two separate groups by different regulators. The first group interacts with SWI4 and the second with CRZ1 transcription factor. In segment 18, four of the five genes are regulated directly by both FKH2 and MCM1 proteins. In segment 1 in chromosome VI all of the genes are regulated directly or indirectly by YAP1 and AFT1 transcription factors. In segment 13 of chromosome XII, the genes are mostly in two regulation groups, HSF1 regulating the first and SUM1 the second group. In segment 23, many of the genes are also reported to be regulated by the same transcription factors.

A detailed look in the segments discussed here shows that in a segment containing direct target genes of two different regulators in mixed order, each such a target gene group has a homogeneous co-expression cluster memberships, and the two target gene groups differ in these memberships (for example segment 5 in chromosome II, segment 11 in chromosome IV, and segment 13 in chromosome XII). Such segments, with a mixture of different expression clusters, might have been harder to detect with a method that aims at modelling the segment with a constant expression.

5 DISCUSSION AND CONCLUSIONS

Our work presents a fully Bayesian score function for segmentation, and proposes its usage for evaluating the results obtained from heuristic segmentation methods. We analyze the results obtained with TD segmentation and monitor how well the selected solution would predict the future observations from our artificial data generation. The obtained results from hierarchical segmentation fit to the model selection evaluation as they represent models with varying number of parameters. These varying solutions show when methods tend to over or under estimate the most optimal model. We show that the use of a Bayesian model clearly outperforms ML-based approximations like AIC and BIC. In addition, we show that the prior has a strong effect on the Bayesian score performance. Especially the priors with a prior sum one showed poor performance. We propose the MEBP that shows improved performance in multi-class datasets. Still, it was slightly outperformed in 2-class datasets by the EBP. This proposes the combined use of the two priors, where we would make the prior slowly turn from EBP to MEBP as the number of classes grows. As an added result, we observed an approximate order from underestimation to overestimation in model selection: BIC, BIC2, Bayesian models with larger prior sum, Bayesian models with smaller prior sum, AIC.

An earlier work on Bayesian modelling proposes quadratic time perfect simulation segmentation algorithm and its roughly linear time approximation [45, 49, 50]. The method uses a prior, where the number of changepoints is also allowed to vary and the position of the new changepoint is conditional on the position of the previous changepoint in a recursive analysis. This method also requires a definition of prior for segment sizes. In our case, we aim to evaluate the result from heuristics that has a fixed number of changepoints, and therefore there is no need to let the number of changepoints to vary. Our Bayesian model results in a simple equation that is hopefully easy to use. Therefore it could aid the users of heuristic methods by allowing a more exact selection of results from the large pool of solutions.

Other work with Bayesian modelling has also used more exotic measures, like extreme likelihood in [35], but we were not sure whether it fits to our model selection framework, due the disadvantages stated by the authors (overestimation of the probability of small segments, obtained score cannot be used as normalized probabilities).

Outside the Bayesian methods, segmentation uses stopping rules that are based either on BIC or Chi Square cumulative distribution. We did not test these here, but it should be noted that the represented testing framework actually allows comparison of all these measures.

The question of the prior with Bayesian model is sometimes brushed under the carpet. Still we show here that the various Dirichlet priors used in our work can change the obtained results dramatically. The current work represents, not just the comparison of various ML methods, but also the comparison of the Bayesian Model with various priors. We hope that this would become more of a standard, when representing new Bayesian models. In comparison to previous research (that has used a flat prior), we propose two priors that show in overall better performance. These priors do not require user input, and they are in that sense objective, although they use the information available on class probabilities in the whole data sequence. Also more advanced priors, like intrinsic priors [46], exist but they were not evaluated as they are significantly more complicated.

The strong point of the exact methods (like dynamic algorithm [11]) is that they calculate all the potential solutions with the selected number of changepoints. They are in that sense fully Bayesian, as they consider also the uncertainty, involved in the change-point parameters. Our method, only takes the given (potentially sub-optimal) solution, reminiscent of the Maximum A Posteriori result analysis. Still, the use of all models has been also criticized ([47]). Often in a set of models, there is only a small subset that fits to the given data well, whereas the rest do not fit at all. Therefore in complex analysis tasks, it might be more reasonable to spend CPU to analyze subsets from several types of model families rather than use it to analyze a single model family in detail. Indeed, here one could use the naïve hierarchical segmentation to obtain initial solutions for more complex and time consuming methods, like DP/DA, MCMC or expect maximization methods or use it to estimate some required parameters, like the number of changepoints. These are our main ideas for optimizing our
work in the future.

The evaluation was based on analysis of artificial datasets, and looking how well the produced clustering could predict future datasets. The drawback of this idea is that artificial data will not certainly include all the features of the real data. Still, the artificial datasets enable testing with varying parameter settings (like varying number of classes or clusters) creating datasets with variable difficulty. Also, the evaluation can be made more reliable as we can base the estimation of performance on the actual model that produced the dataset, instead of the small portion of data points as in cross validation. Furthermore, the solutions are automatically weighted according the Occam’s razor principle. As an example, the weaknesses of some tested priors might have easily gone unnoticed in a simpler test setting. Observed dependencies of the various methods on the test data complexity might explain some of the disagreements around the model selection methods. In the future, the testing situation could be made harder by using gradual changes in the probabilities around the change-point or allowing only part of the probabilities to change at the change-point, requiring changes only in the test data generation. A similar evaluation can be used when evaluating other data mining tools (clustering, classification, regression). Still, the use of prior in this evaluation process is somewhat an open question, although we found that most of the results did not depend on it.

The biological application represents a new approach to analysis of expression data on genomic sequence. We created the clustering and used them as the input to our algorithm, instead of feeding in classes (e.g. up-regulated, down-regulated and constant) created from each measurement for each gene. The idea of the clustering was to combine the weak cell cycle signal from various time points. Our analysis with expression data aims to find regions of activity within the genome during the cell cycle. Earlier, this question has been only analyzed by looking at consecutive genes and their similarity (Pearson correlation) in expression profiles [6]. Comparison to randomized datasets shows that random signal cannot explain many of the results. Also the mapping of cell cycle related genes (cluster) to the genome using our approach was found to combine them. The expression data in the cell cycle related genes corresponds well to the areas of coherent expression.

ACKNOWLEDGMENT

This work was supported in part by a grant from the Finnish Cultural Foundation and Finnish Academy project number 1102273. The authors wish to thank Prof. Lisa Holm for advice and Jussi Paananen for his comments. The authors also thank Prof. Nancy Zhang, Prof. David Siegmund and Dr. Wentian Li for their kind advice with BIC and modified BIC.

APPENDIX I: INFORMATION CRITERIONS

Information criterions AIC, BIC, and BIC2 are based on the penalization of log-ML with the term that is dependent on the number of parameters. In the segmentation model, parameter group $\Theta$ includes parameters associated with representation of the data within each segment. Group $\Psi$ includes parameters associated with division of data with change-points. The first group has $|\Theta| = (m+1) \times \sum_{i=1}^{m} (i-1)$ parameters, where $m$ is the number of change-points (and thus $m+1$ is the number of segments). The second group has simply as many parameters as there are change points in the model ($|\Psi| = m = \text{number of existing segments - 1}$). Note that in the first group $\sum_{i=1}^{m} (i-1)$ represents the number of parameters for a single segment and it has to be further multiplied with the number of segments $m+1$ to obtain the total number of these parameters.

AIC [24] approximates the Kullback-Leibler distance [14] between the model and assumed true model:

$$AIC = -2 \log(L) + 2(|\Theta| + |\Psi|).$$  \hspace{1cm} (A1)

Another common information criterion is BIC [25], which is an asymptotic approximation of BF:

$$BIC = -\log(L) + \frac{|\Theta| + |\Psi|}{2} \log(N).$$  \hspace{1cm} (A2)

BIC differentiates from AIC by penalizing the likelihood with a penalty term that is dependent on the size of the data. This causes more stringent penalizing with larger sample sizes.

In [23], a modified version of BIC is presented, referred here to as BIC2. This addresses particularly the model selection for change-point models. For multinomial data (pers. comm. Nancy Zhang):
BIC2 = \frac{-N}{2} \log(\hat{L}) + \frac{|\Theta| + |\Psi|}{2} \sum c \log(n_c) + \frac{1}{2} \log(N), \quad (A3)

where \( n_c \) is the size of obtained segment \( c \). Therefore each freely varying parameter penalizes likelihood with \( 1/2 \sum c \log(n_c) + 1/2 \log(N) \), instead of \( 1/2 \) shown in the original BIC.

REFERENCES


This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

Petri Pehkonen received the Ph.D and M.Sc. degrees in computer science from the University of Kuopio, Finland, in 2008 and 2003, respectively. He is currently a senior researcher in bioinformatics group and a teacher at the Biosciences Department at the University of Kuopio. His research interests include development and application of machine learning methods in bioinformatics field, and analysis and interpretation of data produced by high throughput technologies.


Petri Törönen received his Ph.D. in biochemistry, from University of Kuopio in 2004. He is currently a post-doctoral researcher at the University of Helsinki, Institute of Biotechnology. His research interests include mathematical modelling of large biological data sets, such as gene expression data, and developing novel approaches for interpretation of experimental data via its integration to biological knowledge.