Searching a multivariate partition space using weighted MAX-SAT

Silvia Liverani
Department of Statistics
University of Warwick
Coventry CV4 7AL, UK
s.liverani@warwick.ac.uk

James Cussens
York Centre for Complex Systems Analysis
& Department of Computer Science
University of York
York YO10 5DD, UK

Jim Q. Smith
Department of Statistics
University of Warwick
Coventry CV4 7AL, UK

Abstract

Because of the huge number of partitions of even a moderately sized dataset, even when Bayes factors have a closed form, a comprehensive search for the highest scoring (MAP) partition is usually impossible. Therefore, both deterministic or random search algorithms traverse only a small proportion of partitions of a large dataset. The main contribution of this paper is to encode the formal Bayes factor search on partitions as a weighted MAX-SAT problem and use well-known solvers for that problem to search for partitions. We demonstrate how, with the appropriate priors over the partition space, this method can be used to fully search the space of partitions in smaller problems and how it can be used to enhance the performance of more familiar algorithms in large problems. We illustrate our method on clustering of time-course microarray experiments.

1 Introduction

Many Bayesian model selection procedures are based on the posterior probability distribution over the appropriate model space. A very common method is MAP selection, where the most a posteriori probable model is selected (Heard et al., 2006). These selection techniques have the advantage that they incorporate scientific judgements (Liverani et al., 2008). However, a full exploration of the partition space is not possible when, as in our case, the number of elements is in the order of tens of thousands, even when using fast conjugate modelling. The number of partitions of a set of n elements grows quickly with n. For example, there are 5.1×10^{13} ways to partition 20 elements.

In this paper we demonstrate how to explore a partition space using weighted MAX-SAT. The SAT problem, which addresses whether a given set of propositional clauses is satisfiable, can be extended to the weighted MAX-SAT problem where weights are added to each clause and the goal is to find an assignment that maximizes the sum of the weights of satisfied clauses. This problem setting has been used by Cussens (2008) for model search over Bayesian networks, a class of models which shares some similarities with the search over partitions. For example, in both scenarios, models are scored using a marginal likelihood which is *local* in the sense of Liverani et al. (2008) and *decomposable* (see Section 2).

The advantage of algorithms encoding the weighted MAX-SAT methodology over many greedy search algorithms such as agglomerative hierarchical clustering (AHC) is that they are not intrinsically sequential. Under AHC once a decision to combine to clusters is made it cannot be reversed. This is not the case with weighted MAX-SAT solvers generally. In our illustrative examples this is a big advantage since under Bayes factor search via AHC early combinations of clusters are prone to be distorted by the presence of outliers (Smith et al., 2008). On the other hand the advantage weighted MAX-SAT has over random search algorithms is that it is typically more efficient and finds local maxima of the Bayes score function for sure in a sense explained later in the paper. Thus in small problems weighted MAX-SAT can be used to find an optimal partition for sure, whilst in large problems it can be used to enhance the performance of faster but less refined and adaptable algorithms.

Provided the appropriate local prior structure over the partition space is used a weighted MAX-SAT algorithm can be very flexible and can be used to search all spaces its competitors can. Here we will illustrate how this method can be used to cluster a class of time-course experiments known to exhibit circadian rhythms (Edwards et al., 2006).

The paper is organized as follows. In Section 2 we illustrate the model used to score partitions and review the current methods used to search the partition space. Section 3 describes how the search on the partition space is encoded as a weighted MAX-SAT problem. We discuss some examples in Section 4 and present ongoing work in Section 5.

2 Evaluating partitions

The main contribution of this paper is to encode the formal Bayes factor search on partitions as a weighted MAX-SAT problem and use well-known solvers for that problem to search over a multivariate partition space.

We use weighted MAX-SAT in conjunction with a conjugate Gaussian regression model developed by Heard et al. (2006). This model has a wide applicability because it can be customized through the choice of a given design matrix X. Conjugacy ensures the fast computation of scores for a given partition because these can be written explicitly and in closed form as functions of the data and the chosen values of the hyperparameters of the prior. Applications range from one-dimensional data points to multidimensional datasets with time dependence among points or where the points are obtained by applying different treatments to the units.

Let $\boldsymbol{Y}_i \in \mathbf{R}^r$ for $i = 1, \ldots, n$ represent the r-dimensional units to cluster. In our example in Section 4 these are log expressions of genes over r time points at which measurements are taken. Let $\boldsymbol{D} = (\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_n)$ and $\boldsymbol{Y} = \text{vec}(\boldsymbol{D})$ satisfy

$$\boldsymbol{Y} = X\boldsymbol{\beta} + \varepsilon$$

where $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots, \beta_p)' \in \mathbf{R}^p$ and $\varepsilon \sim N(\mathbf{0}, \sigma^2 I)$ is a vector of independent error terms with $\sigma^2 > 0$. The posterior Normal Inverse Gamma joint density of the parameters $(\boldsymbol{\beta}, \sigma^2)$ denoted by $NIG(\mathbf{0}, V, a, b)$, is given by

$$p(\boldsymbol{\beta}, \sigma^2 | \boldsymbol{y}) \propto (\sigma^2)^{-(a^* + p/2 + 1)} \times \\ \times \exp\left\{ -\frac{1}{2\sigma^2} \left[(\boldsymbol{\beta} - m^*)'(V^*)^{-1}(\boldsymbol{\beta} - m^*) + 2b^* \right] \right\}$$

with

$$\begin{aligned} m^* &= (V^{-1} + X'X)^{-1} X'\boldsymbol{Y} \\ V^* &= (V^{-1} + X'X)^{-1} \\ \gamma &= \{\boldsymbol{Y}'\boldsymbol{Y} - (m^*)'(V^*)^{-1} m^*\} \\ a^* &= a + rn/2, \quad b^* = b + \gamma/2 \end{aligned}$$

where $a, b > 0$ and V is a positive definite matrix. Throughout this paper we assume that $X = \mathbf{1}_n \otimes B$, where B is a known matrix, and that $X'X = nB'B$ is full rank. The design or basis function matrix B encodes the type of basis used for the clustering: linear splines in Heard et al. (2006), wavelets in Ray and Mallick (2006) or Fourier in Edwards et al. (2006). The latter is the most appropriate choice in the context of a study of daily rhythms as in Section 4.

The Bayes factor associated with this model can be calculated from its marginal likelihood $L(\boldsymbol{y})$; see for example Denison et al. (2002) and O'Hagan and Forster (2004). Thus

$$B = \left(\frac{1}{\pi}\right)^{nr/2} \frac{b^a}{(b^*)^{a^*}} \frac{|V^*|^{1/2}}{|V|^{1/2}} \frac{\Gamma(a^*)}{\Gamma(a)} \qquad (1)$$

Unlike for univariate data, within the class of problems illustrated in Section 4, there are a myriad of different shapes of expressions over time possible for each gene. Consequently, Bayes factors associated with different gene combinations are highly discriminative and informative.

Let $\mathcal{C}$ denote a partition belonging to the space of partitions $\mathbf{C}$, on a space Ω of cardinality n, and c a cluster of such partition. Heard et al. (2006) assume that each observation is exchangeable within its containing cluster. The Normal Inverse-Gamma conjugate Bayesian linear regression model for each observation in a cluster c takes the form

$$Y^{(c)} = X^{(c)}\boldsymbol{\beta}^{(c)} + \varepsilon^{(c)}$$

Here $\boldsymbol{\beta}^{(c)} = (\boldsymbol{\beta}_1^{(c)}, \ldots, \boldsymbol{\beta}_p^{(c)})$ is the vector of parameters with $p \leq r$, $X^{(c)}$ is the design matrix of size $n_c r \times p$, $\varepsilon^{(c)} \sim N(0, \sigma_c^2 I_{rn_c})$ where n_c is number of observations in cluster c and I_{rn_c} is the indicator function of size $rn_c \times rn_c$. A partition C of the observations divides into N clusters of sizes $\{n_1, \ldots, n_N\}$, with $n = \sum_{i=1}^N n_i$.

Assuming the parameters of different clusters are independent then, because the likelihood separates, it is straightforward to check (Smith et al., 2008) that the log marginal likelihood score $\Sigma(\mathcal{C})$ for any partition $\mathcal{C}$ with clusters $c \in \mathcal{C}$ is given by

$$\Sigma(\mathcal{C}) = \log p(\mathcal{C}) + \sum_{c \in \mathcal{C}} \log p_c(\boldsymbol{y}) \qquad (2)$$

Here the prior $p(\mathcal{C})$ is often chosen from the class of cohesion priors over the partition space (Quintana and Iglesias, 2003) which assigns weights to different models in a plausible and convenient way: see e.g. Smith et al. (2008).

An essential property of the search for MAP models - dramatically increasing the efficiency of the partition search - is that with the right family of priors the search is *local*. That is, if $\mathcal{C}^+$ and $\mathcal{C}^-$ differ only in the sense that the cluster $c^+ \in \mathcal{C}^+$ is split into two clusters $c_1^-, c_2^- \in \mathcal{C}^-$ then the log marginal likelihood score is a linear function only of the posterior cluster probabilities on c^+, c_1^- and c_2^-.

2.1 Choosing an appropriate prior over partitions

Although there are many possible choices for a prior over partitions, an appropriate choice in this scenario is the Crowley partition prior $p(\mathcal{C})$ (Crowley, 1997; McCullagh and Yang, 2006; Booth et al., 2008) for partition $\mathcal{C}$

$$p(\mathcal{C}) = \frac{\Gamma(\lambda)\lambda^N}{\Gamma(n+\lambda)} \prod_{i=1}^N \Gamma(n_i) \qquad (3)$$

where $\lambda > 0$ is the parameter of the partition prior, N is the number of clusters and n is the total number of observations, with n_i the number of observations in cluster c_i. This prior is *consistent* in the sense of McCullagh and Yang

(2006). The authors argue that this property is extremely desirable for any partition process to hold. Conveniently if we use a prior from this family then the score in (2) decomposes. Thus

$$\begin{aligned}\Sigma(\mathcal{C}) &= \log p(N, n_1, \ldots, n_N|\boldsymbol{y}) \\ &= \log p(N, n_1, \ldots, n_N) + \sum_{i=1}^{N} \log p(\boldsymbol{y}_i) \\ &= \log \Gamma(\lambda) - \log \Gamma(n+\lambda) + \sum_{i=1}^{N} S_i\end{aligned}$$

where

$$S_i = \log p(\boldsymbol{y}_i) + \log \Gamma(n_i) + \log \lambda$$

Thus, the score $\Sigma(\mathcal{C})$ is *decomposable* into the sum of the scores S_i over individual clusters plus a constant term. This is especially useful for weighted MAX-SAT which needs the score of an object to be expressible as a sum of component scores. The choice of the Crowley prior in (3) ensures that the score of a partition is expressible as a linear combination of scores associated with individual sets within the partition. It is this property that enables us to find straightforward encoding of the MAP search as a weighted MAX-SAT problem.

Note that a particular example of a Crowley prior is the Multinomial-Dirichlet prior used by Heard et al. (2006), where λ is set so that $\lambda \in (1/n, 1/2)$.

2.2 Searching the partition space

The simplest search method using the *local* property is agglomerative hierarchical clustering (AHC). It starts with all the observations in separate clusters, our original $\mathcal{C}_0$, and evaluates the score of this partition. Each cluster is then compared with all the other clusters and the two clusters which increase the log likelihood in (2) by the most are combined to produce a new partition $\mathcal{C}_1$. We now substitute $\mathcal{C}_1$ for $\mathcal{C}_0$ and repeat this procedure to obtain the next partition C_2. We continue in this way until we have evaluated the log marginal score $\Sigma(\mathcal{C}_i)$ for each partition $\{\mathcal{C}_i, 1 \leq i \leq n\}$. We then choose the partition which maximizes the score $\Sigma(\mathcal{C}_i)$.

A drawback of this method is that the set of partitions searched is an extremely small subset of the set of all partitions. The number of partitions of a set of elements n grows quickly with n. For example, there are 5.1×10^{13} ways to partition 20 elements, and the AHC evaluates only 1331 of them!

Despite searching only a small number of partitions, AHC is surprisingly powerful and often finds good partitions of clusters, especially when used for time-course profile clustering as in Section 4. It is also very fast. However one drawback is that the final choice of optimal partition is completely dependent on the early combinations of elements into clusters. This initial part of the combination process is subject to be sensitive and can make poor initial choices, especially in the presence of outliers or poor choices of hyperparameters when used with Bayes factor scores (see Section 4) in a way carefully described in Smith et al. (2008).

Analogous instabilities in search algorithms over similar model spaces have prompted some authors to develop algorithms that devote time to early refinement of the initial choices in the search (Chipman et al., 2002) or to propose alternative stochastic search (Lau and Green, 2007). The latter method appears very promising but is difficult to implement within our framework due to the size of the datasets.

We propose an enhancement of the widely used AHC with weighted MAX-SAT. This is simple to use in this context provided a prior such as (3) is used over the model space which admits a decomposable score. Weighted MAX-SAT is able to explore many more partitions and different regions of the partition space, as we will demonstrate in Section 4, and is not nearly as sensitive to the instabilities that AHC, used on its own, is prone to exhibit.

3 Encoding the clustering algorithm

Cussens (2008) showed that for the class of Bayesian networks a decomposition of the marginal likelihood score allowed weighted MAX-SAT algorithms to be used. The decomposition was in terms of child-parent configurations $p(x_i|\mathrm{Pa}_{x_i})$ associated with each random variable x_i in the Bayesian network. Here our partition space under the Crowley prior exhibits an analogous decomposition into cluster scores.

3.1 Weighted MAX-SAT encoding

For each considered cluster c_i, a propositional atom, also called c_i, is created. In what follows no distinction is made between clusters and the propositional atoms representing them. Propositional atoms are just binary variables with two values: TRUE and FALSE. A partition is represented by setting all of its clusters to TRUE and all other clusters to FALSE.

However, most truth-value assignments for the c_i do not correspond to a valid partition, and so such assignments must be ruled out by constraints represented by logical clauses. To rule out the inclusion of overlapping clusters we assert clauses of the form:

$$\overline{c_i} \vee \overline{c_j} \tag{4}$$

for all non-disjoint pairs of clusters c_i, c_j. (A bar over a formula represents negation.) Each such clause is logically

equivalent to $\overline{c_i \wedge c_j}$: both clusters cannot be included in a partition.

In general, it is also necessary to state that each data point must be included in some cluster in the partition. Let $\{c_{y_1}, c_{y_2}, \ldots, c_{y_{i(y)}}\}$ be the set of all clusters containing data point y. For each y a single clause of the form:

$$c_{y_1} \vee c_{y_2} \vee \cdots \vee c_{y_{i(y)}} \tag{5}$$

is created.

The 'hard' clauses in (4) and (5) suffice to rule out non-partitions; it remains to ensure that each partition has the right score. This can be done by exploiting the decomposability of the partition score into cluster scores and using 'soft' clauses to represent cluster scores. If S_i, the score for cluster c_i, is positive the following weighted clause is asserted:

$$S_i : c_i \tag{6}$$

Such a clause intuitively says: "We want c_i to be true (i.e. to be one of the clusters in the partition) and this preference has weight S_i." If a cluster c_j has a negative score S_j then this weighted clause is asserted:

$$-S_j : c_j \tag{7}$$

which states a preference for c_j not to be included in the partition. Given an input composed of the clauses in (4)–(7) the task of a weighted MAX-SAT solver is to find a truth assignment to the c_i which respects all hard clauses and maximizes the sum of the weights of satisfied soft clauses. Such an assignment will encode the highest scoring partition constructed from the given clusters.

Note that if a given cluster c_i can be partitioned into clusters $c_{i_1}, c_{i_2}, \ldots c_{i_{j(i)}}$ where $S_i < S_{i_1} + S_{i_2} + \cdots + S_{i_{j(i)}}$, then due to the decomposability of the partition score, c_i cannot be a member of any optimal partition: any partition with c_i can be improved by replacing c_i with $c_{i_1}, c_{i_2}, \ldots c_{i_{j(i)}}$. Removing such clusters prior to the logical encoding reduces the problem considerably and can be done reasonably quickly: for example, one particular collection of 1023 clusters which would have generated 495,285 clauses was reduced to 166 clusters with 13,158 clauses using this approach. The filtering process took 25 seconds using a Python script. This cluster reduction technique was used in addition to those mentioned in the sections immediately following.

3.2 Reducing the number of cluster scores

To use weighted MAX-SAT algorithms effectively in this context, the challenge in even moderately sized partition spaces is to identify promising clusters that might be components of an optimal partition. The method in Cussens (2008) of evaluating the scores only of subsets of less than a certain size is not ideal to this context since in our applications many good clusters appear to have a high cardinality.

However there are more promising techniques formulated in other contexts to address this issue. One of these, which we use in the illustrative example, is outlined below and others presented in Section 5.

Reduction by iterative augmentation

A simple way to reduce the number of potential cluster scores for weighted MAX-SAT is to evaluate all the possible clusters containing a single observation and to iteratively augment the size of the plausible clusters only if their score increases too, thanks to the nice decomposability of our score function. We will focus our discussion in this paper to an algorithm, the iterative augmentation algorithm described below.

Step 1 Compute the cluster score for all n observations as if each belonged to a different cluster. Save these scores as input for weighted MAX-SAT. Set $k \leftarrow 0$ and $c \leftarrow \emptyset$.

Step 2 Set $k \leftarrow k+1$, $j \leftarrow k+1$ and $c \leftarrow \{k\}$. Exit the algorithm when $k = n$.

Step 3 Add element j to cluster c and compute the score for this new cluster c'. If $S_{c'} > S_c + S_j$, then

- Save the score for cluster c'
- If $j = n$, go to Step 2.
- $c \leftarrow c'$ and $j \leftarrow j+1$
- Go to Step 3

else

- If $j = n$, go to Step 2.
- Set $j \leftarrow j+1$
- Go to Step 2.

The main advantage of this algorithm is that it evaluates the actual cluster scores, never approximating them by pairwise dissimilarities or in any other way. Furthermore, this method does not put any restriction on the maximum size of the potential clusters.

Hybrid AHC algorithm

Even though this algorithm performs extremely well when the number of clustered units $n < 100$, it slows down quickly as the number of observational vectors increases. However this deficiency disappears if we use it in conjunction with the popular AHC search to refine clusters of less than 100 units. When used to compare partitions of profiles as described in Section 2, AHC performs extremely well when the combined clusters are large. So to improve

its performance we use weighted MAX-SAT to reduce dependence on poor initialization. By running a mixture of AHC together with weighted MAX-SAT we are able to reduce the dependence whilst retaining the speed of AHC and its efficacy with large clusters. AHC is used to initialize a candidate partition. Then weighted MAX-SAT is used as a 'split' move to refine these clusters and find a new and improved partition on which to start a new AHC algorithm. The hybrid algorithm is described below.

Step 1 Initialize by running AHC to find best scoring partition $\mathcal{C}_1$ on this search.

Step 2 *(Splitting step)* Take each cluster c in $\mathcal{C}_1$. Score promising subsets of c and run a weighted MAX-SAT solver to find the highest scoring partition of c. Note that, because our clusters are usually several orders of magnitude smaller than the whole set, this step will be feasible at least for interesting clusters.

Step 3 Substitute all the best sub-clusters of each cluster c in $\mathcal{C}_1$ to form next partition $\mathcal{C}_2$.

Step 4 If $\mathcal{C}_1 = \mathcal{C}_2$ (i.e. if the best sub-cluster for each cluster in $\mathcal{C}_1$ is the cluster itself) then stop.

Step 5 *(Combining step)* If this is not the case then by the linearity of the score $\mathcal{C}_2$ must be higher scoring than $\mathcal{C}_1$. Now take $\mathcal{C}_2$ and - beginning with this starting partition to test combinations of clusters in $\mathcal{C}_2$ - using AHC. (Note we could alternatively use weighted MAX-SAT here as well). This step may combine together spuriously clustered observations that initially appeared in different clusters of $\mathcal{C}_1$ and were thrown out of these clusters in the first weighted MAX-SAT step. Find the optimal partition $\mathcal{C}_3$ doing this.

Step 6 If $\mathcal{C}_3 = \mathcal{C}_2$ stop, otherwise go to Step 2.

This hybrid algorithm obviously performs at least as well as AHC and is able to undo any early erroneous combination of AHC. The shortcomings of AHC, discussed in Smith et al. (2008), are overcome by checking each cluster running weighted MAX-SAT to identify outliers. Note that the method is fast because weighted MAX-SAT is only run on subsets of small cardinalities. We note that at least in the applications that we have encountered most clusters of interest appear to contain less than a hundred units.

4 A Simple Example

We will illustrate the implementation of weighted MAX-SAT for clustering problems in comparison to and in conjunction to the widely used AHC.

Here we demonstrate that weighted MAX-SAT can be used to cluster time-course gene expression data. The cluster scores are computed in C++, on the lines of the algorithm by Heard et al. (2006) and it includes the modifications suggested by Smith et al. (2008) and Anderson et al. (2006). All runs of weighted MAX-SAT were conducted using the C implementation available from the UBCSAT home page http://www.satlib.org/ubcsat. UBCSAT (Tompkins and Hoos, 2005) is an implementation and experimentation environment for Stochastic Local Search (SLS) algorithms for SAT and MAX-SAT. We have used their implementation of WalkSat in this paper.

4.1 Data

Our algorithm will be illustrated by an example on a recent microarray experiment on the plant model organism *Arabidopsis thaliana*. This experiment was designed to detect genes whose expression levels, and hence functionality, might be connected with circadian rhythms. The aim is to identify the genes (of order 1,000) which may be connected with the circadian clock of the plant. A full analysis and exposition of this data, together with a discussion of its biological significance is given in Edwards et al. (2006).

We will illustrate our algorithms on genes selected from this experiment. The gene expression of $n = 22,810$ genes was measured at $r = 13$ time points over two days by Affymetrix microarrays. Constant white light was shone on the plants for 26 hours before the first microarray was taken, with samples every four hours. The light remained on for the rest of the time course. Thus, there are two cycles of data for each of the *Arabidopsis* microarray chip. Subjective dawn occurs at about the 24th and 48th hours – this is when the plant has been trained to expect light after 12 hours of darkness.

4.2 Hybrid AHC using weighted MAX-SAT

Although our clustering algorithms apply to a huge space of over 22,000 gene profiles, to illustrate the efficacy of our hybrid method it is sufficient to show results on a small subset of the genes: here a proxy for two clusters. Thus we will illustrate how our hybrid algorithm can outperform AHC and how it rectifies partitions containing genes clustered spuriously in an initial step. In the example below we have therefore selected 15 circadian genes from the dataset above and contaminated these with 3 outliers that we generated artificially.

We set the parameters $v = 10$, $a = 0.001$, $b = 0.001$ and $\lambda = 0.5$ and ran AHC which obtained the partition formed by 2 clusters shown in Figure 1. AHC is partially successful: the 15 circadian genes have been clustered together, and so have the 3 outliers. The latter cluster is a typical example of misclassification in the sense of Smith et al. (2008) in that it is rather coarse with a relatively high associated variance. The score for this partition is $\Sigma(\mathcal{C}_{\text{AHC}}) = 64.89565$.

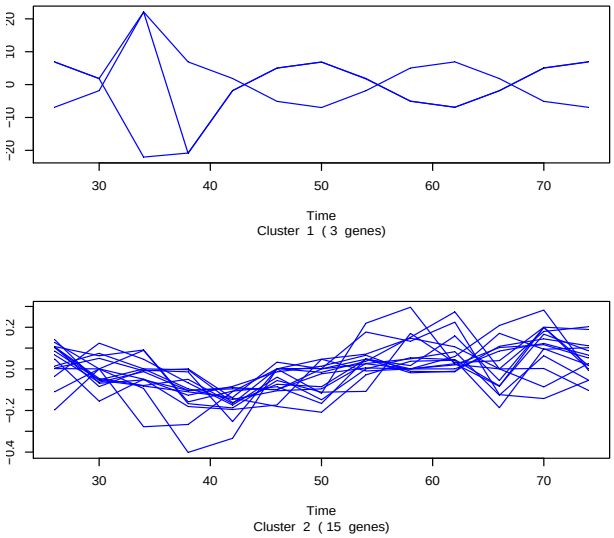

Figure 1: Clusters obtained on 18 genes of *Arabidopsis thaliana* using AHC ($\Sigma(\mathcal{C}_{\text{AHC}}) = 64.89565$). The y-axis is the log of gene expression. Note the different y-axis scale for the two clusters.

Following the hybrid AHC algorithm we then ran MAX-SAT on both the clusters obtained by AHC. The clusters obtained are shown in Figures 2 and 3. Both the clusters obtained by AHC have been split up by MAX-SAT. The score of the partition formed by these 5 clusters, including the constants, is now $\Sigma(\mathcal{C}_{\text{MAX-SAT}}) = 79.43005$. This is the log of the marginal likelihood and taking the appropriate exponential, in terms of Bayes factor, this represents a decisive improvement for our model. Note that the increase in the log marginal likelihood is supported also by the visual display. The outliers are very different between themselves and from the real data and it seems reasonable that each one would generate a better cluster on its own - note the different scale of the y-axis. The other 15 genes have a more similar shape and it seems visually reasonable to cluster them together, as AHC does initially, but MAX-SAT is able to identify a more subtle difference between 2 shapes contained in that cluster. It was not necessary in our case to run AHC again to combine clusters, given the nature of our data. A single iteration of the loop described in our hybrid algorithm identified the useful refinement of the original partition.

This example shows how, as discussed in Smith et al. (2008), AHC can be unstable especially when dealing with outliers at an early stage in the clustering. The weighted MAX-SAT is helpful to refine the algorithm, and obtain a higher scoring partition.

It is clear that in larger examples involving thousands of genes the improvements above add up over all moderate sized clusters of an initial partition, by simply using weighted MAX-SAT over each cluster in the partition, as described in our algorithm and illustrated above.

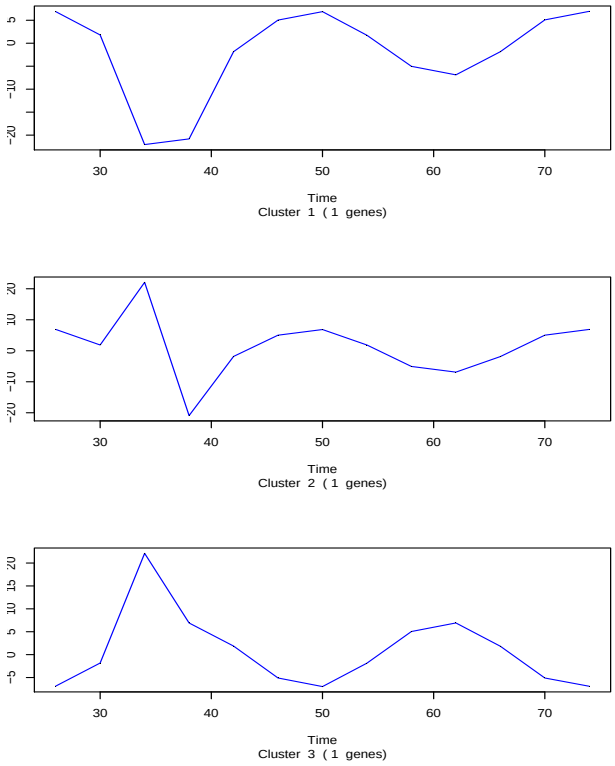

Figure 2: Clusters obtained on 3 outliers of *Arabidopsis thaliana* using AHC (1 cluster, $S_1 = -156.706$) and weighted MAX-SAT (3 cluster, $S_1 = -145.571$).

5 Further work on cluster scores for large clusters

In the approach taken in this paper clusters are explicitly represented as propositional atoms in the weighted MAX-SAT encoding and so it is important to reduce the number of clusters considered as much as possible. The hybrid method with iterative augmentation that we have described in Section 3.2 works very efficiently for splitting clusters with cardinality smaller than 100. However it slows down dramatically for greater cardinalities. It would be useful to generalize the approach so that it can also be employed to split up larger clusters. The main challenge here is to identify good candidate sets. Two methods that we are currently investigating are outlined below.

Reducing cluster scores using cliques

One promising method for identifying candidate clusters is to use a graphical approach based on pairwise proximity between the clustered units. Ben-Dor et al. (1999) - a well known and highly cited paper - proposes the CAST algorithm to identify the clique graph which is closest to the graph obtained from the proximity matrix. A graph is called a clique graph if it is a disjoint union of complete graphs. The disjoint cliques obtained by the CAST algorithm define the partition.

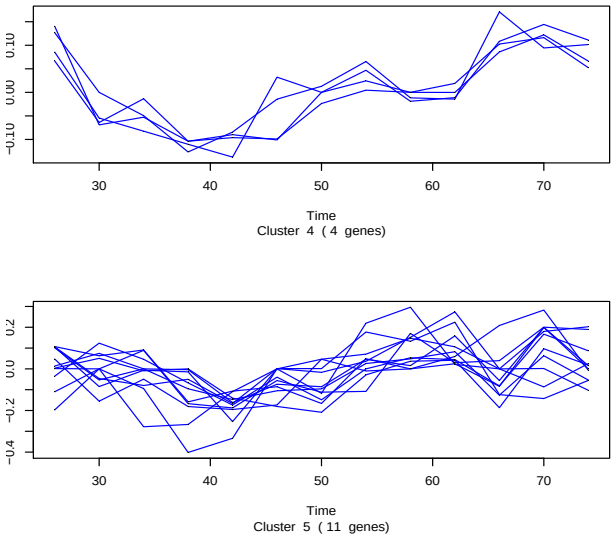

Figure 3: Clusters obtained on 15 genes of *Arabidopsis thaliana* using AHC (1 cluster, $S_2 = 255.973$) and weighted MAX-SAT (2 clusters, $S_2 = 259.372$).

We suggest using an approach similar to Ben-Dor et al. (1999), enhanced by the use of weighted MAX-SAT and a fully Bayesian model.

We focus on maximal cliques, instead of clique graphs as in Ben-Dor et al. (1999), to identify possible clusters to feed into weighted MAX-SAT. A maximal clique is a set of vertices that induces a complete subgraph, and that is not a subset of the vertices of any larger complete subgraph. The idea is to create an undirected graph based on the adjacency matrix obtained by scoring each pair of observations as a possible cluster and then use the maximal cliques of this graph to find plausible clusters. It is reasonable to assume that a group of elements is really close and should belong to the same cluster when it forms a clique. This considerably reduces the number of clusters that need to be evaluated and are the input for weighted MAX-SAT, which will then identify the highest scoring partition.

The first step is to calculate the proximity between observations i and j ($i, j = 1, \ldots, n$) such as

$$D = \{d_{ij}\} = S_{ij} - (S_i + S_j)$$

which gives a matrix of adjacencies A

$$A = \{a_{ij}\} = \begin{cases} 1 & \text{if } d_{ij} > K \\ 0 & \text{otherwise} \end{cases}$$

from which we can draw a graph (S_{ij} is the score for the cluster of 2 elements, i and j). Each vertex represents an observation. Two vertices are connected by an edge according to matrix D. The adjacency matrix defines an undirected graph. The maximal cliques, the intersections between maximal cliques and the union of maximal cliques with common elements define the potential cluster scores for weighted MAX-SAT.

Although such methods are deficient in the sense that they use only pairwise relationships within putative clusters, they identify potentially high scoring clusters quickly. Of course, it does not matter whether some of these clusters turn out to be low scoring within this candidate set, because each is subsequently fully scored for weighted MAX-SAT and their deficiency identified. This is in contrast to the method of Ben-Dor et al. (1999) which is completely based on pairwise dissimilarities. So the only difficulty with this approach is induced by those clusters which are actually high scoring but nevertheless are not identified as promising.

Other advantages of this method are that all the scores that are calculated are used as weights in the weighted MAX-SAT and it does not induce any artificial constraint on cluster cardinalities.

Reducing cluster scores by approximating

An alternative to the method described above is to represent the equivalence relation given by a partition directly: for each distinct pair of data points y_i, y_j, an atom $a_{i,j}$ would be created to mean that these two data points are in the same cluster. Only $O(n^2)$ such atoms are needed. Hard clauses ($O(n^3)$ of them) expressing the transitivity of the equivalence relation would have to be added. With this approach it might be possible to indirectly include information on cluster scores by *approximating* cluster scores by a quadratic function of the data points in it. A second-order Taylor approximation is an obvious choice. Such an approach would be improved by using a different approximating function for each cluster size.

6 Discussion

WalkMaxSat appears to be a promising algorithm for enhancing partition search. It looks especially useful to embellish other methods such as AHC to explore regions around the AHC optimal partition and to find close partitions with better explanatory power. We demonstrated above that this technique can enhance performance on small subsets of the data and on large datasets too, in conjunction with AHC.

Although we have not tested this algorithm in the following regard, the algorithm can also be used as a useful exhaustive local check of a MAP partition found by numerical search (Lau and Green, 2007). Also, note that weighted MAX-SAT can be used not just for MAP identification, but also by following the adaptation suggested by Cussens (2008) in model averaging, using to identify all models that are good.

There are many embellishments of the types of methods described above that will potential further improve our hybrid search algorithm. However, in this paper we have demonstrated that in circumstances where the Crowley priors are appropriate weighted MAX-SAT solvers can provide a very helpful addition to the tool box of methods for MAP search over a partition space.

References

Anderson, P. E., J. Q. Smith, K. D. Edwards, and A. J. Millar (2006). Guided conjugate Bayesian clustering for uncovering rhytmically expressed genes. *CRiSM Working Paper* (07).

Ben-Dor, A., R. Shamir, and Z. Yakhini (1999). Clustering gene expression patterns. *Journal of Computational Biology 6*(3–4), 281–297.

Booth, J. G., G. Casella, and J. P. Hobert (2008). Clustering using objective functions and stochastic search. *Journal of the Royal Statistical Society, Series B 70*(1), 119–139.

Chipman, H. A., E. I. George, and R. E. McCulloch (2002). Bayesian treed models. *Machine Learning 48*(1–3), 299–320.

Crowley, E. M. (1997). Product partition models for normal means. *Journal of the American Statistical Association 92*(437), 192–198.

Cussens, J. (2008). Bayesian network learning by compiling to weighted MAX-SAT. In *Proceedings of the 24th Conference on Uncertainty in Artificial Intelligence (UAI 2008)*.

Denison, D. G. T., C. C. Holmes, B. K. Mallick, and A. F. M. Smith (2002). *Bayesian Methods for Nonlinear Classification and Regression*. Wiley Series in Probability and Statistics. John Wiley and Sons.

Edwards, K. D., P. E. Anderson, A. Hall, N. S. Salathia, J. C. W. Locke, J. R. Lynn, M. Straume, J. Q. Smith, and A. J. Millar (2006). FLOWERING LOCUS C Mediates Natural Variation in the High-Temperature Response of the *Arabidopsis* Circadian Clock. *The Plant Cell 18*, 639–650.

Heard, N. A., C. C. Holmes, and D. A. Stephens (2006). A quantitative study of gene regulation involved in the immune response of anopheline mosquitoes: An application of Bayesian hierarchical clustering of curves. *Journal of the American Statistical Association 101*(473), 18–29.

Lau, J. W. and P. J. Green (2007). Bayesian Model-Based Clustering Procedures. *Journal of Computational and Graphical Statistics 16*(3), 526.

Liverani, S., P. E. Anderson, K. D. Edwards, A. J. Millar, and J. Q. Smith (2008). Efficient Utility-based Clustering over High Dimensional Partition Spaces. *CRiSM Research Report* (22).

McCullagh, P. and J. Yang (2006). Stochastic classification models. In *Proceedings International Congress of Mathematicians*, Volume III, pp. 669–686.

O'Hagan, A. and J. Forster (2004). *Bayesian Inference: Kendall's Advanced Theory of Statistics* (Second ed.). Arnold.

Quintana, F. A. and P. L. Iglesias (2003). Bayesian clustering and product partition models. *Journal of the Royal Statistical Society Series B 65*(2), 557–574.

Ray, S. and B. Mallick (2006). Functional clustering by Bayesian wavelet methods. *J. Royal Statist. Soc.: Series B 68*(2), 305–332.

Smith, J. Q., P. E. Anderson, and S. Liverani (2008). Separation measures and the geometry of Bayes factor selection for classification. *Journal of the Royal Statistical Society, Series B 70*(5), 957–980.

Tompkins, D. A. D. and H. H. Hoos (2005). UBCSAT: An implementation and experimentation environment for SLS algorithms for SAT and MAX-SAT. In H. H. Hoos and D. G. Mitchell (Eds.), *Theory and Applications of Satisfiability Testing: Revised Selected Papers of the Seventh International Conference (SAT 2004, Vancouver, BC, Canada, May 10–13, 2004)*, Volume 3542 of *Lecture Notes in Computer Science*, Berlin, Germany, pp. 306–320. Springer Verlag.