

Appendix A

Proof of Theorem 1

The proof of Theorem 1 consists of two main steps. First, we show that there must be an optimal solution to equation 8 with at least as many 1s as the solution found by our algorithm. This is clearly a necessary condition for Theorem 1 to hold. Next we show that, for any possible solution found by our algorithm, there will be one such optimal solution which will indeed agree with our solution on all its 1s. This proves the theorem. The first step is proven in proposition 5 below, which requires supporting results that we prove first. The theorem is then proved in the sequence.

Lemma 3 *Let $Y_k^* = \operatorname{argmax}_y y^T A^{[k],n} y$, where $A^{[k],n}$ is such that $\operatorname{diag}(A^{[k],n}) := \operatorname{diag}(A) - \frac{2y^n}{k + \|y^n\|^2}$ and $A_{ij} = \langle \theta_{ij}, C_{ij} \rangle$ for $i \neq j$, with $\theta_{ij} \geq 0$ and $C_{ij} \geq 0$. Now let $l \geq k$ and define similarly Y_l^* . Then for every $y_k^* \in Y_k^*$ there exists a $y_l^* \in Y_l^*$ such that $y_k^* \odot y_l^* = y_k^*$ (and, conversely, for every $y_l^* \in Y_l^*$ there exists a $y_k^* \in Y_k^*$ such that $y_k^* \odot y_l^* = y_k^*$).*

Proof The claim states that for any optimal solution $y_k^* \in Y_k^*$, its 1s will also be present in some optimal solution $y_l^* \in Y_l^*$. The proof is by contradiction, assuming that there exists $y_k^* \in Y_k^*$ and an index i such that $y_k^*(i) = 1$ and $y_l^*(i) = 0$ for all $y_l^* \in Y_l^*$. Now consider the binary vector z_l which agrees with y_l^* everywhere except in i , i.e. $z_l(j) = y_l^*(j)$ for $j \neq i$ and $z_l(i) = 1$. This implies that $z_l^T A^{[l],n} z_l = y_l^{*T} A^{[l],n} y_l^* + A_{ii}^{[l],n} + \sum_{j:j \neq i} A_{ij}^{[l],n}$. Now note that we necessarily have $\sum_{j:j \neq i} A_{ij}^{[l],n} + A_{ii}^{[l],n} \geq 0$. This holds because, first, $\sum_{j:j \neq i} A_{ij}^{[k],n} + A_{ii}^{[k],n} \geq 0$ (otherwise y_k^* would not be optimal), second, $A_{ii}^{[l],n} \geq A_{ii}^{[k],n}$ and third that $A_{ij}^{[l],n} = A_{ij}^{[k],n}$ for $i \neq j$. Therefore $z_l^T A^{[l],n} z_l \geq y_l^{*T} A^{[l],n} y_l^*$, which implies that $z_l \in Y_l^*$, which contradicts the assumption that for all $y_l^* \in Y_l^*$, $y_l^*(i) = 0$. The converse is proved analogously. ■

Intuitively, the above result says that due to the non-negativity of the off-diagonal elements of A , the new objective function arisen from an increase in the diagonal of A surely has some maximiser which includes all 1s already present in any maximiser of the previous objective function, prior to the increase in the diagonal.

Corollary 4 *Let $l \geq k$. Then $\max_y y^T A^{[l],n} y \geq \max_y y^T A^{[k],n} y$.*

Proof Note that the set of 1s potentially present in a y_l^* but not in a y_k^* for which $y_l^* \odot y_k^* = y_k^*$ necessarily has non-negative contribution to the objective function $y^T A^{[l],n} y$ (otherwise y_l^* would not be optimal), jointly with the fact that the same is true for the set of 1s present both in y_l^* and y_k^* since $A_{ii}^{[l],n} \geq A_{ii}^{[k],n}$ for any i . ■

We are now ready to prove that there exists an optimal solution to the constraint generation problem in equation 8 which contains at least k_{max} 1s.

Proposition 5 *Let $k' \geq k_{max}$, where k_{max} is as instantiated in Algorithm 2. Then there exists an optimal solution $y^{*n} \in \operatorname{argmax}_y y^T A^{n}(y) y$ such that for some k' , we have $y^{*n} \in \operatorname{argmax}_{y \in \mathcal{Y}_{k'}} y^T A^{[k'],n} y$.*

Proof This follows from equation 9 and from the claim that for all $k \leq k_{max}$, $\max_{y \in \mathcal{Y}_{k_{max}}} y^T A^{[k_{max}],n} y \geq \max_{y \in \mathcal{Y}_k} y^T A^{[k],n} y$, which we now prove. We know that $\max_y y^T A^{[k_{max}],n} y = \max_{y \in \mathcal{Y}_{k_{max}}} y^T A^{[k_{max}],n} y$ holds since, from lines 8 and 10 of Algorithm 2, we have $k_{max} = |y_{k_{max}}^{*n}|$. On the other hand, we have $\max_y y^T A^{[k],n} y \geq \max_{y \in \mathcal{Y}_k} y^T A^{[k],n} y$ since $\mathcal{Y}_k \subseteq \mathcal{Y}$. Both facts, when put together with corollary 4, prove the claim. ■

We are now able to give a proof of Theorem 1.

Proof of Theorem 1 We show that for any solution $y_{k_{max}}^{*n}$ found by Algorithm 2, we have that $y_{k_{max}}^{*n} \odot y^{*n} = y_{k_{max}}^{*n}$ for some optimal solution y^{*n} having at least k_{max} 1s. We proceed by contradiction, assuming that there is no optimal solution y^{*n} respecting $|y^{*n}| \geq k_{max}$ such that $y_{k_{max}}^{*n} \odot y^{*n} = y_{k_{max}}^{*n}$ holds, for any $y_{k_{max}}^{*n}$. This is equivalent to saying that for every y^{*n} respecting $|y^{*n}| \geq k_{max}$ there is an index i (which can be different for different y^{*n}) such that $y^{*n}(i) = 0$ and $y_{k_{max}}^{*n}(i) = 1$ for any $y_{k_{max}}^{*n}$. Now consider a vector z that agrees with y^{*n} everywhere except in index i , i.e., $z(i) = 1$. The key observation now is that $z^T A^{[|z|],n} z \geq (y^{*n})^T A^{[|y^{*n}|],n} y^{*n}$, and therefore z should be an optimal solution as well, which results in a contradiction. To see why this inequality holds, first note that $z^T A^{[|z|],n} z - (y^{*n})^T A^{[|y^{*n}|],n} y^{*n} = \sum_i (A_{ii}^{[|z|],n} - A_{ii}^{[|y^{*n}|],n}) + \sum_{j \neq i} A_{ij}$, where the first sum accounts for the potential change in the diagonal due to the increase in the cardinality of the solution, and the second sum accounts for the newly incorporated off-diagonal terms as a result of $z(i) = 1$. The result then follows from the submodularity assumption ($A_{ij} \geq 0, \forall i \neq j$) and from the fact that the diagonal is non-decreasing with respect to increases in the cardinality of the solution ($A_{ii}^{[|z|],n} - A_{ii}^{[|y^{*n}|],n} \geq 0, \forall i$, since $|z| > |y^{*n}|$). ■

Proof of Theorem 2 The theorem results directly from the fact that inequality 14 characterises the condition when the algorithm fails, as well as from the fact that $\max_{\alpha} \beta_{A,\alpha} \geq \beta_{A,\alpha}$ and $\epsilon_{k_{max}}^V |y^n| \geq \gamma_{\alpha}$. ■

Appendix B

Training time

The time the algorithm takes to train depends on the average time per iteration and on the number of iterations.

At each iteration run time is dominated by the N calls to the constraint generation algorithm, where each one may require at most V calls to the max-flow algorithm. The computational complexity of the whole learning algorithm is therefore $O(iNV^4)$, where i is the number of iterations. In practice, however, it is much faster than that.

In Table 2 we summarise training times along the variables mentioned above, for the experiments with all the features (rightmost bars in Figure 1). Note that our implementation is multi-threaded, and scales almost linearly with the number of available cpus, so wall-clock times can be much smaller.

Table 2: Training times

Dataset	Training time	i	N	V	Time/call to Alg. 2
Yeast	47.8 cpu-seconds	205	1500	14	155 us
Enron	847.3 cpu-seconds	116	1123	53	6504 us

In comparison to RML[11], our method is 5.0 times slower in the Yeast dataset and 8.4 times slower in the Enron dataset. Since it is a strict generalisation of RML, slower runtimes are expected.

Appendix C

Adding positivity constraint to BMRM

BMRM ([18]) approximates the solution to an optimisation problem of the form

$$\underset{\theta}{\text{minimize}} \left[R_{emp}(\theta) + \frac{\lambda}{2} \|\theta\|^2 \right] \quad (16a)$$

$$\text{where } R_{emp}(\theta) = \frac{1}{N} \sum_{n=1}^N l(x_n, y_n, \theta) \quad (16b)$$

by iteratively solving a linear program arising from a lower bound based on a first-order Taylor approximation of l . Denoting $a_{i+1} := \partial_{\theta} R_{emp}(\theta_i)$ and $b_{i+1} := R_{emp}(\theta_i) - \langle a_{i+1}, \theta_i \rangle$, where i corresponds to the iteration number, the problem that is solved is

$$\underset{\theta, \xi}{\text{minimize}} \frac{\lambda}{2} \|\theta\|^2 + \xi \quad (17a)$$

$$\text{subject to } \langle a_j, \theta \rangle + b_j \leq \xi \quad \text{for all } j \leq i \quad \text{and} \quad \xi \geq 0. \quad (17b)$$

This is done by, at each iteration, solving the dual of eq. (17):

$$\underset{\alpha}{\text{maximize}} -\frac{1}{2\lambda} \alpha^T A^T A \alpha + \alpha^T b \quad (18a)$$

$$\text{s.t. } \mathbf{1}^T \alpha \leq 0, \quad \alpha \geq \mathbf{0} \quad (18b)$$

where A denotes the matrix $[a_1 a_2 \dots a_i]$, b the vector $[b_1 b_2 \dots b_i]^T$ and $\theta = -\frac{1}{\lambda} A \alpha$.

Enforcing positivity in θ^2 amounts to adding the additional constraint $\theta^2 \geq \mathbf{0}$ to eq. (17), and the corresponding dual problem now becomes:

$$\underset{\alpha, \gamma}{\text{maximize}} -\frac{1}{2\lambda} (\alpha^T A^T A \alpha + \gamma^T \gamma - 2\alpha^T A_1^T \gamma) + \alpha^T b \quad (19a)$$

$$\text{s.t. } \mathbf{1}^T \alpha \leq 0, \quad \alpha \geq \mathbf{0}, \quad \gamma \geq \mathbf{0} \quad (19b)$$

where A_1 is the subset of the rows of A that correspond to the gradient w.r.t. θ^2 .