Permutation Relaxation for Tumour Deconvolution with Prior Proportion Information

Christopher A. Cremer
1001140650

***Abstract*—Sequenced tumour sample data is used to make predictions about patient prognosis. The accuracy of the predictions are often stifled by the heterogeneous nature of the tumour samples. In an attempt to address this problem, tumour samples can be deconvoluted computationally. With the advent of tumour evolution modelling, we now have access to new information that can be used to improve deconvolution. This information provides the number of different cell populations and their proportions within each sample. Since the assignment of these proportions to specific cell populations is not known, we are faced with the combinatorial optimization problem of finding the optimal permutation that minimizes the residual error. Here, I show that this problem can be approximated by a relaxation to a convex problem and fitting to the nearest permutation.**

***Index Terms* — Convex Optimization, Deconvolution, Permutation Relaxation**

I. Introduction

Tumors are heterogeneous, which means that individual tumour samples are mixtures of different cell populations. This heterogeneity arises from contamination by non-cancerous tissue, as well as different cancerous subpopulations. This heterogeneity in gene expression profiles interferes in the classification of cancer subtype, and it may hinder our ability to predict patient prognosis and treatment response. Therefore, being able to accurately deconvolve tumour gene expression profiles is important for biomarker detection and precision medicine.

Tumour heterogeneity can be modelled as each sample being composed of a convex combination of hidden 'pure' profiles. This linear system can be expressed in matrix notation as:

$$X = WZ$$

where X is the gene expressions of the observed samples, W is the proportion of each component in each sample, and Z is the expression profiles of the hidden components. Thus X is N samples by D genes, W is N samples by K components, and Z is K components by D genes. We can assume that D > N > K. Given the gene expressions of a set of samples X, the challenge is to solve for W and Z. The constraints for this problem are that each element of W must be between zero and one since they are proportions. The elements of Z must be non-negative because they represent gene expressions, which cannot be below zero. A problem like this can be approached with non-negative matrix factorization (NMF) [1] but with the added constraint that each weight vector needs to sum to one. Therefore we are solving a convex optimization problem of this form:

$$\begin{gathered} min_{W,Z} \ \|X - WZ\| \\ s.t. W_{ik}, Z_{kj} \geq 0 \\ 1^T W_i = 1 \end{gathered}$$

for i=1…N, k=1…K, and j=1…D. The norm that is minimized is often chosen to be the L2 norm due to its simplicity and its efficient computational properties.

There are numerous computational tools available for the deconvolution of genomic data from heterogeneous samples. The method developed by Newman et al. [2] can be used for sample deconvolution when the profiles of the hidden cell types (Z) are known. Erkkilä et al. [3] formalized a probabilistic model for deconvolution when we are provided with an estimate for the proportions (W).

Due to recent work in the analysis of DNA sequencing data, specifically subclonal reconstruction [4], we can obtain information about the genetically distinct populations within each sample. The information we obtain is an estimate on the number of subpopulations and their proportions within each sample. We hypothesize that these cancerous subpopulations differ not only in their genetic mutations but also in their population gene expression profiles. Thus we can use these proportion values to push the deconvolution towards meaningful latent factors. Consequently, for each sample i, we have a set S_i of values that correspond to the weights of each W_i vector.

However one challenge that we are faced with is assigning the proportions to each of the hidden profiles. In other words, we know the number of hidden profiles and their proportions within each sample but we don't know to which hidden profile each proportion refers to. The set S_i is padded with zeros so that its size is equal to the number of components K. Therefore we must find the permutation P_i of set S_i that minimizes the L2 norm. Thus we are looking for the permutation vector that minimizes,

$$min_{P_i} \|Z^T P_i - X_i\|$$

This is a combinatorial optimization problem. The most obvious method to solve this would be to perform an exhaustive search over all permutations of S_i. The running time would be K permute Q (KpQ), where K is the number of components and Q is the number of non-zeros in the set S_i. This could be feasible for small values of K, however, it is unclear how many subpopulations actually exist within different types of cancers, and thus we cannot use the exhaustive search since it would limit our ability to model larger numbers of components.

To reduce the computational burden of combinatorial problems, there is occasionally relaxations that can be made to the original problem so that it becomes a convex optimization problem. For instance, semidefinite programming (SDP) has wide applicability in combinatorial optimization. A number of NP−hard combinatorial optimization problems have convex relaxations that are SDPs [5]. Another example is decoding linear error correcting codes which have been shown to be solved by linear programming (LP) [6]. The problem can be relaxed so that the feasible region becomes the convex hull of all d-dimensional binary vectors with an even number of 1s [7]. Therefore, it is possible that a relaxation can be made for my problem so that we are not restricting the feasible region to be a member of the permutation set. It is not clear whether a relaxation will yield results similar to the exhaustive search. In this report, I devise a simple relaxation of the combinatorial optimization problem so that it becomes a convex optimization problem and compare it to the exhaustive search. In the Methods section, I describe the relaxation method and how it will be tested. Next, in the Results section we see how the relaxation performs when recovering latent weights and when performing a deconvolution. Finally, in the Discussion section I explain the benefit of the prior values.

II. Methods

A. Problem Relaxation

Instead of restricting the solution to be a permutation of S_i, we will project X_i onto Z^T resulting in W_i. After the projection, we will fit the W_i to the nearest permutation vector. The feasible space is constrained by the inequality of non-negativity and the affine equality of summing to one. The projection can be formulated as the following quadratic program (QP),

$$min_{W_i} \|Z^T W_i - X_i\|_2$$
$$s.t. W_{ik} \geq 0,\ 1^T W_i = 1$$

This projection step ignores the prior set of values. The second step is to fit W_i to the permutations of S_i so that it minimizes the same objective function. In other words, we are going to swap the values of W_i with those of S_i in such a way that minimizes$\|Z^T W_P - X_i\|_2$, where W_P is the new vector composed of only the values of S_i and zeroes so that the length of W_P is the same length as W_i. We don't need to concern ourselves with the constraints because the set S_i already satisfies them. This problem is similar to a water-filling optimization problem where we are constrained to only use the values given by S_i.

To fit S_i to W_i, first we sort the set S_i. Next, the largest value of W_i is replaced with the largest value of S_i, and the second largest value of W_i is replaced with the second largest value of S_i, and so on. I claim that this procedure will result in the optimal vector. See the Discussion section for a proof that this fitting procedure is optimal in this situation.

B. Simulated Data

In order to obtain results that reflect the reality of gene expression deconvolution, I created artificial samples in-silico by mixing RNASeq data of breast invasive carcinomas from the Cancer Genome Atlas (TCGA). More specifically, a weight vector for each sample is made by random sampling from a uniform distribution then normalized so that they sum to one. The components matrix Z is created by selecting TCGA samples. Finally, X is created by the dot product of W and Z plus the addition of noise. Thus we know the identity of the weights and profiles that went into making each sample of X. Noise was introduced to the gene expressions (X_{ij}) by adding values randomly sampled from the distribution N(0, σ_j^2*noise), with σ_j^2 equal to the variance of gene j.

For the experiments in the Results section, I created 50 samples with 1000 genes each. Accordingly, the dimensions of the X matrix were 50 by 1000. In order to make the weight vectors more realistic, I added a sparsity parameter. This parameter was set so that each weight vector has on average four non-zero values.

C. Deconvolution Procedure

When only the X matrix is given, we need to solve for both the Z and W matrices. First the Z matrix is randomly initialized with random samples. Next, we alternate between solving W then Z by minimizing the L2 norm while staying within the constraints. If there are no prior values provided, the deconvolution stops here. If there is a set of prior values to fit, the prior values are fitted to the weight vectors using either the relaxation or the exhaustive search. The alternation between solving and fitting W then solving Z is repeated until some tolerance is reached. In the end, we have factorized the X matrix into two lower dimensional matrices W and Z and we have used the prior values to modify W so that the Z matrix is more accurate.

D. Implementation

The implementation of the projection was done using the Python package CVXOPT, which can be found at the following url: http://cvxopt.org/. From the package I used the quadratic cone program solver to solve my optimization problem with its constraints.

III. Results

I tested the relaxation in two scenarios. First, given X and Z, I want to see how well the predictied weights W_P match the real weights W_R. In this case, we can measure the performance of the various methods based on the difference between the two matrices. Thus the error is defined as $\|W_P - W_R\|_2$. This result is then divided by the number of samples so that it's invariant to the number of samples in the experiment. The second scenario is a full deconvolution. Here, only X is given and we need to solve for W and Z. Again we measure the performance based on the method's ability to recover the latent matrices, so $\|W_P - W_R\|_2$ and $\|Z_P - Z_R\|_2$ are the measurements of error. For each experiment, I averaged over 10 iterations.

A. *Error vs Noise*

In this first experiment, I examined the relationship between the amount of noise added to the data X and the performance of the models. Given X and Z, the models solve for W. It's conceivable that the relaxation may work when the problem is easy, but fail when there is a large amount of noise. Fig. 1 is a plot of the performance of the different methods with increasing noise. I compared the projection, the projection followed by the fitting of the prior values, and the exhaustive search. I set the number of compoents to five so that the exhasutive search could run in a reasonalbe amount of time. The plot shows that the projection with the fit is nearly exaclty overlapping the exhaustive search, whereas the projection without the fit has higher error.

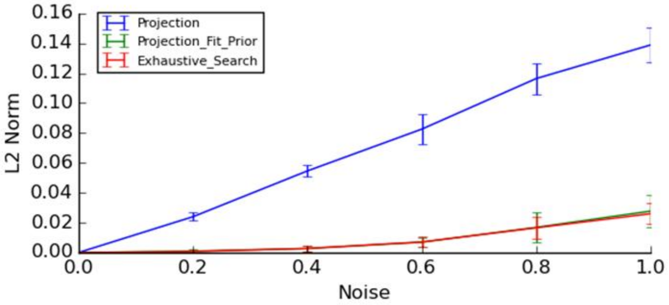

Fig. 1. L2 norm of the difference between the predicted weights and the real weights ($\|W_P - W_R\|_2$) vs amount of noise in the data. Given X and Z, the models are solving for W.

In Fig. 1 the average L2 norm ranges from 0. to 0.15. This does not seem like a significant amount of error until we see the difference between the real and predicted vectors. The following are some examples of different amounts of error of vectors of length five.

L2 norm of 0.05:
Real weights: [0.00, 0.17, 0.15, 0.35, 0.33]
Predicted weights: [0.01, 0.21, 0.14, 0.32, 0.34]
L2 norm of 0.10:
Real weights: [0.15, 0.53, 0.11, 0.19, 0.00]
Predicted weights: [0.09, 0.51, 0.10, 0.25, 0.05]
L2 norm of 0.15:
Real weights: [0.00, 0.00, 0.44, 0.11, 0.45]
Predicted weights: [0.00, 0.11, 0.45, 0.00, 0.44]

These examples demonstrate that even with small amounts of error, there are cases where the real weights are zero but the prediction is non-zero. This is significant because in cancer the presence of a subpopulation, even if it's small, can make a large difference in prognosis.

B. *Error vs Number of Components*

The purpose of doing this relaxation is to allow for the model to accommodate more components. The addition of components to the model is equivalent to assuming that there exists more possible subpopulations within a single cancer type. Thus, I'm interested in examining how the model behaves with an increasing number of components. Given X and Z, the models solve for W. In Fig. 2, I test the performance of the methods with 3, 4, and 5 components, since the exhaustive search limits the number of components to 5.

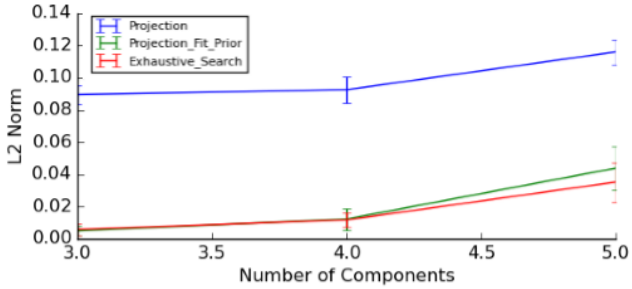

Fig. 2. L2 norm of the difference between the predicted weights and the real weights ($\|W_P - W_R\|_2$) vs the number of components in the data. Given X and Z, the models are solving for W.

We observe that from three to five components, the projection with the fit performs equally well to the exhaustive search. This result indicates that the relaxation is obtaining a similar prediction vector W_P as the exhaustive search. In comparison, the projection without the fit has greater error for all numbers of components. The error of all methods increases with the number of components because it adds more variables to learn and therefore increases the complexity of the problem. Now, given that the projection with fit performs nearly as well as the exhaustive search, I would like to explore how the projection and projection with fit perform with even more components. Fig. 3 plots the L2 norm of the two methods from 5 components to 20. As expected, both increase with the number of components. The fitting of the prior set achieves lower error, however, the difference between the two methods stays constant with the increasing number of components.

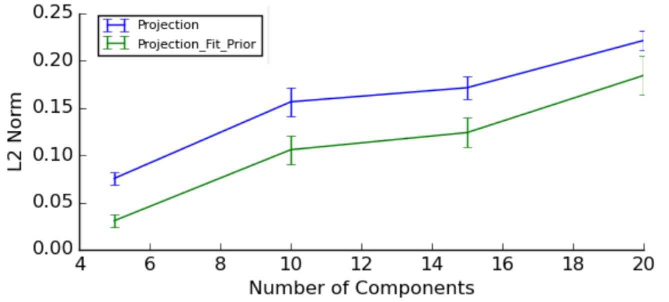

Fig. 3. L2 norm of the predicted weights vs the real weights ($\|W_P - W_R\|_2$) vs the number of components of the model. Given X and Z, the models are solving for W.

C. *Deconvolution*

We’ve seen that the projection with fit performs better than the projection alone and it performs just as well as the exhaustive search up to 5 components. Now we would like to test the full deconvolution process. Given only the gene expressions X, we would like to recover the profiles of the hidden components Z and the proportions of each of the components within each sample W.

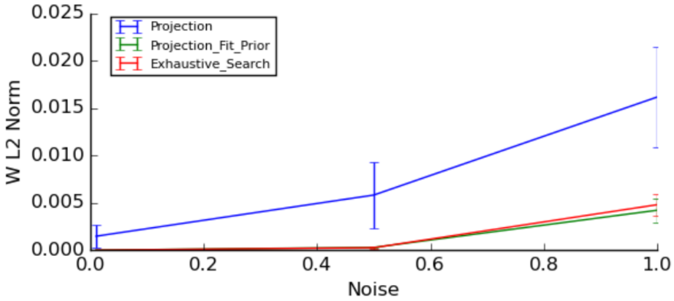

Fig 4. L2 norm of the difference between the predicted weights and the real weights ($||W_P - W_R||_2$) vs the amount of noise in the data. Given X, the models are solving for W and Z.

Fig. 4 is similar to Fig. 1, as it shows the error in predicting W versus the amount of noise added. The plot of Fig. 4 shows that the projection with the fit is nearly exaclty overlapping the exhaustive search, whereas the projection without the fit has higher error at all levels of noise.

The ability for the deconvolution to recover the expression values of the hidden components Z is plotted in Fig 5. Just like the error in W, the erorr in Z is very similar for the projection with the fit and the exhasutive seach, whereas the projection alone is higher. This is a strong indication that the relaxation is accurate enough to perform as well as the exaustive search.

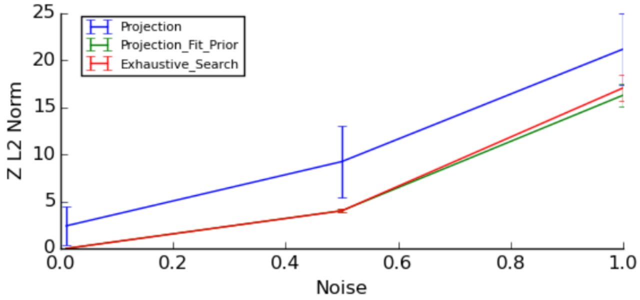

Fig 5. L2 norm of the difference between the predicted components and the real components ($||Z_P - Z_R||_2$) vs the amount of noise in the data. Given X, the models are solving for W and Z.

IV. Discussion

A. *Prior Acts as a Regularization*

When we compare the projection with and without the fitting of the prior values, the main benefit that the fitting has is that acts as a type of regularization. The prior values provides not only the magnitude of the weights, but also the number of non-zero weights. In the cancer deconvolution setting, the weight vectors are sparse, so having the prior values constrains the solution to be sparse as well. That is the main reason we see a constant difference between the two methods.

In other applications, the L1 weight regularization is used to push the models towards more sparse solutions. In this scenario, the L1 regularization would not provide any benefit because the solutions are constrained to sum to one, thus the L1 norm of the weights is equal for all feasible solutions.

B. *Proof of Fitting Optimality*

After the projection of X_i onto Z^T, we end up with the vector W_i. The next step is to fit the set of values S_i to W_i. See Methods for how the fitting is done. The optimal ordering of W_P is defined as the permutation of S_i (with zeros added so that the length of W_P is equal to the number of components) that minimizes $||Z^T W_P - X_i||_2$. Thus we are searching for the ordering of W_P that is closest to W_i, ie. $||W_P - W_R||_2$.

I claim that the fitting method of swapping the values in order of magnitude obtains the optimal ordering. This may be seem obvious, nonetheless, I will prove it in order to remove any doubt. In order to prove that it is optimal, I will show that any other ordering is sub-optimal. Let W_i have values i and j where $i \geq j$ and let S_i have values k and l where $k \geq l$. Consequently, using my method of matching, i will be matched with k and j will be matched with g. The alternative is that i is matched with l and j is matched with k. I need to show that the first ordering is always better than or equal to the second. This can be defined as follows,

$$||i-k|| + ||j-l|| \leq ||i-l|| + ||j-k||.$$

For our application we use the L2-norm, however this proof does apply to all norms. There are two general cases in which we need to consider. The first case is where the second ordering has one norm that is greater than both norms of the first ordering. The second case is where the each norm of the second ordering is individually greater that one of the norms of the first ordering. To demonstrate the first case, let us assume that $k \geq i \geq l \geq j$. Then the proof is as follows,

$$\begin{aligned} ||i-k|| + ||j-l|| &\leq ||j-k|| \\ &\leq ||j-k|| + ||i-l||. \end{aligned}$$

The first inequality stems from the assumption $k \geq i \geq l \geq j$. The second inequality comes from the fact that all norms are positive. To demonstrate the second case, let us assume that $k \geq i \geq j \geq l$. Then the proof is as follows,

$$\begin{aligned} ||i-k|| + ||j-l|| &\leq ||j-k|| + ||j-l|| \\ &\leq ||j-k|| + ||i-l||. \end{aligned}$$

The first inequality is explained by the assumption $k \geq i \geq j$ so that $||i-k|| \leq ||j-k||$ and the second inequality is explained by $i \geq j \geq l$ so that $||j-l|| \leq ||i-l||$. All other orderings of $i, j, k,$ and l follow very similar proofs as the ones above. Either the second ordering has one norm that is greater than both norms of the first ordering or each norm of the second ordering is individually greater that one of the norms of the first ordering. Thus, though it may have been obvious to some, I have shown that swapping the values in order of magnitude obtains the optimal ordering.

V. Conclusion

Deconvolution with priors that are not matched to their components introduces a combinatorial optimization problem. The goal of this project was to compare the exhaustive search method versus relaxing the problem to a simple projection and fit to find the optimal permutation. The results demonstrate that the relaxation performs nearly as well as the exhaustive search. Therefore, in order to allow the model to accommodate more components, it is beneficial to use the relaxation instead of the exhaustive search.

References

[1] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization.," *Nature*, vol. 401, no. October 1999, pp. 788–91, 1999.

[2] A. M. Newman, C. L. Liu, M. R. Green, A. J. Gentles, W. Feng, Y. Xu, C. D. Hoang, M. Diehn, and A. a Alizadeh, "Robust enumeration of cell subsets from tissue expression profiles," *Nat. Methods*, vol. 12, no. 5, pp. 1–10, 2015.

[3] T. Erkkilä, S. Lehmusvaara, P. Ruusuvuori, T. Visakorpi, I. Shmulevich, and H. Lähdesmäki, "Probabilistic analysis of gene expression measurements from heterogeneous tissues," *Bioinformatics*, vol. 26, no. 20, pp. 2571–2577, 2010.

[4] A. G. Deshwar, S. Vembu, C. K. Yung, G. H. Jang, L. Stein, and Q. Morris, "Reconstructing subclonal composition and evolution from whole genome sequencing of tumors," *Genome Biol.*, pp. 0–29, 2014.

[5] M. X. Goemans, "Semidefinite Programming and Combinatorial Optimization," *Doc. Math.*, vol. Extra Volu, pp. 657–666, 1998.

[6] J. Feldman, M. J. Wainwright, and D. R. Karger, "Using linear programming to decode binary linear codes," *IEEE Trans. Inf. Theory*, vol. 51, no. 3, pp. 954–972, 2005.

[7] S. Barman, X. Liu, S. Draper, and B. Recht, "Decomposition methods for large scale LP decoding," *2011 49th Annu. Allert. Conf. Commun. Control. Comput.*, vol. 59, pp. 253–260, 2011.