Scaled matrix completion and cell deconvolution with NanoString data

Yichen Zhang

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science

University of Washington

2018

Committee:

Noah Simon

Ali Shojaie

Program Authorized to Offer Degree:
Biostatistics

University of Washington

Abstract

Scaled matrix completion and cell deconvolution with NanoString data

Yichen Zhang

Chair of the Supervisory Committee:
Noah Simon
Department of Biostatistics

This thesis explores two research problems in Chapters 1 and 2. Chapter 1 combines pivotal penalized estimation, with matrix completion, to introduce a new matrix completion problem, where the optimal tuning parameter does not depend on the variance of the noise. We consider this new "scaled matrix completion" problem, and compare it to standard matrix completion problem. Chapter 2 develops a new cell deconvolution method for data from NanoString Technologies nCounter platform, a relatively new sequencing platform. We assess the performance of this cell deconvolution method with simulated data and experimental data.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

GLOSSARY

C: expression reference matrix used in the cell deconvolution algorithm.

CV: coefficient of variation.

MC: matrix completion.

MD: matrix decomposition.

MSE: mean square error.

MDSE: median square error.

NTC: nominal target concentration.

SNR: signal-to-noise ratio.

SVD: singular value decomposition.

$||X||_*$: nuclear norm of X; $||X||_* := \sum_i^n \sigma_i(X)$.

$\sigma_I(X)$: i-th largest singular value of X.

λ: tuning parameter.

ACKNOWLEDGMENTS

Firstly, none of these work would have been possible without the excellent tutelage of Noah Simon. Noah introduced me to the beautiful field of machine learning. Within this area, I had the distinct pleasure to explore penalized estimation, matrix completion problems, and connections between the two. Noah has also been a great advisor and friend. He has been helping me grow professionally and personally in the past two years.

Thank you to my colleagues at NanoString Technologies, especially Afshin Mashadi-Hossein and Patrick Danaher, for offering me the opportunity to explore novel cell deconvolution methods with NanoString data, and providing valuable advice in the development of the method.

A special thank you to Gitana Garafallo for many relaxed but insightful conversations that makes winters in Seattle a bit sunnier.

Lastly, I am fortunate to have very supportive parents and wife. The trust, patience and love they have given me is remarkable.

INTRODUCTION

0.1 Relation between Chapter 1 and Chapter 2

Both chapters 1 and 2 deal with matrix decomposition (MD) problems. The key idea of MD is that there exists latent structures in the data, and that by uncovering that structure one can obtain a compressed representation of the data. In chapter 1, we consider matrix completion, which is a MD problem with low-rank structure: Here, one assumes that the mean of a data matrix can be written as the outer product of an (a priori unknown) matrix with few columns and a matrix with few rows. In chapter 2, we consider decomposing a gene expression matrix (or vector) of a mixture of cell-types as the product of a reference expression matrix (of pure types) and a proportion matrix (or vector). Unlike in chapter 1 where we did not know either of two factor matrices, here we assume the reference gene expression matrix to be known. We only need to estimate the proportion matrix (or vector) in chapter 2.

In both chapters the solution to the matrix decomposition problem can be characterized as the minimum of an optimization problem. In both chapters, to attain this minimum, we use gradient methods. In chapter 1, we consider two slightly different optimization problems related to matrix completion, using lasso-like and scaled-lasso-like objective functions. In both of the problems, minimizers can be found using the proximal gradient descent framework. In chapter 2, we encode our deconvolution problem into a regression-like objective and give a simple gradient descent algorithm to approximately minimize it. Additional techniques including backtracking line search, warm starts, and cross-validation are discussed in both chapters.

The emphasis of chapter 1 and chapter 2 is different. The methods in chapter 1 are already proposed and discussed in the literature. The focus of chapter 1 is, in particular,

the association between optimal tuning parameter and standard deviation of the noise for both lasso-like and scaled-lasso-like formulations of the matrix completion problem. The methods in chapter 2 were developed in this thesis, in particular for the estimation of cell proportions using a particular recent gene expression platform. The focus of chapter 2 is the discussion of these methods, and their evaluation (in terms of adjusted MDSE) for simulated and experimental data.

Chapter 1

SCALED MATRIX COMPLETION

1.1 Introduction

In this chapter we consider combining pivotal penalized estimation, discussed in the context of high-dimensional linear regression [3], with matrix completion [6]. This results in a penalized matrix completion, where the optimal tuning parameter does not depend on the variance of the noise. We consider this new "scaled matrix completion" problem, and compare it to the standard matrix completion problem.

1.1.1 Motivating example

Matrix completion, which was first introduced in [6], solves the following problem: the recovery of an incomplete data matrix based on its observed entries.

There are a number of practical scenarios in which matrix completion applies. One archetypal example is the Netflix problem: Suppose a group of consumers was asked to rate a selection of movies they have watched. Based on this survey, we want to infer their potential interests on movies each individual has not watched. From the survey data, a matrix M with consumer index as column and the movie index as row can be created. The (i, j)-th entry $M_{i,j}$ of the matrix M can be interpreted as consumer i's feedback on movie j. Since we would not expect consumers to have watched and rated all movies listed in the survey, we will get a partially-answered survey and an incomplete matrix M correspondingly. With that, we can restate our task of inference on consumers' preference for new movies based on their ratings,

as completing the incomplete matrix M from its observed entries.

$$\begin{bmatrix}
M_{1,1} & ? & M_{1,3} & \cdots & \cdots & ? & \cdots & M_{1,n} \\
M_{2,1} & M_{2,2} & M_{2,3} & \ddots & & & & \vdots \\
? & M_{3,2} & M_{3,3} & ? & \ddots & & & \vdots \\
\vdots & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\
\vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\
\vdots & & & \ddots & M_{i,j} & ? & ? & M_{i,n} \\
\vdots & & & & \ddots & M_{i+1,j+1} & ? & ? \\
? & \cdots & \cdots & \cdots & \cdots & ? & M_{m,n-1} & M_{m,n}
\end{bmatrix}$$

The goal of matrix completion, in this context, is to approximate a given matrix M, with a low-rank matrix X, and fill in the missing entries of M.

1.1.2 Low-rank approximation

But why are we using a low-rank matrix for approximation?

What does the rank stand for? In the Netflix problem, if we have m consumers and n movies to rate. One might imagine that entires of M are determined by r hidden consumer features such as gender, age, marital status and occupation, etc. In particular, one might imagine a simple linear relationship between average rating, and those features:

$$M_{m\times n} = A_{m\times r} B_{r\times n}$$

$$\begin{array}{c} \\ consumer_1 \\ consumer_2 \\ \vdots \\ \vdots \\ consumer_m \end{array} \begin{array}{c} \begin{array}{cccc} movie_1 & movie_2 & ... & movie_n \end{array} \\ \begin{pmatrix} m_{11} & m_{12} & \dots & m_{1n} \\ m_{21} & m_{22} & \dots & m_{2n} \\ \vdots & \vdots & \dots & \vdots \\ \vdots & \vdots & \dots & \vdots \\ m_{m1} & m_{m2} & \dots & m_{mn} \end{pmatrix} \end{array} =$$

$$\begin{array}{c} \\ consumer_1 \\ consumer_2 \\ \vdots \\ \vdots \\ consumer_m \end{array} \begin{array}{c} \begin{array}{ccccc} feature_1 & feature_2 & ... & ... & feature_r \end{array} \\ \begin{pmatrix} a_{11} & a_{12} & \dots & \dots & a_{1r} \\ a_{21} & a_{22} & \dots & \dots & a_{2r} \\ \vdots & \vdots & \dots & \dots & \vdots \\ \vdots & \vdots & \dots & \dots & \vdots \\ a_{m1} & a_{m2} & \dots & \dots & a_{mr} \end{pmatrix} \end{array} \begin{array}{c} \\ feature_1 \\ feature_2 \\ \vdots \\ \vdots \\ feature_r \end{array} \begin{array}{c} \begin{array}{cccc} movie_1 & movie_2 & ... & movie_n \end{array} \\ \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & b_{2n} \\ \vdots & \vdots & \dots & \vdots \\ \vdots & \vdots & \dots & \vdots \\ b_{k1} & b_{m2} & \dots & b_{rn} \end{pmatrix} \end{array}$$

Here, M is a product of two matrices of rank r: a factor weight matrix, $A \in R^{m\times r}$, and a rating effect matrix, $B \in R^{r\times n}$. In particular, each row of A represents a consumer as a weighted sum of the r underlying features and each column of B represents the rating effect of r underlying features on one movie. The rank can be intuitively taken as the number of those hidden features. In practice, we only want to find a few most predictive hidden features for ratings. Hence, the low-rank structure is ideal. In general, the low-rank approximation is a common sense approach which seeks the simplest explanation that fits the observed data

[6]. Other applications of low-rank approximation include, but are not limited to hidden partitioning [35] and image compression [38].

1.1.3 Previous studies on matrix completion

Not all incomplete matrices M can be recovered. For example, one cannot fully recover a matrix if, for some rows, only one entry was observed. Thus, after the introduction of matrix completion, researchers have studied and shown many properties necessary for a matrix to be efficiently completed, such as low-rank-structure, limited proportion of missing entries, decent signal-to-noise ratio and the relationship between dimension n and the rank r. We give one result here which was described in [6]:

Suppose entries are missing at random and we observe k entries from an n by n matrix M. Let r denote the rank of the M, It was shown in [6] that M can be recovered with high probability if

$$k \geq Cn^{1.2} r \log n. \tag{1.1}$$

Condition (1.1) assumes the rank r is not too large. In fact, if the 1.2 exponent is replaced by 1.5, the recovery result holds for all values of rank r [6].

1.1.4 Optimization language

A matrix completion estimate is often given as the solution to a convex problem. In the following section, we discuss this further.

Given a $m \times n$ matrix Y, we observe:

$$Y_{ij} = \mu_{ij}, \ ij \in \Omega, \tag{1.2}$$

where here and below Ω is the index set of observed entries.

Our goal is to find a matrix of minimal rank that is close to Y on the observed entries.

One might consider solving

$$\begin{aligned} &\text{minimize } rank(\mu) \\ &\text{s.t. } \mu_{ij} = Y_{ij}, (i,j) \in \Omega \end{aligned} \tag{1.3}$$

However, the optimization (1.3) is not trivial to solve for two related reasons:

1. Minimizing the rank is NP-hard

2. The known algorithms which provide the exact solution for (1.3) require time doubly exponential in the dimension n of the matrix in both theory and practice [6]

It is standard practice to replace the rank function $rank(\mu)$ with a convex relaxation, the nuclear norm $||\mu||_*$. The substitution intuitively makes sense since the rank function counts the number of non-vanishing singular values while the nuclear norm sums their amplitude [6]. This convex relaxation solves

$$\begin{aligned} &\text{minimize } ||\mu||_* \\ &\text{s.t. } \mu_{ij} = Y_{ij}, (i,j) \in \Omega \end{aligned} \tag{1.4}$$

This new problem (1.4) is convex and its minimizer can be efficiently found [25].

The discussion about matrix completion so far is based on the assumption that we observe Y without noise, which is usually not true in practice. It is natural to extend it to the case where Y is observed with noise:

$$Y_{ij} = \mu_{ij} + Z_{ij}, \ (i,j) \in \Omega \tag{1.5}$$

where Z is a mean 0, noise term. In this case, we will be solving:

$$\min_{\mu \in \mathbb{R}^{m \times n}} \frac{1}{2} \sum_{(i,j) \in \Omega} (Y_{ij} - \mu_{ij})^2 + \lambda \|\mu\|_* \tag{1.6}$$

Here, the nuclear norm $||\mu||_*$ is a penalty term and λ is a tuning parameter which controls how much we want to penalize a large nuclear norm $||\mu||_*$.

Alternatively, if we define P_Ω as the projection operator onto Ω, we can rewrite (1.5) as

$$P_\Omega(Y) = P_\Omega(\mu) + P_\Omega(Z) \tag{1.7}$$

and we are solving

$$\min_{\mu \in \mathbb{R}^{m \times n}} \frac{1}{2} ||P_\Omega(Y) - P_\Omega(\mu)||_F^2 + \lambda ||\mu||_* \tag{1.8}$$

Problem (1.8) is similar to the lasso estimator (1.10) that we will discuss shortly except that the penalty term in lasso is the ℓ_1 norm. We will use the same method (proximal gradient descent) as is sometimes used for solving lasso [21] to solve the problem (1.8).

It was shown in [14] that a major shortcoming of lasso is its need for a tuning parameter. The optimal value of the tuning parameter depends on a number of a priori unknown quantities and therefore it can be difficult to calibrate in practice. In order to begin to alleviate this issue, authors have recently proposed modifications to the lasso for which the tuning parameter does not depend on the variance of the error. In this chapter, we will evaluate the adaptation of this modification to the matrix completion scenario.

As a preparatory step, we introduce and compare the lasso and scaled lasso with an emphasis on the tuning parameter selection.

1.1.5 Tuning parameter selection

Consider the linear model

$$Y = X\beta_0 + \sigma\epsilon \tag{1.9}$$

where $Y \in \mathbb{R}^n$ is a response vector, $X \in \mathbb{R}^{n \times p}$ a design matrix, $\sigma > 0$ a constant, and $\epsilon \in \mathbb{R}^n$ a noise vector. For the ease of explanation, we assume that the distribution of the noise vector ϵ has variance 1 so that σ is the standard variation of the entire noise $\sigma\epsilon$.

Lasso

The lasso estimator [30] $\hat{\beta}$ is a solution of the minimization problem:

$$\hat{\beta} := arg \min_{\beta \in \mathbb{R}^p} \left\{ \frac{||Y - X\beta||_2^2}{n} + \lambda ||\beta||_1 \right\} \tag{1.10}$$

It is well understood that, to optimize for mean-squared-error (MSE), one should choose a tuning parameter-value, λ, on the order of

$$\lambda \sim \frac{\sigma ||X^T \epsilon||_\infty}{n} \tag{1.11}$$

(1.11) indicates that the optimal tuning parameter λ in the lasso depends on the standard deviation of the noise, σ, which is usually unknown in practice. To avoid this problem, the square-root lasso was introduced and studied in [3].

Square-Root Lasso

For a fixed tuning parameter λ_0, the Square-root lasso is defined similarly to the Lasso:

$$\hat{\beta}_{sqr} := arg \min_{\beta \in \mathbb{R}^p} \left\{ \frac{||Y - X\beta||_2}{\sqrt{n}} + \lambda_0 ||\beta||_1 \right\} \tag{1.12}$$

For optimal MSE, the tuning parameter λ_0 here should be on the order of

$$\lambda_0 \sim \frac{||X^T \epsilon||_\infty}{n} \tag{1.13}$$

which no longer requires the estimation of σ. Additional intuition for this is given in [15]:

One can see that the square-root lasso optimization problem (1.12) is equivalent to:

$$\hat{\beta}_{sqr} := arg \min_{\beta \in \mathbb{R}^p} \left\{ \frac{\frac{||Y - X\beta||_2^2}{n}}{\frac{||Y - X\beta||_2}{\sqrt{n}}} + \lambda_0 ||\beta||_1 \right\} \tag{1.14}$$

the factor $\frac{||Y - X\beta||_2^2}{\sqrt{n}}$ in the numerator of the first term is the same as the first term in lasso (1.10). The additional factor in the denominator $\frac{||Y - X\beta||_2}{\sqrt{n}}$ acts like an inherent estimator of the standard deviation of the noise σ, therefore the tuning parameter λ_0 is "σ-free"[15].

Scaled Lasso

Scaled lasso, which was introduced in [29], yields the same estimate of β as the square-root lasso (1.12). In scaled lasso, one simultaneously estimates the unknown regression coefficients $\hat{\beta}_b$ and the scale $\hat{\sigma}_b^2$ by solving:

$$\left(\hat{\beta}_b, \hat{\sigma}_b^2\right) := arg \min_{\beta \in \mathbb{R}^p, \sigma^2 > 0} \left\{ \frac{||y - X\beta||_2^2}{n\sigma} + \sigma + \lambda ||\beta||_1 \right\} \tag{1.15}$$

The solution path of the scaled lasso is 1-to-1 with the lasso path, the benefit being that in scaled lasso the penalty parameter is now scale independent ("σ-free").

We conclude by noting that in [31], the name "scaled lasso" was used to refer to the estimator

$$\left(\hat{\beta}_b, \hat{\sigma}_b^2\right) := arg \min_{\beta \in \mathbb{R}^p, \sigma^2 > 0} \left\{ \frac{||Y - X\beta||_2^2}{n\sigma^2} + \log \sigma^2 + \frac{2\lambda_0 ||\beta||_1}{\sigma} \right\} \tag{1.16}$$

This estimator is not equivalent to the version of the scaled/square-root lasso detailed above. In this thesis, when referring to the "scaled lasso", we mean the solution to (1.15).

Square-Root MC and Scaled MC

We introduce the lasso-like and scaled-lasso-like formulations for matrix completion problems here. For the ease of exposition, we call the following two formulations square-root MC and scaled MC for the rest of the thesis.

Let q be the number of observed entries.

1. Square-Root MC

$$\hat{\mu}_{sqr} := arg \min_{\mu \in \mathbb{R}^{m \times n}} \left\{ \frac{||P_\Omega(Y) - P_\Omega(\mu)||_F}{\sqrt{q}} + \lambda ||\mu||_* \right\} \tag{1.17}$$

2. Scaled MC

$$\left(\hat{\mu}_b, \hat{\sigma}_b^2\right) := arg \min_{\mu \in \mathbb{R}^{m \times n}, \sigma^2 > 0} \left\{ \frac{||P_\Omega(Y) - P_\Omega(\mu)||_F^2}{q\sigma} + \sigma + \lambda ||\mu||_* \right\} \tag{1.18}$$

As in the lasso case, (1.17) and (1.18) give identical solutions for the same λ-value. Also, as in the lasso case, the solution path for the scaled/square-root MC estimator is identical to the solution path of the standard MC estimator. However, as we will empirically show, the tuning parameter λ which optimizes MSE for the scaled/square-root MC estimator does not depend on the variance of the noise (unlike for the standard MC estimator).

1.2 Methods

We use proximal gradient descent to solve standard MC (1.8) and scaled MC (1.18). Here, we introduce proximal gradient descent [36] and other techniques used in the simulation study.

1.2.1 Proximal gradient descent

The proximal gradient descent (or generalized gradient descent) is widely used in convex optimization. As the generalization of the gradient descent, proximal gradient descent applies to problems with the objective function with the following form:

$$f(\nu) = g(\nu) + h(\nu),$$

where the objective function $f(\nu)$ can be written as the sum of $g(\nu)$, a convex, differentiable function, and $h(\nu)$, a convex but not necessarily differentiable function. In standard MC (1.8), we have $g(\nu) := \frac{1}{2}||P_\Omega(Y) - P_\Omega(\mu)||_F^2$ and $h(\nu) := \lambda||\mu||_*$

Then we can think about two ingredients needed for proximal gradient descent:

1. $\nabla g(\nu) = -\left(P_\Omega(Y) - P_\Omega(\mu)\right)$

2. the *prox* operator: $prox_t(\mu) = argmin_Z \frac{1}{2t}||\mu - Z||_F^2 + \lambda||Z||_\star$

It was shown in [4] that

$$prox_t(\mu) = S_{\lambda t}(\mu), \text{matrix soft-thresholding at the level } \lambda$$

Here $S_\lambda(\mu)$ is defined by:

$$S_\lambda(\mu) = U\Sigma_\lambda V^T,$$

where $\mu = U\Sigma V$ is a singular value decomposition (SVD), and Σ_λ is diagonal with

$$(\Sigma_\lambda)_{ii} = max\left\{\Sigma_{ii} - \lambda, 0\right\}.$$

The proximal gradient update step is:

$$\mu \leftarrow prox_t\left(\mu - t\nabla_g(\mu)\right) \tag{1.19}$$

$$= S_{\lambda t}(\mu + t(P_\Omega(Y) - P_\Omega(\mu))). \tag{1.20}$$

We use (1.20) to solve standard MC (1.8). Details are given in Algorithm (1).

For scaled MC (1.18), we can combine proximal gradient descent with block coordinate descent [27], and cyclically update the $\hat{\mu}$ and $\hat{\sigma}^2$.

- Fix $\hat{\sigma}^2$, and update μ using proximal gradient descent as we did in lasso

- Fix $\hat{\mu}$, and update the $\hat{\sigma}^2$ by $\sigma \leftarrow ||Y - \hat{\mu}||_F$

Details of this algorithm are given in Algorithm (2).

Algorithm 1: Solver for Standard MC

```
Choose λ;
while stopping criteria was not met do
    U ← Ω ∗ Y + (1 − Ω) ∗ Θ;
    Θ ← S_λ(U) ;
end
```

Algorithm 2: Solver for Scaled MC

Choose λ;

Initialization σ;

while *stopping criteria was not met* **do**

  $U \leftarrow \Omega * Y + (1 - \Omega) * \Theta$;

  $\Theta \leftarrow S_{\sigma\lambda}(U)$;

  $\sigma \leftarrow ||Y - \Theta * \Omega||_2$;

end

1.2.2 Optimal tuning parameter selection

A sequence of λs were used for each set of training values (dimension n, rank, proportion of missing entries p). The "optimal tuning parameter" that we will refer to, is the λ-value with the smallest prediction error (or MSE in the context). Since the true μ is known in the simulated data, we compare it to $\hat{\mu}$ to evaluate the MSE. In real applications with μ unknown, one could use cross-validation error.

$$\lambda^\star = argmin_\lambda \left(MSE(\hat{\mu})\right) \tag{1.21}$$

$$MSE(\hat{\mu}) = \frac{1}{q^2}||P_\Omega(\hat{\mu}) - P_\Omega(\mu)||_F^2 \tag{1.22}$$

1.2.3 Warm start

Warm starts were used in solving both standard MC (1.8) and scaled MC (1.17). Warm starting [8] is a technique to reduce the running time of iterative methods by using the solution of a slightly different optimization problem as an initial point for the current problem. Specifically in this thesis, we solve our problem along a decreasing sequence of λ-values and use the solution of the problem with the previous λ-value as the initial point for the current problem.

1.3 Simulation Study

1.3.1 Overview

It is well studied [3] that the optimal tuning parameter $\lambda^\star$ for scaled lasso is invariant to the standard deviation of the noise σ. We show this invariance still holds for the scaled MC. Keeping all other factors (dimension n, rank k, the proportion of missing entries, etc.) constant, we assess how $\lambda^\star$ changes with σ, for both standard MC and the scaled MC.

1.3.2 Data

For each set of parameters:

$$\begin{aligned} \text{The dimension of the matrix} &: n \\ \text{The matrix rank} &: r \\ \text{Percent of missing entries} &: p \\ \text{Standard deviation of the noise} &: \sigma \end{aligned}$$

We generate the $Y := \mu + \epsilon$ in following steps:

1. Entries of μ were generated independently from the standard normal distribution

2. SVD was then used to make the μ have the rank r

$$\begin{aligned} \text{Decompose } \mu &\to U\Sigma V^T \\ \text{Set } \Sigma_{ii} = 0 \text{ to get } \Sigma^\star, &\text{where } i = r+1, r+2, \ldots, n \\ \text{Update } \mu &\leftarrow U\Sigma^\star V^T \end{aligned}$$

3. The gaussian noise ϵ was generated from normal distribution $N(0, \sigma)$ and added to the μ to get the $Y \leftarrow \mu + \sigma$

4. A mask matrix, Ω, with entries either 0 or 1, where 0 indicates unobserved entries and 1 for observed entries, was generated from bionomial ditribution: $\Omega_{ij} \sim Binomial(n^2, 1, 1-p)$

5. Update Y with Ω: $Y \leftarrow \Omega * Y$

1.3.3 Results

Figure 1.1 shows that regardless of the selection of n, when keeping rank r and percent of missing p constant, the optimal tuning parameter $\lambda^\star$ for standard MC increases as the standard deviation of the noise σ increases, while the optimal tuning parameter for scaled MC does not vary with σ.

If we look at the scaled MC alone, Figure 1.2 shows that the optimal tuning parameter $\lambda^\star$ is invariant to σ regardless of n, p and r.

Although $\lambda^\star$ is affected by σ differently in standard MC and scaled MC, we see similar trends of how $\lambda^\star$ is associated with n, r in both problems. Specifically,

1. Bigger $\lambda^\star$ is expected for larger dimension n

2. Bigger $\lambda^\star$ is expected for larger rank r

1.4 Discussion

We showed that the optimal tuning parameter $\lambda^\star$ of scaled MC is invariant to the standard deviation of the noise, σ. While in a single penalty problem, this may not make too much difference (generally cross validation will be performed over a sequence of lambda-values), in problems involving additional penalties (eg. additive models, or matrix completion with additional structure), this may allow us to decrease computational burden by either coupling tuning parameter values or a priori selecting a tuning parameter value for the nuclear-norm penalty. Future work could involve studying the invariance of optimal tuning parameter in

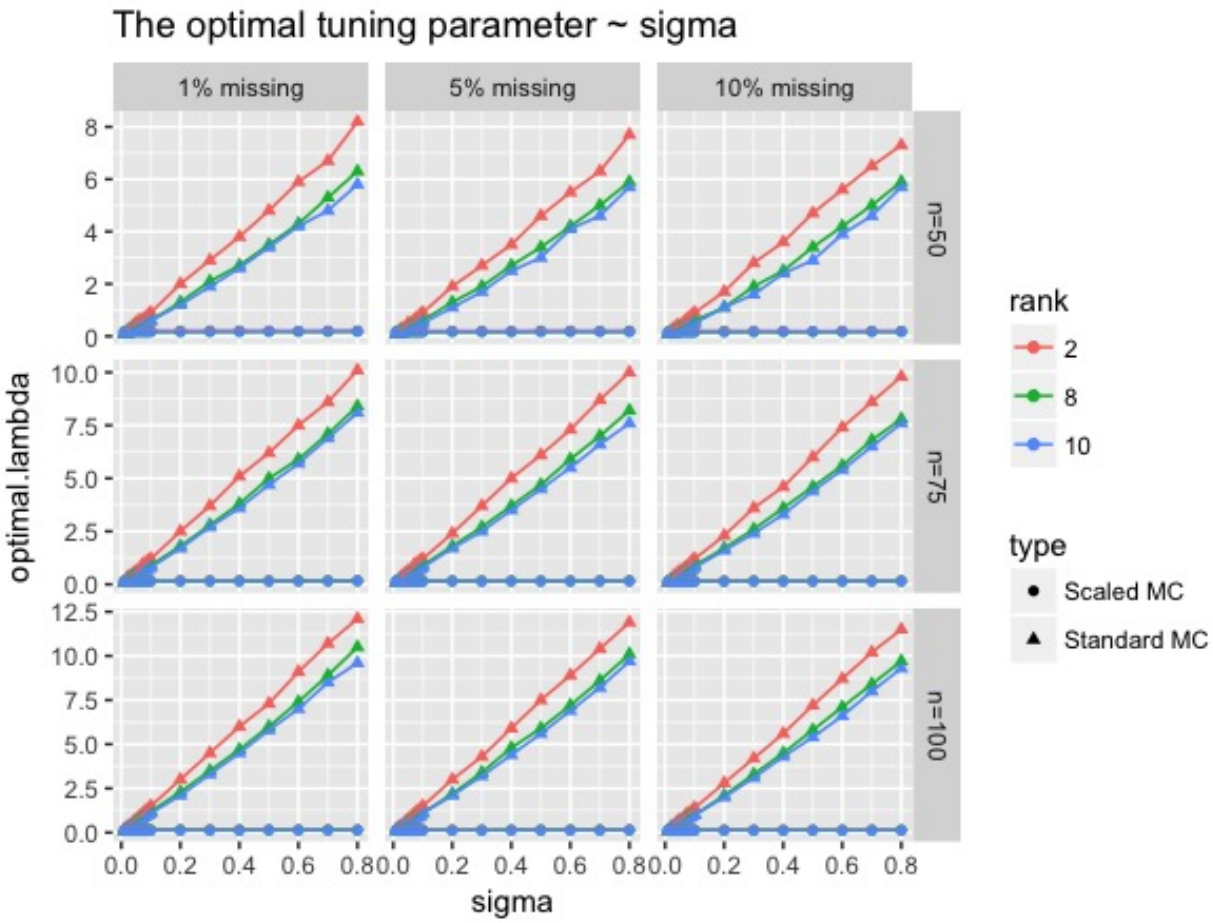


Figure 1.1: Optimal tuning parameter with sigma for scaled MC and standard MC

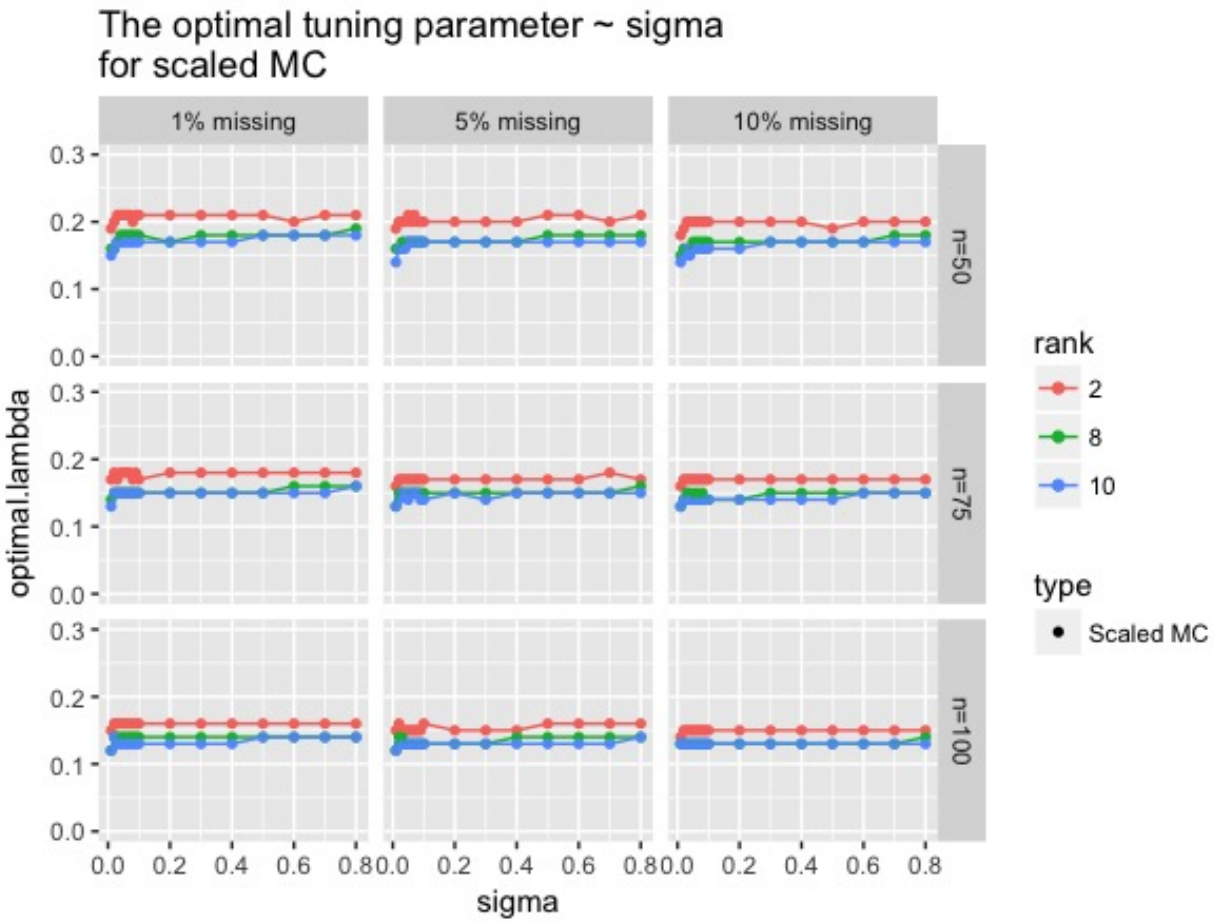


Figure 1.2: Optimal tuning parameter with sigma for scaled MC

problems with multiple penalties for scaled MC. The relationship between optimal tuning parameter values in single and multiple penalty problems is also of interest.

Chapter 2

CELL DECONVOLUTION WITH NANOSTRING DATA

2.1 Introduction

2.1.1 Biological background

In deconvolution, we aim to identify properties and concentrations of components from an observed mixture. In the context of molecular biology, deconvolution methods have been used to identify constituent cell-types in a tissue, along with their relative proportions.

Knowledge of the relative proportion of cell-types can be quite useful in biomedical applications. For example, changes in tissue composition are often indicative of disease progression or drug response. In malignant tumors, levels of infiltrating immune cells are associated with tumor growth, cancer progression and patient outcome. Unfortunately, calculating relative proportion of cell-types can be difficult. Experimental methods, like immunohistochemistry and flow cytometry, fail to perform well not only because they require significant time, effort and expense, but because they may also result in insufficient RNA abundance in the sample preparation step. For example, tissue disaggregation before flow cytometry can lead to lost or damaged cells. There is interest in computational methods that, in silico, deconvolve proportions from a mixture.

2.1.2 State of the art

Computational cell deconvolution methods can be generally categorized into two groups: reference-based and reference-free. Reference-based methods are supervised methods requiring the gene-expression of cell types to be known. Reference-free methods are unsupervised methods for use when the number of cell types or their expression are not fully available.

Several cell deconvolution methods have been proposed [28, 1, 11, 22, 17, 37, 39]. These methods perform accurately on distinct cell subsets in mixtures with well-defined composition (for example, blood) but are criticized as being less effective for mixtures with unknown content and noise (for example, solid tumors) [20]. As a remedy for these limitations, CIBERSOR, a novel method with application of linear support vector regression (SVR) [2] was proposed. CIBERSORT requires an input matrix of reference gene expression signatures, to estimate the relative proportions of each cell type of interest [20]. The most significant current limitation of CIBERSORT, and indeed all signature reference based methods, is the fidelity of reference profiles [16]. In addition, CIBERSORT is less effective with highly correlated features [16]. However, none of these methods provides quantitative information about both cancer and non-malignant cell type proportions directly from tumor gene expression profiles. To overcome this, EPIC [23] was introduced as a robust approach to simultaneously Estimate the Proportion of Immune and Cancer cells (EPIC) from bulk tumor gene expression data. Unfortunately, these methods are not applicable to all measurement platforms — we discuss this in more depth next.

2.1.3 Opportunities

Variation in RNA expression data can be attributed to a variety of factors including the quality of the starting material, the level of cellularity and RNA yield, the platform employed, and the person performing the experiment. Modeling of the variation or the error distribution is fundamental to parametric cell-deconvolution methods. It is necessary to develop platform-specific methods as different sequencing platforms perform differently in terms of error distribution.

We focus on one specific platform, the NanoString Technologies nCounter platform in this chapter. The NanoString Technologies nCounter platform hybridizes fluorescent barcodes directly to specific nucleic acid sequences, allowing for the nonamplified measurement of up to 800 targets within one sample [26, 9]. Assessing expression values without the need for amplification (which may cause technical artifacts) is potentially a very large advantage. A

number of papers have shown that the NanoString platform is, in many ways, comparable with other technologies [33, 34, 10]. However, unlike Microarray and RNA sequencing data, where the error is believe to follow log-normal distributions, NanoString expression counts have a Gaussian error.

Since the NanoString Technologies nCounter platform is a relatively new technology, most existing cell deconvolution methods were designed for platforms like microarrays and RNA sequencing. Few efforts have been made to tailor deconvolution to NanoString Technologies. In this chapter, we propose a cell-deconvolution algorithm for NanoString data. This methodology could also be applied to any expression data with error distribution similar to the NanoString platform.

2.1.4 Error distribution of the NanoString data

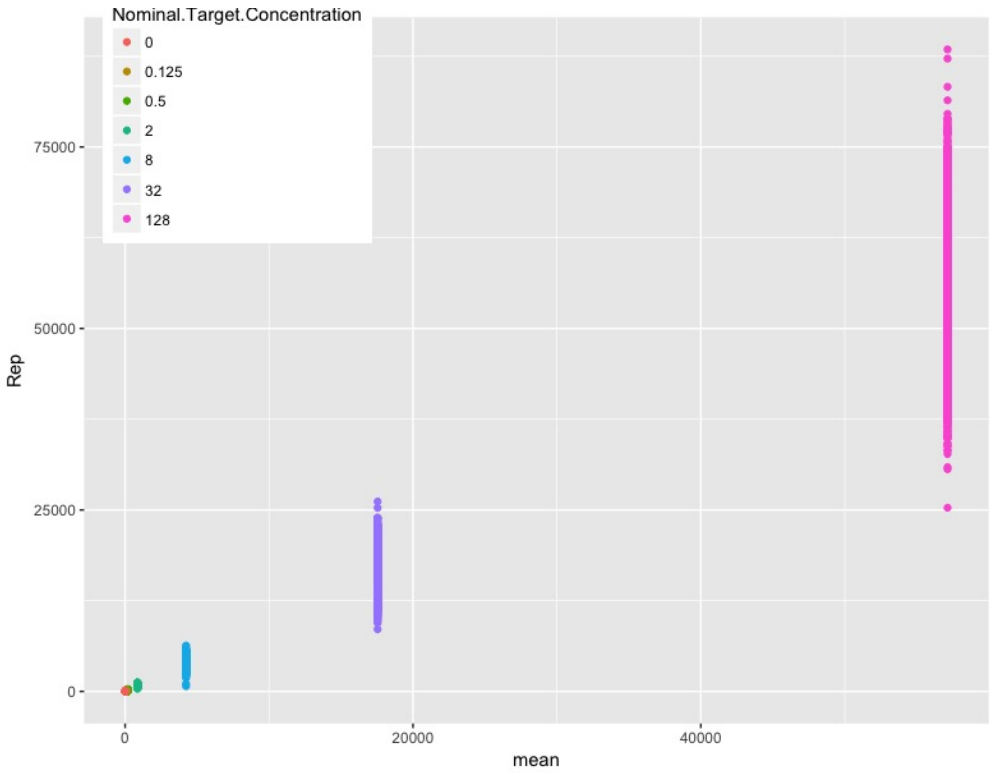
The error distribution of NanoString data was evaluated by studying the ERCC[1] sample measured by NanoString Technologies Platform. The ERCC was designed to have mimic mRNAs with a series of Nominal Target Concentrations[2].

There are two properties of error distribution that differentiate the NanoString data from other platforms: (1) the errors are normally distributed, and (2) the standard deviation of the error is a relatively linear function of the mean (i.e. the coefficient of variation is nearly constant as the mean changes). Our deconvolution method was developed to particularly

[1]To control for these variation in the RNA expression, a common set of external RNA controls has been developed by the External RNA Controls Consortium (ERCC), an ad-hoc group of academic, private, and public organizations hosted by the National Institute of Standards and Technology (NIST). The controls consist of a set of unlabeled, polyadenylated transcripts designed to be added to an RNA analysis experiment after sample isolation, in order to measure against defined performance criteria. Up until the design of such universally accepted controls, it has been difficult to execute a thorough investigation of fundamental analytical performance metrics. From the trusted brand of quality RNA reagents, Ambion ERCC Spike-In Control Mixes are commercially available, pre-formulated blends of 92 transcripts, derived and traceable from NIST-certified DNA plasmids. The transcripts are designed to be 250 to 2,000 nt in length, which mimic natural eukaryotic mRNAs.

[2]Nominal Target Concentration (NTC) quantifies how much synthetic alien mRNA targets we spike in each sample, the target with 0 NTC is absent from the sample, which is commonly used to measure the background noise. Other synthetic alien mRNA targets are spiked in at NTC of 0.125,0.5,2,8,32,128 fM, fM is femto molar.

Figure 2.1: Scatterplot



accommodate these two properties.

We first graphically and numerically show that NanoString counts have a nearly normal error distribution. We then show that Coefficient of Variation (CV) is relatively constant as the true concentration changes, especially when the concentration is large.

We see, from Scatterplot 2.1 and Table 2.1 that the CV starts high (60.35%) when the Nominal Target Concentration (NTC) is 0; it then decreases and stays at around 15%. It is an important finding since it means the standard deviation of the noise grows at roughly the same order as the concentration. We would not worry too much about the high CV in the low concentration since, in practice, low-expressed genes are screened out in the quality control process. The stable CV property also indicates the NanoString data error cannot be simply modeled as an additive normal error with fixed variance (as this would imply a

decreasing CV). To see the normality of NanoString data, we then looked at Q-Q plots of NanoString data. For Each plot, we have the expression counts with the same concentrating. Q-Q plots in Figure 2.2 indicate the normal distribution to be a reasonable approximation for the error distribution.

Based on our empirical findings that the CV is relatively stable and the errors are normal, we consider the model:

$$y = \beta_0 + \mu + \mu\epsilon, \qquad \epsilon \sim N(0, \sigma_0) \tag{2.1}$$

where y is the expression of the mixture measured on one gene, β_0 is an additive background noise, μ is the true expression count (unobserved), ϵ is a normal error with mean 0 and standard deviation σ_0. The model meets our needs of normal error and stable CV which can be verified in (2.2) and (2.3).

$$y \sim N(\beta_0 + \mu, \mu\sigma_0) \tag{2.2}$$

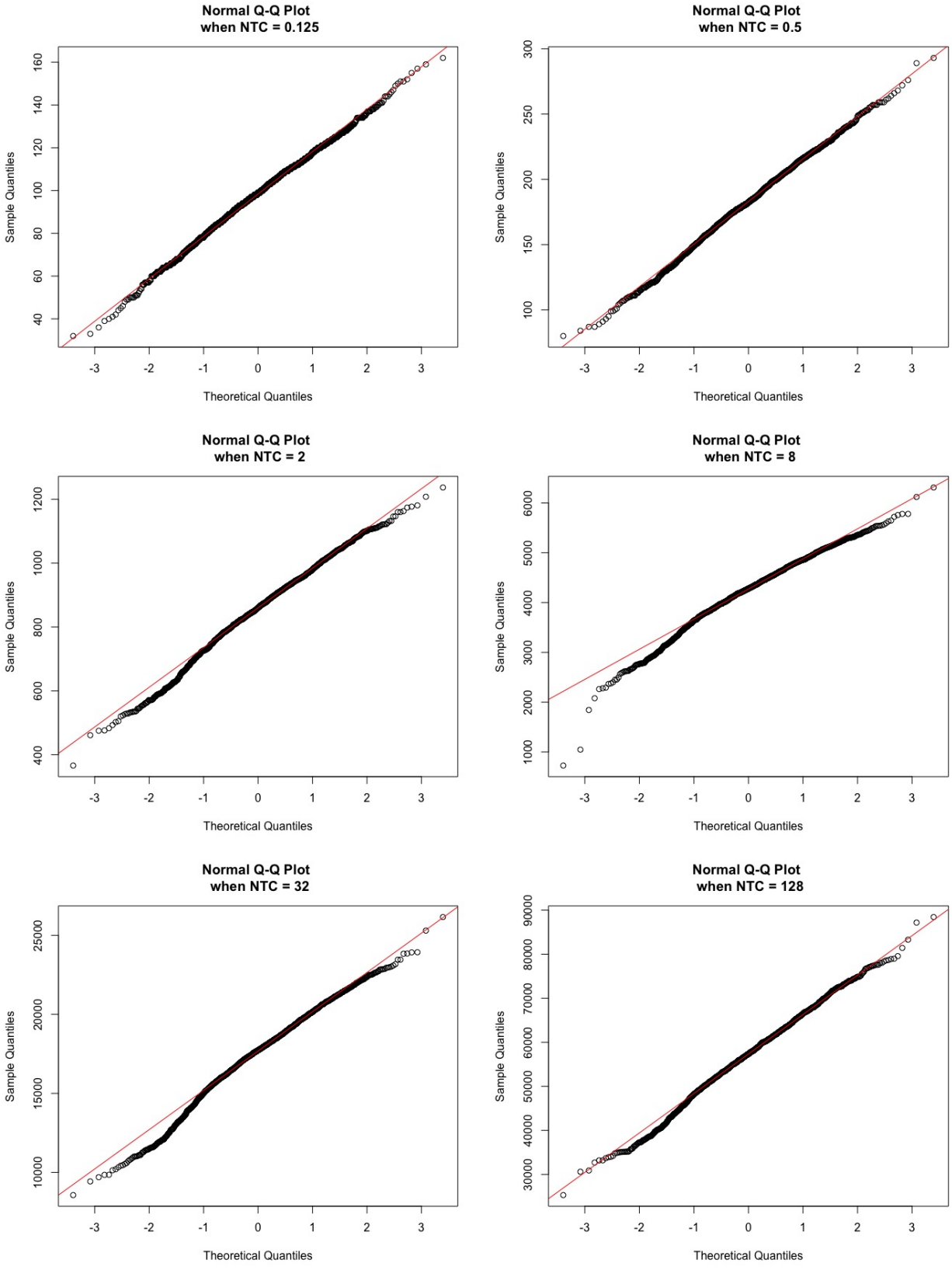
$$CV = \frac{SD(y)}{mean(Y)} = \frac{\mu\sigma_0}{\beta_0 + \mu} \xrightarrow{\mu\to\infty} \sigma_0 \tag{2.3}$$

Now if we measure the mixture expression on n genes instead of only 1, assume independence of our measurements, and that the mixture comes from p cell types, we can write (2.1) in a multivariate form

$$\underset{n\times 1}{Y} = \underset{n\times 1}{\beta_0} + \underset{n\times p}{C} \times \underset{p\times 1}{\alpha} \left(1 + CV\epsilon\right), \epsilon \sim N(\mathbf{0}, \mathbf{1}) \tag{2.4}$$

where Y is the expression vector of the mixture measured on n genes, β_0 is a vector denoting additive background for each gene, C is the cell-type expression profile matrix, where each column represents the (known) expression profile on n genes for one cell type, α is the proportion vector we are estimating (indicating the mixture of each cell-type in our sample), $\mu_i := [C\alpha]_i$, the i-th element of $C\alpha$, is the true expression of gene i in the mixture (unobserved) and CV is the coefficient of variation. ϵ is an error vector following the multivariate normal distribution with mean vector of $\mathbf{0}$, and identity covariance.

Figure 2.2: Q-Q Plots



One can easily verify that, conditional on a fixed C, Y follows a multivariate normal distribution:

$$Y \sim N(\phi, \Sigma) \tag{2.5}$$

where

$$\phi = \beta_0 + C \times \alpha$$

and

$$\Sigma = CV * \text{diag}\,(C\alpha)$$

2.2 Methods

2.2.1 Estimator

Similar to the ordinary least estimator (OLS) in linear regression, we seek the estimates which minimize the sum of the squares of error ϵ.

We re-write (2.4) to get:

$$\epsilon_i = \frac{1}{CV}\left(\frac{Y_i - \beta_{0,i}}{\mu_i} - 1\right) \tag{2.6}$$

and then substitute μ_i by $(C\alpha)_i$ to get

$$|\epsilon|^2 = \frac{1}{CV^2}\sum_i \left(\frac{Y_i - \beta_{0,i}}{(C\alpha)_i} - 1\right)^2 \tag{2.7}$$

In this thesis, we assume the background noise vector β_0 to be known and only estimate α. With this assumption, by minimizing $|\epsilon|^2$, we get the estimator of the form:

$$\hat{\alpha} = argmin_\alpha \frac{1}{2n}\sum_i \left(\frac{y_i - \beta_{0,i}}{(C\alpha)_i} - 1\right)^2 \tag{2.8}$$

2.2.2 Gradient descent

For the rest of the section, we discuss optimization of (2.8) . We use gradient descent with a backtracking strategy for step-size optimization to ensure that each step decreases the objective.

Given the objective function (or loss function),

$$J = \frac{1}{2n} \sum_i \left(\frac{y_i - \beta_{0,i}}{(C\alpha)_i} - 1 \right)^2$$

we have the component of the gradient in α

$$\nabla J = -\frac{1}{2n} \sum_{i=1}^{n} \left(\frac{y_i - \beta_{0,i}}{(C\alpha)_i^2} \right) \left(\frac{y_i - \beta_{0,i}}{(C\alpha)_i} - 1 \right) C_i$$

2.2.3 Backtracking line search

We used a backtracking line search [12] to optimize step-size. In (unconstrained) optimization, the backtracking line search strategy is used as part of a line search method, to compute how far one should move along a given search direction. It depends on two constants $\delta \in (0, 0.5)$ and $\gamma \in (0, 1)$. Let η be the step size. The backtracking line search starts with unit step size $\eta = 1$ and then reduces it by the factor γ until the stopping criteria $f\left(x - \eta \nabla f(x)\right) \leq f(x) - \delta\eta ||\nabla f(x)||^2$ is met. The constant δ can be interpreted as the fraction of the decrease in f predicted by linear extrapolation that we will accept. Although we only got positive values from the algorithm in the simulation study, the backtracking line search in this context is not guaranteed, to give positive α estimates. One can put constraints to force the algorithm to give positive results.

Algorithm 3: Backtracking line search

1 Set iteration counter $k = 0$, initial guess x^0, choose initial $\eta = 1$;
2 **while** $f\left(x^k - \eta^k \nabla f(x^k)\right) > f(x^k) - \delta\eta^k ||\nabla f(x^k)||^2$ **do**
3 $\quad \eta^k = \gamma\eta^k$;
4 **end**
5 $x^{k+1} = x^k - \eta^k \nabla f(x^k)$ and update $k = k + 1$;
6 Go to 1 until $||\nabla f(x^{(k)})|| < \epsilon$;

2.2.4 Stopping criteria

The relative change in loss function of the current iterate compared to the previous iterate is evaluated. We set a precision threshold for this relative loss change. The algorithm is stopped either when (1) the relative loss change is below the precision threshold, or (2) the pre-specified maximum number of steps were taken. For the latter, we call it a "difficult problem" and indicate "not enough steps for this level of precision". The default maximum run is 1000. When there are many cell types, few genes or strong correlation between cell-types, we find that 1000 steps may not be enough to reach convergence. However, these are also cases where there is often not enough signal for strong statistical performance at any level of computational convergence.

2.2.5 Evaluation criteria

The Median Square Error (MDSE) adjusted for the number of cell types p was used to quantify the prediction error. The unadjusted prediction error is problematic especially when the number of cell types p is large. Since all true proportions add up to one $\sum_{i=1}^{p} \alpha_i = 1$, one will see smaller α_i and thus smaller difference $(\hat{\alpha}_i - \alpha_i)$ when p increases. This leads to a negative association between p and unadjusted prediction error which contracts our intuition, since increasing the cell types should not make the prediction easier. We use median square error (MDSE) instead of mean square error (MSE) because the median is more robust to outliers than mean is (otherwise occasional simulation iterates with convergence issues can skew performance). This is especially useful when we look at the association between prediction errors and other factors (dimension n, number of cell types p) when prediction errors are close to 0. For example, when we assess the consistency of the estimator, the prediction error is close to 0 when n is large.

$$Adjusted\ MDSE = \text{Median}\left(\sum_{i=1}^{p} \left(p\left(\hat{\alpha}_i - \alpha_i\right)\right)^2\right)$$

$$Unadjusted\ MDSE = \text{Median}\left(\sum_{i=1}^{p}(\hat{\alpha}_i - \alpha_i)^2\right)$$

2.3 Simulation study

2.3.1 Overview

We numerically assess how our cell deconvolution method is affected by the following factors, in terms of the MDSE:

1. Number of genes (biomarkers) n
2. Number of cell types to deconvolve p
3. Coefficient of Variation (CV)
4. Correlation between cell-types (or genes (biomarkers))

Intuitively, the more uncorrelated genes (biomarkers) and the less cell types we have, the easier the task is. The large coefficient of variation could play a role by inducing small signal-to-noise ratio, but the magnitude of its effect needs to be evaluated. High similarity between expression profiles of different cell types also makes deconvolution more difficult. Correlation between genes within cell types can also be problematic — in some sense the difficulty of the problem is related to the number of uncorrelated measurements we have on each cell type (correlation between genes decreases this number). In practice, biological knowledge can be used to select uncorrelated sets of genes that differentiate the various cell types to partly mitigate these issues.

2.3.2 Data generation

To mimic scenarios in which deconvolution might be more or less difficult, we simulated data with varying degrees of similarity (induced by correlation) between cell types. The correlation between cell types is reflected in the reference expression matrix C.

Reference expression matrix C

Rows of the reference expression matrix C were generated independently from a multivariate normal distribution with specified mean vector θ and covariance matrix Ω.

$$C = [C_1, \ldots, C_n]^T$$
$$C_i \sim N(\theta, \Omega)$$

Mean vector θ

We first generated the standard deviation σ_i of each cell-type from a truncated normal distribution $N(20, 8)$ (left truncated at 0). The selection of $N(20, 8)$ was nothing special but to generate postive values with high probability. With pre-specified CV and standard deviation σ_i of each cell-type, we calculate the mean vector θ by

$$\theta_i = \frac{\sigma_i}{CV}$$

Covariance matrix Ω

The correlation matrix was generated under a block diagonal matrix structure. In each block, all diagonal entries are 1, and all off diagonal entries are a pre-specified value ρ. We set $\rho = 0$ for the uncorrelated cell-type scenario and $\rho > 0$ for correlated cell-types scenario. We then used the standard deviation σ_i of each cell-type and the correlation matrix to get the covariance matrix Ω.

The mixture expression vector Y

With C, pre-specified background vector β_0, and true proportion vectors α, Y is generated from the multivariate normal distribution with mean vector and covariance matrix specified by in (2.4).

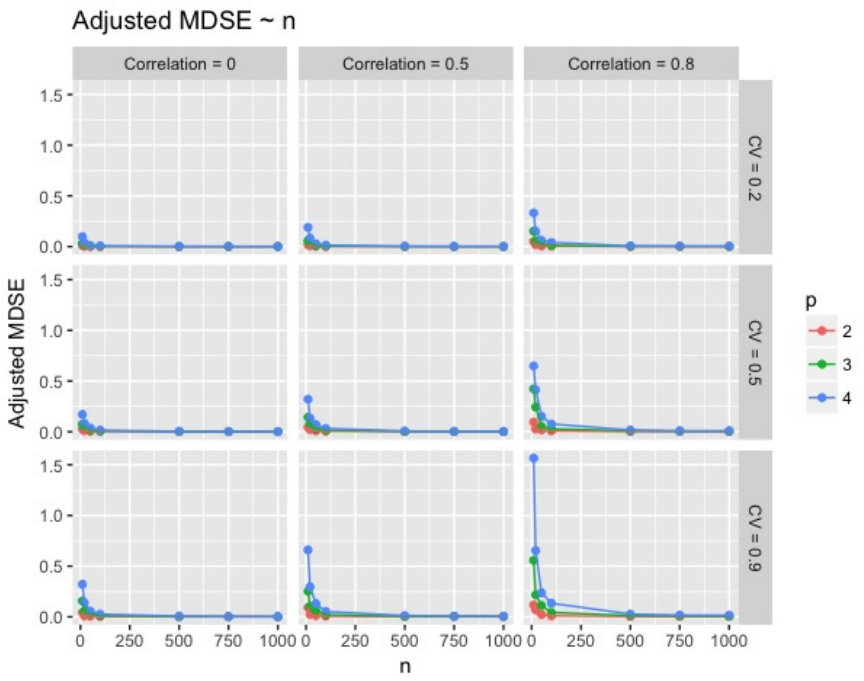


Figure 2.3: Adjusted MDSE $\sim n$

2.3.3 Results

Number of biomarkers n

Figure 2.3 shows that the Adjusted-MDSE decreases with increasing n for all choices of p, CV and correlation between cell types. Although larger p , CV and high correlation between cell types make the deconvolution harder in terms with larger Adjusted-MDSE, the deconvolution method still achieve small Adjusted MDSE when n is large.

To assess the asymptotic consistency of the estimator, we re-plotted the Adjusted MDSE and n in log-log scale. Figure 2.4 indicates that the slope is roughly identical across all simulation settings, thus the convergence rate is the same in all settings. Figure 2.3 and 2.4 jointly indicate the the estimator is asymptotically consistency in terms of adjusted MDSE.

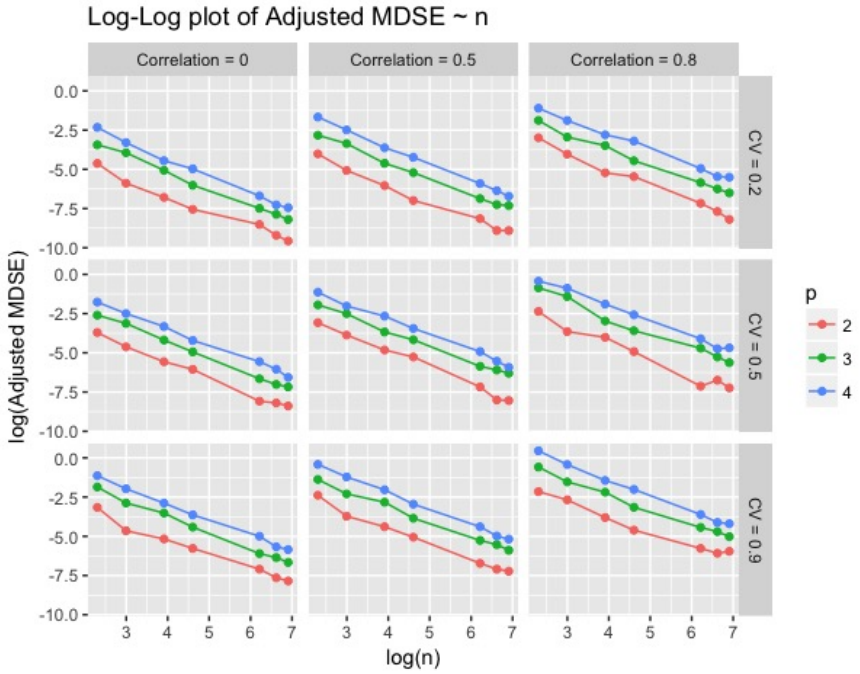


Figure 2.4: Log-Log plot of Adjusted MDSE $\sim n$

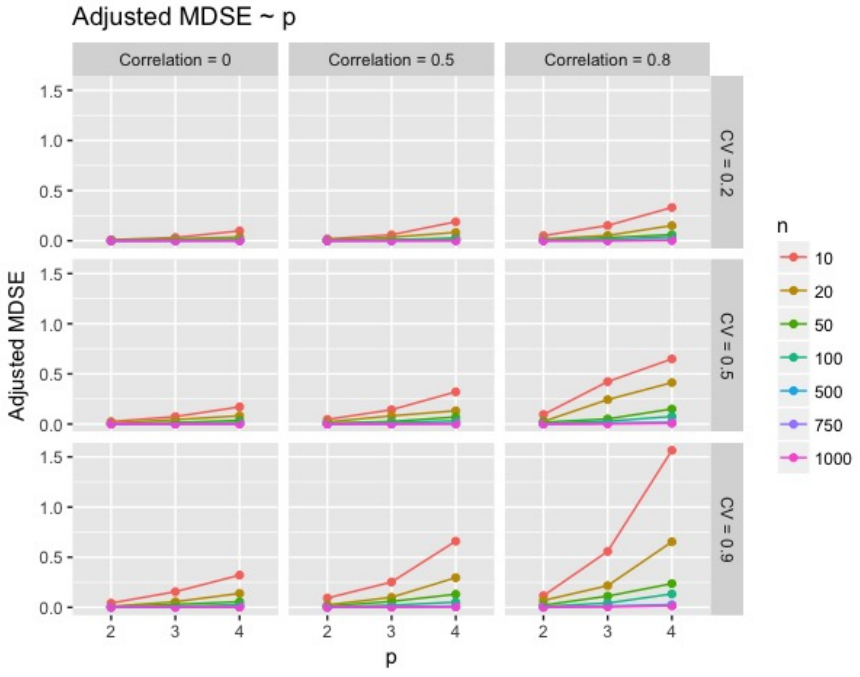


Figure 2.5: Adjusted MDSE $\sim p$

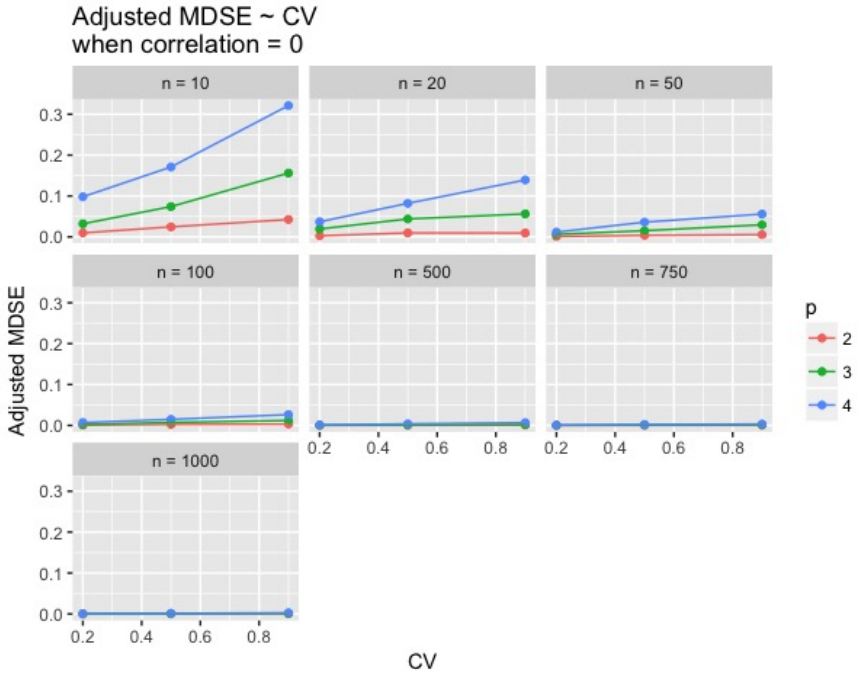


Figure 2.6: Adjusted MDSE $\sim$ CV when correlation $= 0$

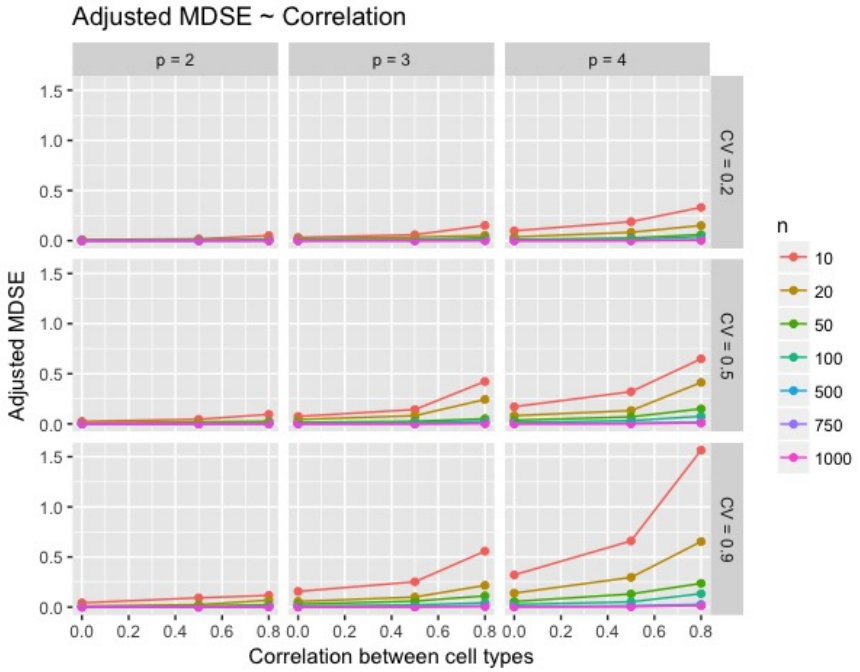


Figure 2.7: Adjusted MDSE $\sim$ correlation

Number of cell types p

It is shown in Figure 2.5 that Adjusted MDSE increases with p. This trend is uniform for all choices of n, CV and correlations. When n is large, the effect of p on Adjusted MDSE is relatively small. This is in concordance with our intuition that, increasing the number of cell types makes the deconvolution harder but we can compensate for the effect of large p on the Adjusted MDSE by having large n.

Correlation between cell types

It is shown in Figure 2.7 that the Adjusted MDSE increases with higher correlation. The correlation between cell types affects the adjusted MDSE by decreasing the number of effective biomarkers. To compensate the effect of high correlation, one needs to have more biomarkers.

Coefficient of Variation CV

It is shown in Figure 2.6 that the Adjusted MDSE increases with large CV for all choices of n and p when cell types are uncorrelated. The effect size of CV is relatively small when the number of biomarkers n is large.

2.4 Performance on the real data

The cell deconvolution method was performed on real sample data from two experiments: (1) Brain and Human-Reference mix experiment and (2) Protein mix experiment

2.4.1 Sample and Data Preparation

Brain and HR mix sample

The Pure Brain sample and the pure HR sample were mixed by technicians with two proportions: 25%Brain + 75%HR and 75%Brain + 25%HR. Two mixed samples and two pure

samples (100%Brain and 100%HR) were each measured three times on 48 biomarkers through NanoString Technologies.

The aggregated measurement of each sample was obtained by taking the mean of three replicates. We excluded biomarkers not highly correlated with true proportions to get a total 39 biomarkers from 48. The expression of each mixed samples was taken as one Y. The expression profile matrix C was constructed by combining expressions of pure Brain sample and pure HR sample. The background vector and slope parameter were assumed to be 0 and 1. The CV was specified as 0.2, as what we estimated from the ERCC sample. The correlation of two pure samples is calculated to be 0.609.

Protein mixed sample

The CCRF-CEM sample and the HEK293 sample were mixed by technicians with four proportions: 50%CCRF-CEM + 50%HEK293, 80%CCRF-CEM + 20%HEK293, 5%CCRF-CEM + 95%HEK293 and 1%CCRF-CEM + 99%HEK293 . Four mixed samples and two pure samples (100%CCRF-CEM and 100%HEK293) were each measured three times on 31 biomarkers through NanoString Technologies.

The aggregated measurement of each sample was obtained by taking the mean of three replicates. We excluded biomarkers not highly correlated with true proportions to get a total 17 biomarkers from 31. The expression of each mixed samples was combined to form Y. The expression profile matrix C was constructed by combining expressions of pure Brain sample and pure HR sample. The background vector and slope parameter were assumed to be 0 and 1. The CV was specified as 0.2, as what we estimated from the ERCC sample. The correlation of two pure samples is calculated to be 0.903.

2.4.2 Results

For Brain and HR mix samples, the proportion estimates and the relative deviance are shown in Table 2.2 and Table 2.3. Protein mixed sample results are shown in the Table 2.4 and Table 2.5.

For both samples, we see some deviance of estimates from the truth which we did not see in the simulated results. There are four major potential reasons causing such deviance:

1. **Improper normalization of the C matrix**

2. **Improper specification of background vector**

3. **Correlation between cell types**

4. **Insufficient number of biomarkers (n) relative to the number of cell types (p)**

For 1, ideally, the pure samples used to construct the C matrix should be all in the same amount in order to get the unbiased proportion estimate. Any improper or lack of normalization would lead to bias. For example, in Brain and HR mixed experiment, if k unit of pure Brain sample and m unit of pure HR sample were used to construct the C, the estimated proportion of Brain would be $\frac{1}{k}$ times of the true, and the estimated proportion of HR would be $\frac{1}{m}$ times of the true. For the ease of illustration, we name those k and m as amplification factors. This kind of bias introduced by improper normalization of C can be reduced by forcing proportions to add up to 1 when amplification factors are equal ($k = m$) across pure samples. However, in the sample preparation stage, the amplification factor is a latent variable and can vary with samples.

For 2, generally, misspecification of any parameters used to perform the algorithm would lead to bias. Misspecification of background is more of an issue when biomarkers are expressed at the low level. This is when the background noise is much bigger than the signal itself. Misspecification of CV is also an issue since the constant CV property does not hold when genes are expressed at the low level as we saw in Table (2.1).

3 and 4 do not introduce bias but increase the variance of the estimate. A high correlation between cell types can be thought as small number of effective biomarkers. If the effective sample size is smaller than the number of cell types, we have no way to deconvolve the

mixed sample. Given the fact that CCRF-CEM and the HEK293 are highly correlated with correlation 0.903 for selected biomarkers, the number of effect biomarkers is very likely not sufficient to give a precise estimate. The correlation can be reduced by selecting good biomarkers.

2.5 Discussion

The cell deconvolution method developed in this thesis performs well when the effective sample size is large even with some correlation between cell types. The discussion in the simulation study can serve as guidance for sample preparation in any future deconvolution study. Technicians could consider increasing the number of biomarkers if cell types are highly correlated.

We only looked at the correlation between cell types and assumed biomarkers to be independent in the simulation study. Future work could extend this to a framework where biomarkers are correlated. We also assumed the background noise vector to be estimated precisely. It would also be of interest to estimate background level (either jointly as part of the optimization problem, or prior to deconvolution).

Table 2.1: Coefficient of variation

NTC[1]	CV[2]	Mean	SD[3]
0.00	60.35%	11.89	7.18
0.12	19.94%	98.31	19.60
0.50	18.15%	182.38	33.11
2.00	15.32%	854.01	130.85
8.00	15.37%	4233.89	650.55
32.00	15.17%	17541.32	2661.70
128.00	16.34%	57140.12	9338.50

[1] Nominal Target Concentration

[2] Coefficient of Variation:=$\frac{\text{SD}}{\text{Mean}}$

[3] Standard Deviation

Table 2.2: Estimated proportions of Brain HR mixed samples

	$Sample_1$	$Sample_2$
α^1	(0.25, 0.75)	(0.75,0.25)
$\hat{\alpha}^2$	(0.341, 0.708)	(0.778, 0.391)
$\hat{\alpha}^{\star 3}$	(0.325, 0.675)	(0.665, 0.335)

[1] the true proportion.

[2] the estimated proprotion.

[3] the normalization (proportions add up to 1) of $\hat{\alpha}$.

Table 2.3: Relative deviance of proportion estimates from true for Brain HR mixed samples

	$Sample_1$	$Sample_2$
α[1]	(0.25, 0.75)	(0.75,0.25)
Relative deviance of $\hat{\alpha}$[2]	(36.59%, -5.59%)	(3.71%, 56.55%)
Relative deviance of $\hat{\alpha}^{\star}$[3]	(30.14%, -10.05%)	(-11.30%, 33.89%)

[1] the true proportion.

[2] $(\hat{\alpha} - \alpha)/\alpha * 100$.

[3] $(\hat{\alpha}^{\star} - \alpha)/\alpha * 100$.

Table 2.4: Estimated proportions of protein mixed samples

	$Sample_1$	$Sample_2$	$Sample_3$	$Sample_4$
α^1	(0.50, 0.50)	(0.20, 0.80)	(0.05, 0.095)	(0.01, 0.99)
$\hat{\alpha}^2$	(0.387, 0.657)	(0.140, 0.872)	(0.032, 0.841)	(0.004, 0.814)
$\hat{\alpha}^{\star 3}$	(0.371, 0.629)	(0.138, 0.862)	(0.037, 0.963)	(0.005, 0.995)

[1] the true proportion.

[2] the estimated proprotion.

[3] the normalization (proportions add up to 1) of $\hat{\alpha}$.

Table 2.5: Relative deviance for protein mixed samples

	$Sample_1$	$Sample_2$	$Sample_3$	$Sample_4$
α[1]	(0.50, 0.50)	(0.20, 0.80)	(0.05, 0.095)	(0.01, 0.99)
Relative deviance of $\hat{\alpha}$ [2]	(-22.6%, +31.4%)	(-30.2%, +8.95%)	(-35.8%, -11.5%)	(-55.7%, -17.7%)
Relative deviance of $\hat{\alpha}^{\star}$[3]	(-25.9%, +25.9%)	(-31.0%, +7.74%)	(-26.5%, 1.39%)	(-45.9%, +10.5%)

[1] the true proportion.

[2] $(\hat{\alpha} - \alpha)/\alpha * 100$.

[3] $(\hat{\alpha}^{\star} - \alpha)/\alpha * 100$.

BIBLIOGRAPHY

[1] Alexander Abbas, Kristen Wolslegel, Zora Modrusan, and Hilary Clark. Deconvolution of blood microarray data identifies cellular activation patterns in systemic lupus erythematosus. *PLoS One*, 4(7), July 2009.

[2] Mariette Awad and Rahul Khanna. *Support Vector Regression*, pages 67–80. Apress, Berkeley, CA, 2015.

[3] A. Belloni, V. Chernozhukov, and L. Wang. Square-root lasso: pivotal recovery of sparse signals via conic programming. *Biometrika*, 98(4):791–806, 2011.

[4] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.

[5] Emmanuel J Candes and Yaniv Plan. Matrix completion with noise. *Proceedings of the IEEE*, 98(6):925–936, 2010.

[6] Emmanuel J. Candes and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717, Apr 2009.

[7] A. L. Chistov and D. Yu. Grigor'ev. Complexity of quantifier elimination in the theory of algebraically closed fields. In M. P. Chytil and V. Koubek, editors, *Mathematical Foundations of Computer Science 1984*, pages 17–31, Berlin, Heidelberg, 1984. Springer Berlin Heidelberg.

[8] Bo-Yu Chu, Chia-Hua Ho, Cheng-Hao Tsai, Chieh-Yen Lin, and Chih-Jen Lin. Warm start for parameter selection of linear classifiers. In *Proceedings of the 21th ACM*

SIGKDD International Conference on Knowledge Discovery and Data Mining, pages 149–158. ACM, 2015.

[9] Gary K Geiss, Roger E Bumgarner, Brian Birditt, Timothy Dahl, Naeem Dowidar, Dwayne L Dunaway, H Perry Fell, Sean Ferree, Renee D George, Tammy Grogan, et al. Direct multiplexed measurement of gene expression with color-coded probe pairs. *Nature Biotechnology*, 26(3):317, 2008.

[10] Gary K Geiss, Roger E Bumgarner, Brian Birditt, Timothy Dahl, Naeem Dowidar, Dwayne L Dunaway, H Perry Fell, Sean Ferree, Renee D George, Tammy Grogan, et al. Direct multiplexed measurement of gene expression with color-coded probe pairs. *Nature Biotechnology*, 26(3):317, 2008.

[11] Ting Gong, Nicole Hartmann, Isaac S. Kohane, Volker Brinkmann, Frank Staedtler, Martin Letzkus, Sandrine Bongiovanni, and Joseph D. Szustakowski. Optimal deconvolution of transcriptional profiling data using quadratic programming with application to complex clinical blood samples.(research article). *PLoS ONE*, 6(11), November 2011.

[12] Geoff Gordon and Ryan Tibshirani. Gradient descent revisited. *Optimization*, 10(725/36):725, 2012.

[13] Edouard Grave, Guillaume Obozinski, and Francis Bach. Trace lasso: a trace norm regularization for correlated designs. September 2011.

[14] Mohamed Hebiri and Johannes Lederer. How correlations influence lasso prediction? *IEEE Transactions on Information Theory*, 59(3):1846–1854, 2013.

[15] Johannes Lederer and Christian Muller. Don't fall for tuning parameters: Tuning-free variable selection in high dimensions with the TREX. April 2014.

[16] B Li, Js Liu, and Xs Liu. Revisit linear regression-based deconvolution methods for tumor gene expression data. *Genome Biology*, 18(1), July 2017.

[17] David A. Liebner, Kun Huang, and Jeffrey D. Parvin. Mmad: microarray microdissection with analysis of differences is a computational tool for deconvoluting cell type-specific contributions from tissue samples. *Bioinformatics*, 30(5):682–689, March 2014.

[18] Shahin Mohammadi, Neta Zuckerman, Andrea Goldsmith, and Ananth Grama. A critical survey of deconvolution methods for separating cell types in complex tissues. *Proceedings of the IEEE*, 105(2):340–366, February 2017.

[19] Sahand Negahban and Martin J. Wainwright. Restricted strong convexity and weighted matrix completion: Optimal bounds with noise. 13:1665–1697, 2012.

[20] Aaron M Newman, Chih Long Liu, Michael R Green, Andrew J Gentles, Weiguo Feng, Yue Xu, Chuong D Hoang, Maximilian Diehn, and Ash A Alizadeh. Robust enumeration of cell subsets from tissue expression profiles. *Nature Methods*, 12(5), March 2015.

[21] Neal Parikh, Stephen Boyd, et al. Proximal algorithms. *Foundations and Trends® in Optimization*, 1(3):127–239, 2014.

[22] Wenlian Qiao, Gerald Quon, Elizabeth Csaszar, Mei Yu, Quaid Morris, and Peter W Zandstra. Pert: A method for expression deconvolution of human blood samples from varied microenvironmental and developmental conditions (pert: A flexible expression deconvolution method). 8(12), December 2012.

[23] Julien Racle, Kaat de Jonge, Petra Baumgaertner, Daniel E Speiser, David Gfeller, and Alfonso Valencia. Simultaneous enumeration of cancer and immune cell types from bulk tumor gene expression data. *eLife*, 6.

[24] E. Raninen and E. Ollila. Scaled and square-root elastic net. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4336–4340, March 2017.

[25] B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed Minimum-Rank Solutions of Linear Matrix Equations via Nuclear Norm Minimization. *ArXiv e-prints*, June 2007.

[26] Patricia P Reis, Levi Waldron, Rashmi S Goswami, Wei Xu, Yali Xuan, Bayardo Perez-Ordonez, Patrick Gullane, Jonathan Irish, Igor Jurisica, and Suzanne Kamel-Reid. mrna transcript quantification in archival samples using multiplexed, color-coded probes. *BMC biotechnology*, 11(1):46, 2011.

[27] Sylvain Sardy, Andrew G Bruce, and Paul Tseng. Block coordinate relaxation methods for nonparametric wavelet denoising. *Journal of computational and graphical statistics*, 9(2):361–379, 2000.

[28] Shai S Shen-Orr and Renaud Gaujoux. Computational deconvolution: extracting cell type-specific information from heterogeneous samples. *Current Opinion in Immunology*, 25(5):571–578, October 2013.

[29] Tingni Sun and Cun-Hui Zhang. Scaled sparse linear regression. *Biometrika*, 99(4):879–898, 2012.

[30] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[31] Sara Van de Geer. Estimation and testing under sparsity. *Lecture Notes in Mathematics*, 2159, 2016.

[32] Margaret H Veldman-Jones, Roz Brant, Claire Rooney, Catherine Geh, Hollie Emery, Chris G Harbron, Mark Wappett, Alan Sharpe, Michael Dymond, J Carl Barrett, et al. Evaluating robustness and sensitivity of the nanostring technologies ncounter platform to enable multiplexed gene expression analysis of clinical samples. *Cancer research*, 75(13):2587–2593, 2015.

[33] Margaret H Veldman-Jones, Zhongwu Lai, Mark Wappett, Chris G Harbron, J Carl Barrett, Elizabeth A Harrington, and Kenneth S Thress. Reproducible, quantitative, and flexible molecular subtyping of clinical dlbcl samples using the nanostring ncounter system. *Clinical cancer research*, 21(10):2367–2378, 2015.

[34] Margaret H Veldman-Jones, Zhongwu Lai, Mark Wappett, Chris G Harbron, J Carl Barrett, Elizabeth A Harrington, and Kenneth S Thress. Reproducible, quantitative, and flexible molecular subtyping of clinical dlbcl samples using the nanostring ncounter system. *Clinical cancer research*, 21(10):2367–2378, 2015.

[35] Van Vu. A simple svd algorithm for finding hidden partitions. *arXiv preprint arXiv:1404.3918*, 2014.

[36] Stefan Vujović, Isidora Stanković, Miloš Daković, and Ljubiša Stanković. Comparison of a gradient-based and lasso (ista) algorithm for sparse signal reconstruction. In *Embedded Computing (MECO), 2016 5th Mediterranean Conference on*, pages 377–380. IEEE, 2016.

[37] Yi Zhong, Ying-Wooi Wan, Kaifang Pang, Lionel ML Chow, and Zhandong Liu. Digital sorting of complex tissues for cell type-specific gene expression profiles. *BMC Bioinformatics*, 14:89–89, March 2013.

[38] X. Zhou, C. Yang, H. Zhao, and W. Yu. Low-Rank Modeling and Its Applications in Image Analysis. *ArXiv e-prints*, January 2014.

[39] Neta S Zuckerman, Yair Noam, Andrea J Goldsmith, and Peter P Lee. A self-directed method for cell-type identification and separation of gene expression microarrays. *PLoS Computational Biology*, 9(8).