Canonical Correlation, an Approximation, and the Prediction of Protein Abundance

Anthony Bonner[*] Han Liu[†]

Abstract

This paper addresses a central problem of Bioinformatics and Proteomics: estimating the amounts of each of the thousands of proteins in a cell culture or tissue sample. Although laboratory methods involving isotopes have been developed for this problem, we seek a simpler method, one that uses fewer laboratory procedures. Specifically, our aim is to use data-mining methods to infer protein levels from the relatively cheap and abundant data available from high-throughput tandem mass spectrometry (MS/MS). In this paper, we develop and evaluate a method for tackling this problem. The method is based on a simple generative model of MS/MS data. We first show how to linearize the model and fit it to data using Canonical Correlation Analysis (CCA). Then, because CCA is computationally expensive for the large datasets we are dealing with, we develop an efficient approximation of CCA, one that exploits the structure of our data. We prove that the method is correct in that it achieves a well-defined optimization criterion. We also evaluate the method on several biological datasets. The datasets themselves were generated by MS/MS experiments performed on various tissue samples taken from Mouse.

keywords: Bioinformatics, Proteomics, Data Mining, Machine Learning, Peptides, Tandem Mass Spectrometry.

1 Introduction

Proteomics is the large-scale study of the thousands of proteins in a cell [9]. In a typical Proteomics experiment, the goal might be to compare the proteins present in a certain tissue under different conditions. For instance, a biologist might want to study cancer by comparing the proteins in a cancerous liver to the proteins in a healthy liver. Modern mass spectrometry makes this possible by enabling the identification of thousands of proteins in a complex mixture [13, 3]. However, *identifying* proteins is only part of the story. It is also important to *quantify* them, that is, to estimate how much of each protein is present in a cell [1, 5]. To this end, a number of laboratory methods have been developed, notably those based on mass tagging with isotopes [4, 12]. However, simpler, more-direct methods may be possible, methods that do not require additional laboratory procedures, but which are simply based on the data provided by tandem mass spectrometers [10]. This paper is an initial exploration of this possibility. In particular, we investigate the use of data-mining techniques to infer protein quantity from tandem mass spectrometry data.

1.1 Tandem Mass Spectrometry Tandem mass spectrometry involves several phases in which proteins are broken up and the pieces separated by mass [9, 13]. First, a complex mixture of thousands of unknown proteins is extracted from a cell culture or tissue sample. Since proteins themselves are too large to deal with, they are fragmented, producing a mixture of tens of thousands of unknown peptides. The peptides are then ionized and passed through a mass spectrometer. This produces a mass spectrum in which each spectral peak corresponds to a peptide. From this spectrum, individual peptides are selected for further analysis. Each such peptide is further fragmented and passed through a second mass spectrometer, to produce a so-called tandem mass spectrum. The result is a collection of tandem mass spectra, each corresponding to a peptide. Each tandem mass spectrum acts as a kind of fingerprint, identifying the peptide from which it came. By searching a database of proteins, it is possible to identify the protein that produced the peptide that produced the tandem mass spectrum. In this way, the proteins in the original tissue sample are identified. Often, the entire process is completely automatic.

A peptide mixture is not analyzed all at once. Instead, to increase sensitivity, the peptides are "smeared

[*]Department of Computer Science, University of Toronto. www.cs.toronto.edu/~bonner

[†]Department of Computer Science, University of Toronto. Mr. Liu was supported by the University of Toronto.

out" over time (often using liquid chromatography), so that different kinds of peptides enter the mass spectrometer at different times. A typical MS/MS experiment may last many hours, with proteins and peptides being identified each second. Copies of a particular peptide may continue to enter the mass spectrometer for several seconds or minutes. As the copies enter, the peptide will be repeatedly identified, once a second. In this way, a peptide may be identified and re-identified many times, increasing the confidence that the identification is correct. The number of times a particular peptide is identified is called the peptide's *spectral count*, since each identification requires the generation of a tandem mass spectrum. A large spectral count indicates that the peptide has been confidently identified.

In general, as protein abundance increases, so does spectral count [10]. However, the exact relationship is not at all clear and seems to depend on many factors, including the amino acid sequence of the peptides and the properties of the experimental set up. At present, there is no complete quantitative theory relating a protein's abundance to the spectral counts of its peptides. This paper is an initial attempt at using data-mining methods to develop such a theory, and using the theory to estimate protein abundance.

1.2 Data Mining To this end, the Emili Laboratory at the Banting and Best Department of Medical Research at the University of Toronto has provided us with datasets of several thousand proteins and peptides. The datasets were derived from MS/MS experiments on protein mixtures extracted from various tissue samples of Mouse. Each mixture contains tens of thousands of proteins, and each protein is present in the mixture with a specific (but unknown) abundance. A small sample of the data is shown in Table 1. (Details on how this data was generated can be found in [8].) Each row in the table represents a peptide ion. The first (left-most) column is the Swissprot accession number identifying a protein. The second column is the amino-acid sequence of the peptide. The third column is the spectral count of the peptide, and the last column is its charge. Notice that there may be many entries for the same protein, since a single protein can produce many peptides, and each peptide can produce ions with different amounts of charge. Protein ID, Peptide and Charge define a key for the table, that is, they uniquely identify a row.

High-throughput MS/MS experiments can provide a large amount of data of this kind on which to train and test data-mining methods. However, they also introduce a complication, since the amount of protein input to the mass spectrometer is unknown. This can be seen in Table 1, where spectral count is provided, but protein abundance is not. Thus, it is in general unclear whether a low spectral count for a peptide is due to the properties of the peptide or to a small amount of protein at the input. One of the challenges is to untangle these two influences. What makes the problem approachable is that we have data on spectral counts for peptides from the *same* protein, so differences in their counts cannot be due to differences in protein abundance. The data-mining methods developed and tested in this paper were chosen, in part, because of their ability to exploit this information. In effect, they treat protein abundance as a latent, or hidden variable, whose value must be estimated. In addition, they lead to efficient algorithms based on well-developed operators of linear algebra (specifically, eigenvector decomposition).

Table 1: A fragment of a data file

Protein ID	Peptide	Count	Charge
Q91VA7	$TRHNNLVIIR$	4	2
Q91VA7	$KLDLFAVHVK$	3	2
...	...	...	...

The methods are based on a simple generative model of MS/MS data. Because this is an initial study, we chose the model for its simplicity and tractability, and the goal was to see how well (or poorly) it fits the data, and to quantify the error. The model predicts the spectral count of a peptide based on two factors: its amino-acid sequence, and the abundance of the protein from which it was derived. We show that the model provides an explanation for the linear relationship between protein abundance and spectral count observed in [10]. More importantly, we show how to use the model to estimate protein abundance from spectral count.

Although the model is non-linear in the unknowns, it can be linearized without difficulty (Section 2.3). In its linearized form, the model can be fit to data using Canonical Correlation Analysis (CCA), a well-known statistical procedure that measures the linear relationship between two random vectors [7]. However, while we have found that CCA is quite adequate for small datasets, we have also found that it is computationally expensive for the large datasets we are dealing with. As an alternative, this paper develops an efficient algorithm for an approximation of CCA, one that avoids the need to deal with large, dense matrices. We show that the algorithm is correct for data of a certain form, which includes the kind of data we are dealing with. Finally, we evaluate the method on the real-world datasets provided by the Emili Laboratory, and compare its performance to CCA.

The rest of this paper is organized as follows. Section 2 shows how to use CCA to estimate protein quantity. Section 3 discusses the computational bottlenecks of CCA and develops our approximate method. Section 4 describes our experiments on real-world data, and presents and discusses our experimental results. Finally, Section 5 presents conclusions and suggests possible extensions. In addition, the appendix provides a proof that our approximation method is correct, *i.e.*, that it computes a well-defined approximate optimization criterion.

2 Using CCA to Mine MS/MS Data

This section first reviews Canonical Correlation Analysis (CCA), then presents our model of MS/MS data, and finally shows how to use CCA to fit the model to experimental data.

2.1 Canonical Correlation Analysis (CCA) Canonical correlation analysis (CCA), first developed by Hotelling [7], is a way of measuring the linear relationship between two multidimensional random variables, $\mathbf{X}$ and $\mathbf{Y}$. In this paper, we are interested in finding the largest of the so-called canonical correlations. This can be defined as finding a linear combination of the $\mathbf{X}$ variables and a linear combination of the $\mathbf{Y}$ variables that are maximally correlated. More formally, if we treat $\mathbf{X}$ and $\mathbf{Y}$ as column vectors, then we want to find two other column vectors, α and β, such that the correlation coefficient between $x = \mathbf{X}^T\alpha$ and $y = \mathbf{Y}^T\beta$ has maximal magnitude. We therefore want to maximize the magnitude of the following expression:

$$\begin{aligned} \rho &= \frac{E[(x-Ex)(y-Ey)]}{\sqrt{E(x-Ex)^2\, E(y-Ey)^2}} \\ &= \frac{E[\alpha^T(\mathbf{X}-E\mathbf{X})(\mathbf{Y}-E\mathbf{Y})^T\beta]}{\sqrt{E[\alpha^T(\mathbf{X}-E\mathbf{X})]^2\, E[(\mathbf{Y}-E\mathbf{Y})^T\beta]^2}} \qquad (2.1) \\ &= \frac{\alpha^T\mathbf{C}_{xy}\beta}{\sqrt{\alpha^T\mathbf{C}_{xx}\alpha\beta^T\mathbf{C}_{yy}\beta}} \end{aligned}$$

Here, $\mathbf{C}_{xy} = E[(\mathbf{X} - E\mathbf{X})(\mathbf{Y} - E\mathbf{Y})^T]$ is the covariance matrix of $\mathbf{X}$ and $\mathbf{Y}$.

Maximizing this expression leads to the following generalized eigenvalue equations:

$$(2.2) \qquad \begin{cases} \rho^2\mathbf{C}_{xx}\beta = \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx}\beta \\ \rho^2\mathbf{C}_{yy}\alpha = \mathbf{C}_{yx}\mathbf{C}_{xx}^{-1}\mathbf{C}_{xy}\alpha \end{cases}$$

The eigenvalue, ρ^2, is the square of the correlation coefficient whose magnitude we want to maximize. We should therefore choose the eigenvectors, α and β, with the largest eigenvalue. In other applications, eigenvectors and eigenvalues other than the maximum can be of interest. In general, the eigenvalues of these equations are known as the squared *canonical correlations* of $\mathbf{X}$ and $\mathbf{Y}$, and the eigenvectors are the canonical correlation *basis vectors.*

2.2 Modeling Spectral Counts This section presents our model of MS/MS data. The model represents a hypothesis about the way MS/MS data is generated. As mentioned above, because this is an initial study, the model was chosen largely for its simplicity and computational tractability. Section 4 evaluates the model on real MS/MS data.

To keep track of different proteins and peptides, we use two sets of indices, usually i for proteins and j for peptides. Proteins are numbered from 1 to N, and the peptides for the i^{th} protein are numbered from 1 to n_i. In addition, we use y to denote spectral count, and z to denote protein abundance. Each protein has a unique abundance, and each peptide has a unique spectral count. We therefore use z_i to denote the abundance of protein i, and y_{ij} to denote the spectral count of peptide j of protein i. With this notation, the following equation provides a simple model of spectral count:

$$(2.3) \qquad y_{ij} = z_i \cdot e_{ij}$$

This equation divides spectral count into two factors: z_i, the amount of protein from which peptide ij was generated; and e_{ij}, the *ionization efficiency* of the peptide. Ionization efficiency can be thought of as the propensity of the peptide to ionize and contribute to a peak, though it includes all factors that contribute to spectral count *other than* the amount of protein. In this way, we hope to untangle the amount of protein (which we want to estimate) from all other factors. Note that y_{ij} is observed, while z_i and e_{ij} are both unknown.

Of course, Equation 2.3 is not exact. It provides at best an approximate description of the data, and it is not yet clear what the errors look like. The rest of this paper spells out this model in greater detail, fits it to real-world data, quantifies the error, and estimates values for z_i and e_{ij} in the process.

It is worth noting that the model already accounts for an experimentally observed property of MS/MS data. Specifically, the abundance of a protein is directly proportional to the total spectral count of its peptides [10]. Formally,

$$z_i \;=\; b_i \sum_j y_{ij}$$

where b_i is an (unknown) proportionality constant that depends on the protein. The notion of ionization efficiency provides an explanation for this proportionality and a way of computing the constants b_i. In particular, it follows immediately from Equation 2.3 that

$$z_i \;=\; \frac{\sum_j y_{ij}}{\sum_j e_{ij}}$$

In other words, $b_i = 1/\sum_j e_{ij}$. Thus, according to the model of Equation 2.3, learning ionization efficiencies, e_{ij}, is the central problem in estimating protein abundance.

It should also be noted that with the model and data described above, we can only learn *relative* values of protein abundance and ionization efficiency, not absolute values. This is because any solution to Equation 2.3 is only unique up to a constant: multiplying all the z_i by a constant, and dividing all the e_{ij} by the same constant gives another, equally good solution. However, inferring relative protein abundance would be an extremely useful biological result. Moreover, by using a small amount of calibration data, the relative values can all be converted to absolute values.

In order to estimate relative values for these unknowns, we need a model of ionization efficiency. In this paper, we investigate the use of linear models, that is, models of the following form:

$$e_{ij} \;=\; x_{ij} \bullet \beta \tag{2.4}$$

Here, β is a vector of parameters (to be learned), x_{ij} is a vector of (known) peptide properties, and $\bullet$ denotes the dot product (or inner product) of the two vectors. The peptide properties could include such things as length, mass, charge, amino-acid composition, and estimates of various biochemical properties such as hydrophobicity. Section 4 spells out the specific properties used in this study.

Combining Equations 2.3 and 2.4 gives

$$y_{ij} \;=\; z_i \cdot (x_{ij} \bullet \beta) \tag{2.5}$$

Here, β is a parameter vector of our model, and z_i, the amount of protein, is a variable. Since the amounts of protein are unknown, each z_i is a latent, or hidden variable, whose values must be estimated. Note that if the values of z_i were known (*i.e.*, if they were included in the training data), then estimating a value for β would be a straightforward problem of multivariate linear regression. Unfortunately, the training data does *not* include this information, as can be seen in Table 1. This makes the model non-linear in the unknowns.

Fortunately, it is not hard to transform this non-linear model into a linear one. We simply divide both sides of the Equation 2.5 by z_i, to give the following linear equations:

$$y_{ij} \cdot \alpha_i \;=\; x_{ij} \bullet \beta \tag{2.6}$$

where $\alpha_i = 1/z_i$ is unknown. This, then, is our final model. Like Equation 2.3, it is an approximation, and our goal is to see how closely we can fit it to the data. Since the model is linear, we can use linear fitting methods, such as CCA.

2.3 Fitting the Model with CCA Since both sides of Equation 2.6 contain unknowns, we cannot estimate their values by minimizing the total error, as in linear regression. This is because the error is trivially minimized to 0 by setting $\alpha_i = 0$ and $\beta = 0$, which is clearly incorrect. However, we can choose values for α_i and β that maximize the correlation coefficient between the two sides of the equation. This is what CCA does.

To apply CCA we first express the problem in matrix notation. To this end, we construct the following sparse $N \times M$ matrix, $\mathbf{Y}$:

$$\left(\begin{array}{ccc|ccc|c|cc}
y_{11} & \cdots & y_{1n_1} & 0 & \cdots & 0 & \cdots & 0 & \cdots \\
0 & \cdots & 0 & y_{21} & \cdots & y_{2n_2} & \cdots & 0 & \cdots \\
 & \cdots & & & \cdots & & & & \cdots \\
0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & 0 & \cdots \\
0 & \cdots & 0 & 0 & \cdots & 0 & \cdots & y_{N1} & \cdots
\end{array}\right)$$

where N is the number of proteins, $M = \sum_i n_i$ is the total number of peptides, and y_{ij} is the spectral count of peptide j of protein i. We also construct the following $p \times M$ matrix:

$$\mathbf{X} \;=\; (x_{11}, ..., x_{1n_1}, x_{21}, ..., x_{2n_2}, ..., x_{N1}, ..., x_{Nn_N})$$

where each x_{ij} is a column vector of length p, the vector of peptide properties defined above. Finally, in addition to these two (known) matrices, $\mathbf{X}$ and $\mathbf{Y}$, we define a new (unknown) column vector, $\alpha = (\alpha_1, ..., \alpha_N)^T$, where $\alpha_i = 1/z_i$. With these definitions, Equation 2.6 can be rewritten as:

$$\mathbf{Y}^T \alpha \;=\; \mathbf{X}^T \beta \tag{2.7}$$

Our aim is now to estimate values for α and β. From the value of α, we can easily estimate an input amount for each protein, using $\hat{z}_i = 1/\alpha_i$; and from the value of β, we can easily estimate an ionization efficiency for any peptide, using $\hat{e}_{ij} = \mathbf{x}_{ij} \bullet \beta$.

We use CCA to find values for α and β that maximize the sample correlation coefficient between the two random vectors $\mathbf{Y}^T\alpha$ and $\mathbf{X}^T\beta$. From Equations 2.2, we need the sample covariance matrices of $\mathbf{X}$ and $\mathbf{Y}$. These are given by the following matrix equations:

$$\begin{aligned}
\mathbf{C}_{xx} &= (\mathbf{X}-\overline{\mathbf{X}})(\mathbf{X}-\overline{\mathbf{X}})^T \\
\mathbf{C}_{yy} &= (\mathbf{Y}-\overline{\mathbf{Y}})(\mathbf{Y}-\overline{\mathbf{Y}})^T \\
\mathbf{C}_{xy} &= (\mathbf{X}-\overline{\mathbf{X}})(\mathbf{Y}-\overline{\mathbf{Y}})^T \\
\mathbf{C}_{yx} &= (\mathbf{Y}-\overline{\mathbf{Y}})(\mathbf{X}-\overline{\mathbf{X}})^T
\end{aligned}$$

Here, $\overline{\mathbf{X}}$ is a matrix of sample means. That is, element ij in matrix $\overline{\mathbf{X}}$ is the average of all the elements in row i of matrix $\mathbf{X}$ (*i.e.*, the sample mean of peptide feature i). Likewise for $\overline{\mathbf{Y}}$. With these covariance matrices, Equations 2.2 give the values of α and β that maximize the correlation coefficient.

Although straightforward, we have found that solving Equations 2.2 is computationally expensive, both in terms of time and space. The main problem is the large size of the matrix $\mathbf{Y}$. The data we are dealing with contains roughly 10,000 different peptides and 2,000 different proteins. Matrix $\mathbf{Y}$ therefore has dimensions $2,000 \times 10,000$, and so the covariance matrix $\mathbf{C}_{yy}$ has dimensions $2,000 \times 2,000$. The first of Equations 2.2 requires inverting this matrix. However, inverting such a large matrix requires considerable time and space, and can lead to severe numerical problems [14]. The second of Equations 2.2 requires inverting the covariance matrix $\mathbf{C}_{xx}$, which is not nearly so large and can easily be inverted. However, on the left side of the equation, we again encounter the large matrix $\mathbf{C}_{yy}$, which makes the generalized eigenvector equation costly to solve. In addition, since $\mathbf{C}_{yy}$ is a $N \times N$ matrix, where N is the number of proteins, the cost of this method increases rapidly with the number of proteins in the dataset.

3 An Efficient Approximation

Although it is very large, the matrix $\mathbf{Y}$ defined in the previous section is also very sparse. In every column, only one element is non-zero. By exploiting the structure and sparseness of this matrix, we develop an efficient algorithm for an approximation of CCA.

In CCA, the statistical measure of similarity between two random vectors is *correlation coefficient.* Formally, we want to find α and β that maximize the following expression:

$$(3.8) \qquad \frac{(\mathbf{Y}^T\alpha - \overline{\mathbf{Y}^T\alpha}) \bullet (\mathbf{X}^T\beta - \overline{\mathbf{X}^T\beta})}{\|\mathbf{Y}^T\alpha - \overline{\mathbf{Y}^T\alpha}\| \cdot \|\mathbf{X}^T\beta - \overline{\mathbf{X}^T\beta}\|}$$

The point to notice here is that the two vectors are centered, by subtracting their means. The correlation coefficient is thus the cosine of the angle between the two centered vectors, and CCA finds α and β to minimize this angle. Unfortunately, from a computational point of view, the centering of the vectors causes a great deal of problems, because it effectively destroys the sparse structure of matrix $\mathbf{Y}$. This is because the large covariance matrix, $\mathbf{C}_{yy}$, is defined not in terms of $\mathbf{Y}$, but in terms of $\mathbf{Y}-\overline{\mathbf{Y}}$. Unfortunately, although $\mathbf{Y}$ is sparse, $\overline{\mathbf{Y}}$ is dense, so $\mathbf{Y}-\overline{\mathbf{Y}}$ is also dense. In fact, in row i of matrix $\overline{\mathbf{Y}}$, each entry is $\sum_{j=1}^{n_i} y_{ij}/N$, so $\overline{\mathbf{Y}}$ has no zeros and is maximally dense. Likewise for $\mathbf{Y}-\overline{\mathbf{Y}}$.

In the approximation method developed here, we retain the idea of minimizing the angle, but without the requirement of centering the vectors first. That is, we minimize the angle between the uncentered vectors $\mathbf{Y}^T\alpha$ and $\mathbf{X}^T\beta$, by maximizing the cosine of the angle between them. This amounts to maximizing the following expression:

$$(3.9) \qquad \frac{\mathbf{Y}^T\alpha \bullet \mathbf{X}^T\beta}{\|\mathbf{Y}^T\alpha\| \cdot \|\mathbf{X}^T\beta\|}$$

Maximizing this expression leads to the same generalized eigenvector equations given in Equation 2.2, except that now the covariance matrices are defined in terms of uncentered random variables. Thus, instead of $\mathbf{C}_{yy} = (\mathbf{Y}-\overline{\mathbf{Y}})(\mathbf{Y}-\overline{\mathbf{Y}})^T$, we now use $\mathbf{C}_{yy} = \mathbf{Y}\mathbf{Y}^T$.

Of course, this does not change the dimensions of any of the covariance matrices. In particular, $\mathbf{C}_{yy}$ is still very large. However, it is now possible to simplify Equations 2.2 so that the remaining matrices are relatively small. This is shown in Theorem 1 below. In this theorem, Y_i is the column vector $(y_{i1}, y_{i2}, ..., y_{in_i})^T$, and $\mathbf{X}_i$ is the matrix $(x_{i1}, x_{i2}, ..., x_{in_i})$. They represent, respectively, the spectral counts and peptide properties for protein i. Note that $\mathbf{X} = (\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_N)$, so each $\mathbf{X}_i$ is a vertical slice of the larger matrix $\mathbf{X}$.

Theorem 1: *Expression 3.9 above is maximized when the parameter vector β is a solution of the following generalized eigenvector equation:*

$$(3.10) \qquad \rho^2 \boldsymbol{X}\boldsymbol{X}^T\beta \;=\; [\sum_i \boldsymbol{X}_i Y_i Y_i^T \boldsymbol{X}_i^T / \|Y_i\|^2]\,\beta$$

Moreover, it is the eigenvector with the largest eigenvalue, ρ^2. In addition, $\rho = \cos\theta$, where θ is the angle *between the vectors $\boldsymbol{Y}^T\alpha$ and $\boldsymbol{X}^T\beta$. Finally, the elements of the parameter vector α are given by the following equation:*

$$\alpha_i \;=\; \boldsymbol{Y}_i^T \boldsymbol{X}_i^T \beta / \rho \|Y_i\|^2 \tag{3.11}$$

Proof: Given in the appendix.

Comparing Equation 3.10 in this theorem with Equation 2.2 in Section 2.1, the important point is that we no longer need to compute or invert the large covariance matrix $\mathbf{C}_{yy}$. As for the other matrices, $\mathbf{X}_i$ has dimensions $p \times n_i$, where p is the number of properties (or features) characterizing each peptide. Matrix $\mathbf{X}$ has dimensions $p \times M$, where $M = \sum_i n_i$ is the total number of peptides. The matrices $\mathbf{X}\mathbf{X}^T$ and $\mathbf{X}_i Y_i Y_i^T \mathbf{X}_i^T$ thus have dimensions $p \times p$. It is the size of these $p \times p$ matrices that is important, since this determines the cost of solving eigenvector equation 3.10.

Since our datasets contain roughly 10,000 peptides, M is large. However, p, the number of peptide features is much smaller. In this paper, we use two different feature sets, one with $p = 21$ and one with $p = 421$, as described in Section 4. With $p = 21$ the eigenvector matrices are very small. With $p = 421$ they are much larger, and although they slowed down the eigenvector computations discernably, they still posed no significant problems. Moreover, they are considerably smaller than the covariance matrix $\mathbf{C}_{yy}$ in Equation 2.2, whose dimensions are roughly 2000×2000, which did cause significant computational problems. Perhaps more importantly, the size of the $p \times p$ matrices is independent of how many proteins or peptides are in the dataset. Thus, computational cost will not be significantly affected as the datasets grow.

3.1 Weighting the Data In fitting a model to data, one may not wish to treat all data points equally, but to place different importance on different data. To allow for this, one can introduce weights into the optimization criterion. For our approximation to CCA, the optimization criterion is given by Equation 3.9, which specifies the angle between two vectors. In this case, we can define what might be called a generalized angle, in which different vector components are weighted differently. If the two vectors are denoted U and V, then the generalized angle is defined by (the arc cosine of) the following expression:

$$\frac{U^T \mathbf{W} V}{\sqrt{U^T \mathbf{W} U}\,\sqrt{V^T \mathbf{W} V}}$$

Here, $\mathbf{W}$ is a diagonal matrix, whose i^{th} diagonal element, $\mathbf{w}_i$, is the weight of the i^{th} component of the vectors. When $\mathbf{W}$ is the identity matrix, we get the ordinary, unweighted angle. To compute the generalized angle between U and V, we can use the above formula, or we can first transform the vectors as follows:

$$U_i' \;=\; U_i \sqrt{\mathbf{w}_i} \qquad\qquad V_i' \;=\; V_i \sqrt{\mathbf{w}_i}$$

and then compute the unweighted angle between U' and V'. This latter approach shows that giving weight $\mathbf{w}_i$ to data point (U_i, V_i) is equivalent to simply multiplying U_i and V_i by $\sqrt{\mathbf{w}_i}$. This makes intuitive sense, since the angle between two vectors is more strongly influenced by large vector components than by small ones.

In the case of our MS/MS data, this corresponds to assigning a different weight to each peptide and transforming its spectral count and feature vector as follows:

$$y_{ij}' \;=\; y_{ij} \sqrt{\mathbf{w}_{ij}} \qquad\qquad x_{ij}' \;=\; x_{ij} \sqrt{\mathbf{w}_{ij}}$$

where $\mathbf{w}_{ij}$ is the weight assigned to peptide j of protein i. We then apply Theorem 1 to y_{ij}' and x_{ij}', instead of to y_{ij} and x_{ij}. The choice of what weights to use is heuristic, and in our experiments, we chose two different sets of weights, $\mathbf{w}_{ij} = ||Y_i||$ and $\mathbf{w}_{ij} = 1/||Y_i||$, respectively. These weights are a simple attempt to address two different sources of noise and error. The first set of weights emphasizes peptides from proteins with high spectral counts, since they have a higher reliability and a better signal-to-noise ratio. The second set of weights attempts to stabilize the model error, assuming that peptides from proteins with larger spectral counts will tend to have larger error.

4 Experiments

This section uses real-world data to experimentally evaluate the data-mining methods and models described above. The main evaluation strategy is ten-fold cross validation, with correlation coefficient used to measure the fit of a learned model to the testing portion of the data. The main difficulty in carrying out the evaluation was the distribution of the spectral counts, which ranges over several orders of magnitude and is highly skewed, with most data concentrated at very low values. To deal with this difficulty, we use the Spearman rank correlation coefficient to measure the goodness-of-fit [2]. Unlike the more common Pearson correlation coefficient, which measures *linear* correlation, Spearman's coefficient measures *monotone* correlation and is insensitive to extreme data values. In addition, we use plots of observed v.s. estimated values to provide an informative visualization of the fit.

4.1 Study Design We evaluated the data-mining methods on three datasets derived from tissue samples taken from Mouse. Similar in form to Table 1, the datasets were provided by the Emili Laboratory at the Banting and Best Department of Medical Research at the University of Toronto. We refer to these datasets as **Mouse Brain Data**, **Mouse Heart Data**, and **Mouse Kidney Data**. In each of these datasets, many proteins have multiple entries, each entry corresponding to a different peptide. In fact, the vast majority of entries correspond to proteins of this type. However, some proteins have only one entry, since they give rise to only one observable peptide. Such proteins provide no information about protein abundance, so we remove them from the datasets. After removal, the Brain dataset contains 8,527 peptides and 1,664 proteins, Heart dataset contains 7,660 peptides and 1,281 proteins, and the Kidney dataset contains 7,074 peptides and 1,291 proteins.

For the data-mining methods developed in this paper, each peptide must be represented as a vector, x. This section evaluates two ways of doing this, using vectors with 21 features and 421 features, respectively. The vectors with 21 features represent the amino-acid composition of a peptide. Since there are twenty different amino acids, the vector has 20 features, $(x_1, ..., x_{20})$, where the value of feature x_i is the number of occurrences of a particular amino acid in the peptide. In addition, the vector has a 21^{st} feature, x_0, whose value is always 1, to represent a bias term, as is common in data-mining and machine-learning models [6]. The vectors with 421 features include the original 21 plus an additional 400 features representing the dimer composition of a peptide. A *dimer* is a sequence of two amino acids, and since there are 20 distinct amino acids, there are 400 distinct dimers.[1]

We evaluated numerous combinations of feature vector, data-mining method and weighting scheme. Due to space limitations, we present only five of them here. In addition, because of the time required to execute CCA, we used it in only one combination: unweighted and with vectors having 21 features. We also evaluated four versions of the approximate method developed in Section 3. The first two versions are both unweighted and use vectors with 21 features and 421 features, respectively. We refer to these two versions as *Approx-21* and *Approx-421*. The other two versions are both weighted and use vectors with 21 features. The two weighting schemes used are $\mathbf{w}_i = ||Y_i||$ and $\mathbf{w}_i = 1/||Y_i||$, as described in Section 3.1.

Using ten-fold cross validation, we evaluated each of these five data-mining methods on each of the three Mouse datasets. Thus, each method was trained on nine tenths of the data (the training set), and the fitted model was then evaluated on the remaining one tenth of the data (the test set), and this was repeated in ten possible ways. Each training session produced an estimate, $\hat{\beta}$, of the parameter vector β, and an estimate, $\hat{z}$, of the input amount for each protein in the training set. Using $\hat{\beta}$, we estimated the ionization efficiency of each peptide in the entire dataset, using the formula $\hat{e} = x \bullet \hat{\beta}$, where x is the vector representation of the peptide. Applying univariate linear regression to Equation 2.3, we then estimated an input amount, $\hat{z}$, for each protein in the test set. We then estimated the spectral count of each peptide in the entire dataset, using $\hat{y} = \hat{z} \cdot \hat{e}$. Finally, we compared the estimated and observed spectral counts (that is, $\hat{y}$ and y) by computing Spearman rank correlation coefficients.

The results are shown in Table 2. In this table, each column corresponds to a Mouse dataset, and each row corresponds to a data-mining method. Each position in the table shows four numbers, stacked vertically. The top two numbers are the mean and standard deviation of the correlation coefficient of $\hat{y}$ and y on the training data. The bottom two numbers are the mean and standard deviation of the correlation coefficient on the test data. (Since at this stage, we are only interested in rough estimates of correlation coefficient, the ten estimates produced by ten-fold cross validation are enough.) In addition, by dividing $\hat{y}$ and y by $\hat{z}$, we get two different estimates of ionization efficiency, which we denote $\hat{e}$ and e, respectively. The correlation coefficient between these two estimates is what CCA tries to maximize. Thus, while the correlation coefficient of $\hat{y}$ and y measures the ability of the fitted model to predict experimental observations, the correlation coefficient of $\hat{e}$ and e provides the most direct measure of fit between the model and the data. The results are shown in Table 3, which has the same format as Table 2.

4.2 Results The first point to notice is that of all the methods that use vectors with 21 features, CCA provides the best fit to the training data in Table 3. (only Approx-421 produces a better fit, and only on the Kidney data, but it uses more features.) This is

[1] In [11] we explore other peptide features, including peptide charge.

Table 2: Correlation of y and $\hat{y}$ on real data

Method	Statistics	Brain Data	Heart Data	Kidney Data
CCA	Mean Train:	0.0350	0.0273	0.0335
	Std Train:	0.0292	0.0162	0.0413
	Mean Test:	0.3694	0.2575	0.3563
	Std Test:	0.1118	0.1683	0.0865
Approx -21	Mean Train:	0.2180	0.3319	0.2850
	Std Train:	0.0356	0.0135	0.0583
	Mean Test:	0.4190	0.4250	0.4109
	Std Test:	0.1133	0.0906	0.0705
Approx -421	Mean Train:	0.0660	0.1299	0.1742
	Std Train:	0.0529	0.1631	0.0745
	Mean Test:	0.2869	0.2839	0.4090
	Std Test:	0.1975	0.1098	0.0999
Weighted Approx $w = \|y\|$	Mean Train:	0.2121	0.4004	0.3518
	Std Train:	0.0678	0.0935	0.0363
	Mean Test:	0.4225	0.4406	0.4302
	Std Test:	0.1004	0.0934	0.0951
Weighted Approx $w = \|1/y\|$	Mean Train:	0.2568	0.3811	0.3005
	Std Train:	0.0067	0.0168	0.0186
	Mean Test:	0.3924	0.4223	0.3999
	Std Test:	0.0924	0.0799	0.0586

Table 3: Correlation of e and $\hat{e}$ on real data

Method	Statistics	Brain Data	Heart Data	Kidney Data
CCA	Mean Train:	0.6027	0.5929	0.5971
	Std Train:	0.0042	0.0054	0.0040
	Mean Test:	0.1054	0.0459	0.2049
	Std Test:	0.0763	0.0486	0.0846
Approx -21	Mean Train:	0.4298	0.3921	0.4078
	Std Train:	0.0057	0.0081	0.0074
	Mean Test:	0.2759	0.2234	0.2530
	Std Test:	0.0432	0.0605	0.0485
Approx -421	Mean Train:	0.5460	0.5469	0.6080
	Std Train:	0.1998	0.2616	0.0071
	Mean Test:	0.0982	0.0913	0.2335
	Std Test:	0.1125	0.1058	0.0424
Weighted Approx $w = \|y\|$	Mean Train:	0.4613	0.3909	0.3916
	Std Train:	0.0051	0.0104	0.0135
	Mean Test:	0.2021	0.1163	0.2272
	Std Test:	0.0737	0.0788	0.0456
Weighted Approx $w = \|1/y\|$	Mean Train:	0.3118	0.2995	0.3072
	Std Train:	0.0056	0.0070	0.0093
	Mean Test:	0.1999	0.1786	0.1905
	Std Test:	0.0349	0.0528	0.0376

to be expected since CCA maximizes the correlation coefficient, which is what the table measures. On the other hand, Approx-21 provides the best fit to the test data. The same is true in Table 2, where Approx-21 provides better test predictions of y, the spectral count (the main biological observable). These results suggest that while our method may be an approximation of CCA, it may also be more appropriate for this problem, in terms of accuracy as well as speed.

The effect of the weighted methods is inconclusive. In Table 3, the unweighted Approx-21 has consistently better performance on the test data than either of the two weighted schemes, but not dramatically better. In Table 2, the three methods perform comparably on the test data, though weights of $|1/y|$ seem to be marginally best, and weights of $|y|$ seems to be marginally worst, with Approx-21 in between.

The effect of the larger feature vector is more conclusive. If we compare the Approx-21 and Approx-421 methods in Table 3, we can see that Approx-421 shows evidence of overfitting, since the fit on the testing data is often much worse than on the training data. The 400 dimer values included among the 421 features thus appear to have little predictive value. Biologically, this a useful negative result.

Table 2 shows some apparently anomalous patterns. For instance, the fit on the testing data is often better than on the training data. Also, Approx-21 has a better fit to the training data than Approx-421, even though the features used in Approx-21 are a subset of those used in Approx-421. These patterns are probably a result of comparing $\hat{y}$ and y, whereas the data-mining methods try to fit $\hat{e}$ and e. The same patterns are not present in Table 3, which compares $\hat{e}$ and e.

In addition to the measurements presented in Tables 2 and 3, Figures 1 and 2 provide a visual representation of how well the estimated models fit the data. In each figure, the horizontal axis is $\hat{e}$, the vertical axis is e, and each point represents a single peptide. Figure 1 was generated by the Approx-21 method, and Figure 2 by the Approx-421 method, with both trained on the entire Kidney dataset. The other methods generate similar figures. The first point to notice is that in both figures, the vast majority of values of $\hat{e}$ and e are positive, which is how things should be, since ionization efficiency is inherently positive. The Approx-421 method has more negative points than Approx-21, but again, this could be a result of overfitting. The second point to notice is that each figure appears to consist of two components—a fairly linear diagonal component, and a less-linear horizontal blob. This suggests that there are two populations of peptides, those whose ionization efficiency is well-modeled by a linear function, and those whose ionization efficiency is much less predictable. This would explain why the correlation coefficients in Table 3 are low. It also suggests a natural topic for future research: characterizing those peptides that can be modeled linearly.

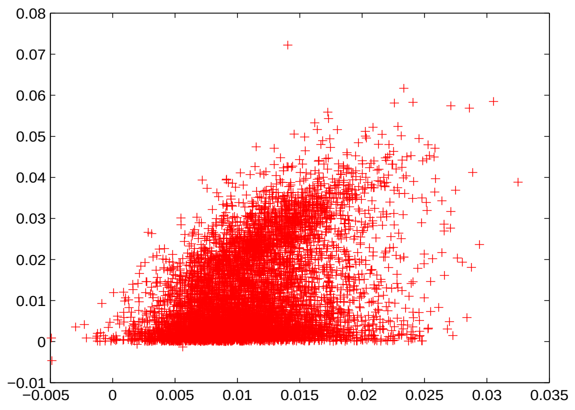

Figure 1: e vs. $\hat{e}$ as estimated by Approx-21.

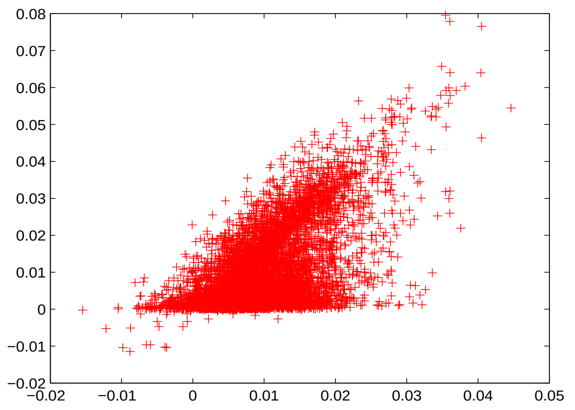

Figure 2: e vs. $\hat{e}$ as estimated by Approx-421.

5 Conclusions and Future Directions

This paper developed and evaluated a data-mining method for estimating protein levels from high-throughput Tandem Mass Spectrometry (MS/MS) data. The method is based on a simple generative model of MS/MS data. We showed how to linearize the model and fit it to data using Canonical Correlation Analysis (CCA). However, for the large data sets we are working with, we found CCA to be computationally expensive. As an alternative, we developed an efficient algorithm for an approximation to CCA, one that exploits the structure of our data.

We provided a proof of correctness of the method, and we evaluated its effectiveness on real MS/MS data derived from tissue samples of Mouse. The evaluations included three different tissue samples, two different vector representations of peptides, three different schemes for weighting the data, and three different evaluation measures (two based on correlation coefficients and one based on data visualization). The results suggest that our method may be better than CCA at fitting the model to MS/MS data. Biologically, they suggest that spectral count is not influenced by the dimers in a peptide. They also suggest that there may be two types of peptide, only one of whose ionization efficiency can be adequately modeled by a linear function.

This research is just a first step, and additional work is needed before protein levels can be predicted accurately. This includes developing non-linear models of ionization efficiency (*e.g.*, models based on regression trees, neural nets, or support vector machines), identifying those peptides for which linear models are adequate, and investigating additional peptide representations, especially ones that retain more of the sequential content of a peptide. Other possibilities include developing more sophisticated models of the entire MS/MS process, especially models that account for interactions between peptides in the mass spectrometer.

Finally, the methodology of Section 4 evaluates a model in terms of its ability to predict the spectral counts of peptides, based on peptide properties and predicted protein abundance. The ability to accurately predict spectral counts in this way would be strong evidence that a model is correct, and would suggest that protein abundance was accurately predicted. However, conclusive proof requires a more direct comparison with known protein amounts (*e.g.*, as measured by the more laboratory-intensive isotope marker experiments [4, 12]).

Acknowledgments

The authors would like to thank Dr. Andrew Emili for his comments and discussions and for generous access to his data. We would also like to thank Clement Chung for generating and preprocessing the data. Finally, we would like to thank the anonymous reviewers for their constructive and helpful comments.

References

[1] R. Aebersold, M. Mann. *Mass spectrometry-based proteomics*, Nature 422, pp 198-207, 2003.

[2] P. Bickel and K. Doksum. *Mathematical Statistics: Basic Ideas and Selected Topics*, Holden-Day INC, 1977.

[3] Joshua E Elias, Francis D Gibbons, Oliver D King, Frederick Roth, Steven P Gygi *Intensity-based protein identification by machine learning from a library of tandem mass spectra*,Nature, biotechnology. Volume 22 Number 2, 2004

[4] S.P. Gygi, B. Rist, S.A. Gerber, F. Turecek, M.H. Gelb, and R. Aebersold. *Quantitative analysis of complex protein mixtures using isotope-coded affinity tags*, Nature Biotechnology 17, pp 994–999.

[5] S.P.Gygi and R. Aeberold. *Mass Spectrometry and Proteomics*, Current Opinion in Chemical Biology, 4, pp 489–494, 2000.

[6] T. Hastie, R. Tibshirani, J. Friedman. *The elements of statistical learning—Data mining, inference and prediction*, Springer 2001.

[7] H. Hotelling. *Relations between two sets of variates.* Biometrika, 28:321-377, 1936.

[8] T. Kislinger, K. Ramin, D. Radulovic, et al. *PRISM, a Generic Large Scale Proteomics Investigation Strategy for Mammals*, Molecular & Cellular Proteomics 2.1 2003.

[9] D.C. Liebler. *Introduction to Proteomics, tools for the new biology*, Humana Press, NJ, 2002.

[10] H. Liu, R.G. Sadygov, and J.R. Yates. *A Model for Random Sampling and Estimation of Relative Protein Abundance in Shotgun Proteomics*, Anal. Chem. 76, pp 4193–4201, 2004.

[11] H. Liu. *Development and Evaluation of Methods for Predicting Protein Levels from Tandem Mass Spectrometry Data*, Masters thesis, Department of Computer Science, University of Toronto. January 2005.

[12] S. Ong, B. Blagoev, I. Kratchmarovat, D.B. Kristensen, H. Steen, A. Pandey, and M. Mann. *Stable Isotope Labelling by Amino Acid in Cell Culture, SILAC, as a Simple and Accurate Approach to Expression Proteomics*, Molecular & Cellular Proteomics 1.5 2002.

[13] G. Siuzdak. *The Expanding Role of Mass Spectrometry in Biotechnology*, Mcc Press, 2003.

[14] David S. Watkins *Fundamentals of Matrix Computation*, Wiley-Interscience 2002.

6 Appendix: Proof of Theorem 1

Observe that the maximum of expression 3.9 is the same as the maximum of the simpler expression $\mathbf{Y}^T\alpha \bullet \mathbf{X}^T\beta$ subject to the constraints $\|\mathbf{Y}^T\alpha\| = 1$ and $\|\mathbf{X}^T\beta\| = 1$. In fact, α and β maximize the unconstrained expression if and only if α' and β' maximize the constrained expression, where $\alpha' = \alpha/\|\mathbf{X}^T\alpha\|$ and $\beta' = \beta/\|\mathbf{X}^T\beta\|$. It is therefore sufficient to maximize the simpler, constrained expression. To carry this out, we use Lagrange multipliers and maximize the following expression:

$$\mathbf{Y}^T\alpha \bullet \mathbf{X}^T\beta \;-\; \lambda(\|\mathbf{Y}^T\alpha\|^2 - 1) \;-\; \mu(\|\mathbf{X}^T\beta\|^2 - 1)$$

It is not hard to see that this expression is equivalent to the following:

$$\sum_i \alpha_i Y_i^T \mathbf{X}_i^T \beta \;-\; \lambda(\sum_i \|\alpha_i Y_i\|^2 - 1) \;-\; \mu(\|\mathbf{X}^T\beta\|^2 - 1)$$

Taking partial derivatives with respect to β and α_i and setting the results to 0 gives the following equations:

$$\sum_i \alpha_i \mathbf{X}_i Y_i \;=\; 2\mu\mathbf{X}\mathbf{X}^T\beta \tag{6.12}$$

$$Y_i^T \mathbf{X}_i^T \beta \;=\; 2\lambda\alpha_i\|Y_i\|^2 \tag{6.13}$$

Left-multiplying Equation 6.12 by β^T gives

$$\beta^T \sum_i \alpha_i \mathbf{X}_i Y_i \;=\; 2\mu\beta^T\mathbf{X}\mathbf{X}^T\beta \tag{6.14}$$

$$= 2\mu\|\mathbf{X}^T\beta\|^2 \tag{6.15}$$

$$= 2\mu \tag{6.16}$$

since, by our constraint, $\|\mathbf{X}^T\beta\| = 1$. In a similar fashion, multiplying Equation 6.13 by α_i and summing over i gives

$$\sum_i \alpha_i Y_i^T \mathbf{X}_i^T \beta \;=\; 2\lambda \sum_i \alpha_i^2 \|Y_i\|^2 \tag{6.17}$$

$$= 2\lambda\|\mathbf{Y}^T\alpha\|^2 \tag{6.18}$$

$$= 2\lambda \tag{6.19}$$

since, by our constraint, $\|\mathbf{Y}^T\alpha\| = 1$. Note that Equations 6.16 and 6.19 can be rewritten as follows:

$$2\lambda \;=\; \mathbf{Y}^T\alpha \bullet \mathbf{X}^T\beta \;=\; 2\mu \tag{6.20}$$

In other words, $\lambda = \mu = \rho/2$, where ρ is the value we are maximizing. From this and Equation 6.13, it follows immediately that

$$\alpha_i \;=\; Y_i^T \mathbf{X}_i^T \beta/\rho\|Y_i\|^2 \tag{6.21}$$

This proves Equation 3.11. To prove Equation 3.10, note that from Equations 6.12 and 6.21, we get

$$2\mu\rho\mathbf{X}\mathbf{X}^T\beta \;=\; [\sum_i \mathbf{X}_i Y_i Y_i^T \mathbf{X}_i^T/\|Y_i\|^2]\,\beta$$

The result follows immediately, since $2\mu\rho = \rho^2$.