

Towards the Prediction of Protein Abundance from Tandem Mass Spectrometry Data*

Anthony J Bonner[†]

Han Liu[‡]

Abstract

This paper addresses a central problem of 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 method that uses simpler laboratory procedures. Specifically, our aim is to use data-mining techniques to infer protein levels from the relatively cheap and abundant data available from high-throughput tandem mass spectrometry (MS/MS). We have developed and evaluated several techniques for tackling this problem, including the development of three generative models of MS/MS data, and methods for efficiently fitting the models to data. In addition, we tested each method on three real-world datasets generated by MS/MS experiments performed on various tissue samples taken from Mouse. This paper outlines the biological problem and presents a selection of our results.

1 Introduction.

Proteomics is the large-scale study of the thousands of proteins in a cell [7]. 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 those in a healthy liver. Modern mass spectrometry makes this possible by enabling the identification of thousands of proteins in a complex mixture [11, 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, 4]. To this end, a number of laboratory methods have been developed, notably those based on mass tagging with isotopes [5, 10]. However, simpler methods may be possible, methods that do not require additional laboratory procedures but are simply based on the data from tandem mass spectrometers [8]. 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 [7, 11]. 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.

A peptide mixture is not analyzed all at once. Instead, to increase sensitivity, the peptides are “smeared 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. Each identification of a peptide is called a *spectral count*, since it requires the generation of a tandem mass spectrum. A large spectral count indicates that a peptide has been confidently identified.

In general, as protein abundance increases, so does spectral count [8]. However, the exact relationship is

*Supported in part by a grant from the Natural Sciences and Engineering Research Council of Canada (NSERC).

[†]Department of Computer Science, University of Toronto, bonner@cs.toronto.edu

[‡]Department of Computer Science, University of Toronto

not 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 techniques to develop such a theory, and using the theory to estimate protein abundance.

1.2 Data Mining. We develop and evaluate three generative models of MS/MS data, and for each, we develop a family of methods for efficiently fitting the model to data. Because this is an initial study, the models were chosen for their simplicity and tractability, and the goal is to see how well (or poorly) they fit the data, and to quantify the error. Each 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. The three models differ in their treatment of peptide ionization. However, they each provide an explanation for a recently observed linear relationship between protein abundance and spectral count [8]. More importantly, we show how to use each model to estimate protein abundance from spectral count.

To evaluate the models, 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 [6].) 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.

Table 1: A fragment of a data file

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

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 models and methods developed here were chosen, in part, because of their ability to exploit this information. In addition, they lead to efficient algorithms based on well-developed operators of linear algebra.

The rest of this paper outlines our methods and experimental results. Details are available in a technical report [2].

2 Modeling the Data.

This sections presents our three models of MS/MS data. Each model represents a different hypothesis about the way MS/MS data is generated, and the main difference between them is their treatment of peptide ionization. Section 4 evaluates the effectiveness of the models using the datasets supplied by the Emili laboratory.

2.1 Modeling Spectral Counts. 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 in to denote the amount of protein input to the mass spectrometer. Each protein has a unique abundance, and each peptide has a unique spectral count. Thus, in_i is the abundance of protein i , and y_{ij} is the spectral count of peptide j of protein i . With this notation, the following equation provides a simple model of spectral count:

$$(2.1) \quad y_{ij} = in_i \cdot ie_{ij}$$

This equation divides spectral count into two factors: in_i , the amount of protein from which peptide ij was generated; and ie_{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 mass spectrum, though in this paper, we are using it to refer to all factors that contribute to a peptide’s 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 in_i and ie_{ij} are both unknown. Of course, Equation 2.1 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 the paper develops the model in three different ways, fits each model to real MS/MS data, quantifies the errors, and estimates values for in_i and ie_{ij} in the process.

It should 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 2.1 is unique only up to a constant: multiplying all the in_i by a constant, and dividing all the ie_{ij} by the same constant gives another, equally good solution. However, estimating the relative amounts of protein is an extremely useful biological result. Moreover, by using a small amount of calibration data, the relative values can all be converted to absolute values.

It is also worth noting that the model already accounts for an experimentally observed property of MS/MS data. Specifically, the abundance of a protein is known to be directly proportional to the total spectral count of its peptides [8]. Formally, $in_i = b_i \sum_j y_{ij}$, where b_i is an (unknown) proportionality constant that depends on the protein. Our model provides an explanation for this proportionality and a way of computing the constants b_i . In particular, it follows immediately from Equation 2.1 that

$$(2.2) \quad in_i = \frac{\sum_j y_{ij}}{\sum_j ie_{ij}}$$

In other words, $b_i = 1 / \sum_j ie_{ij}$.

2.2 Modeling Ionization Efficiency. In order to estimate relative values of protein abundance, we need a model of the ionization efficiencies, ie_{ij} . In this paper, we investigate three relatively simple models:

$$(2.3) \quad \begin{array}{ll} \text{Linear :} & ie_{ij} = \mathbf{x}_{ij} \bullet \beta \\ \text{Exponential :} & ie_{ij} = e^{\mathbf{x}_{ij} \bullet \beta} \\ \text{Inverse :} & ie_{ij} = 1 / (\mathbf{x}_{ij} \bullet \beta) \end{array}$$

Here, β is a vector of parameters (to be learned), \mathbf{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 are all derived from the amino-acid sequence and charge of the peptide ion. They could include such things as length, mass, amino-acid composition, and estimates of various biochemical properties such as hydrophobicity, chargeability, pH under the experimental conditions, etc. Section 4 describes the specific properties used in this study.

We investigate linear models because they are directly amenable to the techniques of linear algebra. We investigate exponential models because, by taking logs, they become linear. In addition, exponential models have the advantage that the ionization efficiency is guaranteed to be positive.

The inverse model has a different motivation. Spectral counts have a very skewed distribution of values, ranging over several orders of magnitude, with most of the values concentrated at the very low end of the spectrum. In fact, as we show in [2], the distribution is $O(1/y^2)$, where y denotes spectral count. It can be difficult to fit a linear model to data with this kind of distribution, since a small number of very large values tends to dominate the fit. However, $1/y$ has a uniform distribution, thus eliminating all skew and transforming the data into a more-manageable form. In addition, all the methods we develop for fitting the linear model are easily adapted to fit the inverse model.

3 Fitting the Models to Data.

The models described above each require the estimation of a parameter vector, β . For each model, this problem would reduce to multivariate linear regression if the amounts of protein, in_i , were included in the training data. Unfortunately, they are not. This makes the models non-linear in the unknowns. We have developed a number of methods for transforming these non-linear models into linear ones and for efficiently fitting them to data. In some cases, the problem still reduces to multivariate linear regression, but in most cases it reduces to generalized eigenvector problems. In all, we have developed three methods for fitting the linear model to data (denoted LIN1, LIN2 and LIN3), three methods for the inverse model (denoted INV1, INV2 and INV3), and two methods for the exponential model (denoted EXP1 and EXP2). To give a taste of what is involved, we discuss here the methods for the linear model and develop one of them in detail. The remaining methods are developed in [2].

3.1 Linear Models. Of our three methods for linear models, the first two divide learning into two phases. The first phase estimates a value for β , and the second phase uses β to help estimate values for the amounts of protein, in_i . The two methods differ in the optimization criteria used to fit the model to the data. In contrast, the third method has only one learning phase, in which all parameters are estimated simultaneously.

Recall that the linear model is given by equations of the form:

$$(3.4) \quad y_{ij} = in_i \cdot (\mathbf{x}_{ij} \bullet \beta)$$

where the parameter vector β and all the in_i are un-

known and must be learned. Of course, these equations are not exact, and provide at best an approximate description of the data. The goal is to see how closely they fit the data, and to estimate values for β and in_i in the process. From the discussion in Section 2.1, we know it is only possible to estimate *relative* values for these quantities. This effectively means we can determine the *direction* of β but not its *magnitude*. In fact, in the absence of calibration data, the magnitude of β is meaningless. For this reason, all the methods for the linear model impose constraints on the magnitude of β in order to obtain a unique solution. We now develop the first (and simplest) of these methods.

3.1.1 LIN1: Two-Phase Learning. From Equations 3.4, we see that protein i gives rise to the following equations, one equation for each peptide:

$$(3.5) \quad \begin{aligned} y_{i1} &= in_i \cdot (\mathbf{x}_{i1} \bullet \beta) \\ y_{i2} &= in_i \cdot (\mathbf{x}_{i2} \bullet \beta) \\ &\dots \\ y_{in_i} &= in_i \cdot (\mathbf{x}_{in_i} \bullet \beta) \end{aligned}$$

By dividing each equation by the previous one, we can eliminate the unknown value, in_i . That is, $y_{ij}/y_{i,j-1} = (\mathbf{x}_{ij} \bullet \beta)/(\mathbf{x}_{i,j-1} \bullet \beta)$, for j from 2 to n_i . Cross multiplying gives $y_{ij}(\mathbf{x}_{i,j-1} \bullet \beta) = y_{i,j-1}(\mathbf{x}_{ij} \bullet \beta)$, and rearranging terms gives the following equation:¹

$$(3.6) \quad \mathbf{z}_{ij} \bullet \beta = 0$$

where $\mathbf{z}_{ij} = y_{ij}\mathbf{x}_{i,j-1} - y_{i,j-1}\mathbf{x}_{ij}$, for j from 2 to n_i , and i from 1 to N . This equation is a restatement of Equation 3.4 with the unknown values in_i removed. Like Equation 3.4, it is an approximation, and our goal is to see how closely it fits the data.

A simple approach is to choose β so that the values of $\mathbf{z}_{ij} \bullet \beta$ are as close to 0 as possible. That is, we can try to minimize the sum of their squares, $\sum_{i,j} (\mathbf{z}_{ij} \bullet \beta)^2$. Of course, this sum can be trivially minimized to 0 by setting $\beta = 0$. But, as described above, the magnitude of β is meaningless, and only its direction is important. So, without loss of generality, we minimize the sum of squares subject to the constraint that the magnitude of β is 1. To do this, we use Lagrange multipliers. That is, we minimize the following function:

$$(3.7) \quad F(\beta, \lambda) = \sum_{i,j} (\mathbf{z}_{ij} \bullet \beta)^2 - \lambda(\|\beta\|^2 - 1)$$

¹Of course, we could generate many more equations of this form by cross multiplying all possible pairs of equations from 3.5, instead of just the successive ones. However, only $n_i - 1$ of the resulting equations would be linearly independent.

Taking partial derivatives with respect to β and setting the result to 0, the algebra leads to the following equations [2]:

$$\sum_{ij} \mathbf{z}_{ij} \mathbf{z}_{ij}^T \beta = \lambda \beta \quad \lambda = \sum_{ij} (\mathbf{z}_{ij} \bullet \beta)^2$$

In the left equation, all multiplications are matrix multiplications, with all vectors interpreted as column vectors. The left equation says that β is an eigenvector of the matrix $\sum_{ij} \mathbf{z}_{ij} \mathbf{z}_{ij}^T$, and the right equation says that its eigenvalue is just the sum of squares we want to minimize. We should therefore choose the eigenvector with the smallest eigenvalue. This provides an estimate of the parameter vector β , and completes the first phase of learning.

In the second phase, we use β to estimate the abundance of each protein, in_i . This can be done in a number of ways, such as using Equation 2.2.

4 Experimental Evaluation.

We conducted an extensive evaluation of the models and methods developed above using both real and simulated datasets [9]. This section presents a small sample of results based on the real data, which consists of tables similar to Table 1. We used three such datasets, derived, respectively, from tissue samples of the brain, heart and kidney of Mouse. The datasets were provided by the Emili Laboratory at the Banting and Best Department of Medical Research at the University of Toronto. Each dataset contains about 8,000 peptides and over 1,200 proteins. We divided each dataset randomly into two subsets, training data and testing data, in a 2:1 ratio.

To use the models and methods developed in this paper, each peptide must be represented as a vector, \mathbf{x} . We evaluated several ways of doing this, using vectors with 22, 42, 62 and 232 components, respectively. The 22-component vector captures the charge and amino-acid composition of a peptide. In addition, the 42- and 62-component vectors capture some sequential information, and the 232-component vector has additional features that are non-linear combinations of the others. Complete details are available in [2].

We used each of the four peptide representations with each of our eight fitting methods, LIN1, LIN2, LIN3, INV1, INV2, INV3, EXP1 and EXP2. We applied each combination to the training data to estimate a value for the parameter vector, β . We then used β to predict the spectral counts of all the peptides in the testing data, and then compared the predictions to the observed values by measuring the correlation between them. Table 2 shows a selection of experimental results for every combination of the eight fitting methods, and four peptide representations. The columns of each table

