2006-08-11

Sources of Variability in a Proteomic Experiment

Scott Daniel Crawford
Brigham Young University - Provo

SOURCES OF VARIABILITY IN A PROTEOMIC EXPERIMENT

by

Scott D. Crawford

A project submitted to the faculty of

Brigham Young University

in partial fulfillment of the requirements for the degree of

Master of Science

Department of Statistics

Brigham Young University

December 2006

BRIGHAM YOUNG UNIVERSITY

GRADUATE COMMITTEE APPROVAL

of a project submitted by

Scott D. Crawford

This project has been read by each member of the following graduate committee and by majority vote has been found to be satisfactory.

Date	David G. Whiting, Chair
Date	G. Bruce Schaalje
Date	Natalie J. Blades

BRIGHAM YOUNG UNIVERSITY

As chair of the candidate's graduate committee, I have read the project of Scott D. Crawford in its final form and have found that (1) its format, citations, and bibliographical style are consistent and acceptable and fulfill university and department style requirements; (2) its illustrative materials including figures, tables, and charts are in place; and (3) the final manuscript is satisfactory to the graduate committee and is ready for submission to the university library.

Date

David G. Whiting
Chair, Graduate Committee

Accepted for the Department

Scott D. Grimshaw
Graduate Coordinator

Accepted for the College

Thomas W. Sederberg
Associate Dean, College of Physical and Mathematical Sciences

ABSTRACT

SOURCES OF VARIABILITY IN A PROTEOMIC EXPERIMENT

Scott D. Crawford

Department of Statistics

Master of Science

The study of proteomics holds the hope for detecting serious diseases earlier than is currently possible by analyzing blood samples in a mass spectrometer. Unfortunately, the statistics involved in comparing a control group to a diseased group are not trivial, and these difficulties have led others to incorrect decisions in the past. This paper considers a nested design that was used to quantify and identify the sources of variation in the mass spectrometer at BYU, so that correct conclusions can be drawn from blood samples analyzed in proteomics. Algorithms were developed which detect, align, correct, and cluster the peaks in this experiment. The variation in the m/z values as well as the variation in the intensities was studied, and the nested nature of the design allowed us to estimate the sources of that variation. The variation due to the machine components, including the mass spectrometer itself, was much greater than the variation in the preprocessing steps. This conclusion inspires future studies to investigate which part of the machine steps is causing the most variation.

Acknowledgements

A special thanks to David Whiting for working with me, teaching me, and offering me support as my chair. To Bruce Schaalje and Natalie Blades, whose advice and suggestions not only helped this paper be more professional, but helped to get the LaTeX coding and figures correct. To Craig Thulin, the biochemistry professor, for the use of his mass spectrometer and his knowledge in the field of proteomics. To Karen Merrell, a graduate student of the biochemistry department, who gave of her time in answering countless questions and even donated her blood to the project. And thanks to the entire Statistics Department for their camaraderie and assistance. Above all, thanks to my Heavenly Father for countless blessings and miracles which made this possible

Funding from the Maternal and Fetal Medicine Unit Network of the NICHHD and the University of Utah Department of Obstetrics & Gynecology

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

Proteomics is the field of science dedicated to studying protein structure and function. Knowledge of proteins unlocks mysteries about the structures of life on earth. The results from a proteomic experiment have a significant amount of variation, and drawing conclusions from the data may be difficult. This project attempts to define and quantify sources of variation that are encountered in proteomic studies.

One avenue of research that biochemists have been investigating is the possibility of using indicator proteins to detect diseases or other conditions. To find these proteins, samples from a control group are compared to a group that has a specific disease or condition. Using chromatographic mass spectroscopy, biochemists test whether subjects in the disease group have discriminating proteins that can be used as a test. Inferences about differences between the two groups is complicated by the variation that is inherent in all proteomic studies.

This project uses chromatographic mass spectrometry to analyze protein in blood serum samples. The preparation, handling, and storage of the blood serum

add variation to the data, does variation in the instrumentation. Additional error is introduced by numerical analysis of the data. The data contains many local maxima which can be artifacts of random variation. Algorithms to detect peaks, correct for random noise, and align peaks across different samples were developed for this data set.

Identifying and quantifying each source of variation was done by performing a nested design on one homogeneous blood sample. To assess this, one blood sample was taken and separated into multiple vials. Each vial was prepared separately and split into two other vials. Half of the vials were analyzed with the chromatographic mass spectroscopy on one day in two separate groups. The other half were analyzed on a second day in two other groups. Several variance components were estimated using data from the multiple runs.

The variance components allow two important concerns to be dealt with. First, if the variation overwhelms the signal in the instrumentation then past results in this field of research may be suspect. Second, to attempt to reduce the error in this type of analysis in the future, it is helpful to identify the largest sources of variation.

Chapter 2

Literature Review

2.1 The Science of Proteomics

Proteomics is a mixture of molecular biology, biochemistry, and genetics used in analyzing the structure, function, and interactions of proteins in cells, tissues, or organisms (Anderson and Anderson 2002). The biology is based on DNA, which codes for the production of messenger RNA (*mRNA*) through a process called transcription. This mRNA codes for protein synthesis by ribosomes during a process called translation (Kolata 1974). Proteins are responsible for most of the active functions in the body, including the reactions of metabolism by enzymes, the effects of hormones in response to signals such as pregnancy, or the capture of diseases with antibodies. Hence, every field of science that relates directly to living organisms may benefit from the findings of proteomics.

The study of proteomics attempts to characterize many aspects of protein; identifying which proteins are present in an organism, quantifying protein abundance, characterizing protein structure, separating complex proteins into simpler pieces, modifying the function of a protein, or analyzing the action of a protein

on the atomic, molecular, and cellular levels.

2.2 Mass Spectrometry

A mass spectrometer measures the mass to charge ratio (m/z) of ions. Many types of gases, liquids, and solids are analyzed with a mass spectrometer, including bodily tissues or fluids. When a sample is introduced into a mass spectrometer, it is broken down into separate ions that are subjected to an electric or magnetic field. After the ions enter this field, they are deflected depending on their mass and their charge; the detector measures how far each ion is deflected. Based on this distance, m/z is calculated (Nelson and Cox 2000).

2.2.1 Ionizers

The first component of a mass spectrometer is an ionizer, which separates the individual ions to be detected. A common type of ionizer, the electron ionizer, uses an electric current to separate gas molecules into different ions, and is used on large molecules such as proteins. A chemical ionizer uses reagent ions for a more stable separation of the sample ions. The chemical ionizer, however, is only used for small molecules, not for proteins.

For liquid samples, three different types of mass spectrometers are commonly used. SELDI or *Surface Enhanced Laser Desorption/Ionization* is used for simple compounds. This technology treats a special metal plate with a chemical that adheres to the molecular species of interest. By flushing the excess sample

away, only preselected species are left. When a laser hits the chemical on the metal plate, it releases the proteins to fly into the mass spectrometer. MALDI or *Matrix Assisted Laser Desorption/Ionization* uses a special matrix of chemicals to trap the entire sample. When the matrix is hit with a laser it vaporizes, leaving the entire sample free to enter the mass spectrometer. The SELDI mass spectrometer is preferable for a scientist who desires to study a certain subset of a sample. MALDI performs better for complex studies where an entire sample is desired.

The third type of ionizer that is used for liquid or solid samples is the *Electrospray Ionizer* or ESI. The ESI traps the sample into tiny water droplets which are electrically separated and quickly evaporated. As the water droplets evaporate, the sample inside is compressed. Each droplet subsequently bursts, propelling the compressed sample into the mass spectrometer, in the same way that two magnets of equal polarization repel each other. The resulting sample that enters the mass spectrometer is more complete because it has not lost substance as it would in SELDI, nor has it been subjected to a laser as in SELDI or MALDI (Suizdak 1994).

2.2.2 Analyzers

Once the sample has been ionized it enters the analyzer. A *Time-of-Flight* (TOF) analyzer uses a magnetic field to accelerate the ions toward the detector. The mass and charge of the ion affect the speed of the ion toward the detec-

tor. This type of analyzer is often used with MALDI or SELDI. The entire mass spectrometer is referred to as MALDI-TOF or SELDI-TOF. TOF analyzers often have an option of using a quadrapole, which is a mass filter that separates the ions based on their m/z value. These analyzers, called QTOF, disable the quadrapole option when it is desired to allow the entire sample into the mass spectrometer (Gross 2004).

A *sector analyzer* uses an electric field to warp the flight of ions; the length of the path for the ion is a result of the mass to charge ratio (Bertsch 1999). Other types of analyzers exist, including the *quadruple mass analyzer* that stabilizes and destabilizes ions and the *orbitrap analyzer* that causes the ions to spin around an electrode. All analyzers use an electric or magnetic field to uniquely alter the journey of various ions to the detector (Nelson and Cox 2000).

2.2.3 Detectors

Detectors sense a charge or current as a result of being in the presence of an ion. When an ion hits a *detector plate*, the electrons jump from the plate onto the ion. This movement of electrons is registered as an electrical flow. If the ion passes through a *Fourier Transform Ion Cyclotron Resonance* (FTICR) it flies between two metal plates. As the ion passes between the plates, it creates an electrical circuit which causes an electrical flow. Although there are several types of detectors, each reacts to the ion by creating a small current or charge. When the ion is detected, the m/z value is calculated (Cole 1997).

2.3 Chromatography

Chromatography is a collection of methods used to physically separate and analyze complex mixtures. A sample flows through a small tube or capillary called a column. As the sample encounters physical or chemical obstacles in the column, certain compounds of the mixture travel more slowly than the other compounds in the mixture. The mixture breaks down into its constituent parts when components of the mixture exit the column. The type of column or obstacles used to separate the mixture depend on what is being analyzed and how the sample needs to be separated.

2.3.1 Gas Chromatography

Gas chromatography uses a high pressure gas to push the liquid sample through the column. To separate the components, the column is either coated with a special chemical or packed with an adsorbent material. *Capillary gas chromatography* uses a solvent that is coated on the column walls. These columns need to be very small and very long to allow the mixture an opportunity to combine with the solvent, and are often wrapped into a small coil. As the gas pushes the mixture through the coil, certain components stick to the coating. This separates the mixture according to the time of component separation from the solvent (Menster et al. 1995).

In *gas absorption chromatography*, an absorbent material such as diatomaceous earth is packed into the column. The absorbent material absorbs different

elements of the mixture. The resulting liquid evaporates as the gas is forced through the column. As the packing material evaporates the mixture separates based on the volatility of each component.

2.3.2 Liquid Chromatography

In liquid chromatography the liquid sample is pushed through the column using a high pressure liquid. *Reverse phase chromatography* uses a solid, such as silica or resin beads, to separate compounds by polarity and determine whether the compound is hydrophobic. *High Performance Liquid Chromatography* (HPLC) pushes the liquid at a higher velocity. This extra force requires solids that can withstand the pressure, and also means the column can be shorter and uncoiled. *Size exclusion chromatography* uses a porous gel to filter the molecules in the mixture. This method is quick and is used to separate molecules by size (Brown and Hartwick 1989).

2.3.3 Ion Exchange Chromatography

Ion Exchange Chromatography uses a resin coating on the walls of the column or on beads inside the column which will magnetically adhere to the ions. *Cation exchange chromatography* uses a negative charge, whereas *anion exchange chromatography* uses a positive charge. The molecules leave the column depending on the strength of their charge (Harris 1998).

2.3.4 Affinity Chromatography

Affinity chromatography uses a specialized column that has been treated with a matrix of chemicals or specially treated beads. As the sample enters the affinity column, separate parts of the sample stick to the chemical or beads depending on its structure. A common choice for the chemical is one that reacts to water in the compounds. This chemical causes the compounds that are hydrophobic to fall further into the column instead of sticking at the top. Another chemical is used to flush the sample through the column so that the sample will elute—separate chemically—through the column (Wulfsberg 2000).

2.4 Chromatographic Mass Spectrometry

An improved analysis of a sample is obtained by combining affinity chromatography with mass spectrometry. The first step is to use an affinity column to separate the sample and, as it elutes, feed it into a mass spectrometer. This means that only a subset of the sample is being analyzed in the mass spectrometer at any given time, but the entire sample is eventually analyzed (Kienle 1992).

The use of the affinity column is due to the fact that ions with similar m/z ratios are likely to elute at different times, allowing them to be analyzed separately. It is difficult to separate ions with similar m/z ratios if only a mass spectrometer is used. The combination of an affinity column with a mass spectrometer clarifies which molecular species are detected in the sample. The output from this combination is a three-dimensional model with m/z along one axis, elution time from

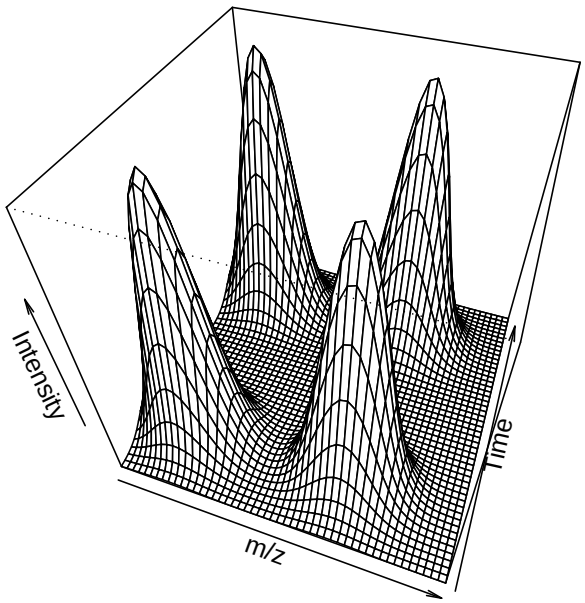

Figure 2.1: A sample of 3-D output from the chromatographic mass spectrometry

the affinity column along another axis, and the intensity of the detection along the third axis. Figure 2.1 shows how this output might appear in three dimensions.

2.5 The Proteomic Search for Biomarkers

One goal of proteomics is the search for *biomarkers*. Biomarkers are specific proteins that are present in a subject due to a disease or condition. If a biomarker is discovered for a certain disease, then a test for that disease may be developed (Master 2005). Proteomics has been used to search for biomarkers to test for ovarian cancer (Petricoin et al. 2002a), prostate cancer (Petricoin et al. 2002b), and breast cancer (Vlahou et al. 2001). Currently, a team of researchers at Brigham Young University and the University of Utah Medical School are researching the possibility of finding biomarkers to test for preeclampsia, a potentially fatal condition that can occur during pregnancy.

The search for preeclampsia biomarkers is done by analyzing blood serum samples from patients who are known to have preeclampsia and comparing them with control patients who are pregnant, but are not diagnosed with preeclampsia. The blood serum samples are analyzed using a high performance liquid chromatography column with hydrophobic beads and an electrospray ionizer-quadrapole time of flight mass spectrometer (ESI-TOF)(Figure 2.2). In many proteomic searches, the quadrapole option on the mass spectrometer is disabled.

Figure 2.2: The Qstar Pulsar i ESI-LC TOF Mass Spectrometer used at BYU

2.6 Statistical Concerns

Analyzing the data from an ESI-TOF is complicated by several statistical concerns. Forming correct conclusions is difficult due to complications in acquiring the data, correctly identifying peaks, and transforming the data.

2.6.1 Data Problems

The computer program that controls the mass spectrometer is a proprietary product of the company that makes the device, so there are severe restrictions on the types of output that are obtained from this instrument. A three-dimensional plot of the data can be viewed, or cuts can be made to create slices of the data along the time or m/z axes, but the raw data used to create the three-dimensional plot is not available. If a slice of the data is used, then the data will be aggregated (i.e., cuts in the time dimension will integrate across a time segment at each m/z).

If a cut crosses through the middle of a peak, then the total signal on each side of the cut is diminished. It will appear that there are two small peaks in each slice on either side of the cut, when one large peak should be stretching across both slices. When two peaks are aligned perpendicular to the cut, only one large peak is detected. Ideally the cuts will go between peaks, so that each peak appears appropriately in each slice, but when that is not the case error will be introduced into the data.

In an effort to reduce this error, the biochemistry department at Brigham Young University has decided to slice along the time axis, with each slice contain-

ing the integrated intensities during two-minute windows. Most peaks are resolved within two minutes of elution from the affinity column, a two-minute slice ensures that the peaks have had time to be fully detected. The three-dimensional plots of raw data were examined and it was concluded that two minute slicing tended to separate peaks well.

Analyzing peaks across different samples in three-dimensional space is a very complex problem, while in two-dimensions it is simpler. Instead of a three-dimensional data set with 22 minutes along the time axis, the data is consolidated into 11 two-minute windows.

2.6.2 Peak Alignment

When a compound is examined, the (time, m/z) coordinate of the peak is subject to variation. The variability in the time dimension is influenced by the elution time through the chromatography column, flight time of the ion in the mass spectrometer, or other uncontrollable sources of noise. To deal with this variation the three-dimensional output is examined and the slicing is aligned in the time dimension according to characteristic peaks. This slicing attempts to ensure that peaks in different runs are consistently aligned in the time dimension.

A current mathematical problem involves finding improved methods for aligning peaks in the m/z dimension (Gentleman et al. 2005). The same issues arise when comparing character strings or matching gene sequences. The greatest strides in this direction have come from voice recognition research. The methods

developed to match voice patterns specialize in aligning characteristics of non-linear functions or time series data.

Dynamic Time Warping (DTW) is a discrete programming algorithm that uses a Euclidean distance to adjust two samples into alignment. In DTW, one of the samples is used as a template and the independent axis of the other sample is "warped" to align it with the template. This warping shrinks or expands different areas of the independent axis, which lines up the maximum and minimum points of each function (Wang and Gasser 1997). *Correlative Optimized Warping* (COW) works similarly to DTW, but uses a penalized likelihood to maximize the correlation between the functions (Glasbey and Mardia 2001).

Other approaches include a *Hidden Markov Model* (HMM) which uses a stochastic process of states to model the most probable set of peaks across samples (Juang and Rabiner 1991). *Cluster analysis* is another method for determining which peaks are similar, and then grouping them together (Zhang et al. 2003). These methods are simpler to implement than DTW or COW, but can be slower when used with large and complex data sets.

2.6.3 Sources of Variability

Once the corresponding peaks have been aligned in each sample, the peak intensities across all samples are used to search for biomarkers. Biomarkers are found by comparing a treatment group with a control group and searching for differences in intensities between the peaks in each group, or for peaks that are

present in one group but missing in the other group.

This search is complicated by the many sources of variability inherent in these studies. Natural variability exists between subjects in the same group. Variation within a single subject can also be considerable, possibly due to the chemicals in their body from day-to-day living conditions, stress level, sleep, or diet. Variability is also influenced by factors such as different people preparing the solution, the lab environment, difference in the vials of blood, whether the blood is frozen, and the duration of the freezing. The instrumentation introduces variability depending on outside conditions, electrical currents, calibration of the machine, or other fluctuations inside the machine. The motion of the ions inside the affinity column and their flight through the mass spectrometer have variation. The detector may also introduce error depending on how precisely it can measure the presence of an ion.

The final output from the chemical analysis has variation from many sources. The variation among different subjects, or even within the same subject, is not easily controlled. It is important to quantify these types of variation, because any tests for a disease must account for this variation. Other sources of variation such as randomness in the flight of the ion or error in the detection can be decreased by improving the instrumentation or methodology. Understanding these sources of variability highlights which areas may benefit the most from continued research (Hartemink et al. 2001).

Certain sources of variation are difficult to examine individually. For ex-

ample, once a sample has been injected into the mass spectrometer, it will pass through the ionizer, analyzer, and detector automatically. There is no way to skip a step or to resend an ion through a stage a second time. The sources of variability that are associated with the ESI-TOF machine must therefore be grouped together as the machine variation.

The machine variation causes small jitters in the output from the ESI-TOF. The concern is whether these tiny jitters represent true peaks or merely fluctuations in the ESI-TOF itself. All mass spectrometers have a range of values where the signal is so small that the machine variation completely masks the peaks that should be detected. The location of this range depends on the mass spectrometer being used and the type of sample being analyzed. Figure 2.3 shows how this signal to noise ratio is a problem in the 1700 m/z range of the ESI-TOF for a specific sample. If the noisiest range of the data is used to search for biomarkers, there is a high probability that anything found in this range is an artifact of noise instead of signal.

2.6.4 Discrimination

In most cases, the data from a proteomic analysis is used to find differences between a control group and a disease group. Some of the most common types of analyses include *cluster analysis* (Bensamil et al. 2005), *discriminant analysis* (Purohit and Rocke 2003), *random forests* (Izmirlian 2004), and simple *multiple testing* (Gentleman et al. 2005). Each type of analysis has its own way of defining

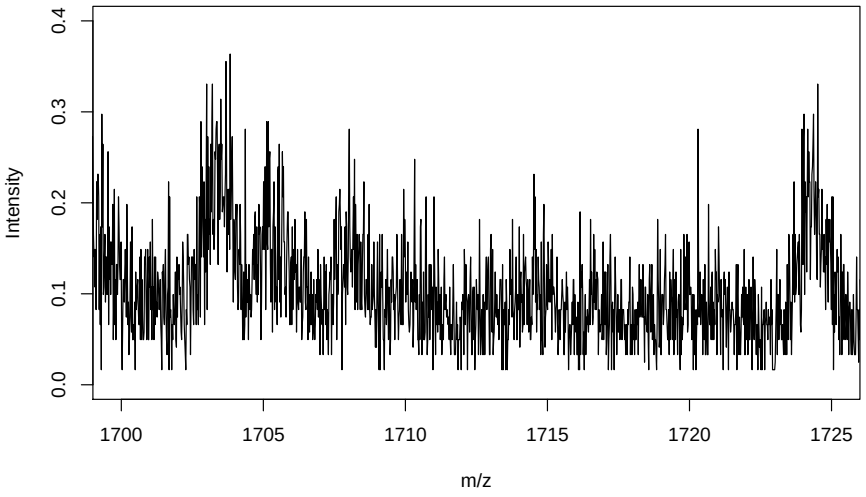

Figure 2.3: An example of a region of the mass spectrometer with a lot of noise

and discovering differences between the groups. Different methods have unique benefits and difficulties; however, in each case the statistics must be done properly to avoid mistaken conclusions (Constans 2005).

A recent article by Petricoin et al. (2002a) emphasizes the difficulty of statistical analysis in proteomics. Petricoin et al. claimed to be able to find biomarkers to detect ovarian cancer with 100% sensitivity and 95% specificity for the samples. Unfortunately, more careful statistical analysis did not support the claims (Sorace and Zhan 2003).

Most of the differences Petricoin et al. used to discriminate the control group from the cancer group were in the noisy m/z range for their mass spectrometer. This suggests the possibility that the differences Petricoin et al. found were random noise (Baggerly et al. 2004b). The design of these experiments is problematic. The samples from healthy subjects were run on one day, while the samples from cancerous subjects were run on the subsequent day. This led to concerns that the results could be due to different influences each day, such as the weather, calibration problems, or electrical impulses (Conrads et al. 2004).

Other concerns such as the population to which the conclusions can be extended have also arisen (Check 2004). These problems make further developments in the field of proteomics subject to intense statistical scrutiny. More work has been done to search for cancerous biomarkers (Conrads et al. 2004; Petricoin et al. 2002b), but each faced similar problems (Baggerly et al. 2004a; Diamandis 2004). These statistical concerns do not necessarily imply that the results are incorrect

(Liotta et al. 2004), but subsequent studies on these results have not been able to verify the claims of sensitivity and specificity for diagnosing cancer (Sullivan 2005).

2.6.5 Multiple Testing

One of the largest statistical concerns faced in proteomics is the large number of peaks that need to be examined. A range of ten m/z values can easily contain nine or ten peaks. Each time window has a range of approximately 2,000 m/z, and there are eleven different time windows to examine. Even if there are areas with few peaks, there could easily be over 15,000 peaks that would need to be compared between groups to find a biomarker. This means the probability of incorrectly finding a difference where there is no difference will be inflated (Baggerly et al. 2004b). This issue can be partially alleviated by dividing the data into two sets: one used to search for biomarkers and the other used to test those biomarkers. The massive number of peaks to be investigated is still problematic with this method.

Several ideas exist on how to deal with the possibility of finding a difference when there is none. Traditional methods such as a Bonferroni or Scheffe correction require a significance level that is usually considered too high (Nichols and Hayasaka 2003). More recently, an approach called *False Discovery Rate* (FDR) has been developed to control the expected proportion of false positives (Benjamini and Hochberg 1995). An extension of this method uses q-values. The

q-value accepts that there are false positives, but also calculates the probability that a certain discovery is not a false positive but a real difference; the q-value's usage is analogous to the p-value (Storey and Tibshirani 2003). Multiple testing may be a concern when searching for biomarker, but this project is not testing between treatment groups, and does not require multiple testing.

2.7 Experimental Design

A designed experiment can quantify many of the variance components needed to analyze proteomic output. Although this type of experiment is not a common procedure for chemists, we used one such design in this project to identify and estimate the sources of variation. This study is an important first step that should come before attempting further statistical analysis in proteomics.

Chapter 3

Methodology

3.1 Proteomic Data

3.1.1 Sample Preparation

When analyzing blood samples, certain steps must precede the use of an HPLC-MS. The blood sample must first have the red blood cells and clotting agents removed. To do this, the blood is left at room temperature for 30 minutes and then centrifuged for 10 minutes. The blood separates, creating a heavy pellet of red blood cells and clotting agents at the bottom of the tube. Plasma is collected off the top of the sample.

The next step is to remove large proteins such as serum albumin and immunoglobulins. These proteins are so large and abundant in blood samples that they mask the detection of the other proteins in the blood. One method for doing this, called *precipitation*, involves the use of an organic solvent, acetonitrile, which causes large proteins to unfold and fall out of solution. Due to the fact that these proteins are not dissolved in the solution, they can be removed easily through a centrifuge and the smaller, less-abundant proteins can be seen by the

mass spectrometer more readily (Merrell et al. 2004).

3.1.2 Chemical Analysis

The analytical steps begin with an autosampler. The autosampler has a bin that can hold five vials and an automatic arm with a needle at the end. The arm is designed to center the needle over a particular vial. The needle lowers into the vial to extract five microliters of the liquid sample. The sample feeds through tubes directly into the chromatography column.

Our data came from a machine that uses HPLC with hydrophobic beads that separate the proteins according to charge. The affinity column is connected to the mass spectrometer, where ions are detected and the resulting electrical flow is registered by the computer. The data is then sliced into eleven two-minute windows, where each window contains an average intensity for those two minutes at each m/z value. Each window is aligned so that the elution times begin at the start of the windows.

3.1.3 Preparing the Data

Figure 3.1 shows the intensity versus m/z for the fifth time window (time 10-12 minutes). Figure 3.2 shows several samples of data in the m/z range of 605.5 to 607, which has a range sufficiently small to see individual peaks. Nine separate runs are shown, each in a different color, to illustrate how the output can vary from sample to sample in the same group. Notice the differences in the peak

center and intensity for the different runs.

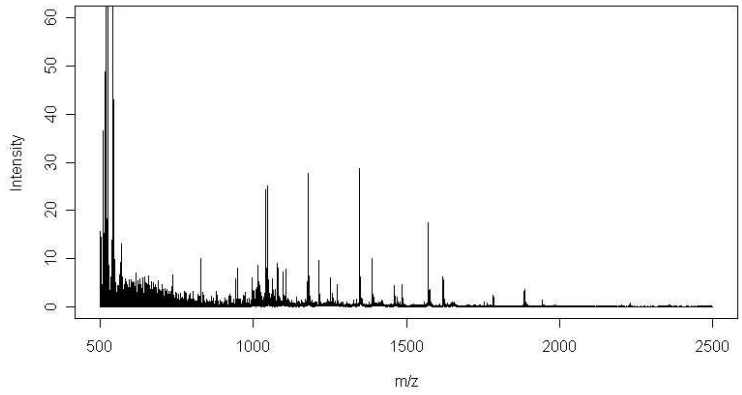

Figure 3.1: The raw data across the entire m/z axis for a single sample

There are transformations to the data set that improve the estimates of the peaks by reducing the variation due to machine fluctuations and alignment problems (Gentleman et al. 2005). These transformations include normalization, log transformation, baseline correction, adaptive threshold, and peak alignment.

The intensities have a distribution that is right skewed, as seen in Figure 3.3. This skewness tends to overemphasize the difference in intensities between a low intensity and a high intensity. Let y be the intensity data, then a transformation of $\log(1+y)$ cause these intensities to become more centrally and symmetrically distributed. Figure 3.4 shows the intensity profile from Figure 3.1 after the $\log(1+y)$ transformation is used. Note that $\log(1+y)$ is used instead

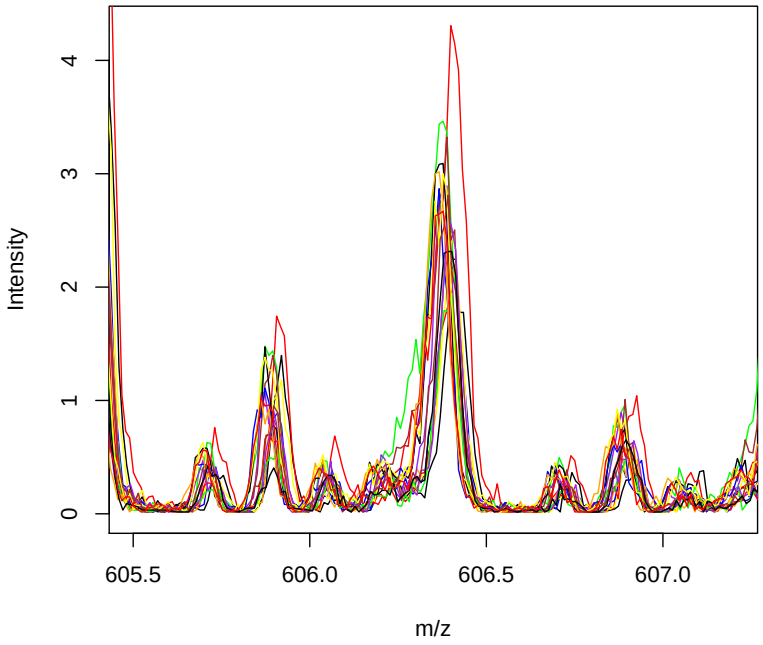

Figure 3.2: The raw data in a small m/z range for eight samples analyzed at the same time

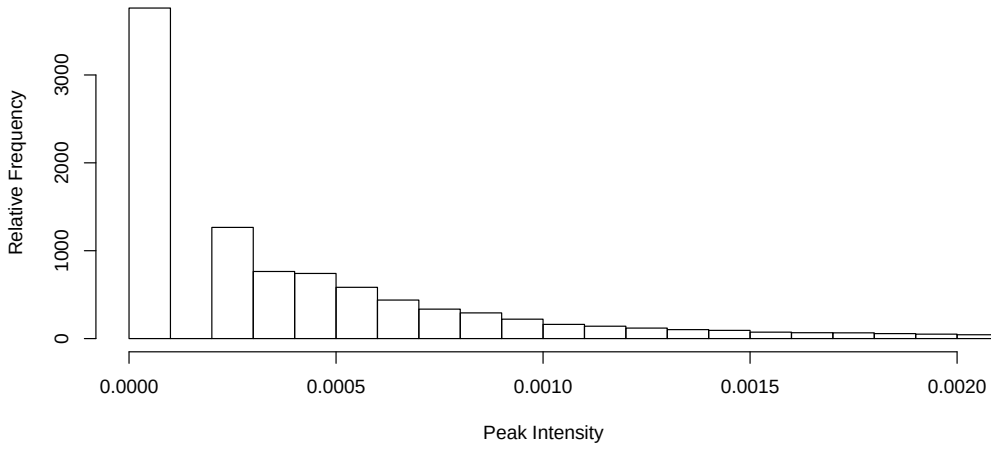

Figure 3.3: Histogram of a sample of intensities across the fifth time window

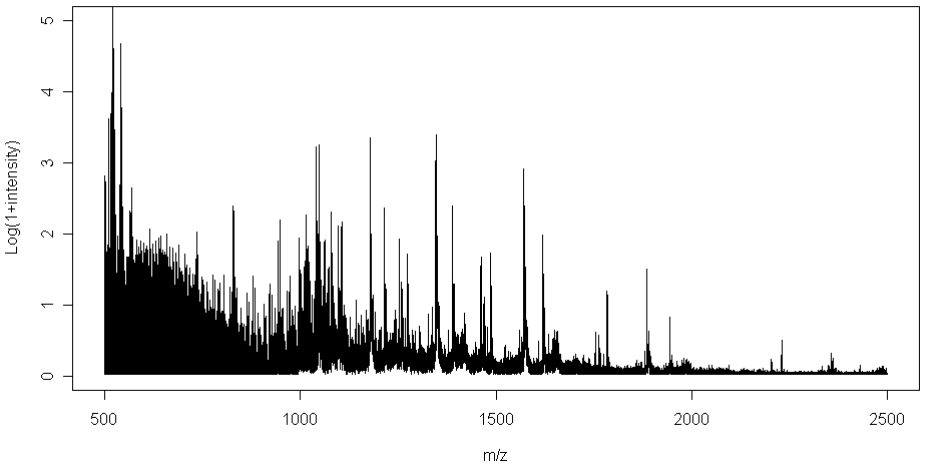

Figure 3.4: The log of one plus the raw output (Time Slice 5)

of $\log(y)$ to keep the intensities positive with the bottom baseline at zero, which makes the peaks easier for the chemists to both visualize and interpret. Many statistical methods are based on the assumption that the data are normally distributed. The transformation results in a more normal distribution of intensities. This implies that using well-known statistical methods is more valid.

If the autosampler injects a larger amount of the sample into the mass spectrometer, the intensities are higher across the entire spectrum. This should not be interpreted as more protein in that particular sample. The samples need to be normalized to a common total intensity. When normalizing the area, it is common to normalize the total signal to one. This implies that for every sample, the total area under the graph of the intensity versus the m/z value is one (Bolstad et al. 2002). Figure 3.5 below shows pre-normalized and normalized data for nine samples. Note how this makes the peak located slightly before m/z = 606.5 more uniform, which tends to lower the variance of the intensities.

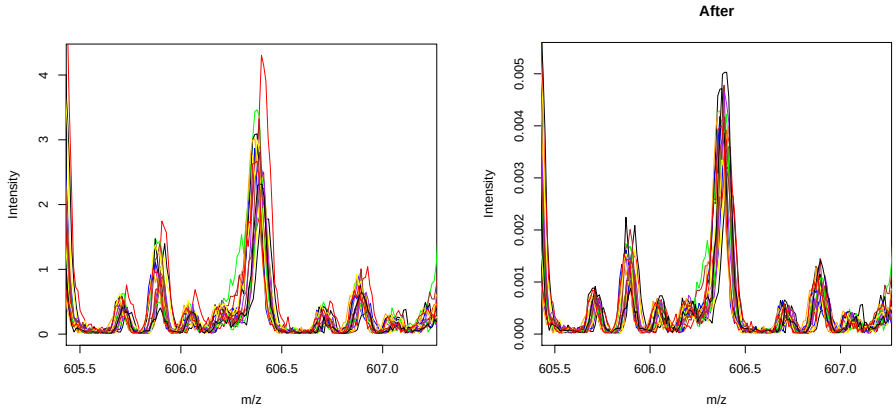

Figure 3.5: Before and after normalization

In many places the baseline of the intensities at a certain range of m/z values appears to rise, and it appears that the mass spectrometer is detecting more signal than it really is, as in Figure 3.6. This can happen when the mass spectrometer detects several peaks very quickly, often as a result of various ionic states of a particular protein's atoms. When the intensity of this baseline is estimated it should be subtracted from the raw data. The new baseline is zero for all the samples at every value of m/z. This baseline correction improves peak detection and increases the accuracy of the peak heights (Sauve and Speed 2004).

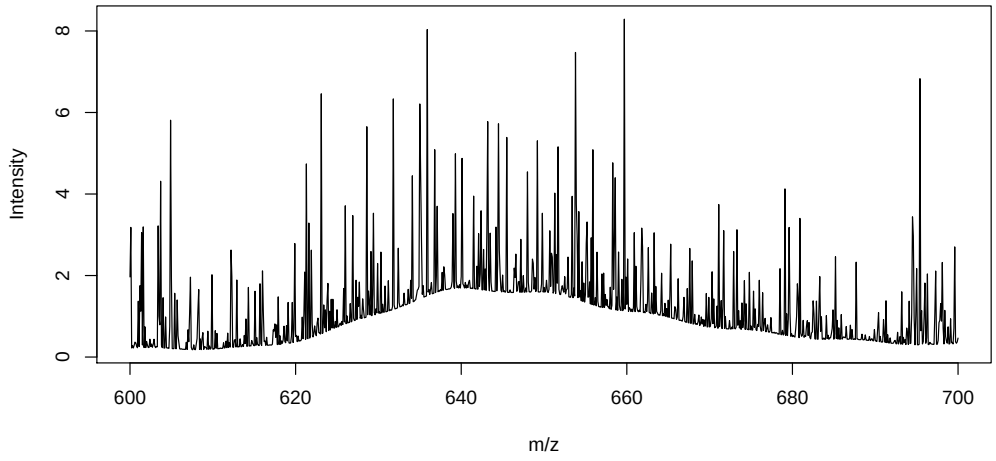

Figure 3.6: An example of baseline drift

At times, even if there are no proteins to be detected, the mass spectrometer still registers signal due to either fluctuations in the machine or random molecules hitting the detector. If there is true signal that happens to be very small in intensity, then it may not be distinguishable from the random noise inherent in mass spectrometer output. One solution to this problem is to set a *cut-off value*.

This means that any signal below a given threshold is considered unusable because it is in the noise region. This solution is easy to implement, although the choice of the cut-off value is somewhat problematic. The trouble with this method is that it assumes the signal to noise ratio is constant over the m/z values, which is not supported by investigation of the data.

A better solution than the cut-off value is an *adaptive threshold*. This approach relaxes the assumption that the signal to noise ratio is constant by using an adaptive statistic, such as a moving average over m/z. In this method, moving windows of the data are selected, and thresholds within each window are estimated. Intensities that are above the threshold are considered to consist of signal and noise. Thus, an estimate of the noise is subtracted from the intensity. Cases where the signal is not above the threshold are considered to be all noise and are set to zero (Gentleman et al. 2005). Figure 3.7 shows how the normalized data changes after applying a cut-off value.

The final goal of the data preparation is to identify the peaks that are representative of the molecular species in the original sample. It is difficult to define what constitutes a peak. Two main problems complicate the issue: overlapping peaks and jitter in the intensity level. Figure 3.8 shows how difficult it is to decide if a maximum is a peak or not. The figure shows three vertical lines in the general area where peaks may be.

In order for a maximum to be identified as a peak, three separate requirements must be fulfilled:

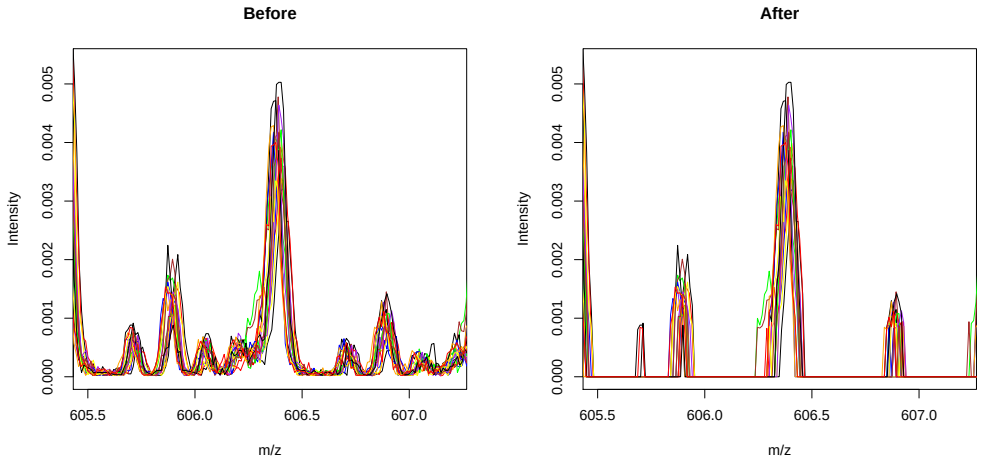

Figure 3.7: Before and after a cut-off algorithm was applied to the normalized data

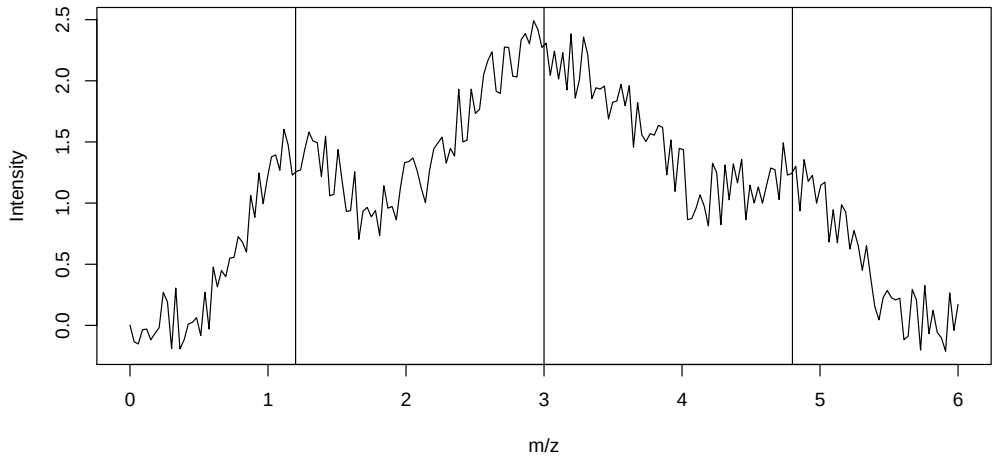

Figure 3.8: An example of noisy peak detection. Vertical lines represent approximate locations of peaks.

(1) The peak must be tall enough to be above the noise region.

(2) The peak must be distinct from other tall peaks. In order for the peak to be a satisfactory distance from tall peaks, the intensity must drop to a certain level before climbing again.

(3) The peak must be wide. The machine may register a spike, or blip, that is a result of some anomaly within the machine itself. These spikes are too sharp to be biological and should not be considered true peaks. Figure 3.9 shows a simulated example of such a spike, occurring at the m/z value of 601.5.

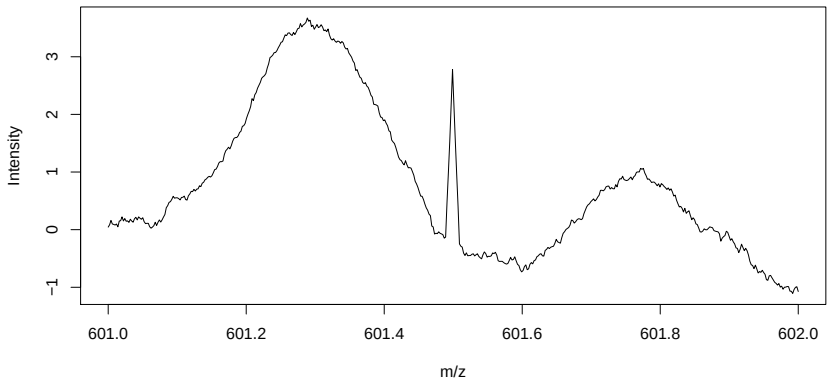

Figure 3.9: An example of a machine error spike

Although there are many algorithms, such as DTW or HMM, that can be used to identify peaks, these algorithms are computationally intense when dealing with complex data sets in proteomics. The data is larger and noisier than the data sets DTW or HMM were designed to handle, so we have developed an alternative

ad-hoc algorithm that requires less time and computer power (Table 3.1). The algorithm is for data that have been log transformed and normalized. The algorithm requires three parameters to be input by the user, a small window width, a large window width, and a tolerance level. The peaks that have been identified each have unique (time, m/z) coordinates and intensity.

The peaks must be combined into clusters that can be identified across samples. This is not trivial, since peaks will have different m/z values in each sample, and it can be difficult to distinguish the cluster to which a detected peak belongs. Figure 3.10 demonstrates the difficulty in grouping peaks with 20 samples. Some groups of peaks are easily distinguished and clearly separated from the other groups. Some peaks do not appear on all 20 of the samples, and therefore should not define a group.

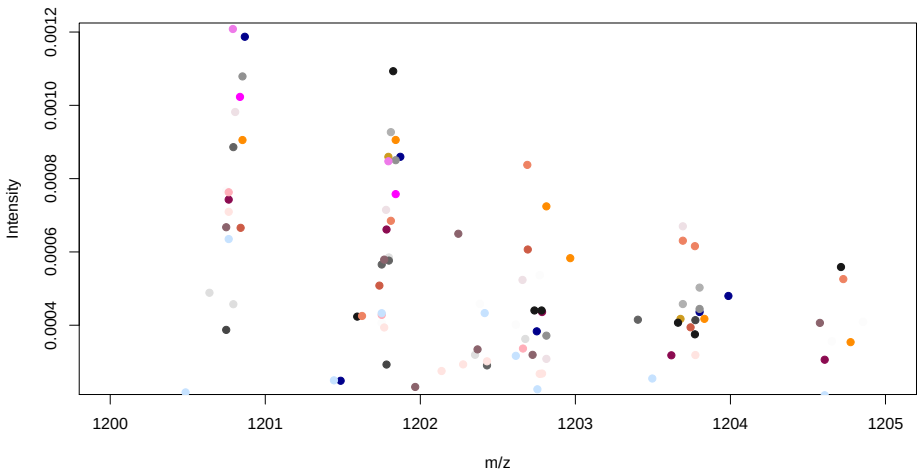

Figure 3.10: Intensity and m/z values of identified peaks in 20 blood samples. Dots of the same color are from the same sample.

Table 3.1: Algorithm for peak detection

Input small window width Input large window width Input tolerance level For each value of m/z perform the following: Calculate a large window as m/z±large window width Calculate average intensity level within the large window Estimate the standard deviation within the large window Calculate the threshold as the average plus the tolerance level times the standard deviation Calculate a small window as m/z±small window width Calculate the maximum within the small window If the maximum is above threshold, label m/z as signal + noise If the m/z is the maximum within the small window, then label this point as a peak. If the maximum is not above threshold, label m/z as pure noise
For all m/z that are labeled as pure noise: Estimate the noise level as average intensity in the large window
For all m/z that are labeled as signal + noise: Estimate the noise level using a straight line between the noise regions on either side of the signal + noise region. Subtract the estimate of the noise level from the intensity
Any signal below zero, set the signal to zero

The peaks could be grouped together using traditional cluster analysis (Bensamil et al. 2005), but there are two assumptions that make the implementation of this difficult. Cluster analysis requires an initial estimate of the number of clusters and a good beginning estimate of the cluster locations themselves. In proteomics, the number of peaks in a sample is unknown, and estimating the locations of the peaks is difficult.

Fortunately, when two assumptions are made an adapted clustering algorithm overcomes these problems. The first assumption is that peaks will have distinct m/z values. Two molecular species in the blood could have identical m/z values and elute from the affinity column at the same time, but in such a case we assume that the two species combine into one peak with a unique m/z value. The second assumption is that an estimate of the variance in peak center is obtained by examining a representative sample of peaks that are grouped together by hand. These two assumptions allow an algorithm to be developed that groups the peaks into clusters (Table 3.2). The clustering algorithm requires one parameter, the *zcritical* distance.

This algorithm solves difficulties of peak detection as well. The random variation creates spikes that are not true peaks. The clustering algorithm is not able to find peaks in the same area in the other samples, so spikes are rejected as noise. At times, when a tall peak fluctuates creating two equal maximums at nearly the same m/z. The clustering algorithm chooses the maximum that is closest to the other peaks, and ignores the other maximum. This solves the

Table 3.2: Algorithm for clustering peaks across samples

Input zcritical value
Initialize the cluster with first peak in first sample Set cluster center at the m/z value of the first peak
Iterate sequentially through the other 19 samples: Estimate the standard error of m/z at cluster center In the sample find the peaks where the m/z is within zcritical*standard error of the cluster center Include the peak that is closest to the cluster center as part of the cluster In case no peaks are within zcritical*standard error of the cluster center skip that particular sample Estimate the new cluster center as the average m/z value of all the peaks included in the cluster
Repeat iteration for all 20 samples until there is no change in the peaks included in the cluster or until the cluster repeats a previous clustering set Label which cluster the peaks belong to, and mark them so they will not be used as part of another cluster later.
Repeat algorithm for the next peak in the first sample Continue the algorithm for all peaks in the first sample

problem of assuming that there are two peaks, when in reality there is only one.

3.2 A Proposed Experiment

In an effort to understand and quantify the sources of variability, a crossed-nested experimental design was proposed. A nested design uses a hierarchical structure to allow the different components of the variance to be estimable.

In the proposed design (Figure 3.11) a quantity of blood is drawn from one subject at one time. The blood is stripped of the red blood cells and clotting agents, leaving the blood serum. The serum is then divided into five separate vials. Each vial goes through the preprocessing steps separately, including the use of acitonitrile to precipitate out albumin. Each vials is then split into two, resulting in a total of ten vials.

One of the vials from each pair of the five original vials is then placed in the freezer for the following day, while the other is analyzed the first day. The vials are randomly placed in the autosampler, and after each vial is run through the autosampler, it is rerun a second time in a new randomized order. The following day the five vials in the freezer are thawed and are placed into the autosampler. Each vial is run in random order twice, following the procedure of day one.

Differences among runs allows the variance due to the HPLC-MS to be estimated. Variation within runs after accounting for vials effects gives an estimate of the variance due to the autosampler. Comparing the vials that were frozen to the vials that were never frozen allows an estimate of variation due to freezing and

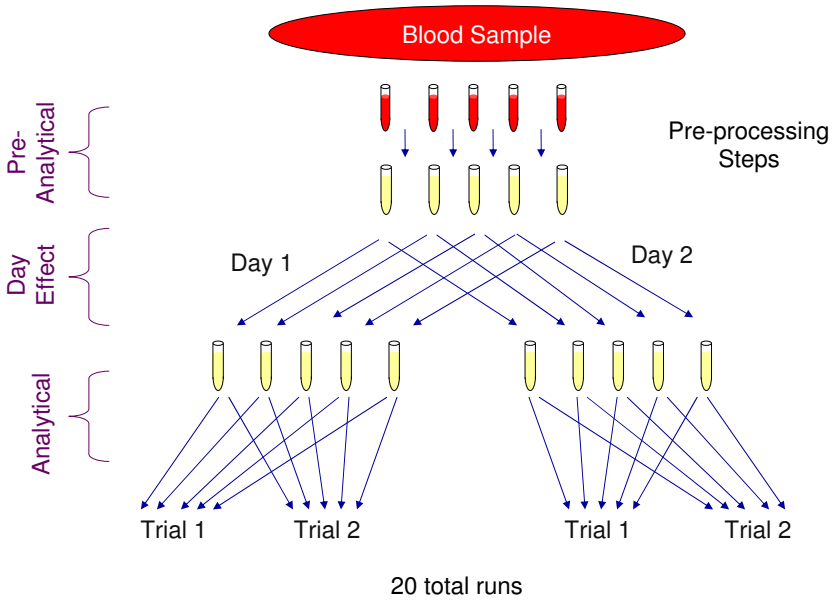

Figure 3.11: The basic outline of the design

other day effects. Comparing the original five vials, an estimate of the variation due to the preprocessing steps is obtained.

An appropriate statistical model is

$$y_{klm} = \mu + f_k + g_l + (fg)_{kl} + h_{m(k)} + \varepsilon_{lm(k)}, \tag{3.1}$$

where y_{klm} is the intensity level, μ is the mean, f_k is the random effect of the kth day, $k = 1, 2$, g_l is the random effect of the lth vial, $l = 1, \ldots, 5$, $(fg)_{kl}$ is the random effect of the day by vial interaction, $h_{m(k)}$ is the random effect of the mth trial run nested within the kth day, $m = 1, 2$, ε_{klm} is the random effect of the vial by trial interaction as well as other measurement errors. The effects f_k, g_l, $(fg)_{kl}$, $h_{m(k)}$, and ε_{klm} are all assumed to be normal random variables with mean zero and variances σ_f^2, σ_g^2, σ_{fg}^2, σ_h^2, and σ_ε^2, respectively.

3.3 The Actual Experiment

The actual experiment was somewhat different from the proposed experiment(Figure 3.12). A quantity of blood was drawn from one subject, and ten vials of the blood serum were taken from that quantity. Each vial went through the preprocessing step of precipitation. Five vials were then placed in the freezer to be run later. Five vials were placed in the autosampler immediately.

The vials were named A B C D and E. In the first run, the autosampler was randomly set to run vials in the order A–D–C–⊘–B–E where ⊘ represents a *blank* run, meaning a sample without any molecular species. In the second run, a different order was used, E–⊘–C–A–B–⊘–D.

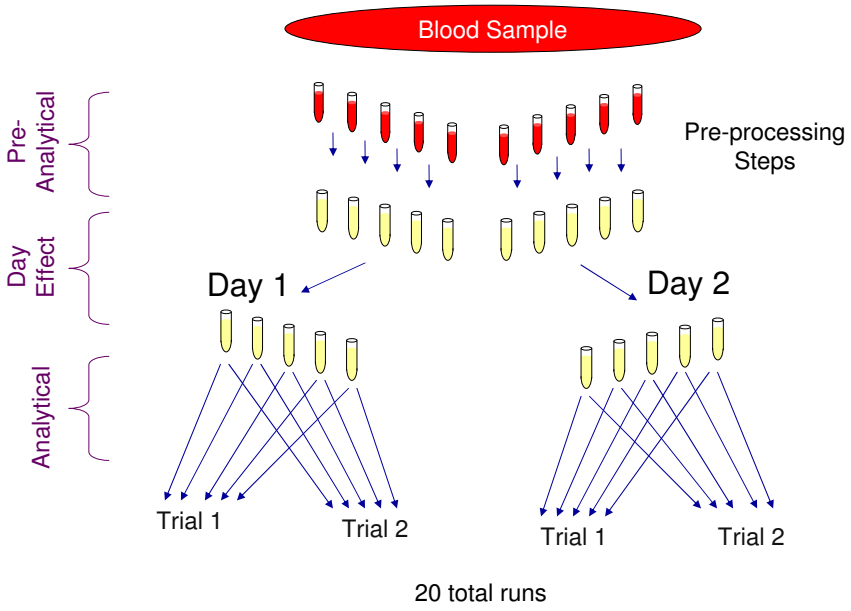

Figure 3.12: The outline of the actual experiment performed

A blank run is sometimes used to make sure the equipment is running properly. A blank run washes out any leftover particles from a previous run. The chance that the equipment is not working properly, or that a significant amount of molecules has been retained in the column, is expected to be very small. Using a blank run is time consuming because each run requires an hour of analysis. For these reasons, a blank run is not commonly used and was not planned in the original design. Using a blank run to clean the affinity column could be advantageous, and for that reason there were three blank runs used in the actual design.

The second day, the other five vials were taken from the freezer and thawed. The autosampler was set to the same ordering as the first five vials and each vial was analyzed twice in the same order as before (instead of re-randomizing). The fact that this second set of vials was not re-randomized is unfortunate.

In the actual experiment, the day effect is confounded with the order due to the lack of re-randomization. The effect of day is estimated with less precision, since it is not crossed with vial. Despite such difficulties, the design still allows estimation of most of the components of variation.

This design includes several nested factors and crossed factors. A nested factor is a factor which is indexed by the factor above it. For example, in this experiment the vials were labeled A, B, C, D, and E for day one, and A, B, C, D, and E for day two. This means vial A in day one is not the same as vial A in day two, in other words, the vial is nested within day. Two different ESI-TOF

runs were done each day, so trials are nested within days as well. Each trial used all five of the vials for that day, so trials are crossed with vials nested within days (Kuehl 2000).

A reasonable statistical model for intensity in a specified peak is

$$y_{klm} = \mu + \beta_0 z_{klm} + f_k + g_{l(k)} + h_{m(k)} + \varepsilon_{klm} \tag{3.2}$$

where y_{klm} is the intensity level, μ is a constant, β_0 is the coefficient for the indicator variable z_{klm} where

$$z_{klm} = \begin{cases} 1 & \text{if a blank run was used previously} \\ 0 & \text{otherwise,} \end{cases}$$

f_k is the random effect of the kth day, $k = 1, 2$, $g_{l(k)}$ is the random effect of the lth vial nested within the kth day, $l = 1, \ldots, 5$, $h_{m(k)}$ is the random effect of the mth trial run nested within the kth day, $m = 1, 2$, ε_{klm} is the random effect of the vial by trial interaction plus measurement errors. The effects f_k, $g_{l(k)}$, $h_{m(k)}$, and ε_{klm} are all assumed to be independent normal random variables with mean zero and variances σ_f^2, σ_g^2, σ_h^2, and σ_ε^2, respectively. The degrees of freedom for each factor are given in Table 3.3.

3.3.1 Likelihood approach

Estimating the variance components means to find values that maximize the likelihood. The model (3.2) assumes that the random effects are normally distributed with a mean of 0 and a variance of σ^2. The sum of normally distributed

Table 3.3: Degrees of freedom for each factor

Name	Source	df
Constant	μ	1
Preceding Blank	β_0	1
Day	f	1
Vial	g	8
Trial	h	2
Error	(gh)	7
Total	y_{klmn}	20

variables is normally distributed, so y_{klm} is also normally distributed. The likelihood is the product of the density functions for each y_{klm}. The likelihood is

$$\mathcal{L}(\boldsymbol{\beta}, \boldsymbol{\theta}|\mathbf{y}) = \frac{\mathbf{1}}{(\mathbf{2}\pi)^{\frac{\mathbf{n}}{\mathbf{2}}}|\mathbf{ZGZ}' + \mathbf{R}|^{\frac{\mathbf{n}}{\mathbf{2}}}} \mathbf{e}^{-\frac{\mathbf{1}}{\mathbf{2}}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})'(\mathbf{ZGZ}'+\mathbf{R})^{-\mathbf{1}}(\mathbf{y}-\mathbf{X}\boldsymbol{\beta})} \tag{3.3}$$

Another representation of the model is in matrix form

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{u} + \boldsymbol{e} \tag{3.4}$$

where $\boldsymbol{y}$ is the vector of intensities, $\boldsymbol{X}$ is the known matrix of predictor variables, and $\boldsymbol{\beta}$ is a vector of unknown coefficients. The random effects are represented by $\boldsymbol{Z}\boldsymbol{u}$ where $\boldsymbol{Z}$ is the matrix relating the respective random effects to each response $\boldsymbol{y}$, and $\boldsymbol{u}$ is the vector of each random effect with variance $\boldsymbol{G}$. The vector $\boldsymbol{e}$ is the random error with variance $\boldsymbol{R}$. The elements of $\boldsymbol{G}$ are the variance components.

The variance of $\boldsymbol{y}$ is $\boldsymbol{ZGZ}' + \boldsymbol{R}$. The solution for the model is found by minimizing $(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})'(\boldsymbol{ZGZ}'+\boldsymbol{R})^{-1}(\boldsymbol{y}-\boldsymbol{X}\boldsymbol{\beta})$. A restricted likelihood is adjusted to yield more unbiased solutions. The equation for the restricted likelihood is given by

$$\text{REML}_{(\boldsymbol{G},\boldsymbol{R})} = -\frac{1}{2}\log|\boldsymbol{V}| - \frac{1}{2}\log|\boldsymbol{X}'\boldsymbol{V}^{-1}\boldsymbol{X}| - \frac{1}{2}\boldsymbol{R}'\boldsymbol{V}^{-1}\boldsymbol{R} - \frac{n-p}{2}\log(2\pi), \tag{3.5}$$

and the Newton-Raphson method can be used to find fixed effects and variance components that maximize the restricted likelihood. Programs such as SAS PROC MIXED are able to solve iteratively for these components (Littel et al. 1996).

Chapter 4

Results

4.1 Peak Detection

Many different combinations of parameters were used to optimize the peak detection algorithm. The viability of the parameters was measured by examining several segments of the data and comparing the peaks found by the algorithm to those that biochemists at Brigham Young University believed should be found. These biochemists chose the peaks that should be detected by examining plots of each segment of data. The parameters chosen are in Table 4.1. The peak detection algorithm was able to find an average of 5,209 peaks in each time window of each sample, for a total of 1,145,996 peaks.

Table 4.1: Parameters used in the peak picking algorithm

Parameter	Value
Small Window Width	0.12
Large Window Width	1.8
Tolerance Level	2

4.2 Peak Alignment

4.2.1 Preliminary Work

The clustering algorithm finds peaks that are close to some initial estimate of the cluster center. To calculate the distance between a peak and the cluster center the difference between them is standardized in the m/z axis. That is, the difference between the peak m/z value and the m/z value of the cluster center is divided by an estimate of the standard deviation of the m/z value for that peak. This requires some preliminary work to estimate the variance in the m/z value for a peak.

To get an initial estimate of this standard deviation I manually selected 43 peaks after a visual inspection of the fifth time window. To find these peaks I examined a graph of the entire window. I chose peaks throughout the m/z spectrum, ranging from 500–2500. These peaks were clearly above the noise region, and distinguishable in all 20 samples.

Using this data to analyze variance components may bias the results, since the peaks chosen were those that were easily recognized. The easiest type of peak to find were those with a high intensity and low variance. This means our estimates of the variability may be biased, but it is a preliminary step to determining the best parameters for the peak alignment.

4.2.1.1 Estimating Variance of Peak Center

The theory of proteomics states that as the m/z values increase, the variation in the peak center increases exponentially (Gentleman et al. 2005). Given that the variation increases exponentially, the regression model is:

$$\log(y) = \beta_0 + \beta_1 x + e, \tag{4.1}$$

where y is the standard error of the m/z value for each group of peaks, x is the average m/z value for the peaks, and e is the error distributed as $N(0, \sigma^2)$. The error term may not be normally distributed, and it appears that the variance is not constant, but this assumption will allow a rough estimate to be used.

Figure 4.1 shows the estimates of the standard error of m/z as a function of the m/z value. The fitted function has been plotted with a solid line using

$$\log(y) = -5.26 + 0.00179x. \tag{4.2}$$

This equation can be used to estimate the variance of peak centers and standardize the distance between any given peak and a given cluster center. Both coefficients are significant with a p-value of nearly zero. Figure 4.2 shows the histogram of m/z values using in the regression.

This regression on the standard deviation of the m/z is an important contribution as it allows for the formation of an adaptive window for peak alignment. Previous studies have suggested that a constant window would be appropriate if a log transformation on the m/z axis were performed (Gentleman et al. 2005). This study shows that a log transformation is not sufficient.

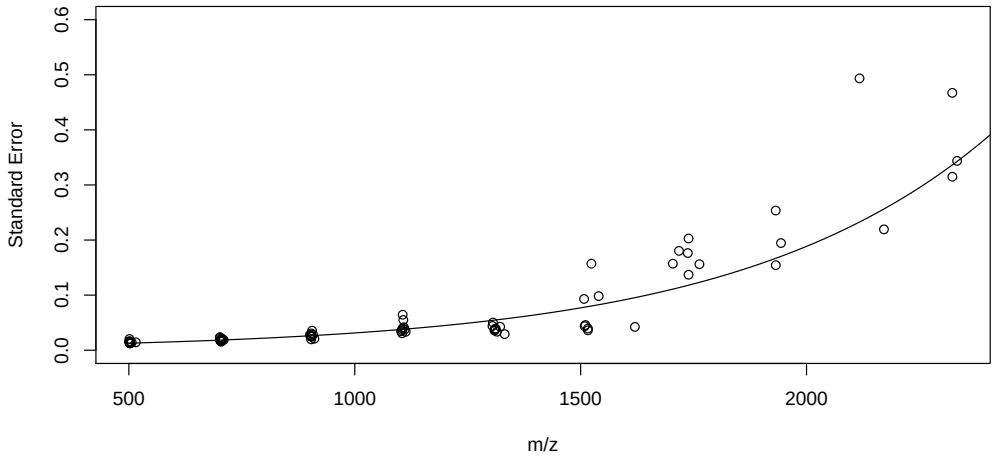

Figure 4.1: The standard error of peak center by m/z

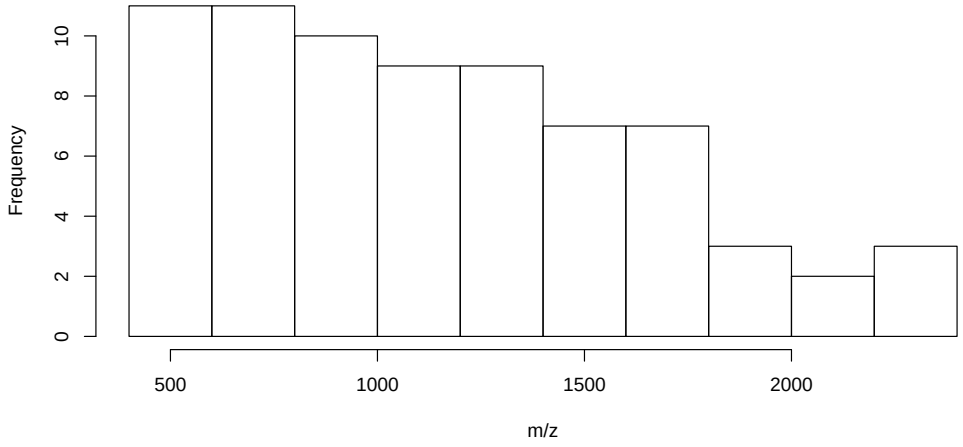

Figure 4.2: Histogram of m/z values used for regression

4.2.1.2 Variance of Peak Intensity

The data suggest that the standard errors of the intensity would have a linear relationship to the intensity level (Figure 4.3) The fitted equation is

$$y = 2.82 \times 10^{-5} + 0.299x, \tag{4.3}$$

where y is the standard error of the intensity of the peaks in each group and x is the average intensity level for each group of peaks.

4.2.2 Clustering

The window size chosen for the clustering algorithm was based on the 99.9% confidence interval. Because the confidence level is high, it should ensure that only the most extreme outliers are not used in the clustering. With 99.9% confidence, the window width is 3.09 times the calculated standard deviation from the mean in the m/z axis. The algorithm (Table 3.2)identified an average of 1,605 clusters in each time window and a total of 17,658 clusters across all eleven time windows.

Figure 4.4 shows a histogram of the number of samples in each cluster. A cluster may have only a few samples if the peak picking algorithm detects a peak that is actually noise. The peak will not be found in the other samples, and any cluster that includes that peak will be incomplete. Of the 17,658 clusters there were 7,665 clusters involving all twenty samples. These clusters were kept and the others were discarded.

When analyzing samples across different subjects, it would not be sensible to restrict the clustering algorithm to peaks that are present in every sample. If

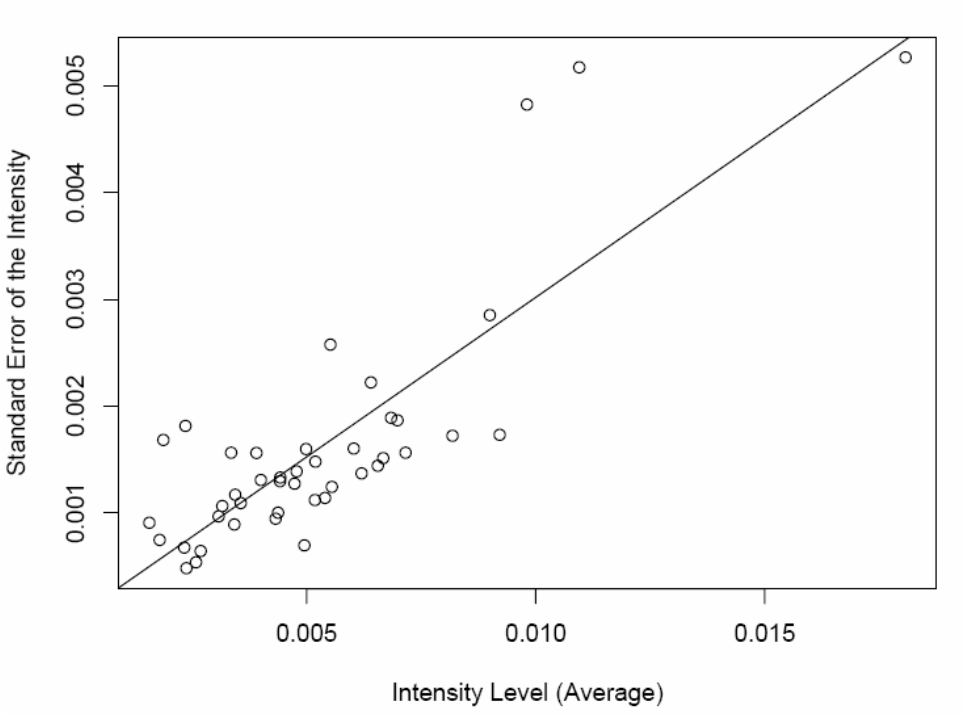

Figure 4.3: The standard error of peak height by intensity

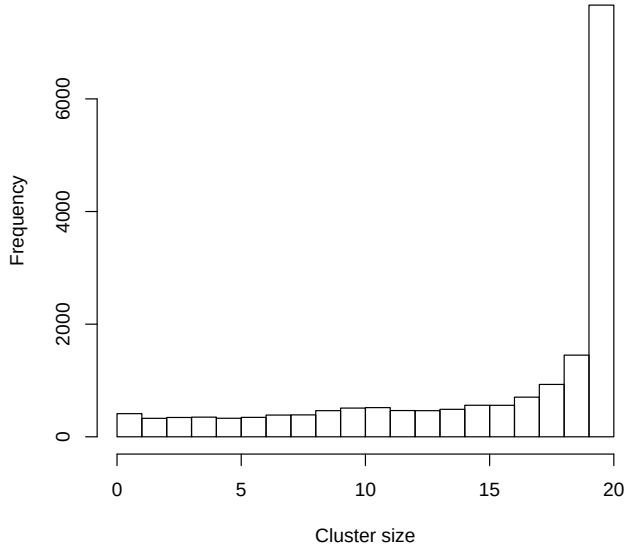

Figure 4.4: Histogram of peak alignment cluster sizes - clusters less than 20 in size were dropped

the goal is to find differences between subjects then peaks that are present in some samples but lacking in others would actually be of greatest interest. The missing peaks could occur due to differing conditions between subject groups, and testing for that peak could also detect the condition. In a proteomic analysis where the goal is to find biomarkers, it would be better to accept peaks that are found in a certain high percentage of the samples in a treatment group, say 95% of the samples in the cancer group and 95% of the samples in the control group.

4.3 Data Analysis

4.3.1 Variability in Peak Intensity

When comparing a control group to a treatment group, it is useful to know the variability in the intensities that are not a result of the treatment. As was discussed earlier, this variability appears to be a function of the intensity level itself.

Estimating the relationship between the variability in intensity to the actual intensity level is not the main aim of this project, but it is valuable for two reasons. First it can be a simple way to examine the bias in the results from the hand chosen peaks, and it demonstrates how to analyze the variability in intensity. In the search for biomarkers the variability in intensity dictates how large a sample size is needed to detect a difference.

Figure 4.5 shows a pair of sequential peaks after the log transformation and normalization, the first with a high intensity and the second with a lower

intensity. It can be seen that the peak with the highest intensity also tends to have the most variation about that intensity level.

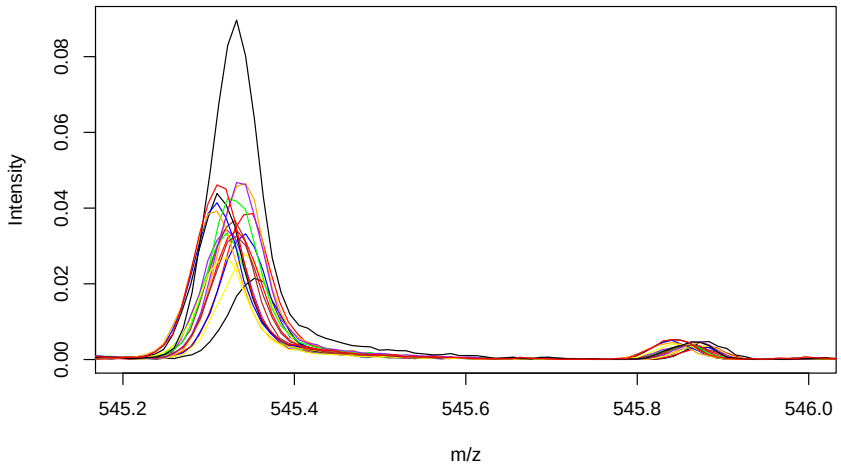

Figure 4.5: Variance in intensity depends on magnitude of intensity

Figure 4.6 shows these standard errors of the intensity as a function of the intensity itself, with the fitted equation given by

$$y = 9.82 \times 10^{-5} + 0.1352x \tag{4.4}$$

where y is the standard error of the intensity of the peaks in each group, and x is the average intensity level for each group of peaks. These estimates are improved from the preliminary work due to the larger sample size and the superior peak detection. The intercept has more than tripled, but when compared to the slope it is still practically zero. The slope had a 50% increase, which suggests that the peaks which were chosen by hand tend to be the peaks with less variability.

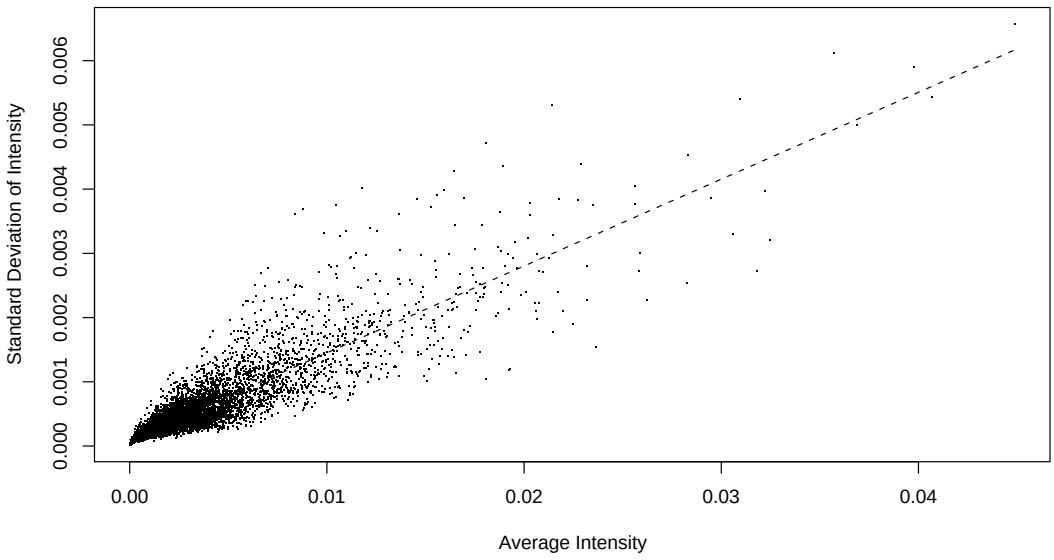

Figure 4.6: The standard error of intensity as a function of intensity

4.3.2 Variance Components

The variance components were estimated by maximizing (3.5) using SAS Proc MIXED. Each cluster of peaks yielded a different set of variance components, and by examining the entire set of estimates we can get a better understanding of the true nature of the variance components. The SAS code used to calculate these estimates is

```
proc mixed data=proteomics2 method=reml;
    class blank day vial trial;
    model intensity=blank/solution;
    random day vial(day) trial(day);
    by group;
run;
```

The "by" statement in the code will run the model on each set of m/z values separately. Since there were 7,665 clusters at unique m/z values, there are 7,665 different estimates of the variance components. Figure 4.7 shows a matrix plot of each of the four different variance components: day, vial, trial, and the error term. Larger versions of each plot will also be given for closer inspection.

It is worth mentioning which of these four sources of variability are contributing the most variation. The trial and vial components were by far the smallest sources (Figure 4.8). This speaks well of the pre-processing steps used to analyze these proteomic samples.

The day to day variation represented by the day variance component is higher compared to the vial or trial components (Figure 4.9 and Figure 4.10). This variability is from two different sources: storing the samples overnight, and

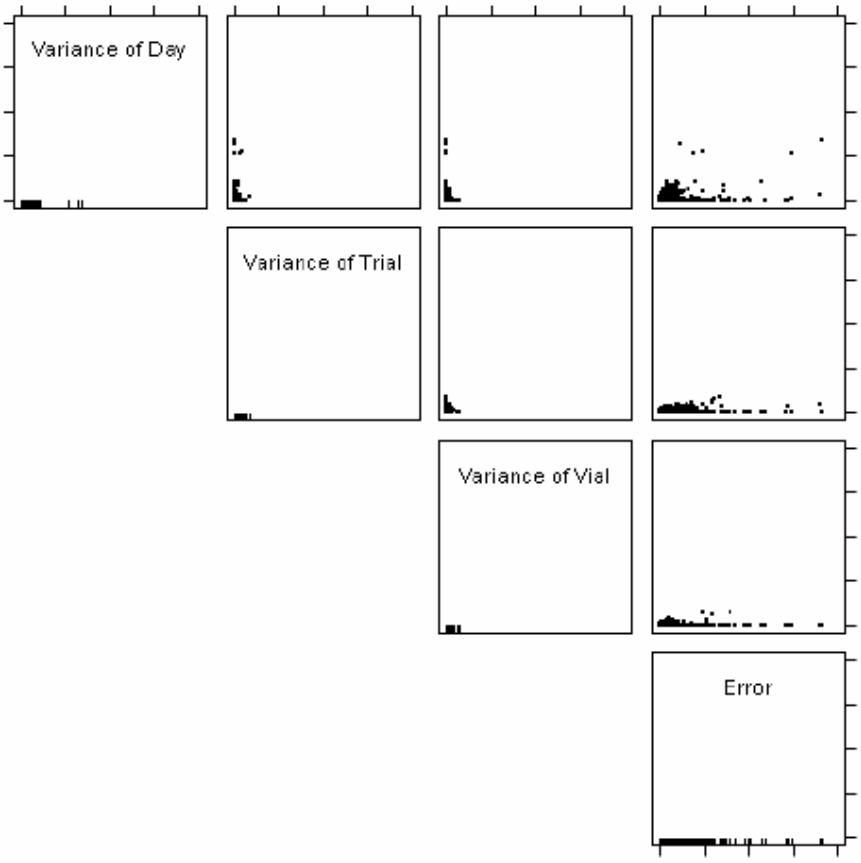

Figure 4.7: A matrix plot of the four variance components

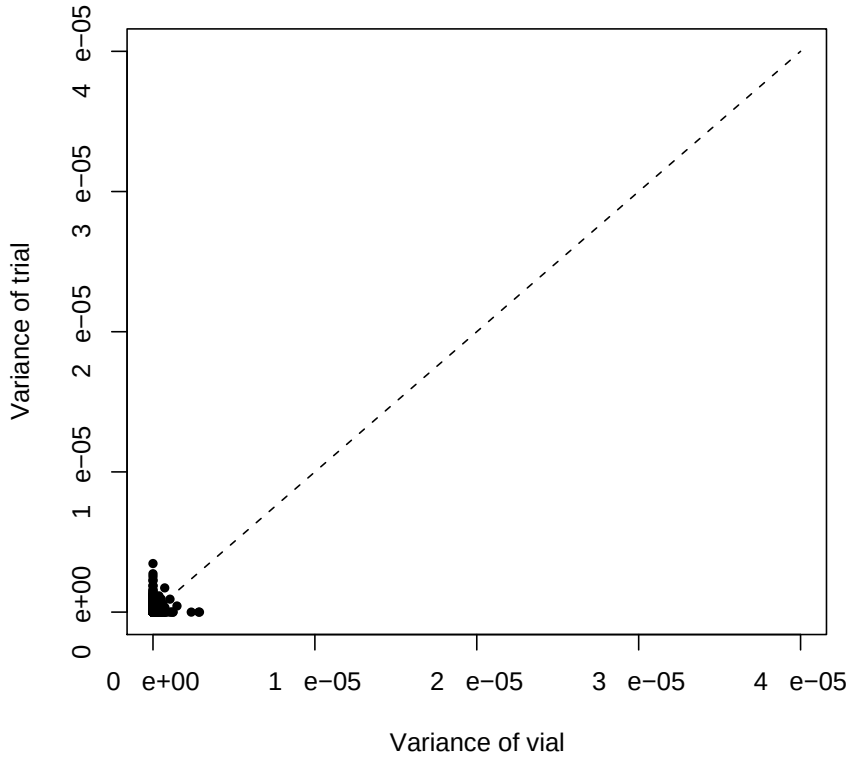

Figure 4.8: Variance of trial vs. variance of vial

the daily calibration of the ESI-TOF instrument. The day, vial, and trial sources contribute a relatively small amount of variability compared to the fourth factor, the random error.

The random error is a crossed effect of the trial by vial interaction with other measurement error, as shown in (3.2). This can be thought of as the error that occurs each time a sample is examined by the ESI-TOF. This is referred to as the machine variability in Chapter 2, and it was hoped that this variability would be negligible. Analyzing a sample in this machine should be automatic, without change in the process. Finding that the machine variability is higher than the vial variability (Figure 4.11) and higher than the trial variability (Figure 4.12) is an alarming discovery. This indicates that to detect a small difference between treatment groups an extremely large sample will be required.

The error term was usually higher than the variance of day, although the variance of day was larger than the variance of trial or vial (Figure 4.13). The day-to-day variability could be high due to the daily calibration of the machine. If that is the case, then the total machine variation would be the combination of the day-to-day variation and the error term. This encourages the idea that the machine variability is the largest source of variation in this experiment.

4.3.3 Effect of a blank run

A blank run was used after every three vials were analyzed in the ESI-TOF, but the question is whether they were needed. Some of the p-values for the

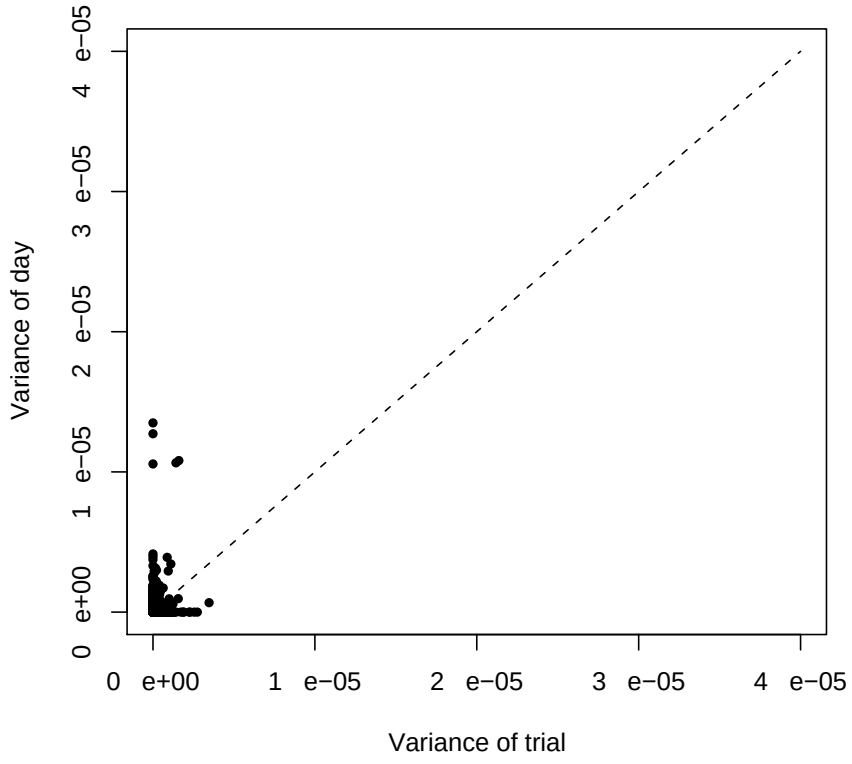

Figure 4.9: Variance of day vs. variance of trial

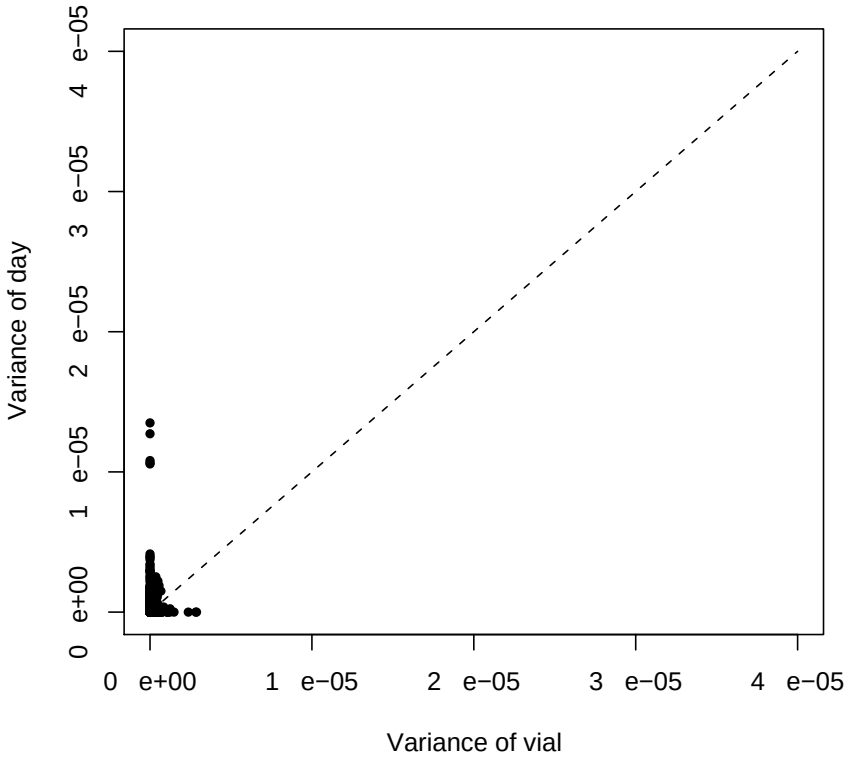

Figure 4.10: Variance of day vs. variance of vial

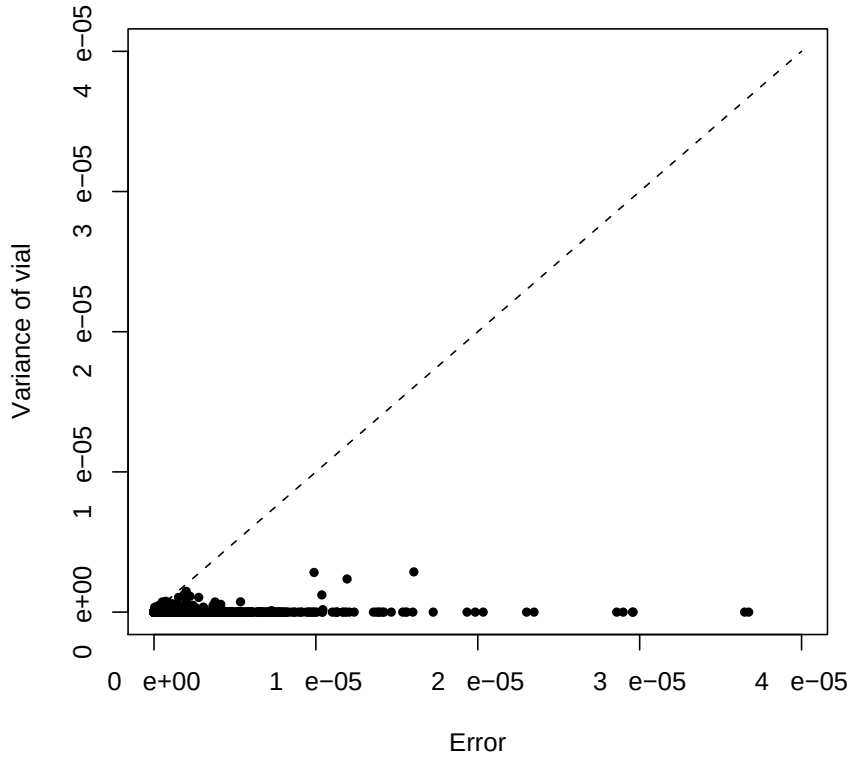

Figure 4.11: Variance of vial vs. random error

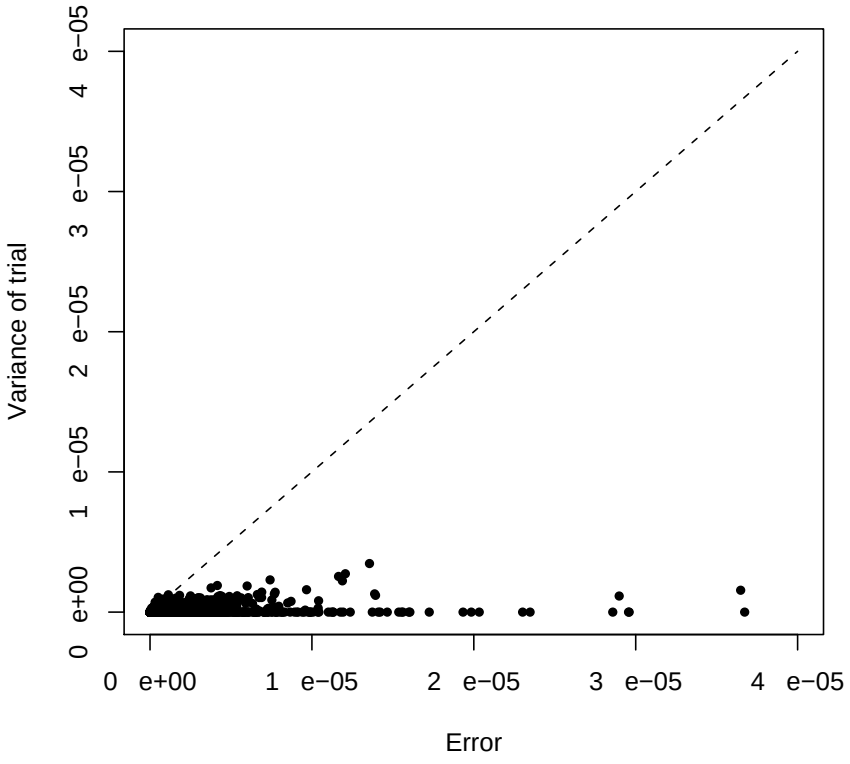

Figure 4.12: Variance of trial vs. random error

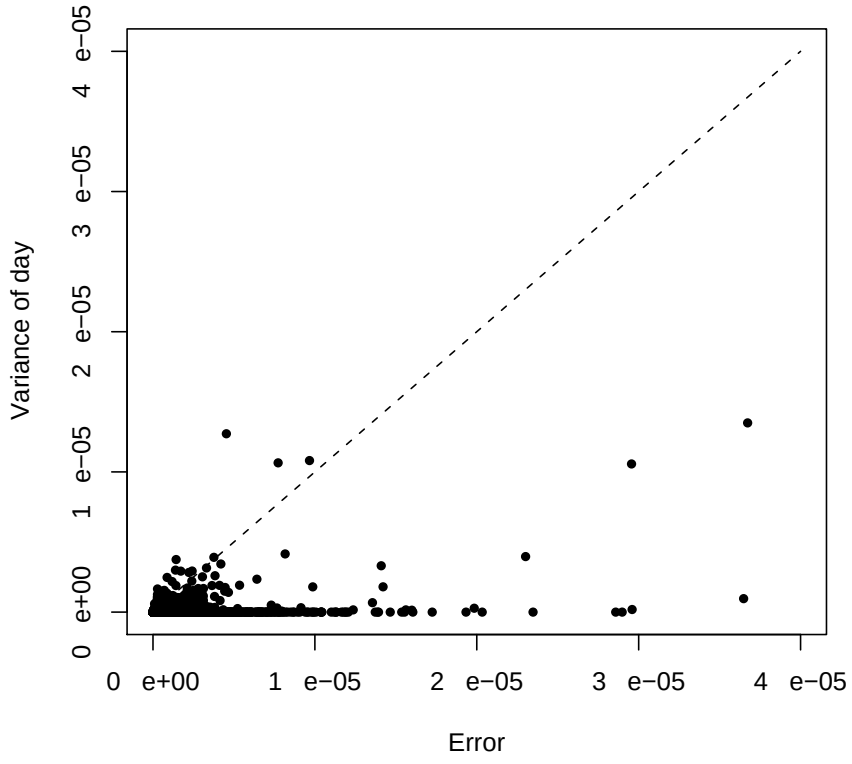

Figure 4.13: Variance of day vs. random error

significance of a blank run were very low, but with 7,665 clusters that should be expected. Even if the blank run is truly significant the effect is quite low compared to the intercept. The presence of a preceding blank run was used as a covariate. Figure 4.14 shows a histogram of the p-values for the blank run for the various peaks.

Even when a factor is statistically significant it may not be practically significant, which would be the case if the intercept was an order of magnitude greater than the effect of a blank run. Figure 4.16 shows the effect of the blank runs against the magnitude of the intercept to examine the practical significance.

Although the blank run does not appear to have a significant effect on the intensity levels, it could still be important. Perhaps blank runs do affect the variability, but the effect can only be noticed when blank runs are used after every fourth vial. While this experiment was not designed to analyze the effectiveness of a blank run, it appears that using a blank run more frequently would not have improved the results.

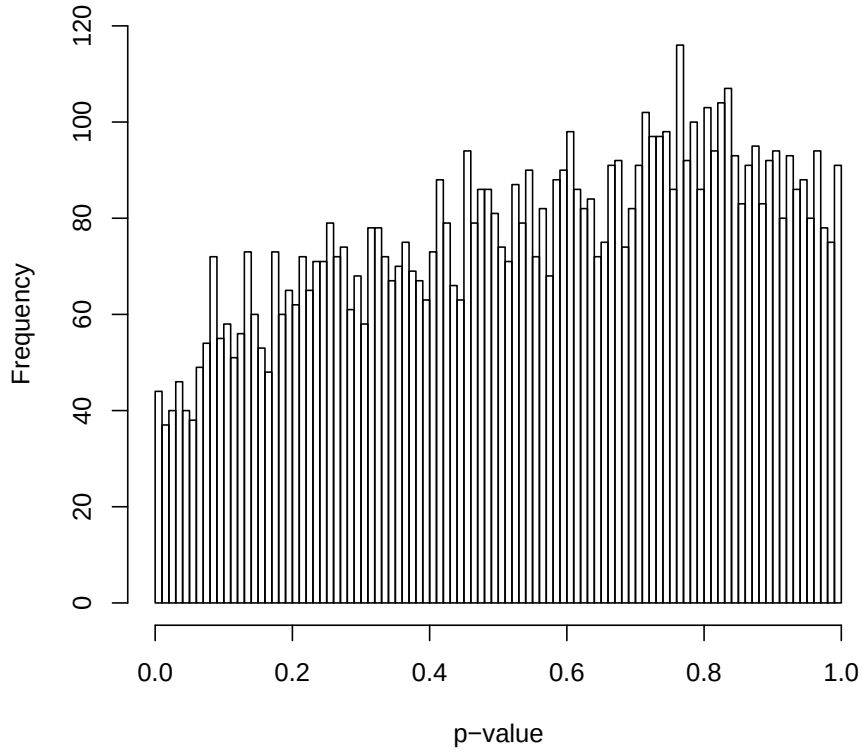

Figure 4.14: Histogram of the p-values for tests of the effect of a preceding blank run

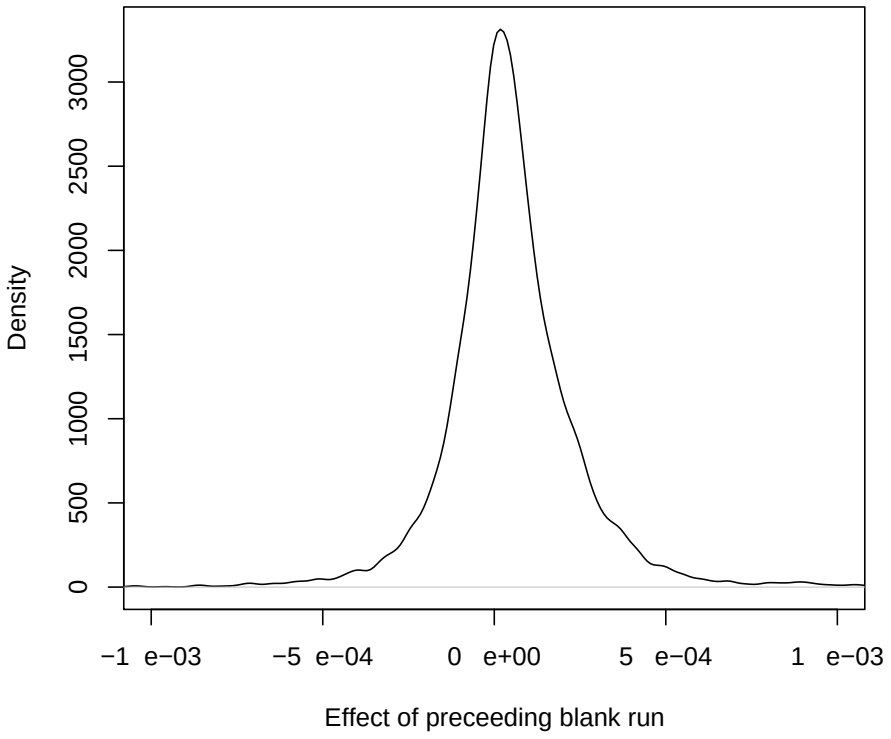

Figure 4.15: Density distribution of the effect of a preceding blank run

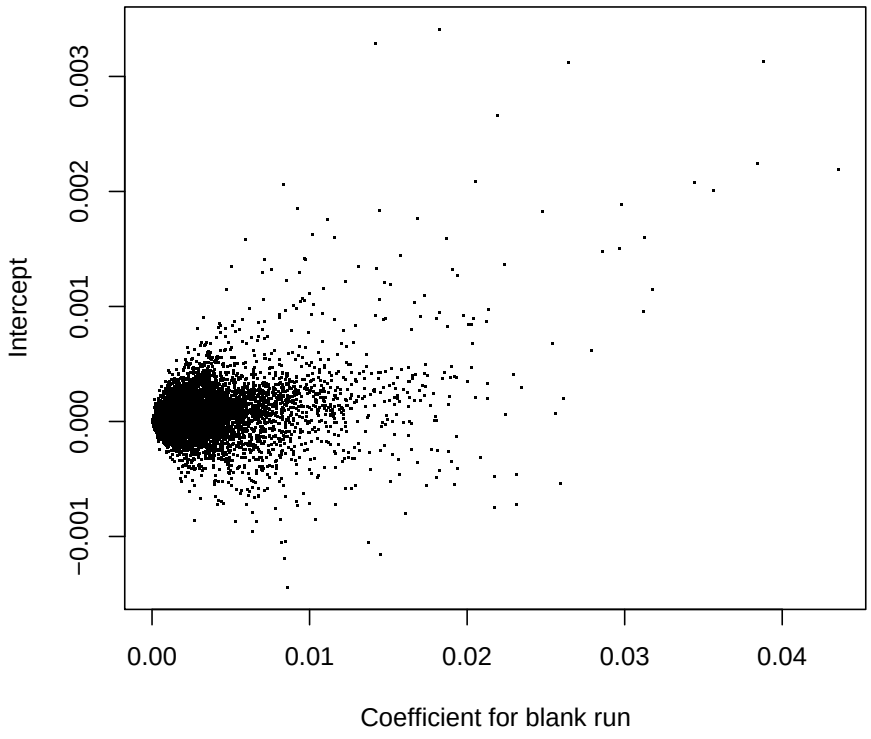

Figure 4.16: Coefficient of blank run vs. intercept

Chapter 5

Conclusions

5.1 Machine Variability

The results of this experiment cause concern about the amount of variability attributed to the ESI-TOF machine. The runs and the pre-processing steps contribute a negligible amount of variability, but the day to day changes and the individual runs through the ESI-TOF do increase the variability dramatically. The day to day variability may be due to the freezing of the vials, but it is most likely due to the daily calibration of the ESI-TOF. If this conclusion is correct it is ironic, because the daily calibration is designed to reduce systematic variability, but in this experiment it appears to increase the variability. When the daily variation is combined with the error term the total amount of error due to the machine itself is by far the largest source of variability.

This project cannot answer why the machine error is so large, nor can it suggest the proper procedure to correct that variability. There should be further experimentation to determine whether the machine is malfunctioning or if better procedures to calibrate the machine are needed. Until the machine variation is

understood and corrected it is doubtful that any biomarkers will be able to be discovered unless an extremely large sample size is used. The results of this analysis can be used to recommend the sample size that would be needed.

5.2 Future Work

This project should be viewed as a preliminary study designed to highlight areas for future research. The main focus of this project was to identify sources of variability, but only four sources were examined. More experimentation could be done to quantify other sources of variability not explored in this study, such as variability due to dietary changes or the location where the blood was drawn. Further experimentation could be used to separate the variance components used in this study, such as separating the machine variability into the error due to the autosampler from the error due to the affinity column and mass spectrometer.

The algorithms used in this paper were thoughtfully designed, and extensively tested, but are far from optimal. The peak detection and peak alignment algorithms could be improved by combining other methods discussed in the literature review, or perhaps with a new algorithm specially designed to handle the problems unique to proteomics. Even the parameters used in the algorithms were chosen because they appeared to yield the best results, while a more rigorous approach might find a better solution.

Bibliography

Anderson, N. L. and Anderson, N. G. (2002), "The human plasma proteome: History, character, and diagnostic prospects," *Molecular and Cellular Proteomics*, 1, 845–867.

Baggerly, K. A., Edmonson, S. R., Morris, J. S., and Coombes, K. R. (2004a), "High-resolution serum proteomic patterns for ovarian cancer detection, Letter to the Editor," *Endocrine-Related Cancer*, 11, 583–584.

Baggerly, K. A., Morris, J. S., and Coombs, K. R. (2004b), "Reproducibility of SELDI-TOF protein patterns in serum: Comparing datasets from different experiments," *Bioinformatics*, 20 no. 5, 777–785.

Benjamini, Y. and Hochberg, Y. (1995), "Controlling the false discovery rate: A practical and powerful approach to multiple testing," *Journal of the Royal Statistical Society*, Series B, 57, 289–300.

Bensamil, H., Golek, J., Moody, M. M., Semmes, J. O., and Haoudi, A. (2005), "A novel approach for clustering proteomics data using Bayesian fast fourier transform," *Bioinformatics*, 21 no. 10, 2210–2224.

Bertsch, W. (1999), "Two-dimensional gas chromatrography: Concepts, instrumentation, and applications," *Journal of High Resolution Chromatography*, 22, Issue 12, 647–665.

Bolstad, B. M., Irizarry, R. A., Astrand, M., and Speed, T. P. (2002), "A comparison of normalization methods for high density oligonucleotide array data based on variance and bias," *Bioinformatics*, 19 no. 2, 185–193.

Brown, P. R. and Hartwick, R. A. (1989), *High Performance Liquid Chromatography*, Wiley-Interscience.

Check, E. (2004), "Proteomics and cancer: Running before we can walk?" *Nature*, 429, 496–497.

Cole, R. B. (1997), *Electrospray Ionization Mass Spectrometry*, John Wiley and Sons.

Conrads, T. P., Fusaro, V. A., Ross, S., Johann, D., Rajapakse, V., Hitt, B. A., Steinberg, S. M., Kohn, E. C., Fishman, D. A., Whiteley, G., Barrett, J. C., Liotta, L. A., Petricoin, E. F., and Veenstra, T. D. (2004), "High-resolution serum proteomic features for ovarian cancer detection," *Endocrine-Related Cancer*, 11, 163–178.

Constans, A. (2005), "Rethinking clinical proteomics," *The Scientist*, September 26, 2005, 20–21.

Diamandis, E. P. (2004), "Analysis of serum proteomic patterns for early cancer diagnosis: Drawing attention to potential problems," *Journal of the National Cancer Institute*, 96, 353–356.

Gentleman, R., Carey, V., Huber, W., Carey, V. J., Irizarry, R. A., and Dudoit, S. (2005), *Statistics for Biology and Health*, Springer Science + Business Media Inc.

Glasbey, C. A. and Mardia, K. V. (2001), "A penalized likelihood approach to image warping," *Journal of the Royal Statistical Society Series B*, 63, Part 3, 465–512.

Gross, J. H. (2004), *Mass Spectrometry: A Textbook*, Springer.

Harris, D. C. (1998), *Quantitative Chemical Analysis*, W.H. Freeman and Company.

Hartemink, A. J., Gifford, D. K., Jakkola, T. S., and Young, R. A. (2001), "Maximum likelihood estimation of optimal scaling factors for expression array normalization," *SPIE BIOS*, 6.

Izmirlian, G. (2004), "Application of the random forest classification algorithm to a SELDI-TOF proteomics study in the setting of a cancer prevention trial," *Annals of the New York Academy of Sciences*, 1020, 154–174.

Juang, B. H. and Rabiner, L. R. (1991), "Hidden Markov models for speech recognition," *Technometrics*, 33, No. 3, 251–272.

Kienle, M. G. (1992), "Determination of plasma (6,6-2H2) glucose by GC/MS," *Journal of Chromatography, Biomedia Applications*, 573, 127–131.

Kolata, G. B. (1974), "Control of protein synthesis (II): RNA in the nucleus," *American Association for the Advancement of Science*, 603–604.

Kuehl, R. O. (2000), *Design of Experiments: Statistical Principles of Research Design and Analysis*, Brooks/Cole Publishing Company.

Liotta, L. A., Petricoin, E. F., Veenstra, T. D., and Conrads, T. P. (2004), "High-resolution serum proteomic patterns for ovarian cencer detection, Letter to the Editor," *Endocrine-Related Cancer*, 11, 585–587.

Littel, R. C., Milliken, G. A., Stroup, W. W., and Wolfinger, R. D. (1996), *SAS Systems for Mixed Models*, SAS Institute Incorporated.

Master, S. R. (2005), "Diagnostic proteomics: Back to basics?" *Clinical Chemistry*, 51, 1333–1334.

Menster, M. K., Bombick, D. D., Flora, D. B., Bunning, T. J., Klei, H. E., and Crane, R. L. (1995), "Liquid crystalline copolymers in capillary gas chromatography," *Polymer Preprints*, 36, No. 1, 381–382.

Merrell, K., Southwick, K., Graves, S. W., Esplin, M. S., Lewis, N. E., and Thulin, C. E. (2004), "Analysis of low-abundance, low-molecular-weight serum proteins using mass spectrometry," *Journal of Biomolecular Techniques*, 15(4), 234–244.

Nelson, D. L. and Cox, M. M. (2000), *Lehninger Principles of Biochemistry, Third Edition*, Worth Publishers.

Nichols, T. and Hayasaka, S. (2003), "Controlling the familywise error rate in functional neuroimaging: A comparative review," *Statistical Methods in Medical Research*, 12(5), 419–446.

Petricoin, E. F., Ardekani, A. M., Hitt, B. A., Levine, P. J., Fusaro, V. A., Steinberg, S. M., Mills, G. B., Simone, C., Fishman, D. A., Kohn, E. C., and Liotta, L. A. (2002a), "Use of proteomic patterns in serum to identify ovarian cancer," *The Lancet*, 359.

Petricoin, E. F., Ornstein, D. K., Paweletz, C. P., Ardekani, A., Hackett, P. S., Hitt, B. A., Velassco, A., Trucco, C., Wiegand, L., Wood, K., Simone, C. B., Levine, P. J., Linehan, W. M., Emmert-Buck, M. R., Steinberg, S. M., Kohn, E. C., and Liotta, L. A. (2002b), "Serum proteomic patterns for detection of prostate cancer," *Journal of the National Cancer Institute*, 94, No. 20, 1576–1578.

Purohit, P. V. and Rocke, D. M. (2003), "Discriminant models for high-throughput proteomics mass spectrometer data," *Proteomics*, 3, 1699–1703.

Sauve, A. C. and Speed, T. P. (2004), "Normalization, baseline correction and alignment of high-throughput mass spectrometry data," *Proceedings of the Genomic Signal Processing and Statistics workshop*, May 26,27.

Sorace, J. M. and Zhan, M. (2003), "A data review and re-assessment of ovarian cancer serum proteomic profiling," *Bioinformatics*, 4, 24.

Storey, J. and Tibshirani, R. (2003), "Statistical significance for genomewide studies," *Proceedings of the National Academy of Sciences*, 100, 9440–9445.

Suizdak, G. (1994), "The emergence of mass spectrometry in biochemical research," *Proceedings of the National Academy of Sciences of the United States of America*, 91, 11290–11297.

Sullivan, M. G. (2005), "Controversy erupts over proteomics studies," *LookSmart*, April 15 2005.

Vlahou, A., Schellhammer, P. F., and Mendrinos, S. (2001), "Development of a novel proteomic approach for the detection of transitional cell carcinoma of the bladder in urine," *American Journal of Pathology*, 154, 1491–1502.

Wang, K. and Gasser, T. (1997), "Alignment of curves by dynamic time warping," *The Annals of Statistics*, 25, No. 3, 1251–1276.

Wulfsberg, G. (2000), *Inorganic Chemistry*, Macmillan's Global Academic Publishing.

Zhang, H., Yu, C. Y., and Singer, B. (2003), "Cell and tumor classification using gene exprerssion data: Construction of forests," *National Academy of Sciences*, 100, no. 7, 4168–4172.

Appendix A

Functions and global variables used in the other R codes

These functions are used in many of the other codes that are run in the programming language R. There are also a few global variables that are referenced by the functions.

```
setwd('C:\\Documents and Settings\\administrator\\Desktop\\Proteomics\\Data')
library(MASS)

##############################DATA FILES#######################################
info<-read.table('info.dat',header=TRUE)
vardata<-read.table('vardata.dat',header=T)
sdmatfix<-read.table('C:\\Documents and Settings\\administrator\\Desktop\\
Proteomics\\Data\\sdmatfix.dat',header=FALSE)
sdmatfix<-as.matrix(sdmatfix)
rownames(sdmatfix)<-c('Time1','Time2','Time3','Time4','Time5','Time6',
'Time7','Time8','Time9','Time10','Time11')
colnames(sdmatfix)<-c('500s','700s','900s','1100s','1300s','1500s',
'1700s','1900s','2100s','2300s')
summits<-read.table('C:\\Documents and Settings\\administrator\\Desktop\\
Proteomics\\Data\\summits.dat',header=FALSE)
summits<-as.matrix(summits)
rownames(summits)<-c('Time1','Time2','Time3','Time4','Time5','Time6',
'Time7','Time8','Time9','Time10','Time11')
colnames(summits)<-c('500s','700s','900s','1100s','1300s','1500s',
'1700s','1900s','2100s','2300s')

####################FUNCTIONS#################################################
##FUNCTION TO READ IN THE PEAK DATA
getpeakdata<-function(){
peaks<-matrix(0,0,5)
colnames(peaks)<-c('mz','intensity','sample','window','obs')
for(sample in 1:20){
for(window in 1:11){
```

```
name<-paste("peaks.",sample,".",window,".dat",sep="")
data<-as.matrix(read.table(name,header=T))
data<-cbind(data,rep(sample,nrow(data)),rep(window,nrow(data))
,cbind(seq(1,nrow(data))))
peaks<-rbind(peaks,data)
}
}
#add a column to keep track of peak groupings (-1 means no group)
peaks<-cbind(peaks,rep(-1,nrow(peaks)))
colnames(peaks)[6]<-'group'
peaks<-as.data.frame(peaks)
peaks
}

##FUNCTION TO PLOT PEAKS ACROSS SAMPLES
plotpeaks<-function(xbegin=500,xend=505,samples=20,window=1,pchp=19){
plot(peaks[peaks$sample==1 & peaks$window==window &
peaks$mz>xbegin & peaks$mz<xend,1],
     peaks[peaks$sample==1 & peaks$window==window &
peaks$mz>xbegin & peaks$mz<xend,2],
xlab="m/z",ylab="Intensity",col=colors()[30],pch=pchp,
xlim=c(xbegin-(xend-xbegin)*.1,xend))
for(i in 2:samples){
points(peaks[peaks$sample==i & peaks$window==window &
peaks$mz>xbegin & peaks$mz<xend,1],
      peaks[peaks$sample==i & peaks$window==window &
peaks$mz>xbegin & peaks$mz<xend,2],
col=colors()[30*i],pch=pchp)
}
legend("topleft",legend=paste("smpl",1:samples),
col=colors()[(1:samples)*30],pch=pchp)
}

## A FUNCTION TO WRITE A DATA SET TO A FILE
writedata<-function(data,name){
filename<-paste("C:\\Documents and
Settings\\administrator\\Desktop\\Proteomics\\Data\\",name,".dat",sep="")
write(t(data),filename,ncolumns=ncol(data))
}

##FUNCTION TO FIND THE MAXIMUM OF A CERTAIN VARIABLE IN A LIST
findmax<-function(lista,name,start=500,stop=2500){
listb<-lista[names(lista)==name]
maxa<-0
for(i in 1:length(listb)){
listb[[i]]<-listb[[i]][lista[[i*2-1]]<stop]
lista[[i*2-1]]<-lista[[i*2-1]][lista[[i*2-1]]<stop]
listb[[i]]<-listb[[i]][lista[[i*2-1]]>start]
if(max(listb[[i]])>maxa){
maxa<-max(listb[[i]])
}
}
```

```
maxa
}

##FUNCTION TO GET THE RIGHT IDS
getid<-function(window,runnumber=1){
ids<-info[info$time_window==window,1]
runid<-as.numeric(as.vector(ids)[runnumber])
runid
}

##FUNCTION TO PLOT A CERTAIN TIME BETWEEN CERTAIN WINDOWS
plotmz<-function(time,xmin,xmax=0,windows=0){
if(xmin<100){
xmax<-xmaxfix[time,xmin]
xmin<-xminfix[time,xmin]
}
colors<-c('black','red','blue','green','orange','purple',
'yellow','brown','magenta')
if(windows==0){
windows<-length(colors)
}
data<-as.list(NULL)
for(i in 1:windows){
getinfo<-getdata(time,getid(time,i),xmin,xmax)
data<-as.list(c(data,getinfo))
}
plot(data[[1]],data[[2]],type="l",col=colors[1],
ylim=c(0,findmax(data,"intens")),xlab="M/Z",
ylab="Intensity",main=paste("Time",time))
for(i in 2:windows){
lines(data[[2*i-1]],data[[i*2]],
col=colors[i%%length(colors)],type="l")
}
}

##FINDS THE STD DEV OF THE PEAKS AT TIME T BETWEEN SPECIFIED M/Z
sdofpeak<-function(time,xmin,xmax){
data<-as.list(NULL)
ids<-info[info$time_window==time,1]
for(i in 1:length(ids)){
getinfo<-getdata(time,ids[i],xmin,xmax)
data<-as.list(c(data,getinfo))
if(i==1){
names(data)<-c('MZ1','Intensity1')
}else{
names(data)<-c(names(data)[1:(length(data)-2)],
paste('MZ',i,sep=""),paste('Intensity',i,sep=""))
}

}
xs<-NULL
for(i in 1:length(ids)){
```

```
xs<-c(xs,mean((data[[i*2-1]])[data[[i*2]]==max(data[[i*2]])]))
}
sd(xs)
}

####A FUNCTION TO GET A CERTAIN PIECE OF DATA
getdata<-function(window,runid,start=500,stop=2500){
if(runid<50){       #WASN'T REALLY A RUNID, IT WAS A RUNNUMBER
runid<-getid(window,runid)
}
filename<-paste("C:\\Documents and Settings\\administrator\\Desktop\\
Proteomics\\Raw Data\\dataW",
window,"R",runid,".dat",sep="")
data<-read.table(filename)
names(data)<-c('mz','intens')
data<-data[data$mz>=start & data$mz<=stop,]
data
}

###GETS DATA FROM SEVERAL TIME WINDOWS
windowlist<-function(window){
datalist<-as.list(NULL)
for(i in 1:20){
getinfo<-getdata(window,i)
datalist<-as.list(c(datalist,getinfo))
}
datalist
}

#PLOTS A WINDOWLIST
plotlist<-function(data,start=500,end=2500,time="unknown"){
colors<-c('black','red','blue','green','orange','purple','yellow',
'brown','magenta')
plot(data[[1]],data[[2]],type="l",col=colors[1],
xlim=c(start,end),ylim=c(0,findmax(data,"intens",start,end)),
xlab="m/z",ylab="Intensity",main=paste("Time",time))
for(i in 2:(length(data)/2)){
lines(data[[2*i-1]],data[[i*2]],col=colors[i%%length(colors)],
type="l")
}
}

#CHOPS OUT VALUES BELOW A CERTAIN THRESHOLD
cutnoise<-function(data){
axis<-c(500,700,900,1100,1300,1500,1700,1900,2100,2300)
unnormal<-c(.2,.2,.2,.5,1.5,1,.6,.25,.2,.13)
normal<-c(.00015,.00025,.0002,.002,.0035,.0015,.0006,.003,.0002,.0002)
unnormalcut<-.4
normalcut<-.0002
for(i in 1:(length(data)/2)){
tempi<-data[[i*2]]
tempm<-data[[i*2-1]]
```

```
n<-length(tempm)
area<-sum((tempm[-1]-tempm[-n])*(tempi[-1]+tempi[-n])/2)
#ASSUME NOT NORMALIZED
if(abs(area-1)>5){
cutoff<-unnormalcut
}
#ASSUME NORMALIZED
if(abs(area-1)<=5){
cutoff<-normalcut
}
tempi[tempi<cutoff]<-0
data[[i*2]]<-tempi
}
data
}

##PERFORMS THE LOG TRANSFORMATION ON THE INTENSITIES
loglist<-function(data){
for(i in 1:(length(data)/2)){
data[[i*2]]<-log(data[[i*2]]+1)
}
data
}

###SETS TOTAL SIGNAL TO ONE
normalize<-function(data){
for(i in 1:(length(data)/2)){
tempm<-data[[i*2-1]]
tempi<-data[[i*2]]
n<-length(tempm)
area<-sum((tempm[-1]-tempm[-n])*(tempi[-1]+tempi[-n])/2)
data[[i*2]]<-data[[i*2]]/area
}
data
}

makepeakmatrix<-function(peaks,data){
runs<-20
peakintensmat<-matrix(0,length(peaks),runs)
peakmzmat<-matrix(0,length(peaks),runs)
for(peak in 1:length(peaks)){
for(run in 1:runs){
peakxy<-findpeak(peaks[peak],data[[run*2-1]],data[[run*2]])
peakintensmat[peak,run]<-peakxy$intens
peakmzmat[peak,run]<-peakxy$mz
}
}
answer<-list(intens=peakintensmat,mz=peakmzmat)
}

findpeak<-function(peak,mz,intens){
b0<-0.0052170670
```

```
b1<-0.001793642
peaksd<-b0*exp(b1*peak)
lower<-peak-3*peaksd
upper<-peak+3*peaksd
yvaluepeak<-max(intens[mz>lower & mz<upper])
xvaluepeak<-mz[intens==yvaluepeak]
xvaluepeak<-min(xvaluepeak[xvaluepeak>lower & xvaluepeak<upper])
answer<-list(mz=xvaluepeak,intens=yvaluepeak)
answer
}

#######################SETTING UP VARIABLES##############################
test5<-getdata(5,1)
test5lnc<-cutnoise(normalize(loglist(test5)))
logtest5<-loglist(getdata(5,1))
test1<-getdata(1,1)
#wtest5<-windowlist(5)
#wtest5n<-normalize(wtest5)
#wtest5nc<-cutnoise(wtest5)
#wtest5c<-cutnoise(wtest5)
#wtest5cl<-loglist(wtest5c)
#wtest5cln<-normalize(wtest5cl)
wtest5<-windowlist(5)
wtest5l<-loglist(wtest5)
wtest5ln<-loglist(wtest5l)
wtest5lnc<-cutnoise(wtest5ln)
sderrors<-cbind(stack(as.data.frame(summits))[,1],
stack(as.data.frame(sdmatfix))[,1])
colnames(sderrors)<-c('mz','error')
sderrors<-sderrors[sderrors[,1]!=0,]
sderrors<-sderrors[order(sderrors[,1]),]
peak5<-c(516.298,516.8446,555.3217,555.8708,606.382,
607.385,652.4168,654.9124,
708.4486,709.9602,756.4866,756.9905,802.4866,803.4991,852.5389,853.5562,
901.5246,957.5817,958.58,1010.65,1013.616,1062.63,1064.662,
1117.678,1119.615,1161.716,1162.726,1214.795,1218.479,1259.473,1300.765,
1301.751,1388.455,1392.353,1461.69,1486.765,1571.078,
1575.713,1624.287,1763.052,1783.762,1885.164,2229.556)
mats<-makepeakmatrix(peak5,wtest5)
meanintens<-apply(mats$intens,1,mean)
sdintens<-apply(mats$intens,1,sd)
peaks<-getpeakdata()
```

Appendix B

Code in R to get the preliminary results based on hand chosen peaks

This code was used to estimate the variability of peak center as a function of m/z. These peaks were chosen by hand, and the hand chosen estimates were used to improve those same estimates by restricting the window widths.

```
##MAKE A TABLE OF STANDARD DEVIATIONS AROUND VARIOUS PEAKS
xmins<-rbind(
cbind(501,701.5,902,1102.1,1309,1510.1,0,0,0,0),           #Time 1
cbind(500.8,709.5,904,1107.1,1309,1507,1717.2,1959,2115,0),       #Time 2
cbind(500,706,903,0,0,0,0,0,0,0),           #Time 3
cbind(501,704,900,1105,1308,0,0,0,0,0),               #Time 4
cbind(502,706.7,910,1104,1321.4,1539.6,1736,1940,2170,2331),    #Time 5
cbind(502,705.2,907,1110.2,1310,1508.5,1760,1942,2167,2320),      #Time 6
cbind(506,704.2,903,1104.4,1331.6,1522.5,1738,0,0,0),         #Time 7
cbind(502,701,903,1112.2,1304.1,0,1737,1931,0,0),           #Time 8
cbind(502.1,701,904.2,1105.3,1311.5,0,0,0,0,0),           #Time 9
cbind(502,703.4,905.2,1104.4,1314.1,1515.5,1703.5,1931.6,2125,2322.4),
#Time 10
cbind(520,703,900,1108,1305.4,1515.5,1738.6,1947.6,2112,2327.4)
#Time 11
)
rownames(xmins)<-c('Time1','Time2','Time3','Time4','Time5','Time6',
'Time7','Time8','Time9','Time10','Time11')
colnames(xmins)<-c('500s','700s','900s','1100s','1300s','1500s','1700s',
'1900s','2100s','2300s')
xmaxs<-rbind(
cbind(501.5,702,902.8,1102.9,1310,1510.8,0,0,0,0),
cbind(502,710,905,1107.8,1310,1508,1718.1,1959.5,2120,0),
cbind(500.8,707,904,0,0,0,0,0,0,0),
cbind(501.7,705,901.2,1106,1309.2,0,0,0,0,0),
cbind(503,707.2,911,1108,1322.1,1540.2,1739,1946,2173,2336),
cbind(502.8,705.8,908,1111.2,1310.6,1509.3,1766,1945,2175,2325),
cbind(506.8,704.8,904,1105,1332.1,1524.5,1740,0,0,0),
```

```
cbind(503,702,904,1113.2,1305.2,0,1740,1933,0,0),
cbind(502.6,702,905,1106,1312.5,0,0,0,0,0),
cbind(502.6,703.8,906,1105,1315.3,1516.5,1704.5,1932.1,2126,2322.9),
cbind(521,704,901,1109,1306.2,1516.5,1739.8,1948.2,2112.8,2327.9)
)
rownames(xmaxs)<-c('Time1','Time2','Time3','Time4','Time5','Time6',
'Time7','Time8','Time9','Time10','Time11')
colnames(xmaxs)<-c('500s','700s','900s','1100s','1300s','1500s',
'1700s','1900s','2100s','2300s')

##CREATE NEW MATRICIES WITH THE CENTER OF THE PEAKS OR THE LEFT ENDPOINT
centermat<-(xmins+xmaxs)/2
leftpoint<-xmins
for(row in 1:nrow(xmins)){
for(column in 1:ncol(xmins)){
if(leftpoint[row,column]==0){
leftpoint[row,column]<-leftpoint[row,column-1]
centermat[row,column]<-centermat[row,column-1]
}
}
}
centers<-apply(centermat,2,mean)

##FIND THE PEAKS FOR THE DISTRIBUTIONS (BETTER THAN CENTERMAT)
summits<-matrix(0,nrow(xmins),ncol(xmins))
par(mfrow=c(1,1))
for(row in 1:nrow(xmins)){
for(column in 1:ncol(xmins)){
if(xmins[row,column]!=0){
plotmz(row,xmins[row,column],xmaxs[row,column])
summits[row,column]<-locator(1)$x
}
}
}
summits<-read.table('C:\\Documents and Settings\\administrator\\Desktop\\
Proteomics\\Data\\summits.dat',header=FALSE)
summits<-as.matrix(summits)
rownames(summits)<-c('Time1','Time2','Time3','Time4','Time5',
'Time6','Time7','Time8','Time9','Time10','Time11')
colnames(summits)<-c('500s','700s','900s','1100s','1300s','1500s',
'1700s','1900s','2100s','2300s')

##CALCULATE THE STANDARD DEVIATIONS BASED ON MY BAD WINDOWS XMINS XMAXS
Sys.time()
sdmat<-matrix(0,nrow(xmins),ncol(xmins))
times<-c(1,2,3,4,5,6,7,8,9,10,11)
for(time in 1:nrow(xmins)){
for(window in 1:ncol(xmins)){
if(xmins[time,window]==0){
xmins[time,window]<-xmins[time,window-1]
xmaxs[time,window]<-xmaxs[time,window-1]
sdmat[time,window]<-0
```

```
}else{
sdmat[time,window]=sdofpeak(times[time],
xmins[time,window],xmaxs[time,window])
}
}
}

#READ IN THE MATRIX OF STANDARD DEVIATIONS AND FIX THE REDUNDANCIES
sdmat<-read.table('C:\\Documents and
Settings\\administrator\\Desktop\\Proteomics\\Raw
Data\\dataWsdRmatrix.dat',header=FALSE)
for(row in 1:nrow(sdmat)){
for(column in ncol(sdmat):2){
if(sdmat[row,column]==sdmat[row,column-1]){
sdmat[row,column]<-0
}
}
}

##CALCULATE THE AVERAGE ERROR WITHOUT THE ZEROS
meansd<-as.vector(NULL)
test<-0
for(column in 1:ncol(sdmat)){
count<-0
sum<-0
for(row in 1:nrow(sdmat)){
if(sdmatold[row,column]!=0){
sum<-sum+sdmatold[row,column]
count<-count+1
}
}
meansd[column]<-sum/count
}

##REGRESSION ON THE STANDARD DEVIATIONS
regress<-lm(log(meansd)~centers)
b0<-regress$coefficients[[1]]
b1<-regress$coefficients[[2]]
b0<--5.094748
b1<-0.002061783

##A FEW STANDARD DEVIATIONS AS EXAMPLES
xx<-seq(500,2300,by=200)
se<-round(exp(b0+b1*xx),4)
rbind(as.character(xx),se)

####################BEGIN SECOND TRY (TO IMPROVE PREVIOUS ESTIMATES
##MAKE A NEW SET OF XMINS AND XMAXS WITH THE FIXED WINDOW SIZE
xminfix<-xmins
xmaxfix<-xmaxs
for(row in 1:nrow(xmins)){
for(column in 1:ncol(xmins)){
```

```
if(summits[row,column]!=0){
halfwidth<-3*(exp(b0+b1*summits[row,column]))
xminfix[row,column]<-summits[row,column]-halfwidth
xmaxfix[row,column]<-summits[row,column]+halfwidth
}
else{
xminfix[row,column]<-xminfix[row,column-1]
xmaxfix[row,column]<-xmaxfix[row,column-1]
}
}
}

###CALCULATE THE STANDARD DEVIATIONS WITH FIXED WINDOWS
sdmatfix<-matrix(0,nrow(xminfix),ncol(xminfix))
times<-c(1,2,3,4,5,6,7,8,9,10,11)
for(time in 1:11){
sdmatfix[time,1]<-sdofpeak(times[time],xminfix[time,1],xmaxfix[time,1])
for(window in 2:ncol(xminfix)){
if(xminfix[time,window]==xminfix[time,window-1]){
sdmatfix[time,window]<-0
}else{
sdmatfix[time,window]=sdofpeak(times[time],
xminfix[time,window],xmaxfix[time,window])
}
writedata(sdmatfix,"sdmatfix")
}
}
sdmatfix<-read.table('C:\\Documents and Settings\\administrator\\Desktop\\
Proteomics\\Data\\sdmatfix.dat',header=FALSE)
sdmatfix<-as.matrix(sdmatfix)
rownames(sdmatfix)<-c('Time1','Time2','Time3','Time4','Time5',
'Time6','Time7','Time8','Time9','Time10','Time11')
colnames(sdmatfix)<-c('500s','700s','900s','1100s','1300s','1500s',
'1700s','1900s','2100s','2300s')
sdmatold<-read.table('C:\\Documents and Settings\\administrator\\Desktop\\
Proteomics\\Data\\sdmatfix(bad).dat',header=FALSE)
rownames(sdmatold)<-c('Time1','Time2','Time3','Time4','Time5','Time6',
'Time7','Time8','Time9','Time10','Time11')
colnames(sdmatold)<-c('500s','700s','900s','1100s','1300s','1500s',
'1700s','1900s','2100s','2300s')
summitsold<-read.table('C:\\Documents and Settings\\administrator\\Desktop\\
Proteomics\\Data\\summits(bad).dat',header=FALSE)
rownames(summitsold)<-c('Time1','Time2','Time3','Time4','Time5','Time6',
'Time7','Time8','Time9','Time10','Time11')
colnames(summitsold)<-c('500s','700s','900s','1100s','1300s','1500s',
'1700s','1900s','2100s','2300s')

##CREATE VECTOR OF X AND Y COORDINATES FOR M/Z VS. SD DEV.
sderrors<-cbind(stack(as.data.frame(summits))[,1],
stack(as.data.frame(sdmatfix))[,1])
colnames(sderrors)<-c('mz','error')
sderrors<-sderrors[sderrors[,1]!=0,]
```

```
sderrors<-sderrors[order(sderrors[,1]),]
par(mfrow=c(1,1))
plot(sderrors[,1],sderrors[,2],xlab="M/Z",ylab="Standard
Error",ylim=c(0,.9),main="Standard Errors by M/Z")

##CREATE A MATRIX IS SDERRORS FOR A UNIFORM DISTRIBUTION (BAD PEAKS)
unifmat<-matrix(0,nrow(summits),ncol(summits))
for(row in 1:nrow(summits)){
for(column in 1:ncol(summits)){
if(summits[row,column]!=0){
a<-xminfix[row,column]
b<-xmaxfix[row,column]
unifmat[row,column]<-sqrt((b-a)^2/12)
}
}
}
##EXAMINE WHETHER ANY PEAKS ARE UNIFORM ACROSS THE DISTANCE
round(unifmat-sdmatfix,4)

##REGRESSION ON THE NEW FIXED WINDOWS STANDARD ERRORS
regress2<-lm(log(sderrors[,2])~sderrors[,1])
b02<-regress2$coefficients[[1]]
b12<-regress2$coefficients[[2]]
b02<--5.25582
b12<-0.001793642
xx<-seq(500,2500,length=100)
yy2<-exp(b02+b12*xx)
```

Appendix C

Code in R to detect peaks

First the data is log transformed and normalized, and then the peaks are identified and recorded.

```
#Given x and y vectors of data points
#Return new x and y vectors with coordinates of peak values
bgsmooth<-function(x,y,tol=.1,width=.1,scan=.2,stop=0){

#ALLOW THE ALGORITHM TO BE CUT SHORT
if(stop>0){
n<-stop
}else{
n<-length(x)
}

#Initializing values
peak<- rep(0,n)
bg<- rep(0,n)
background<- rep(0,n)
jstart<- 1
jend<- 1
smoothy<-y

#Loop through each x value
for(i in 1:n){

#Create the window
#xlow<- x[i]-zcritical*exp(B0+B1*x[i])
#xhigh<- x[i]+zcritical*exp(B0+B1*x[i])
xlow<-x[i]-width
xhigh<-x[i]+width
scana<-x[i]-scan
scanb<-x[i]+scan
```

```
j<- jstart-1
jdone<- 0

#Find the beginning of the window
#(fix if the beginning is before the first x)
while(jdone==0 & j<n){
j<- j+1
if(x[j]>=scana){
jdone<-1
jstart<-j
}
}
if(jend<n){
jdone<- 0
j<- jend-1

#Find the end of the window
#(fix if maximum is after last x)
while(jdone==0 & j<n){
j<-j+1
if(x[j]>scanb){
jdone<-1
jend<-j-1
}
}
if(jdone==0){
jend<- n
}
}

#Find the min and the max in the window
#And add up values for the mean
maxy<- y[jstart]
miny<- y[jstart]
sumy<- 0
sumy2<- 0
tolmean<-0
ny<- 0
toln<- 0
windowvalues<-rep(0,jend-jstart+1)
for(j in jstart:jend){
windowvalues[j-jstart+1]<-y[j]
if(x[j]>xlow & x[j]<xhigh){
if(y[j]>maxy){
maxy<- y[j]
}
if(y[j]<miny){
miny<- y[j]
}
sumy<- sumy+y[j]
sumy2<- sumy2+y[j]^2
ny<- ny+1
```

```
}
tolmean<-tolmean+y[j]
toln<- toln+1
}

#Calculate the mean value in the window for a simple smoother
meany<- sumy/ny
stddevy<- (sumy2-ny*meany^2)/(ny-1)
tolmean<-tolmean/toln
tolmad<-mad(windowvalues)
tolmedian<-median(windowvalues)

#Next find the range
rangey<- maxy-miny

#Calculate the tolerance
tolerance<-tolmean*tol
tolerance2<-tolmedian*tol
tolerancez<-tolmedian+tol*tolmad

#If it's not above the tolerance
if(maxy<tolerancez){
#not a peak
peak[i]<- 0
background[i]<- y[i]
#it is just background
bg[i]<- 1
}else{
#If it is above tolerance
#Then it is signal + background
background[i]<- 0
#it is not just background
bg[i]<- 0
#if this is the maximum in the window (the peak)
if(y[i]==maxy){
#This is a peak
peak[i]<- 1
}else{
peak[i]<- 0
#make this smoothed
smoothy[i]<- meany
}
}
}  #End the for(i in 1:n) for each x value

#Now do the background correction

istart<- FALSE
iend<- FALSE

i<-0
if(bg[1]==0){
```

```
bg[1]<- 1
}
while(i<n){
#Find the start and finish of each interval of noisy baseline
while(!istart & i<n){
i<- i+1
#If this is the beginning of a noisy valley
if(bg[i]==0){
istart<- TRUE
firsti<- i-1
}
}

if(istart){
i<-firsti
#Find the end of each baseline interval
while(!iend & i<n){
i<- i+1
if(bg[i]==1){
iend<- TRUE
lasti<- i
}
}
}

#Linear interplation below
if(istart & iend){
#RESET
istart<- FALSE
iend<- FALSE
m<- (y[lasti]-y[firsti])/(x[lasti]-x[firsti])
b<- y[lasti]-m*x[lasti]
for(k in (firsti+1):(lasti-1)){
background[k]<- m*x[k]+b
}
i<- lasti
}
} #end while(i<n) of fixing background interpolation

#Now get the background corrected version
for(i in 1:n){
smoothy[i]<-smoothy[i]-background[i]
if(smoothy[i]<0){
smoothy[i]<-0
}
}
answerx<-x[peak==1]
answery<-smoothy[peak==1]
answer<-list(x=answerx,y=answery)
answer
}
```

```
#####################################################################
###EXAMINATION OF PARAMETERS
width<-.12
tolerance<-2
scan<-1.8
x1a<-500
x2a<-505
x1b<-1105
x2b<-1110
par(mfrow=c(2,1))
plot(test5ln,xlim=c(x1a,x2a),type='l',
ylim=c(0,max(test5ln$intens[test5ln$mz>x1a & test5ln$mz<x2a])),
main=paste("width:",width," tol:",tolerance," scan:",scan,sep=""))
bg5<-bgsmooth(test5ln$mz[test5ln$mz>x1a &
test5ln$mz<x2a],test5ln$intens[test5ln$mz>x1a &
test5ln$mz<x2a],width=width,tol=tolerance,scan=scan,stop=0)
#lines(bg5$x,bg5$y,lty=2)
points(bg5$x,bg5$y,pch=2,col="red")
plot(test5ln,xlim=c(x1b,x2b),type='l',
ylim=c(0,max(test5ln$intens[test5ln$mz>x1b & test5ln$mz<x2b])),
main=paste("width:",width," tol:",tolerance," scan:",scan,sep=""))
bg5<-bgsmooth(test5ln$mz[test5ln$mz>x1b &
test5ln$mz<x2b],test5ln$intens[test5ln$mz>x1b &
test5ln$mz<x2b],width=width,tol=tolerance,scan=scan,stop=0)
#lines(bg5$x,bg5$y,lty=2)
points(bg5$x,bg5$y,pch=2,col="red")

test5<-getdata(5,1)
test5ln<-normalize(loglist(test5))
histx<-.0015
mzsplit<-700
hist(test5ln$intens[test5ln$intens<histx &
test5ln$mz<mzsplit],breaks=200,xlim=c(0,histx),main="Histogram of
intensities of m/z values below 700",xlab="Intensity",ylab="Frequency")
hist(test5ln$intens[test5ln$intens<histx &
test5ln$mz>mzsplit],breaks=200,xlim=c(0,histx),main="Histogram of
intensities of m/z values above 700",xlab="Intensity",ylab="Frequency")
hist(test5ln$intens[test5ln$intens<histx],breaks=200,
xlim=c(0,histx),main="histogram of intensities",
xlab="Intensity",ylab="Frequency")
tolerance<-.0005

data<-test5ln[test5ln$mz>900,]
peakpeek<-data$intens<tolerance
peakwidths<-0
front<-0
end<-0
for(i in 2:length(peakpeek)){
if(peakpeek[i-1]==0 & peakpeek[i]==1){
front<-data$mz[i]
}
if(peakpeek[i-1]==1 & peakpeek[i]==0 & front>0){
```

```
end<-data$mz[i]
}
if(front>0 & end>0){
peakwidths<-c(peakwidths,end-front)
front<-0
end<-0
}
}
hist(peakwidths[peakwidths>.2 &
peakwidths<.6],breaks=250,xlim=c(.2,.6),main="Estimates of peak widths
above m/z value of 900",xlab="Estimated peak widths")

## A FUNCTION TO WRITE A DATA SET TO A FILE
writedata<-function(data,name){
filename<-paste("C:\\Documents and
Settings\\administrator\\Desktop\\Proteomics\\Data\\",name,".dat",sep="")
write(t(data),filename,ncolumns=ncol(data))
}

###############################################################################
###PEAK PICKING AND SAVING ALL THE DATA
width<-.12
tolerance<-2
scan<-1.8
#CREATE A LIST OF SMOOTHED OUTPUT
#bglist<-as.list(NULL)
#FOR EACH OF THE 20 SAMPLES RUN
for(sample in 1:20){
#bglist<-as.list(NULL)
#FOR EACH OF THE 11 TIME WINDOWS
for(window in 1:11){
data<-getdata(window,sample)
data<-normalize(loglist(data))
data<-bgsmooth(data$mz,data$intens,width=width,tol=tolerance,scan=scan)
data<-cbind(data$x,data$y)
filename<-paste("C:\\Documents and Settings\\administrator\\Desktop\\
Proteomics\\Data\\Peaks.",sample,".",window,".dat",sep="")
write(c('x','y'),filename,2)
write(t(data),filename,ncolumns=ncol(data),append=TRUE)
}
}
```

Appendix D

Code in R to align peaks across samples

Takes the output from the peak detection and clusters them together across the twenty samples.

```
##ALIGNMENT OF PEAKS ACROSS SAMPLES

setwd('C:\\Documents and Settings\\administrator\\Desktop\\Proteomics')

#######################################################################
##READING IN THE PEAK DATA
peaks<-matrix(0,0,5)
colnames(peaks)<-c('mz','intensity','sample','window','obs')
for(sample in 1:20){
for(window in 1:11){
name<-paste("Data\\peaks.",sample,".",window,".dat",sep="")
data<-as.matrix(read.table(name,header=T))
data<-cbind(data,rep(sample,nrow(data)),rep(window,nrow(data))
,cbind(seq(1,nrow(data))))
peaks<-rbind(peaks,data)
}
}
#add a column to keep track of peak groupings (-1 means no group)
peaks<-cbind(peaks,rep(-1,nrow(peaks)))
colnames(peaks)[6]<-'group'
peaks<-as.data.frame(peaks)

#######################################################################
##FUNCTION TO PLOT PEAKS ACROSS SAMPLES
plotpeaks<-function(xbegin=500,xend=505,samples=20,window=1,pchp=19){
plot(peaks[peaks$sample==1 & peaks$window==window &
peaks$mz>xbegin & peaks$mz<xend,1],
     peaks[peaks$sample==1 & peaks$window==window &
peaks$mz>xbegin & peaks$mz<xend,2],
main=paste("Plot of peaks between ",xbegin," and ",xend),
xlab="m/z",ylab="Intensity",col=colors()[30],pch=pchp,
```

```
xlim=c(xbegin-(xend-xbegin)*.1,xend))
for(i in 2:samples){
points(peaks[peaks$sample==i & peaks$window==window &
peaks$mz>xbegin & peaks$mz<xend,1],
      peaks[peaks$sample==i & peaks$window==window &
peaks$mz>xbegin & peaks$mz<xend,2],
col=colors()[30*i],pch=pchp)
}
legend("topleft",legend=paste("smpl",1:samples),
col=colors()[(1:samples)*30],pch=pchp)
}


##########################################################################
##Plotting one specific group
plotpeakgroup<-function(group,pchp=19,zscore=4,xmin=-1,xmax=-1,
ymin=-1,ymax=-1,type="plot"){
plotdata<-peaks[peaks$group==group,]
meanmz<-mean(plotdata$mz)
meani<-mean(plotdata$intensity)
sdmz<-getsdmz(meanmz)
sdi<-getsdi(meani)
if(xmin==-1){
xmin<-meanmz-zscore*sdmz
}
if(xmax==-1){
xmax<-meanmz+zscore*sdmz
}
if(ymin==-1){
ymin<-min(0,meani-zscore*sdi)
}
if(ymax==-1){
ymax<-meani+zscore*sdi
}
if(type=="plot"){
plot(plotdata$mz[plotdata$sample==1],
plotdata$intensity[plotdata$sample==1],
main=paste("Group",group),xlab='m/z',ylab='Intensity',
xlim=c(xmin,xmax),
ylim=c(ymin,ymax),
pch=pchp,col=colors()[30])
}
if(type=="points"){
points(plotdata$mz[plotdata$sample==1],
plotdata$intensity[plotdata$sample==1],
pch=pchp,col=colors()[30])
}
for(i in 2:20){
if(nrow(plotdata[plotdata$sample==i,])!=0){
points(plotdata$mz[plotdata$sample==i],
plotdata$intensity[plotdata$sample==i],
col=colors()[30*i],pch=pchp)
```

```
}
}
legend("topleft",legend=paste("smpl",1:20),
col=colors()[(1:20)*30],pch=pchp)
}

plotpeakgroup(group=1,xmin=499,xmax=505,ymin=0,ymax=.0033)
for(i in 2:10){
plotpeakgroup(group=i,type='points')
}


#########################################################################
##THE ALGORITHM IN FORTRANABLE CODE
###THE FOUR VECTORS NEED TO MATCH (LIKE A DATA FRAME)
grouppeaks<-function(mz,intensity,sample,window,zscore=3.090){
group<-rep(-1,length(mz))
thispeak<-0
b0mz<--5.26
b1mz<-.00179
b0i<-.0000282
b1i<-.299
for(thiswindow in 1:1){
mzspot<--1  #WHICH M/Z VALUE TO INITIALIZE WITH (will start at 0)

#VARIABLES FOR THE PLACES WHERE EACH SAMPLE BEGINS AND ENDS
a<-rep(0,20)
b<-rep(0,20)

#FIND THE BEGINNING AND END OF EACH SAMPLE IN THIS WINDOW
currentsample<-0
for(i in 1:length(mz)){
if(window[i]==thiswindow & currentsample==0){
currentsample<-sample[i]
a[currentsample]<-i
}
if(i==length(mz) & currentsample>0){
b[currentsample]<-i
currentsample<-0
}else{
if(currentsample>0 & (window[i+1]!=thiswindow |
sample[i+1]!=currentsample)){
b[currentsample]<-i
currentsample<-0
}
}
}

#The number of mz values in sample 1 for this time window
lastmz<-b[1]-a[1]

#while there are still mz values to explore in sample 1
```

```
while(mzspot<=lastmz){

##LOOKING FOR THE NEXT PEAK CENTER
thispeak<-thispeak+1 #the number of peaks already found
mzspot<-mzspot+1 #the number of points from sample 1 used to start
peakmzs<-rep(-1,20) #the mz indexes of the peaks in each sample

#KEEP A MATRIX OF SOLUTIONS (TO AVOID OSCILLATION)
solutionrows<-0
oldsolutions<-matrix(0,1,20)

#START THE INTIAL GUESS AS THE NEXT DATA POINT IN SAMPLE 1
initialize<-TRUE
meanmz=0;meani=0;sdmz=0;sdi=0
#keep track of whether the algorithm has converged
changes<-TRUE

while(changes){

for(i in 1:20){
if(initialize){
#PICK INITIAL GROUP (as next point from sample 1)
peakmzs[i]<-mzspot #the peak is the mzth spot in the arrary
meanmz<-mz[a[i]+mzspot] #start at a[1] and continue mzspot more
meani<-intensity[a[i]+mzspot]
sdmz<-exp(b0mz+b1mz*meanmz)
sdi<-b0i+b1i*meani
initialize<-FALSE
}else{
leftpoint<-meanmz-zscore*sdmz
rightpoint<-meanmz+zscore*sdmz
toppoint<-meani+zscore*sdi
bottompoint<-meani-zscore*sdi
distance<-9999999
winner<-0
#find unused dots within the bubble
for(j in a[i]:b[i]){
if(mz[j]>leftpoint & mz[j]<rightpoint & intensity[j]>
bottompoint & intensity[j]<toppoint & group[j]==-1){
zi<-(intensity[j]-meani)/sdi
zmz<-(mz[j]-meanmz)/sdmz
thisdistance<-sqrt(zmz^2 + zi^2)
if(thisdistance<distance){
winner<-j
}
}

}
if(winner>0){

thismz<-mz[winner]
thisi<-intensity[winner]
```

```
peakmzs[i]<-winner-a[i]

#reinitialize values
n<-0
for(j in 1:20){
if(peakmzs[j]>-1){
n<-n+1
}
}
meanmz<-(meanmz*(n-1)+thismz)/n
meani<-(meani*(n-1)+thisi)/n
sdmz<-exp(b0mz+b1mz*meanmz)
sdi<-b0i+b1i*meani

} #END if thisdata wasn't empty
}# end if first initializing
} #end for i in 1:20

#CHECK IF THE ITERATIONS ARE LOOPING
if(solutionrows==1){
for(i in 1:solutionrows){
matches<-0
for(j in 1:20){
if(t(as.matrix(oldsolutions))[i,j]==peakmzs[j]){
matches<-matches+1
}
}
if(matches==20){
changes<-FALSE
}
}
}
if(solutionrows>1){
for(i in 1:solutionrows){
matches<-0
for(j in 1:20){
if(oldsolutions[i,j]==peakmzs[j]){
matches<-matches+1
}
}
if(matches==20){
changes<-FALSE
}
}

}
solutionrows<-solutionrows+1
if(solutionrows==1){
oldsolutions<-peakmzs
}else{
newsolutions<-matrix(0,solutionrows,20)
if(solutionrows==2){
```

```
newsolutions[1:(solutionrows-1),]<-
t(as.matrix(oldsolutions)) [(solutionrows-1),]
}else{
newsolutions[1:(solutionrows-1),]<-oldsolutions
}
newsolutions[solutionrows,]<-peakmzs
oldsolutions<-newsolutions
}

} #end while there have been changes==TRUE

#Store how far it's gone along the m/z axis for sample 1
#which is used to initialize the next loop
mzspot<-peakmzs[1]
#NOW LABEL THE PEAKS ACCORDING TO THEIR GROUP
for(i in 1:20){
if(peakmzs[i]>-1){
group[a[i]+peakmzs[i]]<-thispeak
}
}

#PLACED IN HERE JUST FOR DEBUGGING PURPOSES
#if(thispeak==3){
# mzspot<-999999
#}
} #END LOOPING DOWN THE MZ AXIS
filename<-paste("C:\\Documents and Settings\\administrator\\Desktop\\
Proteomics\\Data\\groups.",thiswindow,".dat",sep="")
write(group,filename)

}#END LOOPING THROUGH EACH WINDOW
#answer<-list(group=group,oldsolutions=oldsolutions,
peakmzs=peakmzs,peaks=thispeak)
answer<-1
answer
}#END FUNCTION

Sys.time()
test<-grouppeaks(peaks$mz,peaks$intensity,
peaks$sample,peaks$window,zscore=3.090)
Sys.time()

mz<-peaks$mz;intensity<-peaks$intensity;sample<-
peaks$sample;window<-peaks$window;zscore<-3.090
plotpeakgroup(group=1,xmin=499,xmax=505,ymin=0,ymax=.0033)
for(i in 2:3){
plotpeakgroup(group=i,type='points')
}

#####################################################################
##Combining the data to be used by SAS
#Add group data to peaks
```

```
sumgroups<-0
#Do 1 and 11 seperately because the program crashed after 1
groups<-scan("groups.1.dat")
groups[groups!=-1]<-16270+groups[groups!=-1]
peaks$group[groups!=-1]<-groups[groups!=-1]
groups<-scan("groups.11.dat")
peaks$group[groups!=-1]<-groups[groups!=-1]
numgroups<-max(peaks$group)
numnow<-NULL
for(i in 1:numgroups){
numnow<-c(numnow,sum(peaks$group==i))
if(numnow[i]<20){
peaks$group[peaks$group==i]<--2
}
}
#####################################################################
##HISTOGRAM OF THE CLUSTER SIZES
hist(numnow,main="",xlab="Cluster size")

peaks2<-peaks[peaks$group!=-1,]
peaks2<-peaks2[peaks2$group!=-2,]
write.table(peaks2,'dataframe2.dat',6,row.names=FALSE,append=FALSE)

#####################################################################
##FUNCTION TO PLOT CLUSTERS ACROSS SAMPLES
plotclusters<-function(xbegin=500,xend=505,window=1,pchp='x',type=1){
if(type==1){
plot(peaks2[peaks2$sample==1 & peaks2$window==window &
peaks2$mz>xbegin & peaks2$mz<xend,1],
     peaks2[peaks2$sample==1 & peaks2$window==window &
peaks2$mz>xbegin & peaks2$mz<xend,2],
xlab="m/z",ylab="Intensity",col=colors()[30],pch=pchp,
xlim=c(xbegin-(xend-xbegin)*.1,xend))
}else{
points(peaks2[peaks2$sample==1 & peaks2$window==window &
peaks2$mz>xbegin & peaks2$mz<xend,1],
     peaks2[peaks2$sample==1 & peaks2$window==window &
peaks2$mz>xbegin & peaks2$mz<xend,2],
col=colors()[30],pch=pchp)
}
for(i in 2:20){
points(peaks2[peaks2$sample==i & peaks2$window==window &
peaks2$mz>xbegin & peaks2$mz<xend,1],
      peaks2[peaks2$sample==i & peaks2$window==window &
peaks2$mz>xbegin & peaks2$mz<xend,2],
col=colors()[30*i],pch=pchp)
}
legend("topleft",legend=paste("smpl",1:20),
col=colors()[(1:20)*30],pch=pchp)
}

plotclusters(xbegin=500,xend=505,window=5)
```

```
####################################TOYING WITH PLOTS
numclusters<-function(window,start){
peaks2[peaks2$window==window & peaks2$sample==1 &
peaks2$mz>start & peaks2$mz<start+5,]
}

examinegraphs<-function(window,start){
plotmz(xmin=start,xmax=start+5,time=window)
plotpeaks(xbegin=start,xend=start+5,window=window,type=2)
plotclusters(xbegin=start,xend=start+5,window=window,type=2)
}
```

Appendix E

Code in R to estimate variability of intensity as a function of intensity

Uses a linear model to estimate how variability in intensity changes in relation to peak height.

```
##########CALCULATING THE VARIANCE OF THE INTENSITY
data<-read.table('dataframe2.dat',header=TRUE) #AKA peaks2
data<-data[order(data$group),]
rownames(data)<-1:nrow(data)
intensdata<-matrix(0,0,2)
colnames(intensdata)<-c('iaverage','istddev')
for(i in 1:(nrow(data)/20)){
thisdata<-data$intensity[(i*20-19):(i*20)]
if(length(thisdata)!=20){
"ERROR!!"
}
intensdata<-rbind(intensdata,cbind(mean(thisdata),sd(thisdata)))
}

fit<-lm(istddev~iaverage,data=as.data.frame(intensdata))
summary(fit)
b0<-fit$coefficients[1]
b1<-fit$coefficients[2]

xx<-seq(0,max(intensdata[,1]),length=500)
yy<-b0+b1*xx
plot(intensdata,pch='.',xlab='Average Intensity',
ylab='Standard Deviation of Intensity')
lines(xx,yy,lty=2)
```

Appendix F

SAS Code to calculate variance components across samples

Estimates the variance components from each peaks, and stores those estimates in a data set.

```
option formDlim="~" ps=1000 ls=110;
s
data proteomics;
infile 'c:\Documents and Settings\administrator\Desktop\
Proteomics\Data\dataframe2.dat' firstobs=2;
input mz intensity sample window obs group;
run;

*proc print data=proteomics;
*run;

data proteomics2;
set proteomics;
if sample=1 then do;
day=1;
vial='A';
run=1;
blank='Y';
end;
if sample=2 then do;
day=1;
vial='D';
run=1;
blank='N';
end;
if sample=3 then do;
day=1;
vial='C';
run=1;
blank='N';
end;
if sample=4 then do;
day=1;
vial='B';
```

```
run=1;
blank='Y';
end;
if sample=5 then do;
day=1;
vial='E';
run=1;
blank='N';
end;
if sample=6 then do;
day=1;
vial='E';
run=2;
blank='N';
end;
if sample=7 then do;
day=1;
vial='C';
run=2;
blank='Y';
end;
if sample=8 then do;
day=1;
vial='A';
run=2;
blank='N';
end;
if sample=9 then do;
day=1;
vial='B';
run=2;
blank='N';
end;
if sample=10 then do;
day=1;
vial='D';
run=2;
blank='Y';
end;
if sample=11 then do;
day=2;
vial='A';
run=1;
blank='Y';
end;
if sample=12 then do;
day=2;
vial='D';
run=1;
blank='N';
end;
if sample=13 then do;
```

```
day=2;
vial='C';
run=1;
blank='N';
end;
if sample=14 then do;
day=2;
vial='B';
run=1;
blank='Y';
end;
if sample=15 then do;
day=2;
vial='E';
run=1;
blank='N';
end;
if sample=16 then do;
day=2;
vial='E';
run=2;
blank='N';
end;
if sample=17 then do;
day=2;
vial='C';
run=2;
blank='Y';
end;
if sample=18 then do;
day=2;
vial='A';
run=2;
blank='N';
end;
if sample=19 then do;
day=2;
vial='B';
run=2;
blank='N';
end;
if sample=20 then do;
day=2;
vial='D';
run=2;
blank='Y';
end;
keep intensity group day vial run blank;
run;

*proc print data=proteomics2;
*run;
```

```
proc sort data=proteomics2;
by group;
run;

*ODS TO ESTIMATE VARIANCE COMPONENTS FOR ALL THE PEAKS;
*ods trace on;
*ods output CovParms=vcs;
ods output SolutionF=solutions;

*proc varcomp data=proteomics2 method=reml;
*class blank day vial run;
*model intensity=blank day vial(day) run(day);
*by group;
*run;

proc mixed data=proteomics2 method=reml;
class blank day vial run;
model intensity=blank/solution;
random day vial(day) run(day);
by group;
run;

*ods trace off;
ods output close;

proc print data=solutions;
run;

proc print data=vcs;
run;

*COMBINING INTO A MATRIX (INSTEAD OF A VECTOR);
data vcs2;
set vcs;
if(CovParm = "day") then do;
cluster=group;
varvial=.;
varerror=.;
varrun=.;
varday=Estimate;
end;
if(CovParm = "vial(day)") then do;
cluster=group;
varvial=Estimate;
varerror=.;
varrun=.;
varday=.;
end;
if(CovParm = "run(day)") then do;
cluster=group;
varvial=.;
```

```
varerror=.;
varrun=Estimate;
varday=.;
end;
if(CovParm = "Residual") then do;
cluster=group;
varvial=.;
varerror=Estimate;
varrun=.;
varday=.;
end;
keep cluster varvial varerror varrun varday;
run;

data vcsrun;
set vcs2;
where varrun is not missing;
clusterrun = cluster;
keep clusterrun cluster varrun;
run;

data vcserror;
set vcs2;
where varerror is not missing;
clustererror = cluster;
keep clustererror cluster varerror;
run;

data vcsvial;
set vcs2;
where varvial is not missing;
clustervial = cluster;
keep clustervial cluster varvial;
run;

data vcsday;
set vcs2;
where varday is not missing;
clusterday = cluster;
keep clusterday cluster varday;
run;

proc sql;
create table components as
select
clustererror, varrun, varerror, clustererror as cluster2
from
vcsrun INNER JOIN vcserror ON (clusterrun=clustererror);
run;

proc sql;
create table components2 as
```

```
select
cluster2, varrun, varerror, varday, cluster as cluster2
from
components INNER JOIN vcsday ON (cluster2=clusterday);

proc sql;
create table components3 as
select
cluster2, varrun, varerror, varday, varvial, cluster as cluster2
from
components2 INNER JOIN vcsvial ON (cluster2=clustervial);

proc print data=components3;
run;

proc export data=components3
outfile = 'c:\Documents and Settings\administrator\Desktop\
Proteomics\Data\components.dat'
dbms=tab
replace;
run;
```

Appendix G

Code in R to plot the figures given in the text

The figures given in the text were not necessary to analyze the data, but were useful in clarifying or explaining aspects of the project. This is the code used to create those plots.

```
#######################PLOTS FOR THE PROJECT#########################

###############################################################################
#MAKE FOUR PEAKS (TO SHOW 3-D NATURE OF MASS SPEC)
fourpeaks<-function(x,y){
answer<-(.5*dnorm(x,2,1)+.5*dnorm(x,8,1))*
(.5*dnorm(y,3,1)+.5*dnorm(y,12,1))
answer
}
xx<-seq(0,12,length=50)
yy<-seq(0,15,length=50)

griddler<-outer(xx,yy,fourpeaks)
persp(griddler,xlab="m/z",ylab="Time",zlab="Intensity",theta=20,phi=45)

###############################################################################
#THE RAW DATA ACROSS THE ENTIRE m/z AXIS
plot(test5,type="l",ylim=c(0,60),ylab="Intensity",xlab="m/z")

###############################################################################
#THE LOGGED DATA ACROSS THE ENTIRE m/z AXIS
plot(loglist(test5),type="l",ylim=c(0,5),ylab="Intensity",xlab="m/z")

###############################################################################
#ONE WINDOW SHOWING THE NUMBER OF PEAKS IN A 50 m/z RANGE
plot(test5,type="l",ylab="Intensity",xlab="m/z",xlim=c(500,550),ylim=c(0,15))

###############################################################################
#ONE WINDOW SHOWING THE NUMBER OF PEAKS IN A 10 m/z RANGE
plot(test5,type="l",ylab="Intensity",xlab="m/z",xlim=c(500,510),ylim=c(0,15))

###############################################################################
#ONE WINDOW SHOWING A NOISY REGION WITH VERY LITTLE SOUND
```

```
plot(test5,type="l",ylab="Intensity",
xlab="m/z",xlim=c(1700,1725),ylim=c(0,.4))

#############################################################################
#RAW PLOTS
plotlist(wtest5,605.5,607.2,5)
plotlist(wtest5,1458,1465,5)

#############################################################################
#CUT-OFF PLOTS
plotlist(wtest5c,605.5,607.2,5)
plotlist(wtest5c,1458,1465,5)
plotlist(wtest5nc,605.5,607.2,5)

#############################################################################
#LOG TRANSFORM PLOTS
plotlist(wtest5cl,605.5,607.2,5)
plotlist(wtest5cl,1458,1465,5)

#############################################################################
#NORMALIZED PLOTS
plotlist(wtest5cln,605.5,607.2,5)
plotlist(wtest5cln,1458,1465,5)
plotlist(wtest5n,605.5,607.2,5)

#############################################################################
##SHOWING HOW VARIANCE OF INTENSITY DEPENDS ON INTENSITY
plotlist(wtest5lnc,545.2,546,5)

#############################################################################
##SHOWING HOW VARIANCE OF m/z IS AT THE 700 RANGE
plotlist(wtest5lnc,705.2,705.8,5)

#############################################################################
##The graph to show difficult parts of the algorithm
trypeaks<-function(jit=.1){
size<-200
xx<-seq(0,6,length=size)
peaks<-dnorm(xx,1.2,.3)+4.5*dnorm(xx,3,.8)+dnorm(xx,4.8,.4)*1
jitter<-NULL
yy<-NULL
jitter[1]<-0
for(i in 2:size){
jitter[i]<-runif(1,-jit,jit)
yy[i]<-max(0,peaks[i]+jitter[i])
}
yy[1]<-0
yy[size]<-0
yy<-peaks+jitter
plot(xx,yy,type="l",xlab="m/z",ylab="Intensity")
abline(v=c(1.2,3,4.8))
}
```

```
store<-as.integer(runif(1,0,1000))
set.seed(store)
#set.seed(93)
set.seed(349)
trypeaks(.3)

#############################################################################
###PLOTTTING A BASELINE PROBLEM
library(evd)
width<-.45
num=10
xxbegin<-600
xxend<-610
gumbela<-604.3
gumbelb<-1.5
scale<-5
shift<-.5
pinheight<-.5
jitteramount<-.4
xx<-seq(xxbegin,xxend,by=.01)
yy<-rep(0,length(xx))
for(i in 1:length(xx)){
jit<-rchisq(1,jitteramount)
raise<-dgumbel(xx[i],gumbela,gumbelb)*scale
normals<-rbind((pinheight*dnorm(xx[i],
seq(xxbegin-1,xxend,length=num),width)))
peakchange<-(cbind(dgumbel(seq(xxbegin-1,xxend,length=num),
gumbela,gumbelb)*scale/2)+shift)
yy[i]<-normals%*%peakchange+raise+jit
}
xx<-(xx-600)*10+600
plot(xx,yy,type="l",xlab="m/z",ylab="Intensity")

#############################################################################
###A HISTROGRAM OF THE INTENSITES (TO SHOW LOG TRANSFORMATION)
hist(test5lnc$intens,breaks=100,freq=FALSE,xlab="Peak Intensity",
ylab="Relative Frequency",main="",xlim=c(0,.002))

#############################################################################
###A HISTOGRAM OF THE INTENSITIES (AFTER LOG TRANSFORMATION)
hist(log(1+vardata$y2),breaks=30,freq=TRUE,
main="Histogram of 43 chosen peaks in log of 1 + intensity",
xlab="Log of Peak Intensity",ylab="Relative Frequency")

#############################################################################
##SHOWING A MASS SPEC BLIP
set.seed(18)
n<-500
xx<-seq(601,602,length=n)
jitter<-rep(0,length(xx))
for(i in 2:n){
jitter[i]<-rnorm(1,jitter[i-1],.04)
```

```
}
yy<-dnorm(xx,601.3,.1)+dnorm(xx,601.8,.1)*.5+jitter
yy[n/2]<-yy[n/2]+3
w<-5
for(i in 1:w){
yy[n/2-i]<-yy[n/2-i]+3/w*(w-i)
yy[n/2+i]<-yy[n/2+i]+3/w*(w-i)
}
plot(xx,yy,type="l",xlab="m/z",ylab="Intensity")

#########################################################################
###PLOTTING VARIANCE OF m/z BY m/z
#GET THE UNIFORM VARIANCE ESTIMATES
xx<-seq(500,2500,length=100)
b0old<--5.094748
b1old<-0.002061783
uniferrors<-sqrt((2*3*exp(b0old+b1old*xx))^2/12)
##REGRESSION ON THE NEW FIXED WINDOWS STANDARD ERRORS
b0<--5.25582
b1<-0.001793642
yy2<-exp(b0+b1*xx)
plot(sderrors[,1],sderrors[,2],xlab="m/z",ylab="Standard Error",
ylim=c(0,.6))
#,main="Standard Error of m/z as a function of m/z")
#sub=paste("Log(Std Err) =",b0,"+",b1,"* m/z"),cex.sub=.7,font.sub=3)
#lines(xx,uniferrors,lty=2)
lines(xx,yy2,lty=1)
#legend("topleft",legend=c('Uniform Standard Error',
#'Expected Standard Error calculated by log regression'),lty=c(2,1))

#####################################################################
##Histogram of m/z values used for the log-linear regression
hist(sderrors[,1],breaks=10,xlab='m/z',main="")

#####################################################################
###Variance of Intensity by Intensity
i0<-.0000282
i1<-.2988815
mats<-makepeakmatrix(peak5,wtest5ln)
meanintens<-apply(mats$intens,1,mean)
sdintens<-apply(mats$intens,1,sd)
meanintens2<-meanintens[-1]
sdintens2<-sdintens[-1]
plot(meanintens2,sdintens2,
main="Standard Error of Intensity by Average Intensity",
xlab="Intensity Level (Average)",
ylab="Standard Error of the Intensity")
#sub=paste("Regression line: SE = ",i0," + ",i1," I"),font.sub=3,cex.sub=.7)
xxi<-seq(0,max(meanintens2),length=1000)
yyi<-i0+i1*xxi
lines(xxi,yyi)
```

```
#########################################################################
##CREATE A PLOT WITH PEAKS AND VALLEY NOISE TO SHOW ALGORITHM
set.seed(42)
n<-500
xx<-seq(699,703,length=n)
jitter<-rep(0,length(xx))
m<-.1/6
b<-.2
yy<-dnorm(xx,700.2,.1)+dnorm(xx,701.7,.2)*3
for(i in 2:n){
jitter[i]<-rnorm(1,(jitter[i-1]*(1-1/20*yy[i])),b-m*yy[i])
}
yy<-yy+jitter
yy<-abs(yy)
plot(xx,yy,type="l",xlab="m/z",ylab="Intensity")

#################################################################
##Plotting the vial variance by machine variance
data<-read.table('varvialerror.dat',header=TRUE)
scale<-6
scaledata<-data
scaledata$varvial<-data$varvial*10^scale
scaledata$varerror<-data$varerror*10^scale
par(mfrow=c(1,1))
plot(scaledata$varerror,scaledata$varvial,
xlab="Variance of Run (machine)",ylab="Variance of Vial (preproccesing)"
,xlim=c(0,30),ylim=c(0,30))
abline(c(0,1),lty=2)

#################################################################
##PLOT PEAK GROUPS
plotpeaks(xbegin=1200,xend=1205,window=5)

#################################################################
##Plotting effect of blank run
data<-read.table('blankdata.dat',header=TRUE)
hist(data$Probt[data$Effect=="blank"],breaks=100,
xlab="p-value for effect of blank",main="")
plot(data$Estimate[data$Effect=="blank"],
data$Estimate[data$Effect=="Intercept"],
xlab="Coefficient for blank run",
ylab="Coefficient for Intercept",pch='.')

#################################################################
##Histograms of variance components
data<-read.table('components.dat',header=TRUE)
temp<-data[,2]
data[,2]<-data[,4]
data[,4]<-data[,5]
data[,5]<-data[,3]
data[,3]<-temp
names(data)<-c('Cluster','Variance of Day',
```

```
'Variance of Trial','Variance of Vial','Error')
pairs(data[,-1],pch=20,xlim=c(0,4*10^-5),
ylim=c(0,4*10^-5),lower.panel=NULL,diag.panel=rug)

xlimtop<-4*10^-5
ylimtop<-xlimtop
xx<-seq(0,xlimtop,length=500)
pchp<-20
plot(data$varrun,data$varday,xlim=c(0,xlimtop),ylim=c(0,ylimtop),
pch=pchp,ylab="Variance of day",xlab="Variance of trial")
lines(xx,xx,lty=2)
plot(data$varvial,data$varday,xlim=c(0,xlimtop),ylim=c(0,ylimtop),
pch=pchp,ylab="Variance of day",xlab="Variance of vial")
lines(xx,xx,lty=2)
plot(data$varerror,data$varday,xlim=c(0,xlimtop),ylim=c(0,ylimtop),
pch=pchp,ylab="Variance of day",xlab="Error")
lines(xx,xx,lty=2)
plot(data$varvial,data$varrun,xlim=c(0,xlimtop),ylim=c(0,ylimtop),
pch=pchp,xlab="Variance of vial",ylab="Variance of trial")
lines(xx,xx,lty=2)
plot(data$varerror,data$varrun,xlim=c(0,xlimtop),ylim=c(0,ylimtop),
pch=pchp,xlab="Error",ylab="Variance of trial")
lines(xx,xx,lty=2)
plot(data$varerror,data$varvial,xlim=c(0,xlimtop),ylim=c(0,ylimtop),
pch=pchp,xlab="Error",ylab="Variance of vial")
lines(xx,xx,lty=2)

################################################
#OTHER STUFF
plot(c(-100,100),c(-100,100),type="l",xlim=c(0,30),ylim=c(0,30),
xlab="Variance of Run (machine)",
ylab="Variance of Vial (preprocessing)",
main="Variance of Vial by Variance of Run")
#plotlist(test,1942.5,1944.5,5)
```