Sparse Proteomics Analysis - a compressed sensing-based approach for feature selection and classification of high-dimensional proteomics mass spectrometry data

Tim OF Conrad[1,*], Martin Genzel[2], Nada Cvetkovic[1], Niklas Wulkow[1], Jan Vybiral[3], Gitta Kutyniok[2], Christof Schütte[1,4],

1 Department of Mathematics, Freie Universität Berlin, Germany
2 Department of Mathematics, Technische Universität Berlin, Germany
3 Department of Mathematical Analysis, Charles University, Prague, Czech Republic
4 Zuse Institute Berlin, Berlin, Germany

*** Corresponding author: conrad@math.fu-berlin.de**

Abstract

Motivation High-throughput proteomics techniques, such as mass spectrometry (MS)-based approaches, produce very high-dimensional data-sets. In a clinical setting one is often interested how MS spectra differ between patients of different classes, for example spectra from healthy patients vs. spectra from patients having a particular disease. Machine learning algorithms are needed to (a) identify these discriminating features and (b) classify unknown spectra based on this feature set. Since the acquired data is usually noisy, the algorithms should be robust to noise and outliers, and the identified feature set should be as small as possible.

Results We present a new algorithm, *Sparse Proteomics Analysis* (SPA), based on the theory of Compressed Sensing that allows to identify a minimal discriminating set of features from mass spectrometry data-sets. We show how our method performs on artificial and real-world data-sets.

Availability The source-code can be downloaded from our homepage: `http://software.medicalbioinformatics.de`

1 Introduction

During the last decade, high-throughput assays systems for measuring a variety of different biological sources have become standard in modern laboratories. This allows for the quick and cheap creation of very large data-sets which characterize for example the status of a cell by its billions of constituents, e.g. nucleotides, RNAs, contained proteins or metabolites. Ideally, analyzing these massive data-sets leads to a better understanding of the underlying biological processes. Especially in the context of characterization and - ultimately understanding - diseases, a first step is often to find significant differences in the data between samples from healthy and diseased individuals. There are many successful examples where this approach based on -omics data (e.g. genomics, proteomics or metabolomics) led to the identification of biological markers, enabling a new type of molecular diagnostics. We call a set of biological markers that represent the differences on the data level a *disease fingerprint*.

Many disease-relevant mechanisms are controlled by proteins (e.g. hormones) which can be detected in biological samples (blood, urine, etc.) using mass spectrometry (MS). Mass spectrometry allows (potentially) for monitoring the entire set of proteins - the so-called proteome - in a given sample.

Through its wide availability in hospitals, MS-based proteomics can bring the next wave of progress in diagnostics, since even subtle changes in the proteome can be detected and linked to disease onset and progression [1–4].

The main idea of the identification of *disease fingerprints* using MS-based proteomics is sketched in Fig. 1: (a) A mass spectrum is generated reflecting the constitution of a given blood-sample, with respect to contained molecules. (b) Based on mass spectra from two sample groups (representing a healthy control group and a group having a particular disease), differences are detected. We call these differences a *disease fingerprint* since it represents a trace caused by a particular disease in the proteome. Several studies have shown that this approach works well in practice and found differences do indeed reflect correlations between changes in the mass spectrum, the proteome, and phenotypic changes ([5–9]). Panels of proteomic markers (fingerprints) have been shown to be more sensitive and specific than conventionally biomarker approaches [2], for example for diagnosing cancer [10–12]. However, a single proteomics data-set can contain tens of millions of signals which is many orders of magnitudes larger than the number of available observations in a typical study.

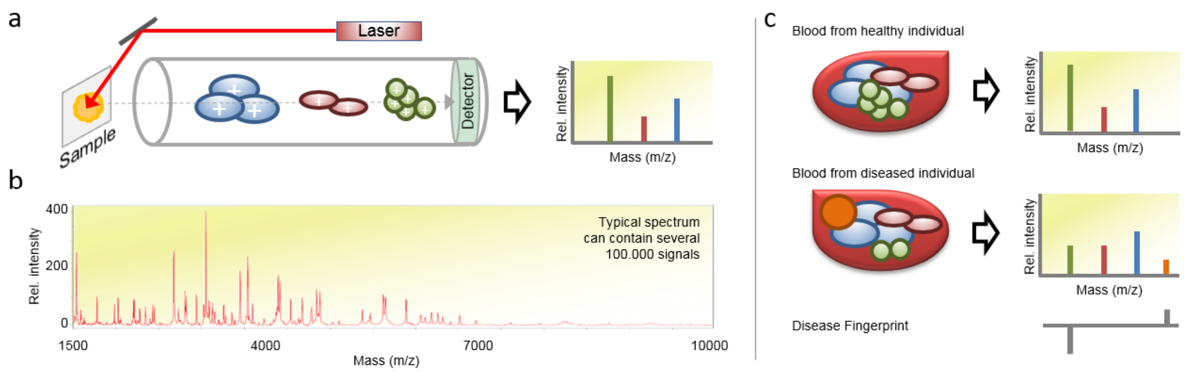

Figure 1. (a) Schematic outline of a linear matrix-assisted laser desorption ionization (MALDI) - time-of-flight (TOF) mass spectrometer (MS). During the measurement process, molecules in a sample are ionized, vaporized and finally analyzed by their respective time-of-flight through an electric field. This process generates a plot (mass spectrum) having mass-to-charge ratio (m/z) on the x-axis and intensity (ion count) on the y-axis. (b) Typical mass spectrum for a mass range of 1500 - 10.000 Dalton. (c) Example disease fingerprint, created by comparing mass spectra from a healthy and a diseased individual.

Our ultimate goal is to build a library of proteomics disease fingerprints which are extracted from high-throughput mass-spectrometry experiments. These would enable to diagnose diseases based on their proteomic fingerprints - just by analyzing an individual's proteome. However, the acquired data from the high-throughput experiments is very high-dimensional and contains a lot of noise levels which makes automatic analysis of mass spectra a very challenging task. Thus, automatic analysis and discovery of biomarkers is still an open research topic and there are several analytic problems that hinder reproduction of results (see e.g. [13]). Thus, a fingerprint should only consist of the minimal set of proteins specific for a particular disease and be robust to noisy measurements.

1.1 Problem Definition

In this article we focus on the following problem setting:

We assume that data is given in the form of pairs $\{x_i, y_i\}_{i=1...n}$ and the index i enumerates the n samples. Here, the $x_i \in \mathbb{R}^{d}$[1] represent the n mass spectra and $y_i \in \{-1, +1\}$ their respective classes, e.g. healthy or diseased. The goal is to identify a (small) set of features distinguishing the two classes. This

[1] For convenience reasons we assume that the data is centered, see below for details.

corresponds to the well known *feature selection* problem[2] and results in a potential disease fingerprint for the given data.

Mathematically, this can be formulated as identifying a sparse[3] vector $\omega \in \mathbb{R}^d$ such that $y_i = f_\omega(x_i)$ for all $i = 1, \dots, n$ with the *linear* decisions function $f_\omega(x_i) = \langle \omega, x_i \rangle = \sum_{j=1}^{d} \omega_j x_{i,j}$. Then the predicted class for a given spectrum x_i depends on the sign of $f_\omega(x_i)$[4]. Here, the entries of ω represent the significance of each data feature. However, in most realistic scenarios for feature selection problems the number of features is much larger than available samples ($d \gg n$) and the data contains noise and measurement errors. Hence, the number of possible classifiers ω can become extremely large, and *overfitting* can occur. In order to allow interpretability and generalization of the classifier, it is inevitable to restrict the solution space for ω. In this paper we are interested in very sparse solutions for ω which corresponds to a minimal fingerprint. We will approach this by formulating the feature selection problem as a regularized optimization problem:

$$\min_{\omega} \sum_{i=1}^{n} L(y_i, f_\omega(x_i)) + \lambda R(\omega), \tag{1}$$

where L is a *loss* (error) function, R is a *regularization* (cost) function that gives preference to a particular structure of ω (e.g. sparsity), and $\lambda \geq 0$ is a trade-off parameter choosing between model complexity and accuracy. Given a prediction $f_\omega(x)$ and a label y, the loss function $L(y, f_\omega(x))$ measures the discrepancy between the actual and the desired result.

As already pointed out, we are particularly interested in a method that produces *optimal and robust solutions* in the case where:

- the data (x and y) is noisy,
- the number of data dimensions, d, is large (typically: $d = 10^5 \dots 10^8$),
- the number of samples, n, is relatively small (typically: $n = 10^2 \dots 10^4$), and
- the selected feature set is small which corresponds to a small number of non-zero elements in ω (typically: $\#\{i \mid \omega_i \neq 0\} \ll 100$).

1.2 State of the Art in Sparse Feature Selection

There are numerous approaches for feature selection which mainly fall into three categories:

- **Filters**: Using some scoring or correlation function (e.g. based on Fisher's, t-test, information theoretic criteria) evaluating the importance of each feature and taking the top features.
- **Wrappers**: Using machine-learning algorithms to evaluate and choose features using some search strategy (e.g. simulated annealing or genetic algorithms)
- **Embedded methods**: Selecting variables by optimizing directly an objective function with respect to: goodness-of-fit and (optionally) number of features. This could be achieved with algorithms like least-square regression, support-vector machines (SVM) or decision trees.

In this paper, we will mainly focus on *embedded methods*. Regarding this category, the literature contains several well-known options for choosing combinations of loss and regularization functions, some of which are exemplarily listed in Table 1.
Different combinations can influence the results drastically: Fig. 2 demonstrates the effect of sparsity by comparing a L_2 and L_1 regularized version. In this example, a proteomics data-set was created that

[2] In feature selection one is interested in identifying relevant dimensions of the data (features) which can be used to distinguish two (or more) classes in a data-set.

[3] We call a vector sparse if the number of non-zero entries is small.

[4] Geometrically, this mean that the vector ω normal to the hyper-plane appropriately separates the data-points of the two classes.

Name	Loss function (L)	Regularizer (R)
AIC/BIC	$\|y_i - \langle\omega, x_i\rangle\|_2$	$\|\omega\|_0$
Lasso	$\|y_i - \langle\omega, x_i\rangle\|_2$	$\|\omega\|_1$
Elastic Net	$\|y_i - \langle\omega, x_i\rangle\|_2$	$\|\omega\|_2^2 + \|\omega\|_1$
Regularized Least Absolute Deviations Regression	$\|y_i - \langle\omega, x_i\rangle\|_1$	$\|\omega\|_1$
Classic SVM	$\max(0, 1 - y_i\langle\omega, x_i\rangle)^*$	$\frac{1}{2}\|\omega\|_2^2$
L1-SVM	$\max(0, 1 - y_i\langle\omega, x_i\rangle)^*$	$\frac{1}{2}\|\omega\|_1$
Logistic Regression	$\log(1 + \exp(-y_i\langle\omega, x_i\rangle))$	$\frac{1}{2}\|\omega\|_1$

*This is the so called *Hinge loss*.

Table 1. Prominent options for choosing loss function and regularizer in feature extraction algorithms. The 1- and 2-norm of a d-vector $z = (z_1, \ldots, z_d)$ are defined by $\|z\|_1 = \sum_{j=1}^{d} |z_i|$ and $\|z\|_2 = (\sum_{j=1}^{d} |z_i|^2)^{1/2}$, respectively. The 0-norm, $\|z\|_0$, simply counts the number of non-zero entries in z.

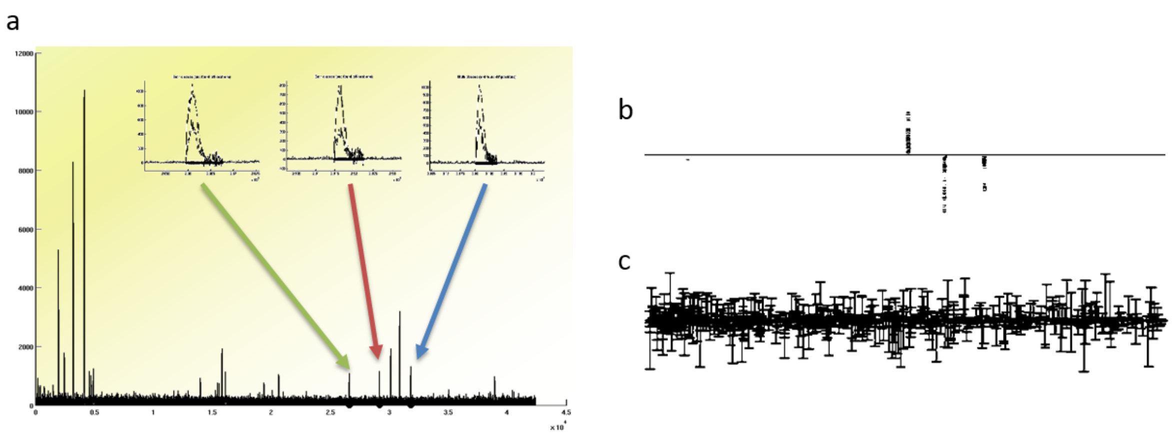

Figure 2. (a) Overlaid spectra from two different groups. The three peaks marked by the arrows (magnified in the inlays) represent the underlying differences between the two groups. (b) Sparse ω found by a L_1 regularized method (L_1-SVM). (c) ω found by L_2 regularized method (classical SVM).

contains three discriminant features between the two sub-groups. It can be easily seen how the results differ: while the L_1 based result is optimized for selecting only few features, the L_2 variant selects much more features which - in combination - results in a better fitted model. In this paper we are interested in developing a method that selects as few features as possible while achieving the best possible fit under this constraint. This is in contrast to methods that aim at only achieving the best possible fit and is of particular interest in biological applications, because each selected feature is usually analyzed in subsequent experiments, thus creating additional cost.

Various ways can be used to evaluate the result, ω, of a feature selection method when appropriate training and test data are available. We will use the following three quality measures: (i) correctness of the selected features, (ii) size of the selected feature set, (iii) performance of classifying an unknown test set. Obviously, (i) can only be used if the correct features are known, which is the case in our benchmark data sets. (For more details see Subsection 4.1.)

1.3 Contribution

The major challenge in sparse feature detection is to robustly identify a *small* number of features (non-zero elements in ω) that can be used to accurately classify unknown proteomics data (e.g. healthy or diseased) by learning from a given training set. This paper introduces *Sparse Proteomics Analysis* (SPA), a novel framework for feature selection and classification. The key step of our method is based on *1-bit compressed sensing* and solves the optimization problem

$$\arg\max_{\omega \in \mathbb{R}^d} \sum_{i=1}^{n} y_i \langle x_i, \omega \rangle \quad \text{subject to } \|\omega\|_1 \leq \sqrt{\lambda} \quad \text{and} \quad \|\omega\|_2 \leq 1, \tag{2}$$

where the regularization is given by the inequality constraints on the feature vector ω. Our approach is motivated by the general theory of *compressed sensing* which was originally introduced in 2005 by Donoho, Candès, Romberg, and Tao [14–16] and efficient algorithms to acquire and process high-dimensional sparse or nearly sparse signals. (For more details see Sections 2 and 3).

We will show the performance of our method by applying it to several synthetic and real-world data-sets and comparing the results to those of other widely used algorithms in this field. Although the core of the algorithm (2) is surprisingly simple, we will show that our method (including the introduced pre- and post-processing steps) finds optimal feature vectors ω that are sparser, allow highly accurate classification, and are more robust against noise than the outcomes of the standard methods listed in Table 1.

Note that standard methods to solve (1) are usually based on solving a convex optimization problem by standard optimization techniques, such as interior point methods. However, these standard methods scale poorly with increasing number of data samples (n) and data dimension (d), as it is the case in the field of -omics data analysis. Several methods have been proposed to speed up the calculations, e.g., by using stochastic approaches ([17–21]). In this article we will not focus on computational complexity but rather on providing a novel way of formalizing and solving the feature selection problem with respect to robustness and precision in the context of compressed sensing.

1.4 Outline of the paper

We start by shortly reviewing the background of *compressed sensing* in Section 2, and then describe our novel feature selection approach in detail (Section 3). We finish with showing benchmark results in Sections 4 and 5 for simulated and real data-sets and compare to current state-of-the-art algorithms.

2 Background: Compressed Sensing

2.1 Compressed Sensing-based Data Analysis

In its most simple form, *compressed sensing* (CS) studies the recovery of a vector $x \in \mathbb{R}^d$ from *linear measurements* $y = Ax$. Here, $A \in \mathbb{R}^{n \times d}$ is an $n \times d$ matrix and the entries of $y \in \mathbb{R}^n$ contain the measurements. The major challenge is now to design the measurement process A in such a way that the number of measurements n is as small as possible and, at the same time, x is still (uniquely) recoverable from y. Thus, we are asking for the maximal *compressibility* of x by linear measurements.

Obviously, when $n \ll d$, we require some additional information to obtain a unique solution of $y = Ax$. The prior information on x which is studied in compressed sensing is the assumption of *sparsity*, i.e., most coefficients of x are assumed to be zero, or at least very small. One naive approach to incorporate this additional property is to search for the sparsest solution of $Az = y$:[5]

$$\operatorname*{argmin}_{z \in \mathbb{R}^d} \|z\|_0 \quad \text{subject to} \quad Az = y. \tag{3}$$

[5] Here, $\|z\|_0 := \#\{i \mid z_i \neq 0\}$ denotes the so-called zero-norm of z, i.e., the number of non-zero elements.

However, this problem is non-convex and cannot be efficiently solved in general. Therefore, one usually replaces (3) by its *convex relaxation*, which is also know as the *basis pursuit* ([22]):

$$\operatorname*{argmin}_{z\in\mathbb{R}^d} \|z\|_1 \quad \text{subject to} \quad Az = y, \tag{4}$$

One of the first key results in compressed sensing states that, if $A \in \mathbb{R}^{n\times d}$ is chosen *randomly*, e.g., with independent and identically distributed Gaussian entries, and $n = O(\lambda \cdot \log(d/\lambda))$, then (with "high probability") every λ-sparse vector x (i.e., $\|x\|_0 \leq \lambda$) can be uniquely recovered by (4). The most surprising fact is that the number of required measurements $n = O(\lambda \cdot \log(d/\lambda))$ is almost of the order of the sparsity level λ. Hence, random measurement processes indeed allow for a very strong compression of sparse vectors (see also [14–16] for more details).

In order to consider more general situations, the stability and robustness of the basis pursuit algorithm was extensively studied. Various results and numerical experiments show that this algorithmic approach can also be applied for the stable recovery of vectors which are only nearly sparse, as well as to noisy measurements of the form $y = Ax + n$. To obtain a robust version of (4), one may replace its equality constraint by $\|Az - y\|_2 \leq \epsilon$ for some appropriate "noise level" $\epsilon > 0$. Not very surprisingly, this approach is also closely related to the LASSO ([23]).

2.2 1-Bit Compressed Sensing

In many practical scenarios, especially when working with computers, there is no way to represent real numbers exactly. Thus, it is reasonable to assume that the measurement vector Ax is acquired in a *quantized* (and therefore non-linear) fashion. The most extreme form directly leads to *1-bit measurements*, i.e., only the signs of Ax are known:[6]

$$y_i = \operatorname{sign}(\langle a_i, x\rangle), \qquad i = 1, \dots, n, \tag{5}$$

where $a_1, \dots, a_n \in \mathbb{R}^d$ are the rows of the measurement matrix $A \in \mathbb{R}^{n\times d}$. As in classical compresses sensing, we are asking for an appropriate recovery of x from (5) using as few measurements as possible. This challenge was originally considered in [24] as *1-bit compressed sensing*, and was extensively studied in [25, 26].

A surprisingly simple convex recovery approach was proposed in [26]:

$$\operatorname*{argmax}_{z\in\mathbb{R}^d} \sum_{i=1}^{n} y_i \langle a_i, z\rangle \quad \text{subject to } \|z\|_1 \leq \sqrt{\lambda} \quad \text{and} \quad \|z\|_2 \leq 1, \tag{6}$$

where $\lambda > 0$ determines a sparsity-controlling parameter. To get some intuition, we should first note that we have $y_i = \operatorname{sign}(\langle a_i, x\rangle)$ if and only if $y_i\langle a_i, x\rangle > 0$ holds. Hence, maximizing the sum in (6) will guarantee the consistency with the measurements for many $i \in \{1, \dots, n\}$. However, an overall consistency is not enforced so that (6) indeed allows noisy inputs y. On the other hand, the constraint of (6) promotes the sparsity of the solution. To see this, consider $S_{d,\lambda} := \{z \in \mathbb{R}^d : \|z\|_0 \leq \lambda, \|z\|_2 \leq 1\}$ and observe that[7]

$$\operatorname{conv}(S_{d,\lambda}) \subset \{z \in \mathbb{R}^d : \|z\|_1 \leq \sqrt{\lambda}, |z|_2 \leq 1\} \subset 2\operatorname{conv}(S_{d,\lambda}).$$

This means that (6) optimizes over a convex relaxation of the set $S_{d,\lambda}$ which contains λ-sparse vector. For more details, see also [25]. The main statement of [26] proves that the robust 1-bit compressed sensing algorithm (6) indeed allows for an appropriate recovery of sparse vectors, using only $n = O(\lambda \cdot \log(d/\lambda))$ measurements, and moreover, that it is very robust to several types of noise such as to random bit-flips in y.

[6]Here, "sign" denotes the sign function, i.e., $\operatorname{sign}(t) = 1$ if $t \geq 0$ and $\operatorname{sign}(t) = -1$ if $t < 0$. Moreover, $\langle\cdot,\cdot\rangle$ denotes the euclidian scalar product.

[7]Here, $\operatorname{conv}(S)$ denotes the convex hull of the set $S \subset \mathbb{R}^d$.

2.3 Why Using Compressed Sensing?

The analysis of compressed sensing usually starts with the assumption that the measurements are independent and identically distributed random variables. Although this setting allows for rigorous proofs and a mathematically sound theory, it is never completely true in practice. Nevertheless, the geometric intuition developed in the theory of compressed sensing allows for much better understanding of the corresponding ℓ_1-based tools from machine learning. For example, the linear program (4) is guaranteed to recover the sparse solutions only under certain constraints on the matrix A, and from the theory of compressed sensing we know what the minimal necessary number of measurements is, and moreover, that the best measurements are those ones which are as uncorrelated as possible. The same applies to the classification problem and 1-bit compressed sensing (6). Although the mathematical assumptions are strictly speaking not fulfilled, it therefore still makes sense to use the corresponding algorithms (4) and (6) also for real-life data.

3 Sparse Proteomics Analysis (SPA)

In this section, we present the details of our novel framework which is based on the idea of 1-bit compressed sensing introduced in the previous section. The first part provides a mathematical formulation of the feature selection problem as well as a brief overview of the steps that are performed in SPA. The rest of this section is then devoted to a detailed description and discussion of the single steps.

3.1 Setting and Overview

As already mentioned in the introduction, we assume that our learning process is *supervised*, i.e., we know which spectrum belongs to the class of healthy ($y_i = +1$) and diseased ($y_i = -1$) samples in advance. If the data vectors $x_i \in \mathbb{R}^d$, $i = 1, \ldots, n$ are (appropriately preprocessed) mass spectra, the indices i of x_i correspond to the m/z-values[8] and its entries represent the intensities. The non-zero entries of the feature vector $\omega \in \mathbb{R}^d$ shall describe the position of the disease fingerprints and its values the significance of these features.

In the setting of classical learning theory, we are asking for a hyperplane $\{\omega\}^\perp$ that correctly separates the data points x_i labeled by y_i. More precisely, this means[9]

$$y_i = \operatorname{sign}(\langle x_i, \omega\rangle), \quad 1 \leq i \leq n. \tag{7}$$

Equivalently, we can view (7) as a problem from 1-bit compressed sensing (cf. Section 2.3), i.e., we have acquired 1-bit measurements and are now looking for a sparse recovery.

In the development of SPA, we have primarily focused on the latter interpretation, and therefore, the 1-bit recovery program (6) forms the key step of our algorithm:

[8] m/z: Mass-over-charge ratio

[9] Compared to Section 2, we are now using the standard notations from learning theory. In particular, the measurement vectors are denoted by x_i (instead of a_i) and the recovered vector is ω (instead of x).

Algorithm 1 (SPA Overview).
Input: Raw data samples $\{x_i, y_i\}_{i=1,\ldots,n}$
Output: Feature vector $\omega \in \mathbb{R}^d$
Preprocessing:

1: Normalize data to make the spectra comparable.

2: Perform smoothing by a convolution with Gaussian density.

3: Center data.

Sparse Feature Selection:

4: Perform 1-bit CS optimization (6) to find feature vector ω.

Postprocessing:

5: Detect the connected components of ω to sparsify even further.

6: Reduce dimension by projecting data onto the feature space.

3.2 Algorithmic Details

In the following, we are going to specify and discuss the single steps of Algorithm 1.

Step 1: Normalization of the data

This step heavily depends on the underlying acquisition method of the data. Every spectrum $x_i \in \mathbb{R}^d$ is normalized by a certain factor $\lambda_i > 0$, i.e., $x_i \mapsto \lambda_i x_i$ for $1 \leq i \leq n$. The individual scalars λ_i should be chosen such that the resulting data vectors are "comparable".

For example, when we assume that the data is acquired by MALDI-TOF-MS as described in Fig. 1, it seems to be quite natural to normalize by the total ion count. Mathematically, this means that we divide every spectrum by its 1-norm, i.e., we choose $\lambda_i = 1/\|x_i\|_1$.

Step 2: Smoothing by Gaussian density

We already pointed out that one major problem is the high noise level of the raw data. Therefore, it is indispensable to perform some noise reduction before trying to extract features. We propose a simple smoothing technique by a Gaussian density:

Let G_σ denote the (centered) *Gaussian density function* with fixed standard deviation $\sigma > 0$, i.e.,

$$G_\sigma(t) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{t^2}{2\sigma^2}\right), \quad t \in \mathbb{R}.$$

The smoothed spectra $\tilde{x}_i \in \mathbb{R}^d$ are then obtained by a discrete convolution

$$(\tilde{x}_i)_k := (x_i * G_\sigma)_k = \sum_{l=1}^{d} (x_i)_l G_\sigma(k-l), \quad k \in \{1, \ldots, d\}, i \in \{1, \ldots, n\}. \tag{8}$$

Using the fast Fourier transform (FFT), this computation can be performed quickly with $O(nd\log(d))$ operations. In a very simplified scenario, a spectrum can be written as the sum of various Gaussian-shaped peaks and some additive noise term. Since the convolution of two Gaussian densities is again Gaussian, the original (local) structure of the spectra is essentially preserved in $\tilde{x}_i$, whereas the noise of x_i is significantly reduced. Note that the deviation $\sigma > 0$ serves as parameter of the algorithm. A good choice of σ clearly depends on the nature of the data; usually it is adapted from the noise level as well as from the (average) width of the peaks.

Finally, we would like to point out another interesting interpretation of the above smoothing technique: The convolution in (8) can be written as a scalar product of x_i with the shifted Gaussian density $G_\sigma(\cdot - k)$ (note that G_σ is symmetric), that is, $(\tilde{x}_i)_k = \langle x_i, G_\sigma(\cdot - k)\rangle$. Thus, the entries of $\tilde{x}_i$ are actually the *analysis coefficients* of the *Gaussian dictionary* $\{G_\sigma(\cdot - k) \mid 1 \leq k \leq d\}$. The perspective of analyzing data by a *dictionary* offers several possibilities of generalization. For instance, one could also consider (redundant) dictionaries with more than one standard deviation or more sophisticated functions than G_σ.

Step 3: Centering the data

The 1-bit optimization in (6) does not incorporate a bias. Hence, it is necessary to center the data first. For this, we compute the *mean spectrum*[10]

$$\bar{x} := \frac{1}{n}\sum_{i=1}^{n} x_i \in \mathbb{R}^d,$$

i.e., $\bar{x}_k$ contains the average of the k-th entry of all spectra. Finally, we obtain the *centered spectra* by subtracting the average:

$$\bar{x}_i := x_i - \bar{x}, \quad 1 \leq i \leq n.$$

Step 4: Sparse Feature Selection

We are now ready to perform the feature detection using the 1-bit recovery method presented in Section 2.2:

Algorithm 2 (1-Bit Compressed Sensing).
Input: Samples $\{x_i, y_i\}_{i=1,\dots,n}$, sparsity level $\lambda > 0$, threshold $\epsilon > 0$
Output: Feature vector $\omega \in \mathbb{R}^d$
Compute:

$$1: \quad \omega' = \operatorname*{arg\,max}_{\tilde{\omega}\in\mathbb{R}^d} \sum_{i=1}^{n} y_i \langle x_i, \tilde{\omega}\rangle \quad \text{subject to } \|\tilde{\omega}\|_1 \leq \sqrt{\lambda} \text{ and } \|\tilde{\omega}\|_2 \leq 1. \tag{9}$$

$$2: \quad \omega_k = \begin{cases} \omega'_k, & |\omega'_k| > \epsilon, \\ 0, & \text{otherwise,} \end{cases} \quad 1 \leq k \leq d. \tag{10}$$

The second computation in (10) is a simple hard thresholding that ensures real sparsity by setting almost zero entries of ω' to 0 (ϵ is usually very small, e.g., $\sim 10^{-3}$).

The actual feature selection takes place in (9). Recalling our measurement model from (7), we observe that the i-th sample is correctly classified by $\tilde{\omega}$ if and only if $y_i\langle x_i, \tilde{\omega}\rangle > 0$. Hence, the optimization functional in (9) will be particularly large when many samples are correctly classified by $\tilde{\omega}$. However, the consistency with *all* measurements, i.e., $y = \operatorname{sign}([\langle x_i, \hat{\omega}\rangle]_{1\leq i\leq n})$, is not enforced, and therefore, it is robust toward (random) bit-flips. This could occur in practice, for instance, when a training sample was wrongly classified. On the other hand, the constraint of (9) guarantees that the solution will be "effectively" sparse (depending on the choice of the sparsity parameter $s > 0$). These observations encourage our intuition that ω will be indeed a sparse vector allowing for an appropriate separation of the two classes.

Step 5: Detecting the connected components

One advantage of Algorithm 2 is that it does not make any assumptions on the structure of x_i; thus, it might be suited for much more general types of data. However, this universality has the drawback that

[10] Actually, we use the smooth data vectors $\tilde{x}_i$ from step 2 for this computation. But in order to keep the notation simple, we still write x_i. This convention holds also for the following steps.

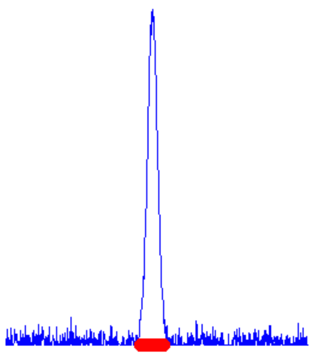

Figure 3. The red stripe indicates the support of ω. Relevant features usually occur as intervals and not as isolated points.

we do not respect the nature of the feature peaks. In fact, a spectrum does not consist of sharp spikes but rather widespread Gaussian shaped peaks. Hence, if the algorithm finds a significant feature position, say in a maximum of some peak, it usually tends to select also features which are close to this position. This phenomenon is not very surprising, because nearby features are highly correlated to the maximum of the peak and therefore, they allow for a good distinction as well.

Empirical results have shown that this process of selection proceeds in a "continuous" fashion when changing the sparsity level λ. As a consequence, the support of a feature vector ω from Algorithm 2 typically consists of only a few connected "intervals" (consecutive sequences of indices) which are centered around the significant peaks (see also Fig. 3). The actual sparsity of ω should therefore be measured in terms of the connected intervals and not by simply counting its non-zero entries.

With this in mind, we may improve the sparsity of ω by reducing every interval to its maximal entry:[11]

Algorithm 3 (Sparsification of ω)**.**
Input: (Sparse) feature vector $\omega \in \mathbb{R}^d$
Output: Sparsified version $\tilde{\omega} \in \mathbb{R}^d$
Compute:

1: Find the connected components $A_1, \ldots, A_N \subset \operatorname{supp}(\omega)$ of ω.

2: For every $1 \leq k \leq N$:
Set all entries of ω in A_k to 0, expect $\arg\max_{l \in A_k} |\omega_l|$.

3: The resulting vector is $\tilde{\omega}$.

Step 6: Dimension reduction

This final step does not perform any further computations but shows how to proceed with our result ω. As mentioned earlier, the main purpose of SPA is not just to classify (unknown) samples, but rather to reduce the data to its significant entries (dimensions). Indeed, we may use ω for a *dimension reduction*: Let $x \in \mathbb{R}^d$ be some (possibly unknown) sample vector. Then, we can project x onto the selected feature positions of $\operatorname{supp}(\omega)$. More precisely, all entries that do not belong to $\operatorname{supp}(\omega)$ are set to 0:

$$(\hat{x})_k := \begin{cases} (x)_k, & k \in \operatorname{supp}(\omega), \\ 0, & \text{otherwise}, \end{cases} \qquad k \in \{1, \ldots, d\}. \tag{11}$$

[11] Here $\operatorname{supp}(\omega) = \{i \mid \omega_i \neq 0\}$ denotes the support of ω, i.e., the set of indices corresponding to the non-zero entries.

The resulting vector $\hat{x} \in \mathbb{R}^d$ is now trivially embedded into a low-dimensional space of dimension $\#\operatorname{supp}(\omega)$;[12] but it still contains the most important information that has been found by the above algorithm. Note that we have not made any use of the actual values of ω but merely of its support.

By this projection, we reduce the danger of overfitting, and working in a low-dimensional space, a large tool set from *machine learning* is now available for classification and clustering. But how to explicitly proceed with the data heavily depends on the specific application and is therefore not part of SPA.

[12] In practice, one would simply discard all indices that are not contained in $\operatorname{supp}(\omega)$.

4 Numerical Experiments

In this section we evaluate our method based on the performance as (1) a feature selection method and (2) a learning algorithm using artificial and typical real-world proteomics mass-spectrometry datasets. We will also compare our results to the widely used state-of-the-art algorithms LIBLINEAR (L1-based SVM), LASSO (using the standard Matlab implementation) and Elastic Net (using again the standard Matlab implementation where the trade-off parameter between L1 and L2 regularization was set to 0.5).

4.1 Evaluation Criteria

The results of the evaluation will be measured with three ideas in mind: (a) the selected feature-set should contain only correct features (which are known by designing the input data), (b) the selected feature-set should be as small as possible, and (c) a resulting classification model should explain the data as good as possible.

Feature Selection Performance To test the feature selection performance, we measure the algorithm's ability to select an optimal feature set. Specifically, on the ability to select features related to the spiked peptides. Because the correct features (true positives) are known in our data set, we will use a measure based on $precision = \frac{TP}{TP+FP}$[13].

Model-Learning Performance Based on the selected features, a classification algorithm (model) was trained and its performance evaluated using a three-fold cross-validation schema. We will use a standard *logistic regression model* implemented by a generalized linear model (GLM) [27] with a canonical logit link and the results will be measured in terms of the *Bayes information criterion* (BIC) defined as $BIC = -2 \cdot \text{log-likelihood}^{14} + k \cdot \log(n)$, where k is the number of parameters (features) and n is the number of data-points. Taking the model complexity into account is crucial, since the number of selected features has a strong impact on interpretability of the final model and also determines the scale of subsequent experiments for the actual biomarker identification (e.g. identification of selected proteins and their validation as biomarkers).

4.2 Data-sets

To test, benchmark, and compare our method we will use two different data-sets with increasing complexity: (1) TOY-GROUP: to explain basic properties of our algorithm and (2) SIM-GROUP: based on real data but with added noise and artificial peptide peaks. In Section 5 we will present results of real data analysis.

[13] TP: true positives, FP: false positives

[14] Likelihood is typically defined as: $p(data \mid parameters)$ or $L(parameters \mid data)$

Dataset 1: TOY-GROUP

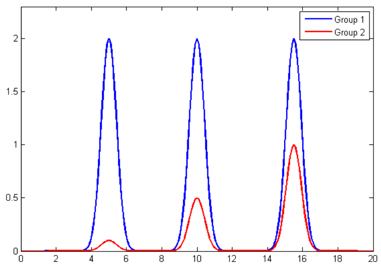

Figure 4. Simple three peak example. Shown is data for two classes, indicated as red and blue curves.

A toy example which will serve as input data is shown in Fig. 4. The aim of the algorithm is to identify a minimal set F of dimensions which - in combination - characterize the difference between the two classes best. In this toy example the dimensions would be: $F = \{5, 10, 15\}$.[15].

Dataset 2: SIM-GROUP

The data used for our simulation studies was generated using our own simulation program. Each simulated data-set consists of two sub-groups H and D (e.g. two groups of healthy and diseased spectra) having n spectra each. Except noise, the two sub-groups only differ by m artificially "spiked in" peptide peaks which are only present in the D group at known positions, see Fig. 5 for an example. The algorithm to create the simulated data is described in Alg. 4. It outputs the simulated raw MS signals. All configuration parameters were hold fix, except the signal variability, i.e. the height of the spiked-in peaks.

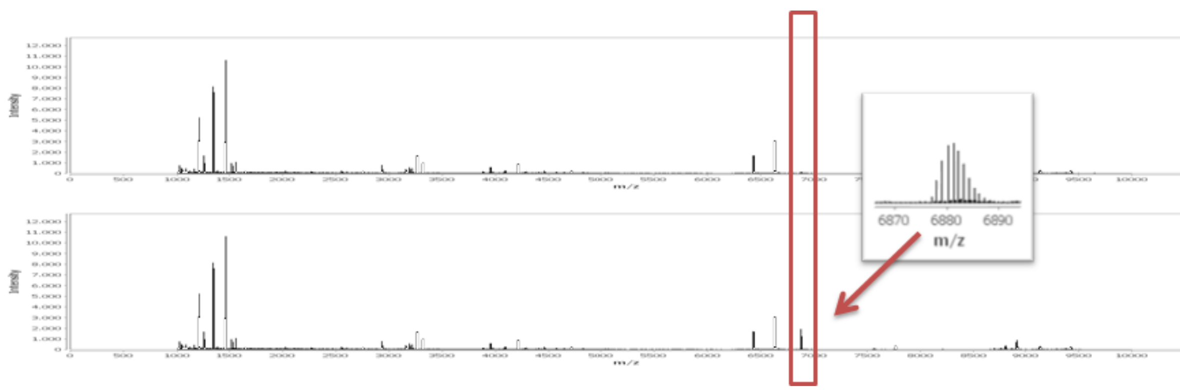

Figure 5. This figure shows two mass spectra which differ by $m = 1$ artificially spiked-in peptide peak at m/z 6880. The inlet shows the calculated isotope distribution for the used peptide.

[15] $g(x) = \exp(-((x-p)/(0.601 * w))^2) * h$ which creates a Gaussian peak centered at position $p = \{5, 10, 15\}$, half-width $w = 1$ and heights $h_1 = \{0.2, 0.5, 1\}$ and $h_2 = \{2, 2, 2\}$ for group 1 and 2, respectively.

Algorithm 4 (Create Simulated Data-set).
Input: $S_0 \in \mathbb{R}^d$: Base spectrum from a real experiment, n: group size, σ variance of added noise, $P = \{p_i, h_i\}_{i=1..m}$: set of m peptide amino-acid sequences which represent the spiked-in peaks and respective peak heights
Output: Two groups H, D of simulated spectra

Compute $H = S_1, \ldots, S_n$**:**

1: Derive n noisy versions of S_0 by adding normally distributed noise with given variance σ to each entry: $S_i(k) = S_0(k) + N(0, \sigma)$ $(i = 1 \ldots n)$

Compute $G = S_{n+1}, \ldots, S_{2n}$**:**

2: Compute, $S_{ID} \in \mathbb{R}^d$, the sum of all isotope distributions (spikes) of all peptides p_i with their respective heights h_i $(i = 1 \ldots m)$

3: Computed spiked version of S_0: $S_{0,spiked} = S_0 + S_{ID}$

4: Derive n noisy versions from $S_{0,spiked}$: $S_i(k) = S_{0,spiked}(k) + N(0, \sigma)$ $(i = (n+1) \ldots 2n)$.

To allow systematic tests of the algorithm's output with respects to different types of input data we simulated four sub-groups:

1. SIM-GROUP-BASE: The D sub-group of this data-set contains $m = 3$ true-positive peaks with centers at m/z positions: 5559 Da, 6191 Da and 6883 Da[16]. The respective peak heights are $h_1 = 150$, $h_2 = 200$ and $h_3 = 250$.
2. SIM-GROUP-NOISE-Y (NY): Based on SIM-GROUP-BASE increasing noise was added to the y-values, to simulate measurement errors.
3. SIM-GROUP-NOISE-X (NX): Based on SIM-GROUP-BASE increasing noise was added to the peak centers (x-values) to simulate non-linear shifts between measurement errors. Specifically, we determined the average peak width ($pw = 75$) and used that to calculate a shift $S = round(N(0, (x \cdot pw)^2)), x = \{0.05, 0.1, 0.2, 0.3, 0.5, 0.75\}$ for each spectrum. If $S > 0$, we added S 0-entries in the beginning of the spectra and deleted the last S entries. If $S < 0$, we deleted the first S entry and added 0-entries at the end. The range for true positive was then defined as: $[x_l + S_{min}, x_r + S_{max}]$, where x_l and x_r are the known interval borders of added true-positive peaks and S_{min}, S_{max} are the minimum and maximum shifts, respectively.
4. SIM-GROUP-PEAKS-SIMILAR (PS): In this dataset, an increasing number of (true positive) peaks of same height was added, to see how it affects the sparsity of the found solution.

4.3 Influence of the algorithmic components

In this section we demonstrate the basic properties of our method using the toy example introduced in the previous section. Recall that the method has three components that can influence the found solution: (1) usage of the dictionary in the pre-processing step, (2) usage of sparsification in the post-processing step and (3) the choice of λ in the 1-bit CS step (see Eq. 2).

In the following, we will show the influence of each component using the toy example introduced in the previous section.

[16] These peaks correspond to peptide sequences: p_1 : LKKVVALYDYMPMNANDLQLRKGDEYFILEESNLPWWRARDKNGQE, p_2 : LKKVVALYDYMPMNANDLQLRKGDEYFILEESNLPWWRARDKNGQEGYIPSN, p_3 : LKKVVALYDYMPMNANDLQLRKGDEYFILEESNLPWWRARDKNGQEGYIPSNYVTEAE

Influence of λ

Fig. 6 shows the influence of λ on the number of selected features. As one would expect from (2), the number of features increases with increasing λ (Fig. 6a), until saturation is reached at $\lambda = 28$. Fig. 6b shows the trace-plot of the selected features with respect to lambda. One can clearly see that the algorithm first (λ small) selects the features of the more discriminating peak centered at position $x = 5$ before it starts (with increasing λ) to select features from the second peak centered at $x = 10$.

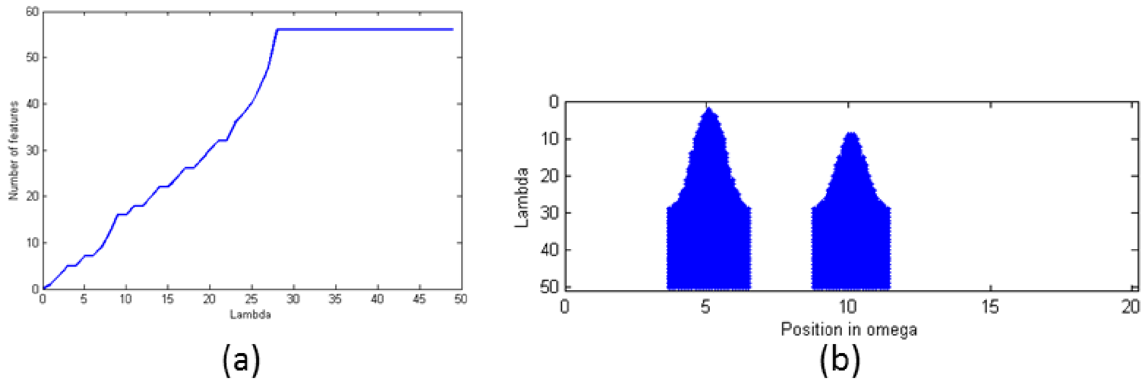

Figure 6. Influence of λ on the "Toy Example" data-set without using pre-processing (dictionary) and post-processing (sparsification). (a) Number of non-zeros entries of ω with respect to λ. (b) Order and position of selected features with respect to λ. The blue area indicates the position of a selected features, i.e. the index of the non-zero entries in ω.

Influence of Post-Processing (sparsification)

Fig. 7 shows the influence of the sparsification step on the number of ω's non-zeros components. As expected, the algorithm only selects the maximum element in a connected range of possible features. Since in this (noise and error-free) toy example the peaks are connected, only the very center features are selected which results in a saturation of two features, as can be seen in the trace-plot (Fig. 7b).

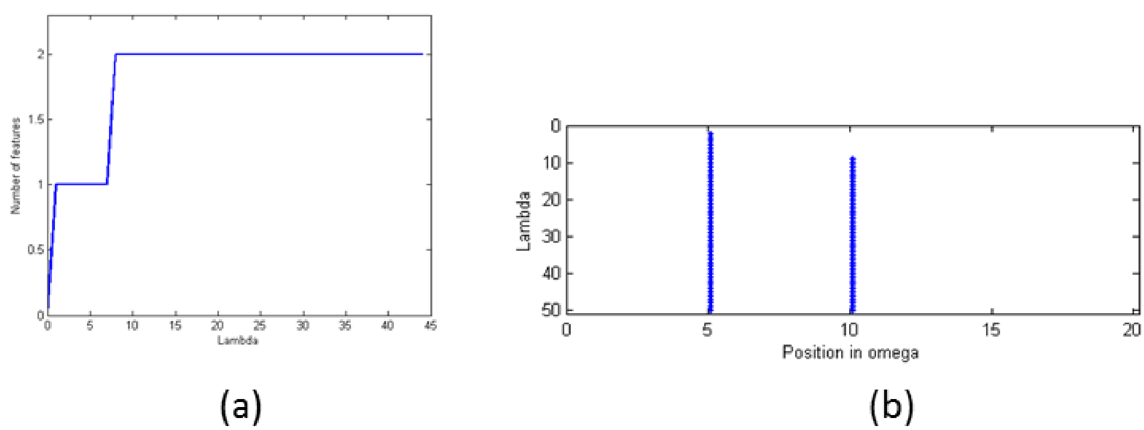

Figure 7. Influence of λ on the "Toy Example" data-set using post-processing (sparsification) but without using pre-processing (dictionary). (a) Influence of λ on number of non-zeros entries of ω. (b) Trace-plot for selected features.

Influence of Pre-Processing (Gaussian Dictionary)

Fig. 8 shows the influence of the dictionary step on the number of ω's non-zeros components. The effect of the convolution of the data with Gaussians leads to a smoothed signal which, by this, can become broader than the original data. This is illustrated in the trace-plot (Fig. 8b): the selected features span a wider range compared to the case where no smoothing was applied in the pre-processing (Fig. 6b).

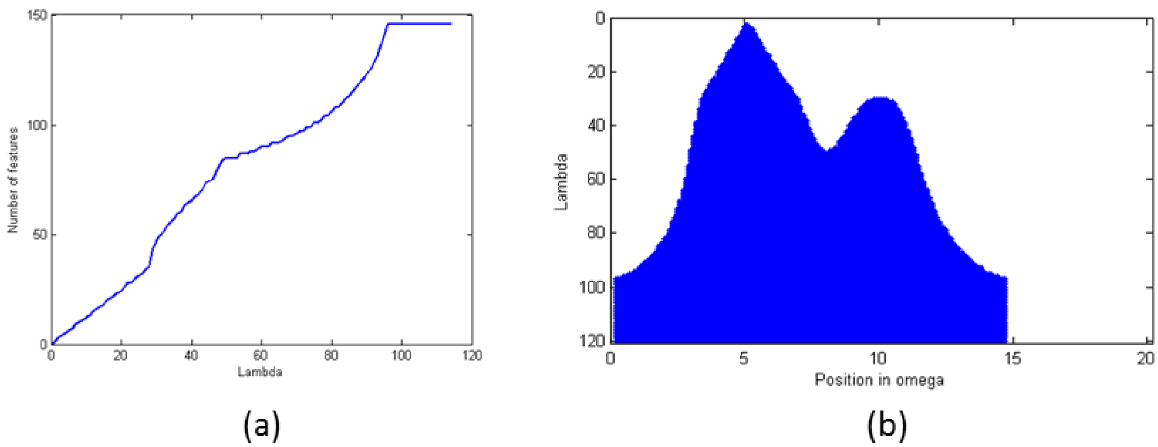

Figure 8. Influence of λ on the "Toy Example" data-set using pre-processing (dictionary) but without using post-processing (sparsification). (a) Influence of λ on number of non-zeros entries of ω. (b) Influence of λ on the "Toy Example" data-set using pre-processing (dictionary) but without using post-processing (sparsification).

Influence of Pre- and Post-Processing

The remaining configuration of the algorithm using pre- and post-processing shows an interesting behavior, with respect to the number of selected features. With increasing λ the algorithm first selects one feature, then two and then jumps back to one again. This effect can be explained when looking at the trace-plot for the order of the feature selection (Fig. 9b): the smoothing of the pre-processing has lead to a broadening of the signal. At a particular value of λ the two broadened peaks merge which create a now connected range of selected features. This fact is then picked up by the sparsification post-processing step which selects only the maximum value in the now connected range, resulting in only one selected feature.

4.4 Robustness of Feature Selection in the Presence of Noise

In the previous section we showed the influence of the algorithmic components when working with "perfect" data that does not contain any noise. We demonstrated that the smoothing pre-processing and the sparsifying post-processing steps reduced the number of selected features significantly. Both steps together lead to the sparsest model even when λ was increased. In this section we will use noisy data - based on the previously used toy data-set - and explain the influence on the results of the feature selection. The noisy data-set consists of 100 spectra for each class as shown in Fig. 10b. The noisy spectra are obtained by using the original toy-example data (Fig. 10a) and then - for each point in the new spectrum - adding Gaussian noise with mean 0 and standard deviation 0.3. Additional to the evaluation of the selected features we also evaluate our method as described in Section 4.1. Thus, we are not only interested in a small feature set but also in the performance of a *generalize linear model* (GLM) that was trained using the selected features in a 3-fold cross-validation schema.

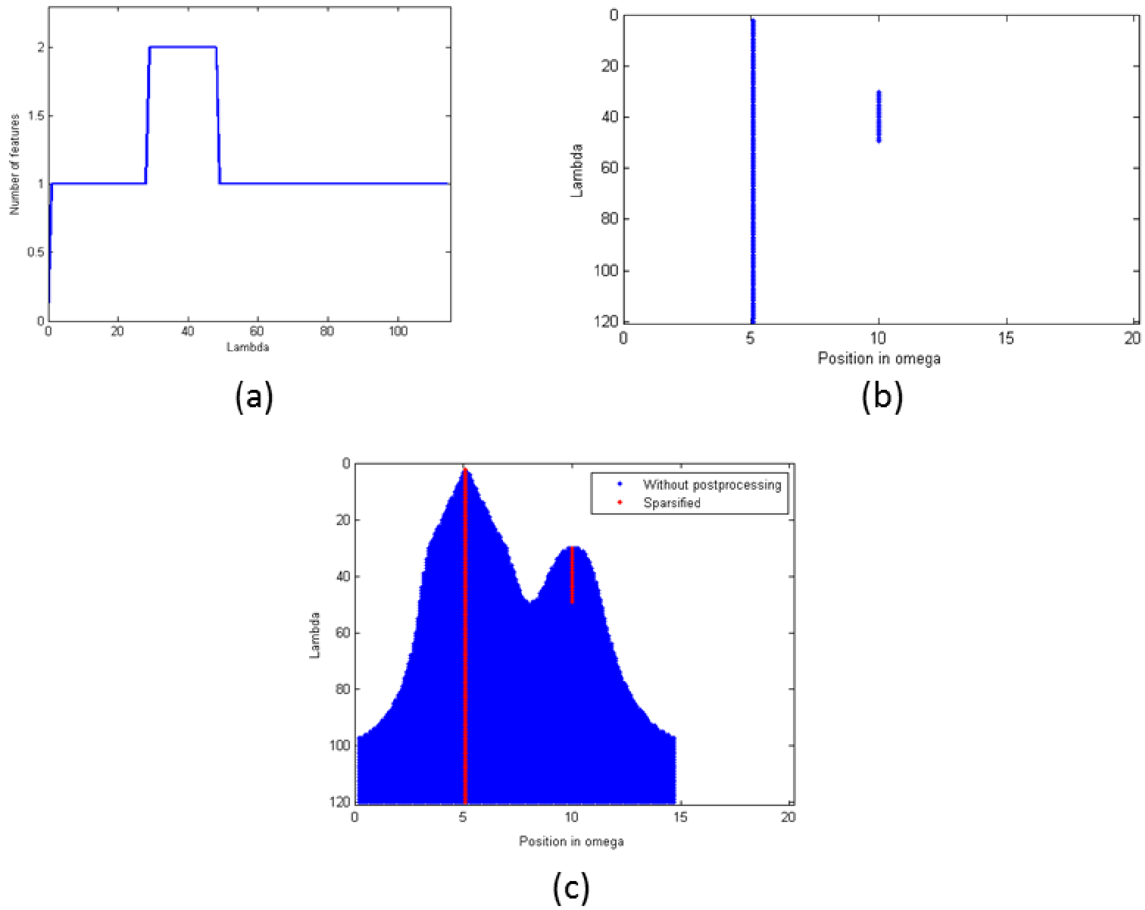

Figure 9. Influence of λ on the "Toy Example" data-set using pre-processing (dictionary) and post-processing (sparsification). (a) Influence of λ on number of non-zeros entries of ω. (b) Trace-plot for selected features. (c) Trace-plot for selected features. The blue area shows - as before - the order of the selected peaks with respect to lambda. The vertical red lines indicate the selected features in the case where sparsification was used. It can be seen that only the feature with the maximum value is selected when the regions of selected features connect.

Influence of λ on Feature Selection and Classification

As in the previous section we will demonstrate the influence of the algorithmic components λ, pre- and post-processing on the results. Again, we are not only interested in the number of selected features, but also the performance of a GLM classifier that was trained with the selected classifiers. The results are shown in Fig. 11. As in the unperturbed case from the previous section, the combination of pre- and post-processing again yields the best results: a very sparse feature vector ω with only one component which can perfectly (AUC=1)[17] classify unknown samples (using a 3-fold cross validation schema). Especially the smoothing effect of the pre-processing step has a large impact on the number of selected features, as shown in Fig. 12: using the pre-processing step leads to a significantly sparser feature vector ω.

[17] The quality or accuracy of a classifier is often measured by the area under the ROC curve (AUC). An area of 1 represents a perfect classifier.

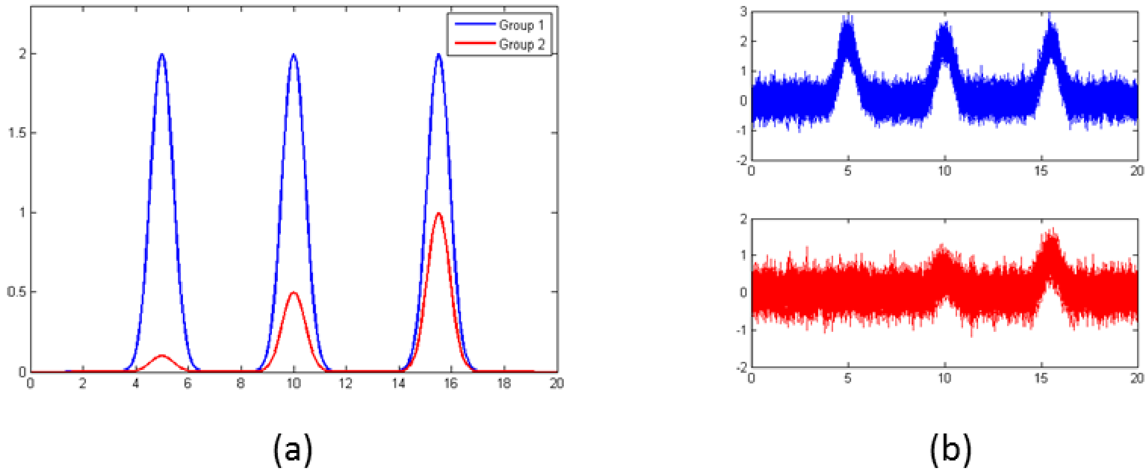

Figure 10. Comparison of the unperturbed (left) and the noisy (right) toy-example data-set. Shown is data for two classes, indicated as red and blue curves.

4.5 Comparison to Other Methods

In the previous sections we have evaluated our method by means of (i) number of features and (ii) performance of a GLM model for classification that was trained only using the selected features. We have shown that our method is capable to find a small and well suited feature-set that allow the robust classification of unknown samples even in the presence of noise, using a simple toy example. In this section we will evaluate our method using more complex data-sets (see Subsection 4.2) and also compare to results obtained by other widely used algorithms that perform the same task as our method: selecting small numbers of features from complex data-sets for classifying unknown samples. For the comparison we have selected three algorithms which are widely used in (but not limited to) the area of bioinformatics, such as analysis of large Microarray, genomics or proteomics data-sets. We will compare (1) SPA, (2) L2-loss L1-regularized SVM (using the LIBLINEAR package), (3) the LASSO approach (using the standard MATLAB implementation), and (4) the Elastic Net method (also using the standard MATLAB implementation with $\alpha = 0.5$). We have chosen these algorithms because they seem to have become standard analysis algorithms which are used in thousands of publications. We will use these algorithms in the standard configuration as given by the respective implementation or in the original publication. Unless otherwise stated, 3-fold cross validation has been used for all experiments. The remaining of this section described results of our feature selection experiments. The main goal is to evaluate which and how many features are selected by the respective algorithms and how a GLM training with these features performs in classifying unknown samples. In particular we are interested in how increasing noise and varying number and quality of features in the given data-sets influences the outcome, measured by the criteria defined in Sec. 4.1. Table 2 summarizes the results and was created using the following steps:

1. The used data-set was divided into $k = 3$ folds.
2. For each of the methods (SPA, LASSO, L1-SVM, Elastic Net) the optimal parameter(s) was determined resulting in the best possible feature-set.
3. Using the optimal parameter(s) from the previous step, k rounds of feature selection were performed resulting in three sets of feature-sets.
4. Each of the feature-sets was used to train a GLM on two folds.
5. Performance of each GLM was computed by classifying the respective third fold.

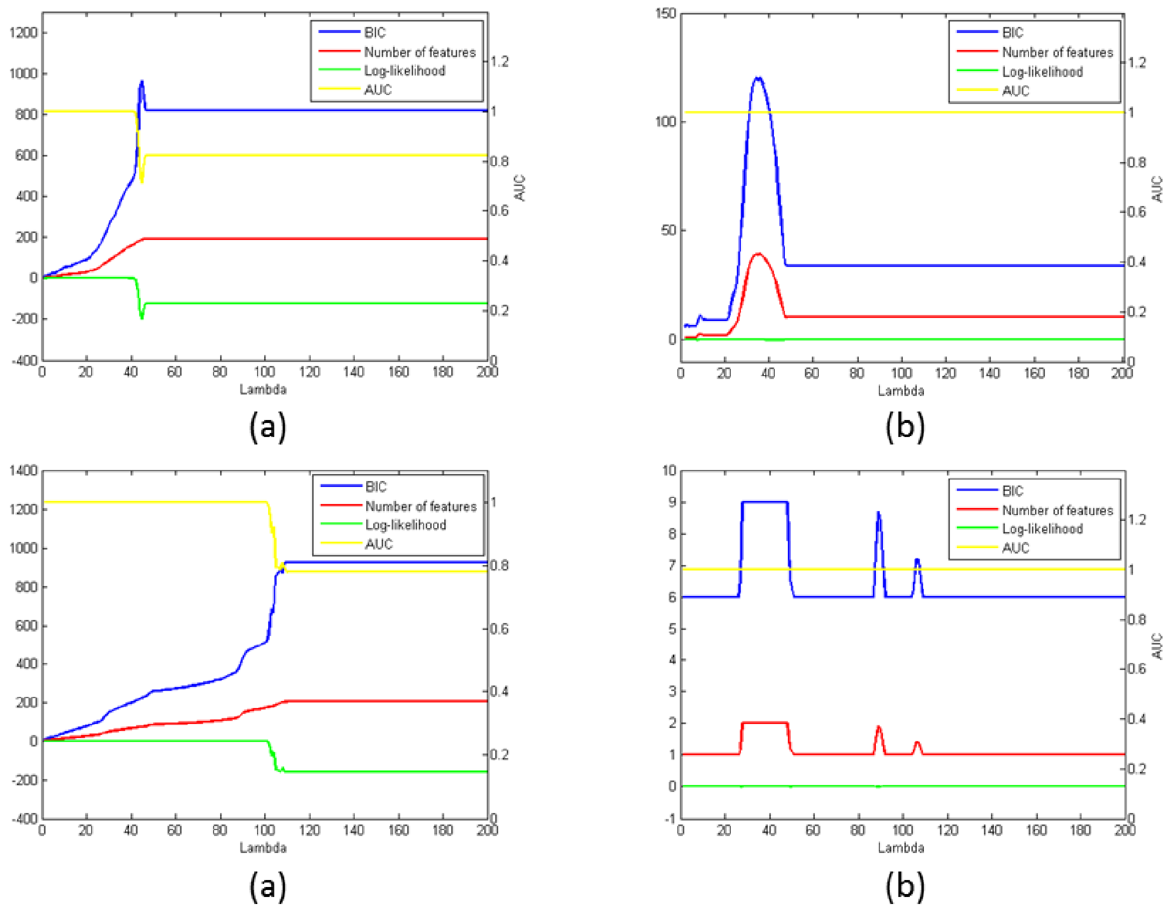

Figure 11. Shown is the performance of our method on the "Noisy toy example" data-set with respect to λ and using the pre- or post-processing. The evaluation is done with respect to number of features (red line), Bayes Information Criterium (BIC, blue line) and log-likelihood (green line) of the trained model and the area under the ROC curve (AUC, yellow line). The algorithmic configuration was: (a) No pre- or post-processing, (b) post-processing (sparsification) only, (c) pre-processing only (Gaussian dictionary), (d) pre- and post-processing.

6. Mean values of number of non-zero features, BIC and precision (or AUC in the case of real data) were computed and are given in the table.

Test Scenarios

We used three test scenarios to evaluate and compare the methods (see also Subsection 4.2): (i) added noise to the peak heights (*SG-NY*Δ), (ii) added noise to the peak positions (*SG-NX*Δ, occurs when the mass spectrometer was not calibrated correctly) and (iii) addition of correlated peaks (*SG-PS*Δ). The Δ values (see first column in Table 2) represent the respective perturbation intensity.

Summary of Results

We evaluate the results with respect to the following categories:

- **Precision**: SPA reaches the highest precision values over all data-sets, except for data-set *SG-NY1*. This means that the SPA almost always only selects true positive features and not others,

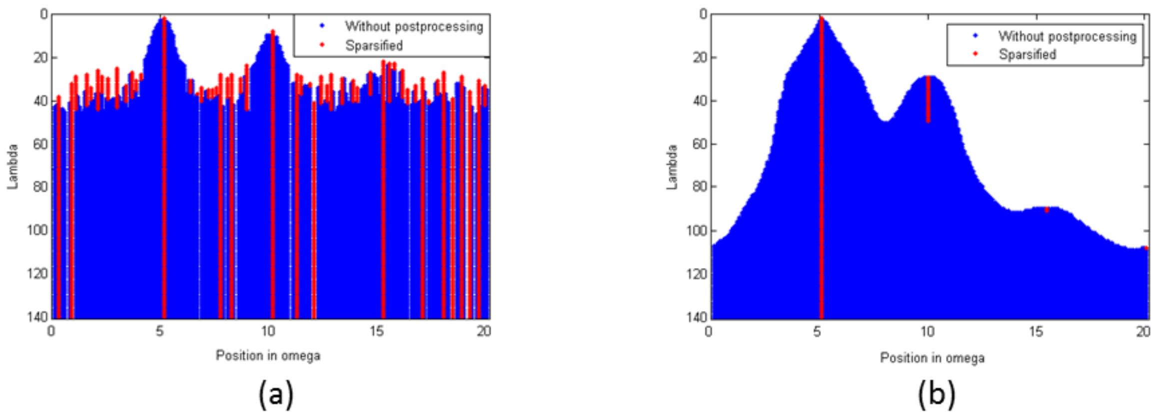

Figure 12. Effect of the pre-processing step on the number of selected features. Shown is the trace-plot for the "Noisy toy example" data-set. The blue area shows the order of selection and distribution of features (non-zero entries of ω) with increasing λ. The vertical red lines indicate the effect of the post-processing (sparsification) step than only selects the feature of maximum value in a range of connected features. (a) Without pre-processing. (b) With pre-processing.

which are potentially introduced by noise.

- **Size of feature set**: SPA always selects the smallest number of features, except for data-set *SG-NY0.5*.
- **BIC**: SPA performs best with respect to the *Bayesian Information Criterion* when (i) peak positions are perturbed on the m/z (y) axis (*SG-NY*) and (ii) when correlated peaks are added (*SG-PS*). In the case of noise added to the signal (*SG-NX*) the L1-SVM algorithm produces better results.

Dataset	SPA				LASSO				Elastic Net				L1-SVM			
	Prec.[a]	Feat.[b]	BIC[c]	λ[d]	Prec.	Feat.	BIC	λ	Prec.	Feat.	BIC	λ	Prec.	Feat.	BIC	λ
SG-BASE	1.000	1 ±0	22.25	1	1.000	37.67 ±3.06	204.48	0.020	1.000	44.67 ±2.08	241.50	0.030	0.557	6 ±1	37.02	$6.12 \cdot 10^{-5}$
SG-NY0.25	1.000	2 ±0	21.66	45	0.189	76 ±6.25	407.20	0.019	0.171	90.33 ±3.79	483.00	0.042	0.500	2 ±0	15.86	$2.83 \cdot 10^{-5}$
SG-NY0.5	0.233	4.33 ±0.58	52.77	305	0.000	79.33 ±7.23	424.82	0.026	0.003	118.33 ±3.51	631.07	0.060	0.000	2 ±0	15.86	$2.85 \cdot 10^{-5}$
SG-NY0.75	0.667	1.67 ±0.58	33.45	133	0.000	72 ±4.36	386.04	0.029	0.004	158.33 ±24.91	842.60	0.058	0.000	2 ±0	15.86	$1.85 \cdot 10^{-5}$
SG-NY1	0.000	2 ±0	63.36	151	0.005	69 ±8.66	370.18	0.029	0.007	196.33 ±22.55	1,043.55	0.058	0.500	2 ±0	15.86	$1.45 \cdot 10^{-5}$
SG-PS1	1.000	1 ±0	10.58	1	1.000	23 ±2.65	126.92	0.014	1.000	26 ±1	142.78	0.033	0.500	2 ±0	15.86	$1.73 \cdot 10^{-5}$
SG-PS2	1.000	1 ±0	10.58	1	1.000	37.33 ±4.51	202.72	0.009	1.000	42 ±4.58	227.40	0.023	0.500	2 ±0	15.86	$1.71 \cdot 10^{-5}$
SG-PS3	1.000	1 ±0	10.58	1	1.000	36.37 ±3.79	199.19	0.018	1.000	52.67 ±3.22	283.80	0.036	0.444	2.33 ±0.58	17.63	$1.75 \cdot 10^{-5}$
SG-NX0.05	1.000	2.67 ±0.58	40.52	315	0.567	92 ±6.08	491.81	0.007	0.779	59 ±3.46	317.30	0.013	0.446	45.33 ±8.39	252.16	0.004
SG-NX0.1	1.000	1.67 ±0.58	44.29	67	0.832	65.33 ±4.51	350.80	0.016	0.949	52 ±1	304.31	0.025	0.184	355.67 ±29.28	2,873.05	0.894
SG-NX0.2	0.670	1 ±0	75.84	1	0.661	92 ±3.46	554.12	0.013	0.952	30.33 ±3.21	238.45	0.070	0.164	615 ±19.52	4,576.06	1.403
SG-NX0.3	0.670	1 ±0	90.18	1	0.554	102.33 ±2.52	592.80	0.013	0.580	104 ±7	619.65	0.013	0.205	419 ±31.51	3,494.87	0.228

Table 2. This table shows the main results comparing the feature selection benchmarks of SPA, LASSO, Elastic-Net and L1-SVM. For explanation see text.
[a] **Prec.**: Precision, defined as: $\frac{TP}{TP+FP}$. (The higher the better.); [b] **Feat.**: Number of features; ± indicates the standard-deviation. (The lower the better.); [c] **BIC**: Bayesian Information Criterium, defined as: $-2 \cdot logLikelihood + k \cdot \log(n)$, where k is the number of parameters (features) and n is the number of data-points. (The lower the better.); [d] λ: the algorithm dependent parameter.

5 Analyzing Experimental MALDI-TOF MS Data

In this section we present results of our algorithm for analysing data from real world mass spectrometry experiments. We demonstrate the performance on two previously published data-sets ([10, 28]):

- *Spiked Data*: Blood serum of 16 apparently healthy individuals was used ([28]) which was spiked with 127nMol/L of the protein calibration standard mix (Part No.: 206355 & 206196) from Bruker Daltronics (Leipzig, Germany). This mix contains the hormones Angiotensin, ACTH, clip 18-39, Substance P and the cell protein Ubiquitin. The peptide mix was added before sample pre-treatment and 64 spectra were measured due to 4-fold spotting (technical replicates).
- *Pancreas Cancer Data (P. CA)*: A total of 120 patients with pancreatic cancer and controls were recruited for this study [10]. For the discovery study sera were obtained from two different clinical centers (University Hospital Leipzig (UHL, set L) and Heidelberg (UHH, set H)).

The only preprocessing step that has been performed on the raw mass spectrometry data was baseline removal, by using TopHat filtering ([29]). In particular, no calibration or noise reduction steps have been applied. More information the data and sample preparation can be found in the supplementary material (see Appendix 6).

Summary of Results

As before, we evaluate the results with respect to the following categories. Note that since we do not know the true-positive feature positions, we omitted this criterion and used the *Area under the ROC curve* (AUC) instead. The results are shown in Table 3.

- **Size of feature set**: SPA always selects significantly less numbers of features.
- **AUC**: SPA has the best AUC values for the *Spiked* and *P. CA - UHH* data-set, while the LASSO and Elastic Net algorithms perform slightly better on the *P. CA - UHL*.
- **BIC**: SPA produces models that have significantly better BIC values.

Taken together, one can see that SPA is much more robust to noise occurring in real world data-sets than the other tested algorithms, which respect to model quality measured by the BIC criterion. Further, the resulting feature-set is much smaller which allows an easier interpretation and less costly follow-up experiments.

	SPA				LASSO				Elastic Net				L1-SVM			
Dataset	Feat.[a]	AUC[b]	BIC[c]	λ[d]	Feat.	AUC	BIC	λ	Feat.	AUC	BIC	λ	Feat.	AUC	BIC	λ
Spiked	1 $\pm$0	1.00	17.49	1	29.67 $\pm$3.51	1.00	139.97	0.029	46.66 $\pm$6.43	1.00	216.05	0.121	83 $\pm$5.29	0.749	927.42	0.068
P. CA - UHL	3.33 $\pm$0.58	0.92	50.24	251	66.33 $\pm$7.77	0.99	397.15	0.031	79 $\pm$1.73	0.99	474.47	0.082	243 $\pm$15.10	0.569	2,677.69	0.120
P. CA - UHH	1.33 $\pm$0.58	0.96	40.76	115	8.67 $\pm$0.58	0.93	314.50	0.154	59.67 $\pm$4.93	0.93	615.01	0.176	662.67 $\pm$71.44	0.825	3,927.10	1.207

Table 3. This table shows the main results comparing the feature selection benchmarks of our approach with LASSO, Elastic-Net and L1-SVM. For explanation see text.

[a] **Feat.**: Number of features; $\pm$ indicates the standard-deviation. (The lower the better.);
[b] **AUC**: Area under the ROC curve. (The higher the better.);
[c] **BIC**: Bayesian Information Criterium, defined as: $-2 \cdot logLikelihood + k \cdot \log(n)$, where k is the number of parameters (features) and n is the number of data-points. (The lower the better.); [d] λ: the algorithm dependent parameter.

6 Conclusion

We presented SPA, a new framework for the analysis of proteomics data, generated by mass spectrometry experiments. The framework solves the problem of selecting a minimum set of features from high-dimensional data in the case where relatively few measurements are available, to (1) allow bio-medical interpretation and (2) enable classification of unknown samples. This is done formulating and solving a regularized optimization problem, using ideas from 1-bit compressed sensing combined with matching pre- and post-processing extensions. We showed that SPA performs better than standard (and widely used) algorithms for analysing proteomics data and that is is robust to random and systematic noise. The evaluation was done using simulated and real-world data, and the resulting classifiers are better than those of the other algorithms with respect to the Bayesian information criterion and precision (ratio of true positives to number of positive (true and false) test outcomes).

Supporting Information

S1 - Mass Spectrometry Data Generation

Chemicals, standards and consumables

Gradient grade acetonitrile, ethanol, and HPLC-water were obtained from J.T. Baker (Phillipsburg, NJ, USA); p.a. trifluoroacetic acid (TFA) and acetone were purchased from Sigma-Aldrich (Taufkirchen, Germany). The peptide- and protein MALDI-TOF calibration standards I and α-cyano-4-hydroxycinnamic acid (HCCA) were purchased from Bruker Daltonics (Bremen, Germany). Automated magnetic bead preparations were performed using 96 well plates, TubePlates from Biozym (Hessisch Oldendorf, Germany), polypropylene tubes (low profile) from Abgene (Surrey, UK), and modular reservoir quarter modules from Beckman (Fullerton, USA). For sample storage 450 μL CryoTubesTM were purchased from Sarstedt (Nümbrecht, Germany). Multifly needle sets and polypropylene serum monovettes with clotting activators were also obtained from Sarstedt.

Peptidome Separation

All serum samples of the discovery set were processed at one time and analyzed simultaneously to avoid procedure-dependent errors. The external validation set was prepared, processed and analyzed separately. Peptidome separation of the samples was performed using the ClinPro Tools profiling purification kits from Bruker Daltonics. Magnetic particles with defined surface functionalities (magnetic beadimmobilized metal ion affinity chromatography (MB-IMAC Cu), magnetic bead-hydrophobic interaction (MB-HIC C8) and weak cation exchange (MB-WCX)) were processed by the ClinPro Tools liquid handling robot according to the manufacturer's protocol (Bruker Daltonics). Serum specimens were thawed on ice for 30 min and immediately processed according to our standardized protocol for serum peptidomics [30].

Mass Spectrometry

A linear MALDI-TOF mass spectrometer (Autoflex I, Bruker Daltonics) was used for the peptidome profiling. Daily mass calibration was performed using the standard calibration mixture of peptides and proteins in a mass range of 1-10 kDa. Mass spectra were recorded and processed using AutoXecute tool of the flexControl acquisition software (Ver. 2.0; Bruker Daltonics). The settings were applied as follows: Ion source 1: 20 kV; ion source 2, 18.50 kV; lens, 9.00 kV; pulsed ion extraction, 120 ns; nitrogen-pressure, 2500 mbar. Ionization was achieved by a nitrogen laser (λ=337 nm) operating at 50 Hz. For matrix suppression a high gating factor with signal suppression up to 500 Da was used. Mass spectra were detected in linear positive mode.

MS Data Preprocessing

We only performed baseline removal. The baseline is an exponential like offset dependent on the m/z value (mass-to-charge; x-value). A baseline correction is performed to remove this rather low-frequency noise from the spectrum. We use a morphological TopHat filter to eliminate certain spatial structures within the signal, in our case the baseline. Note that this technique does not produce negative intensity values.

Acknowledgments

JV was supported by the ERC CZ grant LL1203 of the Czech Ministry of Education. TC, MG, NC, JV, GK and CS were supported by the Einstein Center for Mathematics Berlin (ECMath), project grant CH2, and by the DFG Research Center Matheon *Mathematics for key technologies*, Berlin. TC and CS are supported by the German Ministry of Research and Education (BMBF) project Grant 3FO18501 (Forschungscampus MODAL). GK acknowledges support by the Einstein Foundation Berlin, by the

Deutsche Forschungsgemeinschaft (DFG), and by the DFG Collaborative Research Center TRR 109 *Discretization in Geometry and Dynamics*.

The authors are thankful to Irena Bojarovska for fruitful discussions and help conducting the experiments.

References

1. Aebersold R, Mann M. Mass spectrometry-based proteomics. Nature. 2003;422(6928):198–207.

2. Petricoin EF, Belluco C, Araujo RP, Liotta LA. The blood peptidome: a higher dimension of information content for cancer biomarker discovery. Nat Rev Cancer. 2006 Dec;6(12):961–967. Available from: `http://dx.doi.org/10.1038/nrc2011`.

3. Rai AJ, Chan DW. Cancer proteomics: Serum diagnostics for tumor marker discovery. Ann N Y Acad Sci. 2004 Jun;1022:286–294. Available from: `http://dx.doi.org/10.1196/annals.1318.044`.

4. Coombes KR, Morris JS, Hu J, Edmonson SR, Baggerly KA. Serum proteomics profiling–a young technology begins to mature. Nat Biotechnol. 2005 Mar;23(3):291–292. Available from: `http://dx.doi.org/10.1038/nbt0305-291`.

5. Liotta LA. Clinical proteomics: written in blood. Nature. 2003;425.

6. Phizicky E, Bastiaens PIH, Zhu H, Snyder M, , Fields S. Protein analysis on a proteomic scale. Nature. 2003;422.

7. Issaq HJ, Xiao Z, Veenstra TD. Serum and plasma proteomics. Chem Rev. 2007 Aug;107(8):3601–3620. Available from: `http://dx.doi.org/10.1021/cr068287r`.

8. Stühler K, Meyer HE. MALDI: more than peptide mass fingerprints. Curr Opin Mol Ther. 2004 Jun;6(3):239–248.

9. Sitek B, Waldera-Lupa DM, Poschmann G, Meyer HE, Stühler K. Application of label-free proteomics for differential analysis of lung carcinoma cell line A549. Methods Mol Biol. 2012;893:241–248. Available from: `http://dx.doi.org/10.1007/978-1-61779-885-6_16`.

10. Fiedler GM, Leichtle A, Kase J, Baumann S, Ceglarek U, Felix K, et al. Serum Peptidome Profiling Revealed Platelet Factor 4 as a Potential Discriminating Peptide Associated With Pancreatic Cancer. Clinical Cancer Research. 2009 June;15(11):3812–3819. Available from: `http://publications.mi.fu-berlin.de/155/`.

11. Strenziok R, Hinz S, Wolf C, Conrad TOF, Krause H, Miller K, et al. Surface-enhanced laser desorption/ionization time-of-flight mass spectrometry: serum protein profiling in seminoma patients. World J of Urology. 2009 June;28(2):193–197. Available from: `http://publications.mi.fu-berlin.de/156/`.

12. Leichtle A, Nuoffer JM, Ceglarek U, Kase J, Conrad TOF, Witzigmann H, et al. Serum amino acid profiles and their alterations in colorectal cancer. Metabolomics. 2011 September;Available from: `http://publications.mi.fu-berlin.de/1081/`.

13. Diao L, Clarke CH, Coombes KR, Hamilton SR, Roth J, Mao L, et al. Reproducibility of SELDI Spectra Across Time and Laboratories. Cancer Inform. 2011;10:45–64. Available from: `http://dx.doi.org/10.4137/CIN.S6438`.

14. Donoho DL. Compressed sensing. IEEE Trans Inform Theory. 2006;52:1289–1306.

15. Candés EJ, Tao T. Decoding by linear programming. IEEE Trans Inform Theory. 2005;51:4203–4215.

16. Candés EJ, Romberg J, Tao T. Stable signal recovery from incomplete and inaccurate measurements. Comm Pure Appl Math. 2006;59:1207–1223.

17. Genkin A, Lewis D, Madigan D. Largescale Bayesian logistic regression for text categorization. Technometrics. 2007;49:291–304.

18. Friedman J, Hastie T, Tibshirani R. Regularized paths for generalized linear models via coordinate descent. Department of Statistics, Stanford University; 2008.

19. Efron B, Hastie T, Johnstone I, Tibshirani R. Least angle regression. Ann Statist. 2004;32:407–499.

20. Koh K, Kim S, Boyd S. An Interior-Point Method for Large-Scale l1-Regularized Least Squares. Selected Topics in Signal Processing. 2007;1(4):606–617. Available from: `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4407767`.

21. Wu TT, Lange K. Coordinate descent algorithms for lasso penalized regression. Annals of Applied Statistics. 2008;2:224–244.

22. Chen SS, Donoho DL, Saunders MA. Atomic decomposition by basis pursuit. SIAM J Sci Comput. 1998;20:33–61.

23. Tibshirani R. Regression shrinkage and selection via the Lasso. J Royal Statist Soc B. 1996;58:267–288.

24. Boufounos PT, , Baraniuk RG. 1-Bit compressive sensing. In: 42nd Annual Conference on Information Sciences and Systems (CISS); 2008. .

25. Plan Y, Vershynin R. One-bit compressed sensing by linear programming. Communications on Pure and Applied Mathematics. 2013;66:1275–1297.

26. Plan Y, Vershynin R. Robust 1-bit compressed sensing and sparse logistic regression: a convex programming approach. IEEE Transactions on Information Theory. 2013;59:482–494.

27. Nelder J, Wedderburn R. Generalized Linear Models. Journal of the Royal Statistical Society. 1972;135.

28. Kratzsch J, Fiedler GM, Leichtle A, Brügel M, Buchbinder S, Otto L, et al. New reference intervals for thyrotropin and thyroid hormones based on National Academy of Clinical Biochemistry criteria and regular ultrasonography of the thyroid. Clin Chem. 2005 Aug;51(8):1480–1486. Available from: `http://dx.doi.org/10.1373/clinchem.2004.047399`.

29. Sauve AC, Speed TP. Normalization, baseline correction and alignment of high-throughput mass spectrometry data. In: Proceedings of the Data Procedings Gensips; 2004. .

30. Baumann S, Ceglarek U, Fiedler GM, Lembcke J, Leichtle A, Thiery J. Standardized approach to proteome profiling of human serum based on magnetic bead separation and matrix-assisted laser desorption/ionization time-of-flight mass spectrometry. Clin Chem. 2005 Jun;51(6):973–80.