Published in final edited form as:
J Proteome Res. 2010 October 1; 9(10): 5346–5357. doi:10.1021/pr100594k.

Efficient marginalization to compute protein posterior probabilities from shotgun mass spectrometry data

Oliver Serang,
Department of Genome Sciences, University of Washington, Seattle, WA, USA

Michael J. MacCoss, and
Department of Genome Sciences, University of Washington, Seattle, WA, USA

William Stafford Noble
Department of Genome Sciences, Department of Computer Science and Engineering, University of Washington, Seattle, WA, USA

Oliver Serang: orserang@uw.edu; Michael J. MacCoss: maccoss@uw.edu; William Stafford Noble: william-noble@uw.edu

Abstract

The problem of identifying proteins from a shotgun proteomics experiment has not been definitively solved. Identifying the proteins in a sample requires ranking them, ideally with interpretable scores. In particular, “degenerate” peptides, which map to multiple proteins, have made such a ranking difficult to compute. The problem of computing posterior probabilities for the proteins, which can be interpreted as confidence in a protein’s presence, has been especially daunting. Previous approaches have either ignored the peptide degeneracy problem completely, addressed it by computing a heuristic set of proteins or heuristic posterior probabilities, or by estimating the posterior probabilities with sampling methods. We present a probabilistic model for protein identification in tandem mass spectrometry that recognizes peptide degeneracy. We then introduce graph-transforming algorithms that facilitate efficient computation of protein probabilities, even for large data sets. We evaluate our identification procedure on five different well-characterized data sets and demonstrate our ability to efficiently compute high-quality protein posteriors.

1 Introduction

Tandem mass spectrometry is an increasingly useful tool for identifying proteins in complex mixtures, particularly as an agent for hypothesis generation. However, large data sets, which benefit from the advantages of mass spectrometry, commonly lead to many spurious matches between peptides and spectra. These errors substantially decrease the specificity of predictions made by existing algorithms. For example, we have observed that ProteinProphet [Nesvizhskii et al., 2003] may assign to a protein a very high posterior probability even when that protein only contains a single peptide with a good match to an observed MS/MS spectrum or when a protein contains several peptides with very poor matches to MS/MS spectra. These inflated proteins often receive scores tying or exceeding proteins containing many high-scoring PSMs. When larger proteins or many spectra are considered, the number of these incorrectly matched spectra grows, and the set of proteins suggested by ProteinProphet becomes untrustworthy.

Attempts to compute statistically rigorous protein probabilities, with models derived using relatively few, well-defined assumptions, are hampered by the problem of peptide degeneracy, which arises when a single peptide maps to multiple proteins. In Figure 1, the peptides EEAMPFK and VNILLGLPK are degenerate: EEAMPFK maps to two identically connected proteins, and VNILLGLPK maps to two differently connected proteins. Degenerate peptides are responsible for the apparent intractability of computing protein probabilities, because the

posterior probability of one protein depends on the presence of other proteins when a peptide maps to both proteins.

Existing approaches to protein identification solve the peptide degeneracy problem in quite different ways. Several heuristic methods [Weatherly et al., 2005, Qian et al., 2005, Reiter et al., 2009] use the distribution of decoy peptide scores to estimate protein false discovery rates (FDRs), while ignoring the differences in peptide degeneracy between the target and decoy databases. ProteinProphet, which is perhaps the most widely used procedure for solving the protein identification problem, employs an iterative heuristic probability model to estimate posterior protein probabilities. The similar EBP method [Price et al., 2007] takes a more sophisticated approach for amalgamating replicate experiments, extending the ProteinProphet method and using more explicitly stated assumptions. Both of these methods use a similar expectation-maximization framework and perform very well in practice. However, both methods are described procedurally rather than derived from assumptions, so the resulting models are difficult to understand and improve upon.

In contrast, the nested model of Li et al. [2009], is derived clearly from well-described assumptions but ignores the peptide degeneracy problem entirely. The result is that a protein may be reported as present even though it is only associated high-scoring peptide is explained by a second protein, which contains many more high-scoring peptides. PANORAMICS [Feng et al., 2007] uses a numeric heuristic to jointly compute peptide probabilities from protein probabilities and compute protein probabilities from peptide probabilities and efficiently solves the resulting simultaneous equation in an iterative manner. Both of these models perform well, but neither offers a substantial improvement in sensitivity when using a high threshold for protein probabilities. Despite their shortcomings, these methods illustrate how sensitive ProteinProphet can be when allowing very few false positive protein identifications.

IDPicker [Zhang et al., 2007, Ma et al., 2009] adopts a conservative, non-probabilistic approach, thresholding peptides into “present” and “absent” sets and then using a greedy algorithm to solve (or approximate a solution to) the resulting NP-hard minimum set cover problem. The peptide-level threshold must be set quite strictly, because every present peptide may permit identification of an associated protein. Furthermore, this approach does not distinguish between a protein associated with a single moderately-scoring peptide identification and a protein with several moderately-scoring peptide identifications. Although non-probabilistic approaches are often easy to understand, they are in principle not as informative as probabilistic methods, which can assemble large quantities of weak evidence and use the evidence to estimate the probability that a given protein is present.

Recently, Li et al. [2008] developed perhaps the most rigorous existing treatment of the peptide degeneracy problem. They use a Bayesian framework and a sampling procedure for estimating protein probabilities, and their system slightly out-performs ProteinProphet on a small data set. Unfortunately, their sampling approach requires training a complex model of peptide detectability involving hundreds of parameters. Furtheremore, it has not been demonstrated that the detectability model can be trained on one data set and then employed effectively for protein identification on a different data set; detectability is widely recognized to fluctuate between even similar experiments, and small errors in peptide detectability may result in a large differences to the set of proteins identified. Finally, the running time of the sampling method on this small data set is substantially longer than the running time of ProteinProphet. Sampling methods are known to converge to exact posteriors, but in an infinite time, and the actual time necessary for an acceptable approximation may be too great to be of utility on larger data sets.

In this paper we introduce a novel Bayesian method for computing posterior protein probabilities. Our approach is motivated by our desire to derive a model using a few relatively simple assumptions, but also to create accompanying algorithms that make computation very efficient. Such a model will allow us to evaluate the assumptions and systematically make improvements in a manner that is difficult with many current approaches. Our model uses only three parameters, which can be easily estimated using the same data set used for identifying proteins. With respect to the peptide deneracy problem, our model rewards protein sets that contain independent evidence in addition to degenerate peptides. In particular, the model allows a protein with strong independent supporting evidence to "explain away" supporting data that is shared with other proteins. Thus, our method automatically apportions information from degenerate peptides during the marginalization procedure, rather than requiring an *ad hoc* adjustment.

We then describe a series of three mathematical transformations, which substantially increase the computational efficiency of computing posterior probabilities, while still recognizing peptide degeneracy. The resulting algorithm is mathematically equivalent to the result achieved by *marginalizing*, the process of computing every possible set of present proteins and evaluating their net contribution. In contrast to sampling, marginalizing yields an exact, closed-form solution in a finite amount of time. Naively marginalizing would require enumerating every possible set of proteins, which is exponentially complex and hence impossible even for small problems, but our optimized marginalization procedure is significantly more efficient and computes the same result as the naive approach. Using our method, it is possible to compute discriminative and interpretable posterior probabilities quickly, even on large data sets. This combination of efficiency and rigor allows us to compute accurate and well-calibrated posterior probabilities quickly, and lays the groundwork for more complex models and more optimized procedures.

2 Materials and Methods

2.1 Data sets

We have compared our method to ProteinProphet [Nesvizhskii et al., 2003] on four data sets, yeast lysate [Käll et al., 2007], *H. influenzae* lysate [Nesvizhskii et al., 2003], the ISB 18 mix protein standard [Klimek et al., 2008], and *C*. elegans lysate [Hoopmann et al., 2009], and compared to MSBayes [Li et al., 2008] on one additional data set, the Sigma Aldrich 49 protein standard [Zhang et al., 2007]. Each collection of spectra was searched against a combined database of *target* and *decoy* proteins. For the purposes of the analyses, when a protein identification method identifies a protein from the database, that identification is considered a "true positive" or a "false positive," depending on whether the protein is a target or a decoy, respectively. It should be noted that treating the targets as true positives is not perfectly correct, because the target database is actually a mixture of true and false positives. Consequently, our true positive counts may be slightly inflated; however, because we are only using this estimate of the true positives as a relative comparison between methods, the slight bias introduced will not influence the comparison. Summary statistics describing each data set are given in Table 1.

H. influenzae—*H. influenzae* lysate was digested with trypsin and analyzed by LC-MS/MS on an ESI-ITMS machine. The spectra were searched with SEQUEST [Eng et al., 1994] against a database containing *H. influenzae* (targets) and human proteins (decoys). The resulting PSMs were scored using PeptideProphet with a minimum peptide probability of 0.0.

Yeast—*Saccharomyces cerevisiae* strain S288C were grown to mid log phase on rich media at 30°C. The proteins were digested with trypsin and analyzed using data dependent acquisition

and LC-MS/MS, using an LTQ machine. The resulting spectra were searched using Crux [Park et al., 2008] against a database consisting of all yeast ORFs plus a shuffled version of each ORF. PSMs were assigned probabilities by PeptideProphet using a minimum peptide probability of 0.05.

ISB 18 mix—The ISB 18 protein data was created using proteins purchased from Sigma Aldrich. This data set consisted of four prepared samples, which were analyzed on a variety of mass spectrometry machines with several technical replicates. The proteins were digested together using trypsin, and in one of the four samples, digestion was aided by sonication. These peptides were analyzed using LC-MS/MS on a variety of machines, including LTQ, LCQ Deca, Q-TOF, QSTAR, AGILENT XCT Ultra, Applied Biosystems ABI 4800, AppliedBiosystems 4700, and Thermofinnigan LTQ-FT. The spectra obtained from each experiment were searched using SEQUEST against a database containing these 18 proteins, a set of closely related homologs (obtained from the authors), which are indistinguishable from or may have been purified with the 18 proteins, possible contaminants, and a collection of *H. influenzae* proteins. PSMs were assigned probabilities by using PeptideProphet with a minimum peptide probability of 0.05. The contaminant proteins were not treated as present or absent, because they were identified using the Trans Proteomic Pipeline (http://tools.proteomecenter.org/wiki/index.php?title=Software:TPP), which includes the ProteinProphet algorithm. We analyzed the replicate experiments in two ways: individually, and after pooling all experiments together.

C. elegans—*C. elegans* were grown to various developmental stages on peptone plates containing *E*. coli. After removal from the plate, bacterial contamination was removed by sucrose floating. The lysate was sonicated and digested with trypsin and subject to six technical replicate LC-MS/MS analyses using LTQ machine and data dependent acquisition. The spectra were searched against a database containing the target proteins, the *C. elegans* proteome and known contaminants, as well as a reversed copy of every target protein. PeptideProphet was run using a minimum PSM probability of 0.05.

Sigma 49 mix—The Sigma 49 mixture was prepared using 49 human proteins from Sigma Aldrich. The proteins were digested with trypsin and subjected to three replicate LC-MS/MS analyses using a Thermo LTQ machine. The spectra were searched using MyriMatch16 [Tabb et al., 2007] against a database composed of all Swiss-Prot (54.2) proteins with the _HUMAN as well as a reversed copy of each protein. During the database search, any spectra that matched multiple peptide sequences and that also received equal scores for these matches were excluded. The remaining PSMs were scored using PeptideProphet and any PSMs with probability less than 0.05 were thrown out.

3 Results

3.1 The protein identification problem

In tandem mass spectrometry, a complex protein mixture is first digested with a restriction enzyme to create a population of peptides. These peptides are separated by hydrophobicity using liquid chromatography (LC) and subsequently separated by their mass-to-charge ratios (m/z) using the mass spectrometer. This procedure isolates a population of peptides with a common hydrophobicity and precursor m/z. Ideally, this population of peptides is homogeneous. Each such population is then fragmented, and the fragments are subjected to a second scan with the mass spectrometer to recover the m/z values of the fragments. Together, these fragments produce an MS/MS spectrum. In a typical shotgun proteomics experiment, this process is performed several thousand times, producing many MS/MS spectra and their associated precursor m/z values. Protein identification is the task of ranking the proteins by

the evidence that they are in the sample, given these fragmentation mass spectra. Ideally, each ranked protein also is assigned a score with a well-defined semantics, specifying our confidence in the assertion that the protein was present in the sample.

Typically, the protein identification process consists of two stages. In the first stage, the observed spectra and precursor m/z values are matched to peptides by a database search tool (reviewed in [Nesvizhskii et al., 2007]). During this stage, probability scores may be computed for each match using a tool such as PeptideProphet [Keller et al., 2002]. In the second stage, the proteins are scored and ranked using the scored peptides. We will focus on the second stage.

The process by which proteins create spectra, and the resulting problem of identifying proteins from these spectra, can be represented using graphs. After the first stage, the proteins, peptides, and spectra can be represented by a tripartite graph (Figure 1A). Following previous methods [Nesvizhskii et al., 2003], we collapse the tripartite graph into a bipartite graph by keeping only the edges connecting the best peptide match for each spectrum and the edges connecting the best spectrum match for each peptide. Once each spectrum associates with only one peptide and each peptide associates with only one spectrum, each of these pairs can be merged together to form a layer of "peptide spectrum matches" (PSMs), which are weighted by the quality of the match between the paired peptide and spectrum (Figure 1B). Like ProteinProphet and unlike MSBayes, we currently do not consider peptides that have not matched a spectrum. This bipartite graph serves as the input to the second stage, in which we rank the proteins according to the estimated probability that they are present in the sample.

3.2 A probability model for scoring candidate solutions

To compute the desired protein posterior probabilities, we model the tandem mass spectrometry process using a Bayesian probability model. Our model follows directly from a series of seven simple assumptions, which are illustrated in Figure 2 and described in detail below. First, however, we introduce some terminology. We say a peptide was *emitted* by a protein if that peptide was created by digesting a protein, retrieved by the precursor scan, and analyzed by the fragmentation scan. We say that a PSM is created by the *noise model* if the peptide was identified by the fragmentation scan but the scan was not derived from that peptide.

The seven assumptions underlying our model are as follows:

1. **Conditional independence of peptides given proteins.** The process by which one peptide is retrieved from the precursor scan does not influence the retrieval of other peptides from the precursor scan given the set of proteins in the sample.
2. **Conditional independence of spectra given peptides.** The process by which a spectrum is created and observed does not influence the creation and observation of other spectra given the set of peptides selected by the precursor scan.
3. **Emission of a peptide associated with a present protein.** We model the probability with which a sample peptide is generated from a protein containing it with a constant probability α. This event is independent of all other emission events. Although the probability that a peptide is retrieved may depend on properties of the peptide, the model can account for these variations by adjusting the probability of the PSM. Adjusting the probability of a PSM is equivalent to adjusting the probability that the peptide was retrieved from the precursor scan. These events are only observable in conjunction, so it is not possible to distinguish between the event where a peptide is retrieved from the precursor scan but its spectrum is mistakenly assigned from the event where a peptide is not retrieved from the precursor scan and undergoes no fragmentation scan.

4. **Creation of a peptide from noise.** We model the probability that a truly absent peptide (i.e. not created by an associated protein) is erroneously observed with the constant probability β.
5. **Prior belief a protein is present in the sample.** We model our prior belief that any protein is present in the sample with probability γ. It would be possible to later introduce a more complex prior, but doing so may increase the runtime of the algorithm.
6. **Independence of prior belief between proteins.** The prior probabilities of all proteins are independent.
7. **Dependence of a spectrum only on the best-matching peptide.** Each spectrum depends exclusively on the peptide that it best matches and is paired with to form a PSM.

We consider the validity of these assumptions in more detail in the Discussion section.

From this probability model, we are able to compute the likelihood of a particular set of proteins given the the observed set of spectra, which is proportional to the probability that these proteins would create the observed spectra:

$$L(R=r|D) \propto \Pr(D|R=r) \tag{1}$$

$$= \sum_{\forall e} \prod_{\varepsilon} \Pr(D_\varepsilon|E_\varepsilon=e_\varepsilon) \Pr(E_\varepsilon=e_\varepsilon|R=r) \tag{2}$$

$$= \sum_{\forall e_1} \sum_{\forall e:e_1} \prod_{\varepsilon} \Pr(D_\varepsilon|E_\varepsilon=e_\varepsilon) \Pr(E_\varepsilon=e_\varepsilon|R=r) \tag{3}$$

$$= \sum_{\forall e_1} \Pr(D_\varepsilon|E_1=e_1) \Pr(E_1=e_1|R=r) \sum_{\forall e:e_1} \prod_{\varepsilon \neq 1} \Pr(D_\varepsilon|E_\varepsilon=e_\varepsilon) \Pr(E_\varepsilon=e_\varepsilon|R=r) \tag{4}$$

$$= \prod_{\varepsilon} \sum_{\forall e_\varepsilon} \Pr(D_\varepsilon|E_\varepsilon=e_\varepsilon) \Pr(E_\varepsilon=e_\varepsilon|R=r) \tag{5}$$

where R is the set of present proteins, E is the set of present peptides, D represents the observed spectra, and ε is used to index the peptides. Both R and E are random variables representing the truly present protein and peptide sets; r and e are specific values taken on by these random variables. Equation (1) removes uncertainty from the unknown peptide set E by marginalizing over all possible peptide sets (i.e., all possible values E can take on, denoted $\forall e$). For example, if the set of spectra match 10,000 distinct peptides, then the enumeration over all possible values of e must consider $2^{10,000}$ possibilities.

We compute values proportional to $\Pr(D_\varepsilon|E_\varepsilon = e_\varepsilon)$ using PeptideProphet and $\Pr(E_\varepsilon = e_\varepsilon|R = r)$ using our model of peptide generation. This former is actually computed by an intermediate step in the PeptideProphet algorithm and can be recovered by applying Bayes' rule to

PeptideProphet's probability scores and prior probability estimates; we show this procedure in detail in the supplement. The conditional independence of peptides given proteins allows us to compute the sum over all peptide sets in linear time (rather than exponential time), by transforming the sum into an equivalent product over peptide indices. Essentially, the procedure between Equations (2) and (4) can be repeated on the right sum in Equation (4) using a different peptide index. This operation can be continued inductively on each successive sum, effectively unrolling the sum of products into a product of sums. In the product of sums form, each sum has only two states (a particular peptide is present or absent), so each term in the product is trivial, permitting the likelihood of a set of proteins to be computed in linear time relative to the number of peptides. From a graphical model perspective, once the set of proteins is specified, all of the PSMs are disconnected from each other, making an independent graph for each PSM. The likelihood is thus computable by a product over these independent graphs. A more complete derivation, as well as other derivations used for our model and optimizations, are provided in the supplement.

3.3 Computing posterior probabilities for each protein

By applying Bayes rule to the likelihood proportional to $Pr(D|R)$ and marginalizing over the set of proteins, we can compute a posterior probability for each protein. This approach appears to be prohibitively expensive, because a naive implementation of this marginalization requires explicitly enumerating every possible set of proteins (a so-called "power set"). The computational cost of enumerating this power set is exponential in the number of proteins, making the naive implementation impractical for even small data sets. However, in practice, we do not need to enumerate the power set of peptides, because we have assumed conditional independence of peptides given proteins because, when a protein set is specified, our assumptions allow us to marginalize over all peptide sets using a single product.

In order to make computation of our posterior probabilities computationally feasible for large data sets, we introduce three graph-transforming procedures: partitioning, clustering, and pruning. These procedures, illustrated in Figure 3, dramatically increase the efficiency of computing posterior probabilities for the proteins.

3.3.1 Speedup #1: Partitioning—In our model, a protein is dependent on other proteins within connected subgraphs, but not dependent on proteins that share no peptides with proteins in the connected subgraph. We exploit this property to compute posterior probabilities for proteins in a subgraph by enumerating over the power set of proteins in the subgraph. We accomplish this by partitioning the original graph into connected subgraphs. When a specific digest, such as trypsin, is used, this transformation considerably decreases the number of protein sets that need to be evaluated.

3.3.2 Speedup #2: Clustering—We prove (see supplement) that in our probability model, proteins with identical connectivity can be clustered together to compute their posterior probabilities with greater efficiency. In Figure 3, proteins 1 and 2 are indistinguishable; therefore, the case in which protein 1 is present and protein 2 is absent has the same probability as the case in which protein 1 is absent and protein 2 is present. Thus, these two proteins can be merged into a single node (Figure 3B), which can occupy three distinct states: a state with both proteins absent, a state with only a single protein present, and a state with both proteins present. The state where a single protein is present must now be counted twice because there are two ways for it to occur. Using this transformation, we enumerate the power set in three steps rather than four. Generally, merging n proteins reduces the number of states that must be enumerated from 2^n to $n + 1$.

3.3.3 Speedup #3: Pruning—We also prove (see supplement) that within a connected subgraph, any two partitions of proteins that are only connected by peptides with a probability of zero can be transformed into two subgraphs that do not connect to one another. These zero-probability peptides are often produced by PeptideProphet when the best spectrum match for the peptide is a very poor match. Because they are a special case, each zero-probability peptide implies two necessary events: first, the peptide cannot be emitted by any protein, and second, the peptide cannot be created by the noise model; for if the peptide were emitted by a protein or created by the noise model, it would necessarily raise its probability above zero, resulting in a contradiction. When two protein partitions within a subgraph are connected only through zero-probability peptides, then neither partition may emit any of those zero-scoring peptides.

The pruning operation copies each zero-probability peptide so that each of these protein partitions connects to its own copy; therefore, these necessary events remain the same, except the event that the peptides are not created from noise is now counted twice instead of once, because a copy has been added. We correct for this overcounting, transforming the original problem into two partitioned subproblems.

In Figure 3C, proteins 4 and 5 are only connected by a zero-probability peptide. The only possible events that would produce the observed data require that neither protein 4 nor 5 emit the peptide and require that the peptide not be created by the noise model. Creating a copy of the peptide for each of these proteins and then correcting so that the noise model is only counted once will produce the same posterior probability for each protein.

Table 2 illustrates the effects of these three speed-ups on five different data sets. The first three rows of the table indicate the size of the input graph, the next four rows list the size of the corresponding search space initially and after each of the three graph transformations, and the remaining row shows the runtime of the algorithm. In the most extreme case, *H. influenzae*, the graph transformations reduce the size of the search space by nearly 10,000 orders of magnitude. By reducing the theoretical complexity of the procedure, these graph optimizations lead to efficient runtimes, as shown in the last row of Table 2. In comparison, ProteinProphet took 13.3s on the yeast data and 10.7s on the ISB 18 data. MSBayes took 2m23.5s on the Sigma 49 data. We did not have access to the proper files to time ProteinProphet on the *H. influenzae* data.

Unfortunately, even after the transformations, the search space associated with the larger data sets is still prohibitively large. In order to guarantee the efficiency of our algorithm on large data sets, we approximate the original problem by pruning low-scoring PSMs as if they were zero-scoring PSMs. With this approximation, the pruning procedure creates subgraphs with many fewer proteins. Because the user is only interested in using the smallest threshold that will sufficiently break apart the connected subgraphs, we perform this process recursively and divide each subgraph using a successively greater flooring threshold. This process is continued until the total number of steps necessary for marginalization is less than a user-specified value. The result is that, rather than choosing one strict threshold for the entire data set, the user can specify a permitted computational complexity, and then different thresholds are employed to ensure that the method is as efficient as the user requires.

Occasionally, it is necessary to apply the pruning procedure to a PSM with a larger probability. In these cases, a collection of proteins are connected through a collection of high-scoring PSMs. These cases are already known to be difficult; in the extreme case, when all PSMs have probability 1.0, this problem closely resembles the NP-hard minimal set coverage problem (except, in our case, marginalization requires that each permutation of present and absent protein states must be considered). Fortunately, any error introduced by pruning will only distort the probabilities of proteins connected in this way; therefore, accurate protein posteriors

may be achieved as long as these cases are relatively rare. In the supplement, we show the distribution of PSM probabilities that must be pruned to achieve no more than 2^{18} marginalization steps, and demonstrate that few pruned PSMs have probabilities greater than zero. When such a PSM is pruned, the two partitions it joins are approximated as being independent (even though they may not be). In these cases our method behaves similarly to the first iteration of ProteinProphet, by treating the peptides as independent.

3.4 Comparison of our method, ProteinProphet and MSBayes

We evaluate a C++ implementation of our method using the five data sets described in Materials and Methods. The source code of this implementation is publicly available (http://noble.gs.washington.edu/proj/fido). Starting from the scored peptides, each method computes a probability for each protein, and these probabilities are used to rank the proteins. Groups of identically connected proteins are merged for evaluation, and are treated as a single protein group. Whenever we refer to the number of target proteins or decoy proteins identified at a threshold or use these values in a calculation, each protein group is counted once, rather then once for each protein it contains. Groups containing both targets and decoys are not counted in evaluation; such groups are so infrequent that their treatment doesn't visibly change the figures presented.

From each ranked list of proteins, we evaluate the method by creating a receiver operator characteristic (ROC) curve, which plots true positive counts (i.e., the number of target proteins) as a function of false positive counts (the number of decoy proteins). A curve is produced by varying the probability threshold above which a protein is deemed to be present. Because we are particularly interested in the performance of the algorithms when the false positive rate is low, we only plot the curve out to 100 false positives along the *x*-axis. We also evaluate each ranked list of proteins using a calibration false discovery rate (FDR) plot, which plots the empirical FDR as a function of the estimated FDR. The empirical FDR is calculated as the number of decoys identified divided by the total number of proteins identified. In order to estimate the FDR using posteriors, we exploit the fact that the probability that a protein is absent is equal to one minus the posterior assigned to the protein; therefore, by assuming that the posterior probabilities are independent, we can estimate the FDR for any set of proteins by computing the expected number of false positives (found by the number of proteins minus the sum of their posteriors) divided by the number of proteins identified at the threshold. If our method is perfectly calibrated and if the empirical FDR estimate is accurate, then the empirical FDR and estimated FDR should be equal at every threshold.

Figure 4 shows, for each data set, ROC curves for our method and either ProteinProphet or MSBayes. We compare against ProteinProphet for the first four data sets, and MSBayes for the Sigma 49 protein mixture. ProteinProphet has previously been demonstrated to perform similarly to MSBayes on this data set [Li et al., 2008]. We do not compare against MSBayes on any other data sets, because the model it employs was trained to be used for the Sigma 49 protein mixture; hence, attempting to compare performance on another data set would be unfair. The ISB 18 protein data set includes many replicate analyses, so we show the ROC for protein inference using the pooled replicate data sets. On the yeast data set, our method performs better than ProteinProphet. On the *H.* influenzae data, our method performs nearly identically to ProteinProphet. On the Sigma 49 data set, our method performs similarly to MSBayes, but achieves a smaller minimum FDR. For the pooled ISB 18 data set, we observe a substantial improvement over ProteinProphet in the low false positive region. The diagonal line produced by ProteinProphet for the ISB 18 data set corresponds to 25 proteins that each receive a probability of exactly 1.0; because this data set includes many spectra, many PSMs associated with the decoy proteins are assigned scores larger than zero, and the ProteinProphet algorithm assigns these decoy proteins scores of 1.0.

In addition to ROC curves, the plots in Figure 4 contain series labeled "Overlap," corresponding to the number of proteins identified by both methods at a given number of false positives. In every case, the overlap line is very close to the ROC curves, indicating that the methods are consistent with one another and identify a largely overlapping set of proteins at each score threshold.

Table 3 depicts the sensitivity of the methods at different empirical FDRs. Our method outperforms ProteinProphet on the yeast data and performs significantly better than ProteinProphet on the ISB 18 data set. On the *H. influenzae* data set, our method performs almost identically to ProteinProphet; this similarity can also be observed in the ROC plot in Figure 4A. On the *C*. elegans data set, our method performs better at the 0.0 FDR, worse at the 0.01 FDR and better for higher FDRs. On the Sigma 49 data set, our method performs better than MSBayes, which does not achieve a FDR less than 0.10.

Figure 5 shows, for each data set, the calibration of the posterior probabilities assigned by the different methods. In these figures, we compare the estimated FDR to the empirical FDR. On the yeast data set our method has better calibration accuracy compared to ProteinProphet. For the Sigma 49 data set we achieve better calibration than MSBayes. On the ISB 18 data set our method is much better calibrated than ProteinProphet. On the *H. influenzae* data set, both our model and ProteinProphet are very well calibrated and achieve similar results. On the *C*. elegans data set, our method's calibration is similar or slightly inferior to ProteinProphet.

It should be noted that our empirical FDRs are estimates, which necessarily include some error. In particular, all decoy proteins are known false positives but not all target proteins are always present. As a result, we may underestimate the empirical FDR in Figure 5. However, this observation does not alter our conclusion that our model is similarly or better calibrated compared to ProteinProphet and MSBayes. For four of the five sets (*H. influenzae*, ISB 18, *C*. elegans, and Sigma 49) our FDR calibration curve is nearly identical to or below the curve from ProteinProphet, indicating that our method is at least as conservative as ProteinProphet. Furthermore, on these data sets ProteinProphet is less conservative than an ideal model. Due to the high level of agreement among the algorithms in Figure 4, it is reasonable to assume that both curves would similarly move upward; therefore, any negative bias to the empirical FDR estimation would move both curves similarly upward, causing our model to remain better calibrated than ProteinProphet. In other words, after correcting for absent targets, it is preferable to have a more conservative model, and our model is more conservative on these data sets. Furthermore, the ISB 18 and Sigma 49 data sets consist of several proteins directly purified into the sample. In these cases, there should be little or no error to the estimated empirical FDR, because no proteins from the target database should be absent.

We cannot be certain whether we are better calibrated on the remaining yeast data set. However, at the 0.05 estimated FDR level (which will not be influenced by the potential empirical FDR bias), we estimate the empirical FDR at 0.034. Even if the empirical FDR was underestimated by 50%, our method would be nearly perfectly calibrated. For our method to have significantly worse calibration than ProteinProphet, the bias towards absent targets would need to be substantial.

Our probability model requires the estimation of three free parameters, α, β, and γ. We empirically choose the set of parameters that jointly maximizes the ROC_{50} score (the average sensitivity when allowing between zero and 50 false positives) and minimizes the mean squared error (MSE) from an ideally calibrated probability. We compute the calibrated MSE by integrating the square of the difference between the estimated and the empirical FDR over the estimated FDR range [0, 0.1]. We then perform a rough three-dimensional grid search in the range [0.01, 0.76] at resolution of 0.05 for α, in the range [0.00, 0.80] at resolution 0.05 for

β, and in the range [0.1, 0.9] at resolution 0.1 for γ. For each triplet of parameters, we compute both the ROC_{50} and the calibration MSE. For each data set we then select the triplet of parameters that result in an acceptable compromise between the most accurate model and the best-calibrated model. In order to demonstrate that this comprimise can be achieved objectively, we minimize $(1-\lambda)MSE - \lambda ROC_{50}$, where λ is a parameter selected to emphasize ROC_{50} or MSE; a λ approaching 1.0 will shift the emphasis to the most accurate model, and a λ approaching 0.0 will result in a more calibrated model. We have used λ = 0.15 for every data set.

Because we choose the parameters for each data set, we cannot be certain that the observed differences in performance between our method and ProteinProphet or MSBayes are not partially due to overfitting. On the other hand, the optimal α and β parameter values are similar for these data sets. Furthermore, the influence of the γ parameter is limited because we estimate it with very low resolution, and very few bits of precision are used to define it. Also, the γ parameter almost exclusively contributes to calibration because it upweights or downweights all proteins in a similar manner; using a fixed γ of 0.5, which is equivalent to using a uniform prior for all protein sets, and performing the grid search for only α and β resulted in nearly identical ROC figures. The risk of overfitting is also decreased because we are jointly optimizing both the accuracy and calibration, which are independent values. To demonstrate the robustness of our model to suboptimal parameters, we also used the values of α, β, and γ that were selected using the *H. influenzae* data set, but we applied the parameters to a each of the experiments in the ISB 18 data set. Our method attains a greater ROC_{50} score than ProteinProphet for 193/236 (81%), even when using parameters chosen from completely different data.

Table 4 shows that our method compares favorably to ProteinProphet and MSBayes when identifying proteins that contain a high-scoring degenerate peptide. On all of these data sets, our method identifies no decoy proteins that contain a high-scoring degenerate peptide. Furthermore, it does so without blindly introducing a systematic bias against such proteins. For instance, on the yeast data, we identify 88 proteins with degenerate peptides without introducing any false positives. On the ISB 18 and Sigma 49 data sets, our method identifies nearly the same number of target proteins containing high-scoring degenerate peptides as the competing methods but without identifying any decoy proteins. In contrast, ProteinProphet and MSBayes identify six and two decoy proteins on these data sets, respectively. The only data set where our method does not increase either the sensitivity or specificity without sacrificing the other is the *H. influenzae* data, on which we identify four fewer degenerate target proteins but still maintain perfect specificity. These proteins are not a significant percent of the targets identified. It should be noted that on the *C.* elegans data set, we identify many fewer target and decoy proteins that contain high-scoring degenerate peptides, but on this data set, ProteinProphet is overly permissive, while our method is overly conservative (as shown by Figure 5D). Lowering the protein threshold to 0.8 on our method yields a superior sensitivity and identical or superior specificity for both categories of proteins.

Rather than treating the PeptideProphet values as probabilities and making an *ad hoc* correction, our method analyzes all of the data in a gestalt manner. As a result, our emission model can prevent a degenerate peptide from being counted twice, and the noise model in our method can prevent spurious evidence from accumulating and awarding an absent protein a high score. On large, somewhat noisy data sets like the ISB 18 data set, ProteinProphet effectively aggregates a great deal of noise, resulting in many decoy proteins with estimated probabilities of 1.0. In contrast, our method gives these decoy proteins smaller probabilities than ProteinProphet, and more importantly, gives the decoy proteins smaller probabilities relative to several target proteins.

In Figure 6, we show a decoy protein (gi|1573516|gb|AAC22189.1|) that matches several PSMs, but the majority of these PSMs have fairly low scores and are the result of the enormous number of spectra. In contrast, most of the PSMs associated with the target protein sp|P02643|TNNI2_RABIT have scores above 0.99. Our method estimates that the target and decoy proteins have respective posterior probabilities of 1.0 (which is higher than any decoy protein posterior) and 0.00092. ProteinProphet assigns both proteins posteriors of 1.0, preventing them from being effectively ranked. For completeness, we also show the decoy protein gi|1573522|gb|AAC22195.1|, which shares a common PSM with the target. Our method likewise accumulates the relatively weak evidence supporting this decoy protein to estimate a weak posterior of 0.019 (which is lower than any target protein posterior). ProteinProphet estimates a 0.0 probability for the protein gi|1573522|gb|AAC22195.1|.

In general, even after partitioning, clustering and pruning zero-probability peptides, we cannot be sure that the running time of our algorithm will not be prohibitively high. Therefore, as described above, we allow the user to specify the log of the maximum size of the search space. If, after the graph transformations, one or more connected components contain too many nodes, then the probabilities of low-scoring PSMs in the offending components are temporarily set to zero, increasing the separability. The threshold for this zeroing procedure is adjusted recursively, on a per-component basis, to achieve the desired search space size (and, hence, running time). To test how the performance of our method varies as we adjust the size of the search space, we ran the *H. influenzae* analysis multiple times with a fixed value for α and β but different search space sizes. Figure 7 shows that the performance (as measured by ROC_{50} score) improves in a step-wise fashion as the search space increases.

4 Discussion

We have demonstrated that, using a straightforward probability model, we can efficiently marginalize to compute protein posterior probabilities with respect to a given collection of PSMs. The resulting posteriors provide rankings that often out-perform accurate and widely used existing methods, thus providing evidence that finding an exact or near-exact solution to this problem is beneficial.

As pointed out by one of the reviewers of this manuscript, the timing results shown in Table 2 are slightly unfair, because we only ran our procedure once rather than multiple times to select the parameters α and β. In the experiments reported in Figure 4, for example, we ran our procedure ~1900 times; however, we do not expect users to do such an exhaustive search in practice. Some parameter values do not make much sense; for instance $\alpha = 0$, $\beta = 1$ would award all identified proteins with identical scores. Furthermore, empirically, the optimal parameters from our dense grid search are all fairly similar and would easily be found by a much lower resolution grid search. For instance, a search over $\alpha \in \{0.01, 0.04, 0.09, 0.16, 0.25, 0.36\}$, $\beta \in \{0.01, 0.025, 0.05\}$, $\gamma \in \{0.1, 0.5, 0.9\}$ requires 54 iterations ($9 \times 3 \times 3$) and produces produces similar parameter sets and very similar results. Running our method 54 times (i.e. loading the data once and marginalizing 54 times) would take less time on the yeast data and 20 seconds longer than ProteinProphet on the ISB 18 data. Note, however, that our method results in substantially better performance on the latter.

When the marginalization is too expensive, we have proposed a pruning procedure that approximates the full marginalization. Importantly, the effects of this approximation are restricted to the graph components in which they are carried out. Thus, when a PSM with a non-zero probability must be split during the pruning procedure, the pruning will only change the probabilities assigned to the proteins in that subgraph. In the future, it may be possible to bound the error introduced by pruning around non-zero nodes. This bound could be used to design pruning strategies that would minimize the error incurred. When multiple proteins share

a large group of high-scoring peptides, the problem becomes very similar to the minimum set coverage problem, which is already known to be very difficult; it resides in the complexity class NP-Hard, making it equivalent to the Traveling Salesperson Problem. Hence, in these scenarios even a locally inaccurate approximation may be welcome if the runtime is efficient.

In this work, we have used a relatively simple enumeration strategy to select values of the three parameters, α, β, and γ. A more rigorous approach would set these parameters using cross-validation. However, such an approach would still require users to provide a decoy database. We have shown that α, β, and γ are robust across different data sets. It would be interesting, therefore, to investigate strategies for estimating these parameter values without using a decoy database.

These experiments demonstrate that the quality of ProteinProphet's analysis depends heavily on the data set being analyzed. ProteinProphet's strength and weakness are derived from its implicit assumptions and its reliance on PeptideProphet. ProteinProphet implicitly assumes that PeptideProphet scores are true, unbiased probability estimates. ProteinProphet does not use protein length or the number of associated peptides to correct the PeptideProphet scores associated with a protein. In a similar manner, ProteinProphet does not correct for the total number of spectra observed. When myriad spectra are observed, nearly every peptide has a strong chance of being matched to a spectrum with a high PeptideProphet score. When ProteinProphet's assumptions prove to be reasonable, they add extra information and result in a very accurate and well-calibrated model (e.g., the *H. influenzae* data set). On data sets where these implicit assumptions are not helpful and may even be harmful, ProteinProphet performs similarly or worse than our method (e.g., the yeast and ISB 18 data sets).

One significant difference between our model and ProteinProphet is that our model explicitly allows the possibility that a high-scoring PSM is the result of an error. In data sets containing many spectra, the chance of a protein associating with an erroneous high-scoring PSM becomes higher. This effect can be represented in our model by using a larger value of the β parameter. It is important to note that even if the β parameter is larger than the α parameter, this does not mean that a peptide is more probably created from noise rather than from an associated protein. This is because β is used as the probability that a peptide is matched to a fragmentation scan given that the peptide was absent. When each protein is associated with several identified peptides, then the collective effect of these peptides make it very unlikely that they are absent, substantially lowering the influence of the noise model. ProteinProphet does consider the number of sibling peptides (NSP), the sum of other peptide scores sharing a protein association with a candidate peptide, when interpreting a score from PeptideProphet. But the manner by which this NSP correction lowers inflated peptide scores will not remove a systematic bias from all peptide scores.

The different manner in which our model handles noise is indicative of a larger fundamental difference in how our model employs PeptideProphet scores. ProteinProphet uses PeptideProphet scores as true probabilities and conditions on the NSP score to distinguish multi-hit proteins from so-called "one-hit wonders." Without conditioning on NSP, association with a single high-scoring peptide may be indistinguishable from association with many high-scoring peptides. In contrast, our model removes the prior probability estimates used by PeptideProphet and converts them back into discriminant score-based likelihoods. The difference is that ProteinProphet initially interprets PeptideProphet scores as the probability that a peptide is present given a paired spectrum was observed, whereas our method initially uses these scores to compute a value proportional to the probability that a spectrum would be created given that its paired peptide was present. For a given hypothesized set of present proteins, our model will compute the likelihood that the spectra were observed given that set of proteins was present. This subtle distinction lets our model use protein-level information

when utilizing the PeptideProphet scores to compute protein posteriors. A protein associated with many high-scoring peptides will score higher than a protein associated with a single high-scoring peptide, but without using an iterative heuristic like ProteinProphet's NSP score.

In a similar manner, our method uses protein-level information in a rigorous, gestalt manner when handling degenerate peptides, rather than by using a correction in hindsight. ProteinProphet partitions every degenerate peptide's probability between its associated proteins. Initially, the peptide is split equally between them to compute the protein probabilities, but in the next iteration, each protein is afforded a stake proportional to its most recent probability. Like the NSP correction, this approach works well, but it does so in a somewhat opaque manner; hence, identifying its implicit assumptions and improving their effects is not trivial. Rather than heuristically partitioning each identified peptide amongst its associated proteins, our graphical model allows a protein with strong independent supporting evidence to "explain away" supporting data that is shared between itself and other proteins (e.g., degenerate peptides). This happens simply because the likelihood increases only slightly by including the protein with little independent supporting evidence, but the likelihood decreases substantially when a protein with substantial independent supporting evidence is omitted. In this manner, our method automatically apportions information from degenerate peptides during the marginalization procedure and does not require an *ad hoc* adjustment.

Like other methods, our model's assumptions are not perfect. But because we have derived our model from clear, explicitly stated statistical assumptions, it may be possible to evaluate their accuracy and replace them with more relaxed assumptions. For instance, when using data dependent acquisition, assumption #2 may be inaccurate, because the population of peptides with equal hydrophobicity effectively compete in the MS1 to be selected for collision induced dissociation. Assumption #4 could be improved by treating spurious peptide identifications as mismatches between the observed spectrum and peptides that produce similar spectra at that precursor mass. Assumption #5 may be improved for certain samples by using more complex priors that more aggressively enforce competition between proteins for ownership of shared peptides. But perhaps the most fruitful avenue for future work involves relaxing the assumptions regarding the peptide emission model and the noise model (assumptions #3 and #4). Using the current model, we have observed that likelihood estimates of α and β are not as good as the empirical estimates, suggesting that relaxed assumptions will better model the type of data observed in practice. It is intuitive that the likelihood estimates do not match the empirical estimates, because likelihood estimates essentially infer a uniform prior on all α, β pairs despite the inherent relatedness between the parameters (the contribution of β decreases as α increases or as graph connectivity increases). A more accurate model of these probabilities may preserve the optimizations we have introduced, while taking into account peptide-specific information. Furthermore, a more sophisticated model would make far greater use of the entire graph by including peptides that are not matched to any spectra (without unfairly penalizing proteins that contain many undetectable peptides).

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

We are grateful for the suggestions of Jimmy Eng, Jason Weston, and Marina Spivak. We would like to thank Alexey Nesvizhskii for providing the *H. influenzae* data and for analyzing it with ProteinProphet. We would also like to thank the authors of [Li et al., 2008], who made their processed data and an implementation of their model available to better compare the methods. This work was supported by NIH awards R01 EB007057 and P41 RR0011823.

References

Eng JK, McCormack AL, Yates JR III. An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. Journal of the American Society for Mass Spectrometry 1994;5:976–989.

Feng J, Naiman DQ, Cooper B. Probability model for assessing proteins assembled from peptide sequences inferred from tandem mass spectrometry data. Analytical Chemistry 2007;79:3901–3911. [PubMed: 17441689]

Hoopmann MR, Merrihew GE, von Haller PD, MacCoss MJ. Post analysis data acquisition for the iterative MS/MS sampling of proteomics mixtures. Journal ofProteome Research 2009;8(4):1870–1875.

Käll L, Canterbury J, Weston J, Noble WS, MacCoss MJ. A semi-supervised machine learning technique for peptide identification from shotgun proteomics datasets. Nature Methods 2007;4:923–925. [PubMed: 17952086]

Keller A, Nesvizhskii AI, Kolker E, Aebersold R. Empirical statistical model to estimate the accuracy of peptide identification made by MS/MS and database search. Analytical Chemistry 2002;74:5383–5392. [PubMed: 12403597]

Klimek J, Eddes JS, Hohmann L, Jackson J, Peterson A, Letarte S, Gafken PR, Katz JE, Mallick P, Lee H, Schmidt A, Ossola R, Eng JK, Aebersold R, Martin DB. The standard protein mix database: a diverse data set to assist in the production of improved peptide and protein identification software tools. Journal of Proteome Research 2008;7(1):96–1003. [PubMed: 17711323]

Li Q, MacCoss MJ, Stephens M. A nested mixture model for protein identification using mass spectrometry. Annals of Applied Sciences. 2009 Submitted.

Li, YF.; Arnold, RJ.; Li, Y.; Radivojac, P.; Sheng, Q.; Tang, H. A Bayesian approach to protein inference problem in shotgun proteomics. In: Vingron, M.; Wong, L., editors. Proceedings of the Twelfth Annual International Conference on Computational Molecular Biology, volume 12 of Lecture Notes in Bioinformatics; Springer; Berlin, Germany. 2008. p. 167-180.

Ma Z-Q, Dasari S, Chambers MC, Litton M, Sobecki SM, Zimmerman L, Halvey PJ, Schilling B, Drake PM, Gibson BW, Tabb DL. IDPicker 2.0: Improved protein assembly with high discrimination peptide identification filtering. Journal of Proteome Research. 2009

Nesvizhskii AI, Keller A, Kolker E, Aebersold R. A statistical model for identifying proteins by tandem mass spectrometry. Analytical Chemistry 2003;75:4646–4658. [PubMed: 14632076]

Nesvizhskii AI, Vitek O, Aebersold R. Analysis and validation of proteomic data generated by tandem mass spectrometry. Nature Methods 2007;4(10):787–797. [PubMed: 17901868]

Park CY, Klammer AA, Käll L, MacCoss MP, Noble WS. Rapid and accurate peptide identification from tandem mass spectra. Journal of Proteome Research 2008;7(7):3022–3027. [PubMed: 18505281]

Price TS, Lucitt MB, Wu W, Austin DJ, Pizarro A, Yokum AK, Blair IA, FitzGerald GA, Grosser T. EBP, a program for protein identification using multiple tandem mass spectrometry datasets. Molecular Cell Proteomics 2007;6(3):527–536.

Qian WJ, Liu T, Monroe ME, Strittmatter EF, Jacobs JM, Kangas LJ, Petritis K, Camp DG II, Smith RD. Probability-based evaluation of peptide and protein identifications from tandem mass spectrometry and sequest analysis: The human proteome. Journal of Proteome Research 2005;4(1):53–62. [PubMed: 15707357]

Reiter L, Claassen M, Schrimpf SP, Jovanovic M, Schmidt A, Buhmann JM, Hengartner MO, Aebersold R. Protein identification false discovery rates for very large proteomics data sets generated by tandem mass spectrometry. Molecular and Cellular Proteomics 2009;8(11):2405–2417. [PubMed: 19608599]

Tabb DL, Fernando CG, Chambers MC. Myrimatch: highly accurate tandem mass spectral peptide identification by multivariate hypergeometric analysis. Journal of Proteome Research 2007;6:654–661. [PubMed: 17269722]

Weatherly DB, Astwood JA III, Minning TA, Cavola C, Tarleton RL, Orlando R. A Heuristic method for assigning a false-discovery rate for protein identifications from Mascot database search results. Mol Cell Proteomics 2005 Jun;4(6):762–772. [PubMed: 15703444]

Zhang B, Chambers MC, Tabb DL. Proteomic parsimony through bipartite graph analysis improves accuracy and transparency. Journal of Proteome Research 2007;6(9):3549–3557. [PubMed: 17676885]

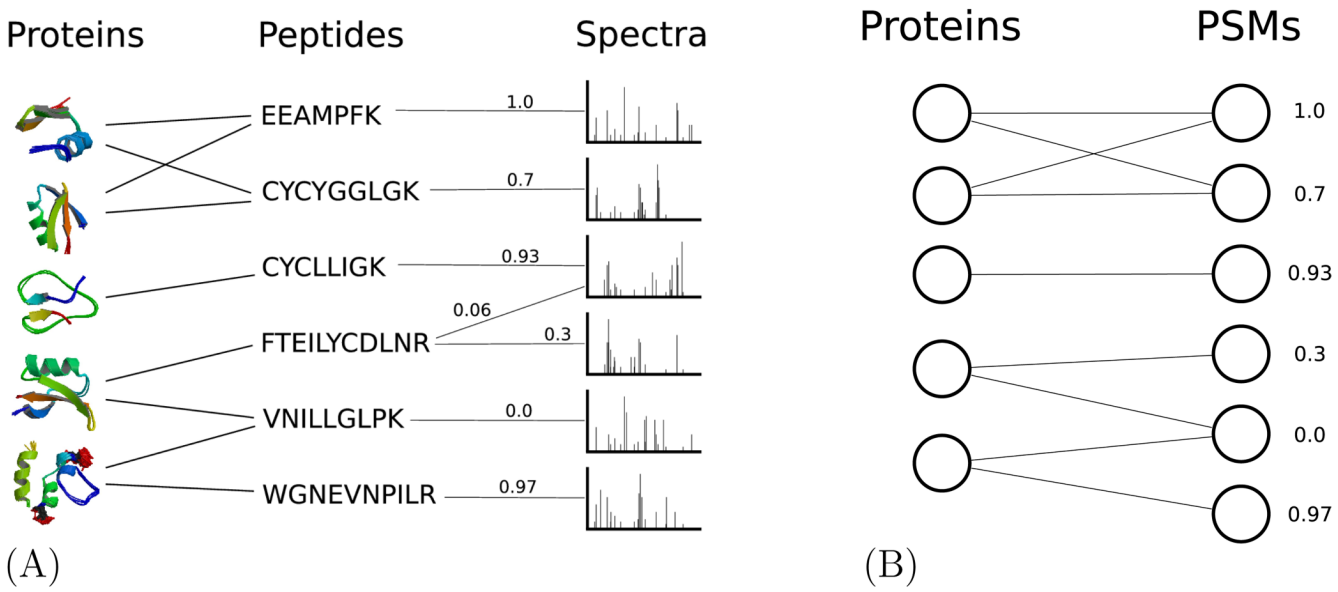


Figure 1. The tandem mass spectrometry process as a graph

(A) Proteins are connected to peptides that they would theoretically yield when digested. Spectra are connected to matching peptides, with weights represented the quality of the match. **(B)** By taking only the highest scoring peptide match for each spectrum and highest scoring spectrum match for each peptide, the peptides and spectra can be merged into a single layer of PSMs.

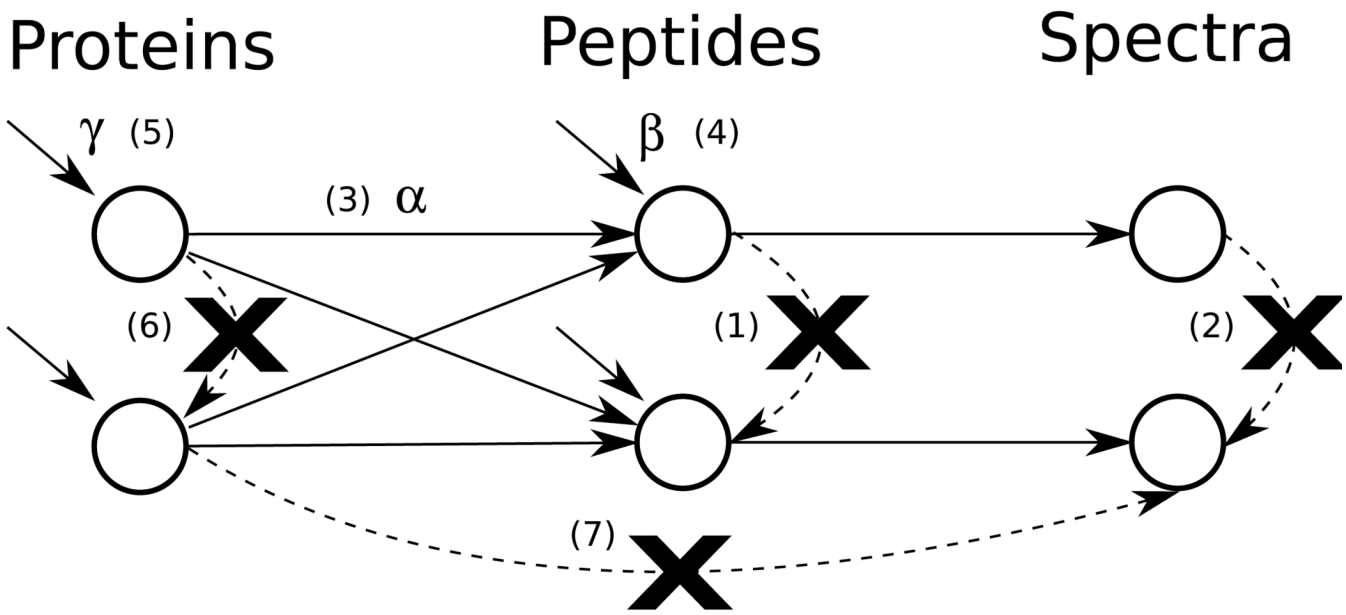


Figure 2. Assumptions

The assumptions of the model are illustrated graphically and numbered by their corresponding assumption numbers from Section 3.2. Solid arrows represent dependencies. Peptides depend on the proteins and the noise model; present proteins emit associated peptides with probability α, and peptides that are not created by associated proteins are created by the noise model with probability β. Spectra depend exclusively on the best-matching peptide. Proteins have identical and independent prior probabilities γ. The marked-out dashed arrows depict dependencies that do not exist within the model.

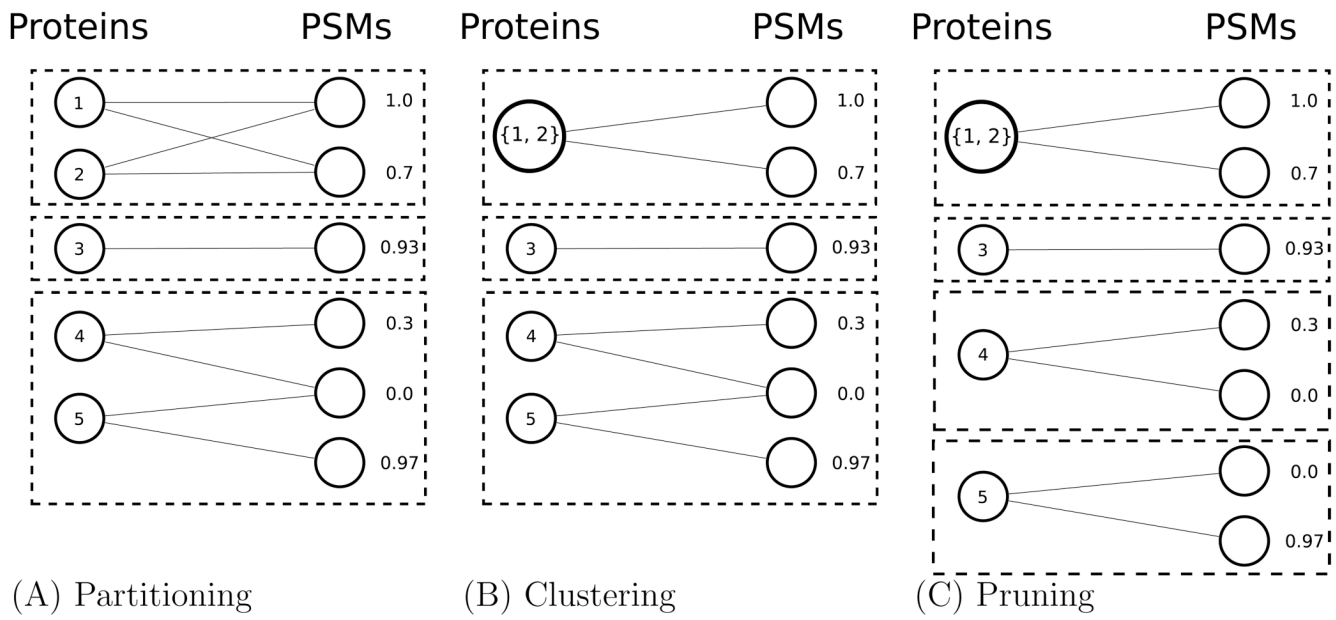
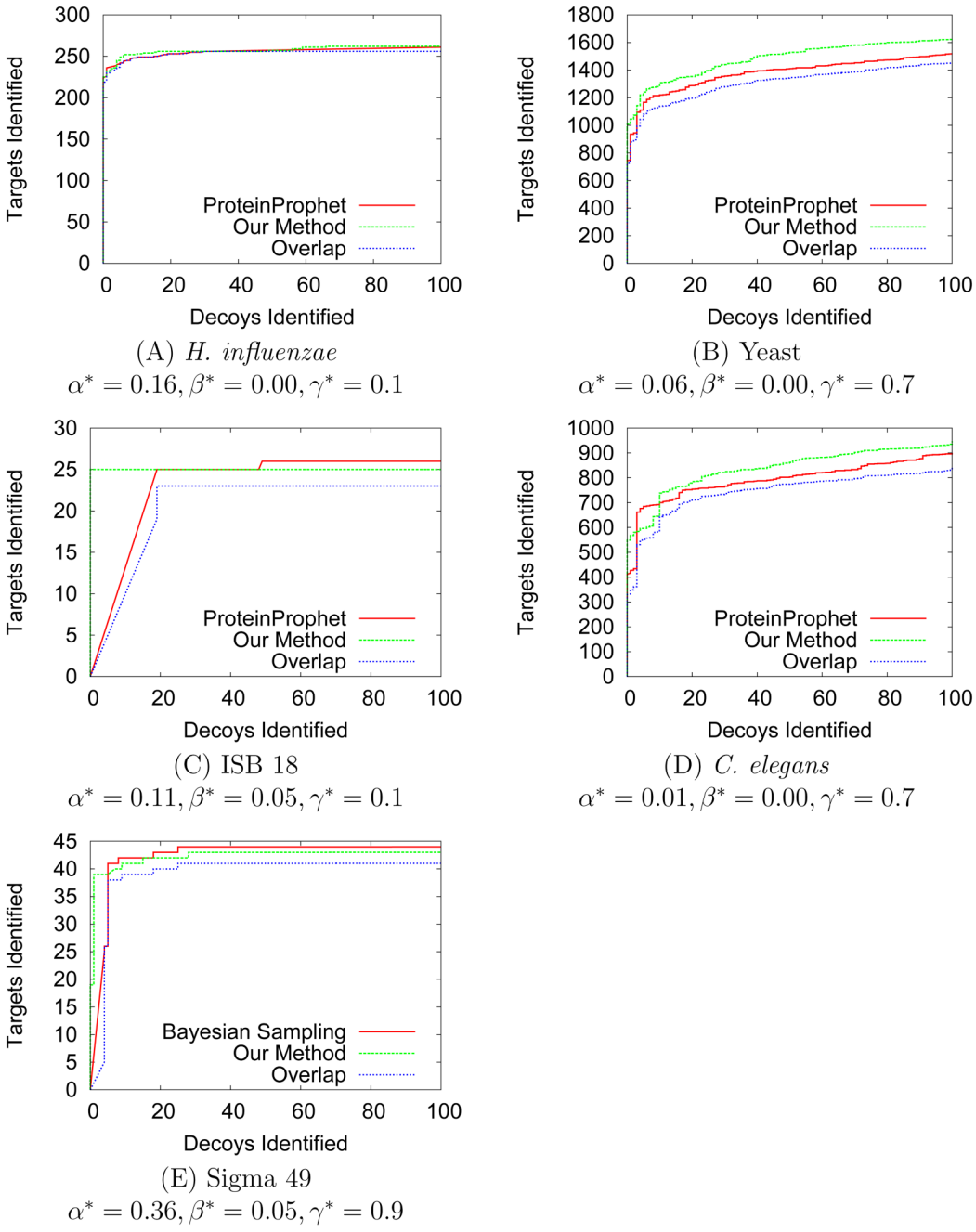


Figure 3. Three speedups

(A) The graph is partitioned into connected subgraphs (enclosed by dashed boxes). Posterior probabilities can be computed individually for each connected subgraph. **(B)** Proteins with identical connectivity are clustered together. In this example, proteins 1 and 2 are clustered to more efficiently enumerate the power set. **(C)** Graph components joined only by zero-probability peptides are separated by creating a copy of each of these peptides for each subsection. This operation further divides existing partitions to create partitions containing fewer proteins.



(A) *H. influenzae*
$\alpha^* = 0.16, \beta^* = 0.00, \gamma^* = 0.1$

(B) Yeast
$\alpha^* = 0.06, \beta^* = 0.00, \gamma^* = 0.7$

(C) ISB 18
$\alpha^* = 0.11, \beta^* = 0.05, \gamma^* = 0.1$

(D) *C. elegans*
$\alpha^* = 0.01, \beta^* = 0.00, \gamma^* = 0.7$

(E) Sigma 49
$\alpha^* = 0.36, \beta^* = 0.05, \gamma^* = 0.9$

Figure 4. ROC plots

For each data set, we plot the number of true positives as a function of the number of false positives. We compare our method to ProteinProphet in **(A)–(D)** and compare to the ABLA model of MSBayes in **(E)**. The figure also shows the overlap between the sets of true positive proteins found at each false positive level. Our method is run on each data set with the same set of parameters used for the same data set in Figure 5.

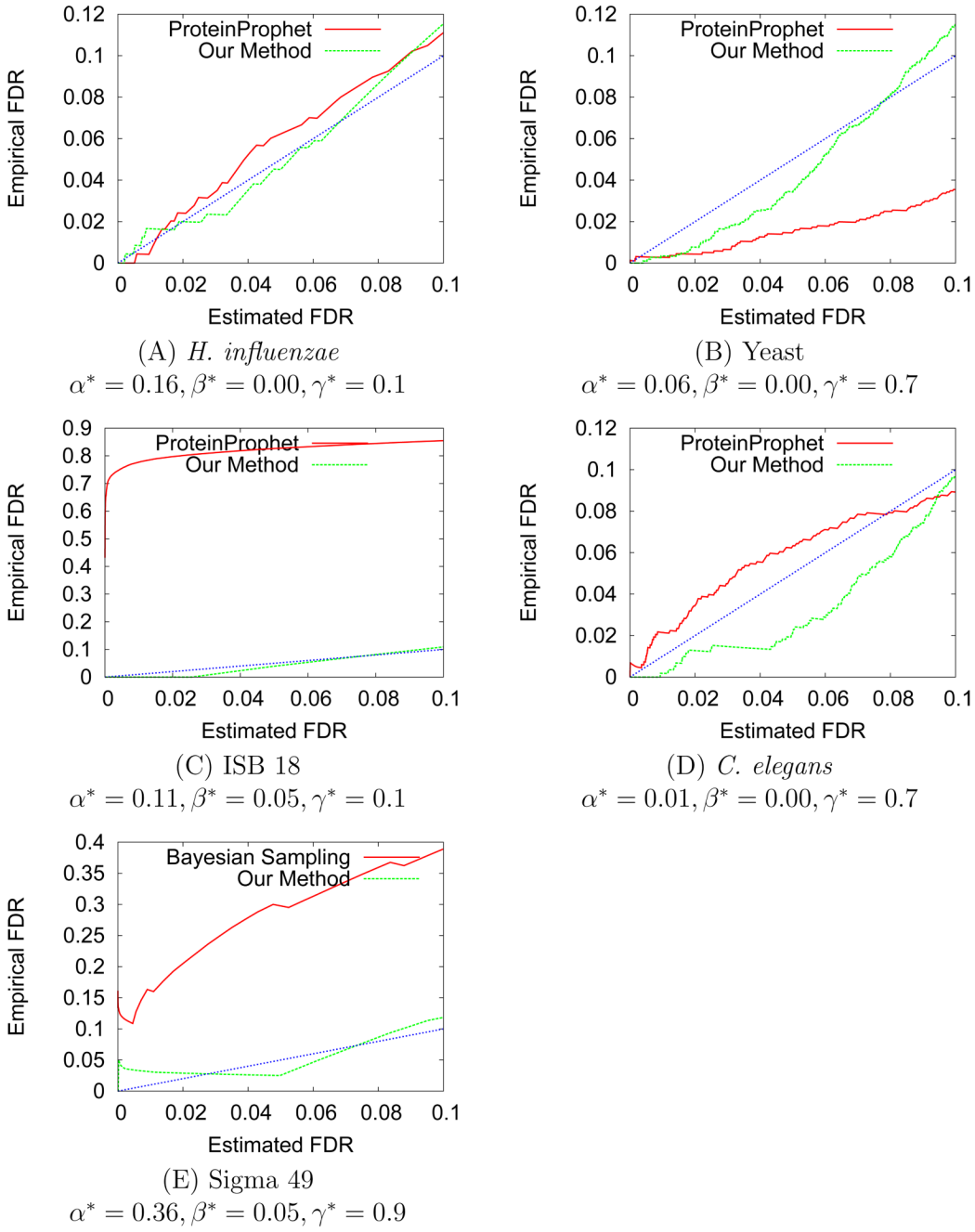


Figure 5. FDR calibration plots

For each data set, we plot the decoy database estimate of the empirical FDR as a function of the estimated FDR. We compare our method to ProteinProphet in **(A)–(D)** and compare to the ABLA model of MSBayes in **(E)**. We also plot the line along which the two axes are equal, which represents an ideally calibrated model (without considering bias introduced in the estimation of the empirical FDR). Our method is run on each data set with the same set of parameters used for the same data set in Figure 4.

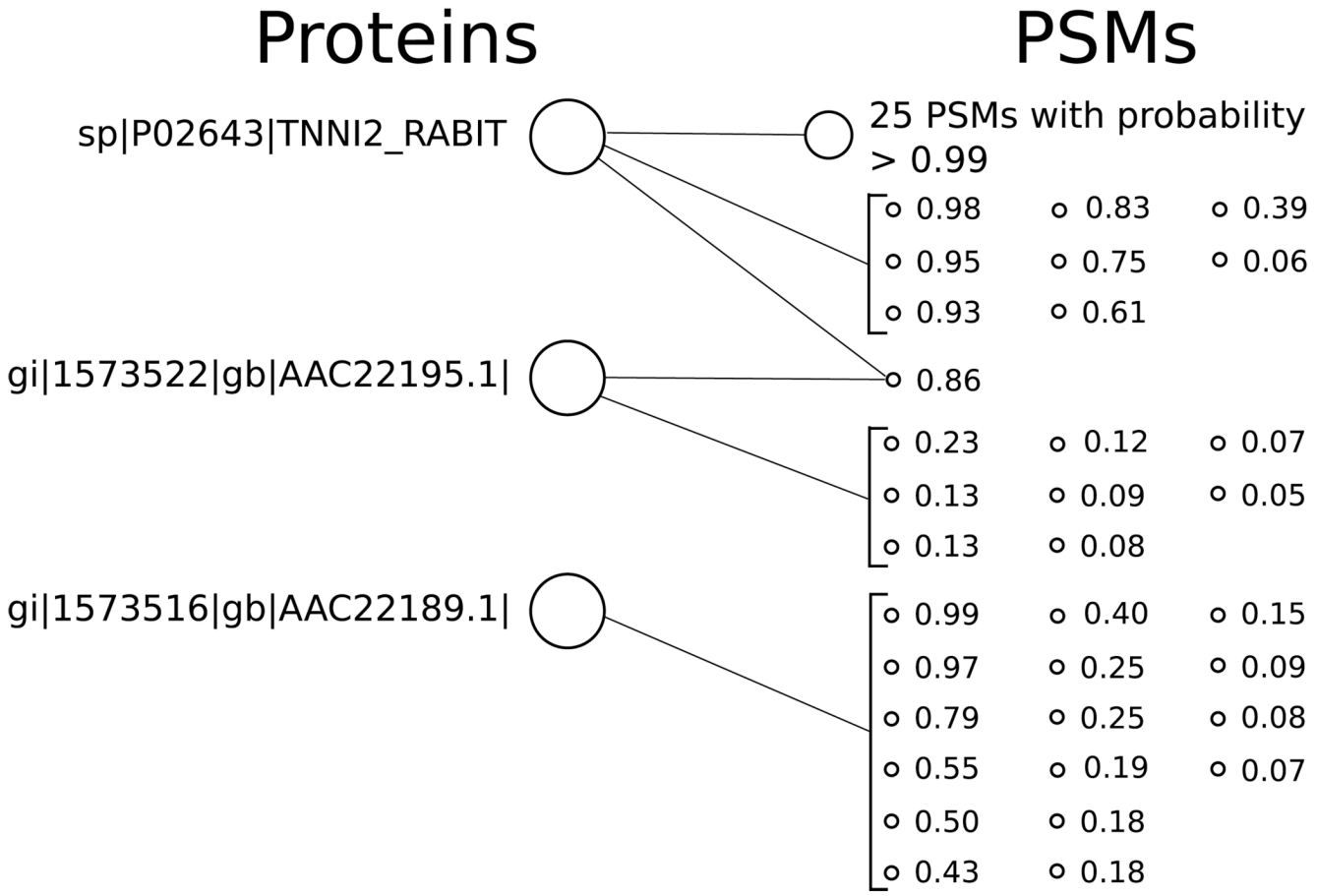


Figure 6. Example of differently ranked ISB 18 proteins

We show an example of three proteins from the ISB 18 data set, one target (sp|P02643|TNNI2_RABIT) and two decoys (gi|1573522|gb|AAC22195.1| and gi|1573516|gb|AAC22189.1|). ProteinProphet assigns posteriors of 1.0 to the target protein sp|P02643|TNNI2_RABIT and to the decoy protein gi|1573516|gb|AAC22189.1|.

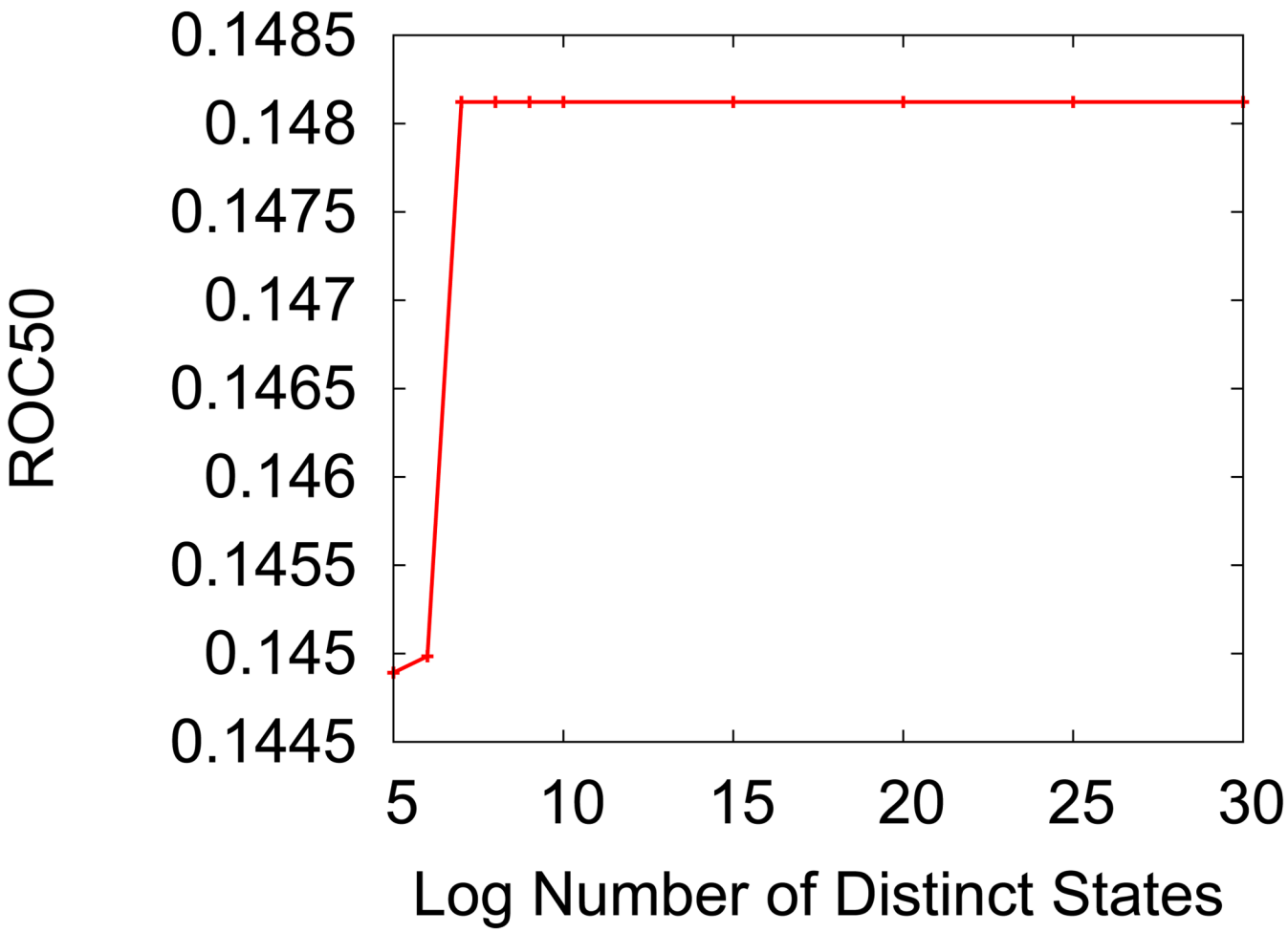


Figure 7. Accuracy against the number of states

The accuracy of the procedure on the *H. influenzae* data (measured by unnormalized ROC_{50} score) increases as the number of states allowed grows. A larger number of states corresponds to less aggressive pruning; a smaller number of states corresponds to pruning many moderately-scoring PSMs. The number of states can be thought of as the number of possible protein and peptide configurations that must be enumerated when marginalizing a connected subgraph with our optimizations. The *H. influenzae* has the most complex graph connectivity (due to the human decoy database used), as illustrated by the number of edges in Table 2.

Table 1

Data set sizes

The table lists, for the five data sets, the number of fragmentation spectra produced, the number of proteins in the target and decoy databases, and the type of decoy proteins used. All numbers are reported before any analysis was performed; Table 2 reports the number of proteins and PSMs that are actually matched and connected in the bipartite graph. Proteins were counted using their unique accession numbers, so this is the true size of the database and is invariant to redundancy or homology. The different decoy databases are taken from existing publications of protein and peptide identification algorithms, and were selected to demonstrate that our method performs well regardless of the decoy database used.

	H. influenzae	**Yeast**	**ISB 18**	***C.* elegans**	**Sigma 49**
spectra	33350	35236	1.1×10^6	42091	32700
target proteins	1709	6734	34	23932	49
decoy proteins	88299	6734	1709	23932	31227
decoy database	human	shuffled	*H. influenzae*	reversed	human and reversed human

Table 2

Utility of the Optimizations

The table lists, for the five data sets, the size of the protein identification graph (using the combined target and decoy databases) in terms of the number of PSMs (after identical peptides are merged), number of proteins, and number of PSM-to-protein edges, as well as the log of the size of the full search space and the search space after each of the three optimizations are successively applied. PRT indicates partitioning, CLST indicates clustering, and PRUNE indicates pruning (using a theshold of 10^{-3}). As before, the proteins are counted by their unique accession numbers; however, peptides are counted using their sequence, so degenerate peptides are only counted once. The final row lists the running time necessary to compute the posterior probabilities for a fixed value of the parameters α, β, and γ using a variable threshold and allowing for up to 2^{18} marginalization steps for each subgraph (using a single-core standard desktop computer). The first runtime listed is the total execution time (including file I/O to read the data); the runtime in parentheses is the time necessary to only run the marginalization procedure. The *H. influenzae* data has the highest degree of peptide degeneracy, because it includes a human decoy database.

	H. influenzae	**Yeast**	**ISB 18**	***C.* elegans**	**Sigma 49**
PSMs	29123	10390	21166	4944	23964
proteins	32748	3742	1777	4303	392
edges	60844	12202	21720	7332	24392
Log_2#	32764.6	11495.6	1833.7	4316.3	407.6
Log_2# PRT	935.2	90.4	71.6	40.0	92.3
Log_2# PRT, CLST	72.6	46.961	71.6	16.2	92.3
Log_2# PRT, CLST, PRUNE	18.3	72.6	31.4	16.2	16.9
Runtime	1.4s (0.1s)	1.6s (0.2s)	8.6s (4.3s)	1.3s (0.1s)	1.0s (0.1s)

Table 3

True Positive Identifications vs. Empirical FDR

The table lists, for the five data sets, the number of true positive identifications that the methods achieve at the greatest empirical FDR not exceeding 0.0, 0.01, 0.05, and 0.10. Methods are abbreviated as PP = ProteinProphet, MSB = MSBayes, and OBM = our optimized Bayesian marginalization method. The true positive counts are boldfaced if that method is better for that data set and at that particular empirical FDR. All FDR values for ProteinProphet applied to the ISB 18 data are missing because the minimum FDR attainable was greater than 0.43. All FDR values for MSBayes applied to the Sigma 49 data are missing because the minimum FDR attainable was greater than 0.10.

FDR	*H. influenzae*		Yeast		ISB 18		*C. elegans*		Sigma 49	
	OBM	PP	OBM	PP	OBM	PP	OBM	PP	OBM	MSB
0.0	**225**	224	**1008**	745	**25**	—	**549**	412	**19**	—
0.01	235	**237**	**1320**	1225	**25**	—	602	**687**	**19**	—
0.05	**254**	249	**1603**	1471	**25**	—	**849**	788	**39**	—
0.10	**256**	255	**1778**	1566	**25**	—	**954**	900	**39**	—

Table 4

Accuracy on Proteins Containing Degenerate Peptides

The table lists, for the five data sets, the number of true positive and false positive proteins identified by each method using a probability threshold of 0.90. The proteins identified are separated into two classes: “proteins with degeneracy” are proteins that share a high-scoring (≥ 0.90) peptide with another protein and “simple proteins” do not share such a peptide with another protein. Proteins that share a high-scoring peptide only through protein grouping are not treated as proteins with degeneracy in order to prevent any artifacts due to the treatment of grouped proteins. Methods are abbreviated as PP = ProteinProphet, MSB = MSBayes, and OBM = our optimized Bayesian marginalization method. When high-scoring degenerate peptides must be resolved, our protein identification method offers a favorable trade-off between sensitivity and specificity.

	***H.* influenzae**				**Yeast**			
	OBM		**PP**		**OBM**		**PP**	
	TP	FP	TP	FP	TP	FP	TP	FP
simple proteins	228	1	227	1	983	5	886	2
proteins with degeneracy	2	0	6	0	251	0	163	1
	ISB 18				*C.* elegans			
	OBM		PP		OBM		PP	
	TP	FP	TP	FP	TP	FP	TP	FP
simple proteins	13	0	15	87	420	4	445	14
proteins with degeneracy	10	0	11	6	173	0	273	1
	Sigma 49							
	OBM		MSB					
	TP	FP	TP	FP				
simple proteins	27	1	36	5				
proteins with degeneracy	5	0	5	2				