University of Oxford

Bayesian linear quantification for bottom-up protein mass spectrometry

by

Jesse Murray

A dissertation submitted in partial fulfillment of the degree of Master of Science in Statistical Science.

Department of Statistics, 24–29 St Giles,
Oxford, OX1 3LB

September 2021

Abstract

In this dissertation, we discuss the role that Bayesian inference has had in bottom-up protein quantification thus far. We discuss how simple protein summarizations that are commonly used can be interpreted as Bayesian models and how prior distributions can regularize summaries. We also discuss the protein inference problem more generally: how uncertainty in peptide identification and quantification are measured, and how distortions can arise from aggregating intensities into a single value when peptides can span multiple differentially expressed proteoforms. To solve these problems, we introduce an empirical Bayesian model that produces summaries that can inform us about the presence of multiple proteoforms and enables the protein summarization to be informed by uncertainty measurements. We apply this model to a THP-1 human leukaemia cell line dataset and discuss how it can provide more faithful summarizations than commonly implemented approaches.

Acknowledgements

I would like to thank my direct supervisor, Olly Crook for his incredible support and guidance through this research. I couldn't ask for a better supervisor. I would also like to thank Charlotte Deane, for her insightful comments and advice.

Contents

List of Figures

Abbreviations

The abbreviations used are: MS, mass spectrometry; MS1 scan, first MS scan; MS2 scan, second MS scan; MS3 scan, third MS scan; CID, collision-induced dissociation; ECD, electron-capture dissociation; PSM, peptide-spectrum-match; SILAC, stable isotope labelling by amino acids in cell culture; ICAT, isotope-coded affinity tag; TMT, tandem mass tag; iTRAQ, isobaric tags for relative and absolute quantification; IPTL, isobaric peptide termini labelling; PEP, posterior error probability; FDR, false discovery rate; VSN, variance stabilization normalization; MLE, maximum likelihood estimate; MAP, maximum a posteriori; MSE, mean squared error; OLS, ordinary least squares; MCMC, Markov chain Monte Carlo; HMC, Hamiltonian Monte Carlo; RWMC, random walk Monte Carlo; NUTS, No-U-Turn Sampler; LOO, leave-one-out; ESS, effective sample size; ANOVA, analysis of variance; SD, standard deviation.

Introduction

Proteins are essential biomolecules that help cells proliferate and survive. Proteins are identified and quantified using mass spectrometry techniques. Current mass-spectrometry technologies allow thousands of proteins to be measured in a single experiment, enabling the investigation of protein abundance, spatial subcellular information and protein-drug interactions.

In this dissertation, we first introduce proteomics and discuss the different general methods for identifying and quantifying proteins, especially in bottom-up proteomics, which is the focus of this dissertation. We also discuss how uncertainty is quantified in proteomics experiments. Then, we discuss the numerous ways bottom-up summarizations of protein abundance are obtained for experimental treatment groups, and how statistical testing is used to determine whether the abundance of a protein is significantly different between treatment groups. We give the Bayesian equivalents of commonly used simple summaries and build towards a more complete bayesian model for protein summarization. We then discuss MCMC methodology used for implementing Bayesian regression models. The problem of proteoforms in protein summarization is explored and we discuss recent attempts to identify proteoforms in bottom-up proteomics experiments. We then give a brief exploration of a THP-1 human leukaemia cell line dataset that we use as a reference point for building and running a new model for protein summarization. We lastly introduce this model and explain its novel advantages and apply it to a protein in the dataset. Conclusions and ideas for future work are discussed at the end.

Bottom-up vs top-down proteomics

The two main approaches to protein mass spectrometry are top-down and bottom-up proteomics (Bludau *et al.*, 2021). The main difference between these two approaches is that top-down proteomics directly measures intact (whole) proteins whereas bottom-up proteomics measures peptides (protein fragments) (Siuti and Kelleher, 2007). A secondary difference between top-down and bottom-up proteomics is that in top-down proteomics, the proteins in the crude extract from lysed cells are separated using (purified) 2-D gel electrophoresis or within the mass spectrometer (Siuti and Kelleher, 2007), whereas in bottom-up proteomics, the proteins in the crude extract are enzymatically digested together by a protease, typically trypsin, which splits proteins mainly at their lysine (K) or arginine (R) residues (Schaffer *et al.*, 2019). An important consequence of this secondary difference is that there is a loss of information in bottom-up proteomics such that the link between tryptic peptides and their corresponding proteins is lost. In a mass spectrometer (MS) the peptides (in bottom-up proteomics) or proteins (in top-down proteomics) are ionized by the electron beam source. These gas-phase ions are then accelerated in a tube and separated by their mass-to-charge ratio through deflection by a magnetic field. The separation occurs such that ions with greater mass are deflected less whereas ions with less mass are deflected more (Sparkman, 2000).

When these ions are peptides, they are often referred to in proteomics literature as 'features', and their signal intensity as 'feature intensity' (Goeminne, Ludger, 2019). Bottom-up proteomics also has deeper proteoform coverage because some of the technical challenges are mitigated, such as limitations in separation techniques and the ability of MS and tandem MS to analyze large ions (proteins) (Toby *et al.*, 2016).

In a mass spectrum plot, the mass-to-charge ratio is on the x-axis (denoted by m/z) and the signal intensity is on the y-axis. The signal intensity is typically used to infer the relative abundance (quantity) of the ion, although molecules with greater ion efficiencies will have greater intensity (Maccoll†, 1999). For clarity, ionization efficiency is the ratio of the number of ions formed to the number of electrons or photons used in the ionization process in a mass spectrometer (IUP, 2014). This is a significant source of noise in bottom-up proteomics, as peptides can have very different ionization efficiencies, leading to large differences in signal intensities between peptides even if their concentrations are equal (Marginean *et al.*, 2008; Riley *et al.*, 2015). Another important source of variability in bottom-up proteomics is that some peptides have greater enzymatic digestive efficiencies, meaning their ends are more likely to be cleaved by trypsin. This can lead to differences in the concentrations of peptides digested from the same concentration of protein (Chiva *et al.*, 2014). These sources of noise have a bearing on the inference problem described next.

For bottom-up proteomics, the first mass spectrometry scan, alternatively MS1 scan, obtains the ion intensities of intact peptides, such that the location of peaks along the mass per charge axis corresponds to different isotopes of the peptides. The second MS scan, alternatively MS2 scan, isolates the peaks of the MS1 scan by breaking up the peptides with collision-induced dissociation (CID) (Shukla and Futrell, 2000) or electron-capture dissociation (ECD) (Zubarev *et al.*, 1998) to obtain their amino acid sequence so that they can be matched to a database of proteins and their corresponding peptides (Han *et al.*, 2008). Thus, the first MS scan is for peptide quantification and the second MS scan is for peptide identification (Sticker *et al.*, 2020). The peptide ion is identified by matching the fragment spectra to peptide sequences using database search engines, such as SEQUEST (Eng *et al.*, 1994), Mascot (Perkins *et al.*, 1999), or X! Tandem (Craig and Beavis, 2004). The peptide sequence candidates are ranked by scores and the highest-ranking peptide sequence is assumed to be the correctly matched sequence of the peptide ion (Frank, 2009). It should be noted that for the Mascot database search system, the score is called the ions score, or just ion score (Savitski *et al.*, 2011). For the Mascot system, the score is calculated by $-10Log_{10}(P)$, where P is the probability that the observed match between the observed sequence of the peptide from the MS2 scan and the peptide sequence in the protein database is a chance event (Perkins *et al.*, 1999). In this way, peptide signals are matched to proteins with some degree of uncertainty (Bludau *et al.*, 2021). These matches are called peptide-spectrum-matches (PSMs), which consist of two values: a score indicating the likelihood that the spectrum matches to a given peptide (identification), and the intensity of the spectrum (quantification), with uncertainty in both (Frank, 2009). This intensity is associated with the quantity of the matched peptide, i.e., relative abundance. The intensity or quantity of a

protein must therefore be inferred from the intensity of its PSMs (Bludau *et al.*, 2021). This inference problem is shown schematically in Figure 1.

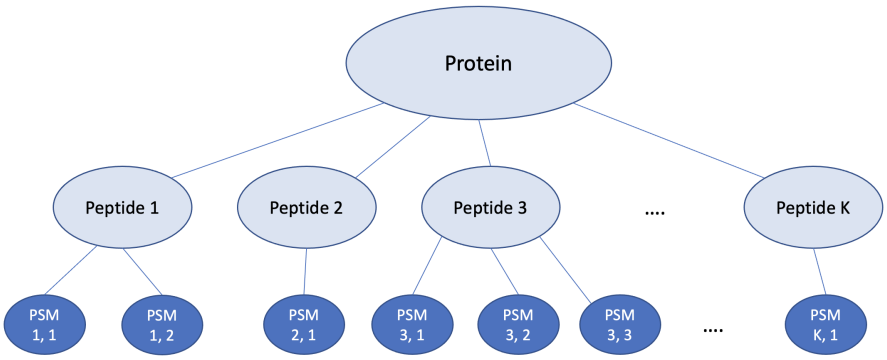


Figure 1: Diagram showing the hierarchy of PSMs matched to K peptides belonging to a protein with K. The intensities of the PSMs are observed (dark blue), though the 'intensity', or relative abundance, of the peptides and protein must be inferred.

The different identification and quantification methods

Protein quantification is key to differential abundance, which are the changes in the proteome between treatment groups, where the contrasts can be different organs, points in cell cycle (Bludau *et al.*, 2021), drug treatments (Chen *et al.*, 2014), etc. Differential abundance can be due to differential expression, which means changes in the biological expression of certain proteins. Crucial to differential abundance analysis is being able to match PSMs to the correct treatment group or experimental condition. This labelling of treatment groups is typically accomplished by the attachment of a stable isotope, which can be 'tagged' onto proteins or peptides in a variety of ways: metabolically, chemically, enzymatically, or through peptide standards (Bantscheff *et al.*, 2007). The mass spectrometer detects the mass difference between the different isotopic labels (or lack thereof), thus each PSM can be assigned to a treatment group.

Metabolic labelling can be accomplished by enriching the cell cultures of different treatment groups with or heavy or light forms of amino acids, such as lysine or arginine. The most common form of this approach is stable isotope labelling by amino acids in cell culture (SILAC), introduced in 2002 (Ong *et al.*, 2002). This has the advantage that biochemical or mass spectrometric sources of error are excluded because samples can be combined as intact cells, so that the treatment groups are treated the same way, reducing technical variance. In this way, metabolic labelling is arguably the most precise form of labelling as it occurs before protein extraction and fractionation, which can differentially affect samples (Li *et al.*, 2012). However, an important limitation of metabolic labelling is that it is restricted to certain cell lines and a maximum of three conditions can be compared (Bantscheff *et al.*, 2007) (maxi-

mum of four conditions if using the recent neutron encoding technique (Merrill *et al.*, 2014)). Alternatively, after purification or fractionation of the crude protein extracts, the proteins or peptides can be labelled with isotopes. One way is the enzymatic incorporation of ^{18}O during the digestion of proteins to peptides with trypsin (Yao *et al.*, 2001). Alternatively, one chemical approach to tagging proteins or peptides is with an isotope-coded affinity tag (ICAT), such that only peptides containing cysteine can be analyzed (Gygi *et al.*, 1999).

An important stable isotope labelling technique is tandem mass tag (TMT), which is a form of isobaric labelling, such that the TMT labels each have the same mass but have different mass after fragmentation (Thompson *et al.*, 2003). It should be noted that the different TMT labels (mass tags) are also sometimes called 'channels' (Huang *et al.*, 2020). This setup allows multiplexing, which means that multiple TMT-labelled samples can be combined in one analysis run, meaning more samples can be compared without increased experimental variance, which would occur if the samples were run separately. At the time of this writing, up to 18 samples can be analyzed in a single analysis run (18plex) (Li *et al.*, 2021). These samples can correspond to different points in time for a cell or a tissue, or different centrifuge fractions to obtain spatial information (Geladaki *et al.*, 2019). This co-isolation problem, discussed later, plagues the TMT workflow because of the large number of samples that can be run simultaneously, though it is mitigated by running a third MS scan on the TMT labelled MS2 fragment ions. The reason the MS3 mitigates co-isolation is that the MS2 peptide ion is once more isolated and fragmented and it is less likely that the selected peptide ion would again be co-isolated with another undesired peptide ion (Ting *et al.*, 2011; McAlister *et al.*, 2014). However, the use of MS3 for quantification reduces proteome coverage (the number of proteins that can be quantified) (Altelaar *et al.*, 2013). The peptide ion that is quantified and selected in the MS1 scan moves to the MS2 scan to be fragmented into its constituent amino acids for identification. For this reason, the initial peptide ion in the MS1 scan is often called the precursor ion because it is the 'precursor' to the collision-induced dissociation (fragmentation) that occurs in the MS2 scan. In this third scan, the TMT labels of different masses are used to distinguish the peptides from different samples (plexes). Often, these samples represent different treatment groups, such as points in time. The TMT labels are often called reporter ions because they 'report' the quantification of the peptide with that particular TMT label. The quantification of these different samples occurs by obtaining the intensities of the different reporter ions in the third scan (MS3 scan) (Berberich *et al.*, 2018). In summary, in the TMT workflow, identical peptides from different samples (the precursor) are ionized and fed into the MS1 scan where their intensity is obtained. Then, in the M2S scan, the identical peptides are fragmented into their constituent amino acids (residues) from which the sequence of the peptide is identified from a database. This fragmentation process results in the TMT labels having reporter ions with different masses. Then, in the MS3 scan, the different TMT labels (reporter ions) with different masses depending on the sample (plex) are identified and quantified, giving the intensity of the peptide for each of the samples (e.g. treatment groups). As mentioned earlier, the TMT labels must each have the same mass before fragmentation so that the identical

peptides from different samples appear as a single peak in the MS1 spectrum (Pappireddi *et al.*, 2019). There are a variety of other stable isotope labelling techniques: isobaric tags for relative and absolute quantification (iTRAQ), isobaric peptide termini labelling (IPTL), etc. (Chahrour *et al.*, 2015).

An alternative to isotope-labelling is label-free proteomics, which does not use a stable isotope to label proteins or peptides. Unlike isotope-labelled quantification, in which compared samples are run in the same MS scan, in label-free proteomics, compared samples are run in separate MS scans. Then, samples are compared via the signal intensity of their peptides or by using the number of spectra matching to a peptide or protein as the indicator for their amounts in the sample. Because of the different conditions across MS runs, accuracy is reduced compared to isotope-labelled methods (Bantscheff *et al.*, 2007; Li *et al.*, 2012). The different conditions between MS runs adds a source of variability (noise) in signal intensities (Anderle *et al.*, 2004). It was shown by O'Connell *et al.* (2018) in a spike-in experiment that TMT labelling has higher precision and fewer missing values than label-free quantification, and is therefore able to accurately detect more changes in differential abundance. However, label-free proteomics has deeper coverage than label-based quantification due to its reduced complexity, that is, fewer high-quality MS/MS spectra can be recorded for peptide and protein identification in label-based proteomics leading to fewer peptide and protein quantifications (Liu *et al.*, 2013).

The trade-offs involved in proteomics methods are often centered around the competing aims of identifying and quantifying as many proteins as possible (deeper proteome coverage) versus maximizing the quality of the data i.e. the signal-to-noise ratio. Both of these aims are important in differential abundance studies. A diagram comparing the proteomics methods is shown in Figure 2 and the general proteomics workflow is shown schematically in Figure 3.

One problem that arises in bottom-up proteomics is that peptides can belong to the amino acid sequence of multiple different proteins. This is more likely to occur for shorter peptides: in the extreme case, a peptide of only two amino acids can belong to an enormous plethora of proteins. For longer peptides, this complication can arise if the proteins are closely related and have similar sequences. A common way to deal with this is to simply remove from analysis those peptides that can be assigned to more than one protein, which is the approach taken for the THP-1 human leukaemia cell line dataset, introduced in section . While this approach greatly simplifies protein identification and quantification, it invariably means that potentially valuable information is discarded. The more complex approach is implemented by Gerster *et al.* (2014) in SCAMPI, which includes information from shared peptides. They found that SCAMPI obtained improved estimates of protein abundance when compared to TOPn (Silva *et al.*, 2006) (described later) and MaxQuant (Cox and Mann, 2008).

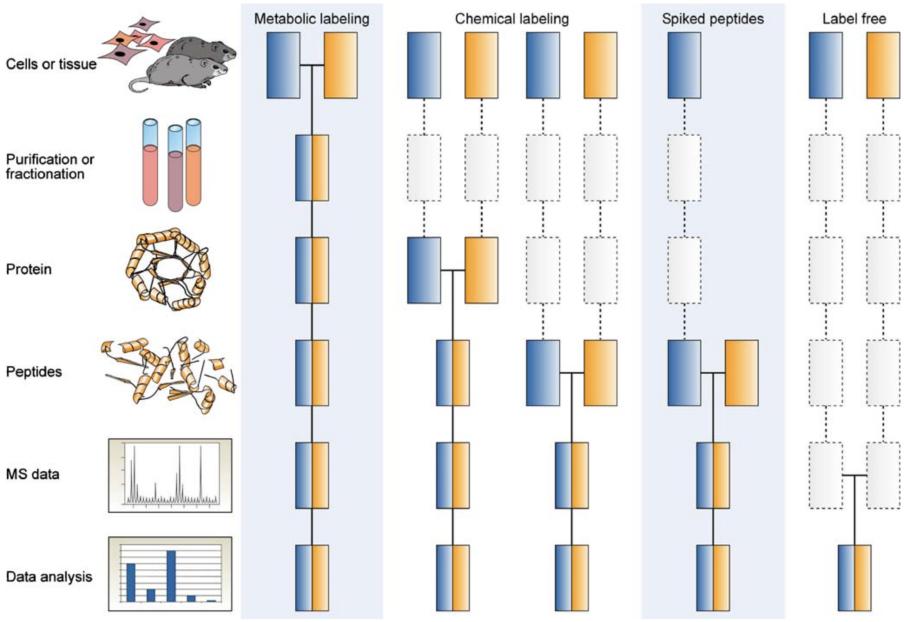


Figure 2: Diagram showing the different proteomics methods, reproduced from Bantscheff *et al.* (2007).

Uncertainty in proteomics

There are different ways of quantifying the level of uncertainty in a PSM. Recalling that the way identification of a peptide ion works is that the peptide ion is matched to the peptide sequence with the highest score, the DeltaScore is defined as the difference between the top PSM score and the score of the next most highly ranked PSM, as a proportion of the higher score. This can be 1.0 when there is only one recorded PSM score (Adamo and Gerber, 2016). Clearly, a larger DeltaScore means more relative confidence in the identification. Additionally, PSMs have q-values and posterior error probabilities (PEPs), which are complementary metrics.

Q-values are related to the false discovery rate, which is the proportion of peptide identifications that are false. Then, the q-value for a PSM is the minimum false discovery rate (FDR) threshold such that the peptide sequence is matched to the peptide ion (Käll *et al.*, 2008). This is obtained by searching the spectrum against a decoy protein database, which contains essentially random amino acid sequences (shuffled or Markov chain-generated) (Elias and Gygi, 2007). In this decoy database, all PSMs are incorrect, though some of these hits will have a higher score than the highest score against the true database. This can be used to obtain the q-value for the true score, where a q-value of 0.01 means that for higher scoring PSMs, for every 99 hits that are from the true database, 1 is from the decoy database. On the other hand, the posterior error probability (PEP) is the probability that the observed

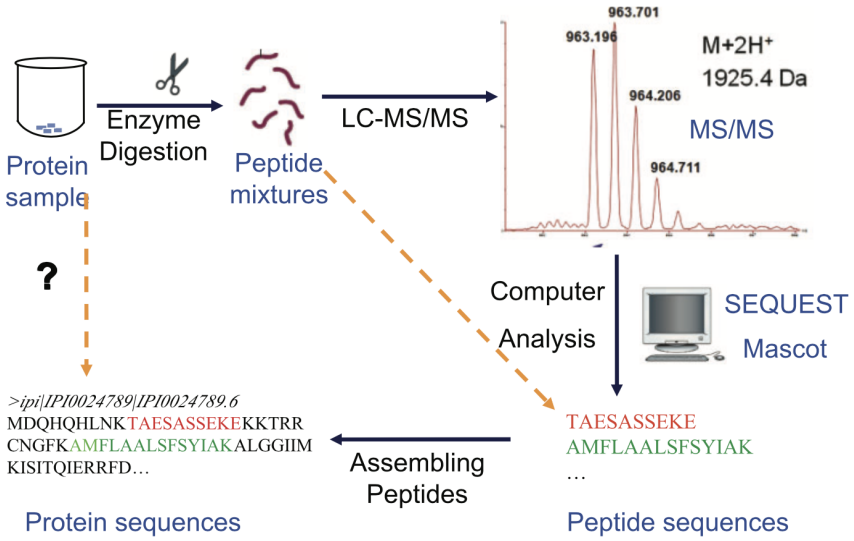


Figure 3: Diagram showing the protein identification workflow for shotgun proteomics, which is a form of bottom-up proteomics, reproduced from Huang *et al.* (2012).

PSM is incorrectly identified. This is the number of incorrect (decoy) PSMs with the score as a proportion of the total number of PSMs with that score (Käll *et al.*, 2008). These are shown graphically in Figure 4. Because the PEP for a PSM will always be greater than or equal to its q-value, the PEP is a more conservative metric. As mentioned earlier, commonly used peptide sequence databases include SEQUEST, mascot, and X! Tandem. The important software tool Percolator, introduced by Käll *et al.* (2007) acts as a post-processor of the results from searches against those databases and assigns q-values and PEPs. This is why in proteomics datasets, the q-value and PEP are called the percolator q-value and percolator PEP. Percolator uses semi-supervised machine learning to discriminate between identifications to the true database (such as mascot) and those to the decoy database (Käll *et al.*, 2007; The *et al.*, 2016).

The metrics we have discussed: PSM score (i.e. ion score), DeltaScore, PEP, and q-value; are each useful for quantifying the uncertainty present in the inference problem shown in Figure 1. Each of these metrics is relevant to uncertainty in *identification* of the PSM; however, we are also interested in metrics relevant to uncertainty in *quantification* of the PSM. These metrics could be relevant to the inference problem, such that a model that considers these sources of uncertainty in the summarization of the relative abundance of the protein could use more of the information available. As briefly discussed earlier, differences in the ionization efficiency and digestive efficiencies can affect the signal intensity of a peptide ion, though these are difficult to predict (Goeminne, Ludger, 2019). However, one quantifiable effect is the level of isolation interference, or co-isolation, which affects multiplex isobaric labelling techniques such as TMT and iTRAQ. Co-isolation is the effect that occurs when multiple ions are analyzed within the same ion selection window. This interference

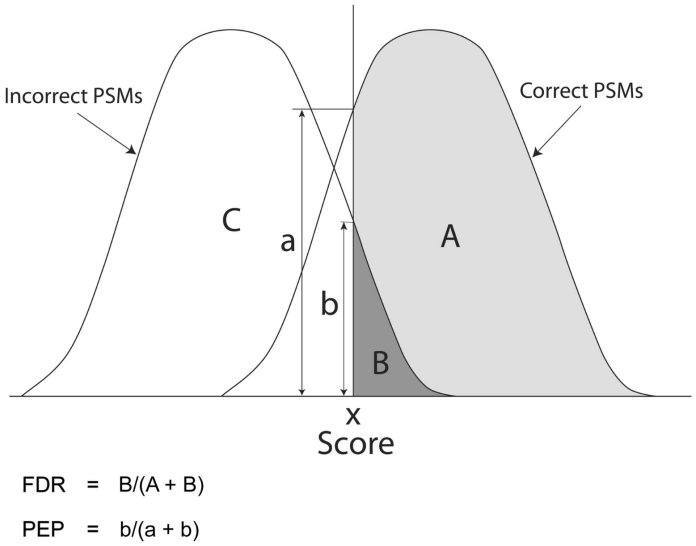


Figure 4: Here, the incorrect PSMs are from searches against the decoy database, the q-value (FDR) and PEP are given for a PSM with score x, reproduced from Käll *et al.* (2008).

causes the actual abundance differences between the reporter ions to vanish erroneously, often called compression, or ratio compression (Ow *et al.*, 2011). This compression means that differences in the relative abundance of peptides across different samples (reporter ions) are underestimated, leading to an increase in false negative identifications (Houel *et al.*, 2010). Ideally, only one precursor ion is selected, isolated, and fragmented at a time. However, it commonly occurs that other precursor ions are caught within the same m/z window (about ±2 Da) (Solis *et al.*, 2019). These precursor ions (peptides) are then fragmented together. This co-isolation affects the number of peptides identified in shotgun proteomics (Michalski *et al.*, 2011). The spectra that result from the co-isolation and co-fragmentation of two or more peptide precursor ions is called 'chimeric spectra.' This deteriorates the search scores of peptide sequence assignments to the mascot or SEQUEST databases (Houel *et al.*, 2010). It also affects quantification in isobaric labelling because the reporter ions from different peptides can have intensities that superimpose in the MS3 quantification, leading to inaccurate PSM intensities (Sandberg *et al.*, 2014).

The percentage of interference by co-isolation within the precursor isolation window is calculated in the Proteome Discoverer software (Colaert *et al.*, 2011) as the relative amount of ion current within the isolation window that is not attributed to the precursor (Solis *et al.*, 2019):

$$\% \text{ isolation interference} = 100 \times [1 - \frac{\text{precursor intensity in isolation window}}{\text{total intensity in isolation window}}]$$

A greater isolation interference percentage indicates that more ions are co-isolated with the desired peptide ion. The ratio compression effect was demonstrated by Sandberg *et al.* (2014) using spiked peptides (peptides with pre-determined measured concentrations). They

concluded that isolation interference up to about 30% yields good quantitative accuracy. Another study found that TMT quantitative measurements deviate from label-free quantification when isolation interference exceeds 20% (Savitski *et al.*, 2013). As discussed earlier, the co-isolation problem is partly resolved by using an MS3 to quantify the reporter ions, such that only about 8% of MS3 spectra are affected by isolation interference. However, this remedy has the undesired side effect of significantly reducing the proteome coverage because of the more complex instrumentation and the slower duty cycle of an MS3 setup (Altelaar *et al.*, 2013). This is consistent with the trade-off involved in proteomics methods that we mentioned earlier.

Protein Summarization

The key aim of protein quantification in bottom-up proteomics is how to obtain summaries of protein concentration in the sample from the PSM intensities. A linear relationship has been established between the intensities of the PSMs of a protein and that protein's concentration in the sample (measured in mol/L) (Silva *et al.*, 2006). However, ad-hoc approaches are used to obtain the overall protein 'intensity' from the individual PSM intensities, such as taking the median (Cox and Mann, 2008), median-polish (Mosteller and Tukey, 1977; Gatto and Lilley, 2011; Choi *et al.*, 2014), mean (Karp *et al.*, 2010), sum (Wu *et al.*, 2011), top-n-mean (Silva *et al.*, 2006), and more recently using linear modelling approaches (Sticker *et al.*, 2020). Once protein summaries have been obtained, complex context-specific modelling can take place. However, these simple approaches neglect important sources of uncertainty in the PSM identification and quantification, as well as the possible effects of different replicates and the dependencies of PSMs within the same peptide. The failure to include these effects in initial protein summarization may affect contingent investigations such as protein localization and protein-drug interactions.

An alternative to these simple summaries is Bayesian inference, which offers several advantages. Firstly, prior information can be encoded, which has important regularizing effects on parameters. Secondly, latent variables can be encoded, which allow dependencies within the data (via hierarchical grouping) to be taken into account when making summarizations. Thirdly, Bayesian inference enables the propagation of uncertainty into predictive distributions, which can be compared to actual data (Bürkner, 2017).

Bayesian inference is not entirely new to proteomics. For example, Bayesian approaches have been taken by Li *et al.* (2009); Serang *et al.* (2012) to solve the identification problem of matching peptides to one or more possible proteins, described earlier. More recently, Bayesian methods have been used to probabilistically assign proteins to sub-cellular locations (Crook *et al.*, 2018, 2019) and have also been applied to the problem of protein quantification for label-free shotgun proteomics through the graphical model *Triqler* (The and Käll, 2019). This model has a variety of advantages over simple point estimates. Instead of imputing missing values, it integrates over their prior distributions. Most importantly, it propagates

several sources of error into the final estimate of the protein quantity by using multiple latents, as shown in Figure 5. This natural framework for obtaining the uncertainty at each step (identification, quantification, and differential expression), and propagating that uncertainty to the next step through probabilities, obtains more sensible final estimates of uncertainty than the typical pipeline of using pointwise cut-offs at individual steps, which can lead to a loss of control of the FDR. *Triqler* makes use of Empirical Bayes to estimate the hyperparameters shown in Figure 5, allowing the data to form its own priors sensibly and flexibly. The prior probabilities allow observations to be smoothed that do not match prior beliefs.

The accuracy (sensitivity and specificity) of different protein summarization methods are compared using known relative amounts of proteins, i.e. ground truth data. This data can be obtained by simply simulating data (Bludau *et al.*, 2021) or by spiking in certain proteins, which means adding known concentrations of select proteins to biological samples that do not already contain the select proteins (The and Käll, 2019).

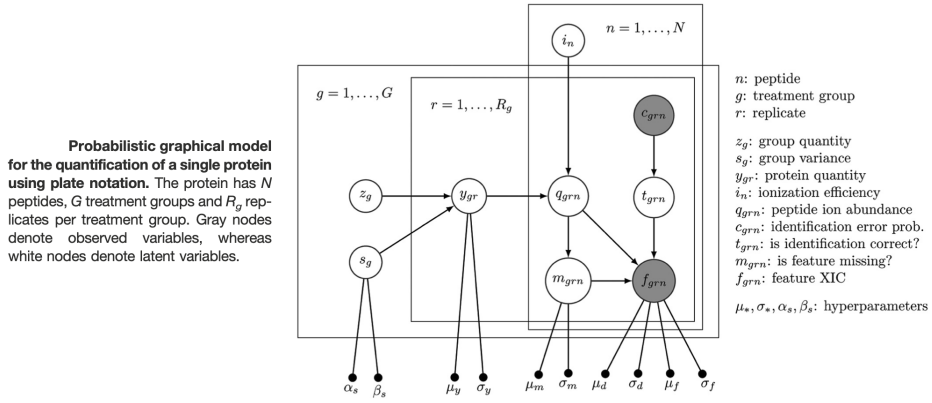


Figure 5: Figure reproduced from The and Käll (2019).

Data pre-processing

It should be noted that in each case, y represents the log-2 transformed PSM intensity. The log-2 transformation is applied because there is a strong right-skewness of PSM intensities, and the log-intensities are much more symmetrical. Additionally, a log-transformation deals with the heteroscedasticity that is observed in PSM intensities, where high intensities have large variances and vice versa (Bantscheff *et al.*, 2007). It is important to note that in relevance to the differential abundance across biological treatments, an increase of 1 in log-2 intensity corresponds to a factor of 2 increase in protein abundance.

After the log-2 transformation, the PSM intensities are commonly normalized. This normalization is intended to remove or dampen the undesired variability between different MS runs (known as technical replicates). This is particularly important for label-free proteomics,

in which the different treatment groups are separated across the different MS-runs, some of which are biological replicates (same treatment group run multiple times) (Karpievitch *et al.*, 2012; Välikangas *et al.*, 2016). There are many different ways in which separate MS runs can be normalized. One simple option is median normalization (used by Bludau *et al.* (2021)), which assumes that the intensities of different MS runs are separated by a constant and scales the data so that the intensities for each MS run have the same median. Another option is quantile normalization (used by Goeminne *et al.* (2016)), which sets the order statistics of the MS runs equal to each other. There is also variance stabilization normalization (VSN) introduced by Huber *et al.* (2002) (and used by Sticker *et al.* (2020)), which assumes that the variance of the raw (not logged) intensities for peptide p (v_p) depends on the mean of the raw intensities for peptide p (μ_p) by

$$v_p = (c_1\mu_p + c_2)^2 + c_3$$

with $c_3 > 0$. Then, a transformation h is obtained using the Delta method (Tibshirani, 1988), where $h(y)$ is of the form

$$h(y) = \gamma \text{arsinh}(a + by)$$

where $\gamma = c_1^{-1}$, $a = c_2/\sqrt{c_3}$, and $b = c_1/\sqrt{c_3}$. This transformation h ensures that the variance of the transformed quantity is independent of the mean of the transformed quantity. Then (for MS run r), a normal linear model is fit such that

$$h_r(y_{pr}) = \mu_p + \epsilon_{pr} \ .$$

There is a calibration step involved that is not explained here, though it can be found in Huber *et al.* (2002). It was shown by Välikangas *et al.* (2016) that VSN reduced variation the most between technical replicates (MS runs) for a spiked label-free experiment when compared to other normalization methods.

These normalizations are important for mitigating the biases of different MS runs, however in stable-isotope labelled data, such as TMT, there is another important source of technical variability, which is the different channels (i.e. isotope tag reporter ions). One way of removing this effect for TMT-labelling is, for each PSM spectra, to divide the log-2 intensities in each channel by the sum of the log-2 intensities across all the channels (Mulvey *et al.*, 2017). (Recall that for TMT, one PSM produces p reporter ion intensities, where p is the plex of the TMT.) For example, if it is a 10-plex TMT experiment, the sum of the 10 reporter ion intensities in each PSM is normalized to 1. This is the standard pRoloc workflow (Breckels *et al.*, 2018). The rationale for this is so that the effect of absolute intensities is cancelled, and the focus is instead on relative intensities between the channels for a PSM. However, this approach discards potentially valuable information contained in the absolute intensities, which would be useful, for example, in comparing the intensities of peptides of the same protein. This information would be discarded by this normalization method as the intensities would only be provided relative to the treatment groups indicated by the channels.

An alternative approach is the following: for each reporter ion (channel), the total summed intensity across *all* PSMs is calculated, and then all individual PSM intensities are divided by this sum. The rationale for this is that the total amount of protein for each reporter ion is set equal (Arntzen *et al.*, 2010). This makes sense in that we would not generally expect the *total* amount of protein between the treatment groups, indicated by the reporter ions, to be significantly different.

After log-2 transformation and normalization, protein summarizations are obtained.

Mean

The mean summarization obtains the protein intensity by simply taking the mean of the log-2 transformed PSM intensities matched to this protein. This is analogous to a Bayesian model in which the likelihood is given by

$$y \sim N(\beta_{\text{protein}}, \sigma^2) \tag{1}$$

with a normal prior on β_{protein} with zero mean and infinite variance (i.e. a flat prior). The posterior distribution is then given by

$$\beta_{\text{protein}}|y \sim N(\hat{\beta}_{\text{protein}}, \frac{\sigma^2}{n}) \tag{2}$$

where $\hat{\beta}_{\text{protein}}$ is the mean of the data ($\hat{\beta}_{\text{protein}} = \bar{y}$). Posterior predictions can then be drawn from $N(\hat{\beta}_{\text{protein}}, \sigma^2(1 + \frac{1}{n}))$. The normal distribution comes from the fact that the mean minimizes the sum of squares (Blitzstein, 2019), and because the normal distribution penalizes the squared distance from the coefficient in its pdf, the mean is also the maximum likelihood estimate (MLE) under an ordinary normal model. The MLE is equivalent to the maximum a posteriori (MAP) estimate when using a flat prior, because the flat prior means that that the posterior distribution is made up of only the likelihood distribution. The mode is identical to the mean and median for a normal distribution. In relation to decision theory, this estimator is the Bayes estimator using a squared loss function (Young, 2005).

The normal distribution error model is consistent with the log-2-transformation converting a multiplicative error model into an additive error model. The normal distribution then applies if the PSM intensity is the product of many exchangeable or independent multiplicative factors. After the log-transformation, these become additive factors and the central limit theorem leads to a normal approximation for the residuals (Gelman, 2014).

Median

The median summarization obtains the protein intensity by taking the median of the log-2 transformed PSM intensities matched to this protein. This is equivalent to a Bayesian model in which the likelihood is given by

$$y \sim L(\beta_{\text{protein}}, b) \tag{3}$$

where L is the Laplace likelihood, and with a flat prior. There is no simple form for the posterior because of the lack of conjugacy. However, maximizing the likelihood still corresponds to the MAP estimate for the same reasons as described earlier for the mean summarization. In relation to decision theory, this estimator is the Bayes estimator using an absolute loss function. Similarly, the equivalence results from the fact that the median minimizes the sum of absolute deviations (Blitzstein, 2019). It is also the Bayes estimator under the absolute error loss (Young, 2005). The median estimator has the advantage that it is less sensitive to extreme values than the mean, because for example, if the largest value in the dataset becomes much larger, the median would not change, but the mean would increase by an amount equal to the increase in the largest value divided by n.

Sum

The sum method obtains the protein intensity by taking the sum of the log-2 transformed PSM intensities matched to this protein. This is equivalent to a Bayesian model in which the likelihood is given by

$$ny \sim N(\beta_{\text{protein}}, n^2\sigma^2) \tag{4}$$

where N is the Normal likelihood, and with a flat prior. The posterior is then given by

$$\beta_{\text{protein}}|y \sim N(\hat{\beta}_{\text{protein}}, n\sigma^2) \tag{5}$$

where $\hat{\beta}_{\text{protein}}$ is the sum of the data ($\hat{\beta}_{\text{protein}} = \Sigma y_i$). Posterior predictions can then be drawn from $\tilde{y}|ny \sim N(\frac{\hat{\beta}_{\text{protein}}}{n}, \sigma^2(1 + \frac{1}{n}))$, where $\tilde{y}$ is a new data point. It should be noted that the posterior predictions of the sum and mean models are identical, which might be unsurprising given that the sum method simply scales the data by n when estimating β_{protein}. However, this does not mean that the sum and mean summarizations are identical, their equivalency with one another breaks down when there are a different number of observations for one protein being compared to another protein. In the extreme case, the intensity of a protein in a treatment group with 5 log-PSM intensity observations, all ones, would be the same as the intensity of the protein in another treatment group with 1 log-PSM intensity observation, a five. Therefore, the sum method becomes an unreliable method of comparison if different treatment groups have different numbers of observations. On the other hand, in the special case that different treatment groups have the same number of observations each, the sum method becomes identical to the mean method, as the scaling by n is simply a constant across treatment groups.

Top-n mean

An important consideration in summarization is that measurement noise varies inversely with intensity (Karpievitch *et al.*, 2009; Carrillo *et al.*, 2009; Breitwieser *et al.*, 2011). This gives the rationale for using the top-n-approach, where typically the number of tryptic peptides

averaged is 3 (Silva *et al.*, 2006; Gerster *et al.*, 2014). For clarity, this method requires at least three different peptides (three different sequences), as opposed to including duplicate measurements of the same peptide. The usefulness of this method was supported by a study that found a linear relationship between the

The Bayesian analogy of the top-n can be obtained by considering the distribution of the sum of the largest $n-k$ out of n normal random variables, which is a problem that has been explored by Wiens *et al.* (2006). Note that this is equivalent to the mean summarization of $k = n$, and for $k < n$, the use of the normal distribution naturally follows from minimizing the sum of squares. The problem comes from electrical engineering, where cellular phone receivers select the strongest $n-k$ out of n received signals from n antennae. The assumption is made that the noise (σ^2) is the same for each of the signals. In this problem, it is assumed that the mean is different for each signal, but we can drop this assumption to match our case. The resulting distribution is defined by Theorem 2 (Wiens *et al.*, 2006), which involves a complicated k-fold integration. This means the numerical task grows with n (the number of PSM observations) grows, as k is necessarily $n-3$ for the top-3-mean.

Additional notes on summarizations

The mean, median, sum, and top-n-mean are each L-estimators, which means they are a linear combination of order statistics of the measurements (de Menezes *et al.*, 2021). We have also shown that the mean, median, and sum are each M-estimators, which means they are an extremum estimator such as a maximum likelihood estimator (de Menezes *et al.*, 2021).

Once these summaries are taken for each treatment group, a 'fold-change' is typically calculated between two treatment groups, which is simply the ratio of the protein intensity in one treatment group to the protein intensity in another treatment group, e.g., in the presence or absence of a drug. It should be noted that there is another way of obtaining the fold-change in intensity between treatment groups, which is to calculate the ratios of each peptide in the two treatment groups and then average those ratios. For example, if there are K peptides, this method calculates K ratios and then averages them to obtain the overall average. It was shown by Carrillo *et al.* (2009) in a latin square designed experiment where protein concentrations were experimentally pre-determined that this method had significantly lower performance than obtaining the sum of the treatment groups separately and then taking their ratios.

How summaries treat missing values

An additional important consideration in summarization is the presence of missing values. PSM intensities are not considered to be missing completely at random. Rather there is intensity-dependent missingness, such that lower intensities are more likely to be missing (Liu and Dongre, 2020). This can occur because those precursor intensities fall below the

limit of detection causing them not to be selected for fragmentation. This is more likely to occur for peptides with low ionization efficiencies. A low-intensity MS1 peak can become indistinguishable from background noise, and when considering the intensity-dependent noise considered earlier, this problem is exacerbated further as the level of noise is greater for values closer to the limit of detection (O'Brien *et al.*, 2018). Additionally, high-intensity peaks from the first MS scan are preferentially selected for fragmentation (the second MS scan), which promotes intensity-dependent missing values. Another source of missingness is misidentification, although this is assumed to be independent of intensity (O'Brien *et al.*, 2018). Additionally, co-isolation, as described earlier, can lead to a chimeric spectrum, which means the peptide residues in the MS2 spectra are not identified, misidentified, or only one of the precursor peptides is identified (Gorshkov *et al.*, 2015). These each result in a missing value.

An important consideration is how the mean median, and sum implicitly treat missing values. One way to consider this is to consider what single value, if imputed in place of the missing value, would not change the final summarization. For the mean summarization, this is simply the sample mean, for the median summarization, this is simply the sample median. The top-3-mean and sum summarizations are more considerate of the inverse relationship between intensity and missingness. That is, the top-3-mean imputes a missing value with any value less than or equal to the third largest value (any value greater than that would change the top-3-mean). Additionally, the sum method imputes a missing value with exactly zero. This is arguably a stronger assumption than the other imputations.

Linear summarization

These methods can be extended into linear models that take into account other variables that may affect the intensity of the PSM (Clough *et al.*, 2009). For example, the samples from the same biological replicate are more similar than the samples from different biological replicates. Therefore, the assumption that the PSMs from the same biological replicates have the same dependency as the PSMs from separate biological replicates is not sensible. And yet this independence assumption is assumed in the summarizations described earlier. These kinds of effects can be accounted for in a linear regression model. These enable a more sensible dependency structure by including the effect of covariates such as the technical replicate or biological replicate (Goeminne *et al.*, 2015). A technical replicate can mean a separate sample from the same biological source with the same treatment, and a biological replicate can mean a sample from a different biological sample, but with the same treatment. These analogize to the Bayesian framework in that the effects in the linear model are essentially latent variables, which means they can have their own prior distributions.

The mean, median, and sum methods can be extended to linear models by using the Normal and Laplace distributions for the error model, as described earlier. It should be noted that the variables in the linear combination are typically categorical (indicator) variables (i.e. factors), which indicate whether or not the PSM is part of the treatment, technical replicate,

biological replicate, or peptide sequence, for example. This means that the data matrix, typically denoted by X contains only 1s and 0s.

A normal linear regression model of this kind was discussed by Goeminne *et al.* (2015) to summarize the association between the treatment group and the differential abundance of a protein, while accounting for the effects of the experimental replicate, given for each protein by

$$y = \beta_{\text{treat}} + \beta_{\text{exp}} + \epsilon \ . \tag{6}$$

In this linear model, the error term ϵ is normally distributed with mean zero and variance σ^2.

Including effects of peptides

Goeminne also extended this linear model to include the effects of the peptide sequence. Further, he separated out the effects of the biological and technical replicates. This updated model is called a peptide-based model and is given for each protein by

$$y = \beta_{\text{pep}} + \beta_{\text{treat}} + \beta_{\text{biorep}} + \beta_{\text{techrep}} + \epsilon \tag{7}$$

where the error term ϵ is also normally distributed with mean zero and variance σ^2. This model accounts for the effects of individual peptides, which were found to be strong by Karpievitch *et al.* (2009) who used a similar linear model that accounted for peptide effects. The within-biological replicate or within-technical replicate correlation can be considered orthogonal to the correlation between peptides of the same protein. Goeminne et al. hypothesized that controlling for these effects by including them in the linear model would improve quantification.

In Equations 6 and 7, the latent variable of interest is β_{treat}, which is the summarization of the protein intensity so that differential abundance (and possibly differential expression) can be investigated across the different treatments. As an example, when there are two treatments, one of them being a control, a t-test can be used to investigate the significance of the treatment of interest (Goeminne *et al.*, 2015).

Goeminne et al. tested the performance of peptide and non-peptide-based models by spiking in human UPS proteins at five known concentrations. They then compared the models using ROC curves of the 10 pairwise comparisons. They were able to show that peptide-based models consistently outperformed summarization-based models, which merely calculated the overall sample mean and median of the PSM intensities belonging to a protein-treatment as described in the earlier sections on the Mean and Median summarizations. Goeminne et al. showed that the peptide-based models had consistently superior performance. That is, accounting for peptide effects through a linear model leads to higher sensitivity and specificity than merely summarizing the protein intensity from all the PSM intensities (Goeminne *et al.*, 2015; Clough *et al.*, 2009). This is a step towards Bayesian regression, in which latent variables are used to take into account nested dependency structures in the data that would

otherwise be ignored, which would not be faithful to the true data structure (Bürkner, 2017). In this case, the dependency structure is the shared peptide sequence of some of the PSMs.

Priors as regularization

So far, we have been using flat priors for our estimates of latent variables. However, a flat prior is generally at odds with the typical expectation for an investigation. That is, a researcher wouldn't consider an extreme fold change of, for example, 1,000,000 times more protein in treatment group B vs treatment group A to be very likely. On the contrary, the researcher generally expects that for most proteins, there is no differential abundance between treatment groups. This can also be done for the peptide, biological replicate, and technical replicate effects. One natural way this can be accomplished is by setting a normal prior for the effect centered at zero and with a non-infinite variance. A normal prior is a natural choice in part because of the normal-normal conjugacy: a normal prior and a normal likelihood will produce a normal posterior for the variable. A normal prior is also natural because it is the maximum entropy distribution with a fixed mean and variance (Young, 2005). A maximum entropy distribution minimizes the amount of prior information and is therefore generally uninformative.

This Bayesian approach is equivalent to ridge regression (such that the prior variance is $\frac{1}{\lambda}$) (Goodfellow *et al.*, 2016). This approach reduces overfitting, which occurs when the model fits too closely to the previous observations and thus becomes too complex that it does not generalize well to new observations. This equivalency between prior belief in mediocrity and regularization is important. It is worth stating that Bayesian inference is driven by Bayes rule (Bayes, 1763), which is given by

$$p(\theta|y) = \frac{p(y|\theta)\,\pi(\theta)}{p(y)} \propto p(y|\theta)\,\pi(\theta)$$

where $p(y|\theta)$ is the likelihood and $\pi(\theta)$ is the prior. De Finetti showed that as long as the likelihood model (i.e. the data generation) is infinitely exchangeable conditional on the parameter θ, then a true generative model, i.e., the true prior $\pi(\theta)$ must exist, enabling Bayesian inference to discover the generative model (Gelman, 2014). If the negative logarithm is taken of the proportional form of Bayes rule, then, in the case of ridge regression, we get something that is equivalent to what was written by Nowlan and Hinton (1992) as

$$\text{cost} = \text{data misfit} + \lambda\,\text{complexity}\ . \tag{8}$$

In this qualitative equation, we can see that λ has the effect of penalizing complexity and thus reducing the variance of the estimator, i.e., its tendency to overfit. Correspondingly, increasing λ for a parameter in a linear regression model, means reducing its prior variance (as $\sigma^2 = 1/\lambda$), which increases the strength of our prior belief that the parameter β is close to zero. It is important to note that the mean squared error (MSE) of the parameter estimate can be written as

$$\text{MSE}(\hat{\beta}) = E[(\hat{\beta} - \beta)^2] = \text{Var}(\hat{\beta}) + \text{Bias}(\hat{\beta})^2$$

where the bias is $E[\hat{\beta}] - \beta$ (Hastie *et al.*, 2009). Minimizing the mean squared error thus involves a bias-variance tradeoff. For ordinary least squares (OLS), which is what is used in a typical normal linear regression model, there are implicitly flat priors with infinite variance, such that λ vanishes to zero. The consequence of this is that the bias of the estimator is zero, however it can be shown that the variance of the ridge estimator ($\lambda > 0$) is always smaller than the variance of the OLS estimator. Similarly, the bias of the ridge estimator is always greater than zero (the OLS bias) (Farebrother, 1976). Thus, the addition of the ridge penalty (i.e. finite prior variance) necessarily reduces variance while increasing bias, managing the bias-variance tradeoff more fluidly than the rigid maximize variance, minimize bias selection that occurs with OLS (i.e. a flat prior).

Ridge regression was used in the MSqRob tool (Sticker *et al.*, 2020; Goeminne *et al.*, 2016) where a peptide-based linear model identical to Equation 7 is used for summarization and separate λ's are used to regularize each latent effect. This is actually equivalent to a mixed effects model where each effect is a separate random effect (that has its own prior distribution). For reference, a mixed effects model is written as

$$y = X\beta + Zu + \epsilon$$

where X and Z are the data matrices (i.e. design matrices) for the fixed and random effects, respectively; and the errors are independent and identically distributed normally with mean zero. Additionally, β and u are the latent vectors for the fixed and random effects, respectively. The difference between the fixed and random effects is that there is a shared prior on the fixed random effects such that they are each drawn independently from the same normal distribution $u_j \sim N(0, \sigma_u^2)$ for all u_j. This means that, for example, the use of a ridge regression λ for the (random) effect of peptides β_{pep} means that the effects of the individual peptides are each drawn independently from the same prior normal distribution. Thus, a ridge penalty for a class of effects is equivalent to parametrizing them as random effects in a mixed model. A simple relation exists between the ridge penalty λ and the prior variance of the random effect. As an example, for the random effect of the treatment group, we have the equivalency

$$\lambda_{\text{treat}} = \frac{\hat{\sigma}^2}{\hat{\sigma}^2_{\beta_{\text{treat}}}}$$

where $\hat{\sigma}^2$ is the estimated residual variance in the linear regression model (Equation 7) and $\hat{\sigma}^2_{\beta_{\text{treat}}}$ is the estimated variance of the random effect of the treatment (Goeminne *et al.*, 2016).

Ridge regression is also called L2 regularization because the L2 norm of the coefficient (β) is penalized. Alternatively, there is L1 regularization, in which the L1 norm of the coefficient is penalized. This is equivalent to a Laplace prior on the coefficient (Bornn *et al.*, 2010). This type of regression is called Lasso, and can allow coefficients to be shrunk to exactly zero, leading to sparse solutions (Tibshirani, 1996). Furthermore, a linear combination of L1 and L2 regularization is known as elastic net regularization (Zou and Hastie, 2005). Elastic net regularization is used for the prior model in BayesENproteomics, which aims to quantify identified post-translational modifications (Mallikarjun *et al.*, 2020).

Differential abundance analysis

For differential abundance analysis, we are concerned with the effect of treatment ($\beta_{\text{treatment}}$) for a particular protein to see if there are significantly different protein concentrations between treatments. Two treatments can be compared using a t-test.

A student t-test requires strict assumptions, however: that the observations are (1) normally distributed, (2) independent, and (3) homoscedastic (σ^2 is the same for all treatment groups, although the Welch two-sample t-test can be used instead of this condition is not met). The t-test then naturally applies to the mean summarization described earlier. These assumptions must be met for the analysis to be relied upon (Boneau, 1960). The t-test statistic is calculated as

$$t = \frac{\hat{\beta}_1 - \hat{\beta}_2}{\sqrt{s^2(\frac{1}{n_1} + \frac{1}{n_2})}} \tag{9}$$

where

$$s^2 = \frac{\Sigma_{i=1}^{n_1}(y_i - \hat{\beta}_1)^2 + \Sigma_{i=1}^{n_2}(y_i - \hat{\beta}_2)^2}{n_1 + n_2 - 2} \tag{10}$$

with $\hat{\beta}_1$ and $\hat{\beta}_2$ as the mean estimates for treatment 1 and treatment 2, respectively. The t-test statistic is used to obtain a p-value. Under the null hypothesis (the protein is equally abundant in the two treatment groups), the test statistic has a t-distribution with $n_1 + n_2 - 2$ degrees of freedom.

For the other summarization methods (not the mean), a t-test is typically calculated by using the empirical Bayes method in the limma package, which is part of the Bioconductor software package (Gentleman *et al.*, 2004). Empirical Bayes is the name given to any Bayesian method in which the hyperparameters that parametrize the priors are estimated from the data. The empirical Bayes moderated t-test (Smyth, 2004) uses the following linear model for each protein:

$$y = \beta_{treat} + \epsilon \tag{11}$$

where it is crucial to stress that y here is actually the protein summarization estimate, e.g., mean, median, sum, or top-3-mean from all the PSMs for that treatment and a particular technical or biological replicate. The error is normally distributed with mean 0 and variance σ^2. A t-test can be performed on β_{treat}. The empirical Bayesian model introduced by Smyth (2004) assumes

$$\frac{1}{\sigma^2} \sim \frac{1}{d_0\sigma_0^2}\chi^2_{d_0} \tag{12}$$

where σ_0^2 and $\chi^2_{d_0}$ are estimated from the data. Then, the residual standard error for a protein is given by

$$s = \sqrt{\frac{d\hat{\sigma}^2 + d_0\hat{\sigma}_0^2}{d + d_0}} \tag{13}$$

where $\hat{\sigma}_0^2$ is the variance across all of the proteins. The resulting t-test statistic for the protein in question has a t-distribution with $d + d_0$ degrees of freedom, where $d = n - p$ where p is

the number of treatments. This estimator allows information about variance to be shared across proteins. The commonly used software package Proteus uses this the limma package to conduct this type of t-test (Gierlinski *et al.*, 2018).

It should be noted that a multiple testing problem results when all of the identified proteins in the dataset are tested for differential abundance in this way. Failing to account for this will lead to an underestimated false discovery rate (FDR). The FDR can be controlled with the Bonferroni correction (Dunn, 1961), which would set the critical value to $\alpha/n_{protein}$, however this approach has been criticized as being too conservative (Diz *et al.*, 2011). Another approach is the Benjamini-Hochberg FDR (Benjamini and Hochberg, 1995), which is used in Perseus (Tyanova *et al.*, 2016) and Proteus (Gierlinski *et al.*, 2018), which are both commonly used software packages for differential abundance analysis.

Bayesian regression models

The kind of linear model discussed in the section are in the class of Bayesian multilevel models, in which data can be measured on different levels simultaneously, taking into account complex dependency structures. These kinds of models can be fit using **lme4**, which uses MLE (Bates *et al.*, 2015). A more recent package is **brms** (Bürkner, 2017), which can create and analyze Bayesian regression models using the probabilistic programming language **Stan** (Carpenter *et al.*, 2017). The **brms** package enables the user to fit and analyze mixed models, like the kind discussed in the last section, which are estimated using MCMC. theory behind this approach is described in later sections. These models incorporate the advantages of Bayesian methodology, described earlier. However, an important disadvantage of these kinds of models is the slow speed of model estimation due to posterior sampling (Bürkner, 2017). This means that there is a limited capacity of these models to conduct differential abundance across the entire proteome of an experiment.

Markov chain Monte Carlo (MCMC) simulation

Markov Chain Monte Carlo (MCMC) is a class of methods for sampling from a target probability distribution for which it would either be too difficult or too expensive to compute the probability density function (pdf) outright, i.e. the problem is not tractable. This is especially the case if the integral to obtain the marginal probability of the data (the denominator in Bayes theorem) is too difficult or impossible to calculate. This approximating approach to simulation was revolutionary when first introduced by Metropolis and Ulam (1949). The idea behind MCMC is to build a Markov chain $(X_t)_{t\geq 1}$ whose stationary distribution at $t \to \infty$ is the desired target distribution π and then average the results over many (cheap) iterations of the chain, instead of computing the (expensive) integral over the pdf. This is possible because over a large number n of such Markov chains, the following result will hold,

$\forall \phi$ functions on the states of the chain:

$$\frac{1}{n}\sum_{t=1}^{n}\phi(X_t) \rightarrow \int \phi(x)\pi(x)\mathrm{d}x = E[\phi(X)] \quad \text{as } n \xrightarrow{a.s} \infty$$

Hamiltonian Monte Carlo (HMC)

However, no single MCMC algorithm is suitable for all problems: for some methods such as the random walk Monte Carlo (RWMC) or Gibbs sampler, the choice of step size may prove to be crucial for the quality of the outcome – if too small, the algorithm will take a very large number of steps until the chains converge; likewise, if too large, the chain will fluctuate a lot in the state space between iterations, increasing the approximation error (Hoffman and Gelman, 2011). Out of this family of algorithms, the Hamiltonian Monte Carlo (HMC) approach remarks itself through an increase in both accuracy and speed as it instead relies on choices of steps informed by first-order gradients (Betancourt, 2017; Hoffman and Gelman, 2011). This makes it uniquely suited to high-dimensional problems, such as linear models with many latent variables.

Inspired by Hamiltonian dynamics, this approach assumes that to find the next position θ_m in the Markov chain we first perform a fixed number of Leapfrog substeps L each, one after another starting from the current position θ_{m-1}. The size of each Leapfrog step is ϵ and at the end of the L Leapfrog substeps, we either move the chain to have moved to this new-found final position $\tilde{\theta}$, or stay at θ_{m-1} with a given probability α, which indirectly depends on the first-order gradients at $\tilde{\theta}$ and θ_{m-1} (Hoffman and Gelman, 2011). Below we can examine the formulation of the Leapfrog step in the HMC:

$$\begin{aligned}
\mathbf{p}(t+\frac{\epsilon}{2}) &= \mathbf{p}(t) - \frac{\epsilon}{2}\nabla U(\mathbf{q}(t)) \\
\mathbf{q}(t+\epsilon) &= \mathbf{q}(t) + \epsilon\mathbf{p}(t+\frac{\epsilon}{2}) \\
\mathbf{p}(t+\epsilon) &= \mathbf{p}(t+\frac{\epsilon}{2}) - \frac{\epsilon}{2}\nabla U(\mathbf{q}(t+\epsilon))
\end{aligned}$$

where $\mathbf{p}$ acts as the position and $\mathbf{q}$ the momentum vector of the particle, while $U(\mathbf{q}(t))$ is the potential field in the wider context of Hamiltonian dynamics.

HMC is however not without its problems: bad choices of step size ϵ and number of steps L will decrease the efficiency of the algorithm dramatically, e.g. if L is too small, random walk behavior can be observed, resulting in large computation times; similarly, if L is too large the chain will fluctuate a lot in the state space between iterations, increasing the approximation error or for certain values lose its ergodicity altogether (Hoffman and Gelman, 2011). A solution to this issue comes in the form of the No-U-Turn Hamiltonian Monte Carlo algorithm that bypasses the need to choose the value for L (Hoffman and Gelman, 2011; Betancourt, 2017).

No-U-Turn Sampler (NUTS)

Instead of using a pre-established number L of Leapfrog steps as was the case with classical HMC, the NUTS algorithm relies instead on criterion evaluation – it stops the computation of the next predicted position $\tilde{\theta}$ once we reach the point when subsequent Leapfrog steps will not increase the distance between prediction $\tilde{\theta}$ and the previous position of the chain at θ_{m-1}. Therefore, no L is needed to be chosen to run the NUTS algorithm with good performance (the other parameter on which it depends, the step size ϵ is taken to be adaptive in NUTS, so there is no need to tune it) (Hoffman and Gelman, 2011).

Due to its fast convergence especially for high-dimensional models, NUTS has become the MCMC algorithm of choice for programming languages like Stan (Carpenter *et al.*, 2017). The R package **brms** (Bürkner, 2017) which we use extensively throughout this project is based on Stan, therefore making NUTS the core MCMC simulation method for our Bayesian linear modeling.

Prior and posterior predictive checks

To do modeling in a Bayesian framework, we start with what previously acquired knowledge we have about the parameters of the model we want to fit for the data; this knowledge is collected under the prior distribution over the parameters. Then, we combine it with information we glean from the data (in the form of the likelihood of observing said data under the model's condition) and hence obtain the posterior distribution, which updates our knowledge of good choices of the model parameters.

However, a model created under this framework is only as sensible as its prior. A poor prior can seriously skew our posterior distribution, resulting in a poor parametrisation of the model. Therefore, we conduct predictive checks on both the prior and posterior – that is we check the probability of observing future data points under the model described by the prior (or posterior), which we obtain from the empirical distribution of the data points (i.e. a histogram of the prior or posterior samples). The definitions for these two predictive distributions are

$$p(\tilde{\mathbf{x}}) = \int \mathrm{L}(\tilde{\mathbf{x}}|\theta)\pi(\theta)\mathrm{d}\theta \quad \text{(prior predictive)}$$

$$p(\tilde{\mathbf{x}}|\underline{\mathbf{x}}) = \int \mathrm{L}(\tilde{\mathbf{x}}|\theta,\underline{\mathbf{x}})\pi(\theta|\underline{\mathbf{x}})\mathrm{d}\theta \quad \text{(posterior predictive)}$$

where $\pi(\theta)$ is the prior, $\pi(\theta|\underline{\mathbf{x}})$ the posterior distribution and $\mathrm{L}(\tilde{\mathbf{x}}|\theta,\underline{\mathbf{x}})$ is the likelihood of the future observation $\tilde{\mathbf{x}}$ with respect to parameters θ and previously observed data $\underline{\mathbf{x}}$.

A good prior should produce a predictive distribution that aligns reasonably with a histogram of the observed data. Therefore, careful consideration needs to be put into choosing the prior.

Leave-one-out (LOO) cross-validation

As discussed previously, making a good choice of parametrisation for the model of the data is difficult. From machine learning, we know that once we use the data for model fitting, we would usually use different data to assess the efficacy of the model, hoping to see a small error. This means we need large (and hence expensive) data sets which we split into two: one for our model fitting and one for validation. Also, this gives rise to a different issue: use too little data for the training set and the model will be ill-fitting; use too little for the validation set and the error in model assessment will be large.

Hence, cross-validation becomes necessary for model assessment, using the same data as we do in training to perform the validation efficiently: the package that we use throughout this project, **brms** uses the Leave-One-Out (LOO) cross-validation method (Bürkner, 2017). The LOO method essentially uses validation sets made up by the whole of the data except one data point, after which the results are aggregated to obtain the best parameter choices for the model. This is done so that parameter values are not skewed by low-weight, high-leverage data points. LOO is relatively easy to calculate in a Bayesian model because the model only changes by one evaluation in the likelihood.

Convergence diagnostics

There are several measures we compute to assess the quality of the model we use with the MCMC sampling: these tell us how fast our chains converge, or how many parallel chains need to be run for our Bayesian linear modeling. Below we give their definitions and what they tell us about the convergence quality of the MCMC. These convergence diagnostics are most informative when during their computation we use multiple chains initialized at a wide range of starting points to avoid falsely diagnosing mixing when beginning at a different point, which would lead to a qualitatively different posterior (Vehtari *et al.*, 2021).

Potential scale reduction factor $\hat{R}$

The potential scale reduction factor $\hat{R}$ is computed for each quantity of interest and is the go-to diagnostic for convergence assessment of MCMC in software packages like Stan. Stan uses the so called split-$\hat{R}$ which tells us how well the simulations mix, i.e. the variance of all the chains mixed together should be higher than the variance of individual chains. It is defined as the standard deviation of that quantity from all the chains included together, divided by the root mean square of the separate within-chain standard deviations (Vehtari *et al.*, 2021; Betancourt, 2017), i.e.

$$\hat{R} = \sqrt{\frac{\hat{\text{var}}^{+}(\theta|y)}{W}}$$

where $\hat{\text{var}}^{+}(\theta|y)$ is the estimate of the variance of the parameter θ with respect to the observed data y from the simulated Markov chains, and W is the within-chain variance. The

estimator for var$(\theta|y)$ depends on both the within- (W) and between-chain variance (B) and N the number of draws per chain:

$$\hat{\text{var}}^{+}(\theta|y) = \frac{N-1}{N}W + \frac{1}{N}B$$

If chains have not mixed well (i.e. the between- and within-chain estimates do not agree), the split-$\hat{R}$ is larger than 1. The recommended upper limit of tolerance for this diagnostic quantity is 1.05 (Vehtari *et al.*, 2021).

Effective sample size ESS

Just as the name suggests, the effective sample size (ESS) tells us how many independent draws from the target distribution contain the same amount of information as the dependent sample obtained by the MCMC algorithm to give an equivalent estimator precision (Vehtari *et al.*, 2021; Betancourt, 2017). The ESS is defined as

$$\text{ESS} = \frac{N}{1 + 2\sum_{l=1}^{\infty}\rho_l}$$

where ρ_l is the lag-l autocorrelation of the quantity of interest θ over the history of the Markov chain. Along with $\hat{R}$, it is a standard diagnostic computed to inform on the stability estimate of uncertainty as a measure of the quality of the convergence (Vehtari *et al.*, 2021).

In Stan, we can compute two special types of ESS: the bulk- and tail-ESS. We are focusing on the latter in this project. The tail-ESS is computing the minimum of effective sample sizes for 5% and 95% quantiles and is capturing the effectiveness of estimators in the tail of the distribution for the data, i.e. the extremes (Vehtari *et al.*, 2021).

Controlling divergence of Markov chains

MCMC estimation of parameters is an approximating procedure that includes many runs of the same chain to capture mean behavior. When working with this method, it is important to account for the warm-up period: for each chain, we need to let the Markov Chain run long enough to reach convergence. Therefore, we typically drop the first e.g., 1000 steps of the chain each time because, should we plot the chains, we would see on average that after this first 1000 the chain will seem to settle in its stationary state. Therefore, observing divergent transitions is not uncommon. However, having too many such chains will cause a bias in the obtained posterior samples (Bürkner, 2017). The way the **brms** package deals with this problem is through the *control* input on the NUTS sampler. For example, if we receive a warning about the high number of divergent transitions after the warm-up has been taken out, the appropriate step is to increase the step-size *alpha-delta* argument. This change however comes at the price of speed, as the chain will run slower (Bürkner, 2017). Hence, fine-tuning for this parameter is needed for optimal performance of the NUTS sampling procedure for model fitting.

Proteins vs proteoforms

Standard quantitative proteomics operates under the assumption that peptides of the same protein have the same quantitative response, i.e., there is a single quantity per protein coding gene. However, this is not actually the case. For example the International Human Genome Sequencing Consortium sequenced the human genome in 2001 and found that there are only about 20,000–25,000 protein-coding genes (IHG, 2004). However, this relatively small proteoform is enlarged by post-translational modifications, proteolysis, and alternative splicing. As a result, the roughly 20,000 coding genes can generate more than a million different proteoforms (Kelemen *et al.*, 2013). This means that peptides of the same protein-coding gene can map to different proteoforms with different quantities (Plubell *et al.*, 2021). As a result, genotype-based phenotype inference is more difficult than might originally be thought, and this problem motivates the importance of understanding proteome diversity (Bludau *et al.*, 2021).

Until now, we have been using the term protein rather loosely. Though we can be more precise by using the term proteoform, which refers to a specific final 'form' of a protein, and there can be multiple proteoforms for each protein coding gene. The term proteoform is appealing as it is the protein analog of the genetic term 'isoform' (Smith *et al.*, 2013). As briefly mentioned, there are a few ways for different proteoforms to arise from the same gene. One way, alternative splicing, occurs when different mRNA exons are spliced together in alternative ways, such as in different orders or with exclusions. This enables multiple final mRNA scripts (and therefore proteins) to be produced from the same gene Black (2003). There are also post-translational modifications (PTMs), which are the chemical modification of proteins after they have been synthesized. One highly studied PTM is phosphorylation (Collins *et al.*, 2007), which plays an important role in cell signaling pathways (Ardito *et al.*, 2017). Glycosylation is also important for protein folding and stability (Shental-Bechor and Levy, 2008; Solá and Griebenow, 2009).

While top-down proteomics can identify proteoforms, bottom-up proteomics has a much deeper proteoform coverage and is thus the standard method of choice in differential analysis experiments (Schaffer *et al.*, 2019; Fornelli *et al.*, 2018; Skinner *et al.*, 2017). However, an important challenge facing bottom-up proteomics is that because it can only infer protein identity and intensity from the peptide-level, it is particularly unequipped to resolve different proteoforms (Bludau and Aebersold, 2020). It was shown by Bludau *et al.* (2021) through their introduction of COPF, which stands for COrrelation-based functional ProteoForm assessment, that functional proteoforms can be identified through correlation analysis of bottom-up proteomics data. This is because some peptides belonging to the same protein-coding gene exhibit distinct quantitative patterns across samples or treatment groups. There is another recently developed tool called PeCorA (Dermit *et al.*, 2020), which stands for peptide correlation analysis (of bottom-up proteomics data). This has a linear model with an interaction term, where the interaction is between the peptide and the treatment group. Then, the p-value is recorded for that interaction coefficient. Thus, PeCorA is specifically

designed for detecting proteoforms differing by a single peptide. Bludau *et al.* (2021) were able to show that COPF had better performance for detecting proteoforms differing by multiple peptides. COPF can identify a maximum of two proteoforms for a protein-coding gene. Detecting more proteoforms would seem a far more complex task given the limited amount and noisiness of PSM peptide-level data. A graphical overview of the COPF approach is shown in Figure 6.

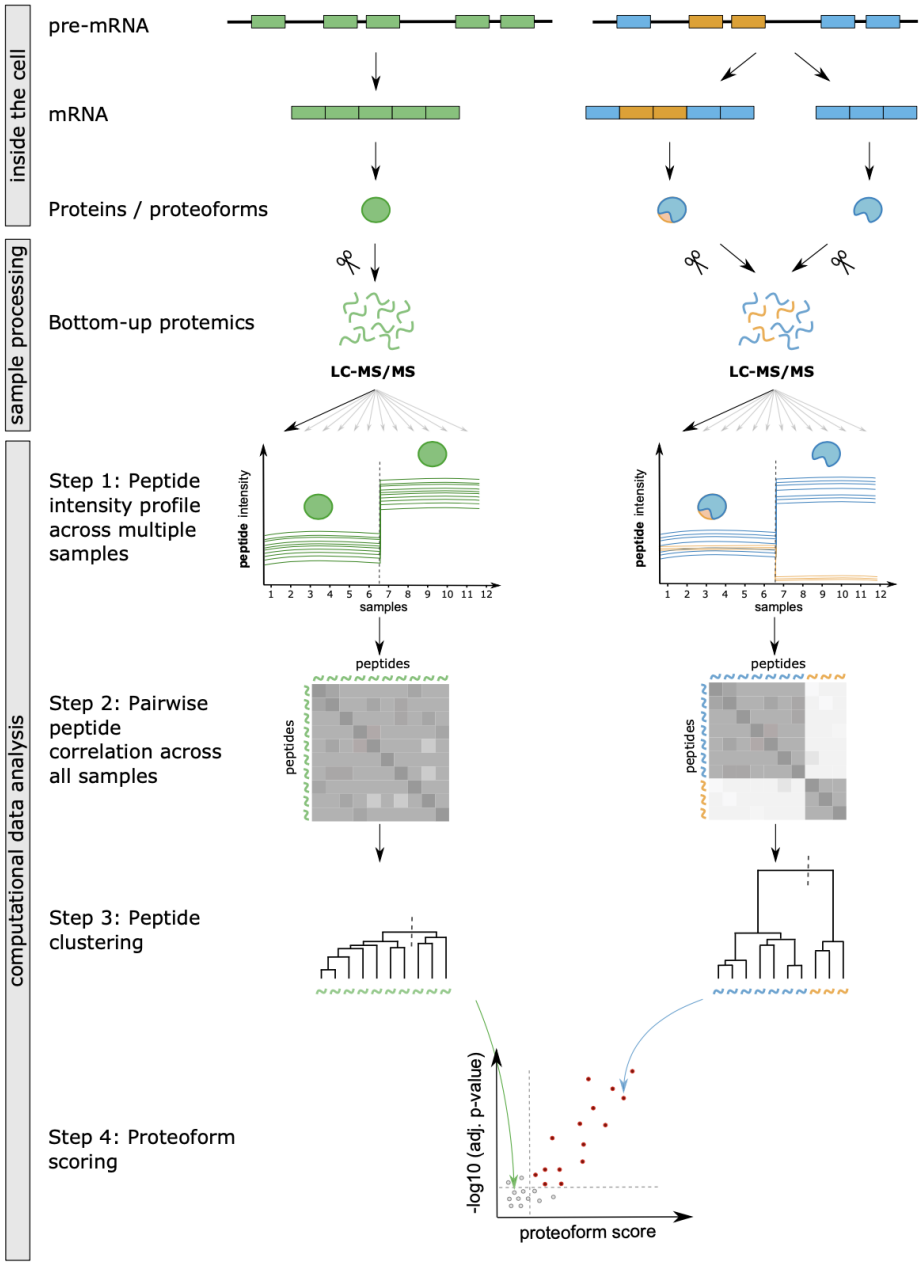


Figure 6: Figure reproduced from Bludau *et al.* (2021) showing a proteoform produced by alternative splicing on the right side.

As discussed by Plubell *et al.* (2021), the problem of proteoforms raises an issue for bottom-up proteomics, which is that all of the protein summarization methods we have discussed combine peptide measurements into a single protein quantity even though there may not be a single proteoform from the expressed gene, and these proteoforms can have different concentrations between treatment groups.

Data exploration

The dataset used here results from MS experiments with PSMs generated by the Mascot database system. The THP-1 monocytic leukaemic cell line is investigated by cell lysates collected at 0, 2, 4, 6, 12, and 24 hour time-points, which are considered to be the treatment groups, where each is labeled by a TMT reporter. The idea is that as monocytes differentiate into macrophages, there may be differences in the abundance of certain proteins over the time intervals and these could suggest differential expression that occurs in this process (Italiani and Boraschi, 2014). The hyperLOPIT method (Mulvey *et al.*, 2017) was used to obtain fractions using ultra performance liquid chromatography and there were three biological replicates taken. The fractions from these three biological replicates were run using the MS3 protocol described earlier. There are no shared peptides, such that every peptide is unique to a protein. The number of peptides per protein and PSMs per peptide are shown in Figures 7 and 8, in relation to the inference problem of figure 1.

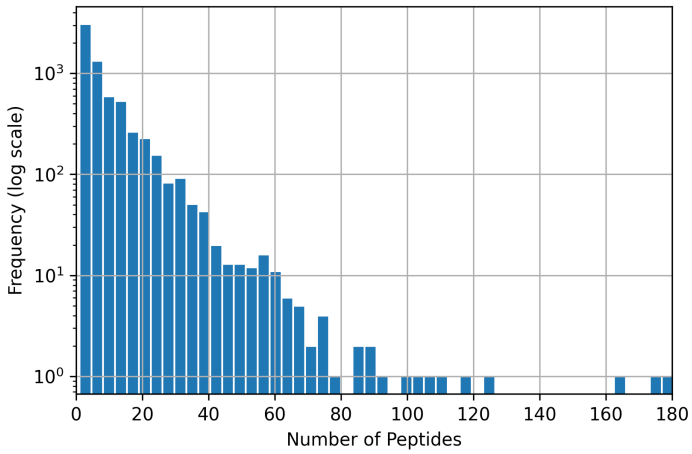


Figure 7: Histogram of number of Peptides per Protein.

New model for protein summarization

In this project, we seek to obtain protein summaries that can take into account the possibility of different proteoforms and enable the protein summarization to be informed by the

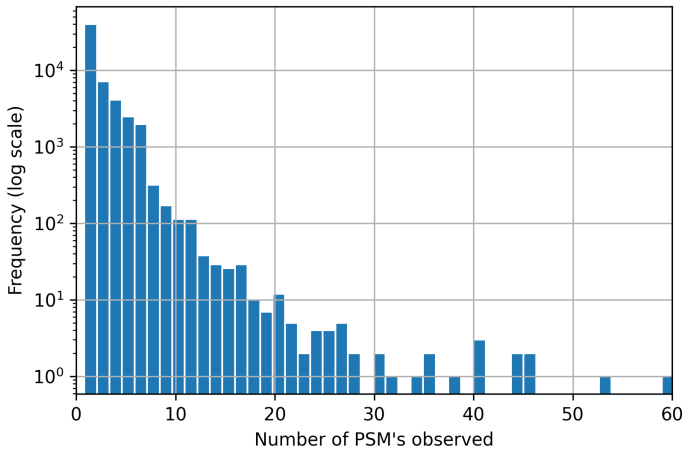


Figure 8: Histogram of number of PSMs per Peptide.

uncertainty metrics described in the earlier section. The approach we have taken to do this is to allow the posterior distribution of protein intensity to arise from the intensities of peptides through a mixed model. The rationale for doing this is that, if there is bi-modality in the peptide intensities, which could arise as a result of proteoforms, as shown in the right-side Step 1 chart in Figure 6. A protein summarization model like the one we are using would not only show this bi-modality in the posterior distribution of the protein, but would also provide the identities of the peptides belonging to the separate modes. Furthermore, the contribution of each peptide to the overall protein summarization can be weighted by its quality or uncertainty. We can do this with the quality metrics, but we would like to check first that the uncertainty metrics of PSMs tend to be more similar within the same peptide, i.e., that they say something meaningful about the uncertainty about the peptide.

The uncertainty metrics we used are PEP, q-value, ion score, DeltaScore, and isolation interference %. We might also call these 'quality' metrics because they are associated with the quality of the PSM measurement. The correlation matrix for these metrics across the entire dataset is given in Figure 9. In this correlation matrix, the negative of the q-Value, PEP, and isolation interference % so that greater values of each uncertainty metric corresponds to more certainty (i.e. greater quality). Notably, all of the uncertainty metrics are positively correlated, and some of these correlations are moderate. Unsurprisingly, q-value and PEP are highly correlated, since they have very similar meanings. Likewise, it is also unsurprising that the ions score (PSM identification score) correlates with q-value and PEP.

We have included the length of the peptide sequence here because it was reported by Fischer and Renard (2015) that peptide sequence length positively correlates with quantification error with Spearman correlation coefficients ranging from 0.17 to 0.39. However, it is observed in Figure 9 that peptide length inversely correlates with PEP and q-value, -0.33 and -0.38, respectively. This means that longer peptide sequences are associated with less

certainty in quantification, but more certainty in identification. Therefore, we decided not to include peptide length as a quality metric in the model.

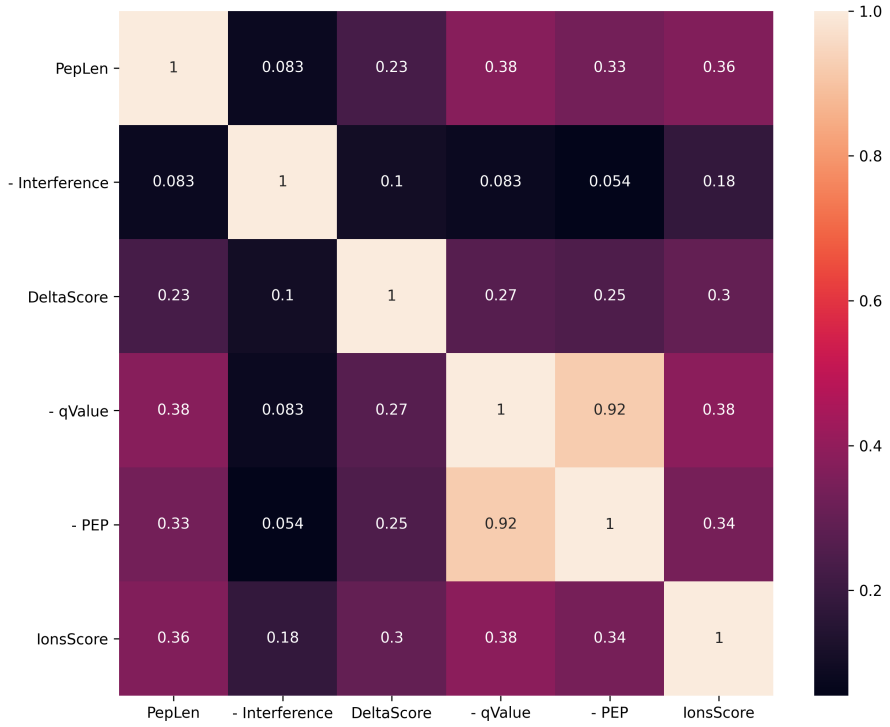


Figure 9: Correlation matrix for uncertainty/quality metrics across the monocytic leukaemic dataset.

It should also be noted that the uncertainty metrics for PSMs of the same peptide tend to be more similar to each other. That is, the PEP values for PSMs of the same peptide tend to be meaningfully similar, so that the quality-metrics can be considered, in part, peptide-based. One way to show this is to take the ratio of the variance between peptides to the variance within peptides, i.e. the between-group variance to the within-group variance, where the group is the peptide. This is also the conceptual basis for the F test in analysis of variance (ANOVA). We do this for each of the quality metrics. Specifically, the variance of the mean for each peptide divided by the mean of the variance for each peptide ranges from 2.8 for isolation interference % to 13 for DeltaScore. The ratios are 3.4 and 4.0 for PEP and q-value, respectively. This tells us that we can use the quality metrics for the PSMs of a peptide to describe the overall quality of the peptide.

The model we are introducing here is an empirical Bayes model, which aims to solve the protein summarization inference problem shown in Figure 1 with greater consideration of the uncertainties in identification and quantification and the proteoform problem that we have discussed. The first step of this model is hyperparameter estimation. This is done with the following linear model for a particular protein in a particular treatment group

$$y \sim \mu + u_{pep} + u_{rep} \tag{14}$$

where u_{pep} is the random effect for peptides and u_{rep} is a random effect for a biological or technical replicate, which have the following generating models:

$$u_{pep} \sim N(0, \sigma^2_{pep}) \tag{15}$$

$$u_{rep} \sim N(0, \sigma^2_{rep}) \ . \tag{16}$$

Recall that for our dataset, the only replicates are three biological replicates, so we include only one u_{rep} term. The rationale for treating the peptide as a random effect is so that two important values can be estimated. These are $\hat{\mu}$, the estimate of the protein intensity and $\hat{\sigma}^2_{pep}$, which comes from the prior on the random effect. These two hyperparameters are used for the prior of the peptide effect

$$\beta_{pep} \sim N(\hat{\mu}, \hat{\sigma}^2_{pep}) \tag{17}$$

Then, the following linear observation model is fit

$$y \sim \beta_{pep} + u_{rep} \tag{18}$$

and posterior samples are taken of $\beta_{pep}|y$. Finally, the protein summarization is made through the mixture model

$$P(\beta_{protein}|y) = \Sigma_{j=1}^{K} \pi_j \, P(\beta_{pep,j}|y) \tag{19}$$

where $\beta_{pep,j}$ refers to the latent intensity of peptide j and there are K peptides observed for this protein. Naively, we might have $\pi_j = 1/K$ for all j if we believed each peptide was an equally reliable representative of the protein. However, we know this is not the case, instead we would want peptides with less uncertainty (i.e. higher quality measured peptides) to contribute more to the overall protein summarization. Thus we can we can construct a model for the weights π_j that uses the sigmoid function

$$\pi_j \propto S(s(l + \gamma_j)) = \frac{1}{1 + e^{-s(l+\gamma_j)}} \tag{20}$$

where $S(s(l + \gamma_j))$ is the sigmoid function with scale parameter s and location parameter l, and with the constraint that

$$\Sigma_{j=1}^{K} \pi_j = 1 \ . \tag{21}$$

We can think of the sigmoid function as telling us the proportion or percentage of the posterior samples from $P(\beta_{pep,j}|y)$ that we use, as each peptide will have the same number of posterior samples. We need to relate γ_j to the uncertainty metrics described earlier: Isolation interference %, DeltaScore, qValue, PEP, and Ion Score. One way to do this is through a linear combination of coefficients relating to these uncertainty matrices, where for a particular peptide j

$$\gamma_j = \hat{\beta}_{\text{Interference},j} + \hat{\beta}_{\text{DeltaScore},j} + \hat{\beta}_{\text{qValue},j} + \hat{\beta}_{\text{PEP},j} + \hat{\beta}_{\text{IonScore},j} \tag{22}$$

and these coefficients are estimated by simple normal linear models, for example the DeltaScore coefficient for peptide j is estimated by the following model

$$y_{\text{DeltaScore},j} = \beta_{\text{DeltaScore},j} + \epsilon$$

where $y_{DeltaScore,j}$ is the DeltaScore of a PSM of peptide j and ϵ is normally distributed with mean zero. We know from normal linear summarization with one term that $\beta_{DeltaScore,j}$ is simply estimated by the mean of the DeltaScores for peptide j ($\bar{y}_{DeltaScore,j}$). Note that we need to flip the signs for isolation interference %, qValue, and PEP, as mentioned earlier, so that we fit, for the example of qValue,

$$y_{\text{qValue},j} = -\beta_{\text{qValue},j} + \epsilon \ .$$

In order to ensure that the coefficients in Equation 22 are on the same scale, we standardize and scale the uncertainty metrics to have mean 0 and variance 1.

This approach ensures that low-quality peptides, where we have uncertainty in identification or quantification contribute less to the protein summarization because of their lower γ score. Furthermore, this approach makes sense in light of the proteoform problem. Consider the case that all of the peptides in the sample belong to the same proteoform, any of the samples on the left-side chart in Step 1 of Figure 6. Then, we would expect peptide intensities to follow the generating model

$$\beta_{pep} \sim N(\mu, \sigma^2_{pep})$$

implied in Equation 14. If this is the true generating model for β_{pep}, then the mixture of the peptide distributions

$$\Sigma_{j=1}^{K} \pi_j \, N(\mu, \sigma^2_{pep}) = N(\mu, \sigma^2_{pep}) \Sigma_{j=1}^{K} \pi_j = N(\mu, \sigma^2_{pep})$$

and we get that the protein distribution is normally distributed centered at μ. Then, in the initial hyperparameter summarization from Equation 14 the value μ would accurately estimate the protein intensity. On the other hand, if we have a case of proteoforms, then $\beta_{pep,j}$ for some peptides j would be generated one distribution with one mean, and $\beta_{pep,j}$ for other peptides j would be generated from another distribution with a different mean. The resulting protein distribution would then be bimodal, and its mean would not be equal to the MAP estimate, as would be the case for a unimodal distribution. Further, would know the identities of the peptides that contribute to the modes, so that we could identify which peptides belong to the respective proteoforms (or possible proteoforms).

The hyperparameter model and observation model in Equations 14 and 18, respectively, were left intentionally vague as to what distribution would be used for the likelihood. We have discussed the normal and Laplace likelihoods, but another option is the t-distribution, which has the advantage of being more robust to outliers due to its heavier tail than the normal distribution (Gelman, 2014).

Application of the model

We applied the model to the protein P14324 for hours 0, 2 and 4 in the monocytic leukaemic dataset. This protein did not have differential abundance over time. This dataset shows bi-modal PSMs, as shown in Figure 13. The hyperparameter model (Equation 14) was fit with a normal likelihood, for simplicity. For this model, a flat prior was used for the μ parameter and we used half student-t priors with 3 degrees of freedom and scale parameters of 2.5 for the standard deviation (SD) of the peptide and replicate random effects (σ_{pep} and σ_{rep}), as well as for SD of the errors (σ). The half student-t distribution is the special case of a folded student-t distribution centered at 0 (Psarakis and Panaretoes, 1990). The MCMC NUTS sampler used four chains, each with 8,000 iterations, 2,000 of which were for warmup. The posterior samples of the hyperparameter model are plotted in 10, and the R-output is given below. We can see that the chains converged, because the $\hat{R}$ values were equal to 1. The estimates given in the R-output are the posterior means (i.e. the sample mean of the posterior samples). In **brms**, random effects are called group-level effects and fixed effects are called population-level effects.

```
Group-Level Effects:
~Peptide (Number of levels: 16)
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)     0.90      0.21     0.59     1.38 1.00     4389     5546

~Replicate (Number of levels: 3)
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)     1.12      0.72     0.33     3.11 1.00     1103      397

Population-Level Effects:
          Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
Intercept     5.40      0.73     3.67     6.89 1.00     1266      352

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.82      0.04     0.74     0.91 1.00     5654     2765
```

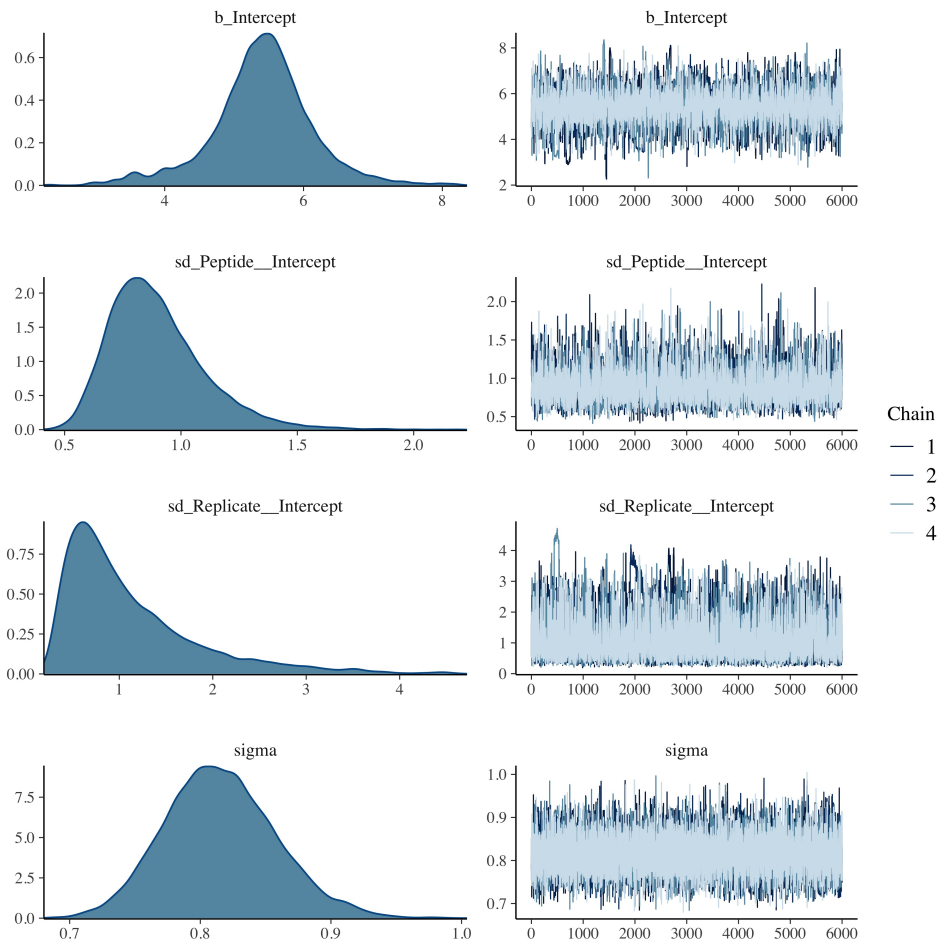


Figure 10: Posterior samples of the hyperparameter model, given in Equation 14, with a normal likelihood. From top to bottom the samples are of μ, σ_{pep}, σ_{rep}, and σ.

The estimates of interest to us are the intercept (population-level effect) and the SD of the peptide (group-level effect). We then use these as the hyperparameters in Equation 17. Then, we fit the linear observation model in Equation 18 using the normal distribution again for convenience. The R output for this model is given directly below. Note the large tail ESS for the estimates of the peptides.

```
Group-Level Effects:
~Replicate (Number of levels: 3)
              Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sd(Intercept)     0.84      0.51     0.29     2.30 1.00     3532     1903

Population-Level Effects:
                                Estimate Est.Error l-95% CI u-95% CI Rhat
PeptideaTPEQYQILk                   5.44      0.31     4.83     6.04 1.00
```

```
PeptideaTPEQYQILkENYGQk                    6.28      0.30     5.68     6.88 1.00
PeptidecSWLVVQcLQR                         5.11      0.29     4.54     5.67 1.00
PeptideeVLEYNAIGGk                         5.61      0.25     5.11     6.11 1.00
PeptidegLTVVVAFR                           4.97      0.43     4.12     5.82 1.00
PeptidegQIcWYQkPGVGLDAINDANLLEAcIYR        4.79      0.26     4.27     5.31 1.00
PeptideiGTDIQDNk                           5.71      0.26     5.20     6.23 1.00
PeptideiGTDIQDNkcSWLVVQcLQR                6.20      0.35     5.52     6.88 1.00
PeptidekQDADSLQR                           6.48      0.44     5.61     7.35 1.00
PeptidelkEVLEYNAIGGk                       5.69      0.23     5.23     6.14 1.00
PeptidemNGDQNSDVYAQEk                      5.69      0.35     5.00     6.37 1.00
PeptideqDADSLQR                            6.67      0.44     5.81     7.51 1.00
PeptideqDFVQHFSQIVR                        4.82      0.26     4.31     5.32 1.00
PeptiderGQIcWYQkPGVGLDAINDAnLLEAcIYR       4.79      0.35     4.12     5.48 1.00
PeptidevLTEDEmGHPEIGDAIAR                  4.23      0.23     3.78     4.68 1.00
PeptidevLTEDEMGHPEIGDAIAR                  4.06      0.25     3.58     4.54 1.00
                                        Bulk_ESS Tail_ESS
PeptideaTPEQYQILk                           8116    13881
PeptideaTPEQYQILkENYGQk                     8987    10934
PeptidecSWLVVQcLQR                          7541     8687
PeptideeVLEYNAIGGk                          6276     8116
PeptidegLTVVVAFR                           14528    14705
PeptidegQIcWYQkPGVGLDAINDANLLEAcIYR         7261    12374
PeptideiGTDIQDNk                            6819     8491
PeptideiGTDIQDNkcSWLVVQcLQR                10331    12975
PeptidekQDADSLQR                           12410     9744
PeptidelkEVLEYNAIGGk                        5952    11068
PeptidemNGDQNSDVYAQEk                       9767    14439
PeptideqDADSLQR                             9408     5404
PeptideqDFVQHFSQIVR                         6800    12330
PeptiderGQIcWYQkPGVGLDAINDAnLLEAcIYR        9107    13472
PeptidevLTEDEmGHPEIGDAIAR                   5625    11525
PeptidevLTEDEMGHPEIGDAIAR                   6949    11196

Family Specific Parameters:
      Estimate Est.Error l-95% CI u-95% CI Rhat Bulk_ESS Tail_ESS
sigma     0.81      0.04     0.74     0.90 1.00    12420     7090
```

The posterior samples of the 16 peptides are shown in Figure 11. There do not appear to be two isolated modes as occurs in Figure 6 as we've described earlier. That is, the log-2 intensities don't follow an S shape, with one cluster at a lower intensity and another cluster at a higher intensity. However, if this were the case, we would be able to see it and

immediately identify which peptides belonged to which mode.

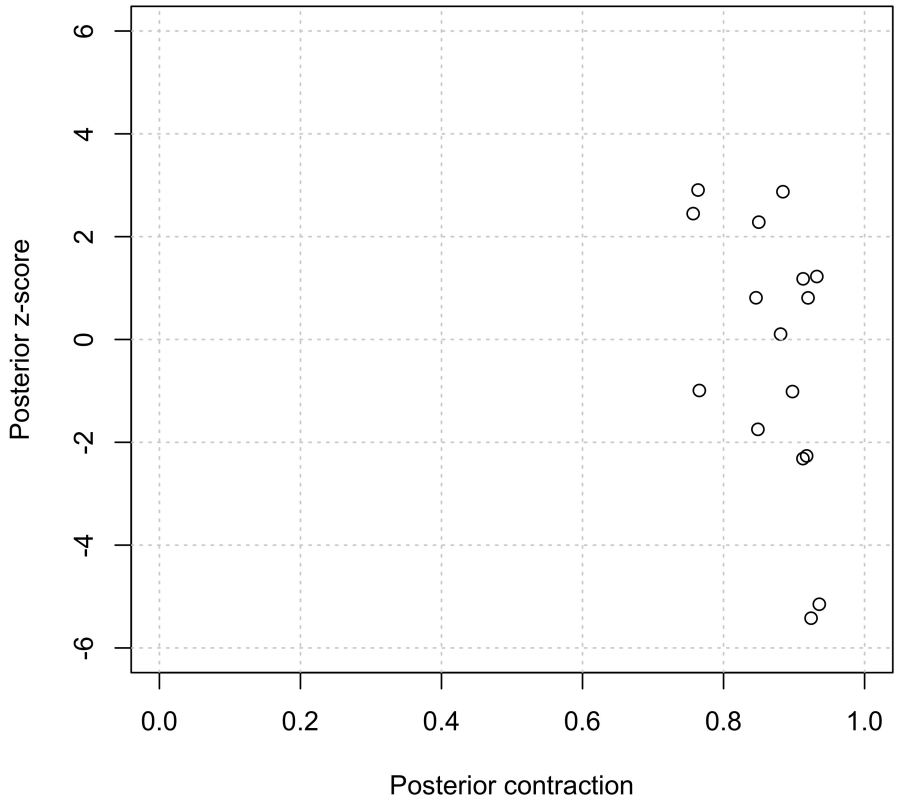


Figure 11: Boxplot of the posterior samples of the peptide intensities. The identities of the peptides cannot be shown here as the sequences are too long, though they are listed in the R output earlier.

The S term in Equation 20 is shown in Figure 12. As mentioned earlier, these give the proportion of posterior samples from each peptide to include. The location (l) and scale (s) parameters adjust the relationship between γ and S, such that a larger location parameter shifts the points to the right along the function and thus increases the S value for every peptide. The scale parameter s affects the scale of the sigmoid function about 0, such that a smaller s makes the relationship increasingly linear, and a very small s brings every S value to 0.5. Likewise, a very large s causes the S values to jump from 0 to 1 around $\gamma = 0$. We used $l = s = 0.5$, though there is clearly room for flexibility here.

Kernel density estimates are shown in Figure 13, which show that that the typical protein summarization, given by the hyperparameter model does not give any indication of the multiple modes present in the PSM intensities. Though not shown here, the posterior predictive samples of the hyperparameter model are unimodal and fail to capture the bi-modality of

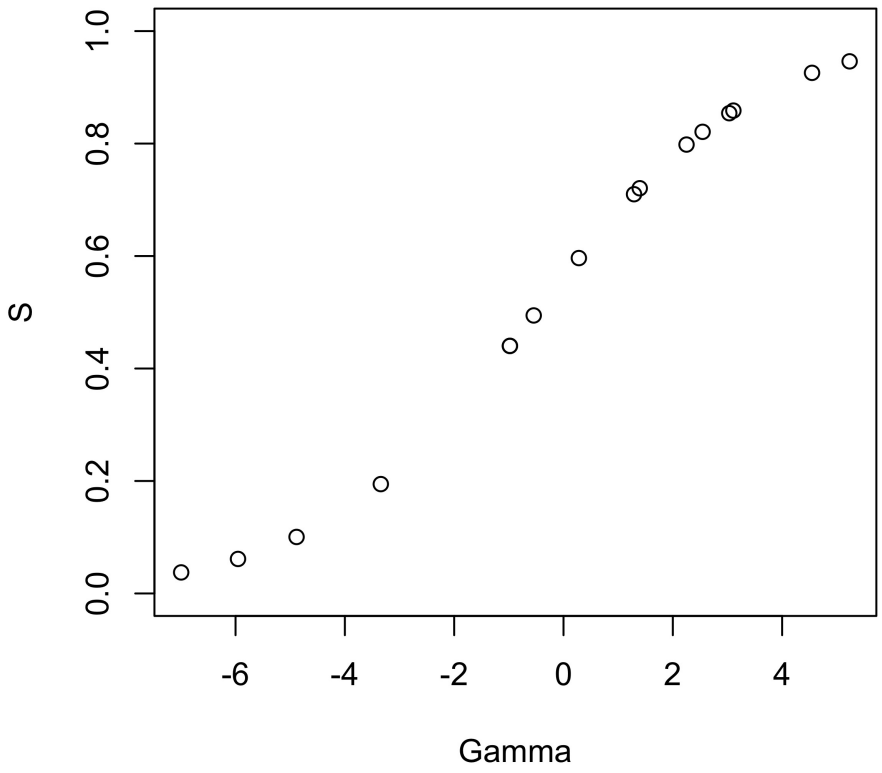


Figure 12: Plot of γ_j vs $S(s(l+\gamma_j))$ for the 16 peptides (see Equations 22 and 20). For each peptide (represented by a circle), this plot shows the relation between the quality of its PSM measurements and its contribution to the final protein summarization.

the actual PSM intensities. On the other hand, the summarization given by the model (in green) has a much wider distribution with multiple modes. It is also interesting that one of the modes from the even mixture (around 6) falls away in the weighted mixture, which means that the PSMs for the peptide(s) contributing to it were of low quality.

Conclusions and future work

We have introduced a model that attempts to answer the problem raised by Plubell *et al.* (2021) about the problem of aggregating PSM intensities into a single value when there can be multiple proteoforms present. We have also attempted to include quantifications of uncertainty in both identification and quantification in our protein summarization model. It is also worth mentioning that the simple protein summarizations described can give different summaries depending on the missingness, and distribution of the intensities. Examples of

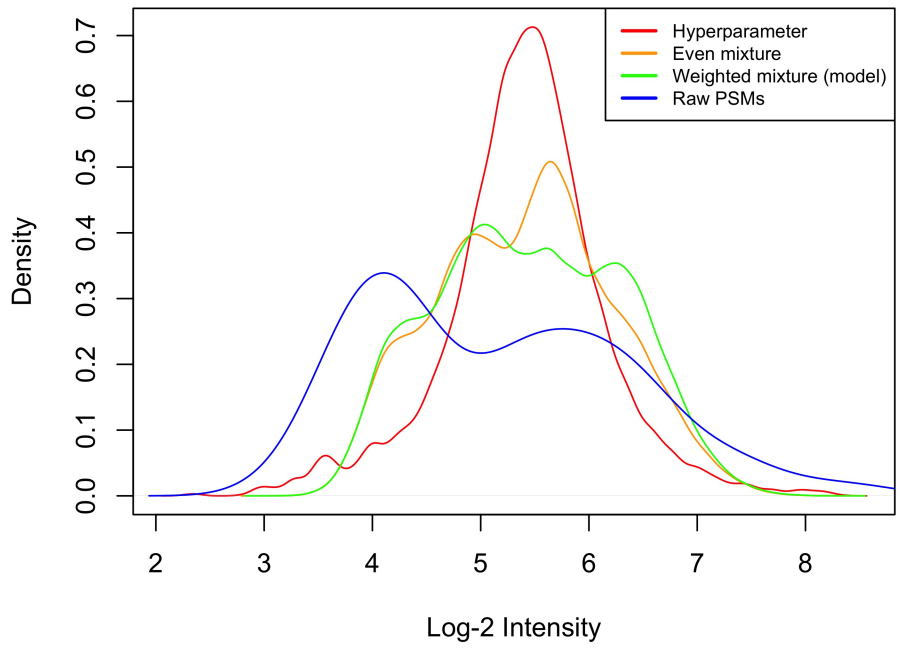


Figure 13: Kernel density estimates for the protein summary μ in the hyperparameter model (Equation 14), an even mixture of peptide intensities (where $\pi_j = \frac{1}{K} \forall j$), and the weighted mixture of the proposed model (shown in Figure 12), and finally for all the log-2 PSM intensities belonging to the protein.

these are shown in the appendix, but not discussed here for space constraints.

In future work, it would be productive to use this kind of model to investigate differential abundance analysis and its ability to identify proteoforms. One immediate application would be to apply this model to proteoform examples investigated by Bludau *et al.* (2021). Additionally, different likelihood models and priors can be used for the hyperparameter estimation and model fitting, as discussed earlier. It would also be advantageous to develop a systematic method for setting s and l, such as ensuring that the maximum and minimum S values are two quantities.

We could imagine a scenario for our model of two distinct modes in the peptide intensities, but where one of two modes consisted of highly uncertain peptides. In which case only the other mode would remain for the final protein summarization of our model. In fact, this case corresponds to a situation where Figures 11 and 12 highly correlate and are both shaped like a sharp S, that is, relatively flat on the left, sharp jump up, then flat on the right. More generally, our model has the capability of removing the effect of an outlying peptide with low-quality measurements that would otherwise distort the final protein summarization.

References

(2004). Finishing the euchromatic sequence of the human genome. *Nature*, **431**(7011), 931–945.

(2014). ionization efficiency. In *The IUPAC Compendium of Chemical Terminology*. International Union of Pure and Applied Chemistry (IUPAC).

Adamo, M. E. et al. (2016). Tempest: Accelerated MS/MS database search software for heterogeneous computing platforms. *Current Protocols in Bioinformatics*, **55**(1).

Altelaar, A. M. et al. (2013). Benchmarking stable isotope labeling based quantitative proteomics. *Journal of Proteomics*, **88**, 14–26.

Anderle, M. et al. (2004). Quantifying reproducibility for differential proteomics: noise analysis for protein liquid chromatography-mass spectrometry of human serum. *Bioinformatics*, **20**(18), 3575–3582.

Ardito, F. et al. (2017). The crucial role of protein phosphorylation in cell signaling and its use as targeted therapy (review). *International Journal of Molecular Medicine*, **40**(2), 271–280.

Arntzen, M. Ø. et al. (2010). IsobariQ: Software for isobaric quantitative proteomics using IPTL, iTRAQ, and TMT. *Journal of Proteome Research*, **10**(2), 913–920.

Bantscheff, M. et al. (2007). Quantitative mass spectrometry in proteomics: a critical review. *Analytical and Bioanalytical Chemistry*, **389**(4), 1017–1031.

Bates, D. et al. (2015). Fitting linear mixed-effects models usinglme4. *Journal of Statistical Software*, **67**(1).

Bayes, T. (1763). An essay towards solving a problem in the doctrine of chances. *Phil. Trans. of the Royal Soc. of London*, **53**, 370–418.

Benjamini, Y. et al. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological)*, **57**(1), 289–300.

Berberich, M. J. et al. (2018). MS3-IDQ: Utilizing MS3 spectra beyond quantification yields increased coverage of the phosphoproteome in isobaric tag experiments. *Journal of Proteome Research*, **17**(4), 1741–1747.

Betancourt, M. (2017). A conceptual introduction to hamiltonian monte carlo.

Black, D. L. (2003). Mechanisms of alternative pre-messenger RNA splicing. *Annual Review of Biochemistry*, **72**(1), 291–336.

Blitzstein, J. (2019). *Introduction to probability*. CRC Press, Boca Raton.

Bludau, I. et al. (2020). Author correction: Proteomic and interactomic insights into the molecular basis of cell functional diversity. *Nature Reviews Molecular Cell Biology*, **21**(6), 353–353.

Bludau, I. et al. (2021). Systematic detection of functional proteoform groups from bottom-up proteomic datasets. *Nature Communications*, **12**(1).

Boneau, C. A. (1960). The effects of violations of assumptions underlying the t test. *Psychological Bulletin*, **57**(1), 49–64.

Bornn, L. et al. (2010). Grouping priors and the bayesian elastic net.

Breckels, L. M. et al. (2018). A bioconductor workflow for processing and analysing spatial proteomics data. *F1000Research*, **5**, 2926.

Breitwieser, F. P. et al. (2011). General statistical modeling of data from protein relative expression isobaric tags. *Journal of Proteome Research*, **10**(6), 2758–2766.

Bürkner, P.-C. (2017). brms: An r package for bayesian multilevel models using stan. *Journal of Statistical Software*, **80**(1).

Carpenter, B. et al. (2017). Stan: A probabilistic programming language. *Journal of Statistical Software*, **76**(1).

Carrillo, B. et al. (2009). Methods for combining peptide intensities to estimate relative protein abundance. *Bioinformatics*, **26**(1), 98–103.

Chahrour, O. et al. (2015). Stable isotope labelling methods in mass spectrometry-based quantitative proteomics. *Journal of Pharmaceutical and Biomedical Analysis*, **113**, 2–20.

Chen, S. et al. (2014). Proteomic identification of differentially expressed proteins associated with the multiple drug resistance in methotrexate-resistant human breast cancer cells. *International Journal of Oncology*, **45**(1), 448–458.

Chiva, C. et al. (2014). Influence of the digestion technique, protease, and missed cleavage peptides in protein quantitation. *Journal of Proteome Research*, **13**(9), 3979–3986.

Choi, M. et al. (2014). MSstats: an r package for statistical analysis of quantitative mass spectrometry-based proteomic experiments. *Bioinformatics*, **30**(17), 2524–2526.

Clough, T. et al. (2009). Protein quantification in label-free LC-MS experiments. *Journal of Proteome Research*, **8**(11), 5275–5284.

Colaert, N. et al. (2011). thermo-msf-parser: An open source java library to parse and visualize thermo proteome discoverer msf files. *Journal of Proteome Research*, **10**(8), 3840–3843.

Collins, M. O. et al. (2007). Analysis of protein phosphorylation on a proteome-scale. *PROTEOMICS*, **7**(16), 2751–2768.

Cox, J. et al. (2008). MaxQuant enables high peptide identification rates, individualized p.p.b.-range mass accuracies and proteome-wide protein quantification. *Nature Biotechnology*, **26**(12), 1367–1372.

Craig, R. et al. (2004). TANDEM: matching proteins with tandem mass spectra. *Bioinformatics*, **20**(9), 1466–1467.

Crook, O. M. et al. (2018). A bayesian mixture modelling approach for spatial proteomics. *PLOS Computational Biology*, **14**(11), e1006516.

Crook, O. M. et al. (2019). A bioconductor workflow for the bayesian analysis of spatial proteomics. *F1000Research*, **8**, 446.

de Menezes, D. et al. (2021). A review on robust m-estimators for regression analysis. *Computers & Chemical Engineering*, **147**, 107254.

Dermit, M. et al. (2020). Peptide correlation analysis (PeCorA) reveals differential proteoform regulation. *Journal of Proteome Research*, **20**(4), 1972–1980.

Diz, A. P. et al. (2011). Multiple hypothesis testing in proteomics: A strategy for experimental work. *Molecular & Cellular Proteomics*, **10**(3), M110.004374.

Dunn, O. J. (1961). Multiple comparisons among means. *Journal of the American Statistical Association*, **56**(293), 52–64.

Elias, J. E. et al. (2007). Target-decoy search strategy for increased confidence in large-scale protein identifications by mass spectrometry. *Nature Methods*, **4**(3), 207–214.

Eng, J. K. et al. (1994). 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*, **5**(11), 976–989.

Farebrother, R. W. (1976). Further results on the mean square error of ridge regression. *Journal of the Royal Statistical Society. Series B (Methodological)*, **38**(3), 248–250.

Fischer, M. et al. (2015). iPQF: a new peptide-to-protein summarization method using peptide spectra characteristics to improve protein quantification. *Bioinformatics*, **32**(7), 1040–1047.

Fornelli, L. et al. (2018). Top-down proteomics: Where we are, where we are going? *Journal of Proteomics*, **175**, 3–4.

Frank, A. M. (2009). A ranking-based scoring function for peptide-spectrum matches. *Journal of Proteome Research*, **8**(5), 2241–2252.

Gatto, L. et al. (2011). MSnbase-an r/bioconductor package for isobaric tagged mass spectrometry data visualization, processing and quantitation. *Bioinformatics*, **28**(2), 288–289.

Geladaki, A. et al. (2019). Combining LOPIT with differential ultracentrifugation for high-resolution spatial proteomics. *Nature Communications*, **10**(1).

Gelman, A. (2014). *Bayesian data analysis*. CRC Press, Boca Raton.

Gentleman, R. C. et al. (2004). *Genome Biology*, **5**(10), R80.

Gerster, S. et al. (2014). Statistical approach to protein quantification. *Molecular & Cellular Proteomics*, **13**(2), 666–677.

Gierlinski, M. et al. (2018). Proteus: an r package for downstream analysis of MaxQuant output.

Goeminne, L. et al. (2016). Peptide-level robust ridge regression improves estimation, sensitivity, and specificity in data-dependent quantitative label-free shotgun proteomics. *Molecular & Cellular Proteomics*, **15**(2), 657–668.

Goeminne, L. J. E. et al. (2015). Summarization vs peptide-based models in label-free quantitative proteomics: Performance, pitfalls, and data analysis guidelines. *Journal of Proteome Research*, **14**(6), 2457–2465.

Goeminne, Ludger (2019). *Statistical methods for differential proteomics at peptide and protein level*. Ph.D. thesis, Ghent University.

Goodfellow, I. et al. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Gorshkov, V. et al. (2015). SuperQuant: A data processing approach to increase quantitative proteome coverage. *Analytical Chemistry*, **87**(12), 6319–6327.

Gygi, S. P. et al. (1999). Quantitative analysis of complex protein mixtures using isotope-coded affinity tags. *Nature Biotechnology*, **17**(10), 994–999.

Han, X. et al. (2008). Mass spectrometry for proteomics. *Current Opinion in Chemical Biology*, **12**(5), 483–490.

Hastie, T. et al. (2009). *The Elements of Statistical Learning*. Springer New York.

Hoffman, M. D. et al. (2011). The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo.

Houel, S. et al. (2010). Quantifying the impact of chimera MS/MS spectra on peptide identification in large-scale proteomics studies. *Journal of Proteome Research*, **9**(8), 4152–4160.

Huang, T. et al. (2012). Protein inference: a review. *Briefings in Bioinformatics*, **13**(5), 586–614.

Huang, T. et al. (2020). MSstatsTMT: Statistical detection of differentially abundant proteins in experiments with isobaric labeling and multiple mixtures. *Molecular & Cellular Proteomics*, **19**(10), 1706–1723.

Huber, W. et al. (2002). Variance stabilization applied to microarray data calibration and to the quantification of differential expression. *Bioinformatics*, **18**(Suppl 1), S96–S104.

Italiani, P. et al. (2014). From monocytes to m1/m2 macrophages: Phenotypical vs. functional differentiation. *Frontiers in Immunology*, **5**.

Käll, L. et al. (2007). Semi-supervised learning for peptide identification from shotgun proteomics datasets. *Nature Methods*, **4**(11), 923–925.

Käll, L. et al. (2008). Posterior error probabilities and false discovery rates: Two sides of the same coin. *Journal of Proteome Research*, **7**(1), 40–44.

Karp, N. A. et al. (2010). Addressing accuracy and precision issues in iTRAQ quantitation. *Molecular & Cellular Proteomics*, **9**(9), 1885–1897.

Karpievitch, Y. et al. (2009). A statistical framework for protein quantitation in bottom-up MS-based proteomics. *Bioinformatics*, **25**(16), 2028–2034.

Karpievitch, Y. V. et al. (2012). Normalization and missing value imputation for label-free LC-MS analysis. *BMC Bioinformatics*, **13**(S16).

Kelemen, O. et al. (2013). Function of alternative splicing. *Gene*, **514**(1), 1–30.

Li, J. et al. (2021). TMTpro-18plex: The expanded and complete set of TMTpro reagents for sample multiplexing. *Journal of Proteome Research*, **20**(5), 2964–2972.

Li, Y. F. et al. (2009). A bayesian approach to protein inference problem in shotgun proteomics. *Journal of Computational Biology*, **16**(8), 1183–1193.

Li, Z. et al. (2012). Systematic comparison of label-free, metabolic labeling, and isobaric chemical labeling for quantitative proteomics on LTQ orbitrap velos. *Journal of Proteome Research*, **11**(3), 1582–1590.

Liu, M. et al. (2020). Proper imputation of missing values in proteomics datasets for differential expression analysis. *Briefings in Bioinformatics*, **22**(3).

Liu, N. Q. et al. (2013). Quantitative proteomic analysis of microdissected breast cancer tissues: Comparison of label-free and SILAC-based quantification with shotgun, directed, and targeted MS approaches. *Journal of Proteome Research*, **12**(10), 4627–4641.

Maccoll†, A. (1999). Mass spectrometry, historical perspective. In *Encyclopedia of Spectroscopy and Spectrometry*, pages 1241–1248. Elsevier.

Mallikarjun, V. et al. (2020). BayesENproteomics: Bayesian elastic nets for quantification of peptidoforms in complex samples. *Journal of Proteome Research*, **19**(6), 2167–2184.

Marginean, I. et al. (2008). Analytical characterization of the electrospray ion source in the nanoflow regime. *Analytical Chemistry*, **80**(17), 6573–6579.

McAlister, G. C. et al. (2014). MultiNotch MS3 enables accurate, sensitive, and multiplexed detection of differential expression across cancer cell line proteomes. *Analytical Chemistry*, **86**(14), 7150–7158.

Merrill, A. E. et al. (2014). NeuCode labels for relative protein quantification. *Molecular & Cellular Proteomics*, **13**(9), 2503–2512.

Metropolis, N. et al. (1949). The monte carlo method. *Journal of the American Statistical Association*, **44**(247), 335–341.

Michalski, A. et al. (2011). More than 100, 000 detectable peptide species elute in single shotgun proteomics runs but the majority is inaccessible to data-dependent LC-MS/MS. *Journal of Proteome Research*, **10**(4), 1785–1793.

Mosteller, F. et al. (1977). *Data analysis and regression : a second course in statistics*. Addison-Wesley Pub. Co, Reading, Mass.

Mulvey, C. M. et al. (2017). Using hyperLOPIT to perform high-resolution mapping of the spatial proteome. *Nature Protocols*, **12**(6), 1110–1135.

Nowlan, S. J. et al. (1992). Simplifying neural networks by soft weight-sharing. *Neural Computation*, **4**(4), 473–493.

O'Brien, J. J. et al. (2018). The effects of nonignorable missing data on label-free mass spectrometry proteomics experiments. *The Annals of Applied Statistics*, **12**(4).

O'Connell, J. D. et al. (2018). Proteome-wide evaluation of two common protein quantification methods. *Journal of Proteome Research*, **17**(5), 1934–1942.

Ong, S.-E. et al. (2002). Stable isotope labeling by amino acids in cell culture, SILAC, as a simple and accurate approach to expression proteomics. *Molecular & Cellular Proteomics*, **1**(5), 376–386.

Ow, S. Y. et al. (2011). Minimising iTRAQ ratio compression through understanding LC-MS elution dependence and high-resolution HILIC fractionation. *PROTEOMICS*, **11**(11), 2341–2346.

Pappireddi, N. et al. (2019). A review on quantitative multiplexed proteomics. *ChemBioChem*, **20**(10), 1210–1224.

Perkins, D. N. et al. (1999). Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis*, **20**(18), 3551–3567.

Plubell, D. L. et al. (2021). Can we put humpty dumpty back together again? what does protein quantification mean in bottom-up proteomics?

Psarakis, S. et al. (1990). The folded t distribution. *Communications in Statistics - Theory and Methods*, **19**(7), 2717–2734.

Riley, N. M. et al. (2015). The negative mode proteome with activated ion negative electron transfer dissociation (AI-NETD). *Molecular & Cellular Proteomics*, **14**(10), 2644–2660.

Sandberg, A. et al. (2014). Quantitative accuracy in mass spectrometry based proteomics of complex samples: The impact of labeling and precursor interference. *Journal of Proteomics*, **96**, 133–144.

Savitski, M. M. et al. (2011). Confident phosphorylation site localization using the mascot delta score. *Molecular & Cellular Proteomics*, **10**(2), S1–S12.

Savitski, M. M. et al. (2013). Measuring and managing ratio compression for accurate iTRAQ/TMT quantification. *Journal of Proteome Research*, **12**(8), 3586–3598.

Schad, D. J. et al. (2021). Toward a principled bayesian workflow in cognitive science. *Psychological Methods*, **26**(1), 103–126.

Schaffer, L. V. et al. (2019). Identification and quantification of proteoforms by mass spectrometry. *Proteomics*, **19**(10), 1800361.

Serang, O. et al. (2012). Recognizing uncertainty increases robustness and reproducibility of mass spectrometry-based protein inferences. *Journal of Proteome Research*, **11**(12), 5586–5591.

Shental-Bechor, D. et al. (2008). Effect of glycosylation on protein folding: A close look at thermodynamic stabilization. *Proceedings of the National Academy of Sciences*, **105**(24), 8256–8261.

Shukla, A. K. et al. (2000). Tandem mass spectrometry: dissociation of ions by collisional activation. *Journal of Mass Spectrometry*, **35**(9), 1069–1090.

Silva, J. C. et al. (2006). Absolute quantification of proteins by LCMSE. *Molecular & Cellular Proteomics*, **5**(1), 144–156.

Siuti, N. et al. (2007). Decoding protein modifications using top-down mass spectrometry. *Nature Methods*, **4**(10), 817–821.

Skinner, O. S. et al. (2017). Top-down characterization of endogenous protein complexes with native proteomics. *Nature Chemical Biology*, **14**(1), 36–41.

Smith, L. M. et al. (2013). Proteoform: a single term describing protein complexity. *Nature Methods*, **10**(3), 186–187.

Smyth, G. K. (2004). Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Statistical Applications in Genetics and Molecular Biology*, **3**(1), 1–25.

Solá, R. J. et al. (2009). Effects of glycosylation on the stability of protein pharmaceuticals. *Journal of Pharmaceutical Sciences*, **98**(4), 1223–1245.

Solis, M. I. V. et al. (2019). Exploiting the dynamic relationship between peptide separation quality and peptide coisolation in a multiple-peptide matches-per-spectrum approach offers a strategy to optimize bottom-up proteomics throughput and depth. *Analytical Chemistry*, **91**(11), 7273–7279.

Sparkman, O. (2000). *Mass spectrometry desk reference*. Global View Pub, Pittsburgh, Pa.

Sticker, A. et al. (2020). Robust summarization and inference in proteome-wide label-free quantification. *Molecular & Cellular Proteomics*, **19**(7), 1209–1219.

The, M. et al. (2019). Integrated identification and quantification error probabilities for shotgun proteomics. *Molecular & Cellular Proteomics*, **18**(3), 561–570.

The, M. et al. (2016). Fast and accurate protein false discovery rates on large-scale proteomics data sets with percolator 3.0. *Journal of the American Society for Mass Spectrometry*, **27**(11), 1719–1727.

Thompson, A. et al. (2003). Tandem mass tags: a novel quantification strategy for comparative analysis of complex protein mixtures by MS/MS. *Analytical Chemistry*, **75**(8), 1895–1904.

Tibshirani, R. (1988). Variance stabilization and the bootstrap. *Biometrika*, **75**(3), 433–444.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, **58**(1), 267–288.

Ting, L. et al. (2011). MS3 eliminates ratio distortion in isobaric multiplexed quantitative proteomics. *Nature Methods*, **8**(11), 937–940.

Toby, T. K. et al. (2016). Progress in top-down proteomics and the analysis of proteoforms. *Annual Review of Analytical Chemistry*, **9**(1), 499–519.

Tyanova, S. et al. (2016). The perseus computational platform for comprehensive analysis of (prote)omics data. *Nature Methods*, **13**(9), 731–740.

Välikangas, T. et al. (2016). A systematic evaluation of normalization methods in quantitative label-free proteomics. *Briefings in Bioinformatics*, page bbw095.

Vehtari, A. et al. (2021). Rank-normalization, folding, and localization: An improved r^ for assessing convergence of MCMC (with discussion). *Bayesian Analysis*, **16**(2).

Wiens, D. P. et al. (2006). On the exact distribution of the sum of the largestn-kout ofnnormal random variables with differing mean values. *Statistics*, **40**(2), 165–173.

Wu, Z. et al. (2011). Quantitative chemical proteomics reveals new potential drug targets in head and neck cancer. *Molecular & Cellular Proteomics*, **10**(12), M111.011635.

Yao, X. et al. (2001). Proteolytic18o labeling for comparative proteomics: model studies with two serotypes of adenovirus. *Analytical Chemistry*, **73**(13), 2836–2842.

Young, G. A. (2005). *Essentials of statistical inference : G.A. Young, R.L. Smith*. Cambridge University Press, Cambridge, UK New York.

Zou, H. et al. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, **67**(2), 301–320.

Zubarev, R. A. et al. (1998). Electron capture dissociation of multiply charged protein cations. a nonergodic process. *Journal of the American Chemical Society*, **120**(13), 3265–3266.

Appendix

```
library(rstan)
library(brms)
library(pheatmap)
library(tidyr)
library(dplyr)



#### Data preparation ####

# Load the data
data_all <- read.csv('cleaned_psm-accessions.csv')

head(data_all)

proteins <- c('P14324', 'O15144', 'P29401-2')

# Select the protein
data <- data_all[data_all$Protein == proteins[1], ]

# Trim off the accession columsns that are all 0s
data <- data[, colSums(data != 0) > 0]

# Drop these columns
data <- data[, !names(data) %in% c('Modifications', 'Descriptions')]

# Get the log-2 intensities
data$LogPsm <- log2(data$Psm)

# Select data for up to hour 4
data <- data[data$Hour <= 4, ]

# Histogram of the PSMs
hist(data$LogPsm, breaks=30)
```

```
#### Hyperparameter model ####

# Hyperparameter model equation
eq_hyper <- LogPsm ~ 0 + Intercept + (1 | Peptide) + (1 | Replicate)

# Fit the model in brms
fit_hyper <- brm(eq_hyper, family = gaussian(),
                 data=data, cores=6,
                 iter = 8000, warmup = 2000) # family = student()

get_prior(eq_hyper, data=data)

# Diagnostic plot
pairs(fit_hyper)

# Get the summary estimates (R output)
summary(fit_hyper)

# Plot the fit
jpeg("../latex/images/hyper-plot.jpg", units="in", width=8, height=8, res=500)
plot(fit_hyper)
dev.off()

# Posterior predictive check of the hyperparameter model
pp_check(fit_hyper)

# Get the hyperparameter estimates
(pep_mean <- fixef(fit_hyper)[, 'Estimate'])
(pep_sd <- VarCorr(fit_hyper)$Peptide$sd[, 'Estimate'])




#### Observation model ####

# Model equation
eq_model <- LogPsm ~ 0 + Peptide + (1 | Replicate)

# Set the prior based on the hyper-parameter estimation
prior_peptide <- c(prior_string(
  paste("normal(", pep_mean, ", ", pep_sd^2, ")", sep="") ,
```

```
  class = "b",
  coef = paste("Peptide", unique(data$Peptide), sep="")))

# Fit the observation model
fit_model <- brm(eq_model,
                 family = gaussian(),
                 prior = prior_peptide,
                 data = data, cores=6,
                 iter = 8000, warmup = 2000)

# View the prior of the model
prior_summary(fit_model)

# Posterior predictive check
pp_check(fit_model)

# Posterior predictions
# posterior_epred(fit_model)

# Plot model
# plot(fit_model)

# Get the summary estimates (R output)
summary(fit_model)

# Get the names of the peptides
peps <- c(paste("b_Peptide", unique(data$Peptide), sep=""))

# Get the posterior peptide samples
pep_samples <- posterior_samples(fit_model)[, peps]

# Sort the peptide posterior distributions (for boxplot)
pep_names_order <- names(sort(apply(data.frame(pep_samples),
                                    MARGIN=2, FUN=median)))

# Plot boxplot of peptide posterior distributions
jpeg("../latex/images/pep_samples.jpg", units="in", width=6, height=6, res=500)
boxplot(pep_samples[, pep_names_order], xaxt='n', ylab='Log-2 Intensity',
        xlab='Peptides')
dev.off()
```

```
# Get an even distribution (unweighted)
protein_dist <- unlist(pep_samples)

# Check the histogram and kde of the even distribution
hist(protein_dist, breaks=60, freq = FALSE)
lines(x=density(protein_dist), col='red')

# Check the histogram and kde of all the PSMs
hist(data$LogPsm, breaks=60, freq = FALSE)
lines(x=density(data$LogPsm), col='red')

# Get the posterior z-score
post_z <- (fixef(fit_model)[, 'Estimate'] - pep_mean) / fixef(fit_model)[,
                                                                  'Est.Error']


# Get the posterior contraction
post_s <- 1 - (fixef(fit_model)[, 'Est.Error'] / pep_sd)^2

# Plot posterior z-scores as a function of posterior contraction
jpeg("../latex/images/z_contraction.jpg", units="in",
     width=5, height=5, res=500)
plot(post_s, post_z, xlim=c(0, 1), ylim=c(-6, 6),
     xlab='Posterior contraction', ylab='Posterior z-score')
grid()
dev.off()




#### Quality/Uncertainty metrics ####

# Get the mean quality metrics for each peptide
quality <- data %>%
  group_by(Peptide) %>%
  summarise_at(vars(c('Isolation', 'DeltaScore',
                      'qValue', 'PEP', 'IonsScore')), mean)

# Center and scale metrics, and put in a dataframe
quality <- data.frame(scale(quality[-1]), row.names = unlist(quality[, 1]))

# Flip the signs of these metrics
```

```
quality[, c('PEP', 'qValue', 'Isolation')] = quality[, c('PEP',
                                                'qValue', 'Isolation')] * -1

# Show that the quality metrics are correlated within this protein
pheatmap(cor(quality))

# Calculate gamma and sort the metrics for plotting
net_quality <- rowSums(quality)
net_quality <- sort(net_quality)

# Compare actual to expected SD
sd(net_quality)
sqrt(dim(quality)[2])

# Define the sigmoid function
sigmoid <- function(x, shift, scale) {
  1 / (1 + exp(scale * -(shift + x)))
}

# Set the location (shift) and scale parameters
shift <- 0.5
scale <- 0.5

# Calculate contribution (S)
contribution <- sigmoid(net_quality, shift, scale)

# Plot the contributions
jpeg("../latex/images/contributions.jpg", units="in",
     width=5, height=5, res=500)
plot(net_quality, contribution, ylim=c(0, 1), xlab='Gamma',
     ylab='S')
dev.off()

# Get the number of posterior peptide samples (S is the proportion)
pep_nums <- round(dim(pep_samples)[1] * contribution)

# Sample the number of peptide samples from the posterior peptide distributions
pep_selects <- unlist(mapply(sample, x=pep_samples, size=pep_nums))

# Plot the densities
jpeg("../latex/images/densities.jpg", units="in", width=6, height=5, res=500)
```

```
plot(x=density(posterior_samples(fit_hyper)[, "b_Intercept"]),
     col='red', main="", xlab="Log-2 Intensity")
lines(x=density(protein_dist), col='orange')
lines(x=density(pep_selects), col='green')
lines(x=density(data$LogPsm), col='blue')
legend("topright", legend=c("Hyperparameter",
                            "Even mixture", "Weighted mixture (model)",
                            "Raw PSMs"),
       col=c('red', 'orange', 'green', 'blue'), bty='l', lwd=2, pt.cex=1, cex=0.7)
dev.off()
```

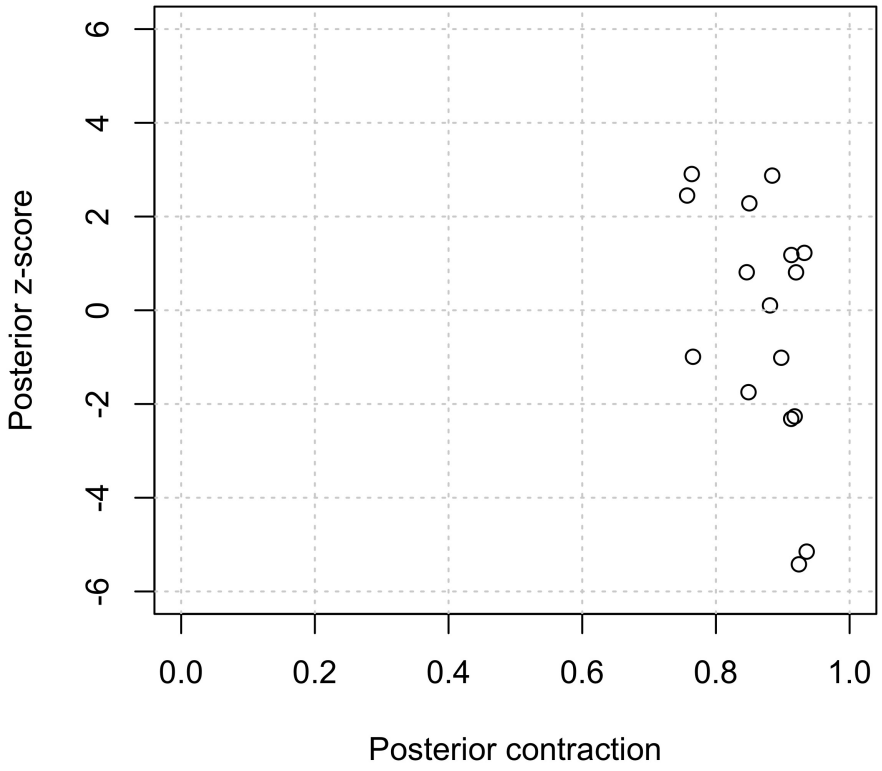


Figure 14: Posterior z-scores as a function of posterior contraction, for each peptide (Schad *et al.*, 2021).

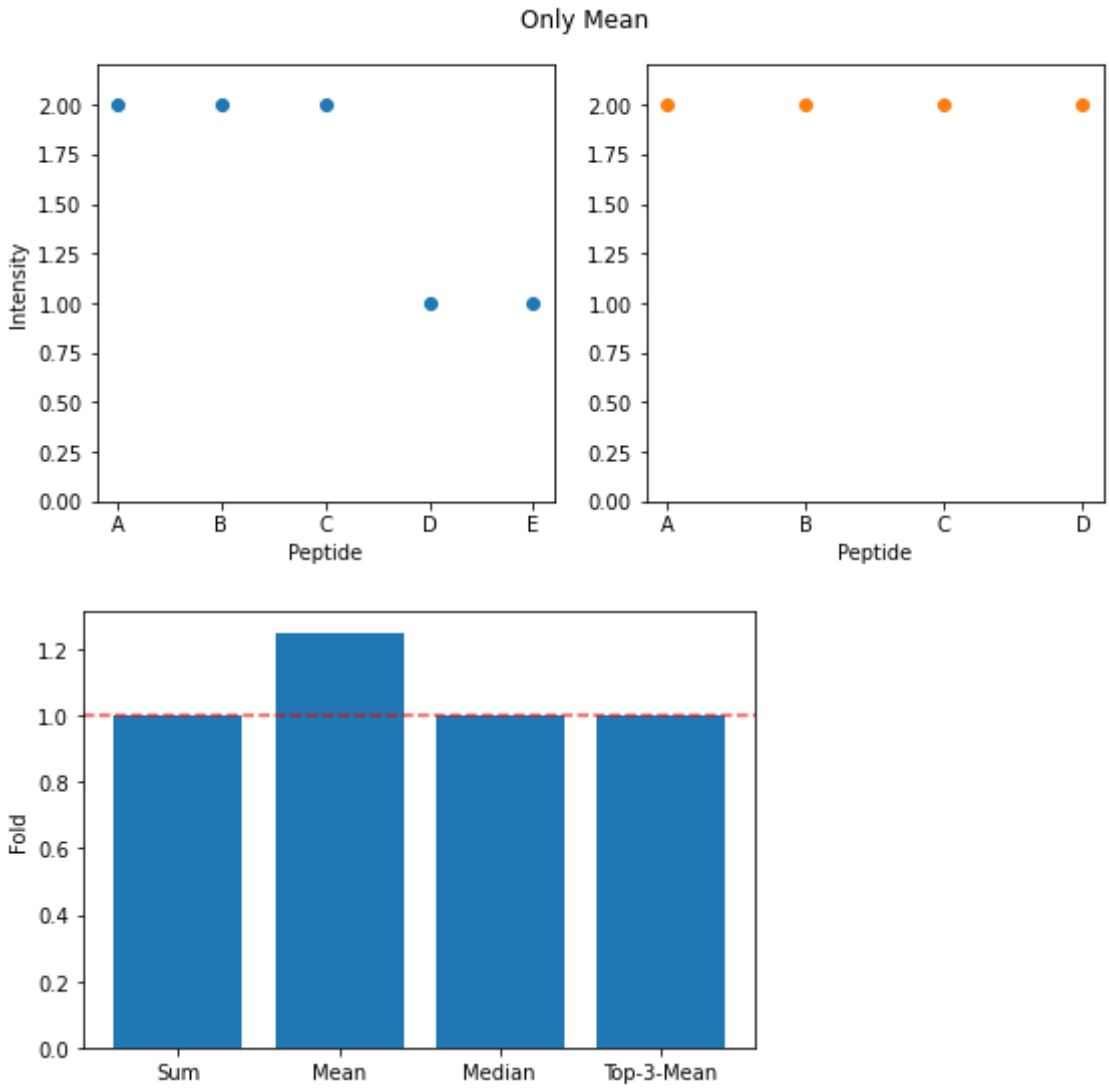
In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import string
```

In [2]:
```
def top_3_mean(array):
    return mean(sorted(array)[-3:])

def mean(array):
    return np.array(array).mean()

def median(array):
    return np.quantile(np.array(array), 0.5)

def get_fold(fun, psm1, psm2):
    entries = list(map(fun, [psm1, psm2]))
    return entries[1] / entries[0]


def plot_break(psm1, psm2, title):
    fig, ax = plt.subplots(1, 2, figsize=(9, 4))
    plt.suptitle(title)

    ax[0].plot(list(string.ascii_uppercase[:len(psm1)]), psm1,'o',
               color='tab:blue')
    ax[1].plot(list(string.ascii_uppercase[:len(psm2)]), psm2,'o',
               color='tab:orange')
    ax[0].set_ylim(0, 1.1 * max(psm1 + psm2))
    ax[1].set_ylim(0, 1.1 * max(psm1 + psm2))
    ax[0].set_xlabel('Peptide')
    ax[1].set_xlabel('Peptide')
    ax[0].set_ylabel('Intensity')
    plt.show()


    folds = [get_fold(fun, psm1, psm2) for fun in [
             sum, mean, median, top_3_mean]]
    plt.bar(['Sum', 'Mean', 'Median', 'Top-3-Mean'], folds)
    plt.axhline(y=1, color='red', linestyle='--', alpha=0.7)
    plt.ylabel('Fold')
```

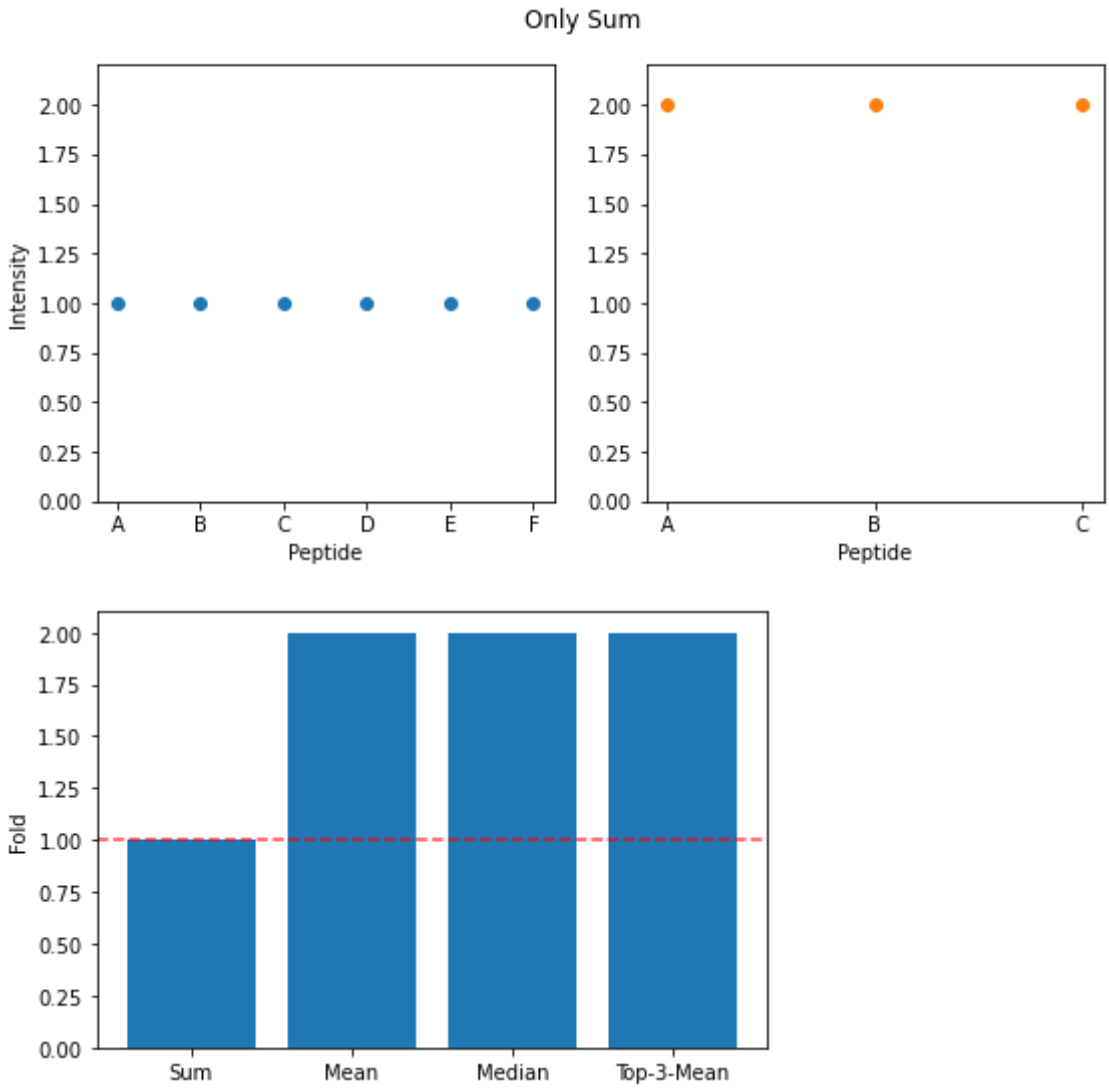
Break the simple summary statistics

Basic summary statistics are used to obtain an overall protein intensity from its peptide intensities, and ultimately a 'fold change' between two proteins. However, these summaries can give unreliable results depending on the distribution and 'missingness' of the data peptides.

In [3]:
```
plot_break([2, 2, 2, 1, 1], [2, 2, 2, 2], 'Only Mean')
```

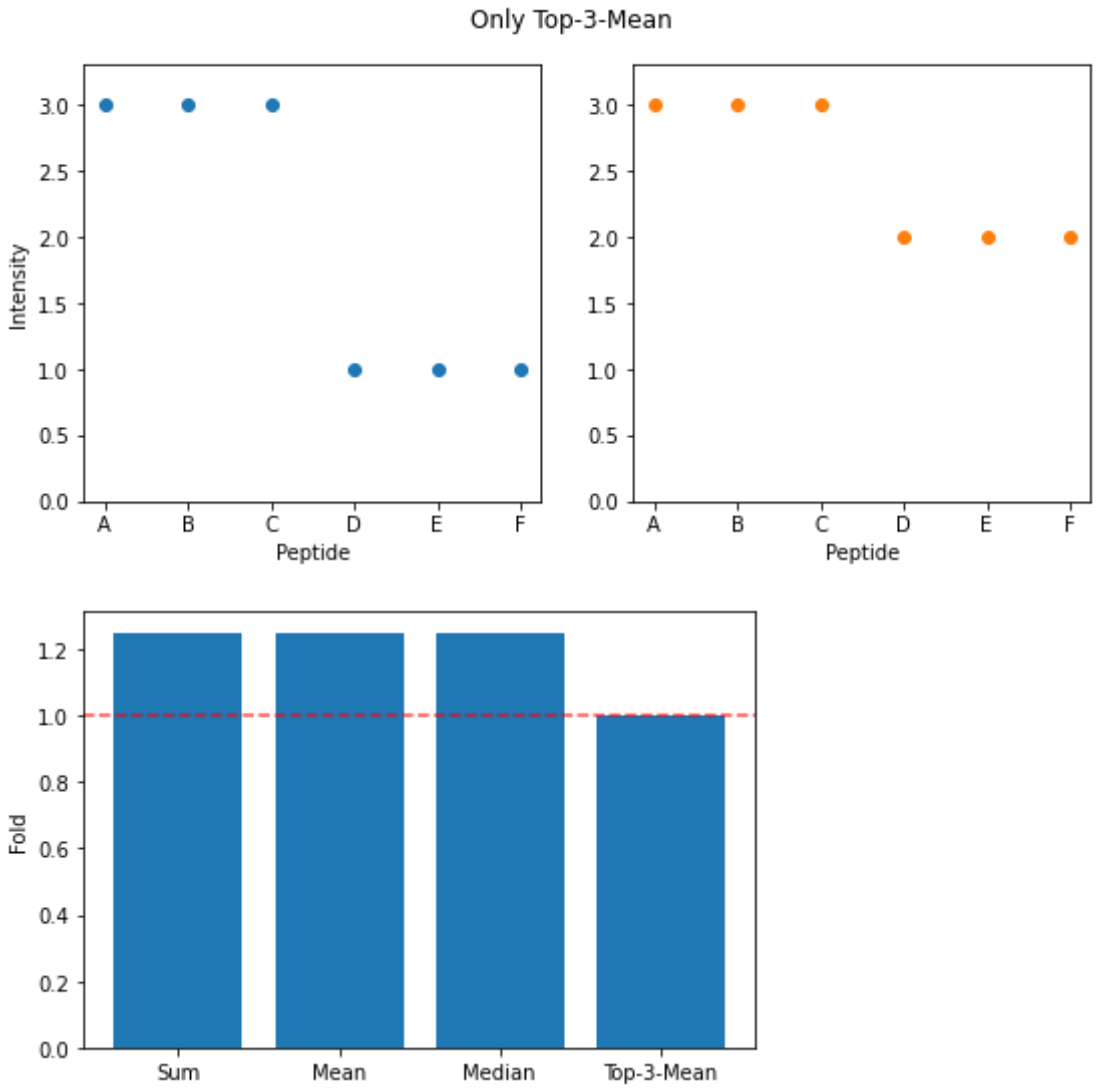


```
In [4]: plot_break([1, 1, 1, 1, 1, 1], [2, 2, 2], 'Only Sum')
```



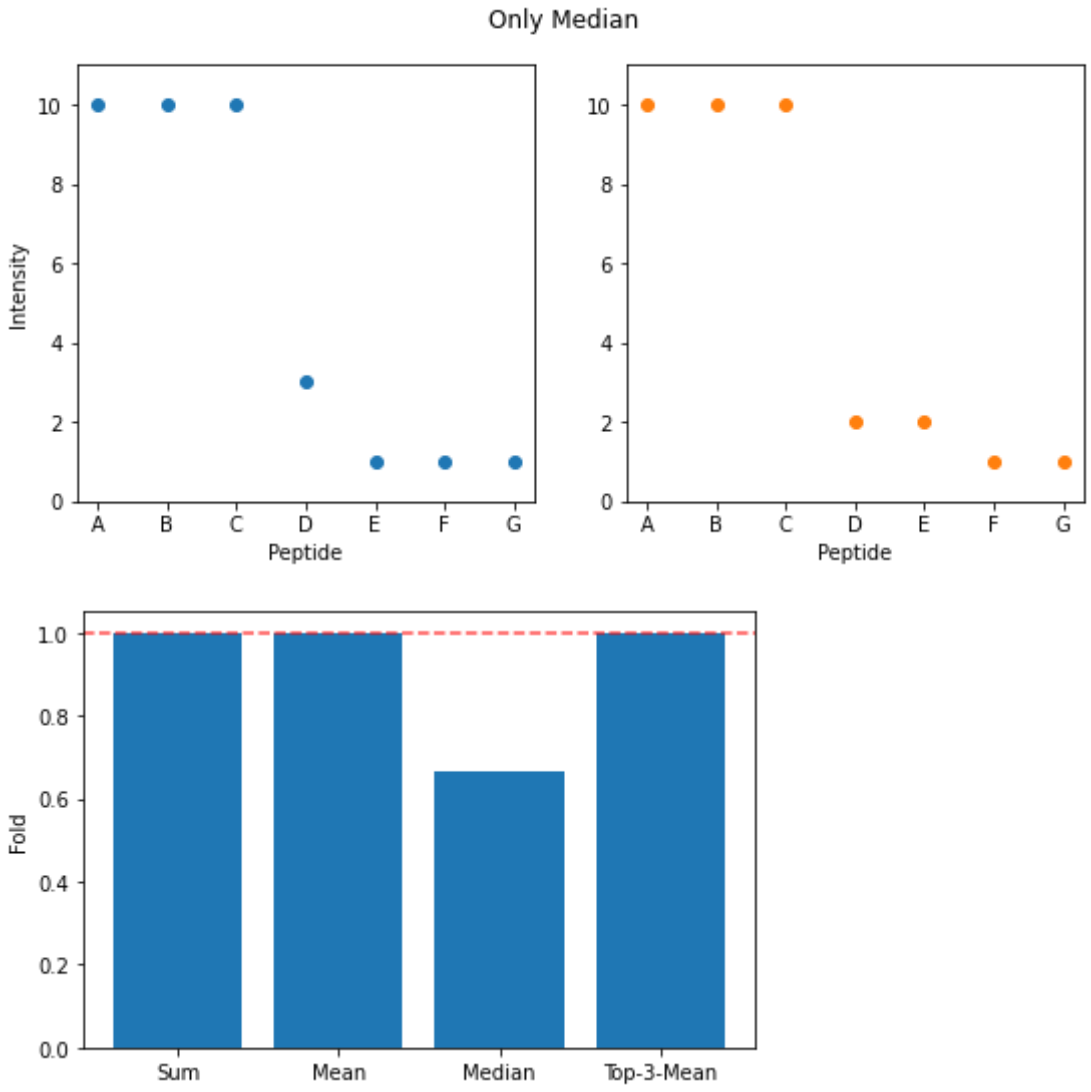
If both the mean and the sum give the same fold, there must be the same number of peptides. Therefore, the only way to have mean and sum give different folds is to have different numbers of peptides, as above.

```
In [5]: plot_break([3, 3, 3, 1, 1,1], [3, 3, 3, 2, 2, 2], 'Only Top-3-Mean')
```



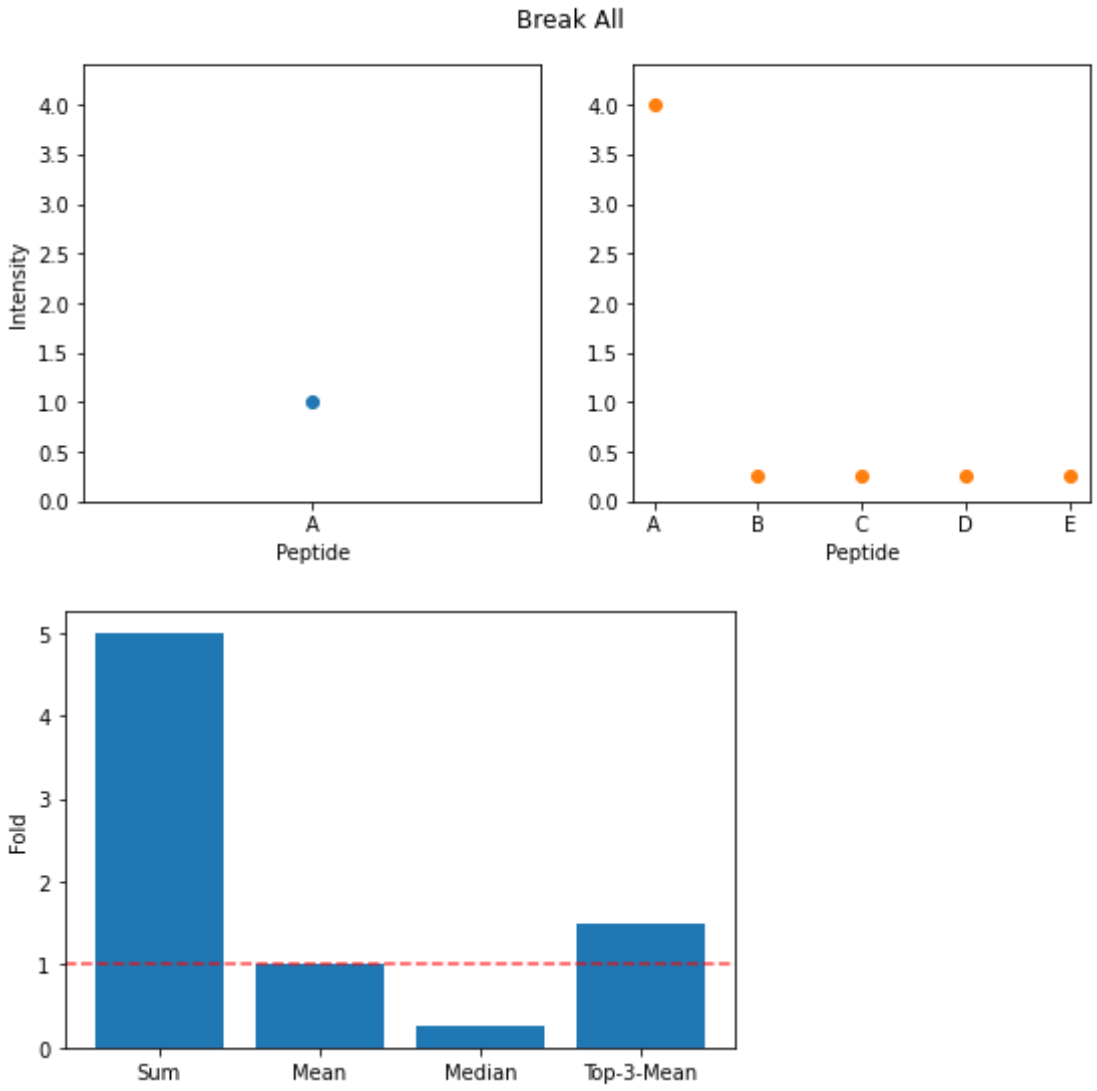
This is easy, as long as the the top 3 intensities have the same mean, but the rest of the peptides can tell a completely different story.

In [6]:
```
plot_break([10, 10, 10, 3, 1, 1, 1], [10, 10, 10, 2, 2, 1, 1], 'Only Median')
```

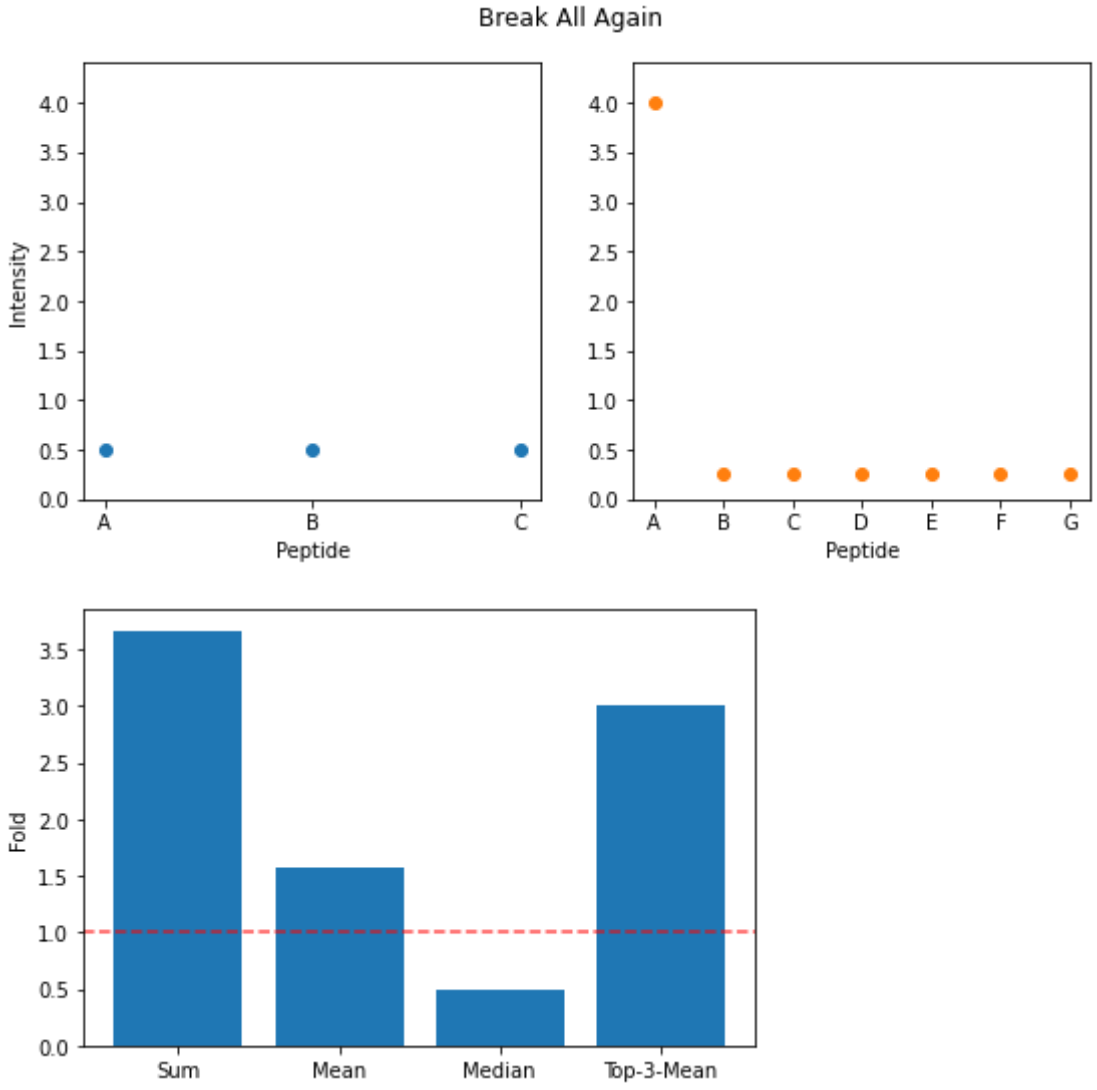


Breaking only the median requires skewness in the distribution of intensities, while keeping the top 3 mean the same.

In [8]:
```
plot_break([1], [4, 0.25, 0.25, 0.25, 0.25], 'Break All')
```



In [10]:
```
plot_break([0.5, 0.5, 0.5], [4, 0.25, 0.25, 0.25, 0.25, 0.25, 0.25], 'Break All Again')
```



In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv('cleaned_psm-accessions.csv')

# Show all columns when displaying
pd.set_option('display.max_columns', None)
```

In [3]:
```
df['LogPsm'] = np.log2(df.Psm)
```

In [ ]:
```

```

Variance within the protein

In [4]:
```
df.groupby(['Protein']).LogPsm.var().mean()
```

Out[4]: 0.6838966930505698

Variance within the protein and peptide

In [5]:
```
df.groupby(['Protein', 'Peptide']).LogPsm.var().mean()
```

Out[5]: 0.34304199032625743

In [6]:
```
df.groupby(['Protein', 'Peptide', 'Replicate']).LogPsm.var().mean()
```

Out[6]: 0.18461602127360752

In [ ]:
```

```

In [ ]:
```

```

Variance by Peptide

In [7]:
```
# Between peptide variance of Psm (explained)
df.groupby(['Peptide']).LogPsm.mean().var()
```

Out[7]: 0.7378434612265585

In [8]:
```
# Within peptide variance of Psm (unexplained)
df.groupby(['Peptide']).LogPsm.var().mean()
```

Out[8]: 0.3430419903262574

In [9]:
```
# Within peptide, replicate variance of Psm (unexplained)
df.groupby(['Peptide', 'Replicate']).LogPsm.var().mean()
```

Out[9]: 0.18461602127360752

In [ ]:
```

```

Variance by Protein

In [10]:
```
# Between protein variance of Psm (explained)
df.groupby('Protein').LogPsm.mean().var()
```

Out[10]: 0.2892763514098141

In [11]:
```
# Within protein variance of Psm (unexplained)
df.groupby('Protein').LogPsm.var().mean()
```

Out[11]: 0.6838966930505698

Variance by Replicate

In [12]:
```
# Between replicate variance of Psm
df.groupby('Replicate').LogPsm.mean().var()
```

Out[12]: 0.10785913638706915

In [13]:
```
# Within replicate variance of Psm
df.groupby('Replicate').LogPsm.var().mean()
```

Out[13]: 1.0327928527440549

Variance by Fraction

In [14]:
```
# Between fraction variance of Psm
df.groupby('Fraction').LogPsm.mean().var()
```

Out[14]: 0.1727213104990265

In [15]:
```
# Within fraction variance of Psm
df.groupby('Fraction').LogPsm.var().mean()
```

Out[15]: 0.8915972732788634

Variance by Replicate and Fraction

In [16]:
```
df.groupby(['Replicate', 'Fraction']).LogPsm.mean().var()
```

Out[16]:
```
0.24929007474783213
```

In [17]:
```
df.groupby(['Replicate', 'Fraction']).LogPsm.var().mean()
```

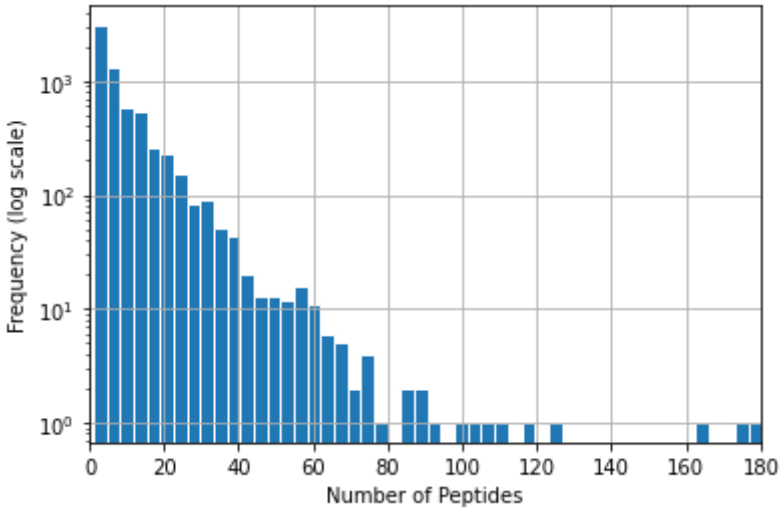
Out[17]:
```
0.801839221702455
```

In [ ]:

In [ ]:

Peptides per Protein

In [18]:
```
df.groupby('Protein').Peptide.nunique().hist(bins=90,
            histtype='bar', ec='white')
plt.xlim(0, 180)
plt.yscale('log')
# plt.title('Histogram of number of Peptides per Protein')
plt.xlabel('Number of Peptides')
plt.ylabel('Frequency (log scale)')

plt.savefig('../latex/images/peps_per_prot.png', dpi=300)
```



In [19]:
```
# Mean
df.groupby('Protein').Peptide.nunique().mean()
```

Out[19]:
```
8.658477033419807
```

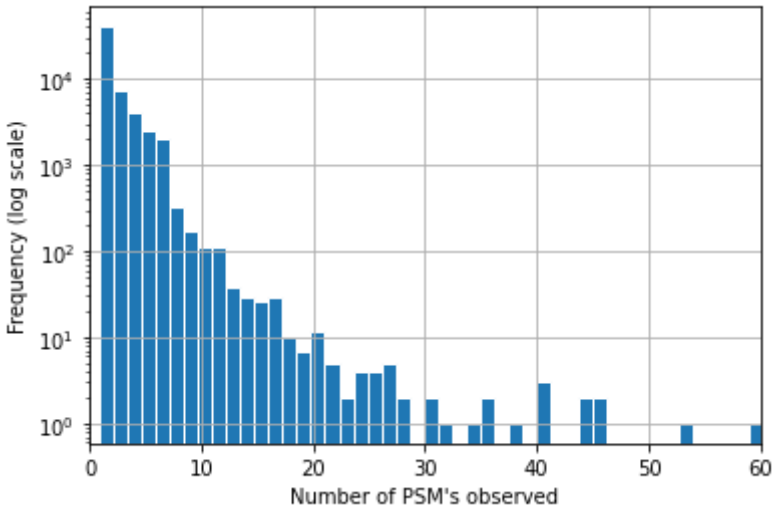
In [20]:
```
df.groupby('Protein').Peptide.nunique().nlargest(n=3)
```

Out[20]:
```
Protein
Q09666    324
Q14204    179
P35579    175
Name: Peptide, dtype: int64
```

PSM's per Peptide

(Not including the 55 missing PSM's)

In [21]:
```
(df.groupby(['Peptide']).Psm.count() / 6).hist(bins=50,
            histtype='bar', ec='white')
plt.xlim(0, 60)
plt.yscale('log')
# plt.title('Histogram of number of PSM\'s per Peptide')
plt.xlabel('Number of PSM\'s observed')
plt.ylabel('Frequency (log scale)')

plt.savefig('../latex/images/psms_per_pep.png', dpi=300)
```



In [22]:
```
# Mean
df.groupby(['Peptide']).Psm.count().mean()
```

Out[22]:
```
13.354817673910361
```

In [23]:
```
df.groupby(['Peptide']).Psm.count().nlargest()
```

Out[23]:
```
Peptide
eSYSVYVYk         384
iSGLIYEETR        354
gNPTVEVDLFTSk     318
dAVTYTEHAk        276
mDSTANEVEAVk      276
Name: Psm, dtype: int64
```

In [ ]:

In [ ]:

Ratio of within peptide to between peptide variance for each 'quality feature'

If the within peptide variance is smaller than the between peptide variance, which it almost always is, that means there is consistency of the quality feature within peptides.

Eve's law is checked by taking the ratio of the square root of the equation to the overall st. dev. of the feature.

In [24]:
```
features = ['PEP', 'qValue', 'IonsScore', 'DeltaScore', 'Isolation']

for feature in features:
    within_group = df.groupby('Peptide')[feature].var().mean()
    between_group = df.groupby('Peptide')[feature].mean().var()

    print('##', feature, '##')

    print('Var Ratio:', round(between_group / within_group, 4))
    print('St Dev Ratio:', round(np.sqrt(within_group / between_group),
4))
    print()
    print('Check Eve\'s law:', np.sqrt(within_group + between_group) / d
f[feature].std())
    print('##')
    print('\n')
```

```
## PEP ##
Var Ratio: 3.4444
St Dev Ratio: 0.5388

Check Eve's law: 1.0348893070903207
##


## qValue ##
Var Ratio: 4.0269
St Dev Ratio: 0.4983

Check Eve's law: 1.0437121515309686
##


## IonsScore ##
Var Ratio: 4.6264
St Dev Ratio: 0.4649

Check Eve's law: 0.9678848461903465
##


## DeltaScore ##
Var Ratio: 12.6879
St Dev Ratio: 0.2807

Check Eve's law: 1.0872784369162383
##


## Isolation ##
Var Ratio: 2.797
St Dev Ratio: 0.5979

Check Eve's law: 1.0552724266090934
##
```

In [ ]:

In [ ]:

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
df = pd.read_csv('cleaned_psm-accessions.csv')

# Show all columns when displaying
pd.set_option('display.max_columns', None)
```

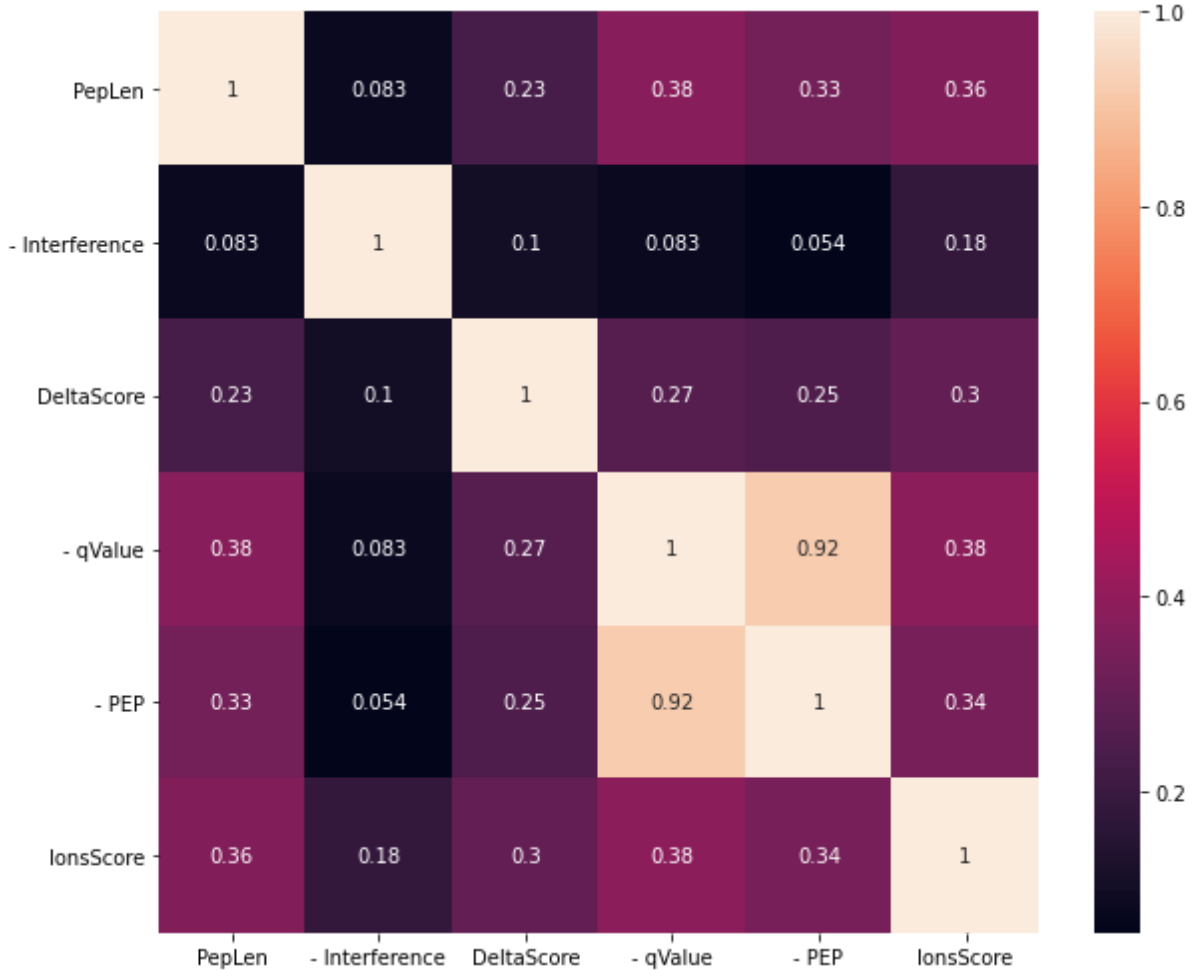
In [3]:
```
df.rename(columns={'Isolation': 'Interference'}, inplace=True)
```

In [4]:
```
df[['PEP', 'qValue', 'Interference']] = df[['PEP', 'qValue', 'Interferen
ce']] * -1
```

In [5]:
```
df.rename(columns={'PEP': '- PEP', 'qValue': '- qValue',
                   'Interference': '- Interference'}, inplace=True)
```

In [6]:
```
quality_terms = ['PepLen', '- Interference', 'DeltaScore', '- qValue',
'- PEP', 'IonsScore']
```

In [7]:
```
corr = df[quality_terms].corr()

plt.figure(figsize=(9, 7))
sns.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns,
            annot=True);
plt.tight_layout()
plt.savefig('../latex/images/uncertainty_corr.png', dpi=300)
```



In [ ]:
```
```