Bioinformatic Analysis of Temporal and Spatial Proteome Alternations During Infections

Matineh Rahmatbakhsh[1†], Alla Gagarinova[2†] and Mohan Babu[1*]*

[1] Department of Biochemistry, University of Regina, Regina, SK, Canada, [2] Department of Biochemistry, Microbiology, & Immunology, University of Saskatchewan, Saskatoon, SK, Canada

Microbial pathogens have evolved numerous mechanisms to hijack host's systems, thus causing disease. This is mediated by alterations in the combined host-pathogen proteome in time and space. Mass spectrometry-based proteomics approaches have been developed and tailored to map disease progression. The result is complex multidimensional data that pose numerous analytic challenges for downstream interpretation. However, a systematic review of approaches for the downstream analysis of such data has been lacking in the field. In this review, we detail the steps of a typical temporal and spatial analysis, including data pre-processing steps (i.e., quality control, data normalization, the imputation of missing values, and dimensionality reduction), different statistical and machine learning approaches, validation, interpretation, and the extraction of biological information from mass spectrometry data. We also discuss current best practices for these steps based on a collection of independent studies to guide users in selecting the most suitable strategies for their dataset and analysis objectives. Moreover, we also compiled the list of commonly used R software packages for each step of the analysis. These could be easily integrated into one's analysis pipeline. Furthermore, we guide readers through various analysis steps by applying these workflows to mock and host-pathogen interaction data from public datasets. The workflows presented in this review will serve as an introduction for data analysis novices, while also helping established users update their data analysis pipelines. We conclude the review by discussing future directions and developments in temporal and spatial proteomics and data analysis approaches. Data analysis codes, prepared for this review are available from https://github.com/BabuLab-UofR/TempSpac, where guidelines and sample datasets are also offered for testing purposes.

Keywords: temporal proteomics, spatial proteomics, host-pathogen interactions, clustering, principal component analysis, self-organizing maps, data imputation, normalization

OPEN ACCESS

Edited by:
Nicole Hansmeier,
Luther College, Canada

Reviewed by:
Vikas Kumar,
University of Nebraska Medical Center, United States
Salvador Martínez-Bartolomé,
The Scripps Research Institute, United States

***Correspondence:**
Alla Gagarinova
alla.gagarinova@mail.utoronto.ca;
alla.gagarinova@usask.ca
Mohan Babu
mohan.babu@uregina.ca

[†]These authors have contributed equally to this work

Specialty section:
This article was submitted to Systems Biology Archive, a section of the journal Frontiers in Genetics

Received: *15 February 2021*
Accepted: *08 June 2021*
Published: *02 July 2021*

Citation:
Rahmatbakhsh M, Gagarinova A and Babu M (2021) Bioinformatic Analysis of Temporal and Spatial Proteome Alternations During Infections. Front. Genet. 12:667936. doi: 10.3389/fgene.2021.667936

INTRODUCTION

Intracellular pathogens, including viruses, bacteria (Auweter et al., 2011; Schweppe et al., 2015; Lopez et al., 2016), parasites, and fungi (Iyer et al., 2007; Gilbert et al., 2015; May and Casadevall, 2018; Eisenreich et al., 2019), cause numerous deaths and impose staggering healthcare costs (Kamaruzzaman et al., 2017). Spatially and temporally intricate progression of interplay between the host and the pathogen results in disease. This interplay in host-pathogen interactions (HPI) is highly complex and dynamic. Although mechanistic details vary, all intracellular pathogens need to enter the host cell, avoid or exploit the host's defense mechanisms, and exploit the host's resources (e.g., lipids, proteins, and metabolites) for replication and spread to neighboring cells (Jean Beltran et al., 2016). Studies of HPI-dependent alterations to the host cell's proteome not only reveal components required for pathogenesis but also provide critical insights into host processes (e.g., see Alto and Orth, 2012; Jo, 2019 and references therein).

A major key to combatting intracellular pathogens lies in the understanding of how they hijack host systems. This, in turn, requires the mapping the spatial and temporal proteome changes underlying disease progression. These may include changes in protein abundance, interactions, localizations, or posttranslational modifications (Ribet and Cossart, 2010; Gagarinova et al., 2017; Scott and Hartland, 2017; Grishin et al., 2021). For example, Weekes et al. (2014) mapped proteome changes occurring during the course of the human cytomegalovirus infection, thereby identifying key temporal changes and potential new targets for antiviral therapies. Likewise, pathogens actively regulate organelle dynamics (Auweter et al., 2011; Schweppe et al., 2015; Lopez et al., 2016; Selkrig et al., 2020).

Technological advances in mass spectrometry (MS)-based proteomics and bioinformatics allow achieving temporal and spatial resolution of the infection process at previously unseen levels (e.g., see Lopez et al., 2016; Jean Beltran et al., 2017; Selkrig et al., 2020 and references therein). See Kumar and Mann (2009) and Jean Beltran et al. (2016) for an overview of quantitative proteomic approaches and relevant computational methods. The typical output of a quantitative MS experiment that maps temporal and/or spatial changes during an infection includes highly complex, multi-dimensional data matrices with protein abundances across space or time represented by ion intensities or spectral counts, depending on the MS approach. Such data are challenging to analyze and interpret. However, a review covering such downstream analyses has been lacking. We present frameworks for the analysis of temporal (section "Temporal Analysis of Proteome Changes in an Infected Cell") and spatial (section "Exploring Subcellular Proteome Organization During Infection") proteomic data from HPI studies, focusing on specific examples and robust methods adapted from statistics and machine learning. We also discuss measures for validating the results and describe how these frameworks can be implemented in R programming language, suggesting appropriate software packages where applicable (all packages are summarized in **Table 1**). Moreover, we combined useful functions into workflows in R programming language. These are available at https://github.com/BabuLab-UofR/TempSpac. The workflows we discuss and present can be used to extract biological meaning from MS data to model disease progression and drive therapeutics discovery. Although examples in this review focus on intracellular pathogens, the same pipelines can be used, e.g., in the analysis of genetic or environment-induced disease.

TABLE 1 | List of packages and useful functions.

Sections	Packages	Descriptions and useful functions	References
Unsupervised clustering (section "Clustering Analyses")	stats	hclust() = agglomerative hierarchical clustering	RStudio Team, 2020
		cuttree() = control the number of generated clusters	
		kmeans() = *K*-mean clustering	
	cluster	diana() = divisive hierarchical clustering	Maechler et al., 2019
		agnes() = agglomerative hierarchical clustering	
		fanny() = fuzzy clustering	
	hybridHclust	mutualCluster() = mutual cluster	Chipman and Tibshirani, 2006
	Mfuzz	mfuzz() = fuzzy clustering	Kumar and Futschik, 2007
	e1071	cmeans() = fuzzy clustering	Meyer et al., 2020
	ppclust	fcm() = fuzzy clustering	Cebeci, 2019
	pracma	Kmeanspp() = *k*-means++ clustering algorithm	Borchers, 2019
	Kohonen	som() = self-organizing map (som) clustering	Wehrens and Kruisselbrink, 2019
	clValid	clValid() = clMethods argument specifies clustering methods e.g., "hierarchical," "kmeans," etc.	Brock et al., 2011
		clValid() = validation argument specifies validation measures e.g., "biological," "internal," etc.	
	ClusterR	external_validation() = external measures	Mouselimis et al., 2020
	factoextra	fviz_nbclust() = define optimal number of clusters	Kassambara and Mundt, 2020
	ggfortify	autoplot() = output appropriate plots based on the type of unsupervised clustering	Tang et al., 2016
Supervised clustering (section "Predicting Protein Localizations in Each Condition")	pRoloc	knnClassification() = *k*-nearest neighbors (*k*-NN) algorithm	Gatto et al., 2014b
		knnOptimisation() = classification parameter optimization for *k*-NN	
		svmClassification() = support vector machine (svm) algorithm	
		svmOptimisation() = classification parameter optimization for svm	
		nnetClassification() = neural net (nnet) algorithm	
		nnetOptimisation() = classification parameter optimization for nnet	
		nbClassification() = naïve bayes (nb) algorithm	
		nbOptimisation() = classification parameter optimization for nb	

(Continued)

TABLE 1 | Continued

Sections	Packages	Descriptions and useful functions	References
	caret	train() = to fit a model trainControl() = control parameters for training predict() = predict probability scores	Kuhn et al., 2020
Data normalization (sections "Quantitative Temporal Data Visualization, Preprocessing, and Quality Control" and "Data Normalization")	edgeR	caclNormFactors() = method argument specifies the type of normalization (e.g., TMM)	Robinson et al., 2010
	DESeq2	estimateSizeFactors() = compute scaling factor for normalization by RLE method counts() = retrieve normalized matrix counts	Love et al., 2014
	DEP	normalize_vsn() = perform normalization using that variance stabilization normalization (Vsn)	Zhang et al., 2018
	vsn	Justvsn() = perform normalization using variance stabilization normalization (Vsn)	Huber et al., 2002
	MASS	lm() = perform linear regression	Venables and Ripley, 2002
Missing value imputation (sections "Quantitative Temporal Data Visualization, Preprocessing, and Quality Control" and "Missing Value Imputation")	pcaMethods	llsImpute() = perform imputation of missing value using local least squares approach (LLS)	Stacklies et al., 2007
	bnstruct	knn.impute() = perform imputation of a missing value using *k*-NN	Franzin et al., 2017
	DEP	impute() = missing value imputation; fun argument specifies imputation method e.g., *k*-NN, "QRILC," etc.	Zhang et al., 2018
	mice	mice() = perform imputation of missing values using mice	Van Buuren and Groothuis-Oudshoorn, 2010
	MSstats	MBimpute() = impute missing values	Choi et al., 2014
Differential expression analysis (Section "Statistical Analysis of Quantitative Temporal Proteomics Data")	MSstats	detect differentially expressed proteins in both label-free and labeling-based experimental approaches	Choi et al., 2014

(Continued)

TABLE 1 | Continued

Sections	Packages	Descriptions and useful functions	References
	MSstatsTMT	detect differentially expressed proteins in experiments with isobaric labeling	Huang et al., 2020
	DEP	detect differentially expressed proteins in experiments from both label-free and labeling-based experimental approaches	Zhang et al., 2018
	limma	detect differentially expressed proteins when sample sizes are small (<10)	Ritchie et al., 2015
	DESeq2	detect differentially expressed proteins when sample sizes are small (<10)	Love et al., 2014
	edgeR	detect differentially expressed proteins when sample sizes are small (<10)	Robinson et al., 2010
Dimensionality-reduction techniques (sections "Quantitative Temporal Data Visualization, Preprocessing, and Quality Control" and "Dimensionality Reduction Tools for Visualizing Organellar Map")	tnse	tnse() = t-SNE dimensionality reduction	Donaldson, 2016
	umap	umap() = UMAP dimensionality reduction	Konopka, 2020
	stats	prcomp() = PCA dimensionality reduction princomp() = PCA dimensionality reduction	RStudio Team, 2020

Additional scripts, written for this review are provided at https://github.com/BabuLab-UofR/TempSpac.

TEMPORAL ANALYSIS OF PROTEOME CHANGES IN AN INFECTED CELL

Several temporal studies employed quantitative whole-cell proteomics in order to quantify changes occurring during the course of a productive viral infection, helping elucidate HPI mechanisms, immune responses, and mechanisms of immune system evasion by the pathogen (Weekes et al., 2014; Greenwood et al., 2016; Clements et al., 2017; Caller et al., 2019; Soday et al., 2019). For instance, Soday et al. (2019) achieved extensive host and viral proteome coverage, and their downstream analyses revealed multiple pathways dysregulated in response to Vaccinia virus infection. These included antiviral factors, collagens, and interferon-stimulated genes (e.g., IFITM3) (Soday et al., 2019).

Moreover, quantitative temporal whole-cell proteomics in the presence and absence of a specific viral protein has been used to elucidate how the selected viral protein contributes to disease (e.g., Lapek et al., 2017; Greenwood et al., 2019). For instance, Lapek and colleagues reported that Vpr, a human immunodeficiency virus protein, mediated the modulation of serine/arginine-rich protein-specific kinases, spindle and centromere proteins, and others (Lapek et al., 2017). Thus, a potential role for Vpr in RNA splicing *via* serine/arginine-rich protein-specific kinases has been suggested (Lapek et al., 2017). Although other approaches would need to be employed to distinguish direct vs. indirect effects of specific viral proteins, this approach provides a framework for dissecting the activities of specific proteins in pathogenesis.

Although whole-cell temporal quantitative proteomic analyses reveal key pathways and proteins affected by infection, they do not contain spatial information about protein dynamics within subcellular compartments, which is essential to understand the organization of proteome upon infection and the underlying mechanisms. Organelle temporal proteomics reveal dynamic changes on sub-cellular level with higher resolution than whole-cell proteomics due to better ability to detect low-abundance proteins. Moreover, temporal proteomic data from whole cells and subcellular fractions can be integrated in order to compare the total abundance of a given protein in whole-cell lysate vs. a specific organelle to better understand disease progression and pathogenesis strategies. For instance, Weekes et al. (2014) quantified temporal human cytomegalovirus-induced changes both in whole-cell lysates and at cell surface. The results indicated that human cytomegalovirus infection resulted in rapid depletion of CD155 (poliovirus receptor, PVR) from the cell surface at the same time as the total amount of CD155 in the whole cell increased (Weekes et al., 2014). CD155 is a ligand involved in the activation of natural killer cell-mediated immunity against human cytomegalovirus (Tomasec et al., 2005). Therefore, sequestration of CD155 may be one of the pathogenesis strategies of human cytomegalovirus (Weekes et al., 2014).

In a temporal proteomic HPI study, infected and uninfected cells or organelles are collected and processed for quantitative MS (**Figure 1**). The MS data are then analyzed by specialized software, such as MaxQuant (Cox and Mann, 2008; Chen et al., 2020). As a result, multidimensional data with information about protein identities and abundances in infected vs. uninfected cells across time are obtained. Sections “Quantitative Temporal Data Visualization, Preprocessing, and Quality Control”–“Evaluation Measures for Temporal Clustering” present a robust workflow for the downstream analysis of such data (**Figure 2**).

Quantitative Temporal Data Visualization, Preprocessing, and Quality Control

The data generated by temporal profiling can be represented in a matrix format with features (i.e., proteins) and different time points along rows and columns, respectively **Figure 2A**. The first step in data analysis is to check data quality by using a set of metrics or through visualization. Here, first, data distribution, variation, and other descriptive statistics are assessed (e.g., using box plot, line chart, histogram, and density plot). There is a handy multipurpose function called summary() in R language (RStudio Team, 2020) that provides descriptive statistics for each variable or column and reports the number of missing values in the dataset. One can also directly visualize the temporal profile of each protein by plotting its abundance or relative intensity across different time points (**Figure 2B**i). Such plots help detect misidentified features (i.e., proteins) with inconsistent temporal quantitative profiles or missing values.

Data preprocessing includes the imputation of missing values and the normalization of the data (**Figures 2B**i,ii). Data preprocessing is essential in the analysis of quantitative proteomics data (Karpievitch et al., 2012). For example, data may be missing for low-abundance proteins (Karpievitch et al., 2012). These missing values may be removed or imputed. The easiest way to impute a missing value in temporal data is to fit a curve to the incomplete temporal data, and then missing abundance could be imputed based on the fitted values in the curve. However, this method has a disadvantage of introducing potential artifacts as the time series will be constrained to follow the fitted curve (Du et al., 2008). It is usually assumed that there is no dramatic change in abundance values between nearest time points; therefore, abundance values at the nearby time points could be used to impute each missing abundance value (Du et al., 2008). The imputation of missing values in time-series datasets and the effect of different imputation methods on the resulting inferences have been extensively studied for microarray analyses. The same methods can be applied to the temporal proteomic data. For instance, Chiu et al. (2013) evaluated the performance of nine different imputation algorithms. LLS-like algorithms [local least squares (LLS), iterative local-least-squares (ILLS), and sequential local-least-squares (SLLS)] outperformed other algorithms for time-series datasets. The LLS imputation algorithm (Kim et al., 2005) first identifies genes similar to the gene with a missing value by applying a distance measure (e.g., Euclidean Distance or Pearson correlation coefficient). Then, the missing value is estimated by representing the target gene as a linear combination of similar genes. This method is implemented as llsImpute() function in the pcaMethods Bioconductor package (Stacklies et al., 2007) in the R environment.

Data normalization aims to eliminate systematic biases to allow statistical inferences (Karpievitch et al., 2012). MS-based data are typically biased due to a number of factors. One of them is inadequate normalization before LC/MS (Wisiniewski and Mann, 2016; Chen et al., 2020). Therefore, selecting an appropriate normalization method is essential. The concept of data normalization in transcriptomics and proteomics has been explored extensively (see Karpievitch et al., 2012; Välikangas et al., 2018; Chen et al., 2020), albeit not in the context of temporal quantitative proteomics data. The dynamic pattern of protein abundances is a key measure in a time-course study. Different time points can be normalized against the same reference sample, included in each MS run (Du et al., 2008; Li et al., 2018; Nusinow and Gygi, 2020). For instance, the relative intensity for each protein can be normalized to the sum or average of all protein intensities in the reference sample (Nusinow and Gygi, 2020). However, in this case, biases may persist (see Murie et al., 2018 and references therein). Therefore, data may instead be adjusted for protein loading across all channels/samples by normalizing

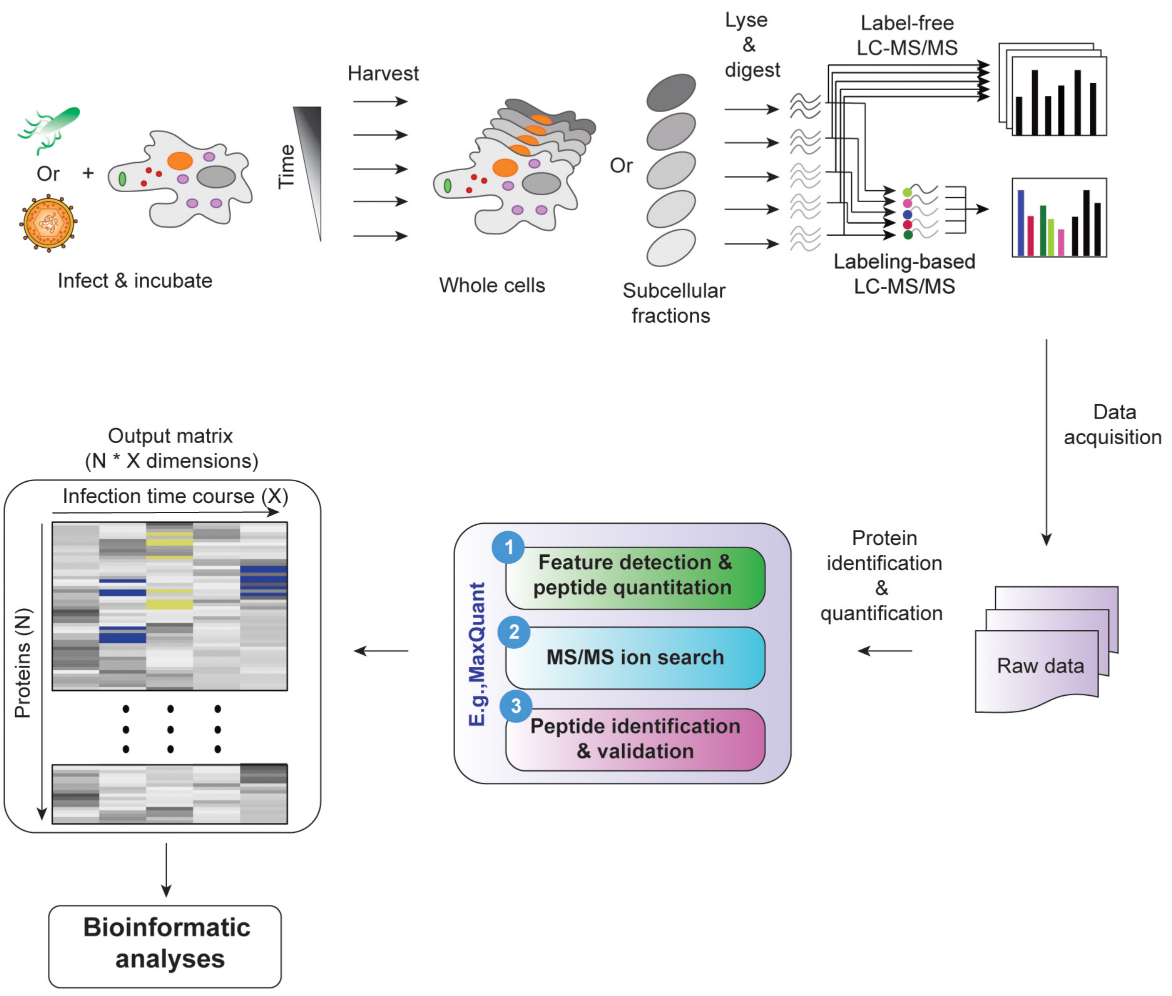


FIGURE 1 | Temporal MS-based proteomic HPI studies combine advanced experimental platforms, instrumentation, and software to study alterations in protein abundances over the course of an infection. Following infection, cells (whole-cell temporal proteomics) or organelles (organelle temporal proteomics) are harvested at different time points post-infection, followed by lysis and proteolysis. The resulting peptides are then analyzed by label-free or multiplexed label-based (e.g., TMT or SILAC) quantitative MS (for a review, see Jean Beltran et al., 2017). Software, such as MaxQuant can be used to process the data and generate a multidimensional matrix that contains information about protein identities (N) and their relative abundances across X time points in infected vs. uninfected samples. This is then used to assess temporal changes in cell proteome during infection.

the intensity of each particular protein to mean, median, or overall sum of all intensities across all channels/samples (Jean Beltran et al., 2016; Soday et al., 2019; Nusinow and Gygi, 2020). For instance, the "total count" approach aims to equalize protein loading across all channels/samples using a normalization factor, which is the sum of all intensities (or spectral counts) in a given MS run.

However, the application of a more sophisticated normalization method would be welcome when comparing multiple experimental conditions (e.g., "infected" vs. "uninfected") to identify differentially expressed proteins. When multiple conditions are included in the analysis, the "total count" approach can bias the results to be skewed toward one experimental condition if proteins between biological conditions are disproportionately represented. The TMM ["weighted trimmed mean of M-values (i.e., log-intensity ratios)"] normalization may provide a more robust approach to calculate a normalization factor (Robinson and Oshlack, 2010). The TMM normalization method is based on the hypothesis that most features (e.g., proteins) are not differentially expressed. It selects one sample as a reference and computes a TMM normalization factor for the remaining non-reference samples (Robinson and Oshlack, 2010). This approach is employed in the edgeR Bioconductor package as the default normalization method (Robinson et al., 2010). Like TMM, "relative log expression" (RLE) normalization assumes that most genes or proteins are not differentially expressed. For a given sample, the RLE scaling factor is determined by dividing the observed counts of each feature (e.g., protein) by its geometric mean across all samples. This normalization method is included in the DESeq and DESeq2 Bioconductor packages (Anders and Huber, 2010; Love et al., 2014). The TMM and RLE have been thoroughly

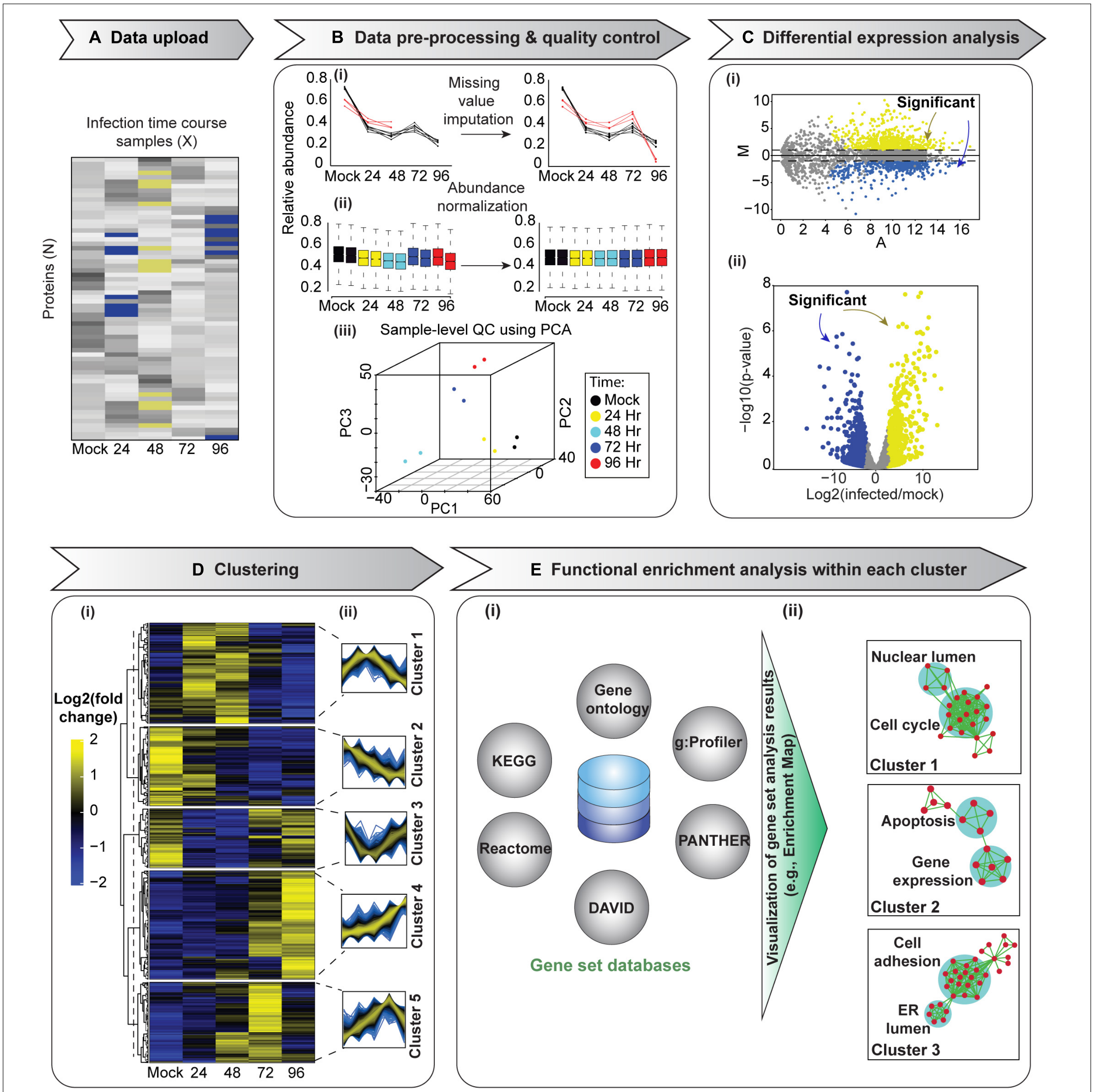


FIGURE 2 | A schematic of temporal proteomic data analysis pipeline. After upload **(A)**, data pre-processing and quality control steps are performed **(B)**. These include the imputation of missing values (i), normalization (ii), and sample-level quality control, e.g., using principal component analysis (PCA, iii). **(C)** Differential expression analysis can then be performed to identify proteins with significantly altered expression between states, for example, different relative (infected/mock) expression between time points. Results can be visualized in (i) an MA plot [log fold change, M, vs. log of the mean expression level between conditions, (**A**)] or in (ii) a volcano plot. Subsequently, clustering analysis can be performed to group proteins with similar temporal expression patterns **(D)**. Here, we applied hierarchical clustering, which is displayed in conjunction with a heat map visualization of the clustered data. The dendrogram was cut at the level indicated by the dashed line to yield five clusters. (ii) Each cluster's temporal profiles can be visualized in a simple plot with relative abundances (y-axis) of proteins within each cluster across all time points (x-axis). Finally, functional enrichment analysis of each of these clusters could provide information on pathways and cellular processes rewired in response to an infection **(E)**. (i) Groups for the analysis can be derived from annotations (e.g., gene ontology; Gene Ontology Consortium, 2004), a pathway database (e.g., KEGG; Kanehisa et al., 2012, Reactome; Croft et al., 2011), or a combination of these (e.g., DAVID; Sherman et al., 2007, PANTHER; Mi et al., 2013, and g:Profiler; Reimand et al., 2007). (ii) Since many inter-dependent gene sets may be enriched, organizing results into a network, e.g., by means of EnrichmentMap (Merico et al., 2010) can be useful. Here, gene sets that share many proteins are grouped together, thereby offering an intuitive visualization of the results. We used 1,000 randomly selected proteins from Weekes et al. (2014) experiment 1 to exemplify data analysis steps.

investigated. They effectively eliminate biases in RNA sequencing data due to unequal library size (Dillies et al., 2013), as well as in proteomic data (Branson and Freitas, 2016). They have been developed for spectral count data but should be applicable to intensity measurements as well. Indeed, normalization methods applicable to count data (e.g., RNA sequencing, peptide counts) have been successfully used with proteomic intensity measurements (Gatto et al., 2014a). Note that both, spectral counts and ion intensity values, should be $\log_2$ transformed [i.e., $\log_2$(value+1)] to normalize their distributions prior to using DESeq2 Bioconductor package (Huang et al., 2020). Log2 normalization is included in edgeR package, so it is not required prior to TMM normalization by means of edgeR package.

The selection of the optimum normalization approach is greatly dependent on the experimental design. The efficiency of each normalization method can be evaluated using box (**Figure 2B**ii) or MA (log fold change, M, vs. log of the mean expression level between conditions, A) plots (**Figure 2C**i). The MA plots are a convenient pairwise representation of conditional data commonly employed in proteomics (Breitwieser et al., 2011; Gatto and Lilley, 2012). After appropriate normalization, most data points in an MA plot will be clustered around $M = 0$, while points away from $M = 0$ identify differentially expressed proteins (**Figure 2C**i; Chen et al., 2020). Moreover, if data preprocessing and normalization were effective, patterns are expected to emerge in subsequent analyses steps (see below).

Following data normalization, visualizing entire dataset in one figure is often needed to evaluate data quality and structure. Principal component analysis (PCA) allows the visualization of high-dimensional data in a reduced set of dimensions, generally in two or three, while retaining as much of the initial information as possible (**Figure 2B**iii). It does this by transforming correlated variables into fewer uncorrelated variables called principal components (PCs), which are then arranged according to the amount of variability described in each component. The first PC accounts for the most variation in the original data, the second PC accounts for most of the residual variation, etc. (Lever et al., 2017). Each succeeding component represents as much of the remaining variability as possible: it represents more variability than the PC after it and less than the one before, with the last PCs containing mostly noise and very little information. Therefore, well-structured datasets could be visualized as a projection of the first two or three PCs in a 2-D or a 3-D plot (**Figure 2B**iii); such a plot is a simple yet informative depiction of the whole dataset. If data structure exists, proteins with similar informative characteristics (e.g., with similar functional annotations) will be grouped together in a 2-D or a 3-D plot, and dissimilar groups (i.e., with divergent unrelated functions) would be located far from each other. However, if normalization and the correction of technical artifacts were insufficient, proteins from different technical replicates may form distinct clusters instead.

PCA is an unsupervised machine learning technique, meaning samples are not associated with a class label. Instead, a pattern in the graphical representation results from similarities in attributes. The use of an unsupervised PCA clustering without external information is an efficient quality control and data analysis approach. For example, PCA helps detect the presence of batch effects and outliers: it will reveal how replicates cluster together. If, for example, a batch effect is apparent, normalization was not sufficient and needs to be adjusted. Furthermore, PCA allows an unbiased representation of the main patterns in the data before biologically relevant parameters are mapped (e.g., subjects clustering in line with predicted treatment group) (**Figure 2B**iii). If no structure is evident, one would not expect well-defined temporal clusters, and hence, the statistical inferences from such data will be challenging. PCA has been widely applied in the analysis of proteomic data (Purohit and Rocke, 2003; Hou et al., 2017; Itzhak et al., 2019; Santana-Codina et al., 2020), as well as in temporal proteomic HPI studies (Diamond et al., 2010; Weekes et al., 2014; Greenwood et al., 2019). However, if more than three PCs are required to describe at least 60% of variance in the data (Hair et al., 2009), other dimensionality reduction tools must be used to visualize the dataset (see sections "Self-Organizing Map" and "Dimensionality Reduction Tools for Visualizing Organellar Map"). The stats (RStudio Team, 2020) package in R has two built-in functions including prcomp() and princomp() that can be used to perform PCA analysis. Other packages, such as factoextra (Kassambara and Mundt, 2020) and ggfortify (Tang et al., 2016) also offer users ggplot2-based elegant visualization capabilities.

Statistical Analysis of Quantitative Temporal Proteomics Data

Following data pre-processing, the next step is to accurately identify proteins with significantly different expression between samples. The outcome of differential analysis is often visualized *via* MA or Volcano plots (**Figures 2C**i,ii; Kucukural et al., 2019). The identification of differentially expressed proteins across time points can help pinpoint predictive factors or biomarkers for disease. A traditional *t*-test; its nonparametric equivalent, the Wilcoxon test; or the analysis of variance (ANOVA) are the most standard approaches to delineating significantly altered protein abundances (Diamond et al., 2010; Weekes et al., 2014; Itzhak et al., 2019; Soday et al., 2019). Another method to identify differentially expressed proteins from both label-free and labeling-based experimental approaches is offered by MSstats. MSstats is a Bioconductor package in R environment that relies on linear mixed models. It takes the data input, detects study design, and, based on the design, fits an appropriate linear mixed model to discover differentially expressed proteins (Choi et al., 2014). The MSstatsTMT Bioconductor R package is also available for applying Empirical Bayes procedure in R to detect differentially expressed proteins in experiments with isobaric labeling (Huang et al., 2020). Furthermore, the DEP Bioconductor package in R can be used to detect differentially expressed proteins from both label-free and labeling-based experimental approaches (Zhang et al., 2018).

However, in a shotgun discovery-based proteomics experiment, sample sizes are often small (<10), which results in ambiguity in the estimation of variability. This, in turn, may result in a non-significant *p*-value for proteins with a substantial fold change due to large sample variance and a significant *p*-value for proteins with small fold change due to small sample variance.

To overcome this issue, Kammers et al. (2015) improved the detection of differentially expressed proteins by applying moderated *t*-test statistics from an empirical Bayes method called "Linear Models for Microarray Data" (limma). Limma adjusts variances toward a pooled estimate based on all sample data. This results in a more powerful detection of differences, especially for experiments with a relatively small sample size (Smyth, 2004) while still allowing a distribution of variances (Kammers et al., 2015). Limma is available as a Bioconductor package in R (Ritchie et al., 2015), and has been frequently used for proteomics data analyses (Brusniak et al., 2008; Margolin et al., 2009; Ting et al., 2009; Schwämmle et al., 2013; Zhao et al., 2013; Greenwood et al., 2019). DESeq (Anders and Huber, 2010) and edgeR (Robinson et al., 2010) are Bioconductor R packages suitable for differential expression analyses with relatively small proteomic datasets (Branson and Freitas, 2016). Differential expression can likewise be assessed by the generalized linear mixed-effects model (GLMM) (Choi et al., 2008), linear mixed-effects model (Hill et al., 2008), or quasi-likelihood modeling generalized linear model (GLM) (Li et al., 2019).

Isobaric-labeling based MS approaches are suffering from "ratio compression," i.e., the estimated protein abundance ratio level across samples is typically underestimated. This occurs due to interference with quantification from co-fragmented peptides (Rauniyar and Yates, 2014). This undermines the ability of isobaric labeling to be genuinely quantitative (Li et al., 2019). Methods exist to correct for "ratio compression" (Savitski et al., 2013), but detailed discussion of the topic is outside the scope of this manuscript. Several statistical models have been developed to accurately measure and correct the technical variability of isobaric-labeling based MS approaches (Huang et al., 2020). Some of these models depend on either prior knowledge or separate experiments to evaluate noise levels in the data (Zhang et al., 2010; Breitwieser et al., 2011; Zhou et al., 2012, 2014a). "Model-based analysis of proteomic data" (MAP) is also available for label-based experimental workflows. Unlike earlier algorithms that required technical replicates to determine experimental error and identify proteins with significantly altered expression, MAP uses regression analysis to calculate local error. These error estimates are then employed in the detection of proteins with significantly altered expression without the need for technical replicates to model technical and systematic errors (Li et al., 2019). In comparative analyses, MAP outperformed other, replicate-based algorithms (Zhang et al., 2010). It is therefore the currently preferred tool for the analysis of label-based proteomics data.

For all statistical tests above, it is essential to correct for multiple hypothesis testing, as many tests are conducted simultaneously. This can be done by controlling the false discovery rate (FDR). Here, FDR is calculated and then a selected threshold is applied. Otherwise, the number of false positives will be increased with the number of tests. Benjamini-Hochberg procedure (Benjamini and Hochberg, 1995) and permutation-based FDR estimates (Tusher et al., 2001) can be used to efficiently compute FDRs from *p*-value s to reduce the number of false positives. Such approaches can be performed using the p.adjust() function in R language (RStudio Team, 2020). In order to further remove background noise and identify the most significant differentially expressed proteins, fold-change can be used along with adjusted *p*-value cutoff. Fold change calculation is typically followed by log2 numerical transformation to center the distribution of the resulting values around 0. In all cases, follow-up studies are required to accurately describe biological phenomena underlying the detected changes (Dalman et al., 2012).

Clustering Analyses

Identifying individual proteins differentially expressed between time points or conditions is often insufficient for extracting biologically relevant information from a proteomics experiment. Instead, grouping similar items (i.e., proteins or samples/conditions) may be necessary to enable the exploration of data patterns without getting lost in lists (**Figure 2D**). The biological basis for this is that proteins often act in groups and the expression of proteins participating in the same processes may be co-regulated (Do and Choi, 2008). Clustering can help with the necessary data grouping. Clustering is a machine learning technique used in pattern recognition, data mining, and bioinformatics. The goal of clustering is to distinguish similar from different. Specifically, similar items (e.g., proteins) are grouped into the same cluster based on a common parameter or parameters (e.g., relative fold expression change across time), while different items are found in distinct clusters.

There are two types of clustering: supervised and unsupervised. In supervised clustering, class labels are provided and are used to guide learning (see section "*K*-means Clustering"). In contrast, in unsupervised clustering, observations are not associated with class labels. Unsupervised algorithms are primarily used for pattern discovery. For example, they can be used for exploratory data analysis, where the aim is to generate hypotheses rather than verify them.

Clustering of temporal proteomic HPI data is primarily performed by unsupervised learning, due to the lack of information about known expression patterns at different time points. A fundamental weakness of unsupervised approaches is that they assume there is an underlying pattern within the data; therefore, outputs from such methods should be carefully statistically and experimentally validated (Do and Choi, 2008). Temporal quantitative proteomic HPI data can be most appropriately clustered based on protein expression differences (see section "Statistical Analysis of Quantitative Temporal Proteomics Data"). For instance, relative fold expression change could be included in the data matrix used for clustering (**Figure 2D**). Meunier et al. (2007) demonstrated the application of unsupervised hierarchical clustering for proteomic data mining and its potential for characterizing tumor samples. Likewise, clustering is frequently used in temporal proteomic studies to uncover temporal trends and molecular signatures for infectious disease (Olsen et al., 2006; Weekes et al., 2014; Yang et al., 2015; Hou et al., 2017; Lapek et al., 2017; Itzhak et al., 2019; Soday et al., 2019; Hashimoto et al., 2020).

Clustering of a matrix containing relative fold expression changes for N proteins across X time points can be achieved in one of three ways (Oyelade et al., 2016). One option is to cluster proteins with similar relative fold expression changes across X time points. Here, proteins are considered objects, while samples are regarded as features. The goal in this case is to identify groups of proteins with similar pattern of relative protein expression change across time. Such groups may indicate co-function or co-regulation (Thalamuthu et al., 2006). Another option is to cluster samples/time points across all proteins. In this case, samples are regarded as objects and proteins are regarded as features, and the goal may be to reveal, for example, the phenotypic structure of samples (e.g., cyclic changes with groups of samples from different time points exhibiting similar changes). The third option is to cluster the data matrix along both, protein and sample axes.

Grouping similar items into the same clusters and dissimilar items into different clusters requires ways to measure the (dis)similarity or distance between each pair of items. This is accomplished by means of distance (aka proximity, dissimilarity, or similarity) measures, which are at the core of distance-based clustering algorithms. The performance of the clustering algorithm depends on the efficiency of its distance measures, and the results may change depending on the distance measure (Shirkhorshidi et al., 2015) and the algorithm. The choice of the distance measure and the clustering algorithm depends on the dataset (e.g., if the data is log transformed). For example, Gibbons and Roth (2002) reported superior performance of Euclidean distance measure for ratio-based data and Pearson distance measure for non-ratio measurements. The most commonly used distance measures are: Minkowski, Euclidean, Manhattan, Cosine, Pearson correlation, and Spearman correlation distances. Their performance with high-dimensional datasets has been extensively reviewed elsewhere (D'haeseleer, 2005; Brusniak et al., 2008; Kerr et al., 2008; Shirkhorshidi et al., 2015).

Understanding clustering algorithms is a prerequisite for their proper application to the clustering of temporal HPI proteomics data. Clustering algorithms can be classified by a number of parameters (Oyelade et al., 2016). For example, Clustering algorithms can be categorized as exclusive (hard, or crisp) or overlapping (soft). Exclusive clustering assigns each input item (e.g., protein) to a single cluster, whereas overlapping (soft) clustering allows a data point to belong to more than one group (Kerr et al., 2008). In the remainder of this section, we discuss the principles of selected unsupervised clustering algorithms. We focus on algorithms frequently used in the analysis of temporal proteomic HPI data as well as on the algorithms we believe to be particularly useful for the analyses of such data.

Hierarchical Clustering

In hierarchical clustering, all proteins are joined into clusters that form a nested dendrogram (aka a tree-shaped data structure, **Figure 3A**, right panel). The dendrogram reflects how similar or different objects (i.e., proteins and/or samples/time points) are across all features. The most similar objects are connected by clusters near the tree's terminal branches (i.e., leaves. **Figure 3B**); root cluster connects objects that are most different. Therefore, in a hierarchical cluster, rows and/or columns, depending on the analysis option, are re-ordered placing similar objects close to each other. The data are transformed to color scale to help visualize the matrix (**Figures 2D**, **3A**). The tree can be cut at varying levels to obtain the desired number of clusters (**Figure 3A**).

Hierarchical clustering algorithms work on distance measure matrices, which are calculated for each pair of objects using the input data matrix and the selected distance measure. Depending on how the clusters are formed, hierarchical clustering can be classified as agglomerative or divisive. Agglomerative clustering, also called agglomerative nesting (AGNES), or bottom-up clustering, works from the bottom up. It starts by treating individual objects as clusters, followed by computing distance measures between all pairs of clusters and then recursively joining the closest pairs according to their distance until a single cluster is made (**Figures 3B,C**). Here, intercluster distance, or "linkage function," determines how distances between clusters are calculated, and which clusters are connected. The most common linkage functions are: minimum/single, maximum/complete, average/UPGMA (unweighted pair-group method using arithmetic averages), and centroid/UPGMC (unweighted pair-group method using centroids) (D'haeseleer, 2005). The distance between two clusters per minimum linkage function equals the distance between the closest two members of each of the two clusters; conversely, per maximum function it equals the distance between the furthest two members of each of the two clusters. Average and centroid functions calculate the distance between any two clusters as average distance between cluster members and as the distance between cluster centroids, respectively (D'haeseleer, 2005). Unlike agglomerative clustering, divisive clustering, aka divisive analysis (DIANA), is a top-down approach. It starts by including all objects in a single cluster and works to iteratively split the most heterogeneous cluster into components. This is repeated until all clusters are made up of single indivisible objects (i.e., proteins, **Figure 3B**; Karimpour-Fard et al., 2015). There are several methods for splitting clusters, see Roux (2018) for review. Agglomerative clustering is strongest at identifying small clusters and may deliver suboptimal performance in the detection of large clusters. Conversely, divisive clustering is best at identifying large clusters (Chipman and Tibshirani, 2006). To combine the strengths of the two approaches, Chipman and Tibshirani (2006) proposed a combined "mutual cluster" approach that informs the divisive approach by the results of an agglomerative analysis.

R has some useful built-in functions for performing hierarchical clustering. For instance, the hclust() function in stats R package (RStudio Team, 2020) and agens() function in cluster package (Maechler et al., 2019) are commonly used to perform agglomerative hierarchical clustering. Both functions include parameters that allow one to select the appropriate linkage and distance measures. Divisive clustering is often performed using diana() function in cluster package (Maechler et al., 2019). The number of generated clusters (**Figure 3A**) can be controlled by the cutree() function in stats package (RStudio Team, 2020). The mutual cluster approach is also available as mutualCluster() function in hybridHclust package in R (Chipman and Tibshirani, 2006).

The main weakness of hierarchical clustering lies in the dependence of results on various parameters, including distance

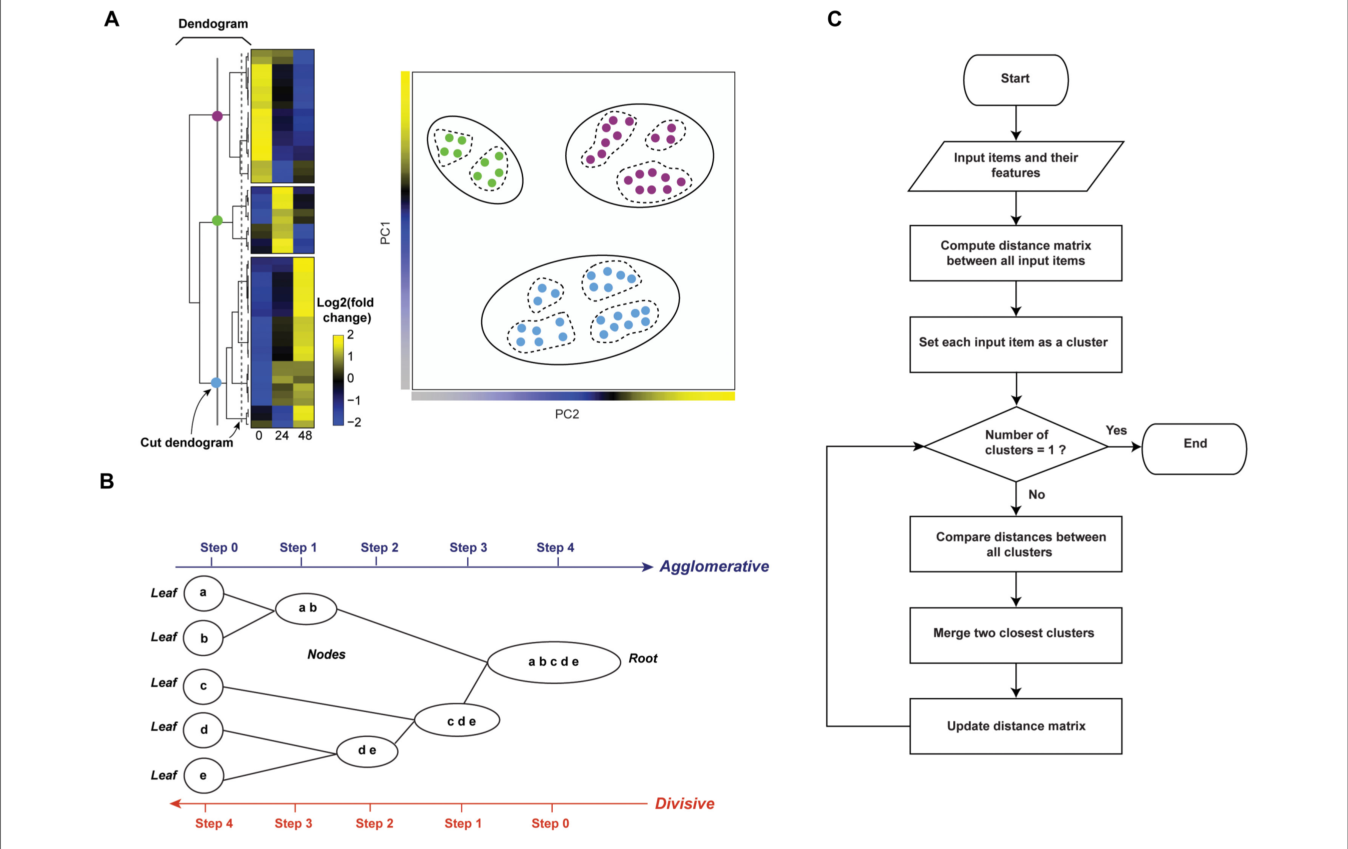


FIGURE 3 | Hierarchical clustering. **(A)** Relative abundances of 50 proteins across three time points were clustered by hierarchical clustering (left panel). The dendrogram reflects relationships between proteins. The tree was cut at two levels indicated by colored circles and a dashed line to yield 3 and 9 clusters, respectively. The separation of subclusters is illustrated in the PCA plot in the right panel. Here, each dot represents a protein. **(B)** Agglomerative and divisive are the two types of hierarchical clustering. The agglomerative works in bottom-up manner, recording the sequence of cluster merges. The divisive algorithm works in a top-down manner, recording the sequence of cluster splits. **(C)** Flowchart of unsupervised hierarchical clustering algorithm (agglomerative).

measures and algorithm type. As a consequence, there is no one correct and true result (Oyelade et al., 2016). Therefore, the parameters of hierarchical clustering need to be tuned and the resulting clusters must be validated (see section "Evaluation Measures for Temporal Clustering"). Moreover, calculations can be computationally intensive, but previous steps (e.g., erroneous merging/division decisions) cannot be undone (Karimpour-Fard et al., 2015).

Despite its weaknesses, hierarchical clustering is a method of choice for visualizing and exploring large datasets, including temporal proteomic HPI data (Diamond et al., 2010; Weekes et al., 2014; Jean Beltran et al., 2016; Hou et al., 2017; Soday et al., 2019). For example, a study of Vaccinia Virus (VACV), the causative agent of smallpox, used multiplexed proteomics to quantify the changes of viral and host proteomes over a series of time points in infected vs. mock samples (Soday et al., 2019). Subsequently, agglomerative hierarchical clustering with centroid linkage and Pearson correlation distance measure revealed clear separation between infection stages, with the greatest changes occurring at late time points. Moreover, protein-level clustering identified multiple dysregulated pathways. This included, for example, the down-regulation of proteins involved in "cell attachment site" during infection, which suggested that VACV targets cell surface proteins to evade host's defensive immune responses. Thereby, clustering helped make insights into biological processes modulated by VACV infection.

K-Means Clustering

K-means clustering is an iterative algorithm that partitions the dataset into a predetermined *k* number of clusters, in a way that intra-cluster and inter-cluster similarities are maximized and minimized, respectively (D'haeseleer, 2005). The algorithm first shuffles the dataset and then randomly chooses *k* patterns as initial centroids for each cluster. After that, each data point (in our case, protein) is assigned to the cluster by finding the pattern's closest centroid (a centroid is a data point at the center of a cluster) using proximity measures (e.g., Euclidean distance). The new centroid is then computed for each cluster by taking the average of all proteins assigned to that cluster. This process repeats until no more proteins change the cluster (**Figures 4A,B**i).

To exemplify the application of *K*-means clustering to temporal HPI proteomics data, we randomly selected 3,000 host proteins from experiment 1 data by (Weekes et al., 2014).

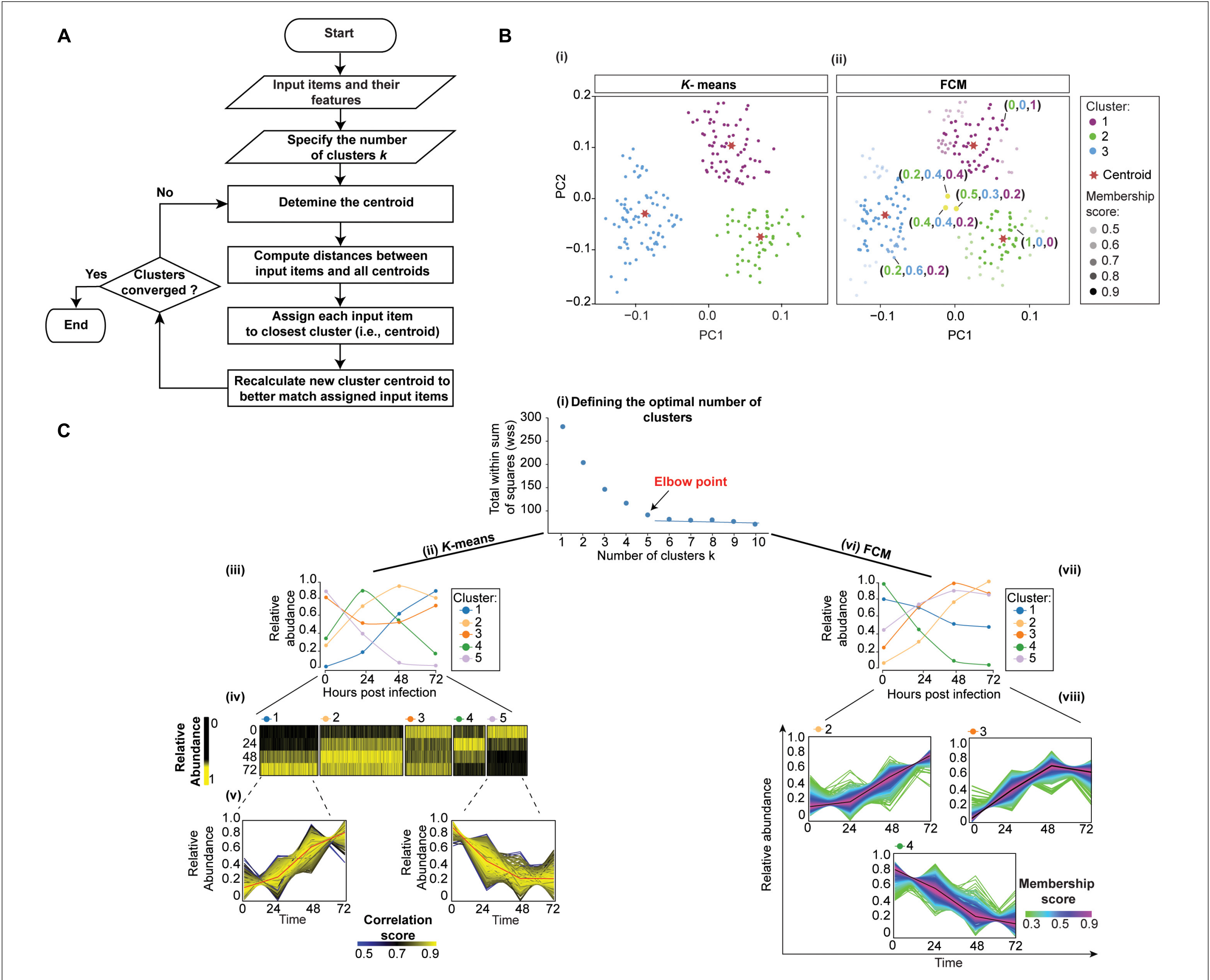


FIGURE 4 | Clustering temporal HPI data using *K*-means (crisp clustering) and Fuzzy C-means (FCM, soft clustering) methods. **(A)** Flowchart of the unsupervised *K*-means algorithm. Initialization, iterations, and termination in FCM are the same as in the *K*-means algorithm. However, FCM uses a weighted centroid based on memberships of data points within each cluster because membership scores can vary from 0 to 1. **(B)** We illustrate the difference between *K-means* (i) and FCM (ii) by using mock temporal HPI data with 3-time points and principal components to display the data. (i) *K*-means divides data into three distinct clusters. (ii) On the other hand, FCM assigns each data point coefficients that reflect memberships in each of the clusters and range from 0 to 1. Then, each data point is assigned to the cluster in which its membership is highest. Some proteins (yellow dots) are assigned high membership coefficients to more than one cluster. **(C)** To demonstrate the application of *K*-means and FCM, we randomly selected 3,000 proteins from experiment 1 data by Weekes et al. (2014) and used the Elbow method to define 5 as the optimal number of clusters for both *K*-means and FCM; the plot for *K*-means is shown (i). We then used *K*-means (ii–v) and FCM (vi–viii) to cluster the data. The results of the *K*-means clustering are displayed (iii) as centroid (or average) temporal profiles of each cluster, and (iv) as heatmaps. (v) Each cluster can also be visualized by displaying the data for all proteins in the cluster, and coloring the corresponding protein profiles based on how well they correlate with the cluster's centroid (red line); examples for clusters 1 and 5 are shown. Here, proteins with similar profiles to the cluster's centroid have a score approaching 1 (yellow), while those with divergent patterns have a correlation score closer to 0 (blue). This plot can help filter proteins whose profiles don't fit well into the selected cluster or identify core proteins of particular interest with profiles closely matching the centroid. Likewise, FCM results are displayed as the centroid temporal profile of each cluster (vii). (viii) Clusters can also be visualized by displaying each protein and color-coding its profile according to its membership in the respective cluster; plots for clusters 2, 3, and 4 are shown. Here, proteins with a profile close to the cluster's centroid (black lines) have a membership score close to 1 (purple). These proteins are prominent members of a cluster (viii). In contrast, proteins with divergent patterns have a score closer to 0 (green). One may wish to apply a membership score threshold to eliminate proteins with divergent patterns from the downstream analyses of the cluster.

The first step in *K*-means clustering is to define the optimal number of clusters, which is critical for generating biologically meaningful groupings (Yang et al., 2015). For instance, the overestimation of parameter *k* will partition related proteins into different clusters, thereby confounding downstream inferences (Yang et al., 2015). The two most commonly used *k*-value selection algorithms are the Elbow method and the Average silhouette method (Yuan and Yang, 2019). The Elbow method, for instance, measures the variability within each cluster (i.e., within cluster-sum of a squares, WSS) as a function of the number of clusters (**Figure 4C**i; Yuan and Yang, 2019). One should choose the number of clusters at but not after elbow point (**Figure 4C**i). Here, we used fviz_nbclust() function in factoextra R package (Kassambara and Mundt, 2020) to define 5 as the optimal number of clusters (**Figure 4C**i). We then applied kmeans() function in stats R package (RStudio Team, 2020) to cluster host proteins into five distinct clusters (**Figure 4C**ii). To observe the temporal profiles of each cluster, one may wish to visualize centroid profiles (**Figure 4C**iii). Furthermore, temporal profiles of each group can be visualized using heatmap *via* heatmap() function in stats R package (RStudio Team, 2020) (**Figure 4C**iv). To identify core proteins whose expression closely matches the cluster's centroid, one can correlate each protein in the cluster with the cluster centroid (**Figure 4C**v).

K-means is one of the most straightforward clustering methods due to the ease of programming and computational efficiency. However, one of its drawbacks is that it generates hard and unrelated clusters. Therefore, it is not suitable for expression datasets containing overlapping clusters, or for assessing between-cluster relationships (Oyelade et al., 2016). Moreover, it works well in capturing the structure of the data if clusters have a spherical-like shape but performs poorly if clusters have complex geometric shapes. Therefore, it is not a good candidate for high-dimensional data and highly connected clusters (Oyelade et al., 2016). Furthermore, since *K*-means clustering is sensitive to outliers, it is always appropriate to remove the outliers before clustering. Additionally, *K*-means clustering is sensitive to initialization, meaning that the final clustering result depends on the position of the initial cluster centroid (i.e., seed). Therefore, it is essential to run the algorithm several times by applying different random seeds, or use an advanced version of *K*-means. For example, *K*-means++, available in pracma R package (Borchers, 2019), which runs a preliminary iterative algorithm to determine the most appropriate initial seeds.

K-means algorithm has been used in the analysis of temporal proteomic HPI data, rendering intuitively clear summaries of temporal patterns (Weekes et al., 2014; Clements et al., 2017; Hou et al., 2017; Lapek et al., 2017). For example, the aforementioned Soday et al. (2019) used *K*-means to cluster temporal profiles of co-expressed viral proteins. Subsequent functional enrichment of clusters revealed the co-expression of viral proteins involved in "Host interaction" and "DNA replication" at early time points, followed by virion-associated proteins at later times. Therefore, these temporal clusters revealed patterns in the expression of proteins with specific biological functions. Furthermore, the functions of uncharacterized pathogen proteins may be inferred *via* their associations with known proteins in their respective clusters, or by analyzing their expression in the context of changes occurring within the host. For instance, the authors compared viral temporal profiles with the inverted temporal profile of the host's cellular protein HDAC5 using Euclidean distance. This matched C6 viral and HDAC5 host profiles, suggesting that C6 targets HDAC5 for proteasomal degradation. This was subsequently experimentally confirmed. Therefore, *K*-means clustering helped gain direct insights into viral-host interactions.

Fuzzy Clustering

Although hard clustering methods, including hierarchical and *K*-means, can accurately group distinct expression patterns, they are unable to identify input items (e.g., proteins) with similarities to multiple distinct clusters (Gasch and Eisen, 2002). For example, inaccurate clusters may be produced in the analysis of large datasets, where a protein has expression pattern similar to one group in one set of biological samples and another group for the remaining samples.

Several fuzzy clustering algorithms, including the fuzzy C-means clustering (FCM), have been developed to deal with such complicated relationships between objects (Bezdek et al., 1984, 1999; Friedman et al., 2000; Oyelade et al., 2016). FCM is very similar to *K*-means, and is likewise widely used. As in *K*-means, the number of clusters must be pre-determined. However, unlike *K*-means, FCM does not simply assign an input item (e.g., a protein) to a single cluster. Instead, FCM facilitates the identification of overlapping groups of data points by allowing each input item to belong to more than one cluster with a probability (membership coefficient) ranging from zero to one (**Figure 4B**ii). Proteins whose expression patterns are very similar to the center (or centroid) of a cluster will be assigned a high membership in that cluster (i.e., 1). Conversely, proteins that lie far away from the center of the cluster (i.e., with little similarity to the centroid) will have a low degree of membership to that cluster (i.e., 0). Therefore, FCM reveals the relative degree of each input item belonging to each of the clusters (**Figures 4B**ii,**C**viii).

Like *K*-means clustering, FCM is sensitive to initialization and is affected by initial parameter values. One of these parameters is the *c*-value (Kerr et al., 2008; Oyelade et al., 2016), which defines the number of clusters and is equivalent to the *k*-value in *K*-means clustering. Approaches like the Elbow method are recommended to determine the optimal number of clusters. Another parameter is the fuzziness parameter *m* (Dembéle and Kastner, 2003). This parameter needs to be tuned, and one should apply validity indices to determine the optimal value of *m* (Zhou et al., 2014b). FCM is available as cmeans() in e1071 (Meyer et al., 2020), fanny() in cluster (Maechler et al., 2019), fcm() in ppclust (Cebeci, 2019), and mfuzz() in Mfuzz R packages. To demonstrate the application of FCM, we used mfuzz() function in Mfuzz R package (Kumar and Futschik, 2007) with the default value for parameter *m* to re-analyze data used for *K*-means clustering (**Figure 4C**vi).

Despite limitations, FCM is frequently used particularly in phosphoproteomics studies to elucidate the dynamics of phosphorylation signaling events. Indeed, partitioning the identified phosphorylation sites into distinct clusters can help identify corresponding kinases and important regulatory events (Blagoev et al., 2004; Zhang et al., 2005; Olsen et al., 2006; Schmutz et al., 2013; Zhuang et al., 2013). For instance, Schmutz et al. (2013) used LC-MS/MS and FCM to study quantitative phosphorylation changes during *Shigella flexneri* infection. Most of the early phosphorylation events clustered together and were related to the regulation of cytoskeleton and cell adhesion.

Self-Organizing Map

Self-organizing map (SOM; aka Kohonen map, or self-organizing feature map, SOFM) is a an artificial neural network approach for reducing the dimensionality of input data in an organized manner that preserves the similarities between original input items (Kohonen, 1990). Since the output dimensionality is predefined for SOM, it is capable of reducing dimensionality further than PCA. Indeed, 2 or 3 components of PCA may not explain variability, but a SOM may provide a 2-dimensional rendering of relationships within the original data. Furthermore, like *K*-means clustering, SOM may not reproduce each item one-to-one from the original data on the output grid (i.e., mapping may be many-to-one). However, unlike *K*-means clustering, relationships between clusters are preserved. Therefore, SOMs allow for better visualization and interpretation of high-dimensional datasets than *K*-means, FCM, or PCA (Tamayo et al., 1999; Kohonen, 2014).

SOMs are particularly recommended for large and complex datasets, as simpler clustering approaches are available for small datasets (Kohonen, 2013). In other words, if simpler approaches do not produce satisfactory groupings for small datasets (e.g., hundreds of input items), SOMs should be applied. Conversely, with millions of input data items, clustering approaches described above may be less efficient than SOMs, and SOM analyses can be used at the outset. SOMs are useful, for instance, in medical image processing (Chang and Teng, 2007), astrophysics (Naim et al., 1997), industry (Simula et al., 1999), robotics (Sayers, 1991), and data mining. In biology and biochemistry (Tamayo et al., 1999; Olsen et al., 2006), it is frequently used for reducing data dimensionality: high-dimensional data is displayed in a reduced, usually two-dimensional space. In a SOM, similar objects are located close to each other and different objects are far apart, based on the characteristics of the input data. SOMs have been applied in proteomics (Schmidt et al., 2007; Peng et al., 2012) and temporal phosphoproteomics. For example, Zhang et al. (2005) identified modules of phosphorylation sites with similar temporal patterns within the epidermal growth factor receptor signaling network. Moreover, the potential functions of uncharacterized phosphorylation sites were inferred based on the functions of other components within the modules. For instance, hypothetical protein FLJ30532 was predicted to be involved in the immediate-early response to epidermal growth factor stimulation. Likewise, the algorithm can be applied to temporal proteomic HPI studies.

Various SOM variants are available. They all typically require users to specify the size and shape of the grid or array on which the output will be mapped. The size of the grid (with the corresponding number of nodes) needs to be determined by trial and error. If an output with fine details is expected, a larger grid should be used; however, if only coarse details are expected, a smaller grid will suffice. Indeed, too few output nodes (i.e., too few positions in the grid) can result in sizeable intra-cluster variation. In contrast, too many nodes can result in meaningless clusters (Kerr et al., 2008). A good starting point is 50 input data items per node (or location) on the grid (Kohonen, 2013). For example, a grid with 100 nodes (e.g., a 10×10 grid) would be used for five thousand input items. Various array shapes are possible, including a 2D rectangle or hexagon, or a 3D torus. From among 2D shapes, a hexagon is recommended for its visual clarity and accuracy (Kohonen, 2013). Oblong shapes are recommended due to faster convergence in learning with length and width corresponding to the two largest principal components (Kohonen, 2013). Depending on the data, a special shape of the output array may be justified. For example, relationships of data on the edges of a standard 2D rectangular output may not be represented effectively (Kohonen, 2013). Options exist to dynamically modify the shape and the size of the output array depending on the input data (Fritzke, 1994). However, in practice, border distortions should not pose issues for the use of SOM in proteomic data analysis or interpretation (Kohonen, 2014). Moreover, SOM algorithms rely on distance or similarity measures for determining how input items relate to each other. Euclidean distance is an example of a good measure that can be used for clustering proteomics HPI data (Giraudel and Lek, 2001).

There are two main types of SOM algorithms. The first one is the sequential, or step-wise recursive SOM algorithm. The SOM using a step-wise recursive algorithm is constructed as follows (**Figure 5A**). One first selects a size and geometry of the output grid (e.g., a 2D 9×7 rectangle). A model is associated with each node of the grid. These models must be initialized before the algorithm is run (Kohonen, 2012). After SOM algorithm is completed, models essentially become the projections of the original data onto the output grid of the SOM. Random numbers can be used for initializing models. Alternatively, PCs (see PCA, in section "Quantitative Temporal Data Visualization, Preprocessing, and Quality Control") can be used to initialize the models, thus speeding up the analysis. Here, PCs that describe the most variation are used. Subsequently, a random input item (x_i, **Figures 5A,B**) is presented and compared to all the models. The distance between the input item and each model is calculated (e.g., using Euclidean distance) to find the best-matched model, or Best Matching Unit (BMU) (**Figures 5A,B**). The BMU's neighborhood is then identified using a neighborhood function. Right after, BMU and its neighborhood models are updated to better match the input data item (x_i). This is repeated for all input items (x_1–x_n) multiple times (**Figure 5B**). Neighborhood function and learning rate, which defines how adjustments are made to the model(s) at each step, are reduced with SOM algorithm iterations (see Kohonen, 2013). Hence, over time, the amount by which models are adjusted decreases. Therefore, SOM converges to show nonlinear projections of the input items (vectors) on the 2D output array (**Figure 5C**). Moreover, the correlation between each protein in the cluster and cluster's centroid can be computed to identify core proteins whose expression closely matches the cluster's centroid (**Figure 5D**). Several different options for defining learning rates and neighborhoods are available. A nonlinear learning rate and Gaussian neighborhood function typically produce robust results and are good starting points (Stefanovič and Kurasova, 2011).

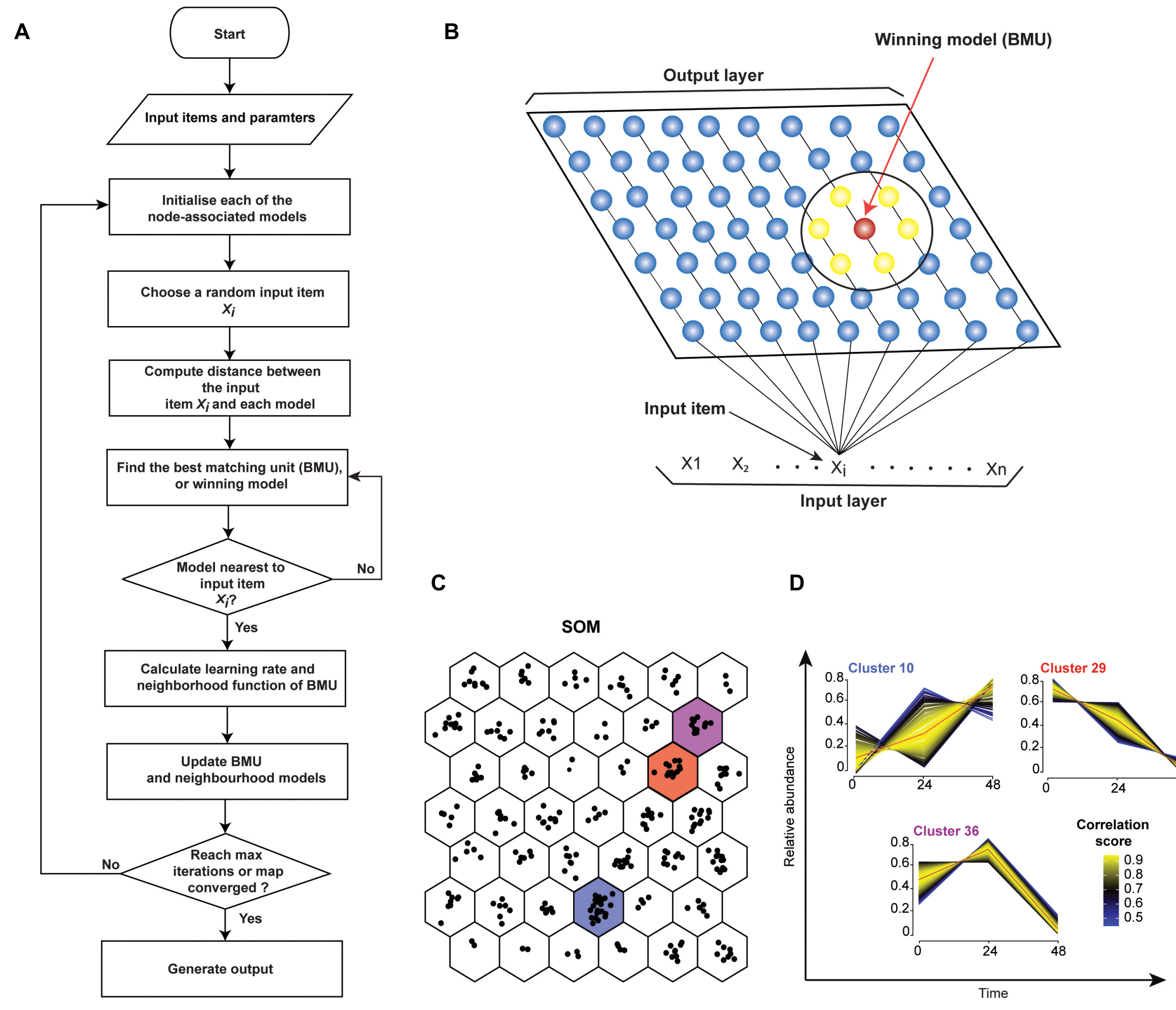

A
Start
Input items and paramters
Initialise each of the node-associated models
Choose a random input item X_i
Compute distance between the input item X_i and each model
Find the best matching unit (BMU), or winning model
Model nearest to input item X_i?
No
Yes
Calculate learning rate and neighborhood function of BMU
Update BMU and neighbourhood models
Reach max iterations or map converged ?
No
Yes
Generate output
B
Winning model (BMU)
Output layer
Input item
X1 X2 Xi Xn
Input layer
C
SOM
D
Cluster 10
Cluster 29
Cluster 36
Relative abundance
Time
Correlation score
0.9
0.8
0.7
0.6
0.5

FIGURE 5 | Self-Organizing Map (SOM). **(A)** Flowchart of a sequential SOM algorithm. **(B)** SOM input and output layers. Input items (x_1 - x_n) will be compared individually to models/nodes. The red model represents the best match, and the surrounding models (yellow, within the larger circle) are its neighbors. **(C)** Mock 3-time-point data was grouped into 42 clusters in a hexagonal 6 × 7 output grid using SOM. Points in each hexagon represent proteins. **(D)** Temporal profiles of proteins from panel (B) clusters 10, 29, and 36 are shown. Profile colors reflect the correlation of the protein profiles to cluster's centroid (red line). This can help to select or filter proteins from clusters. Proteins with similar profiles to the cluster's centroid have the correlation score approaching 1 (yellow), while those with divergent patterns have a correlation score closer to 0 (blue).

The second main type of SOM algorithm is the batch algorithm. There are several benefits to using the batch SOM algorithms for the analyses of proteomic HPI data instead of the sequential SOM algorithm. Firstly, the learning rate parameter is eliminated. Therefore, the result is more robust and less affected by the user's input. Secondly, it is faster than the step-wise method. When running a batch algorithm with Euclidean distance measure, it is recommended that the models be initialized by PCs. This speeds up the completion of the algorithm. In the beginning of the first training cycle of a batch SOM algorithm, all input data items are passed to each of the nodes in a grid. The input items matching each model at each node are saved in association with that corresponding node. Again, neighborhood function defines how and which nodes adjacent to the node with the best matching model will be modified. Then, adjusted model values are calculated for all nodes in all neighborhoods in one concurrent operation. The models are then updated, concluding one training cycle and brining values closer to the equilibrium. For datasets of up to a few thousand nodes, it is generally recommended that a training process incorporate the coarse and the fine training stages (Kohonen, 2014). Here, during the coarse stage a large neighborhood function equaling about 20% of the larger grid dimension is initially used. It should be reduced to about 5% of the smaller dimension over a few dozen training steps. During the fine training stage, the smallest neighborhood function that was reached during the coarse stage should be used. Each of the two training stages usually continues for several dozen steps until equilibrium is reached (Kohonen, 2014).

Some versions of SOMs combine the strengths of artificial neural networks (i.e., speed and robustness to noise) with other types of analyses to improve the overall performance of the algorithm (Dopazo and Carazo, 1997; Dogan et al., 2013). For example, the Self-Organizing Tree Algorithm (SOTA) integrates SOM and hierarchical clustering to quickly produce a dendrogram output while eliminating the need for pre-selecting the size of the output grid (Dopazo and Carazo, 1997; Yin et al., 2006; Kerr et al., 2008). The SOTA algorithm has a binary tree topology structure, in which the algorithm first selects a node with the largest heterogeneity and splits it into two nodes, called daughter cells. The tree's growth continues until all observations (in our case proteins) map onto a unique leaf node (Kerr et al., 2008). The SOTA algorithm is implemented in the clValid R package (Brock et al., 2011). Another algorithm, called the adaptive double SOM (ADSOM) combines SOM and two-dimensional position vector analyses with parameter training (Ressom et al., 2003). This eliminates the biases arising from user-specified parameters while allowing quick and efficient clustering.

Evaluation Measures for Temporal Clustering

The evaluation and validation of clustering are frequently excluded from analysis. However, they are essential for obtaining meaningful clusters (Bhargavi and Gowda, 2015). The quality of clusters can be evaluated computationally by external and/or internal measures. External measures evaluate the quality of clusters by comparing clustering-derived partitions to known labels, e.g., a gold standard dataset. Conversely, internal validation measures evaluate the quality of the clusters without any external data, by simply evaluating intra- and inter-cluster variation. The same internal methods are used to define, for example, the optimal number of clusters in *K*-means clustering (see section "K-means Clustering"). See Handl et al. (2005); Liu et al. (2010), Bruno and Fiori (2013), and Oyelade et al. (2016) for a comprehensive overview of internal and external measures and their biases.

Moreover, several biological measures are available for assessing the ability of clustering algorithms to generate biologically meaningful groupings. The most widely used biological measure is the functional enrichment analysis (**Figure 2E**). It typically assesses the overrepresentation of biologically meaningful categories within clusters, i.e., whether more members of a category belong to a cluster than expected by chance (Boyle et al., 2004). See Huang et al. (2009); Karimpour-Fard et al. (2015), and Chen et al. (2020) for an overview of relevant statistical and bioinformatics methods. Two additional biological measures are the biological homogeneity index, which evaluates the homogeneity of clusters, and the biological stability index, which captures cluster stability through clustering iterations with similar data sets (reviewed in Bruno and Fiori, 2013). The lists of biological categories and the assessment methods are determined by the study's research question as well as the data.

Many temporal proteomics studies utilized functional enrichment analysis to evaluate the quality of clusters (Blagoev et al., 2004; Zhang et al., 2005; Diamond et al., 2010; Weekes et al., 2014; Jean Beltran et al., 2016; Hou et al., 2017; Lapek et al., 2017; Breen et al., 2018; Li et al., 2018; Soday et al., 2019; Itzhak et al., 2019; Hashimoto et al., 2020; Santana-Codina et al., 2020). Conversely, external measures are rarely considered. This could stem, in part, from the lack of suitable gold standard datasets. However, whenever possible, all three measures should be used to evaluate clustering results (Bruno and Fiori, 2013). There is a useful package in R called clValid (Brock et al., 2011) that includes many built-in functions for internal and biological measures. Moreover, this package contains built-in-functions for various clustering algorithms, including hierarchical, *K*-means, SOMs, and FCM. External validation can be performed by external_validation() function in the ClusterR package in R, which also includes a number of clustering algorithms (Mouselimis et al., 2020).

EXPLORING SUBCELLULAR PROTEOME ORGANIZATION DURING INFECTION

Localization of proteins within subcellular niches enables them to find their partners and substrates, and thus become functional. These subcellular niches include macromolecule assemblages, such as the ribosome or centrosome, as well as organelles, which are physically demarcated by a lipid bilayer (Christoforou et al., 2014). These organelles can change their number, localization, structure, and composition in response to an infection or an external stimulus (Jean Beltran et al., 2016). The corresponding compartmentalization of proteins is likewise highly dynamic, enabling a quick response (Borner, 2020). Indeed, controlled protein localization is crucial to cellular hemostasis and its aberrations are associated with disease (Kau et al., 2004; Luheshi et al., 2008; Laurila and Vihinen, 2009; Park et al., 2011; Valastyan and Lindquist, 2014; Beltran et al., 2017; Siljee et al., 2018).

Pathogen-induced localization alterations can occur throughout the course of an infection. These often include the placement of a pathogen's proteins in the compartment as well as the reorganization of the host proteome. These changes must be mapped to understand disease progression. Here, we focus on the spatial proteomics (aka organelle proteomics) that uses fractionation and MS (**Figure 6**, reviewed by Gatto et al., 2010; Borner, 2020). Label-based localization of organelle proteins by isotope tagging (LOPIT) (Dreger, 2003) and label-free protein correlation profiling (PCP) (Foster et al., 2006) are two spatial proteomics approaches. They allow the detection of thousands of proteins in multiple subcellular compartments. Moreover, they improve the detection of low-abundance proteins compared to whole-cell proteomics. By thus increasing the depth of proteome coverage, they can help access specialized pathways manipulated by pathogens.

The output of both of these approaches are relative protein abundances across fractionations. These can be displayed, for example, as a matrix (**Figures 6**, **7A**). Subsequently, pattern recognition is used to assign proteins to specific sub-cellular compartments by relating their localizations to known organelle markers (Gatto et al., 2010). At the same time, actual residents of

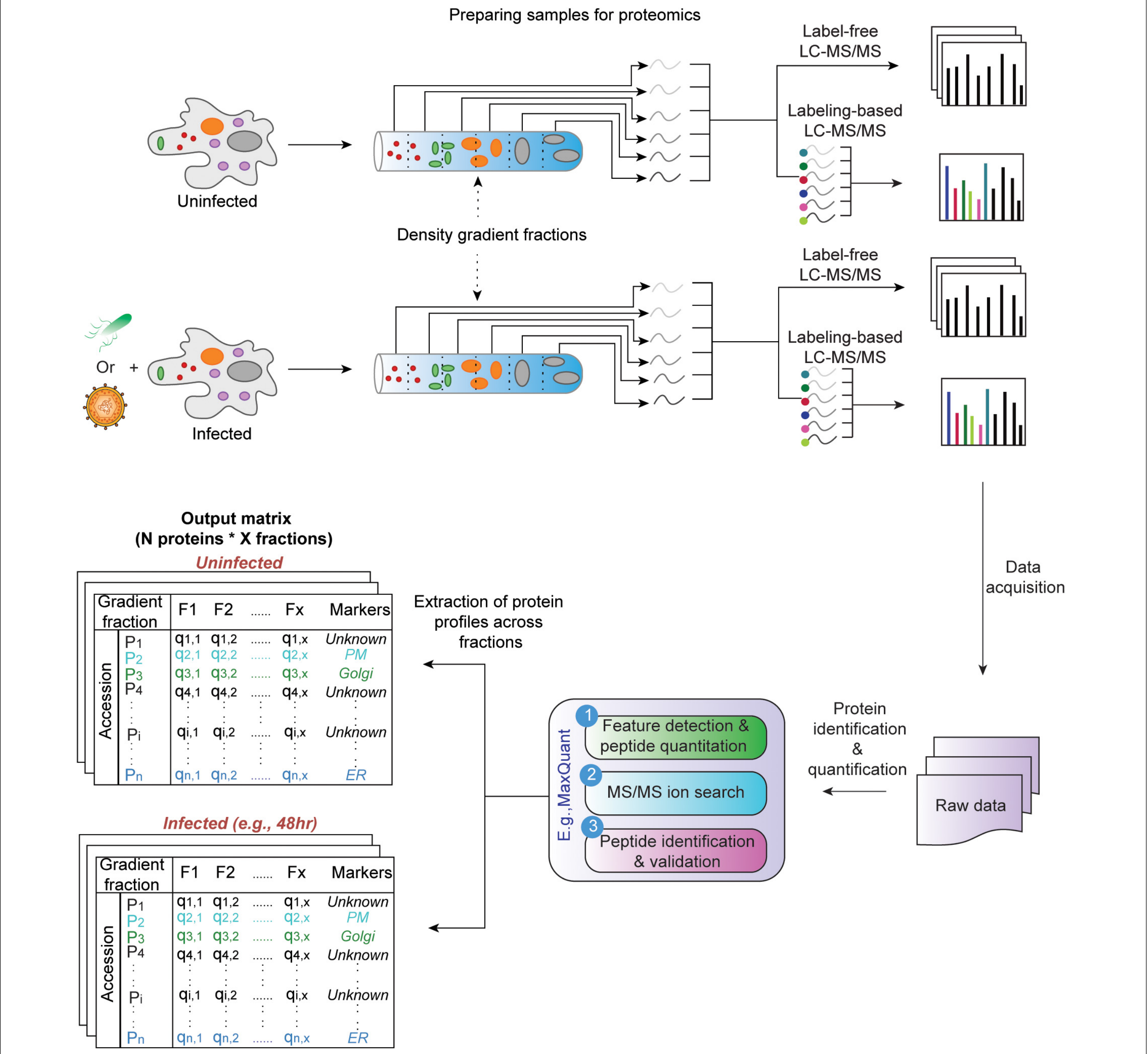

Preparing samples for proteomics
Uninfected
Density gradient fractions
Or +
Infected
Label-free LC-MS/MS
Labeling-based LC-MS/MS
Label-free LC-MS/MS
Labeling-based LC-MS/MS
Data acquisition
Raw data
Protein identification & quantification
E.g.,MaxQuant
1 Feature detection & peptide quantitation
2 MS/MS ion search
3 Peptide identification & validation
Extraction of protein profiles across fractions
Output matrix (N proteins * X fractions)
Uninfected
Infected (e.g., 48hr)
Gradient fraction
Accession
F1 F2 ...... Fx Markers
P1 q1,1 q1,2 ...... q1,x Unknown
P2 q2,1 q2,2 ...... q2,x PM
P3 q3,1 q3,2 ...... q3,x Golgi
P4 q4,1 q4,2 ...... q4,x Unknown
Pi qi,1 qi,2 ...... qi,x Unknown
Pn qn,1 qn,2 ...... qn,x ER

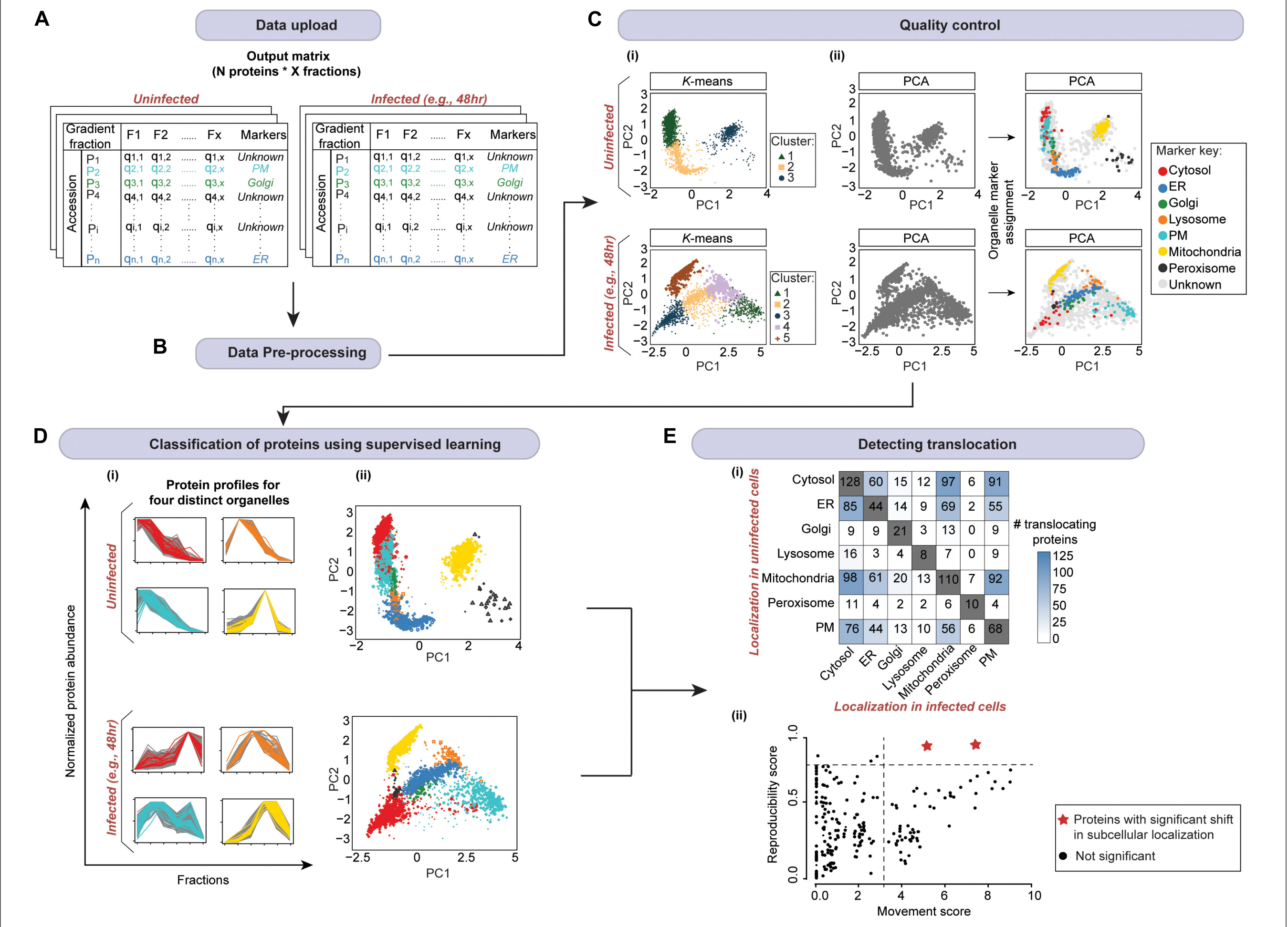
FIGURE 6 | Schematic overview of gradient fractionation-based spatial proteomics. Infected and uninfected cells are lysed, and lysates are fractionated (Borner, 2020). The resulting fractions are then analyzed by label-free or multiplexed label-based quantitative MS. Proteins are then identified and quantified using specialized software (e.g., MaxQuant). This generates a data matrix containing relative protein abundances across fractions for *N* proteins along *X* fractions for infected and uninfected samples. Additional metadata (e.g., manually curated organelle markers) are added to the matrix to facilitate subsequent analyses. "Unknown" in the "marker" column indicates non-marker proteins with unknown localization.

organelles are distinguished from contaminants without the need for high-purity organelle isolations (Gatto et al., 2014a). Below, we present a robust workflow for spatial proteomics data analysis. It is implemented in pRoloc Bioconductor package (Gatto et al., 2014b) in R. Moreover, we update the existing workflows by presenting additional statistical approaches and algorithms, along with their principles, pros, and cons. In addition, we illustrate how these steps work by applying them to *in silico*-generated or published HPI datasets.

Preparatory Steps

Organelle Markers

The prediction of protein localization in spatial proteomics traditionally depends on supervised machine learning (see section "Predicting Protein Localizations in Each Condition"), wherein a list of "bona fide" organelle markers (i.e., proteins with known localizations) retrieved from public databases (i.e., labeled training dataset) is utilized to map proteins of unknown



	Cytosol	ER	Golgi	Lysosome	Mitochondria	Peroxisome	PM
Cytosol	128	60	15	12	97	6	91
ER	85	44	14	9	69	2	55
Golgi	9	9	21	3	13	0	9
Lysosome	16	3	4	8	7	0	9
Mitochondria	98	61	20	13	110	7	92
Peroxisome	11	4	2	2	6	10	4
PM	76	44	13	10	56	6	68

FIGURE 7 | A schematic of spatial proteomic data analysis workflow for mapping localization changes in response to an infection. **(A)** After uploading the data (from **Figure 6**), **(B)** data pre-processing is carried out. This includes the imputation of missing values and normalization. Subsequently, **(C)** quality control is performed. (i) Here, first, unsupervised clustering can be applied to assess the overall data structure. (ii) Then organelle markers can be overlaid on the PCA plot to assess marker separation (ER, endoplasmic reticulum; PM, plasma membrane). **(D)** The organelle markers are then used to train a model (in supervised machine learning). The model will assign the profiles of proteins with unknown localization to organelles based on their similarity to the profiles of markers. Here, we randomly selected 3,000 proteins from Jean Beltran et al. (2016) uninfected and 48 h post-infection samples, and used SVM in pRoloc package in R (Gatto et al., 2014b) to assign protein localizations. (i) Profiles for four organelles from each condition are shown; marker key is same as in panel **(C)**. The plots demonstrate that non-marker and marker proteins that were predicted to colocalize by SVM exhibit similar profiles across the gradient. (ii) The whole multivariate data can then be visualized in two dimensions using PCA or t-SNE to portray organelle separation. Here, all proteins with localizations predicted by SVM as well as markers are colored according to the organelle they are assigned to; marker key is same as in panel **(C)**. **(E)** Finally, localizations in infected and uninfected cells are compared to determine (i) the most affected organelles, and (ii) candidate proteins with significantly shifted subcellular localization (red stars) by computing movement and reproducibility scores for each protein. Note, data from **(D)** was used for panel (**E**i); mock data was used for panel (**E**ii).

localization to subcellular compartments. See Christoforou et al. (2014); Gatto et al. (2014a), and Borner (2020) for a detailed explanation on organelle markers and how to curate a set of reliable markers. Mapping spatial protein dynamics during an infection represents a special challenge as marker localizations may change. Therefore, it is essential to carefully select and validate markers for spatial proteomic HPI studies (Beltran et al., 2017).

Supervised methods, such as support vector machines (SVM) have been used to map proteins to organelles with good accuracy (Trotter et al., 2010). The application of such methods, however, is limited by the availability of the organelle marker training datasets. Specifically, all localizations existing in the experimental output must be represented in the training dataset. If this condition is not met, protein localizations may be predicted incorrectly (Breckels et al., 2013; Christoforou et al., 2014; Gatto et al., 2014a). To address this, Breckels et al. (2013) developed phenoDisco—a semi-supervised machine learning schema to identify putative subcellular phenotype groupings in spatial proteomics experiments (**Figure 8**). The phenoDisco algorithm first uses novelty detection (unsupervised learning) to cluster the training data and expose the representative labels of each cluster/phenotype. Secondly, the algorithm examines and extracts points closest to cluster centroid for use as

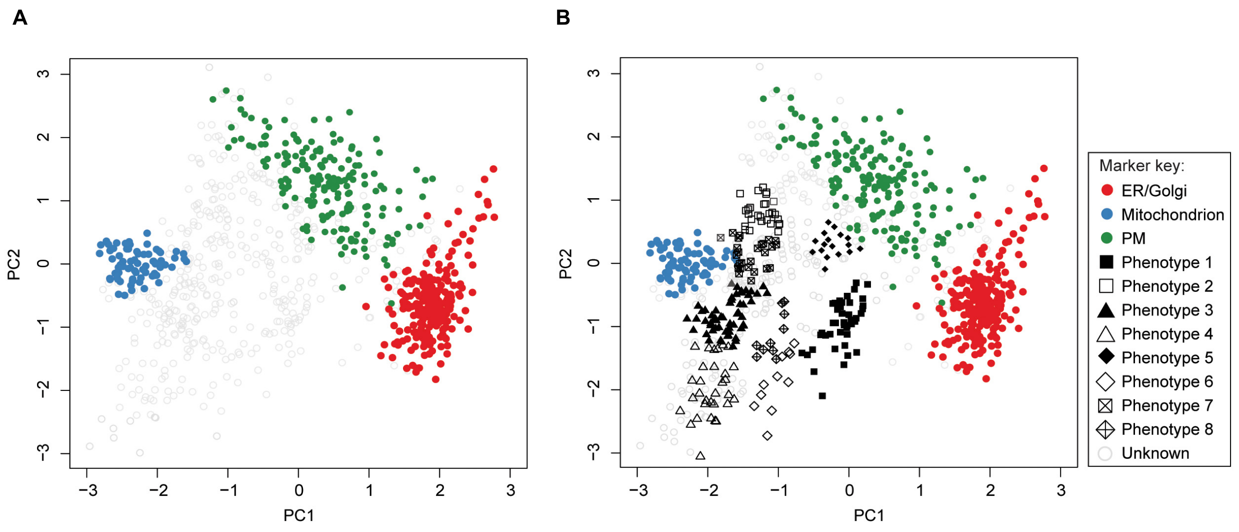


FIGURE 8 | PhenoDisco algorithm helps assign localizations to proteins. First, the algorithm uses a minimal set of markers and unlabeled data as input to detect new phenotypes (i.e., clusters) *via* unsupervised learning. These are then used in supervised learning to predict protein localizations. **(A)** *Drosophila melanogaster* data (Tan et al., 2009) and ER/Golgi, Mitochondrion, and PM markers were used as input (ER, endoplasmic reticulum; PM, plasma membrane). **(B)** The phenotypes we identified by using phenoDisco correspond to ribosomal subunits (phenotypes 1 and 3), proteasome (phenotype 2), nucleus (phenotype 4), peroxisome (phenotype 6), and proteins with uncertain localization (phenotypes 5, 7, 8).

labeled training instances (Christoforou et al., 2014). These labeled training instances need to be carefully validated before the subsequent round of supervised learning that will use them alongside markers to assign protein localizations. The phenoDisco algorithm is available in pRoloc R Bioconductor package (Gatto et al., 2014b). The ability to identify and examine clusters offered by this algorithm is particularly useful in the study of infectious disease, which might trigger protein localization changes.

Dimensionality Reduction Tools for Visualizing Organellar Map

Dimensionality reduction techniques are a convenient tool for visualizing high-dimensional spatial proteomics data (**Figure 9**). There are two types of dimensionality reduction techniques: linear transformation (e.g., PCA, refer to section "Quantitative Temporal Data Visualization, Preprocessing, and Quality Control" for more details on PCA) and non-linear transformation (e.g., t-SNE, t-distributed stochastic neighborhood embedding, or UMAP, uniform manifold approximation and projection). t-SNE finds a pattern in the data by calculating pairwise similarities between points in the high-dimensional space, and projects this onto a low-dimensional space, progressively minimizing the difference between the two sets of similarities while preserving the local structure of the data (Van Der Maaten and Hinton, 2008). Due to the probability distribution used to measure the embedding, t-SNE produces better-resolved clusters in a map, which makes it popular for visualizing subcellular clusters (Jean Beltran et al., 2016; Orre et al., 2019). Unlike PCA, t-SNE does not work on global variance and instances that contribute the most to the variability. Instead, t-SNE groups similar input items and emphasizes the separation of different input data items; thereby, the neighboring points in the high-dimensional space end up close to each other in low-dimensional space (Van Der Maaten and Hinton, 2008). However, the t-SNE algorithm is sensitive to its tunable parameters (i.e., perplexity, learning rate, and maximum iterations). Indeed, even slightly different parameter values can generate different output, making such maps difficult to compare (**Figures 9B–D**). Therefore, tunable parameters must be optimized. Furthermore, higher iterations of t-SNE result in better quality maps at the expense of longer runs; while too few iterations may not resolve clusters (McInnes et al., 2018). The best practice involves optimizing parameters using a grid-search (i.e., testing all combinations of parameters, refer to section "Predicting Protein Localizations in Each Condition"), and choosing the best combination for a given dataset (Orre et al., 2019). t-SNE algorithm is implemented in the tsne package (Donaldson, 2016) in R, as well as the pRoloc Bioconductor package (Gatto et al., 2014b).

Unlike t-SNE, which is a locally focused method, the UMAP preserves both local and global structure. It also boasts shorter run times and applicability to big datasets (Allaoui et al., 2020). Briefly, UMAP uses the *k*-nearest neighbor concept and builds a high-dimensional graph of the input data. It then optimizes the layout of the low-dimensional output graph to make it as similar as possible to the high-dimensional graph of the input data. The constructed high-dimensional graph is weighted, with edge weights showing the possibility that two points are connected. The UMAP algorithm extends the radius around each point and connects each point and its neighbors with intersecting radii to determine connectedness. The choice of the radius is essential, as isolated clusters can result from radii that are too small, and vice versa. However, UMPA deals with this issue

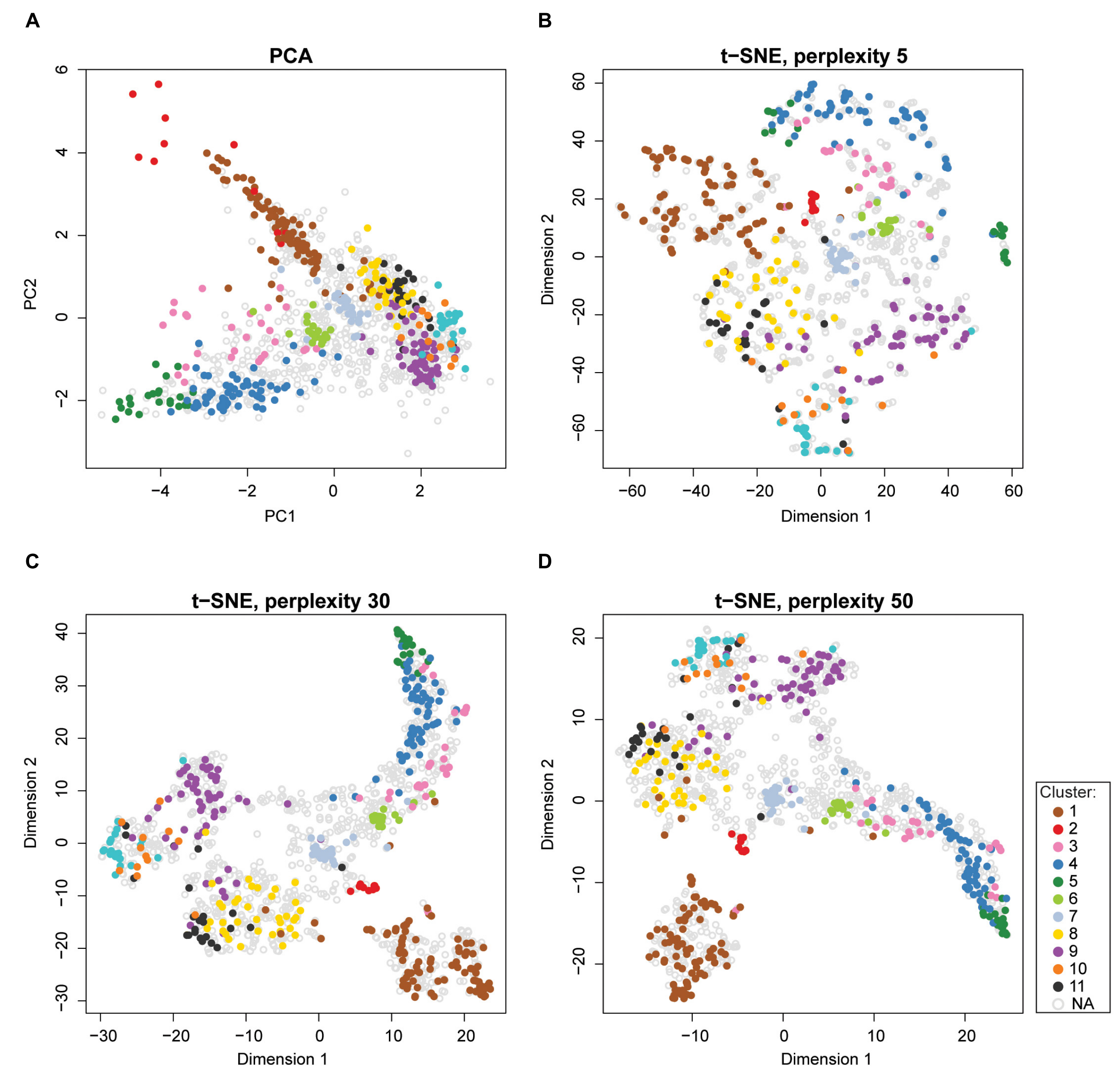


FIGURE 9 | Visualization of high-dimensional spatial proteomics data using PCA and t-SNE. **(A)** Data can be displayed along principal component values (PCs) of PCA in two or more dimensions. **(B–D)** Better resolution of clusters can be achieved by t-SNE. However, the results of the t-SNE algorithm depend on the values of its tunable parameters. For instance, the perplexity parameter typically ranges between 5 and 50 (Van Der Maaten and Hinton, 2008). Perplexity settings of 5, 30, and 50 are shown in panels **(B–D)**. All other parameters were kept constant. Human Embryonic Kidney (HEK293T) sample data from pRoloc package was used for all plots.

by restricting the size of the local neighborhood when learning the manifold structure of data. Once the high-dimensional graph is constructed, it optimizes the graph's low-dimensional embedding through stochastic gradient descent (Sainburg et al., 2020). Like t-SNE, the UMAP algorithm is sensitive to its tunable parameters (i.e., n_neighbors, the number of nearest neighbors; min_dist, minimum distance between two points), which can affect the balance between the local and global structure in the final projection. Similar to t-SNE, the tunable parameters must be optimized by using grid-search. The UMAP algorithm is implemented in the umap package (Konopka, 2020) in R. Since UMAP preserves both, the local and the global data structure, and is faster than t-SNE, it is typically preferred in data dimensionality reduction applications.

Note that the aforementioned techniques are mainly used to visualize the overall structure of the data and should not be used for assigning localizations to proteins. In particular, they can be used to evaluate marker proteins, to assess the resolution, tightness, and similarity of clusters, and to inspect whether the data has a well-defined structure (**Figure 7C**ii).

Data Preprocessing and Quality Control

Proteins identified by spatial proteomics are first annotated based on their localization as markers or non-marker proteins

(**Figure 7A**). Marker proteins will subsequently be used as reference points to find new proteins with the same localization pattern (Gatto et al., 2014a). The next step in the analysis is to evaluate the quality of the dataset and collect descriptive statistics as detailed for temporal data in section "Quantitative Temporal Data Visualization, Preprocessing, and Quality Control". Similarly to temporal proteomic data analysis, missing values must be imputed, and data normalized to generate reliable and comparable results for downstream analyses (**Figure 7B**).

Missing Value Imputation

A number of algorithms used for spatial proteomics data analysis cannot deal with incomplete data. However, the impact of imputation has not been thoroughly addressed in spatial proteomics literature. The raw quantitative data may contain missing ion intensity values for a number of reasons. These include low protein abundance and low instrument sensitivity (Karpievitch et al., 2012). For low abundance proteins, missing values can be imputed by substituting lowest observed intensity or peptide count (Karpievitch et al., 2012). This is computationally easy and fast. However, this does not take into account the patterns in the data and may introduce bias in the data as the number of imputed values increases. Another approach is to exclude proteins with missing values (Dunkley et al., 2006; Du et al., 2008; Hall et al., 2009; Tan et al., 2009). This is a suitable approach if only a small number of such proteins were identified; however, if many proteins are missing data, simply dropping them may increase the bias.

The effect of imputation on the downstream analysis of microarray data has been studied (see Oh et al., 2011 for an overview of different imputation methods). The same approaches are used in spatial proteomics. For instance, the *k*-nearest neighbors (*k*-NN) approach uses feature similarity to assign a value to a random missing point. It finds proteins (i.e., *k*-NN) with expression profiles similar to that of the protein with the missing data. A weighted average based on *k*-NN is then used to impute the missing value. Although this approach is accurate, it is sensitive to outliers (Kim et al., 2005). Another useful approach for imputing missing values is the Multivariate Imputation by Chained Equation (MICE) (Van Buuren and Groothuis-Oudshoorn, 2010). It assumes that the data are missing at random, and calculates multiple imputations with their corresponding estimates of uncertainty. Both, *k*-NN and MICE are suitable for datasets with a several data points missing at random. They are implemented in bnstruct (Franzin et al., 2017) and mice (Van Buuren and Groothuis-Oudshoorn, 2010) R packages, respectively. The *k*-NN approach is also available as impute() function in the DEP Bioconductor package (Zhang et al., 2018).

In cases when many data points are missing not at random, for example when proteins are not quantified in a specific condition (i.e., are below the detection limit), simply excluding such data from analyses will introduce bias (Luo et al., 2009). The missing values in such data can be imputed for example by using quantile regression-based left-censored function ("QRILC"). This approach is available as impute() function in the DEP Bioconductor package (Zhang et al., 2018). Values missing due to random chance (e.g., technical variability) and missing not at random can also be imputed by MSstats Bioconductor package in R (Choi et al., 2014).

However, all aforementioned imputation approaches are likely introduce bias (Karpievitch et al., 2012; Wei et al., 2018; Goeminne, 2019). For example, Gatto et al. (2014a) imputed missing values in a spatial proteomics dataset using *k*-NN, and the results generated from the imputed data indicated the misclassification of protein localizations. The best way to obtain unbiased estimates for missing values is to explicitly model the missing data by using methods, such as maximum-likelihood model (Karpievitch Y. et al., 2009; Kang, 2013), Bayesian (Luo et al., 2009), and Expectation-Maximization (EM) approach (Kang, 2013). These modeling approaches are applicable to data missing at random and not at random. Irrespective of the imputation method, the effect of data imputation must be carefully evaluated each time.

Data Normalization

Another critical aspect of data preprocessing is data normalization. Normalization methods must make samples statistically comparable, while correcting for intragroup differences (e.g., batch effects) and preserving between-group differences (e.g., differences between organelles) (Välikangas et al., 2018; Chen et al., 2020). The most common normalization approach for reducing unwanted variation in spatial proteomics involves dividing each ion intensity (or spectral counts) by the sum or maximum of ion intensities (or spectra counts) in each row (Gatto et al., 2014a; Breckels et al., 2016) or column (Hu et al., 2019). Such normalization can be done quickly and easily (Gatto et al., 2014a; Breckels et al., 2016; Hu et al., 2019). Callister et al. compared linear regression, central tendency, locally weighted regression, and quantile normalization methods (Callister et al., 2006). They all decreased systematic bias in the data, but linear regression normalization performed best. Linear regression assumes that bias and peptide abundance are linearly dependent. This means that as the measured peptide ion abundance increases, the systematic bias also increases (Callister et al., 2006). Rlr, RlrMA, and RlrMA (Välikangas et al., 2018) are linear regression variants available in MASS (Venables and Ripley, 2002) R package.

Similarly, Välikangas et al. (2018) systematically evaluated 11 popular normalization methods using four proteomic datasets. Results indicated that variance stabilization normalization (Vsn) reduced the intragroup variation the most and performed well in differential protein expression analysis with all tested datasets. Likewise, local regression normalization and linear regression normalization performed well. Moreover, excellent performance of Vsn was demonstrated by Gatto et al. (2014a), who used two biological spatial proteomic replicates with substantial technical variability to simulate multiple conditions. Vsn significantly reduced variation and improved overlap between replicates. Vsn aims to remove the dependencies between sample variances and mean intensities, and scale the samples to the same intensity range by using maximum likelihood estimation and parametric transformation (Huber et al., 2002). This statistical method is available as justvsn() function in the Vsn Bioconductor package (Huber et al., 2002) and normalize_vsn() function in the DEP Bioconductor package (Zhang et al., 2018). As in temporal proteomics experiments, the efficiency of normalization must be evaluated, e.g., using MA plots (see section "Quantitative Temporal Data Visualization, Preprocessing, and Quality Control").

Several other normalization methods are available. Among these are LOWESS (locally weighted scatterplot smoothing) regression, and EigenMS (Quackenbush, 2002; Bolstad et al., 2003; Callister et al., 2006; Leek and Storey, 2007; Karpievitch Y.V. et al., 2009; Zhang et al., 2015). EigenMS uses singular value decomposition (SVD) on model residuals to capture trends that lead to the formation of bias. This algorithm is capable of handling missing values and is available as a stand-alone function implemented in R at http://sourceforge.net/projects/eigenms/ (Karpievitch Y.V. et al., 2009; Karpievitch et al., 2014).

After data preprocessing, the best practice is to use unsupervised clustering to assess the overall structure of the data (see section "Clustering Analyses"; **Figure 7C**i). According to the De Duve's principle (De Duve and Beaufay, 1981), proteins belonging to the same organelle will co-fractionate, thereby leading to the formation of distinct groups upon clustering. The whole dataset can also be visualized using dimensionality reduction techniques, such as PCA or t-distributed stochastic neighbor embedding (tSNE) without the addition of organelle markers to the map (**Figure 7C**ii; *left panel*). Indeed, overlaying markers at this stage can confer a false sense of structure to the data, precluding the visualization of the data's overall structure (Gatto et al., 2014a). The lack of structure in the data is indicative of the clustering algorithm's inability to separate clusters at later analysis stages. Next, organelle markers are overlaid (on plots or clusters) to assess the resolution in the data (**Figure 7C**ii; *right panel*). Here, a clear separation of markers by organelle membership is expected. Its absence will undermine all subsequent analyses and interpretation. Such a situation may be remedied by adjusting imputation and normalization methods. Furthermore, these overlaid maps can also be inspected for outliers, for instance an unexpected marker position. These may indicate unreliable protein quantitation, identification, or annotation as a marker (Gatto et al., 2014a).

Predicting Protein Localizations in Each Condition

Supervised machine learning is the method of choice for predicting the subcellular localizations of proteins (**Figures 7D, 10**). During training, a supervised learning algorithm will learn to associate independent variables (i.e., protein abundances across fractions) and protein labels (i.e., marker assignments). After training, the algorithm predicts the labels for proteins with unknown localizations (Swan et al., 2013). Supervised learning algorithms can be subdivided into two groups based on the characteristics of the label: classification (labels are discrete categories) and regression (labels are continuous numeric values). In spatial proteomics, multiclass classification algorithms are typically used: labels are discrete and cover many (i.e., three or more) possible localizations. The first step in multiclass classification is selecting the algorithm (**Figure 10A**). Among common multiclass classification algorithms are naïve Bayes, *k*-nearest neighbor, SVM, random forest, and artificial neural networks. See Kotsiantis et al. (2007) and Crisci et al. (2012) for a comprehensive overview of supervised machine learning algorithms and their biases. However, it is widely accepted that how the algorithm is applied and the quality of the training data have a greater effect on the final result than the choice of the algorithm (Gatto et al., 2014a).

The second step in multiclass classification is defining the range of possible hyperparameter values (**Figure 10B**i). Hyperparameters are adjustable parameters that have to be set before training to obtain a model with optimal performance. A model is defined by the combination of the selected classification algorithm and a specific set of hyperparameters. This model is then used to predict class labels. Examples of hyperparameters are: the regulation and constant parameter C in SVM algorithm, the number of nearest neighbors used (*k*) in *k*-nearest neighbors, and the number of decision trees in random forest. See Luo (2016) for a comprehensive overview of hyperparameters for different machine learning algorithms. Using default hyperparameter settings cannot ensure optimal learning performance. Moreover, wrongly selected parameters can adversely impact the resulting model's performance (Gatto et al., 2014a; Luo, 2016; Probst et al., 2019; Schratz et al., 2019). Therefore, various combinations of hyperparameters must be tested to choose the best set.

Defining which hyperparameter combinations will be evaluated is the third step in multiclass classification (**Figure 10B**ii). Manual selection can be inefficient. Random or grid search allow for automated and efficient selection of hyperparameter combinations. In grid search (also known as exhaustive search), every possible combination of parameters within a specified grid is selected for subsequent evaluation (Rojas-Domínguez et al., 2017). In a random search, a fixed number of random combinations of hyperparameters is selected (Bergstra and Bengio, 2012).

Next, i.e., fourth step is the testing of all selected hyperparameter combinations by means of cross-validation. Cross-validation (out-of-sample testing) is a model evaluation method that estimates how accurately the model will predict the labels of unseen (i.e., out-of-sample) data (Payam et al., 2009; **Figures 10B**iii,iv). During cross-validation, labeled data is split into a training set (to train the classifier; **Figure 10B**iii) and a testing set (**Figure 10B**iv), which is used to evaluate the model's performance with each pre-selected combination of hyperparameters. Subsequently, different evaluation metrics (see below) are used to evaluate the model's performance (**Figure 10Bv**), and the hyperparameter combination resulting in the best model performance is selected (**Figure 10B**vi).

Among the most common cross-validation techniques are the: (i) holdout, (ii) *k*-fold, (iii) leave-one-out, and (iv) leave-p-out methods. See Payam et al. (2009) for an overview of cross-validation techniques and their biases. For example, *k*-fold cross-validation has been used frequently in proteomic studies (Granholm et al., 2012; Swan et al., 2013; Hu et al., 2019).

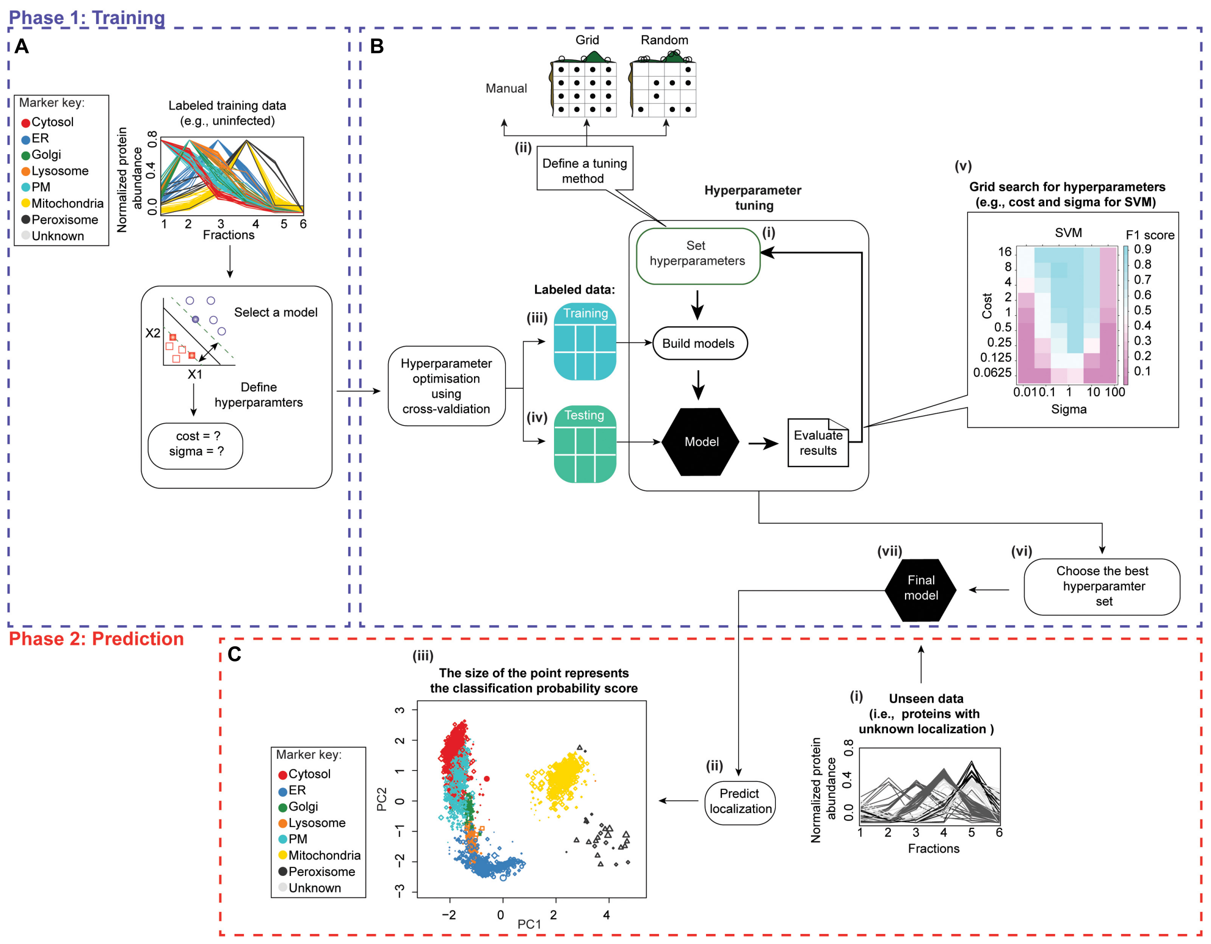

Phase 1: Training
A
Marker key:
Cytosol
ER
Golgi
Lysosome
PM
Mitochondria
Peroxisome
Unknown
Labeled training data (e.g., uninfected)
Normalized protein abundance
Fractions
Select a model
Define hyperparamters
cost = ?
sigma = ?
B
Manual
Grid
Random
(ii) Define a tuning method
Hyperparameter tuning
(i) Set hyperparameters
Build models
Model
Evaluate results
Labeled data:
(iii) Training
(iv) Testing
Hyperparameter optimisation using cross-validation
(v) Grid search for hyperparameters (e.g., cost and sigma for SVM)
SVM
F1 score
Cost
Sigma
(vi) Choose the best hyperparamter set
(vii) Final model
Phase 2: Prediction
C
(iii) The size of the point represents the classification probability score
PC1
PC2
(ii) Predict localization
(i) Unseen data (i.e., proteins with unknown localization)

FIGURE 10 | Schematic overview of supervised learning for subcellular localization mapping. **(A)** The first step in multiclass classification is to select an algorithm with its corresponding hyperparameters. Here, we show SVM. **(B)** Next, the hyperparameters must be tuned. This includes: (i) defining the range of possible hyperparameter values, (ii) selecting a method for sampling hyperparameter values (i.e., manual, grid, or randomized), and choosing a cross-validation technique (e.g., *k*-fold) for evaluating the model's performance with labeled data correspondingly divided into training (iii) and testing (iv) datasets. Moreover, (v) a metric to judge the model's performance with each set of hyperparameters must be defined. Based on the metric, the best set of hyperparameter values is chosen (vi). For instance, (v) the two SVM hyperparameters (i.e., cost and sigma) were optimized over 50 rounds of 5-fold cross-validation through a grid search, and then the best pair of hyperparameters was chosen based on the evaluation of *F1* score. Once the best combination of hyperparameters has been selected (vi), it can be used to build the final model (vii). **(C)** Proteins with unknown localizations are then presented to the model (i), and the model predicts their labels (i.e., localizations; ii). (iii) Classification scores for unlabeled instances that correspond to their most likely subcellular compartment are reflected by point sizes. Each point here represents a protein. Uninfected data from **Figure 7** was used throughout this figure. The same workflow can also be applied to the infected data.

In *k*-fold cross-validation, the data is randomly split into *k* number of equally sized folds (i.e., groups). Then *k* iterations of training and testing are performed with a single combination of hyperparameters, such that at each iteration a different *k* fold is held-out as the test data set to evaluate the model's performance, and the remaining (*k* - 1) folds are put together to form a training set. After each iteration, testing accuracy metric (see below) is computed using the testing dataset (i.e., the left-out fold). The overall out-of-sample accuracy is the average of all *k* iterations performed with a given set of hyperparameters (Payam et al., 2009).

During cross-validation, a confusion matrix is typically used to gain insight into the model's performance and errors (Kautz et al., 2017). A confusion matrix is an N^2 matrix (N is the number of classes) that compares the number of actual assignments to N classes with the number predicted by the model (**Table 2**). Based on whether the classes were correctly predicted by the model, observations can be categorized as true positive (TP, correctly identified), false positive (FP, incorrectly identified), true negative (TN, correctly rejected), and false negative (FN, incorrectly rejected). Then, based on the confusion matrix-derived categorization, performance metrics (**Table 3**) can be calculated and used to assess how well a given model performs on a testing data set (**Figure 10B**v). The fifth step in multiclass classification is to choose the best combination of hyperparameters that maximize the model's performance (**Figure 10B**vi) and use that combination to build the final model from the training data (**Figure 10B**vii). The sixth step is predicting class labels for unseen testing data (**Figures 10C**i,ii). Here, proteins with unknown localizations

TABLE 2 | A confusion matrix with $N = 2$.

	Predicted: No	Predicted: Yes
Actual: No	TN	FP
Actual: Yes	FN	TP

TABLE 3 | Performance metrics.

Performance metrics	Definition	Formula
Accuracy	The ratio of the number of correctly predicted observations to total observations	$\frac{TP+TN}{TP+FP+FN+TN}$
Sensitivity or recall	The proportion of positives that are correctly identified as positive by the model	$\frac{TP}{(FN+TP)}$
Specificity	The proportion of negatives that are correctly identified as negative by the model	$\frac{TN}{(TN+FP)}$
Precision	The proportion of true positives out of all predicted positives	$\frac{TP}{(TP+FP)}$
F1 Score	The harmonic mean of precision and recall	$2\frac{precision \times recall}{precision+recall}$

are assigned classification scores that reflect their most likely localization (**Figure 10C**iii). The seventh step is evaluating model performance with the testing dataset. This can be done for example, by performing functional enrichment analyses (see section "Evaluation Measures for Temporal Clustering").

All classification algorithms mentioned in this section have been implemented in the pRoloc package (Gatto et al., 2014b). Moreover, the caret R package (Kuhn et al., 2020) has 233 built-in classification algorithms and several functions for cross-validation and hyperparameter tuning using grid and random search. To demonstrate the application of supervised learning, we took the data from infected (i.e., 48 h post infection) and uninfected samples from Jean Beltran et al. (2016). We then randomly selected 3,000 host proteins and used the aforementioned framework to predict subcellular localizations (**Figures 7D**, **10**). First, the labeled training data were constructed by mapping the markers available in the pRoloc package to the selected 3,000 proteins (**Figure 10A**). We chose to use the SVM implemented in the pRoloc Bioconductor package (Gatto et al., 2014b) to assign protein localizations (**Figure 10A**). Two hyperparameter values of the SVM model, cost and sigma, were optimized over 50 rounds of 5-fold cross-validation through a grid search (**Figures 10B**i,ii,iii,iv). *F1* score metric was then used to evaluate the model's performance and select the best parameters that result in the highest out-of-sample testing accuracy (**Figure 10B**v). The optimized model was then utilized to predict the label for each protein with unknown localization (**Figures 10B**vii,**C**).

Detecting Protein Translocation Events

One of the main applications of spatial proteomics in the context of infectious disease is comparing organellar proteome maps (e.g., infected vs. uninfected) to identify proteins with altered subcellular localization (Jean Beltran et al., 2016). A simple contingency table tracking the total number of assignments to each of the compartments in each map can be used to assess the global pattern of change (**Figure 7E**i). However, this method is prone to error. For instance, situations when identities of proteins in each compartment change but their numbers remain would be missed (Borner, 2020).

A more sensitive assessment of localization changes evaluates each protein individually (**Figure 7E**ii; Gatto et al., 2014a; Itzhak et al., 2016; Borner, 2020). Indeed, if a protein's localization changes during an infection, it's profile, or abundance/enrichment across compartments would change as well. Conversely, if a protein's localization were not affected, its profile would remain the same in infected and uninfected samples. Moreover, the profile's change (or lack thereof) should be reproducible across replicates. This is the basis for detecting protein localization changes by means of MR plots (**Figure 7E**ii). Here, M stands for the magnitude of translocation score, which compares the profiles between two conditions, and R stands for the reproducibility of translocation score. See Itzhak et al. (2016) for details and formulas. Note that replicates are essential for the statistical power of this test, and cut-offs are determined based on the comparison of two sets of control data (e.g., uninfected) samples. In an MR plot, significantly translocating proteins are found in an upper right-hand quadrant (**Figure 7E**ii; red stars).

However, when infection results in drastic morphological alterations, and significant changes in profiles of most proteins are observed, it might be better to identify translocation events based on altered predicted compartment association (Jean Beltran et al., 2016).

DISCUSSION

Advances in sample preparation methods, mass spectrometry, as well as computational facilities and approaches allow producing and analyzing a plethora of proteomic HPI data to reveal changes occurring across space and time in response to an infection. Analyses of spatial and temporal proteomic HPI data can be especially challenging due to the data's high complexity. In this review, we present the workflow pipelines for the analysis of such datasets. Moreover, we discuss the pros, cons, best practices, and challenges associated with each step. Novices in the field can use this review as a workflow tutorial, while experienced users may find it helpful for updating their data analysis pipelines.

Numerous pathogenesis strategies exist. Since only a small portion of spatial or temporal infection-related proteome changes has been mapped, and only for a small subset of pathogens (Sánchez-Quiles et al., 2011; Gudleski-O'Regan et al., 2012; Weekes et al., 2014; Matheson et al., 2015; Greenwood et al., 2016; Jean Beltran et al., 2016; Karniely et al., 2016; Soday et al., 2019; Depierreux et al., 2020; Tiku et al., 2020), we expect to see many more publications in the area of spatial and temporal HPI proteomics. Moreover, since disease progression is a dynamic process, simultaneous analyses of multiple subcellular compartments at critical time points throughout the course of an infection (i.e., a combined spatiotemporal study) can help understand disease processes on the molecular level better than each of the approaches alone. An example of such a study is the analysis of organelle alterations occurring during the course of human cytomegalovirus infection (Jean Beltran et al., 2016). Here, essentially, spatial proteomics experiments were repeated across a series of time points post-infection. The resulting maps defined protein trafficking in space and time, and elucidated essential disease processes and protein roles.

The next level of complexity in investigating HPIs is the mapping of protein complexes and protein-protein interactions on a global scale with resolution in space and time. This is a critical component of understanding how the observed changes are orchestrated. Future experimental and computational efforts will be moving in this direction. Moreover, bioinformatics pipelines will be developing to better integrate spatial and temporal maps of proteome and protein complex changes during disease progression. Furthermore, since pathogens alter multiple interconnected systems (e.g., RNA, proteins, lipids, and metabolites), integrating proteomic with other omics datasets is gaining traction (Nesvizhskii, 2014; Miranda-CasoLuengo et al., 2016; Cambiaghi et al., 2018; Zhou et al., 2018; Xu et al., 2020). Adopting such approaches can help answer many questions about disease processes as well as the normal functions they disrupt. Such insights cannot be derived by a single method. Depending on the types of omics data that are being integrated, different challenges may be faced and different new tools may be required. For example, in comparing transcriptomics and proteomics datasets, one main issue is the mapping between proteins and genes. Databases, such as BioMart (Smedley et al., 2009) and Uniprot (Magrane and Consortium, 2011) map proteins to transcriptomic/genomic identifiers, but discrepancies remain as a result of error propagation from legacy issues during automated data integration (Kumar and Mann, 2009). After data are collected for various species, the next important step in the understanding of how pathogens hijack host systems. This will involve comparing similarities and differences across diseases, pathogens, and hosts (for example, see Shah et al., 2015; Gordon et al., 2020). These efforts will help glean comparative insights about disease progression patterns, and guide the design of pathogen-specific as well as pan-pathogen therapies.

AUTHOR CONTRIBUTIONS

MR wrote all the R scripts. MR and AG wrote the text and made the figures. AG provided the feedback to MR. MB supervised and supported MR. All authors read and approved the manuscript.

FUNDING

AG was a recipient of a CIHR Postdoctoral Fellowship. This work was supported by grants from the Natural Sciences and Engineering Research Council to MB (DG-20234).

REFERENCES

Alto, N. M., and Orth, K. (2012). Subversion of cell signaling by pathogens. *Cold Spring Harb. Perspect. Biol.* 4:a006114. doi: 10.1101/cshperspect.a006114

Allaoui, M., Kherfi, M. L., and Cheriet, A. (2020). "Considerably improving clustering algorithms using umap dimensionality reduction technique: a comparative study," in *Image and Signal Processing. ICISP 2020. Lecture Notes in Computer Science*, eds Moataz A El, D. Mammass, A. Mansouri, and F. Nouboud (Cham: Springer), 317–325.

Anders, S., and Huber, W. (2010). Differential expression analysis for sequence count data. *Genome Biol.* 11:R106. doi: 10.1186/gb-2010-11-10-r106

Auweter, S. D., Bhavsar, A. P., de Hoog, C. L., Li, Y., Chan, Y. A., van der Heijden, J., et al. (2011). Quantitative mass spectrometry catalogues *Salmonella* pathogenicity island-2 effectors and identifies their cognate host binding partners. *J. Biol. Chem.* 286, 24023–24035. doi: 10.1074/jbc.M111.224600

Beltran, P. M. J., Cook, K. C., and Cristea, I. M. (2017). Exploring and exploiting proteome organization during viral infection. *J. Virol.* 91, e00268–017. doi: 10.1128/JVI.00268-17

Benjamini, Y., and Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. R. Stat. Soc. Ser. B* 57, 289–300. doi: 10.2307/2346101

Bergstra, J., and Bengio, Y. (2012). Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* 13, 281–305.

Bezdek, J. C., Ehrlich, R., and Full, W. (1984). FCM: the fuzzy c-means clustering algorithm. *Comput. Geosci.* 10, 191–203. doi: 10.1016/0098-3004(84)90020-7

Bezdek, J. C., Keller, J., Krisnapuram, R., and Pal, N. (1999). *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*. New York NY: Springer Science & Business Media.

Bhargavi, M. S., and Gowda, S. D. (2015). A novel validity index with dynamic cut-off for determining true clusters. *Pattern Recognit.* 48, 3673–3687. doi: 10.1016/j.patcog.2015.04.023

Blagoev, B., Ong, S. E., Kratchmarova, I., and Mann, M. (2004). Temporal analysis of phosphotyrosine-dependent signaling networks by quantitative proteomics. *Nat. Biotechnol.* 22, 1139–1145. doi: 10.1038/nbt1005

Bolstad, B. M., Irizarry, R. A., Åstrand, M., and Speed, T. P. (2003). A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics* 19, 185–193. doi: 10.1093/bioinformatics/19.2.185

Borchers, H. W. (2019). *pracma: Practical Numerical Math Functions. R package Version 2.2.9.* Available online at: https://CRAN.R-project.org/package=pracma (accessed December 16, 2019).

Borner, G. H. H. (2020). Organellar maps through proteomic profiling–a conceptual guide. *Mol. Cell. Proteomics* 19, 1076–1087. doi: 10.1074/mcp.R120.001971

Boyle, E. I., Weng, S., Gollub, J., Jin, H., Botstein, D., Cherry, J. M., et al. (2004). GO::termfinder-open source software for accessing Gene Ontology information and finding significantly enriched gene ontology terms associated with a list of genes. *Bioinformatics* 20, 3710–3715. doi: 10.1093/bioinformatics/bth456

Branson, O. E., and Freitas, M. A. (2016). A multi-model statistical approach for proteomic spectral count quantitation. *J. Proteomics* 144, 23–32. doi: 10.1016/j.jprot.2016.05.032

Breckels, L. M., Gatto, L., Christoforou, A., Groen, A. J., Lilley, K. S., and Trotter, M. W. B. (2013). The effect of organelle discovery upon sub-cellular protein localisation. *J. Proteomics* 88, 129–140. doi: 10.1016/j.jprot.2013.02.019

Breckels, L. M., Holden, S. B., Wojnar, D., Mulvey, C. M., Christoforou, A., Groen, A., et al. (2016). Learning from heterogeneous data sources: an application in spatial proteomics. *PLoS Comput. Biol.* 12:1004920. doi: 10.1371/journal.pcbi.1004920

Breen, M. S., Ozcan, S., Ramsey, J. M., Wang, Z., Ma'ayan, A., Rustogi, N., et al. (2018). Temporal proteomic profiling of postnatal human cortical development. *Transl. Psychiatry* 8:267. doi: 10.1038/s41398-018-0306-4

Breitwieser, F. P., Müller, A., Dayon, L., Köcher, T., Hainard, A., Pichler, P., et al. (2011). General statistical modeling of data from protein relative expression isobaric tags. *J. Proteome Res.* 10, 2758–2766. doi: 10.1021/pr1012784

Brock, G., Pihur, V., Datta, S., and Datta, S. (2011). clValid, an R package for cluster validation. *J. Stat. Softw.* 25, 1–22. doi: 10.18637/jss.v025.i04

Bruno, G., and Fiori, A. (2013). "Spread of evaluation measures for microarray clustering," in *Biological Knowledge Discovery Handbook*, eds M. Elloumi and A. Y. Zomaya (Hoboken NJ: John Wiley & Sons, Inc), 569–590. doi: 10.1002/9781118617151.ch24

Brusniak, M. Y., Bodenmiller, B., Campbell, D., Cooke, K., Eddes, J., Garbutt, A., et al. (2008). Corra: computational framework and tools for LC-MS discovery and targeted mass spectrometry-based proteomics. *BMC Bioinformatics* 9:542. doi: 10.1186/1471-2105-9-542

Caller, L. G., Davies, C. T. R., Antrobus, R., Lehner, P. J., Weekes, M. P., and Crump, C. M. (2019). Temporal proteomic analysis of BK polyomavirus infection reveals virus-induced G 2 arrest and highly effective evasion of innate immune sensing. *J. Virol.* 93, e00595–19. doi: 10.1128/jvi.00595-19

Callister, S. J., Barry, R. C., Adkins, J. N., Johnson, E. T., Qian, W., Webb-Robertson, B.-J. M., et al. (2006). Normalization approaches for removing systematic biases associated with mass spectrometry and label-free proteomics. *J. Proteome Res.* 5, 277–286. doi: 10.1021/pr050300l

Cambiaghi, A., Díaz, R., Martinez, J. B., Odena, A., Brunelli, L., Caironi, P., et al. (2018). An innovative approach for the integration of proteomics and metabolomics data in severe septic shock patients stratified for mortality. *Sci. Rep.* 8, 1–12. doi: 10.1038/s41598-018-25035-1

Cebeci, Z. (2019). Comparison of internal validity indices for fuzzy clustering. *J. Agric. Informatics* 10, 1–14. doi: 10.17700/jai.2019.10.2.537

Chang, P.-L., and Teng, W.-G. (2007). "Exploiting the self-organizing map for medical image segmentation," in *Proceedings of the 20th IEEE International Symposium on Computer-Based Medical Systems (CBMS'07)*, (Maribor: IEEE), 281–288.

Chen, C., Hou, J., Tanner, J. J., and Cheng, J. (2020). Bioinformatics methods for mass spectrometry-based proteomics data analysis. *Int. J. Mol. Sci.* 21:2873. doi: 10.3390/ijms21082873

Chipman, H., and Tibshirani, R. (2006). Hybrid hierarchical clustering with applications to microarray data. *Biostatistics* 7, 286–301. doi: 10.1093/biostatistics/kxj007

Chiu, C. C., Chan, S. Y., Wang, C. C., and Wu, W. S. (2013). Missing value imputation for microarray data: a comprehensive comparison study and a web tool. *BMC Syst. Biol.* 7:S12. doi: 10.1186/1752-0509-7-S6-S12

Choi, H., Fermin, D., and Nesvizhskii, A. I. (2008). Significance analysis of spectral count data in label-free shotgun proteomics. *Mol. Cell. Proteomics* 7, 2373–2385. doi: 10.1074/mcp.M800203-MCP200

Choi, M., Chang, C.-Y., Clough, T., Broudy, D., Killeen, T., Maclean, B., et al. (2014). Systems biology MSstats: an R package for statistical analysis of quantitative mass spectrometry-based proteomic experiments. *Bioinformatics* 30, 2524–2526. doi: 10.1093/bioinformatics/btu305

Christoforou, A., Mulvey, C., Breckels, L. M., Gatto, L., and Lilley, K. S. (2014). "Spatial proteomics: practical considerations for data acquisition and analysis in protein subcellular localisation studies," in *Quantitative Proteomics*, eds C. E. Eyers and S. J. Gaskell (London: The Royal Society of Chemistry), 185–210.

Clements, D. R., Murphy, J. P., Sterea, A., Kennedy, B. E., Kim, Y., Helson, E., et al. (2017). Quantitative temporal in vivo proteomics deciphers the transition of virus-driven myeloid cells into M2 macrophages. *J. Proteome Res.* 16, 3391–3406. doi: 10.1021/acs.jproteome.7b00425

Cox, J., and Mann, M. (2008). MaxQuant enables high peptide identification rates, individualized p.p.b.-range mass accuracies and proteome-wide protein quantification. *Nat. Biotechnol.* 26, 1367–1372. doi: 10.1038/nbt.1511

Croft, D., O'Kelly, G., Wu, G., Haw, R., Gillespie, M., Matthews, L., et al. (2011). Reactome: A database of reactions, pathways and biological processes. *Nucleic Acids Res.* 39, D691–D697. doi: 10.1093/nar/gkq1018

Crisci, C., Ghattas, B., and Perera, G. (2012). A review of supervised machine learning algorithms and their applications to ecological data. *Ecol. Modell.* 240, 113–122. doi: 10.1016/j.ecolmodel.2012.03.001

Dalman, M. R., Deeter, A., Nimishakavi, G., and Duan, Z.-H. (2012). Fold change and p-value cutoffs significantly alter microarray interpretations. *BMC bioinformatics* 13:S11. doi: 10.1186/1471-2105-13-S2-S11

De Duve, C., and Beaufay, H. (1981). A short history of tissue fractionation. *J. Cell Biol.* 91, 293–299. doi: 10.1083/jcb.91.3.293s

Dembéle, D., and Kastner, P. (2003). Fuzzy C-means method for clustering microarray data. *Bioinformatics* 19, 973–980. doi: 10.1093/bioinformatics/btg119

Depierreux, D. M., Altenburg, A. F., Soday, L., Fletcher-Etherington, A., Ferguson, B. J., Weekes, M. P., et al. (2020). Temporal analysis of the plasma membrane proteome after vaccinia virus infection sheds light on virus strategies to evade the immune response. *J. Immunol.* 204:249.2.

D'haeseleer, P. (2005). How does gene expression clustering work? *Nat. Biotechnol.* 23, 1499–1501. doi: 10.1038/nbt1205-1499

Diamond, D. L., Syder, A. J., Jacobs, J. M., Sorensen, C. M., Walters, K. A., Proll, S. C., et al. (2010). Temporal proteome and lipidome profiles reveal hepatitis C virus-associated reprogramming of hepatocellular metabolism and bioenergetics. *PLoS Pathog.* 6:e1000719. doi: 10.1371/journal.ppat.1000719

Dillies, M. A., Rau, A., Aubert, J., Hennequet-Antier, C., Jeanmougin, M., Servant, N., et al. (2013). A comprehensive evaluation of normalization methods for Illumina high-throughput RNA sequencing data analysis. *Brief. Bioinform.* 14, 671–683. doi: 10.1093/bib/bbs046

Do, J. H., and Choi, D. K. (2008). Clustering approaches to identifying gene expression patterns from DNA microarray data. *Mol. Cells* 25, 279–288.

Dogan, Y., Birant, D., and Kut, A. (2013). "SOM++: integration of self-organizing map and k-means++ algorithms," in *Machine Learning and Data Mining in Pattern Recognition*, ed. P. Perner (Berlin: Springer), 246–259. doi: 10.1007/978-3-642-39712-7

Donaldson, J. (2016). *tsne: T-Distributed Stochastic Neighbor Embedding for R (t-SNE). R package Version 0.1-3*. Available online at: https://CRAN.R-project.org/package=tsne (accessed July 16, 2016).

Dopazo, J., and Carazo, J. M. (1997). Phylogenetic reconstruction using an unsupervised growing neural network that adopts the topology of a phylogenetic tree. *J. Mol. Evol.* 44, 226–233. doi: 10.1007/PL00006139

Dreger, M. (2003). Subcellular proteomics. *Mass Spectrom. Rev.* 22, 27–56. doi: 10.1002/mas.10047

Du, X., Callister, S. J., Manes, N. P., Adkins, J. N., Alexandridis, R. A., Zeng, X., et al. (2008). A computational strategy to analyze label-free temporal bottom-up proteomics data. *J. Proteome Res.* 7, 2595–2604. doi: 10.1021/pr0704837

Dunkley, T. P. J., Hester, S., Shadforth, I. P., Runions, J., Weimar, T., Hanton, S. L., et al. (2006). Mapping the *Arabidopsis* organelle proteome. *Proc. Natl. Acad. Sci.U.S.A.* 103, 6518–6523. doi: 10.1073/pnas.0506958103

Eisenreich, W., Rudel, T., Heesemann, J., and Goebel, W. (2019). How viral and intracellular bacterial pathogens reprogram the metabolism of host cells to allow their intracellular replication. *Front. Cell. Infect. Microbiol.* 9:42.

Foster, L. J., de Hoog, C. L., Zhang, Y., Zhang, Y., Xie, X., Mootha, V. K., et al. (2006). A mammalian organelle map by protein correlation profiling. *Cell* 125, 187–199. doi: 10.1016/j.cell.2006.03.022

Franzin, A., Sambo, F., and Di Camillo, B. (2017). bnstruct: an R package for Bayesian Network structure learning in the presence of missing data. *Bioinformatics* 33, 1250–1252. doi: 10.1093/bioinformatics/btw807

Friedman, N., Linial, M., Nachman, I., and Pe'er, D. (2000). Using Bayesian networks to analyze expression data. *J. Computat. Biol.* 7, 601–620. doi: 10.1089/106652700750050961

Fritzke, B. (1994). Growing cell structures—a self-organizing network for unsupervised and supervised learning. *Neural Networks* 7, 1441–1460. doi: 10.1016/0893-6080(94)90091-4

Gagarinova, A., Phanse, S., Cygler, M., and Babu, M. (2017). Insights from protein-protein interaction studies on bacterial pathogenesis. *Expert Rev. Proteomics* 14, 779–797. doi: 10.1080/14789450.2017.1365603

Gasch, A. P., and Eisen, M. B. (2002). Exploring the conditional coregulation of yeast gene expression through fuzzy k-means clustering. *Genome Biol.* 3, research0059.1. doi: 10.1186/gb-2002-3-11-research0059

Gatto, L., and Lilley, K. S. (2012). MSnbase-an R/Bioconductor package for isobaric tagged mass spectrometry data visualization, processing and quantitation. *Bioinformatics* 28, 288–289. doi: 10.1093/bioinformatics/btr645

Gatto, L., Breckels, L. M., Burger, T., Nightingale, D. J. H., Groen, A. J., Campbell, C., et al. (2014a). A foundation for reliable spatial proteomics data analysis. *Mol. Cell. Proteomics* 13, 1937–1952. doi: 10.1074/mcp.M113.036350

Gatto, L., Breckels, L. M., Wieczorek, S., Burger, T., and Lilley, K. S. (2014b). Mass-spectrometry-based spatial proteomics data analysis using pRoloc and pRolocdata. *Bioinformatics* 30, 1322–1324. doi: 10.1093/bioinformatics/btu013

Gatto, L., Vizcaíno, J. A., Hermjakob, H., Huber, W., and Lilley, K. S. (2010). Organelle proteomics experimental designs and analysis. *Proteomics* 10, 3957–3969. doi: 10.1002/pmic.201000244

Gene Ontology Consortium. (2004). The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Res.* 32, D258–D261. doi: 10.1093/nar/gkh036

Gibbons, F. D., and Roth, F. P. (2002). Judging the quality of gene expression-based clustering methods using gene annotation. *Genome Res.* 12, 1574–1581. doi: 10.1101/gr.397002

Gilbert, A. S., Wheeler, R. T., and May, R. C. (2015). Fungal pathogens: survival and replication within macrophages. *Cold Spring Harb. Perspect. Med.* 5:a019661. doi: 10.1101/cshperspect.a019661

Giraudel, J. L., and Lek, S. (2001). A comparison of self-organizing map algorithm and some conventional statistical methods for ecological community ordination. *Ecol. Modell.* 146, 329–339. doi: 10.1016/S0304-3800(01)00324-6

Gordon, D. E., Hiatt, J., Bouhaddou, M., Rezelj, V. V., Ulferts, S., Braberg, H., et al. (2020). Comparative host-coronavirus protein interaction networks reveal pan-viral disease mechanisms. *Science* 370:eabe9403. doi: 10.1126/science.abe9403

Goeminne, L. (2019). *Statistical Methods for Differential Proteomics at Peptide and Protein Level*. Ghent: Ghent University.

Granholm, V., Noble, W. S., and Käll, L. (2012). A cross-validation scheme for machine learning algorithms in shotgun proteomics. *BMC Bioinformatics* 13:S3. doi: 10.1186/1471-2105-13-S16-S3

Greenwood, E. J., Matheson, N. J., Wals, K., van den Boomen, D. J., Antrobus, R., Williamson, J. C., et al. (2016). Temporal proteomic analysis of HIV infection reveals remodelling of the host phosphoproteome by lentiviral Vif variants. *Elife* 5:e18296. doi: 10.7554/eLife.18296.001

Greenwood, E. J. D., Williamson, J. C., Sienkiewicz, A., Naamati, A., Matheson, N. J., and Lehner, P. J. (2019). Promiscuous targeting of cellular proteins by Vpr drives systems-level proteomic remodeling in HIV-1 infection. *Cell Rep.* 27, 1579–1596.e7. doi: 10.1016/j.celrep.2019.04.025

Grishin, A., Voth, K., Gagarinova, A., and Cygler, M. (2021). Structural biology of the invasion arsenal of Gram-negative bacterial pathogens. *FEBS J.* doi: 10.1111/febs.15794 [Epub ahead of print].

Gudleski-O'Regan, N., Greco, T. M., Cristea, I. M., and Shenk, T. (2012). Increased expression of LDL receptor-related protein 1 during human cytomegalovirus infection reduces virion cholesterol and infectivity. *Cell Host Microbe* 12, 86–96. doi: 10.1016/j.chom.2012.05.012

Hall, S. L., Hester, S., Griffin, J. L., Lilley, K. S., and Jackson, A. P. (2009). The organelle proteome of the DT40 lymphocyte cell line. *Mol. Cell. Proteomics* 8, 1295–1305. doi: 10.1074/mcp.M800394-MCP200

Handl, J., Knowles, J., and Kell, D. B. (2005). Computational cluster validation in post-genomic data analysis. *Bioinformatics* 21, 3201–3212. doi: 10.1093/bioinformatics/bti517

Hair, J., Black, W., Babin, B., and Anderson, R. (2009). *Multivariate Data Analysis*, 7th Edn. New Jersey NJ: Pearson Prentice Hall: Upper Saddle River.

Hashimoto, Y., Sheng, X., Murray-Nerger, L. A., and Cristea, I. M. (2020). Temporal dynamics of protein complex formation and dissociation during human cytomegalovirus infection. *Nat. Commun.* 11:806. doi: 10.1038/s41467-020-14586-5

Huang, T., Choi, M., Tzouros, M., Golling, S., Pandya, N. J., Banfai, B., et al. (2020). MSstatsTMT: statistical detection of differentially abundant proteins in experiments with isobaric labeling and multiple mixtures. *Mol. Cell. Proteomics* 19, 1706–1723. doi: 10.1074/mcp.RA120.002105

Hill, E. G., Schwacke, J. H., Comte-Walters, S., Slate, E. H., Oberg, A. L., Eckel-Passow, J. E., et al. (2008). A statistical model for iTRAQ data analysis. *J. Proteome Res.* 7, 3091–3101. doi: 10.1021/pr070520u

Hou, J., Li, Z., Zhong, W., Hao, Q., Lei, L., Wang, L., et al. (2017). Temporal transcriptomic and proteomic landscapes of deteriorating pancreatic islets in type 2 diabetic rats. *Diabetes* 66, 2188–2200. doi: 10.2337/db16-1305

Hu, L. Z. M., Goebels, F., Tan, J. H., Wolf, E., Kuzmanov, U., Wan, C., et al. (2019). EPIC: software toolkit for elution profile-based inference of protein complexes. *Nat. Methods* 16, 737–742. doi: 10.1038/s41592-019-0461-4

Huber, W., Von Heydebreck, A., Sultmann, H., Poustka, A., and Vingron, M. (2002). Variance stabilization applied to microarray data calibration and to the quantification of differential expression. *Bioinformatics* 18(Suppl. 1), S96–S104. doi: 10.1093/bioinformatics/18.suppl_1.S96

Itzhak, D. N., Sacco, F., Nagaraj, N., Tyanova, S., Mann, M., and Murgia, M. (2019). SILAC-based quantitative proteomics using mass spectrometry quantifies endoplasmic reticulum stress in whole HeLa cells. *Dis. Model. Mech.* 12:dmm040741. doi: 10.1242/dmm.040741

Itzhak, D. N., Tyanova, S., Cox, J., and Borner, G. H. H. (2016). Global, quantitative and dynamic mapping of protein subcellular localization. *Elife* 5:e16950. doi: 10.7554/eLife.16950

Iyer, J., Grüner, A. C., Rénia, L., Snounou, G., and Preiser, P. R. (2007). Invasion of host cells by malaria parasites: a tale of two protein families. *Mol. Microbiol.* 65, 231–249. doi: 10.1111/j.1365-2958.2007.05791.x

Jean Beltran, P. M., Federspiel, J. D., Sheng, X., and Cristea, I. M. (2017). Proteomics and integrative omic approaches for understanding host–pathogen interactions and infectious diseases. *Mol. Syst. Biol.* 13:922. doi: 10.15252/msb.20167062

Jean Beltran, P. M., Mathias, R. A., and Cristea, I. M. (2016). A portrait of the human organelle proteome in space and time during cytomegalovirus infection. *Cell Syst.* 3, 361–373.e6. doi: 10.1016/j.cels.2016.08.012

Jo, E.-K. (2019). Interplay between host and pathogen: immune defense and beyond. *Exp. Mol. Med.* 51, 1–3. doi: 10.1038/s12276-019-0281-8

Kamaruzzaman, N. F., Kendall, S., and Good, L. (2017). Targeting the hard to reach: challenges and novel strategies in the treatment of intracellular bacterial infections. *Br. J. Pharmacol.* 174, 2225–2236. doi: 10.1111/bph.13664

Kammers, K., Cole, R. N., Tiengwe, C., and Ruczinski, I. (2015). Detecting significant changes in protein abundance. *EuPA Open Proteomics* 7, 11–19. doi: 10.1016/j.euprot.2015.02.002

Karniely, S., Weekes, M. P., Antrobus, R., Rorbach, J., Van Haute, L., Umrania, Y., et al. (2016). Human cytomegalovirus infection upregulates the mitochondrial transcription and translation machineries. *MBio* 7, e00029–016. doi: 10.1128/mBio.00029-16

Kanehisa, M., Goto, S., Sato, Y., Furumichi, M., and Tanabe, M. (2012). KEGG for integration and interpretation of large-scale molecular data sets. *Nucleic Acids Res.* 40, D109–D114. doi: 10.1093/nar/gkr988

Kang, H. (2013). The prevention and handling of the missing data. *Korean J. Anesthesiol.* 64, 402–406. doi: 10.4097/kjae.2013.64.5.402

Karimpour-Fard, A., Epperson, L. E., and Hunter, L. E. (2015). A survey of computational tools for downstream analysis of proteomic and other omic datasets. *Hum. Genomics* 9:28. doi: 10.1186/s40246-015-0050-2

Karpievitch, Y. V., Nikolic, S. B., Wilson, R., Sharman, J. E., and Edwards, L. M. (2014). Metabolomics data normalization with EigenMS. *PLoS One* 9:e116221. doi: 10.1371/journal.pone.0116221

Karpievitch, Y. V., Taverner, T., Adkins, J. N., Callister, S. J., Anderson, G. A., Smith, R. D., et al. (2009). Normalization of peak intensities in bottom-up MS-based proteomics using singular value decomposition. *Bioinformatics* 25, 2573–2580. doi: 10.1093/bioinformatics/btp426

Karpievitch, Y. V., Dabney, A. R., and Smith, R. D. (2012). Normalization and missing value imputation for label-free LC-MS analysis. *BMC Bioinformatics* 13:S5. doi: 10.1186/1471-2105-13-S16-S5

Karpievitch, Y., Stanley, J., Taverner, T., Huang, J., Adkins, J. N., Ansong, C., et al. (2009). A statistical framework for protein quantitation in bottom-up MS-based proteomics. *Bioinformatics* 25, 2028–2034. doi: 10.1093/bioinformatics/btp362

Kassambara, A., and Mundt, F. (2020). *factoextra: Extract and Visualize the Results of Multivariate Data Analyses. R Package Version 1.0.7*. Available online at: https://CRAN.R-project.org/package=factoextra (accessed April 2, 2020).

Kau, T. R., Way, J. C., and Silver, P. A. (2004). Nuclear transport and cancer: From mechanism to intervention. *Nat. Rev. Cancer* 4, 106–117. doi: 10.1038/nrc1274

Kautz, T., Eskofier, B. M., and Pasluosta, C. F. (2017). Generic performance measure for multiclass-classifiers. *Pattern Recognit.* 68, 111–125. doi: 10.1016/j.patcog.2017.03.008

Kerr, G., Ruskin, H. J., Crane, M., and Doolan, P. (2008). Techniques for clustering gene expression data. *Comput. Biol. Med.* 38, 283–293. doi: 10.1016/j.compbiomed.2007.11.001

Kim, H., Golub, G. H., and Park, H. (2005). Missing value estimation for DNA microarray gene expression data: local least squares imputation. *Bioinformatics* 21, 187–198. doi: 10.1093/bioinformatics/bth499

Konopka, T. (2020). *umap: Uniform Manifold Approximation and Projection. R Package Version 3.1.2.* Available online at: https://CRAN.R-project.org/package=umap (accessed November 4, 2020).

Kohonen, T. (1990). The self-organizing map. *Proc. IEEE* 78, 1464–1480. doi: 10.1109/5.58325

Kohonen, T. (2012). *Self-Organizing Maps*. Berlin: Springer Science & Business Media.

Kohonen, T. (2013). Essentials of the self-organizing map. *Neural Networks* 37, 52–65. doi: 10.1016/j.neunet.2012.09.018

Kohonen, T. (2014). *MATLAB Implementations and Applications of the Self-Organizing Map*. Available online at: http://docs.unigrafia.fi/publications/kohonen_teuvo/ (accessed December 10, 2014).

Kotsiantis, S. B., Zaharakis, I., and Pintelas, P. (2007). Supervised machine learning: a review of classification techniques. *Emerg. Artif. Intell. Appl. Comput. Eng.* 160, 3–24.

Kucukural, A., Yukselen, O., Ozata, D. M., Moore, M. J., and Garber, M. (2019). DEBrowser: interactive differential expression analysis and visualization tool for count data. *BMC Genomics* 20:6. doi: 10.1186/s12864-018-5362-x

Kuhn, M., Wing, J., Weston, S., Williams, A., Keefer, C., Engelhardt, A., et al. (2020). *caret: Classification and Regression Training. R Package Version 6.0–86.* Available online at: https://CRAN.R-project.org/package=caret (accessed March 21, 2020).

Kumar, C., and Mann, M. (2009). Bioinformatics analysis of mass spectrometry-based proteomics data sets. *FEBS Lett.* 583, 1703–1712. doi: 10.1016/j.febslet.2009.03.035

Kumar, L., and Futschik, M. E. (2007). Mfuzz: a software package for soft clustering of microarray data. *Bioinformation* 2:5.

Lapek, J. D., Lewinski, M. K., Wozniak, J. M., Guatelli, J., and Gonzalez, D. J. (2017). Quantitative temporal viromics of an inducible HIV-1 model yields insight to global host targets and phospho-dynamics associated with protein Vpr. *Mol. Cell. Proteomics* 16, 1447–1461. doi: 10.1074/mcp.M116.066019

Laurila, K., and Vihinen, M. (2009). Prediction of disease-related mutations affecting protein localization. *BMC Genomics* 10:122. doi: 10.1186/1471-2164-10-122

Leek, J. T., and Storey, J. D. (2007). Capturing heterogeneity in gene expression studies by surrogate variable analysis. *PLoS Genet* 3:e161. doi: 10.1371/journal.pgen.0030161

Lever, J., Krzywinski, M., and Altman, N. (2017). Points of Significance: principal component analysis. *Nat. Methods* 14, 641–642. doi: 10.1038/nmeth.4346

Li, M., Tu, S., Li, Z., Tan, F., Liu, J., Wang, Q., et al. (2019). MAP: model-based analysis of proteomic data to detect proteins with significant abundance changes. *Cell Discov.* 5:40. doi: 10.1038/s41421-019-0107-9

Li, Z., Liu, H., Niu, Z., Zhong, W., Xue, M., Wang, J., et al. (2018). Temporal proteomic analysis of pancreatic β-cells in response to lipotoxicity and glucolipotoxicity. *Mol. Cell. Proteomics* 17, 2119–2131. doi: 10.1074/mcp.RA118.000698

Liu, Y., Li, Z., Xiong, H., Gao, X., and Wu, J. (2010). "Understanding of internal clustering validation measures," in *Proceedings of the IEEE International. Conference. Data Mining*, (New York City NY: IEEE), 911–916. doi: 10.1109/ICDM.2010.35

Lopez, V., Villar, M., Queiros, J., Vicente, J., Mateos-Hernández, L., Díez-Delgado, I., et al. (2016). Comparative proteomics identifies host immune system proteins affected by infection with *Mycobacterium bovis*. *PLoS Negl. Trop. Dis.* 10:e0004541. doi: 10.1371/journal.pntd.0004541

Love, M. I., Huber, W., and Anders, S. (2014). Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. *Genome Biol.* 15:550. doi: 10.1186/s13059-014-0550-8

Luheshi, L. M., Crowther, D. C., and Dobson, C. M. (2008). Protein misfolding and disease: from the test tube to the organism. *Curr. Opin. Chem. Biol.* 12, 25–31. doi: 10.1016/j.cbpa.2008.02.011

Luo, G. (2016). A review of automatic selection methods for machine learning algorithms and hyper-parameter values. *Netw. Model. Anal. Heal. Informatics Bioinforma.* 5:18. doi: 10.1007/s13721-016-0125-6

Luo, R., Colangelo, C. M., Sessa, W. C., and Zhao, H. (2009). Bayesian analysis of iTRAQ data with nonrandom missingness: identification of differentially expressed proteins. *Stat. Biosci.* 1, 228–245. doi: 10.1007/s12561-009-9013-2

Maechler, M., Rousseeuw, P., Struyf, A., Hubert, M., and Hornik, K. (2019). *cluster: Cluster Analysis Basics and Extensions. R Package Version 2.1.0.* Available online at: https://CRAN.R-project.org/package=cluster (accessed June 20, 2019).

Magrane, M., and Consortium, U. P. (2011). UniProt knowledgebase: a hub of integrated protein data. *Database* 2011:bar009. doi: 10.1093/database/bar009

Margolin, A. A., Ong, S.-E., Schenone, M., Gould, R., Schreiber, S. L., Carr, S. A., et al. (2009). Empirical bayes analysis of quantitative proteomics experiments. *PLoS One* 4:e7454. doi: 10.1371/journal.pone.0007454

Matheson, N. J., Sumner, J., Wals, K., Rapiteanu, R., Weekes, M. P., Vigan, R., et al. (2015). Cell surface proteomic map of HIV infection reveals antagonism of amino acid metabolism by Vpu and Nef. *Cell Host Microbe* 18, 409–423. doi: 10.1016/j.chom.2015.09.003

May, R. C., and Casadevall, A. (2018). In fungal intracellular pathogenesis, form determines fate. *MBio* 9, e02092–018. doi: 10.1128/mBio.02092-18

McInnes, L., Healy, J., and Melville, J. (2018). Umap: uniform manifold approximation and projection for dimension reduction. *J. Open Source Softw.* 3:861. doi: 10.21105/joss.00861

Merico, D., Isserlin, R., Stueker, O., Emili, A., and Bader, G. D. (2010). Enrichment map: a network-based method for gene-set enrichment visualization and interpretation. *PLoS One* 5:e13984. doi: 10.1371/journal.pone.0013984

Meunier, B., Dumas, E., Piec, I., Béchet, D., Hébraud, M., and Hocquette, J. F. (2007). Assessment of hierarchical clustering methodologies for proteomic data mining. *J. Proteome Res.* 6, 358–366. doi: 10.1021/pr060343h

Meyer, D., Dimitriadou, E., Hornik, K., Weingessel, A., Leisch, F., Chang, C. C., et al. (2020). *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien. R Package Version 1.7–4.* Available online at: https://CRAN.R-project.org/package=e1071 (accessed October 15, 2020).

Mi, H., Muruganujan, A., Casagrande, J. T., and Thomas, P. D. (2013). Large-scale gene function analysis with the PANTHER classification system. *Nat. Protoc.* 8, 1551–1566. doi: 10.1038/nprot.2013.092

Miranda-CasoLuengo, A. A., Staunton, P. M., Dinan, A. M., Lohan, A. J., and Loftus, B. J. (2016). Functional characterization of the *Mycobacterium abscessus* genome coupled with condition specific transcriptomics reveals conserved molecular strategies for host adaptation and persistence. *BMC Genomics* 17:553. doi: 10.1186/s12864-016-2868-y

Mouselimis, L., Sanderson, C., Curtin, R., Agrawal, S., Frey, B., and Dueck, D. (2020). *ClusterR: Gaussian Mixture Models, K-Means, Mini-Batch-Kmeans, K-Medoids and Affinity Propagation Clustering. R Package Version 1.2.2.* Available online at: https://CRAN.R-project.org/package=ClusterR (accessed March 13, 2020).

Murie, C., Sandri, B., Sandberg, A. S., Griffin, T. J., Lehtiö, J., Wendt, C., et al. (2018). Normalization of mass spectrometry data (NOMAD). *Adv. Biol. Regul.* 67, 128–133. doi: 10.1016/j.jbior.2017.11.005

Naim, A., Ratnatunga, K. U., and Griffiths, R. E. (1997). Galaxy morphology without classification: self-organizing maps. *Astrophys. J. Suppl. Ser.* 111, 357–367.

Nesvizhskii, A. I. (2014). Proteogenomics: concepts, applications and computational strategies. *Nat. Methods* 11, 1114–1125. doi: 10.1038/nmeth.3144

Nusinow, D. P., and Gygi, S. P. (2020). A guide to the quantitative proteomic profiles of the cancer cell line encyclopedia. *bioRxiv*[Preprint] doi: 10.1101/2020.02.03.932384 bioRxiv 2020.02.03.932384,

Oh, S., Kang, D. D., Brock, G. N., and Tseng, G. C. (2011). Biological impact of missing-value imputation on downstream analyses of gene expression profiles. *Bioinformatics* 27, 78–86. doi: 10.1093/bioinformatics/btq613

Olsen, J. V., Blagoev, B., Gnad, F., Macek, B., Kumar, C., Mortensen, P., et al. (2006). Global, in vivo, and site-specific phosphorylation dynamics in signaling networks. *Cell* 127, 635–648. doi: 10.1016/j.cell.2006.09.026

Orre, L. M., Vesterlund, M., Pan, Y., Arslan, T., Zhu, Y., Fernandez Woodbridge, A., et al. (2019). SubCellBarCode: proteome-wide mapping of protein localization and relocalization. . *Mol. Cell* 73, 166–182.e7. doi: 10.1016/j.molcel.2018.11.035

Oyelade, J., Isewon, I., Oladipupo, F., Aromolaran, O., Uwoghiren, E., Aameh, F., et al. (2016). Clustering algorithms: their application to gene expression data. *Bioinform. Biol. Insights* 10, 237–253. doi: 10.4137/BBI.S38316

Park, S., Yang, J., Shin, Y., Park, J., Jang, S. K., and Kim, S. (2011). Protein localization as a principal feature of the etiology and comorbidity of genetic diseases. *Mol. Syst. Biol.* 7:494. doi: 10.1038/msb.2011.29

Payam, R., Lei, T., and Huan, L. (2009). "Cross-validation," in *Encyclopedia of Database Systems*, eds L. Liu and M. T. Özsu (Boston, MA: Springer), 532–538.

Peng, Y., Li, X., Wu, M., Yang, J., Liu, M., Zhang, W., et al. (2012). New prognosis biomarkers identified by dynamic proteomic analysis of colorectal cancer. *Mol. Biosyst.* 8, 3077–3088. doi: 10.1039/c2mb25286d

Probst, P., Boulesteix, A.-L., and Bischl, B. (2019). Tunability: importance of hyperparameters of machine learning algorithms. *J. Mach. Learn. Res. arXiv*[Preprint] 20, arXiv 1802.09596,

Purohit, P. V., and Rocke, D. M. (2003). Discriminant models for high-throughput proteomics mass spectrometer data. *Proteomics* 3, 1699–1703. doi: 10.1002/pmic.200300518

Quackenbush, J. (2002). Microarray data normalization and transformation. *Nat. Genet.* 32, 496–501. doi: 10.1038/ng1032

Rauniyar, N., and Yates, J. R. III (2014). Isobaric labeling-based relative quantification in shotgun proteomics. *J. Proteome Res.* 13, 5293–5309. doi: 10.1021/pr500880b

RStudio Team (2020). *RStudio: Integrated Development for R.* Boston, MA: RStudio, PBC. Available online at: http://www.rstudio.com/

Reimand, J., Kull, M., Peterson, H., Hansen, J., and Vilo, J. (2007). g:Profiler—a web-based toolset for functional profiling of gene lists from large-scale experiments. *Nucleic Acids Res.* 35, W193–W200. doi: 10.1093/nar/gkm226

Ressom, H., Wang, D., and Natarajan, P. (2003). Clustering gene expression data using adaptive double self-organizing map. *Physiol. Genomics* 14, 35–46. doi: 10.1152/physiolgenomics.00138.2002

Ritchie, M. E., Phipson, B., Wu, D., Hu, Y., Law, C. W., Shi, W., et al. (2015). limma powers differential expression analyses for RNA-sequencing and microarray studies. *Nucleic Acids Res.* 43:e47. doi: 10.1093/nar/gkv007

Ribet, D., and Cossart, P. (2010). Post-translational modifications in host cells during bacterial infection. *FEBS Lett.* 584, 2748–2758. doi: 10.1016/j.febslet.2010.05.012

Robinson, M. D., and Oshlack, A. (2010). A scaling normalization method for differential expression analysis of RNA-seq data. *Genome Biol.* 11:R25. doi: 10.1186/gb-2010-11-3-r25

Robinson, M. D., Mccarthy, D. J., and Smyth, G. K. (2010). edgeR: a bioconductor package for differential expression analysis of digital gene expression data. *Bioinformatics* 26, 139–140. doi: 10.1093/bioinformatics/btp616

Rojas-Domínguez, A., Padierna, L. C., Valadez, J. M. C., Puga-Soberanes, H. J., and Fraire, H. J. (2017). Optimal hyper-parameter tuning of SVM classifiers with application to medical diagnosis. *IEEE Access* 6, 7164–7176. doi: 10.1109/ACCESS.2017.2779794

Roux, M. (2018). A comparative study of divisive and agglomerative hierarchical clustering algorithms. *J. Classif.* 35, 345–366. doi: 10.1007/s00357-018-9259-9

Sainburg, T., McInnes, L., and Gentner, T. Q. (2020). Parametric UMAP: learning embeddings with deep neural networks for representation and semi-supervised learning. *arXiv* [Preprint] arXiv2009.12981,

Sánchez-Quiles, V., Mora, M. I., Segura, V., Greco, A., Epstein, A. L., Foschini, M. G., et al. (2011). HSV-1 Cgal+ infection promotes quaking RNA binding protein production and induces nuclear-cytoplasmic shuttling of quaking I-5 isoform in human hepatoma cells. *Mol. Cell. Proteomics* 10, M111–M009126. doi: 10.1074/mcp.M111.009126

Santana-Codina, N., Chandhoke, A. S., Yu, Q., Małachowska, B., Kuljanin, M., Gikandi, A., et al. (2020). Defining and targeting adaptations to oncogenic KRASG12C Inhibition using quantitative temporal proteomics. *Cell Rep.* 30, 4584–4599.e4. doi: 10.1016/j.celrep.2020.03.021

Savitski, M. M., Mathieson, T., Zinn, N., Sweetman, G., Doce, C., Becher, I., et al. (2013). Measuring and managing ratio compression for accurate iTRAQ/TMT quantification. *J. Proteome Res.* 12, 3586–3598. doi: 10.1021/pr400098r

Sayers, C. (1991). *Self Organizing Feature Maps and Their Applications to Robotics. University of Pennsylvania Department of Computer and Information Science Technical Report No. MS-CIS-91-46. May 1991*. Philadelphia PA: University of Pennsylvania.

Schmidt, M. W., Houseman, A., Ivanov, A. R., and Wolf, D. A. (2007). Comparative proteomic and transcriptomic profiling of the fission yeast *Schizosaccharomyces pombe*. *Mol. Syst. Biol.* 3:79. doi: 10.1038/msb4100117

Schmutz, C., Ahrné, E., Kasper, C. A., Tschon, T., Sorg, I., Dreier, R. F., et al. (2013). Systems-level overview of host protein phosphorylation during *Shigella flexneri* infection revealed by phosphoproteomics. *Mol. Cell. Proteomics* 12, 2952–2968. doi: 10.1074/mcp.M113.029918

Schratz, P., Muenchow, J., Iturritxa, E., Richter, J., and Brenning, A. (2019). Hyperparameter tuning and performance assessment of statistical and machine-learning algorithms using spatial data. *Ecol. Modell.* 406, 109–120. doi: 10.1016/j.ecolmodel.2019.06.002

Schwämmle, V., León, I. R., and Jensen, O. N. (2013). Assessment and improvement of statistical tools for comparative proteomics analysis of sparse data sets with few experimental replicates. *J. Proteome Res.* 12, 3874–3883. doi: 10.1021/pr400045u

Schweppe, D. K., Harding, C., Chavez, J. D., Wu, X., Ramage, E., Singh, P. K., et al. (2015). Host-microbe protein interactions during bacterial infection. *Chem. Biol.* 22, 1521–1530. doi: 10.1016/j.chembiol.2015.09.015

Scott, N. E., and Hartland, E. L. (2017). Post-translational mechanisms of host subversion by bacterial effectors. *Trends Mol. Med.* 23, 1088–1102. doi: 10.1016/j.molmed.2017.10.003

Selkrig, J., Li, N., Hausmann, A., Mangan, M. S. J., Zietek, M., Mateus, A., et al. (2020). Spatiotemporal proteomics uncovers cathepsin-dependent macrophage cell death during *Salmonella* infection. *Nat. Microbiol.* 5, 1119–1133. doi: 10.1038/s41564-020-0736-7

Shah, P. S., Wojcechowskyj, J. A., Eckhardt, M., and Krogan, N. J. (2015). Comparative mapping of host–pathogen protein–protein interactions. *Curr. Opin. Microbiol.* 27, 62–68. doi: 10.1016/j.mib.2015.07.008

Sherman, B. T., Huang, D. W., Tan, Q., Guo, Y., Bour, S., Liu, D., et al. (2007). DAVID Knowledgebase: a gene-centered database integrating heterogeneous gene annotation resources to facilitate high-throughput gene functional analysis. *BMC Bioinformatics* 8:426. doi: 10.1186/1471-2105-8-426

Shirkhorshidi, A. S., Aghabozorgi, S., and Wah, T. Y. (2015). A comparison study on similarity and dissimilarity measures in clustering continuous data. *PLoS One* 10:e0144059. doi: 10.1371/journal.pone.0144059

Siljee, J. E., Wang, Y., Bernard, A. A., Ersoy, B. A., Zhang, S., Marley, A., et al. (2018). Subcellular localization of MC4R with ADCY3 at neuronal primary cilia underlies a common pathway for genetic predisposition to obesity. *Nat. Genet.* 50, 180–185. doi: 10.1038/s41588-017-0020-9

Simula, O., Vasara, P., Vesanto, J., and Helminen, R. (1999). "The self-organizing map in industry analysis," in *Intelligent Techniques in Industry*, eds L. Jain and V. Vemuri (Boca Raton, FL: CRC Press), 87–112.

Smedley, D., Haider, S., Ballester, B., Holland, R., London, D., Thorisson, G., et al. (2009). BioMart–biological queries made easy. *BMC Genomics* 10:22. doi: 10.1186/1471-2164-10-22

Smyth, G. K. (2004). Linear models and empirical bayes methods for assessing differential expression in microarray experiments. *Stat. Appl. Genet. Mol. Biol.* 3:3. doi: 10.2202/1544-6115.1027

Soday, L., Lu, Y., Albarnaz, J. D., Davies, C. T. R., Antrobus, R., Smith, G. L., et al. (2019). Quantitative temporal proteomic analysis of vaccinia virus infection reveals regulation of histone deacetylases by an interferon antagonist. *Cell Rep.* 27, 1920–1933.e7. doi: 10.1016/j.celrep.2019.04.042

Stacklies, W., Redestig, H., Scholz, M., Walther, D., and Selbig, J. (2007). Gene expression pcaMethods-a bioconductor package providing PCA methods for incomplete data. *Bioinformatics* 23, 1164–1167. doi: 10.1093/bioinformatics/btm069

Stefanovič, P., and Kurasova, O. (2011). "Influence of learning rates and neighboring functions on self-organizing maps," in *Advances in Self-Organizing Maps. WSOM 2011*, eds J. Laaksonen and T. T. Honkela (Berlin: Springer), 141–150.

Swan, A. L., Mobasheri, A., Allaway, D., Liddell, S., and Bacardit, J. (2013). Application of machine learning to proteomics data: classification and biomarker identification in postgenomics biology. *Omi. J. Integr. Biol.* 17, 595–610. doi: 10.1089/omi.2013.0017

Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Kitareewan, S., Dmitrovsky, E., et al. (1999). Interpreting patterns of gene expression with self-organizing maps: Methods and application to hematopoietic differentiation. *Proc. Natl. Acad. Sci. U.S.A.* 96, 2907–2912. doi: 10.1073/pnas.96.6.2907

Tan, D. J. L., Dvinge, H., Christoforou, A., Bertone, P., Arias, A. M., and Lilley, K. S. (2009). Mapping organelle proteins and protein complexes in *Drosophila* melanogaster. *J. Proteome Res.* 8, 2667–2678. doi: 10.1021/pr800866n

Tang, Y., Horikoshi, M., and Li, W. (2016). ggfortify: unified interface to visualize statistical results of popular R packages. *R J.* 8, 478–489. doi: 10.32614/RJ-2016-060

Thalamuthu, A., Mukhopadhyay, I., Zheng, X., and Tseng, G. C. (2006). Evaluation and comparison of gene clustering methods in microarray analysis. *Bioinformatics* 22, 2405–2412. doi: 10.1093/bioinformatics/btl406

Tiku, V., Tan, M. W., and Dikic, I. (2020). Mitochondrial functions in infection and immunity. *Trends Cell Biol.* 30, 263–275. doi: 10.1016/j.tcb.2020.01.006

Ting, L., Cowley, M. J., Hoon, S. L., Guilhaus, M., Raftery, M. J., and Cavicchioli, R. (2009). Normalization and statistical analysis of quantitative proteomics data generated by metabolic labeling. *Mol. Cell. Proteomics* 8, 2227–2242. doi: 10.1074/mcp.M800462-MCP200

Tomasec, P., Wang, E. C. Y., Davison, A. J., Vojtesek, B., Armstrong, M., Griffin, C., et al. (2005). Downregulation of natural killer cell-activating ligang CD155 by human cytomegalovirus UL141. *Nat. Immunol.* 6, 181–188. doi: 10.1038/ni1156

Trotter, M. W. B., Sadowski, P. G., Dunkley, T. P. J., Groen, A. J., and Lilley, K. S. (2010). Improved sub-cellular resolution via simultaneous analysis of organelle proteomics data across varied experimental conditions. *Proteomics* 10, 4213–4219. doi: 10.1002/pmic.201000359

Tusher, V. G., Tibshirani, R., and Chu, G. (2001). Significance analysis of microarrays applied to the ionizing radiation response. *Proc. Natl. Acad. Sci.U.S.A.* 98, 5116–5121. doi: 10.1073/pnas.091062498

Valastyan, J. S., and Lindquist, S. (2014). Mechanisms of protein-folding diseases at a glance. *Dis. Model. Mech.* 7, 9–14. doi: 10.1242/dmm.013474

Välikangas, T., Suomi, T., and Elo, L. L. (2018). A systematic evaluation of normalization methods in quantitative label-free proteomics. *Brief. Bioinform.* 19, 1–11. doi: 10.1093/bib/bbw095

Van Buuren, S., and Groothuis-Oudshoorn, K. (2010). mice: multivariate imputation by chained equations in R. *J. Stat. Softw.* 45, 1–67. doi: 10.18637/jss.v045.i03

Van Der Maaten, L., and Hinton, G. (2008). Visualizing data using t-SNE. *J. Mach. Learn. Res.* 9, 2579–2605.

Venables, W. N., and Ripley, B. D. (2002). *Modern Applied Statistics With S.* New York, NY: Springer Science & Business Media, doi: 10.1007/978-0-387-21706-2

Weekes, M. P., Tomasec, P., Huttlin, E. L., Fielding, C. A., Nusinow, D., Stanton, R. J., et al. (2014). Quantitative temporal viromics: an approach to investigate host-pathogen interaction. *Cell* 157, 1460–1472. doi: 10.1016/j.cell.2014.04.028

Huang, D. W., Sherman, B. T., and Lempicki, R. A. (2009). Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists. *Nucleic Acids Res.* 37, 1–13. doi: 10.1093/nar/gkn923

Wei, R., Wang, J., Jia, E., Chen, T., Ni, Y., and Jia, W. (2018). GSimp: a Gibbs sampler based left-censored missing value imputation approach for metabolomics studies. *PLoS Comput. Biol.* 14:e1005973.

Wehrens, R., and Kruisselbrink, J. (2019). *kohonen: Supervised and Unsupervised Self-Organising Maps R Package Version 3.0.10*. Available online at: https://CRAN.R-project.org/package=kohonen. (accessed December 16, 2019).

Wisìniewski, J. R., and Mann, M. (2016). A proteomics approach to the protein normalization problem: selection of unvarying proteins for MS-based proteomics and western blotting. *J. Proteome Res.* 15, 2321–2326. doi: 10.1021/acs.jproteome.6b00403

Xu, J., Bankov, G., Kim, M., Wretlind, A., Lord, J., Green, R., et al. (2020). Integrated lipidomics and proteomics network analysis highlights lipid and immunity pathways associated with Alzheimer's disease. *Transl Neurodegener.* 9:36. doi: 10.1186/s40035-020-00215-0

Yang, P., Zheng, X., Jayaswal, V., Hu, G., Yang, J. Y. H., and Jothi, R. (2015). Knowledge-based analysis for detecting key signaling events from time-series phosphoproteomics data. *PLoS Comput. Biol.* 11:e1004403. doi: 10.1371/journal.pcbi.1004403

Yin, L., Huang, C. H., and Ni, J. (2006). Clustering of gene expression data: Performance and similarity analysis. *BMC Bioinformatics* 7(Suppl. 4):S19. doi: 10.1186/1471-2105-7-S4-S19

Yuan, C., and Yang, H. (2019). Research on K-value selection method of K-means clustering algorithm. *J.* 2, 226–235. doi: 10.3390/j2020016

Zhang, Y., Askenazi, M., Jiang, J., Luckey, C. J., Griffin, J. D., and Marto, J. A. (2010). A robust error model for iTRAQ quantification reveals divergent signaling between oncogenic FLT3 mutants in acute myeloid leukemia. *Mol. Cell. Proteomics* 9, 780–790. doi: 10.1074/mcp.M900452-MCP200

Zhang, Y., Wen, Z., Washburn, M. P., and Florens, L. (2015). Improving label-free quantitative proteomics strategies by distributing shared peptides and stabilizing variance. *Anal. Chem.* 87, 4749–4756. doi: 10.1021/ac504740p

Zhang, Y., Wolf-Yadlin, A., Ross, P. L., Pappin, D. J., Rush, J., Lauffenburger, D. A., et al. (2005). Time-resolved mass spectrometry of tyrosine phosphorylation sites in the epidermal growth factor receptor signaling network reveals dynamic modules. *Mol. Cell. Proteomics* 4, 1240–1250. doi: 10.1074/mcp.M500089-MCP200

Zhang, X., Smits, A. H., van Tilburg, G. B. A., Ovaa, H., Huber, W., and Vermeulen, M. (2018). Proteome-wide identification of ubiquitin interactions using UbIA-MS. *Nat. Protoc.* 13:530. doi: 10.1038/nprot.2017.147

Zhao, S., Li, R., Cai, X., Chen, W., Li, Q., Xing, T., et al. (2013). The application of SILAC mouse in human body fluid proteomics analysis reveals protein patterns associated with IgA nephropathy. *Evidence Based Complement. Altern. Med.* 2013:275390. doi: 10.1155/2013/275390

Zhou, C., Walker, M. J., Williamson, A. J. K., Pierce, A., Berzuini, C., Dive, C., et al. (2014a). A hierarchical statistical modeling approach to analyze proteomic isobaric tag for relative and absolute quantitation data. *Bioinformatics* 30, 549–558. doi: 10.1093/bioinformatics/btt722

Zhou, C., Simpson, K. L., Lancashire, L. J., Walker, M. J., Dawson, M. J., Unwin, R. D., et al. (2012). Statistical considerations of optimal study design for human plasma proteomics and biomarker discovery. *J. Proteome Res.* 11, 2103–2113. doi: 10.1021/pr200636x

Zhou, K., Le, Fu, C., and Yang, S. L. (2014b). Fuzziness parameter selection in fuzzy c-means: the perspective of cluster validation. *Sci. China Inf. Sci.* 57, 1–8. doi: 10.1007/s11432-014-5146-0

Zhou, Y., Wang, H., Guo, F., Si, N., Brantner, A., Yang, J., et al. (2018). Integrated Proteomics and lipidomics investigation of the mechanism underlying the neuroprotective effect of N-benzylhexadecanamide. *Molecules* 23:2929. doi: 10.3390/molecules23112929

Zhuang, G., Yu, K., Jiang, Z., Chung, A., Yao, J., Ha, C., et al. (2013). Phosphoproteomic analysis implicates the mTORC2-FoxO1 Axis in VEGF signaling and feedback activation of receptor tyrosine kinases. *Sci. Signal.* 6:ra25. doi: 10.1126/scisignal.2003572

Conflict of Interest: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Copyright © 2021 Rahmatbakhsh, Gagarinova and Babu. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.