TARAM

Table of Contents

About this document

This document outlines the various steps available in TARAM. It is not intended to serve as a user manual.

Introduction

TARAM is a proteomics analysis pipeline developed in-house by the Protein Analysis Facility at the University of Lausanne. This web-based tool requires installation on a server[1] (refer to the Availability and installation section for more details).

TARAM was developed with four goals in mind:

1. **Flexible workflow:** there are various data analysis steps available which can be chained one after the other. These steps can be added or removed at any position within the chain and each step can be utilized multiple times.
2. **Automatic recomputation:** If a step is added, removed, or its parameters are modified, all subsequent steps are automatically recomputed.
3. **Workflow comparison:** A chain of steps can be copied, and the copy can be edited. This is a handy way to compare different analysis approaches and assess the impact of parameter changes.
4. **Keep and share results:** Analysis are stored on the server, and since the application is web based, results can be easily shared via a link. Additionally, results can be exported as a PDF file or a ZIP file containing all plots and tables.

Availability and installation

TARAM is free and open source. For installation instructions and technical details, visit our GitHub repository, which is organized into three sections:

- TARAM backend
- TARAM user-interface
- TARAM svg

Description of data analysis steps

This section provides a list of the currently available data analysis steps. They are loosely ordered by what a typical analysis workflow might look like.

Initial results

This is the first step of any TARAM analysis. In this step the proteomics results from MaxQuant or Spectronaut are parsed. The following table explains which result files are used:

[1] It can also run as a local instance on a personal computer, although it is not primarily designed for this purpose.

TARAM	MaxQuant	Spectronaut
Table 1	txt/proteinGroups.txt	*_Report.tsv (or similar)
Databases/Libraries/Parameters	txt/parameters.txt	*_Report.setup.txt
Groups	-	*_Report.setup.txt
Experiment names	txt/summary.txt	*_Report.tsv

If multiple intensity columns are available, it can be selected which one should be used (e.g. iBAQ or Intensity).

MaxQuant parsing:

The initial table (Table 1) is essentially the *proteinGroups.txt*. All spaces in the column names are replaced by dots (e.g. "Gene names" gets "Gene.names"). If the "Gene.names" or "Protein.names" columns are missing or incomplete, they will be generated or completed by parsing the "Fasta.headers".

Spectronaut parsing:

Spectronaut provides a table with the ending *_Report.tsv* (or .txt or xls) which serves as the initial table (Table 1). TARAM tries to reduce the typically very long column names, while keeping the essential information. TARAM can parse the group information (if available) from the *_Report_setup.txt* file.

Remove columns

This step simply allows to remove unused columns from the table. This is especially useful to make the table clearer and to reduce disk usage.

Order columns

The order of the columns can be changed. It can be useful to put the columns with the selected intensity values first.

Rename columns

The group names can be added in the front of the column names. The new column names resulting in something like "*Group_1.Sample_1.iBAQ*".

Add column

TARAM can add new columns based on information from already existing columns:

- If the selected columns contain numerical data, you can calculate their minimum, maximum, mean, or sum and store the result in a new column.
- If the selected column contains character data, you can search for a specific expression (with the asterisk * as a wildcard) and create a new column marking each position where the expression is found with a '+' symbol.

Filter rows

The protein-group tables usually contain some irrelevant data (e.g. contaminants or reverse hits) which are not of biological interest. It can also be interesting to remove low quality hits.

Rows are removed, based on defined criteria for a specified column:

- If the column is numerical, rows that are smaller, greater, equal or not equal a given value are kept or removed.
- If the column is character based, rows that match or do not match a given character are kept or removed (asterisk * matches anything).

There are also three predefined filters for cases that are commonly removed. Two of them are only available for MaxQuant data. The following table shows which rows are removed when the defined columns have a match:

Predefined filter	**MaxQuant**	**Spectronaut**
only-identified-by-site	*Only.identified.by.site* = +	-
reverse	*Reverse* = +	-
potential contaminant	*Potential.contaminant* = +	*PG.ProteinGroups* = *Cont_**

Filter on valid

Often it is appropriate to filter out Protein groups that have many missing values. Those results are typically of low confidence and can lead to false significant results when doing statistical tests.

This method removes all rows that have less then the indicated number of valid (non missing) values in either one group, in all groups or in total.

Log transformation

Since protein-expression tends to be log normal, the intensity signal is usually log2 transformed at some point during the analysis workflow. This facilitates plotting of data and allows application of statistical tests.

Boxplot

Boxplots are a convenient way to display and compare samples and groups. It shows the five following numbers:

- **Bottom whisker**: the **minimum** (Q_0 or 0th percentile), which corresponds to the lowest data point (no outliers are considered).
- **Bottom box border**: the **first quartile** (Q_1 or 25th percentile), corresponds to the median of the lower half of the dataset.

- **Middle box line**: the **median** (Q_2 or 50th percentile), corresponds to the middle value of the dataset.
- **Top box border**: the **third quartile** (Q_3 or 75th percentile), corresponds to the median of the upper half of the dataset.
- **Top whisker**: the **maximum** (Q_4 or 100th percentile), corresponds to the highest data point (no outliers are considered).

TARAM computes the quantiles with Google Common Quantiles.

Summary

Summary statistics show a summary of the data and can be useful to compare samples and groups among each other. A separate summary table is created containing the following fields: *min, max, mean, median, sum, standard deviation, standard error, coefficient of variance, number of valid values* and *number of NaN values*.

The median is computed with Google Common Quantiles and standard deviation with Apache Commons StandardDeviation.

Normalization

Normalization of the samples is used to compensate for global differences in sample amount. Median subtraction is the simplest and most conservative normalization. Some intensity data such as MaxQuant LFQ or Spectronaut Quantity are already normalized, so they don't need any normalization.

Impute missing values

Missing values (NaN) are a common issue in Proteomics. These gaps cannot be used to calculate fold changes and can also hinder certain statistical methods. Generally, missing values diminish the power of statistical inference.

Imputation is the process of replacing NaN's by a value. TARAM proposes two imputation methods:

1. Drawing random values from a distribution meant to simulate expression below detection limit: TARAM re-implements the imputation method used in Perseus, where the random values are generated by following a normal distribution with:
    a. **mean** = (mean of intensities) – (standard deviation of intensities) * *downshift*
    b. **standard deviation =** (standard deviation of intensities) * *width*

   whereas the parameters *width* and *downshift* can be chosen (by default 0.3, respectively 1.8). The seed parameter allows for "reproducable" random values. TARAM has a Kotlin based implementation of this method (to be found in replaceByNormal).

The mean and standard deviation of intensities are calculated separately for each column, resulting in a distinct normal distribution for each column.

2. Replace all missing values by a (usually low) fixed value, set by the user.

TARAM stores the information of which positions were imputed (column and protein group) in a boolean matrix. This information are used by the steps Remove imputed values and t-test.

PCA

Principal component analysis (PCA) is a statistical technique used to reduce high dimensional data, in our case all protein groups, to reveal the internal structure of the data in a way that best explains the global variance. This is makes it a very useful tool to examin similarities between samples and groups.

TARAM uses prcomp from R with default parameters. By default prcomp does not accept any missing values. This implies that either filtering or imputation must be used before PCA.

UMAP

Uniform Manifold Approximation and Projection (UMAP) is, similarly to PCA, a method to reduce high data dimensionality. They main difference between PCA and UMAP is that the latter is a non-linear method. UMAP works well on data with many samples and conditions.

TARAM uses the umap package from R. There are two parameters which can be defined in TARAM:

1. **Number of neighbours:** This parameter controls how UMAP balances local versus global structure in the data. Low values will force UMAP to concentrate on very local structure (potentially to the detriment of the big picture), while large values will push UMAP to look at larger neighborhoods of each point when estimating the manifold structure of the data, losing fine detail structure for the sake of getting the broader pictureof the data.
2. **Minimum distance:** The minimum distance parameter controls how tightly UMAP is allowed to pack points together. It, quite literally, provides the minimum distance apart that points are allowed to be in the low dimensional representation. This means that low values will result in clumpier embeddings. This can be useful if you are interested in clustering, or in finer topological structure. Larger values will prevent UMAP from packing points together and will focus on the preservation of the broad topological structure instead. The default value is 0.1. You can use a range of values from 0.01 to 0.99.

For the other parameters, their default value is used.

T-test

A t-test is a statistical test used to determine whether there is a significant difference between the means of two groups. TARAM uses the JAVA library Apache Commons TTest to compute t-tests. The default t-test in TARAM corresponds to an independent two-sample t-test (homoscedastic) and is computed with the function *homoscedasticTTest*. The paired t-test uses the function *pairedTTest*. If a protein group has less then two valid values (non NaN), then the p-value results in NaN.

For large datasets, the t-test p-value should be corrected for multiple-testing. TARAM uses p.adjust from the R package stats. In TARAM the resulting adjusted p-value is called *q-value*.

A significance threshold (typically at 0.05) decides whether or not a q-value (or p-value, if no multiple-testing correction has been used) is significant.

TARAM adds the following columns for every comparision, followed by the names of the two groups to compare:

- *p.value*
- *q.value* (only added if multiple-testing correction is computed)
- *log2.fold.change*
- *is.significant*

For example if *Group 1* is compared to *Group 2* columns "*p.value.Group 1-Group 2*", "*q.value.Group 1-Group 2*", etc are added.

An additional filter allows to ignore t-test computation for all protein groups that do not have a minimum number of non-imputed (real) values in one group. This prevents the calculation of spurious values based mostly or only on imputed data. The rows which don't pass the filter are set to NaN.

Volcano plot

Volcano plots are used to visualize and identify statistically significant changes in large data sets when comparing two conditions. The x-axis shows the log2 fold change. The y-axis shows either -log10(p-values) or -log10(q-values). Thresholds for log2 fold change and p or q-value allow to highlight the interesting part of the plot.

Proteins groups of interest can be selected to display their name. By default the gene name (with an added asterisk (*) in case there are multiple gene names) is shown. If no gene name is available, the first protein AC is displayed.

TARAM uses the columns *p.value*, *q.value* and *log2.fold.change* from the result tables for the volcano plots. This plot requires that a t-test was run in a previous step.

Scatter plot

Taram can create a scatter plot between any two numerical columns from the result table. Furthermore it can use a third column to color the protein groups according to it.

A linear regression of the data points can be shown. The regression is computed with Apache Commons SimpleRegression. To give a visual help to assess the similarity between two columns, a *y=x* line can be added.

Remove imputed values

As mentioned in the imputation step, TARAM stores the information of which positions were imputed (column and protein group) in a boolean matrix. Any values which are marked as "imputed" can be later replaced by NaN or 0, restoring the table to its original state before imputation.

1D enrichment

1D enrichment is a statistical method that helps with functional interpretation of the results. The 1D enrichment method is published and was first integrated into Perseus. For this, the list of proteins is first ranked based on a numerical variable (e.g. fold - change). The algorithm then examines the distribution of each annotation term to check if it is enriched toward the top or the bottom of the list. The method essentially uses a two-sided Wilcoxon-Mann-Whitney test, followed by a Benjamin-Hochberg (FDR) multiple test correction. TARAM re-implements the method in Kotlin. The source code can be found in the class OneDEnrichmentComputation.

There are compatible annotation files for proteins from many species available for download from the Max-Planck-Gesellschaft (MPG) web page. They integrate organism specific data from various sources (e.g GO and KEGG annotations). But actually any table could be used as an annotation file. The only prerequisit is a column "UniProt" which contains UniProt protein accession numbers (AC), separated by semicolons.

In TARAM you can select one or several columns (typically fold-changes) and any number of columns from the annotation files (typically GOBP name, GOMF name, GOCC name and KEGG). The resulting 1D enrichment table contains the following columns:

Column name	Description
Column	Selected column from the TARAM result table.
Type	Selected column from the annotation file.
Name	Name of the annotation term.
Size	Number of protein groups with the corresponding annotation term.
Score	The position score reflects the location of the center of the value distribution for protein groups associated with a specific annotation term, relative to the overall distribution of values. The score is defined as: $$s = \frac{2(R1 - R2)}{n}$$ where R1 and R2 are the average ranks within the group under consideration and its complement (all remaining proteins in the experiment), and n is the total number of data points.
P-value	Result from the Wilcoxon-Mann-Whitney test, comparing the values from the group of proteins with the annotation term, with the ones without the term.
Q-value	Adjusted p-values, after multiple testing correction (FDR).
Mean	Mean of values from protein groups with the corresponding annotation term.
Median	Median of values from protein groups with the corresponding annotation term.

The results are filtered by the q-value with a significance threshold (default is 0.02).

Beside the enrichment table, TARAM also adds a column for each selected annotation type to the main result table. Every row lists the annotation terms matching the current protein group, separated by semicolons.