******

User Guide:

Help and documentation

(1.2.12)

# Table of contents

* [Introduction](#_Introduction)
* [License](#_License)
* [Installation](#_Installation)
  + [Download](#_Download)
  + [Available operating systems](#_Available_operating_systems)
  + [Windows distribution](#_Windows_distribution)
  + [MacOS distribution](#_MacOS_distribution)
  + [Linux distribution](#_Linux_distribution)
* [Getting Started](#_Get_Started)
* [Modules](#_Modules)
  + [Relation Tables Module](#_Basic_modules)
    - [RELS CREATOR](#_RELS_CREATOR)
  + [Basic Modules](#_Basic_modules)
    - [LEVEL CREATOR](#_LEVEL_CREATOR)
    - [LEVEL CALIBRATOR](#_LEVEL_CALIBRATOR)
    - [INTEGRATE](#_INTEGRATE)
    - [NORCOMBINE](#_NORCOMBINE)
    - [RATIOS](#_RATIOS)
    - [SBT](#_SBT)
  + [Compound Modules](#_Compound_modules)
    - [WSPP-SBT](#_WSPP-SBT_1)
    - [WSPPG-SBT](#_WSPPG-SBT_1)
    - [WPP-SBT](#_WPP-SBT_1)
    - [WPPG-SBT](#_WPPG-SBT_1)
  + [Report Modules](#_Report_modules)
    - [REPORT](#_REPORT)
    - [SANSON](#_SANSON)
  + [Special Parameters](#_Special_parameters)
    - [Multiple samples](#_Multiple_samples)
    - [Asterisk is our jack of all trades](#_Asterisk_is_our)
    - Multiple samples [Multiple samples in the inputs and outputs](#_Multiple_samples_in)
    - [More params](#_More_params)
    - [Filter (for REPORT and SANSON)](#_Filter_param_(in)
* [Sample Workflows with Application to Case Studies](#_Sample_Workflows_with)
  + [Workflow 1: One-step quantification in a labeled experiment](#_Workflow_1:_One-step)
  + [Workflow 2: Step-by-step quantification and sample combination in a labeled experiment](#_Workflow_2:_Step-by-step)
  + [Workflow 3: Quantification of posttranslationally modified peptides in a labeled experiment](#_Workflow_3:_Quantification)
  + [Workflow 4: Label-free quantification](#_Workflow_4:_Label-free)
* [Importing a workflow template](#_Importing_a_workflow)
* [Creating the identification/quantification file from proteomics pipelines](#_Creating_the_identification/quantif)
  + [Preparing the ID-q file from Proteome Discoverer output](#_Preparing_the_ID-q)
  + [Preparing the ID-q file from MaxQuant output](#_Preparing_the_ID-q_1)
  + [Preparing the ID-q file from FragPipe output](#_Preparing_the_ID-q_2)
* [Adapting the results from proteomics pipelines for iSanXoT](#_Adapting_the_results)
* [References](#_References)

# Introduction

**iSanXoT** is a standalone application for statistical analysis of mass spectrometry-based quantitative proteomics data. iSanXoT builds upon SanXoT [1], our previous publicly available implementation of the weighted spectrum, peptide, and protein (WSPP) statistical model [2] using the Generic Integration Algorithm (GIA) [3].

iSanXoT executes several kind of workflows for quantitative high-throughput proteomics, **systems biology** and the **statistical analysis**, **integration and comparison of experiments**.

iSanXoT was developed by the ***Cardiovascular Proteomics Lab***/***Proteomic Unit*** at **The National Centre for Cardiovascular Research** (CNIC, <https://www.cnic.es>).

## Download

The multiple releases are available in the "release" section, located in the following link:

<https://github.com/CNIC-Proteomics/iSanXoT/releases>

## Installation

### Available operating systems

iSanXoT maintains the following operating systems and architectures and may add additional ones in the future:

Windows 10 Pro (x64)

MacOs High Sierra (10.13.6)

Ubuntu 20.04 (x64)

For more details, read the “Installation” section in the iSanXoT wiki:

<https://github.com/CNIC-Proteomics/iSanXoT/wiki/Installation>

## Getting Started

This chapter describes iSanXoT’s graphical user interface and how to set up an analysis with iSanXoT.

For more details, read the “Getting Started” section in the iSanXoT wiki:

<https://github.com/CNIC-Proteomics/iSanXoT/wiki/Getting_started>

## Modules

The iSanXoT desktop application houses a number of modules based on the SanXoT software package [1]. The information required to setup and execute every module is provided in a task-table.

There are four types of modules:

* Relation tables is a module that creates the relation tables used by the iSanXoT modules.
* The Basic modules call the individual scripts included in the SanXoT software package [1].
* The Compound modules perform a sequence of consecutive integrations based on the weighted spectrum, peptide and protein (WSPP) statistical model [2] and the systems-biology triangle (SBT) algorithm [3].
* Finally, there are two Reports: REPORT generates report files displaying the quantitative results produced by the above Basic and Composite modules when a workflow is executed; SANSON generates a similarity graph showing relationships between functional categories on the basis of the protein elements they share.

For more details, read the “Modules” section in the iSanXoT wiki:

<https://github.com/CNIC-Proteomics/iSanXoT/wiki/Modules>

## Input Adaptor

The iSanXoT Input Adaptor offers users the possibility to either provide their own Identification/Quantification file, which contains the identification and quantification data, or have the Input Adaptor prepare this file from the results obtained using any of the mainstream proteomics pipelines.

For further details, read the “Input Adaptor” section in the iSanXoT wiki:

<https://github.com/CNIC-Proteomics/iSanXoT/wiki/Input_adaptor>

## Sample Workflows

We describe in detail four sample workflows that illustrate the capacity of iSanXoT to statistically ascertain abundance changes in both multiplexed, isotopically labeled [3-5] and label-free [6] proteomics experiments.

* [Workflow 1](https://github.com/CNIC-Proteomics/iSanXoT/wiki/WKF_1): One-step quantification in a labeled experiment.
* [Workflow 2](https://github.com/CNIC-Proteomics/iSanXoT/wiki/WKF_2): Step-by-step quantification and sample combination in a labeled experiment.
* [Workflow 3](https://github.com/CNIC-Proteomics/iSanXoT/wiki/WKF_3): Quantification of posttranslationally modified peptides in a labeled experiment.
* [Workflow 4](https://github.com/CNIC-Proteomics/iSanXoT/wiki/WKF_4): Label-free quantification.

For further details, read each section in the iSanXoT wiki.

## More Documents

User Guide (v1.2.12):

<https://github.com/CNIC-Proteomics/iSanXoT/wiki/docs/user_guides/User_Guide_iSanXoT-1.2.12.pdf>

## License

This application is licensed under a **Creative Commons Attribution-NonCommercial-NoDerivs 4.0 Unported License**.

<https://creativecommons.org/licenses/by-nc-nd/4.0/>

#### You are free to:

Share - copy and redistribute the material in any medium or format.

The licensor cannot revoke these freedoms as long as you follow the license terms.

### Under the following terms

<https://github.com/CNIC-Proteomics/iSanXoT/wiki/License>

# Installation

### Download

The multiple releases are available in the "release" section, located in the following link:

<https://github.com/CNIC-Proteomics/iSanXoT/releases>

### Available operating systems

iSanXoT maintains the following operating systems and architectures and may add additional ones in the future:

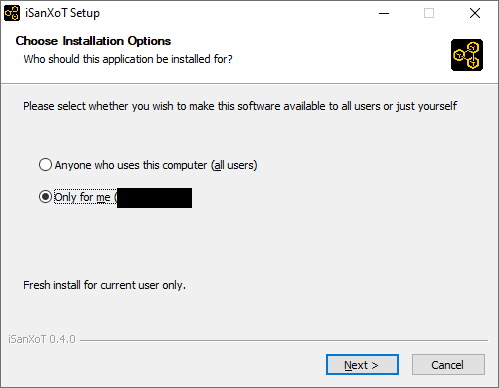
* Windows 10 Pro (x64)
* MacOs High Sierra (10.13.6)
* Ubuntu 20.04 (x64)

### Windows distribution

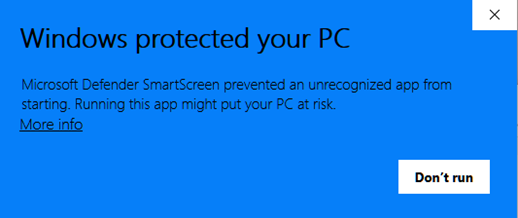
The iSanXoT Windows distribution is packaged in a NSIS Launcher (exe file).

Download the exe Launcher: iSanXoT\_Launcher\_1.X.X.win32-x64.exe

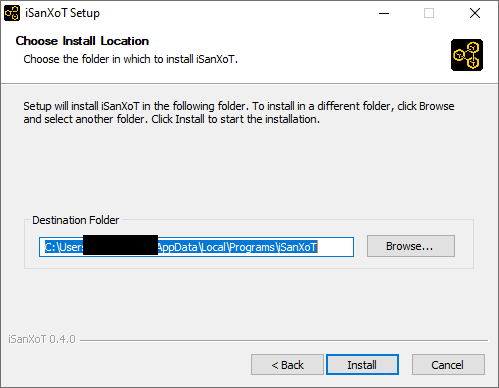
Double-click the Launcher file; the Installer window will show up:



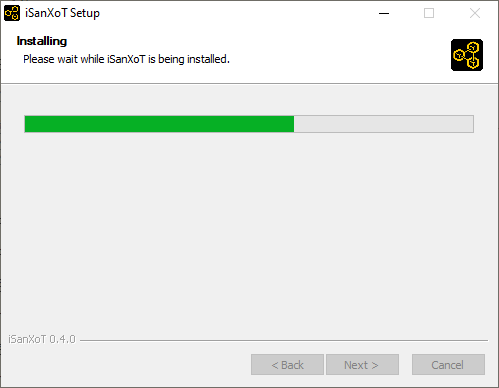
WARNING: Windows Defender SmartScreen might show a prompt suggesting that you cancel the installation; in such case click “More info” and then select the “Run anyway” option.



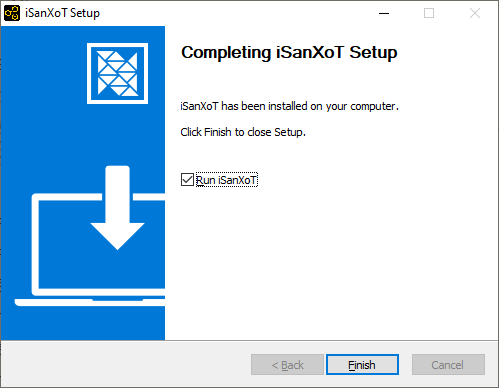
Then, you can choose the iSanXoT installation folder:



Wait while iSanXoT is being installed.



Once the installation has been completed, you are ready to run.

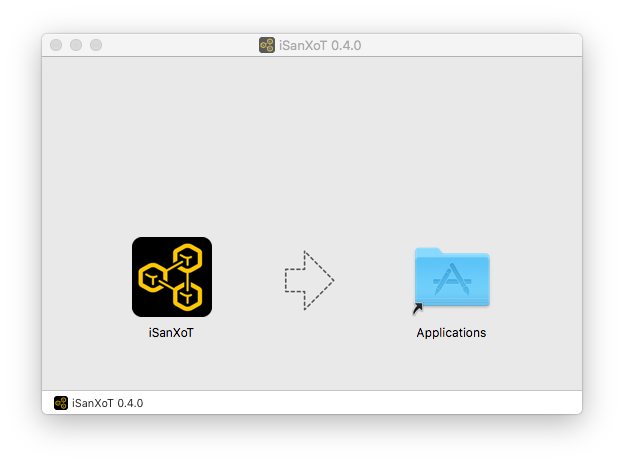


### MacOS distribution

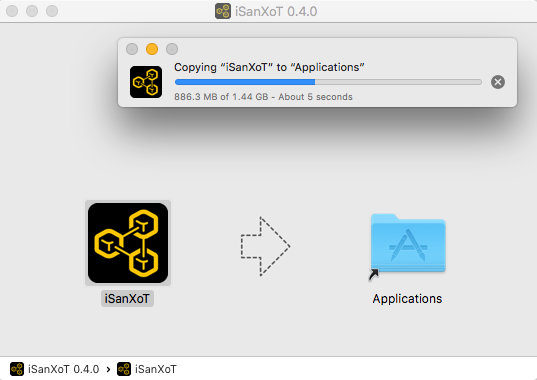
The iSanXoT MacOS distribution is packaged in a DMG container.

Download the DMG file: iSanXoT\_Launcher\_1.X.X.darwin-x64.dmg

Double-click the DMG file, then a Finder window will show up. This window will usually display iSanXoT’s installer icon and a shortcut to the Applications folder, together with some sort of linking arrow:



Simply drag the iSanXoT icon to your Applications folder...

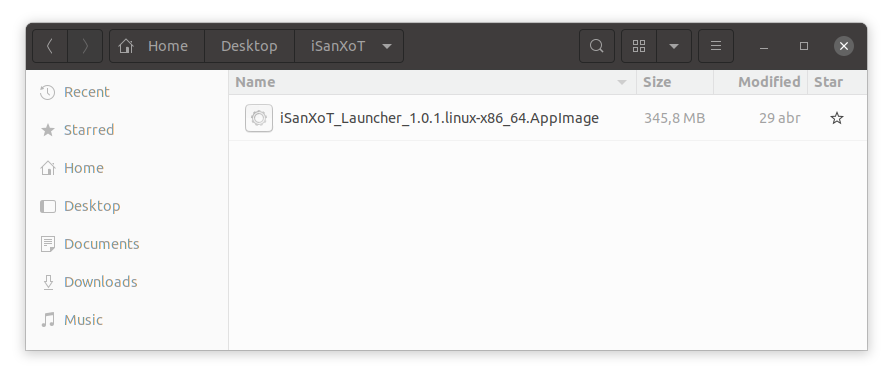


and you're done: the iSanXoT application is now installed.

### Linux distribution

The iSanXoT Linux distribution is packaged in an *AppImage*.

Download the AppImage: iSanXoT\_Launcher\_1.X.X.linux-x86\_64.AppImage



The AppImage file is just the application's compressed image. When executed, the application is mounted in a temporal folder. However, for the correct behaviour, the application has to be extracted to the "squashfs-root" folder in the current working directory using:

./iSanXoT\_Launcher\_1.X.X.linux-x86\_64.AppImage --appimage-extract

Then, launch the iSanXoT application using:

squashfs-root/AppRun

# Getting Started

This chapter describes iSanXoT’s graphical user interface and how to set up an analysis with iSanXoT.

### Opening the iSanXoT application

To open the iSanXoT application:

* In Windows: from the Start menu choose *Programs > iSanXoT*; or double-click the iSanXoT desktop icon.
* In MacOS: double-click the iSanXoT icon from the Applications folder.
* In Linux: from the *AppImage* file.

The contents are extracted to the "squashfs-root" directory in the current working directory using:

./iSanXoT\_Launcher\_1.x.linux-x86\_64.AppImage --appimage-extract

Now you can launch the iSanXoT application:

squashfs-root/AppRun [...]

### Installing required packages

The first time iSanXoT is run a window will show up displaying a progress bar to inform you about the percentage of packages that have been installed. These packages contain the libraries required by iSanXoT’s backend, and are installed the first time you open the application.



**Figure 1.** **Installation window.**

### Closing the iSanXoT application

WARNING: If valid changes were made to your project, make sure to save it before quitting iSanXoT, as any changes will be lost otherwise (you won’t be prompted for saving upon closing).

To close the iSanXoT application:

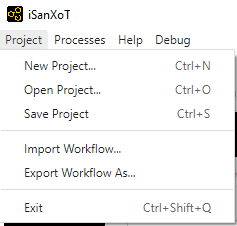
In Windows and Linux: choose *Project > Exit*, or click the X in the upper right corner of the main iSanXoT window.

In Mac: choose *iSanXoT (menu) > Exit*, or click the red X in the upper left corner of the main iSanXoT window.

A dialog window will show up asking you to confirm the application closing. Click “Yes” if you really want to quit iSanXoT.

### iSanXoT Projects

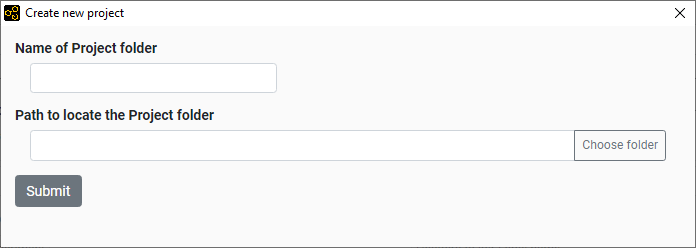
An iSanXoT project is primarily a container used to structure the data coming from your input file(s) and your workflow. The input file contains the identification and quantification data (for further details see the [Input Adaptor](#_Input_Adaptor) Section). These fully-customisable workflows can perform quantitative proteomics analysis, systems biology analysis, and comparison and merging of experimental data from technical or biological replicates.



**Figure 2. Project menu.**

#### Creating a new project

Creating a project is the first step when conducting an analysis with iSanXoT. By selecting *Project > New Project* a window will show up where you can provide a name for the project as well as select a project folder where iSanXoT output files will be stored.



**Figure 3. Window that creates a new project.**

#### Opening a project

By selecting *Project > Open Project* as a folder selection dialog box shows up that allows the user to indicate the location of an already existing project folder to be opened by iSanXoT.

### iSanXoT Main Window

The iSanXoT main window consists of an overhead menu, adaptor and module tabs, and content and execution panels (**Figure 4**).

#### Menu

The Menu contains the following items:

* Project: Allows operations related to projects and workflows.
* Processes: Links to real-time display of the processes currently in execution by iSanXoT (see [Running Processes](#_Running_Processes) Section).
* Help: Houses the different Sections of iSanXoT help.

#### Content tabs

1. Five tabs are displayed in iSanXoT’s project page. The Input tab displays the Project folder, where iSanXoT output files are stored, as well as the Identification file used in the project (see [Input Adaptor](#_Adaptors) Section). The remaining four tabs give access to iSanXoT modules: [Relation Tables](#_Relation_tables_module), [Basic Modules](#_Basic_modules_1), [Compound Modules](#_Compound_modules), and [Report Modules](#_Reports_modules).



**Figure 4. Main View of iSanXoT.**

#### Content panel

This panel houses the elements of the Input and Modules tabs, which can be accessed through the sidebar menu showing on the left side of the panel. A title and a brief description of the Input/Module element is provided, as well as help icon linking to additional information on the specific Input element or Module selected.

#### Execution panel

The execution panel, located on the bottom of the main window, allows the user to indicate the number of processors to be used by iSanXoT, 4 by default. The Start button launches the execution of the workflow shaped by the Input elements and Modules.

## Importing and Exporting Workflows

A project is shaped by a workflow that instructs iSanXoT how to process the data provided by the input file(s). While the whole project, including workflow and data, can be saved as indicated below (see the [Saving a project](#_Saving_a_project) section), there is a way to import and export just the workflow structure using iSanXoT menu (**Figure 2**).

#### Import Workflow

This option allows to import the task-tables of a workflow. For that, you have to provide the folder where the workflow is saved.

#### Export Workflow

The export workflow saves the task-tables of a workflow in the folder indicated by the user.

## Executing a Project

Once your project contains all the necessary input data and workflow elements, you can execute the workflow by clicking “Save and Run” in the “Execution panel” after indicating the number of processors to be allocated for iSanXoT (see [Execution panel](#_Execution_panel) above).

Bear in mind that every time you click the “Save and Run” button to execute a workflow, the project is first validated for consistency and saved. To save a project without executing it, you must use the appropriate menu item as explained in the next Section.

#### Saving a project

The *Project > Save Project* option saves your project, which contains the input data and the workflow elements, to the “Project folder”. The project files are saved in the “.isanxot” folder. WARNING: Do not manipulate or delete the information stored in the “.isanxot” folder; you risk losing your project.

Whenever iSanXoT is prompted to save a project, the corresponding workflow is first validated for consistency and won’t be saved when failed. Neither will the workflow execute if it has not been validated previously.

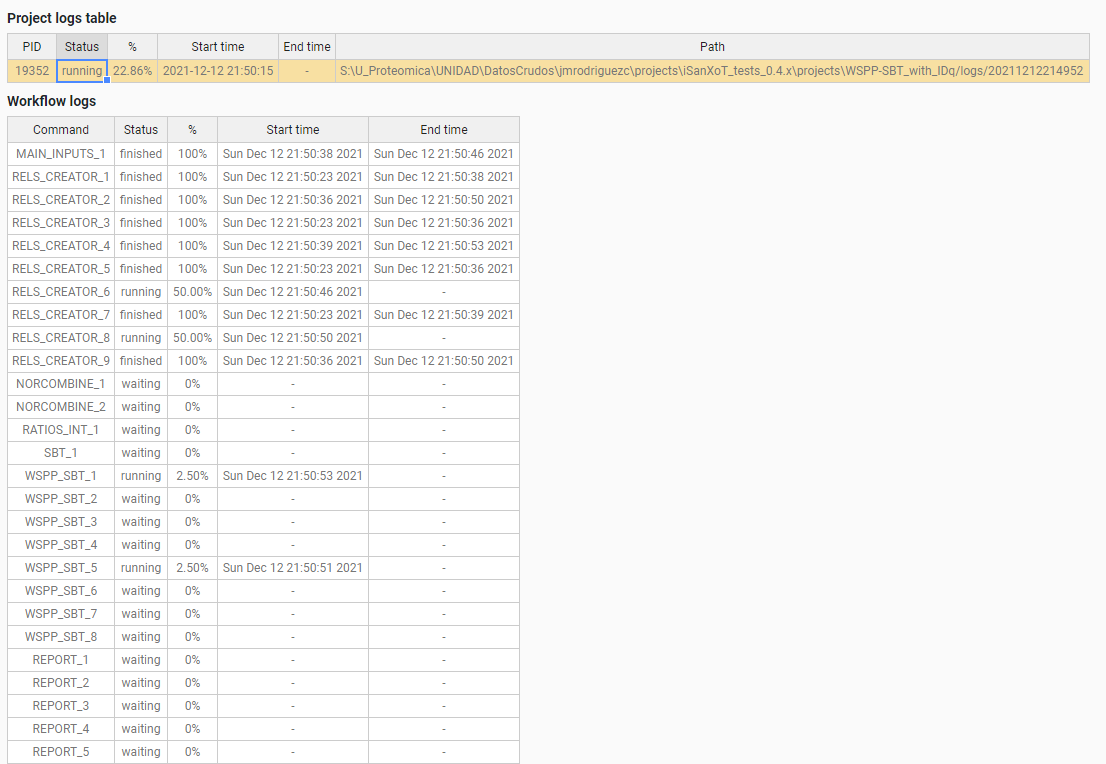
As well as the “.isanxot” folder, the following folders are necessary to shape your project:

* Exps, to store the files created by the Input Data adapters.
* Jobs, to store the *sample folders* of your workflow.
* Rels, to store the Relation Tables created by the RELS CREATOR module (see below).
* Reports, to store the Report files created by the REPORT module (see below).
* Stats, to store statistical data.
* Logs, to store the workflow execution log files.

## Running Processes

When workflow execution successfully starts, a new window shows up displaying information about the processes currently running:

* The Project logs table shows project execution status. Several project executions can be monitored here, and the user must click a row to have the corresponding workflow logs displayed (see below).
* Workflow logs table displays status for the jobs set up in the workflow modules. If you click on a row you will see the trace log of the involved jobs (unless the job status is “cached”).



**Figure 5. View of running processes.**

The running processes window can be also reached from the menu by selecting the “Processes > Main page” option.

The “Processes” menu includes a submenu, “Open Processed Project”, which opens the processed project that has been selected in the “Project Logs” table (*Figure 6*).



Figure 6. The “Open Processed Project” submenu in the “Processes” menu allows the opening of a selected project that has been executed or is currently executing.

This submenu is accessible exclusively in the “running processes” view and is activated when a row is selected in the “Project Logs” table (refer to *Figure 7*). As iSanXoT facilitates the execution of various types of workflows, the “Project Logs” table may include different project and workflow types. Consequently, users can reopen their chosen processed projects using this method.



Figure 7. Reopening a previously executed project.

# Modules

The iSanXoT desktop application houses a number of modules based on the SanXoT software package [1]. The information required to setup and execute every module is provided in a task-table.

There are four types of modules:

* Relation tables is a module that creates the relation tables used by the iSanXoT modules.
* The Basic modules call the individual scripts included in the SanXoT software package [1].
* The Compound modules perform a sequence of consecutive integrations based on the weighted spectrum, peptide and protein (WSPP) statistical model [2] and the systems-biology triangle (SBT) algorithm [3].
* Finally, there are two Reports: REPORT generates report files displaying the quantitative results produced by the above Basic and Composite modules when a workflow is executed; SANSON generates a similarity graph showing relationships between functional categories on the basis of the protein elements they share.

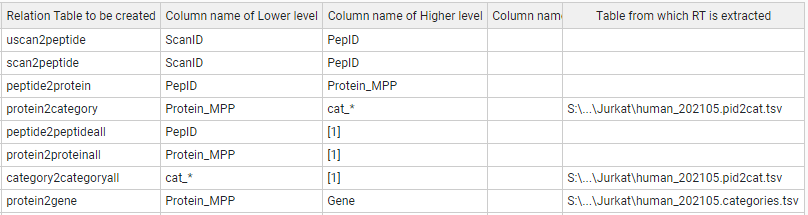
## Relation tables module

### RELS CREATOR

This module generates relation tables (RT) from tab-separated values (TSV) files. Relation tables, which are TSV files relating lower level identifiers (e.g. peptides) to the corresponding higher level elements (e.g. proteins), are required for module execution. For this reason, the naming convention for the file indicated under “Relation Table to be created” in the RELS CREATOR task table is *lower level + “2” + higher level*; e.g. whether an integration is from the “peptide” level to the “protein” level, a relation table called “peptide2protein” will be necessary.

The fields showing in the RELS CREATOR task table are (**Figure 8**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Relation Table to be created* specifies relation table filenames. As commented above, the naming convention for these files is *lower level + “2” + higher level*; e.g. whether an integration is from the “peptide” level to the “protein” level, a relation table called “peptide2protein” will be necessary.
* *Column name of Lower level* is the column header that designates which elements from the indicated file (see below) will be taken as lower level elements in the resulting relation table.
* *Column name of Higher level* is the column header that designates which elements from the indicated file (see below) will be taken as higher level elements in the resulting relation table.
* *Column name of 3rd column* is the column header that designates which elements from the indicated file (see below) will be taken as third column elements in the resulting relation table.
* Table from which RT is extracted is the full path name for the TSV file to be used to build the relation tables. If the cell is empty, the Input file (ID-q.tsv) is applied for iSanXoT workflow.



**Figure 8. A sample Task-Table in the example of RELS CREATOR module.**

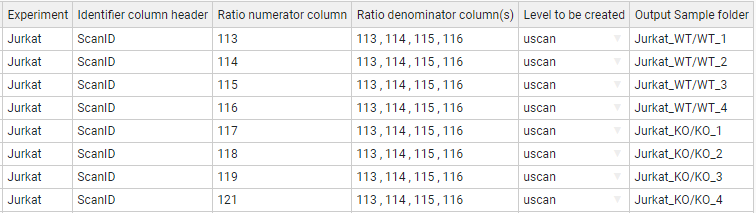
## Basic modules

### LEVEL CREATOR

This module creates levels, which are TSV files containing identifiers, log2-ratio values and statistical weight values. The data are extracted from the Identification/Quantification file (see [Input Adaptor](#_Adaptors) section).

The following fields are displayed in the LEVEL CREATOR task table (**Figure 9**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Experiment* is the column header that designates which elements from the Identification file will be used to create the level indicated.
* *Identifier column header* is the identification file column header that unambiguously identifies the scans.
* *Ratio numerator column* specifies which column header from the identification file designates the quantitative values to be used as a numerator for the log2-ratio calculation.
* *Ratio denominator column(s)* specifies which column header from the identification file designates the quantitative values to be used as a denominator for the log2-ratio calculation.
* *Level to be created* designates the level name.
* *Output Sample folder* indicates the name of the folder where the level data file will be saved.



**Figure 9. A sample task-table in the LEVEL CREATOR module.**

### LEVEL CALIBRATOR

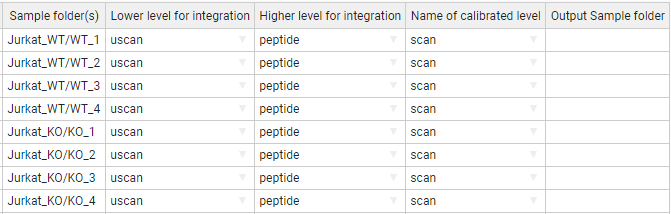
This module calibrates the above-described levels using the “Klibrate” program included in the SanXoT software package [1]. To perform the calibration, two parameters (weight constant and variance) are iteratively calculated using the Levenberg-Marquardt algorithm (for more details see the information about “Klibrate” in the SanXoT software package [1]).

The output calibrated level contains new statistic weight values for the identifier and log2-ratio elements displayed in the uncalibrated level data file. This is necessary for the levels to be used as inputs to the INTEGRATE module.

#### Standard parameters

The fields to be completed in this module are (**Figure 10**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Sample folder(s)* indicates the name(s) of the folder(s) containing the uncalibrated data file(s) that were previously generated by the LEVEL CREATOR module.
* *Lower level for integration* indicates which lower level elements are to be used in the integration carried out for the calibration.
* *Higher level for integration* indicates which higher level elements are to be used in the integration carried out for the calibration.
* *Name of calibrated level* is the name for the output data file containing the new, calibrated statistical weight values.
* *Output Sample folder* specifies the name of the folder where the output data file containing the new, calibrated statistical weight values will be saved. If the cell is empty, the output sample folder is the given “Sample folder” (second column).



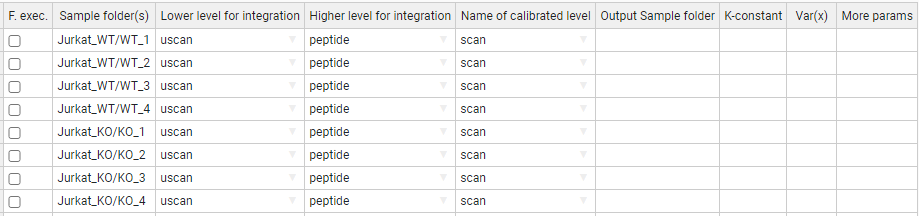
**Figure 10. A sample task table in the LEVEL CALIBRATOR module.**

#### Advanced parameters

The LEVEL\_CALIBRATOR module accepts the following additional parameters (**Figure 11**):

To perform the calibration two parameters, have to be calculated: the k (weight constant), and the variance.

* *K-constant* sets a forced value for the k-constant. Using this parameter, the introduced value is forced as K-constant.
* *Var(x)* sets a forced value for the variance. Using this parameter, the introduced value is forced as the variance.
* *More params* allows adding more parameters to the internal programs of the module. For more details see [More params](#_More_params) in the “Special Parameters” Section.

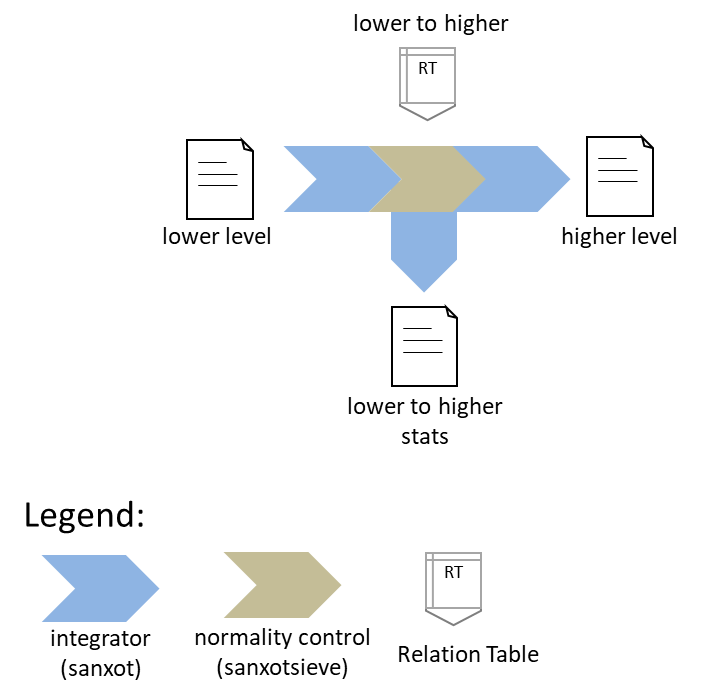


**Figure 11. A task-table displaying advanced parameters for the LEVEL CALIBRATOR module.**

### INTEGRATE

The INTEGRATE module performs statistical calculations based on the WSPP model by iteratively applying the generic integration algorithm (GIA) [3] on calibrated data files (**Figure 12**).

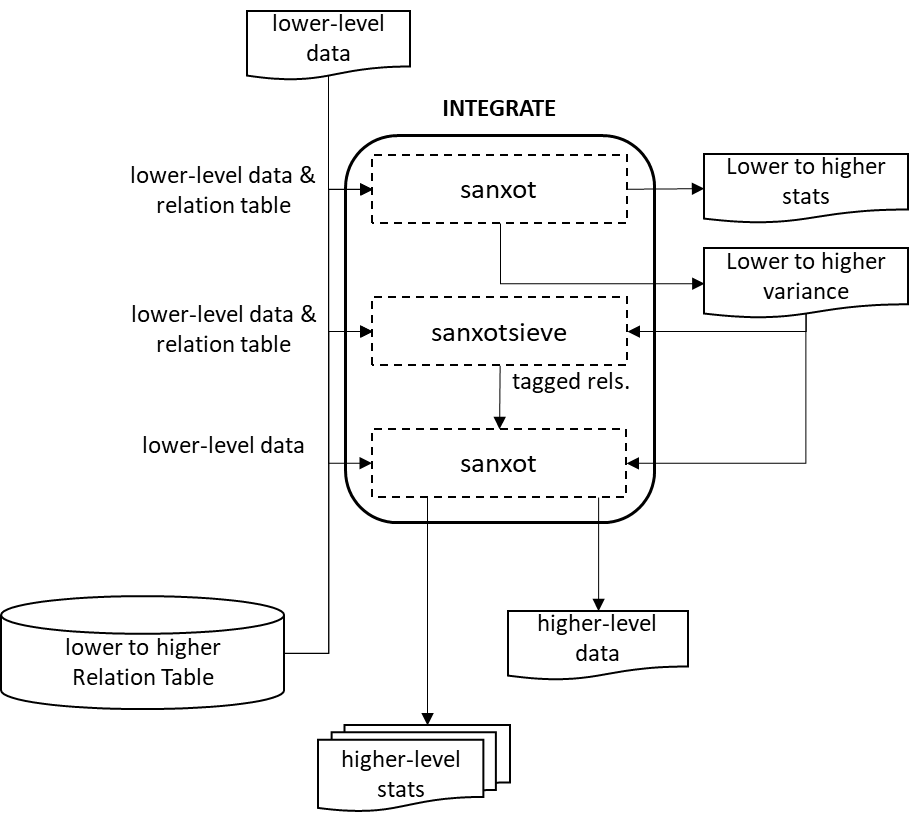
Integrations are carried out from lower level data to higher level data (e.g. from the peptide level to the protein level and from the protein level to the gene level).



**Figure 12. Schematic representation of the INTEGRATE module.** The integration is carried out from any lower level to any higher level using the programs “SanXoT” and “SanXoTSieve” and the generic integration algorithm (GIA).

More in detail, the INTEGRATE module needs two TSV files as inputs:

1. A data file containing three data columns: identifier (a text string that is used to unambiguously identify the low level elements), quantitative value (log2-ratio of the two measurements to be compared) and statistical weight (a parameter that measures the accuracy of the quantitative value).
2. A relation table, which links the lower level identifiers to those in the higher level. This file contains two columns: higher level identifiers on the left and lower level identifiers on the right.



**Figure 13. The INTEGRATE module flowchart.** A first integration is done with “SanXoT” that calculates the variance; then “SanXoTSieve” removes outliers tagging them in a new relation table; finally, a second integration is done with “SanXoT” using the variance calculated.

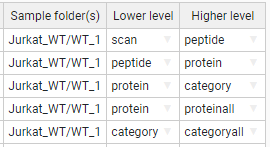
For every integration, the SanXoT program calculates the general variance using a robust iterative method. Then SanXoTSieve is used to tag outlier elements [2] by assessing the probability that a lower level element be a significant outlier of the standardized (i.e. N(0,1)) log2-ratio distribution. The most extreme outliers are thus removed sequentially and the integration repeated until all outliers below a user-defined false discovery rate (FDR) threshold have been removed. Finally, a second integration is carried out by SanXoT using the variance calculated in the first integration and discarding the outliers tagged in the new relation table (**Figure 13**).

The output data files generated by INTEGRATE contain the quantitative data for the higher level and can be used as inputs to other modules; in addition, each integration generates several additional files which contain information about the integration. For further details see SanXoT documentation [1].

#### Standard parameters

The parameters to be provided in the INTEGRATE module task table are (**Figure 14**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Sample folder(s)* indicates the names of the folder(s) where the lower level data file is located.
* *Lower level* indicates the name of the lower level data file to be used. This file contains three data columns: identifier, quantitative value and statistical weight.
* *Higher level* indicates the name of the higher level to which the lower level elements will be integrated.



**Figure 14. A sample task-table in the INTEGRATE module.**

#### Advanced parameters

The INTEGRATE module accepts the following additional parameters (**Figure 15**):

* *Output Sample folder* is the name of the folder where the level data and statistics are saved.
* *Tag* is a text label that indicates which elements from the lower level are integrated into the higher level. The tags must be specified in the third column of the corresponding Relation Table. This allows the user to discard elements for integration without needing to eliminate them from the Relation Table. Thus, if the label “marked” is used as a Tag, only the lower level elements containing the label marked in the third column of the lower\_level2higher\_level Relation Table will be integrated. Logical operators can also be used in the Tag field to make complex decisions.

*Tag* is a parameter to distinguish groups to perform the integration. For instance, if the user specifies “marked” the elements containing the label “marked” in the third column of the “lower\_level2higher\_level” Relation Table will be included in (or discarded from) the integration.

The tag can be used by inclusion, such as "mod" or by exclusion, putting first the "!" symbol, such as "!mod". Tags should be included in a third column of the relations file.

Different tags can be combined using logical operators "and" (&), "or" (|), and "not" (!), and parentheses. Some examples:

!out&mod

!out&(dig0|dig1)

(!dig0&!dig1)|mod1

mod1|mod2|mod3

***Warning***: Unless specified otherwise by the user, by default iSanXoT eliminates outliers from the lower level according to an FDR<1% threshold.

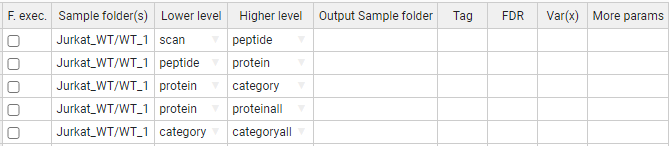
iSanXoT automatically adds the tag “out” in the third column of the relation table to label outliers, so that they are not integrated. It is not thus recommended to use this tag for other purposes.

***Note*** that although the discarded elements will not be included in calculations, the parameter Z will be calculated and tabulated in the corresponding output (outStats) file.

For further details see SanXoT wiki

(<https://www.cnic.es/wiki/proteomica/index.php/SanXoT_software_package>).

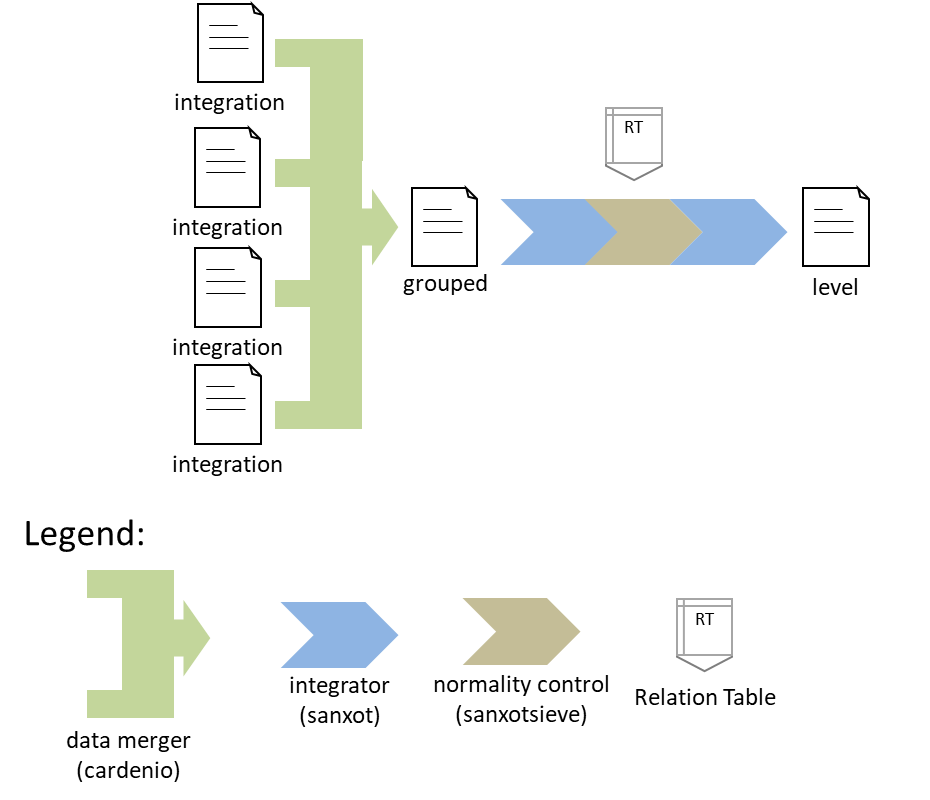
* *FDR* is an FDR threshold other than the default value (0.01, i.e. 1%) for outlier removal. If “0” is specified as the FDR value, then no outliers will be discarded.
* *Var(x)* sets a fixed value for the variance. The default value (blank) means that the variance will be iteratively calculated based on the Levenberg-Marquardt algorithm in the first *Lower level-to-Higher level* integration.
* *More params* allows adding more parameters to the internal programs of the module. For more details see [More params](#_More_params) in the “Special Parameters” Section.



**Figure 15. A sample task-table displaying advanced parameters for the INTEGRATE module.**

### NORCOMBINE

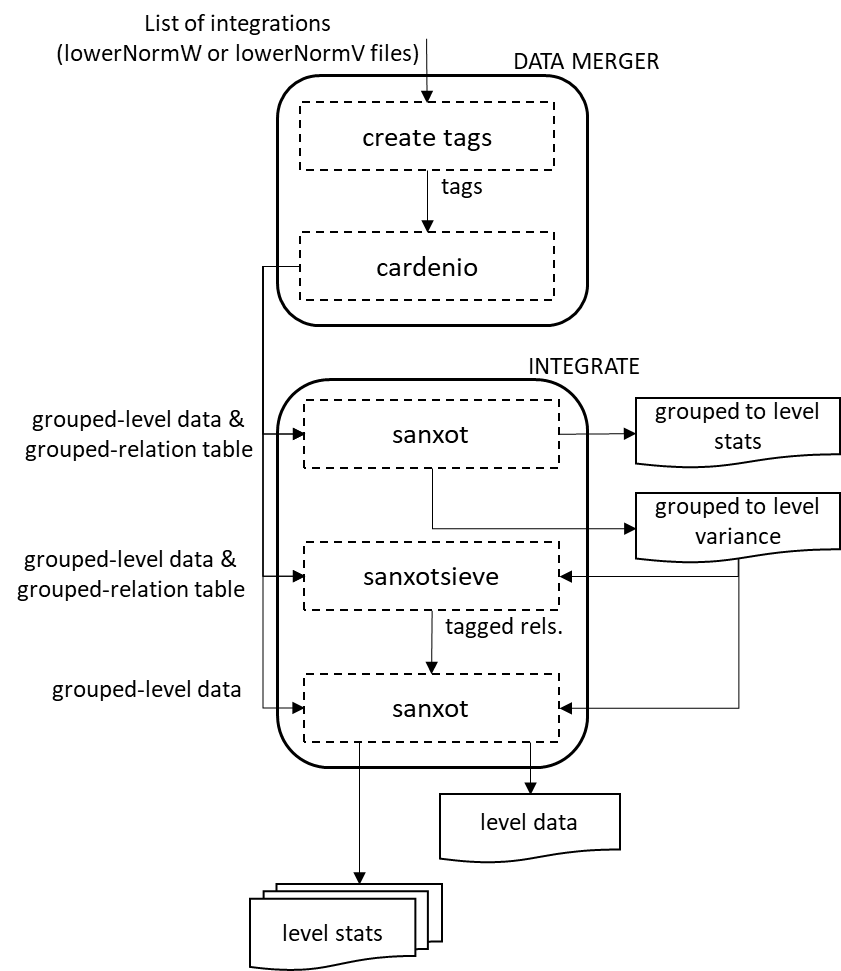
The NORCOMBINE module combines technical or biological replicates (**Figure 16**). For example, NORCOMBINE can be used to merge the protein level data from 4 individual patients and 4 individual controls into a patient- and a control level protein dataset, respectively.



**Figure 16. Schematic representation of the NORCOMBINE module used to combine technical or biological replicates.**

Experiment merging relies on the “Cardenio” program from the SanXoT software package [1], which is used to generate merged data files and relation tables that are later integrated to the grouped level using “SanXoT” and “SanXoTSieve” (**Figure 16**).

NORCOMBINE requires the user to specify which *lowerNorm* files contain the necessary data for the samples to be combined. These *lowerNorm* files, previously generated by the INTEGRATE module, display the lower level identifiers on the left, followed by the corresponding centred log2-ratio values (i.e. the values obtained after subtracting the high level value) in the second column, and either the integration statistical weight (in the case of *lowerNormV*) or the variance (for *lowerNormW).* The SanXoT program “Cardenio” [1] is then used to generate merged data files and relation tables that are later integrated to the grouped level using “SanXoT” and “SanXoTSieve” (**Figure 17**).

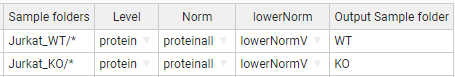


**Figure 17. The NORCOMBINE module flowchart.**

#### Standard parameters

The default NORCOMBINE module task table shows the following fields (**Figure 18**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Sample folders* indicates the names of the folder(s) containing the lower level data (samples) to be combined.
* *Level* indicates the type of elements to be combined (e.g. peptides or proteins).
* *Norm* specifies the normalization scheme to be used in the integrations.
* *lowerNorm* specifies the type of *lowerNorm* file (see above) to be used.
* *Output Sample folder* is the name of the folder where the grouped level data and statistics are saved.



**Figure 18. A sample task-table in the NORCOMBINE module.** In this case, the asterisk wildcard has been used to select multiple sample folders.

#### Advanced parameters

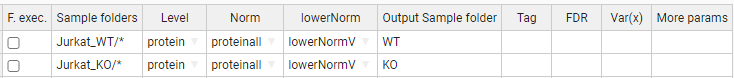
The NORCOMBINE module accepts the following additional parameters (**Figure 19**):

* *Tag* is a text label that indicates which elements from the lower level are integrated into the higher level. The tags must be specified in the third column of the corresponding Relation Table. This allows the user to discard elements for integration without needing to eliminate them from the Relation Table. Thus, if the label “marked” is used as a Tag, only the lower level elements containing the label marked in the third column of the lower\_level2higher\_level Relation table will be integrated. Logical operators can also be used in the Tag field to make complex decisions.

By default, iSanXoT eliminates outliers from the lower level according to an FDR<1% threshold.

For further details, read the Advanced Parameters for the [INTEGRATE](#_Advanced_parameters) module.

* *FDR* is an FDR threshold other than the default value (0.01, i.e. 1%) for outlier removal. If “0” is specified as the FDR value, then no outliers will be discarded.
* Var(x) sets a fixed value for the variance. The default value (blank) means that the variance will be iteratively calculated based on the Levenberg-Marquardt algorithm.
* *More params* allows adding more parameters to the internal programs of the module. For more details see [More params](#_More_params) in the “Special Parameters” Section.

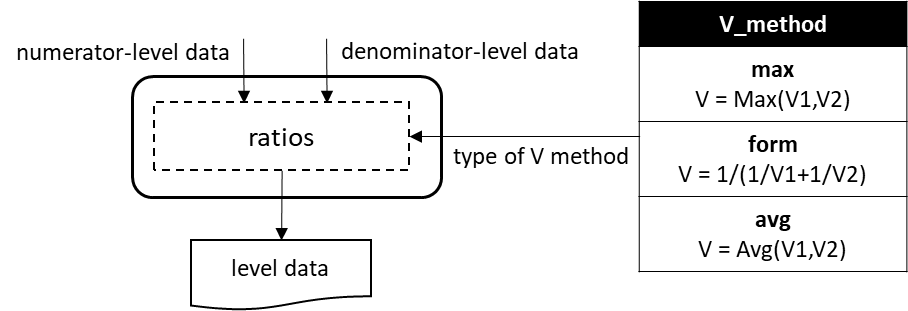


**Figure 19. A sample task table with advanced parameters in the NORCOMBINE module.**

### RATIOS

This module prepares the data file and relation table required as a first step in the calculation of a ratio defined by the user (e.g KO vs WT). For that, the new log2-ratio is calculated as the difference between numerator and denominator values, whereas the corresponding statistical weight is assessed according to the method indicated by the user in the V Method filed of the RATIOS task table (**Figure 20** and **Figure 21**):

* *max* uses the maximum value between the numerator and denominator statistical weight value.
* *form* uses the value resulting from 1/(1/Vn + 1/Vd), where Vn and Vd are the statistical weight value for the numerator and the denominator, respectively.
* *avg* uses the average value between the numerator and denominator statistical weight value.



**Figure 20. The RATIOS module flowchart.**

The RATIOS module task table displays the following parameter fields (**Figure 21**):

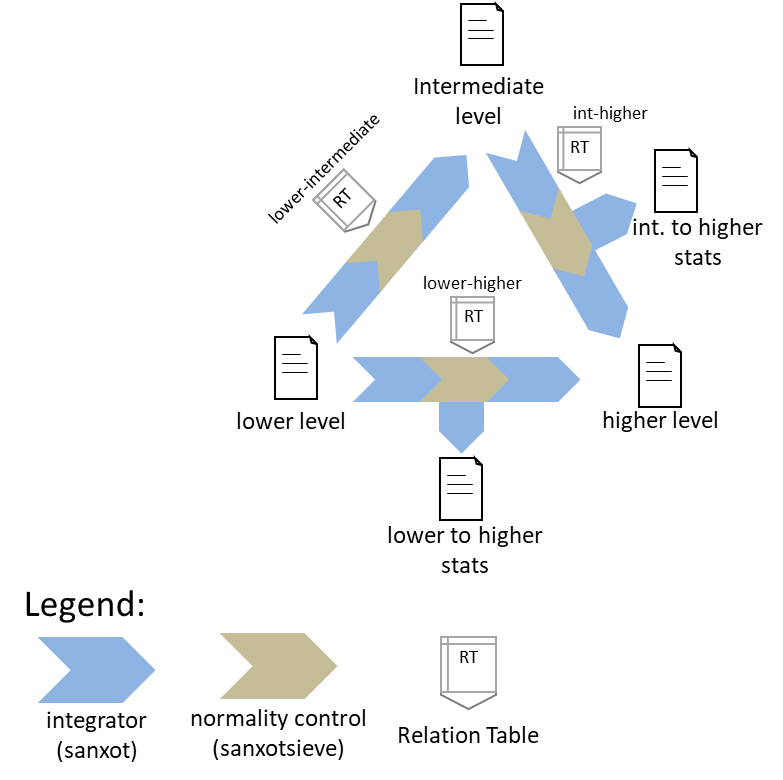
* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Ratio numerator column* specifies the name of the folder containing the quantitative value to be used as a numerator for the new log2-ratio calculation.
* *Ratio denominator column(s)* specifies the name of the folder(s) containing the quantitative values to be used as a denominator for the new log2-ratio calculation. The sample folders have to be separated by comma.
* *Level* designates the level (i.e. peptide, protein, gene or category) at which the ratio is to be calculated.
* *Output Sample folder* indicates the name of the folder where the resulting log2-ratio and statistical weight values will be saved (e.g. KO\_vs\_WT).



**Figure 21. A sample task-table in the RATIOS module.**

### SBT

This module is based on the Systems Biology Triangle (SBT) algorithm [3], which performs an integration between the lower and the higher levels using the variance previously obtained in an integration between the lower and an intermediate level (**Figure 22**). Usually the SBT module is applied to carry out the protein-to-grand mean integration using the variance associated with the protein-to-category integration.



**Figure 22.** **Schematic representation of the SBT module.**

#### Standard parameters

The standard parameters to be entered in the SBT module task table are (**Figure 23**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Sample folder(s)* indicates the name of the folder(s) where the lower level data are located.
* *Lower level* indicates the name of the lower level (e.g. protein).
* *Intermediate level* indicates the name of the intermediate level (e.g. category).

By default, the higher level is the grand mean of the lower level elements.



**Figure 23. A sample task table in the SBT module.**

#### Advanced parameters

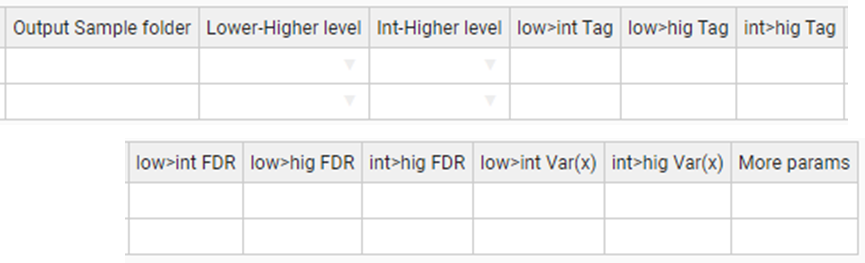
This module accepts the following additional parameters (**Figure 24**):

* *Output Sample folder* indicates an alternative folder to store the resulting log2-ratio and statistical weight values other than “Sample folder(s)”.
* *Lower-Higher level* and *Int(ermediate)-Higher level* specify an alternative higher level other than the grand mean of the lower level elements.
* *Low(er)-to-Int(ermediate)* Tag and *Int(ermediate)-to-Hig(her)* are the text label that indicates which elements from the lower level are integrated into the intermediate level, and the intermediate level are integrated into higher level. The tags must be specified in the third column of the corresponding Relation Table. This allows the user to discard elements for integration without needing to eliminate them from the Relation Table. Thus, if the label “marked” is used as a Tag, only the lower level elements containing the label marked in the third column of the lower\_level2intermediate\_level and intermediate\_level2higher\_level Relation Tables will be integrated. Logical operators can also be used in the Tag field to make complex decisions.

By default, iSanXoT eliminates outliers from the lower level according to an FDR<1% threshold.

For further details, read the Advanced Parameters for the [INTEGRATE](#_Advanced_parameters) module.

* *Low(er)-to-Int(ermediate) FDR* and *Int(ermediate)-to-Hig(her) FDR* determine an FDR threshold other than the default value (0.01, i.e. 1%) for outlier removal in the lower level-to-intermediate level and intermediate level-to-higher level integration, respectively. If “0” is specified as the FDR value, then no outliers will be discarded.
* *Low(er)-to-Int(ermediate) Var(x)* and *Int(ermediate)-to-Hig(her) Var(x)* indicate the variance to be used in the lower level-to-intermediate level and intermediate level-to-higher level integration, respectively, as an alternative to the variance calculated in the lower level-to-intermediate level integration.
* *More params* allows adding more parameters to the internal programs of the module. For more details see [More params](#_More_params) in the “Special Parameters” Section.



**Figure 24. A sample task table with advanced parameters in the SBT module.**

## Compound modules

The Compound modules perform a sequence of consecutive integrations based on the WSPP statistical model [2] and the SBT algorithm [3]. In addition, each module creates the initial level and calibrate this initial level. The WSPP-SBT and WSPPG-SBT modules create and calibrate the “scan” level; and the WPP-SBT and WPPG-SBT create and calibrate the “peptide” level.

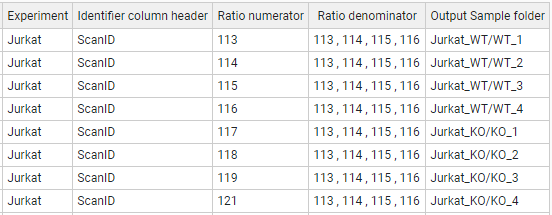
### WSPP-SBT

The WSPP-SBT module performs the following integrations: scan-to-peptide, peptide-to-protein, protein-to-category, protein-to-proteinall, and category-to-categoryall. In addition, the SBT algorithm is used to calculate the variance associated to the protein-to-category integration, which is applied to the protein-to-proteinall integration.

#### Standard parameters

The standard parameters required by the WSPP-SBT module are (**Figure 25**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Experiment* is the column header that designates which elements from the Identification file will be used in the starting scan-to-peptide integration.
* *Identifier column header* is the identification file column header that unambiguously identifies the scans.
* *Ratio numerator* specifies which identification file column header designates the quantitative values to be used as a numerator for the log2-ratio calculation.
* *Ratio denominator* specifies which identification file column header designates the quantitative values to be used as a denominator for the log2-ratio calculation.
* Output Sample folder: indicates the name of the folder where the resulting data files will be saved.



**Figure 25. A sample task table in the WSPP-SBT, WSPPG-SBT, WPP-SBT, and WPPG-SBT modules.**

#### Advanced parameters

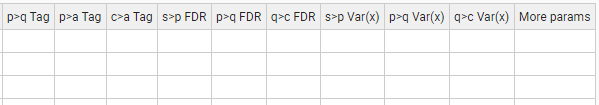
The WSPP-SBT module accepts the following advanced parameters (**Figure 26**):

* *p>q Tag*, *p>a Tag*, *c>a Tag* are the text label that indicates which elements from the lower level (“p” and “c”) are integrated into the higher level (“q” and “a”). The tags must be specified in the third column of the corresponding Relation Table. This allows the user to discard elements for integration without needing to eliminate them from the Relation Table. Thus, if the label “marked” is used as a Tag, only the lower level elements containing the label marked in the third column of the lower\_level2higher\_level Relation table will be integrated. Logical operators can also be used in the Tag field to make complex decisions.

By default, iSanXoT eliminates outliers from the lower level according to an FDR<1% threshold.

For further details, read the Advanced Parameters for the [INTEGRATE](#_Advanced_parameters) module.

* *s>p FDR*, *p>q FDR*, *q>c FDR* establish an FDR threshold other than 0.01 (1%) for outlier removal in the integrations scan-to-peptide, peptide-to-protein, and protein-to-category, respectively. If FDR = 0 is selected, then the outliers are not discarded.
* *s>p Var(x)*, *p>q Var(x)*, *q>c Var(x)* set a fixed value for the variance in the integrations scan-to-peptide, peptide-to-protein, and protein-to-category, respectively. By default, the variance will be iteratively calculated based on the Levenberg-Marquardt algorithm in the first lower level-to-higher level integration (**Figure 12**).
* *More params* allows adding more parameters to the internal programs of the module. For more details see [More params](#_More_params) in the “Special Parameters” Section.



**Figure 26. Task-Table with advanced parameters in the WSPP-SBT module.**

### WSPPG-SBT

The WSPPG-SBT module performs the following integrations: scan-to-peptide, peptide-to-protein, protein-to-gene, gene-to-category, gene-to-geneall, and category-to-categoryall. In addition, the SBT algorithm is used to calculate the variance associated to the gen-to-category integration, which is applied to the gen-to-geneall integration.

#### Standard parameters

The standard parameters required by the WSPPG-SBT module are (**Figure 25**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Experiment* is the column header that designates which elements from the Identification file will be used in the starting scan-to-peptide integration.
* *Identifier column header* is the identification file column header that unambiguously identifies the scans.
* *Ratio numerator* specifies which identification file column header designates the quantitative values to be used as a numerator for the log2-ratio calculation.
* *Ratio denominator* specifies which identification file column header designates the quantitative values to be used as a denominator for the log2-ratio calculation.
* Output Sample folder: indicates the name of the folder where the resulting data files will be saved.

#### Advanced parameters

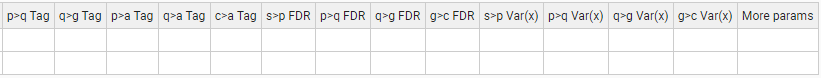
The WSPPG-SBT module accepts the following advanced parameters (**Figure 27**):

* *p>q Tag*, *q>g Tag*, *p>a Tag*, *c>a Tag* are the text label that indicates which elements from the lower level (“p”, “q” and “c”) are integrated into the higher level (“q”, “g” and “a”). The tags must be specified in the third column of the corresponding Relation Table. This allows the user to discard elements for integration without needing to eliminate them from the Relation Table. Thus, if the label “marked” is used as a Tag, only the lower level elements containing the label marked in the third column of the lower\_level2higher\_level Relation table will be integrated. Logical operators can also be used in the Tag field to make complex decisions.

By default, iSanXoT eliminates outliers from the lower level according to an FDR<1% threshold.

For further details, read the Advanced Parameters for the [INTEGRATE](#_Advanced_parameters) module.

* *s>p FDR*, *p>q FDR*, *q>g FDR*, *g>c FDR* establish an FDR threshold other than 0.01 (1%) for outlier removal in the following integrations: scan-to-peptide, peptide-to-protein, protein-to-gene, and gene-to-category, respectively. If FDR = 0 is selected, then the outliers are not discarded.
* *s>p Var(x)*, *p>q Var(x)*, *q>g Var(x)*, *g>c Var(x)* set a fixed value for the variance in the integrations: scan-to-peptide, peptide-to-protein, protein-to-gene, and gene-to-category, respectively. By default, the variance will be iteratively calculated based on the Levenberg-Marquardt algorithm in the first lower level-to-higher level integration (**Figure 12**).
* *More params* allows adding more parameters to the internal programs of the module. For more details see [More params](#_More_params) in the “Special Parameters” Section.



**Figure 27. Task-table with advanced parameters in the WSPPG-SBT module.**

### WPP-SBT

The WPP-SBT module performs the integrations peptide-to-protein, protein-to-category, protein-to-proteinall and category-to-categoryall. In addition, the SBT algorithm is used to calculate the variance associated to the protein-to-category integration, which is applied to the protein-to-proteinall integration.

#### Standard parameters

The standard parameters required by the WPP-SBT module are (**Figure 25**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Experiment* is the column header that designates which elements from the Identification file will be used in the starting scan-to-peptide integration.
* *Identifier column header* is the identification file column header that unambiguously identifies the scans.
* *Ratio numerator* specifies which identification file column header designates the quantitative values to be used as a numerator for the log2-ratio calculation.
* *Ratio denominator* specifies which identification file column header designates the quantitative values to be used as a denominator for the log2-ratio calculation.
* Output Sample folder: indicates the name of the folder where the resulting data files will be saved.

#### Advanced parameters

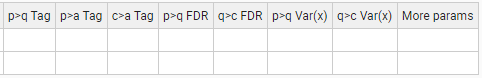
The WPP-SBT module accepts the following advanced parameters (**Figure 28**):

* *p>q Tag*, *p>a Tag*, *c>a Tag* are the text label that indicates which elements from the lower level (“p” and “c”) are integrated into the higher level (“q” and “a”). The tags must be specified in the third column of the corresponding Relation Table. This allows the user to discard elements for integration without needing to eliminate them from the Relation Table. Thus, if the label “marked” is used as a Tag, only the lower level elements containing the label marked in the third column of the lower\_level2higher\_level Relation Table will be integrated. Logical operators can also be used in the Tag field to make complex decisions.

By default, iSanXoT eliminates outliers from the lower level according to an FDR<1% threshold.

For further details, read the Advanced Parameters for the [INTEGRATE](#_Advanced_parameters) module.

* *p>q FDR*, *q>c FDR* establish an FDR threshold other than 0.01 (1%) for outlier removal in the peptide-to-protein and protein-to-category integrations, respectively. If FDR = 0 is selected, then the outliers are not discarded.
* *p>q Var(x)*, *q>c Var(x)* set a fixed value for the variance in the peptide-to-protein and protein-to-category integrations, respectively. By default, the variance will be iteratively calculated based on the Levenberg-Marquardt algorithm in the first lower level-to-higher level integration (**Figure 12**).
* *More params* allows adding more parameters to the internal programs of the module. For more details see [More params](#_More_params) in the “Special Parameters” Section.



**Figure 28. Task-table with advanced parameters in the WPP-SBT module.**

### WPPG-SBT

The WPPG-SBT module performs the integrations peptide-to-protein, protein-to-gene, gene-to-category, gene-to-geneall, and category-to-categoryall. In addition, the SBT algorithm is used to calculate the variance associated to the gene-to-category integration, which is applied to the gene-to-geneall integration.

#### Standard parameters

The standard parameters required by the WPPG-SBT module are (**Figure 25**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Experiment* is the column header that designates which elements from the Identification file will be used in the starting scan-to-peptide integration.
* *Identifier column header* is the identification file column header that unambiguously identifies the scans.
* *Ratio numerator* specifies which identification file column header designates the quantitative values to be used as a numerator for the log2-ratio calculation.
* *Ratio denominator* specifies which identification file column header designates the quantitative values to be used as a denominator for the log2-ratio calculation.
* Output Sample folder: indicates the name of the folder where the resulting data files will be saved.

#### Advanced parameters

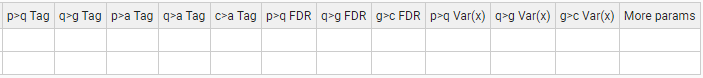
The WPPG-SBT module accepts the following advanced parameters (**Figure 29**):

* *p>q Tag*, *q>g Tag, p>a Tag, q>a Tag,* *c>a Tag* are the text label that indicates which elements from the lower level (“p”, “q” and “c”) are integrated into the higher level (“q”, “g” and “a”). The tags must be specified in the third column of the corresponding Relation Table. This allows the user to discard elements for integration without needing to eliminate them from the Relation Table. Thus, if the label “marked” is used as a Tag, only the lower level elements containing the label marked in the third column of the lower\_level2higher\_level Relation table will be integrated. Logical operators can also be used in the Tag field to make complex decisions.

By default, iSanXoT eliminates outliers from the lower level according to an FDR<1% threshold.

For further details, read the Advanced Parameters for the [INTEGRATE](#_Advanced_parameters) module.

* *p>q FDR, q>g FDR*, *g>c FDR* establish an FDR threshold other than 0.01 (1%) for outlier removal in the integrations peptide-to-protein, protein-to-gene, and gene-to-category, respectively. If FDR = 0 is selected, then the outliers are not discarded.
* *p>q Var(x)*, *q>g Var(x), g>c Var(x)* set a fixed value for the variance in the integrations peptide-to-protein, protein-to-gene, and gene-to-category, respectively. By default, the variance will be iteratively calculated based on the Levenberg-Marquardt algorithm in the first lower level-to-higher level integration (**Figure 12**).
* *More params* allows adding more parameters to the internal programs of the module. For more details see [More params](#_More_params) in the “Special Parameters” Section.



**Figure 29. Task-table with advanced parameters in the WPPG-SBT module.**

## Reports modules

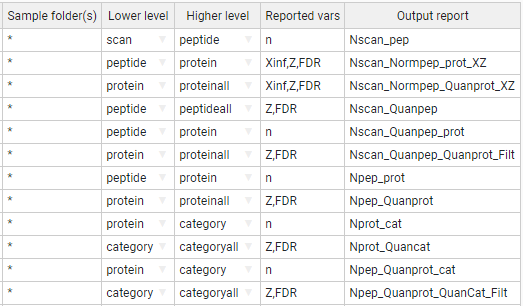
### REPORT

The REPORT module allows the collection of the statistical variables (n, tags, Xinf, Vinf, Xsup, Vsup, Z, and FDR) from the different integrations performed into result tables.

#### Standard parameters

The standard parameters showing in the REPORT task table are (**Figure 30**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Sample folder(s)* indicates the name(s) of the folder(s) where the values of the statistical variables to be retrieved are located.
* *Lower level* indicates the starting level (i.e. peptide, protein, or category) for the integration whose statistical variables are to be reported.
* *Higher level* indicates the ending level for the integration whose statistical variables are to be reported.
* *Reported vars* specifies which statistical variables will be reported. The available variables are n, tags, Xinf, Vinf, Xsup, Vsup, Z, FDR, X’inf and Winf.
* *Output report* is the report filename (without extension).



**Figure 30. A sample task table in the REPORT module.**

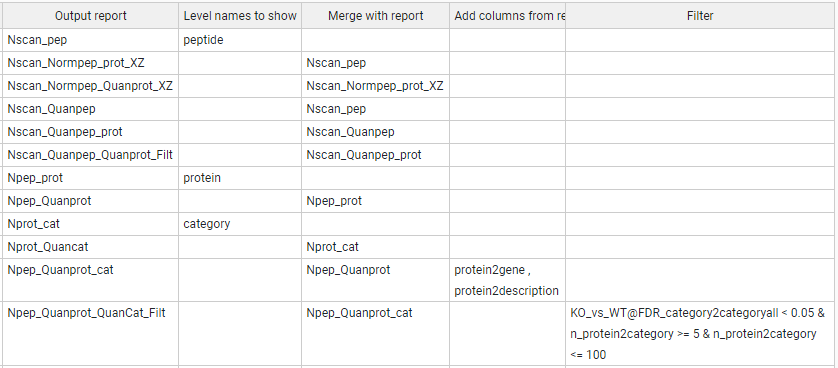
For instance, the first row of the task table shown in **Figure 30** prompts the REPORT module to read the variable “n” from the *scan2peptide\_outStats.tsv* file that contains the statistical outcome from the scan-to-peptide integration (the asterisk wildcard character in *Sample folder(s)* causes REPORT to retrieve the “n” variable from every sample). These “n” values are written to a report file named “Nscan\_pep” that is stored in the project “reports” folder.

The second row instructs the module to read the variables “Xinf”, “Z” and “FDR” from the statistical outcome of the peptide-to-protein integration (once again for every sample). These values are written to a report file named “Nscan\_Normpep\_prot\_XZ”.

#### Advanced parameters

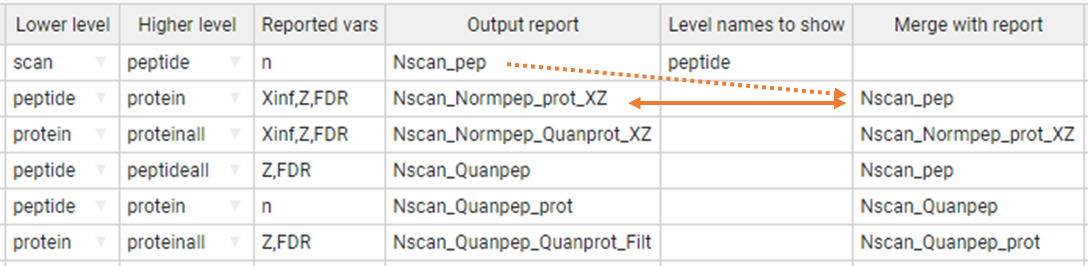
The REPORT module accepts the following advanced parameters (**Figure 31**):

* *Level names to show* allows the user to restrict the elements to be written to the *Output report* to those from the *Lower level* or the *Higher level*. Both levels are used by default.
* *Merge with report* designates the file whose *Reported vars* will be incorporated into the *Output report* after intersection with the latter file.
* *Add columns from relation table* appends *Lower level* elements, extracted from the relation table designated, to the *Output report*. It is possible to indicate multiple relation tables separated by a comma.
* *Filter* allows to filter the data to be transferred to the *Output report* based on the *Reported vars* (n, Z, FDR, etc.). For more details, see [Filter](#_Filter_(for_REPORT) in the “Special Parameters” Section.



**Figure 31. A sample task-table with advanced parameters in the REPORT module.**

The reports indicated under *Output report* and *Merge with report* are merged according to the column header that they share. Thus, the REPORT task table shown in **Figure 32** will cause the module to incorporate the number of scans per peptide, displayed in the report “Nscan\_pep”, to the report “Nscan\_Normpep\_prot\_XZ”, as these two reports share the lower level elements showing under the “peptide” header.



**Figure 32. Report merging in the REPORT module.** The first task table row creates a report file (“Nscan\_pep”) with the (n)umber of scans per peptide. The second row creates a report file called “Nscan\_Normpep\_prot\_XZ” that contains, apart from the variables “Xinf”, “Z”, and “FDR” coming from the peptide-to-protein integration, the (n)umber of scans per peptide previously stored in the “Nscan\_pep” report, as these two reports share the lower level elements showing under the “peptide” header.

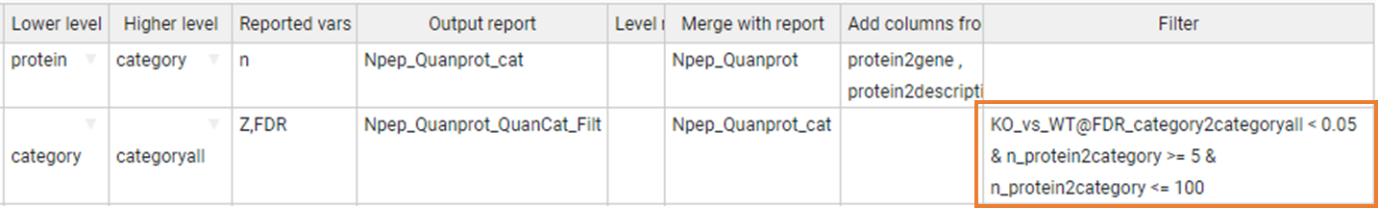
In addition, it is possible to incorporate additional data from one or more relation tables into the reports (**Figure 33**). When a given Relation Table is indicated under *Add columns from relation table*, the REPORT module will first attempt to incorporate to the *Output report* (“Npep\_Quanprot\_cat” in the example) the elements related to the *Lower level* (“protein” in this case) elements in the relation table. If the *Lower level* elements are missing in the relation table, then REPORT will try to incorporate the elements related to the Higher level (“category” in this case) instead. If neither the *Lower level* nor the *Higher level* can be found in the relation table, then no action is performed.



**Figure 33. Adding data from relation tables into the report files.** The relation table “protein2gene” contains a column with protein identifiers under the “protein” header and another column with the corresponding gene name under the “gene” header, whereas the relation table “protein2description” contains, apart from the “protein” elements, a column with the corresponding protein description. The first task table row will prompt REPORT to incorporate the gene names and protein descriptions contained in the relation tables to the report file “Npep\_Quanprot\_cat” report file, as “protein” is the Lower level they all three shares.

Finally, the report data can be filtered performing logical operations with the *Reported vars* in the *Filter* field. For instance, in the report task table displayed in **Figure 34**:

* n\_protein2category <= 100, filters out from the report the variables Z and FDR for the category-to-categoryall integration when the (n)umber of proteins per category is greater than 100.
* n\_protein2category >= 5 & n\_protein2category <= 100 retrieves the variables Z and FDR for the category-to-categoryall integration when the (n)umber of proteins per category is in the [5, 100] range.
* KO\_vs\_WT@FDR\_category2categoryall < 0.05, retrieves the variables Z and FDR for the category-to-categoryall integration provided that the FDR corresponding to the “KO\_vs\_WT” samples is less than 0.05.



**Figure 34. Filtering the report data.**

The compound variables shown in the *Filter* field on **Figure 34** follow the structure *Reported var*\_*integration*, like “n\_protein2category”. Such filter applies to all samples.

However, the filter “KO\_vs\_WT@FDR\_category2categoryall” is applied based on the variable “FDR” from the category-to-categoryall integration, but only to the “KO\_vs\_WT” sample. Moreover, the filter “WT1,WT2@FDR\_category2categoryall” is applied to the “WT1” and “WT2” samples.

### SANSON

The SANSON module generates a similarity graph showing the relationship between functional categories based on their protein components.

#### Standard parameters

The standard parameters to be provided for this module are (**Figure 35**):

* *Forced execution*: This checkbox field indicates whether to force the execution or not.
* *Sample folder(s)* indicates the names of the folder(s) where the lower level data file is located.
* *Lower level* indicates the name of the lower level elements (“protein” in this case) to be used.
* *Higher level* indicates the name of the higher level (“category” in this case) to which the lower level elements will be integrated.
* *Output Sample folder* designates a folder other than *Sample folder(s)* where the results will be saved.



**Figure 35. A sample task table in the SANSON module.**

#### Advanced parameters

The SANSON module accepts the following additional parameters (**Figure 36**):

* *Lower norm* specifies the normalization scheme to be used with the lower level elements. The default value is included the normalization of lower level to all.
* *Higher norm* specifies the normalization scheme to be used with the higher level elements. The default value is included the normalization of higher level to all.
* *Tag* is a text label that indicates which elements from the lower level are integrated into the higher level. The tags must be specified in the third column of the corresponding Relation Table. This allows the user to discard elements for integration without needing to eliminate them from the Relation Table. Thus, if the label “marked” is used as a Tag, only the lower level elements containing the label marked in the third column of the lower\_level2higher\_level Relation table will be integrated. Logical operators can also be used in the Tag field to make complex decisions.

By default, iSanXoT eliminates outliers from the lower level according to an FDR<1% threshold.

For further details, read the Advanced Parameters for the [INTEGRATE](#_Advanced_parameters) module.

* *Filter* allows to filter the data based on the FDR and number of proteins. For more details, see [Filter](#_Filter_param_(in) in the “Special Parameters” Section.

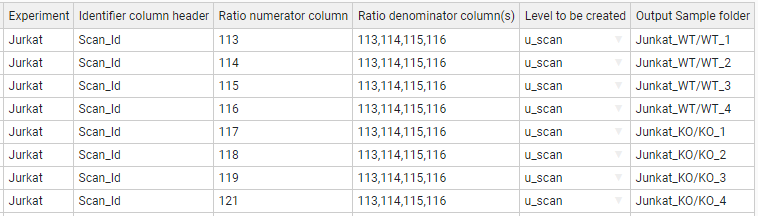


**Figure 36. A sample task-table with advanced parameters in the SANSON module.**

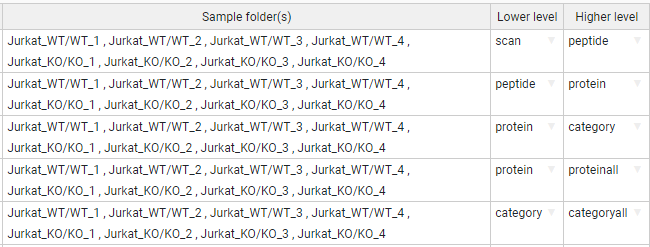
## Special parameters

### Multiple samples

The “Sample folder(s)” field of the different module task tables admit multiple samples. For instance, let's consider the samples created with the following LEVEL CREATOR task table:

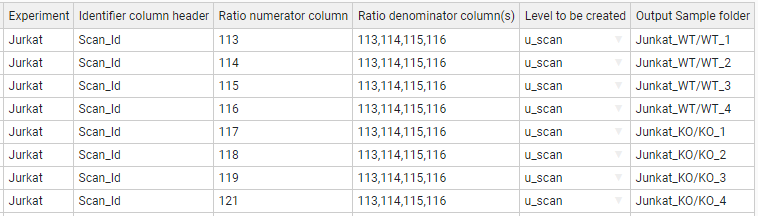


We can include multiple samples separated by a comma, for example, in the INTEGRATE module task table:

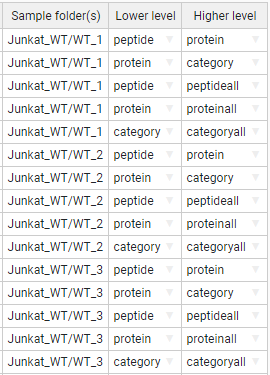


### Asterisk is our jack of all trades

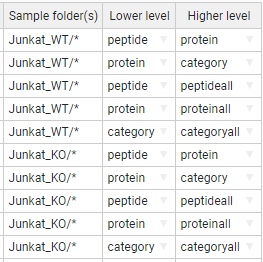
The module task tables admit the usage of the asterisk symbol as a wildcard character. Let’s once more consider the samples created with the following LEVEL CREATOR task table:



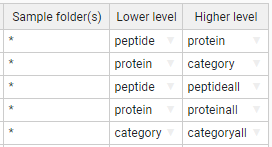
Each row calculates a ratio that is saved to the corresponding *Output Sample folder*. Thus, the ratio of 113 to the mean of 113, 114, 115, and 116 is saved to the “Jurkat\_WT/WT\_1” folder; the 114 to the mean of 113, 114, 115 ratios is saved to “Jurkat\_WT/WT\_2”, and so on. One way to create the task table of the INTEGRATE module could be the following, where each row represents an integration for a given sample:



However, this task table can be simplified applying the asterisk wildcard. For instance, the task-table below allows to indicate multiple sample folders, namely every folder starting with “Jurkat\_WT/” or “Jurkat\_KO/”.

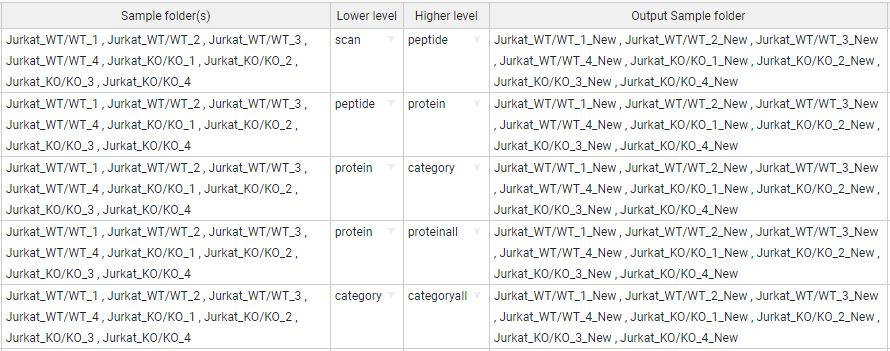


We can reduce this expression even more using just an asterisk: the first row of the following task table performs the integrations peptide-to-protein, protein-to-category, peptide-to-peptideall, protein-to-proteinall, and category-to-categoryall in every sample folder defined with LEVEL CREATOR.

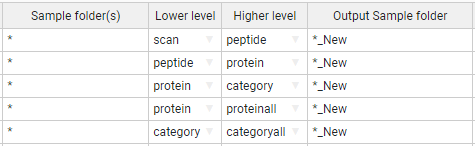


### Multiple samples in the inputs and outputs

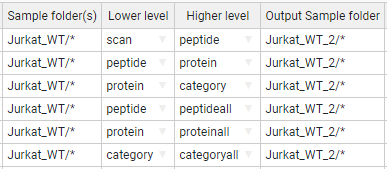
In the cases we have multiple input samples separated by comma but we want to save the results in another output sample folder, we indicate them in the same way giving the output folders separated by comma. It is required to provide the same number of folders:



In the same way happens with the asterisk character (jack of all trades). In the “Output Sample folder(s)” we can add a suffix in the input samples. The following task-table illustrate that the output sample folders would be contain the “\_New” suffix:



In addition, we can rename the subfolder adding the new name in the “Output Sample folder(s)” or add a new subfolder:



### More params

Some modules accept a column parameter in the Task-Table called “More params”. This column allows you to provide advanced parameters for the SanXoT programs [1]. The program descriptions are in the following wiki link:

<https://www.cnic.es/wiki/proteomica/index.php/SanXoT_software_package>

The iSanXoT module are composed by several programs of SanXoT. For this reason, the “More params” of a module accepts the advanced parameters of composed programs indicated by a name.

For example,

INTEGRATE:

"sanxot1": " -m 300 -g ", "sanxot2": "-s --sweepdecimals=2.5"

In above example the first “sanxot” program that compose the INTEGRATE module, receives the “-m 300 -g” as parameter, and the second “sanxot” receives also the parameter "-s --sweepdecimals=2.5".

WSPP-SBT:

"p2q\_sanxot2": " -m 100 -s ", "q2a\_sanxot1": "-m 100"

In this example, the WSPP-SBT module has multiple integrations: scan-to-peptide, peptide-to-protein, etc. For more information, see the WSPP-SBT section. Thus, the second “sanxot” program of peptide-to-protein (q) integration will receive the parameter “-m 100 -s”, and then, the first “sanxot” of protein (q)-to-proteinall (a) integration will receive the “-m 100”.

#### The program names for each Module

INTEGRATE: sanxot1, sanxotsieve, sanxot2

NORCOMBINE: create\_exp\_tags, cardenio, sanxot1, sanxotsieve, sanxot2

SBT: l2i\_sanxot1, l2i\_sanxotsieve, l2i\_sanxot2, i2h\_sanxot1, i2h\_sanxotsieve, i2h\_sanxot2, l2h\_sanxot1, l2h\_sanxotsieve, l2h\_sanxot2

WSPP\_SBT: level\_creator, klibrate, s2p\_sanxot1, s2p\_sanxotsieve, s2p\_sanxot2, p2q\_sanxot1, p2q\_sanxotsieve, p2q\_sanxot2, q2c\_sanxot1, q2c\_sanxotsieve, q2c\_sanxot2, p2a\_sanxot1, p2a\_sanxotsieve, p2a\_sanxot2, q2a\_sanxot1, q2a\_sanxotsieve, q2a\_sanxot2, c2a\_sanxot1, c2a\_sanxotsieve, c2a\_sanxot2

WSPPG\_SBT: level\_creator, klibrate, s2p\_sanxot1, s2p\_sanxotsieve, s2p\_sanxot2, p2q\_sanxot1, p2q\_sanxotsieve, p2q\_sanxot2, p2g\_sanxot1, p2g\_sanxotsieve, p2g\_sanxot2, q2g\_sanxot1, q2g\_sanxotsieve, q2g\_sanxot2, g2c\_sanxot1, g2c\_sanxotsieve, g2c\_sanxot2, p2a\_sanxot1, p2a\_sanxotsieve, p2a\_sanxot2, q2a\_sanxot1, q2a\_sanxotsieve, q2a\_sanxot2, g2a\_sanxot1, g2a\_sanxotsieve, g2a\_sanxot2, c2a\_sanxot1, c2a\_sanxotsieve, c2a\_sanxot2

WPP\_SBT: level\_creator, klibrate, p2q\_sanxot1, p2q\_sanxotsieve, p2q\_sanxot2, q2c\_sanxot1, q2c\_sanxotsieve, q2c\_sanxot2, p2a\_sanxot1, p2a\_sanxotsieve, p2a\_sanxot2, q2a\_sanxot1, q2a\_sanxotsieve, q2a\_sanxot2, c2a\_sanxot1, c2a\_sanxotsieve, c2a\_sanxot2

WPPG\_SBT: level\_creator, klibrate, p2q\_sanxot1, p2q\_sanxotsieve, p2q\_sanxot2, p2g\_sanxot1, p2g\_sanxotsieve, p2g\_sanxot2, q2g\_sanxot1, q2g\_sanxotsieve, q2g\_sanxot2, g2c\_sanxot1, g2c\_sanxotsieve, g2c\_sanxot2, p2a\_sanxot1, p2a\_sanxotsieve, p2a\_sanxot2, q2a\_sanxot1, q2a\_sanxotsieve, q2a\_sanxot2, g2a\_sanxot1, g2a\_sanxotsieve, g2a\_sanxot2, c2a\_sanxot1, c2a\_sanxotsieve, c2a\_sanxot2

### Filter in REPORT module

The REPORT module module accepts a Filter parameter. This parameter filters the data based on some variables depending on the module.

In the case of REPORT module, the filtered variables are the Reported vars: n, Z, FDR, etc. For instance:

*(FDR\_category2category < 0.05) & (n\_protein2category >=5) & (n\_protein2category <= 100)*

*(FDR\_category2category < 0.05) & (Z\_protein2proteinall >= 2 | Z\_protein2proteinall <= -2)*

### Filter in SANSON module

For the SANSON module, the filtered variables are FDR and the related number (n\_rel). For example:

([FDR] < 0.05) & ([n\_rel] >= 10) & ([n\_rel] <= 100)

Different variables can be combined using the comparisons: >=, <=, !=, <>, ==, >, <; and using logical operators “and” (&), “or” (|), and “not” (!).

# Sample Workflows with Application to Case Studies

Below, we provide detailed descriptions of four sample workflows that illustrate the capability of iSanXoT to statistically ascertain changes in protein or peptide abundance across various biological contexts. It is important to note that these workflows can be easily reused to process new data (refer to the next section).

**Workflow 1: One-step quantification in a labeled experiment**

**Experimental**

The identification and quantification data from García-Marqués *et al*. [3] were used to illustrate this workflow. This study characterizes the molecular alterations that take place over time when vascular smooth muscle cells (VSMCs) are treated with angiotensin-II (AngII) for 0, 2, 4, 6, 8, and 10 hours. Quantitative proteomics were performed using isobaric iTRAQ 8-plex labeling. Workflow 1 analyzes a) changes in protein abundance and b) alterations in functional categories produced by the coordinated behavior of proteins at each of the specified times, in relation to time 0. This is achieved in only one step using the compound module WSPP-SBT, which automatically performs all the required tasks.

**Workflow execution**

The workflow template and required input files for executing this workflow can be downloaded from

<https://github.com/CNIC-Proteomics/iSanXoT/wiki/studies/cases/templates/WSPP-SBT.zip>

Refer to the *Importing a Workflow Template* section below for detailed instructions.

**Workflow operation**

Workflow 1 requires the RELS CREATOR module, the WSPP-SBT compound module, and the REPORT basic module (*Figure 37*). The relation tables necessary for performing the integrations are created by the RELS CREATOR module (*Figure 37*A) from a table provided by the user. The WSPP-SBT module performs a sequence of consecutive integrations based on the WSPP statistical model [2] and the SBT algorithm [3] (*Figure 37*B). Finally, the REPORT module organizes the data into tables containing the required information.

A diagram of a level calibration

Description automatically generated

*Figure 37. Scheme of workflow 1 (one-step quantification in a labeled experiment) showing module components: RELS CREATOR (A) and WSPP-SBT and REPORT (B)*

The WSPP-SBT module requires the user to define the meaning of relative abundances, which iSanXoT consistently expresses as log2ratios. In this case, the abundance data corresponds to the intensities of iTRAQ reporters at the scan level, tabulated in the “ID-q” file with the name of each reporter as a column header (see below for how these tables are generated). The intensities of each scan at 0 h are in the “Abundance: 113” column and serve as a common reference to express abundance ratios; thus, they are used as the denominator. The reporter intensities corresponding to different time points serve as numerators for the ratios. The task table also enables the user to assign an easily identifiable name to the folders where the quantitative values of each sample are stored (*Figure 38*).

A screenshot of a number of numbers

Description automatically generated

*Figure 38. The WSPP-SBT task table for workflow 1.*

The WSPP-SBT module initially conducts a calibration process, assigning a statistical weight to each log2ratio value at the scan level (*Figure 37*B), as described [2]. The statistical weight of each scan is the inverse of the estimated variance associated with the log2 of intensity ratios [2]. Following the calibration of data at the scan level, the workflow proceeds with integrations from *scan-to-peptide* and *peptide-to-protein*.

At the protein level, the SBT algorithm is applied for the detection of functional category changes originating from the coordinated behavior of proteins (*Figure 37*B). The algorithm first calculates the variance of the *protein-to-category* integration, providing an improved estimate of the technical protein variance that is less influenced by biological changes [3]. This protein variance is then utilized to perform the *protein-to-grand* mean integration (hereinafter referred to as *protein-to-proteinall*), from which statistically significant abundance changes are detected. Finally, the algorithm conducts the *category-to-grand* mean integration (hereinafter referred to as *category-to-categoryall*), identifying statistically significant category changes. All results from the integrations performed by the WSPP-SBT module are saved for each sample in the *Output Sample folder*, as indicated in the module task table (*Figure 38*).

In every integration step, a relation table (a text file) is required to link lower- to higher-level elements. These relation tables can be automatically generated by the RELS CREATOR module (*Figure 37*A, upper) or provided by the user (*Figure 37*A, lower). In this example (*Figure 39*) the relation tables linking scans to peptides and peptides to proteins are obtained from the “ID-q” file by specifying the column names where they are located (in this case, “Scan\_Id”, “Pep\_Id”, and “Master Protein Accessions”). The columns “Master Protein Accessions” and “Master Protein Descriptions” in the “ID-q” file contain the accession numbers and complete names of the proteins, respectively. Consequently, a relation table, *protein2description*, is also created, which can later be used to append the full name of the protein to any of the created reports (see below).

An example of the *peptide2protein* relation table, linking the identified peptides to the proteins they originate from, is shown *Figure 40*A. The elements of the relation table *protein2category* were extracted from a text file containing functional annotations for mouse proteins, compiled from various protein function databases (*Figure 40*B), as described by the authors [3]. It is important to note that relation tables are by default extracted from the “ID-q” file. To use other text files, the absolute path with the location of the text file must be indicated. The relation tables *protein2proteinall* and *category2categoryall* guide the integration to a grand mean (a common element called “[1]”). Although the integration *peptide2peptideall* is not necessary in this workflow, it is included in this example since it may be useful to inspect quantifications at the peptide level.

A screenshot of a computer

Description automatically generated

*Figure 39. The RELS CREATOR task table was specifically designed for workflow 1.*

A screenshot of a computer

Description automatically generated

*Figure 40. An excerpt from the peptide2protein (A) and protein2category (B) relation tables, illustrating the links between peptides and proteins and proteins and categories, respectively.*

After the integrations are executed, the REPORT module is employed to gather the specified statistical variables from the *Output Sample folders* designated by the user and organize them into tables (*Figure 41*). In this instance, the tabulation is focused on the results from the samples (2h-AngII, 4h-AngII, 6h-AngII, 8h-AngII, and 10h-AngII).

In this example the REPORT module creates a protein table and a category table by performing the following steps:

* Create a table named “Npep2prot”, which contains the count of peptides used to quantify each protein.
  + This involves extracting the number of elements (n) from the *peptide-to-protein* integrations in the specified folders, representing the lower level (peptide) used for the quantitation of the higher level (protein).
* Create a table named “Npep2prot\_Quantprot\_filtered”, including protein changes (Zqa) and their statistical significance (FDRqa).
  + This is achieved by extracting standardized log2 ratios (Z) and False Discovery Rates (FDR) from the *protein-to-proteinall* integration in the indicated folders, representing the lower level (protein).
* Add the count of peptides with which each protein is quantified to the “Npep2prot\_Quantprot\_filtered” table.
  + This step involves merging the previous table with the existing “Npep2prot” table based on the common level (protein), excluding a specific column (peptide), and eliminating duplicate entries.
* Add an additional column to this table with the complete description of the proteins.
  + This is accomplished by merging the previous table with the relation table *protein2description* based on the common level in both tables (protein).
* Filter the table to include only proteins with a statistically significant abundance change (FDR < 0.01).
  + This is achieved by applying a condition based on the FDR to the results from the *protein2proteinall* integration. For more detailed information, refer to the “Filter for report” section in the iSanXoT wiki: <https://github.com/CNIC-Proteomics/iSanXoT/wiki>.
* Create a table named “Nprot2cat”, which contains the count of proteins used to quantify each category.
  + Extract the number of elements (n) from the *protein-to-category* integrations in the specified folders, representing the lower level (protein) used for the quantitation of the higher level (category).
* Create a table named “Nprot2cat\_Quantcat\_filtered”, incorporating category changes (Zca) and their statistical significance (FDRca).
  + Extract standardized log2 ratios (Z) and False Discovery Rates (FDR) from the *category-to-categoryall* integration in the indicated folders, representing the lower level (category).
* Add to this table the count of proteins with which each category is quantified.
  + This is achieved by merging the previous table with the existing table “Nprot2cat” based on the common level in both tables (category), excluding a specific column (protein), and eliminating duplicate entries.
* Filter the table to include only categories with a statistically significant change (FDR < 0.01).
  + This is done by applying a condition based on the FDR to the results from the *category2categoryall* integration.
* Create a table named “Npep2prot\_Quanprot” containing the number of peptides per protein, protein changes (Zqa), and their statistical significance (FDRqa).
  + This is done as previously explained, excluding the protein descriptions and filters.
* Create a table named “Nprot2cat\_Quancat\_Quanprot\_filtered”, including category changes (Zca) and their statistical significance (FDRca).
  + Extract standardized log2 ratios (Z) and False Discovery Rates (FDR) from the *category-to-categoryall* integration in the indicated folders, representing the lower level (category).
* Add to this table the count of proteins per category, protein changes (Zqa), and their statistical significance (FDRqa).
  + This is achieved by merging the previous table with the existing tables “Nprot2cat” and “Npep2prot\_Quantprot”.
* Filter the table to include only categories containing 5 or more proteins or 100 or fewer proteins.
  + This is done by applying a set of conditions in the *Filter* column, joined with the "&" operator.

Note that the commands in the REPORT module, facilitating the construction of tables essential for typical quantitative proteomics projects, are easily adaptable and reusable for other projects.

*A screenshot of a computer

Description automatically generated*

*Figure 41. The REPORT task table that was designed for workflow 1.*

In *Figure 43,* two heat maps are presented, constructed from the protein and category tables obtained using the REPORT module.

A comparison of a number of dna samples

Description automatically generated with medium confidence

Figure 42. Relative abundance changes of proteins (Zqa, left) and functional categories (Zca, right) are derived from the “Npep2prot\_Quanprot\_filtered” and “Nprot2cat\_Quancat\_filtered” reports, respectively, generated by the REPORT module in workflow 1 (Figure 41). Both report tables were sorted based on the averages of Zqa and Zca, respectively.

In Figure 43 illustrates examples of functional categories displaying statistically significant changes resulting from coordinated protein behavior. The data for these plots are derived from the “Nprot2cat\_Quancat\_Quanprot\_filtered” table generated by the REPORT module.

A group of graphs with different colors

Description automatically generated with medium confidence

Figure 43. Examples of time-dependent coordinated protein behavior in VSMCs treated with angiotensin-II, revealed by the distribution of the standardized log2 ratio (Zqa) of protein components in each category.

**Workflow 2: Step-by-step quantification and sample combination in a labeled experiment**

**Experimental**

The workflow presented here utilizes data from González-Amor *et al*. [5] focusing on the contribution of interferon-stimulated gene 15 (ISG15) to vascular damage associated with hypertension. The study employs knockout mutants for the ISG15 gene, subjecting animals to AngII treatment or not. A total of 16 samples from mouse aortic tissue represent four groups: four WT-Control mice, four ISG15-KO mice, four WT+AngII mice, and four ISG15-KO+AngII mice. The experiment, conducted in two isobaric iTRAQ 8-plex batches, serves as a case study to guide the step-by-step creation of a workflow. This workflow integrates quantitative results from individual samples to the protein level, consolidates protein data across the four biological replicates in each group, establishes ratios between conditions, and analyzes functional category changes due to coordinated protein behavior using the SBT model.

A diagram of a diagram

Description automatically generated

*Figure 44. Schematic representation of workflow 2, which involves step-by-step quantification and sample combination in a labeled experiment. The figure illustrates the key modules in the workflow, including RELS CREATOR (A) and LEVEL CREATOR, LEVEL CALIBRATOR, INTEGRATE, NORCOMBINE, RATIOS, SBT, and REPORT (B)*

**Workflow execution**

The workflow template and input files that are needed to execute this workflow can be downloaded from <https://github.com/CNIC-Proteomics/iSanXoT/wiki/studies/cases/templates/WSPP_NORCOM_RATIOS_SBT.zip>

Refer to the *Importing a Workflow Template* section below for detailed instructions.

**Workflow operation**

Workflow 2 encompasses all six basic modules: LEVEL CREATOR, LEVEL CALIBRATOR, INTEGRATE, NORCOMBINE, RATIOS, and SBT, along with the REPORT module (*Figure 44*). The task table in the starting module, LEVEL CREATOR, generates files at the scan level containing log2 ratios and the corresponding sample folders (*Figure 45*). In this example, similar to workflow 1, the name of each iTRAQ reporter served as a column header in the “ID-q” file containing the intensities. Additionally, the “Experiment” column indicates whether the intensities come from the first or second iTRAQ 8-plex batch. Each iTRAQ batch comprises two biological replicates from each of the four groups. The average of reporter intensities from the two untreated WT mice (reporters in the columns “113” and “117”) is used as an internal control within each batch, serving as the denominator for the log2 ratios.

A screenshot of a computer

Description automatically generated

*Figure 45. The LEVEL CREATOR task table specifically designed for workflow 2.*

LEVEL CREATOR generates the *u\_scan* (uncalibrated scan) files, comprising scan identifiers (extracted from the “Scan\_Id” column in the “ID-q” table), log2-ratios at the scan level (*Xs*, as defined in the task table), and uncalibrated weights (*Vs*, corresponding to the intensities of the reporters in the Ratio numerator column) (*Figure 46*). These uncalibrated weights (*Vs*) are indicative of quantification quality, where a higher weight implies more accurate quantification. However, at this stage, they are not associated with statistical variance.



*Figure 46. An excerpt from one of the u\_scan files generated by the LEVEL CREATOR module in workflow 2 is displayed, showing element identifiers in the left column, log2 ratios in the center column, and statistical weights in the right column.*

The LEVEL CALIBRATOR module calibrates the Vs weights by conducting a *u\_scan-to-peptide* integration, generating scan files with true, calibrated statistical weights, as defined in the WSPP model (the inverse of the estimated individual scan variances) (*Figure 47*, Top).

A screenshot of a computer

Description automatically generated

Figure 47. The LEVEL CALIBRATOR (Top) and INTEGRATE (Bottom) task tables for workflow 2.

Note that the LEVEL CALIBRATOR automatically generates a plot to supervise the accuracy of calibrations (the "\*\_outGraph\_VRank" PNG file) in each sample folder. This plot shows whether the model is able to predict experimental scan variances as a function of the calibrated statistical weights (see *Figure 48*).

A graph with green and red dots

Description automatically generated

Figure 48. Automatically generated graphs to supervise the accuracy of calibrations. These graphs illustrate 1/MSD versus the rank of Vs (scan weight, which at this level corresponds to reporter intensity). MSD represents the Mean Squared Deviation of the scans versus the respective mean of the peptide to which they belong. The scans are ordered by Vs, and the MSD is calculated in a sliding window of 200 scans [2].

Similar to workflow 1, before conducting the integrations, the relation tables need to be created using the RELS CREATOR module. These tables exhibit a similar structure, as depicted in *Figure 49*.

A screenshot of a computer

Description automatically generated

Figure 49. Illustrates the RELS CREATOR task table for workflow 2.

The INTEGRATE module performs the *scan-to-peptide*, *peptide-to-protein*, and *protein-to-protein* integrations based on the module task table (*Figure 47*, Bottom). It's important to note that, for consistency, all files created for each sample are automatically stored in the folder specified in the task table of the LEVEL CREATOR module, unless an alternative location is specified in the *Output Sample folder* column.

Note that the INTEGRATE module automatically generates a plot to assess the accuracy of the GIA integration model in each integration step. This is achieved by comparing the distribution of Z values with that of the null hypothesis (standard normal distribution) (refer to *Figure 51*, left panels). These graphs are then stored in the respective sample folders as “\*\_outGraph.png” files.

By default, iSanXoT removes integration outliers. However, to avoid the removal of outlier elements in *the protein-to-proteinall* integration, particularly as these represent significantly altered proteins, a 0 FDR value was specified in the INTEGRATE task table for this integration (refer to Figure 47, Bottom).

Once protein levels are established, workflow 2 utilizes the NORCOMBINE basic module (*Figure 44*B) to integrate protein values from the four biological replicates within each group. The resulting integrated protein values per group are then stored in the folders WT-C, WT-AngII, ISG15-C, and ISG15-AngII (*Figure 50*).

A screenshot of a computer

Description automatically generated

Figure 50. The NORCOMBINE task table for workflow 2.

The NORCOMBINE module integrates biological replicates within sample groups using the GIA algorithm [3]. This algorithm models the distribution of protein values around the average, considering error propagation theory and estimating a global variance for the integration. The GIA algorithm operates under the assumption that individual variances of all lower elements (proteins) are influenced by a global variance, arising from biological variability within the same group. While this assumption may not hold in all the cases, it can be easily checked by inspecting the test distributions.

Similar to the INTEGRATE module, the NORCOMBINE module automatically generates graphs comparing the distribution of the integrated Z variables with those of the standard normal distribution. As depicted in *Figure 51* (right), the distribution of protein Z values estimated by the model in the case of the ISG15-AngII group aligns well with the null hypothesis. This agreement demonstrates that the assumption of the model is a suitable approach for handling the biological variance of samples within this group. Comparable results were obtained in the other three groups (not shown).

A graph of a function

Description automatically generated with medium confidence

Figure 51. Distribution of the standardized log2 protein ratios (Zqa) from the four individual ISG15-AngII VSMC samples (Left panel). This illustrates how the WSPP model aligns well with the expected null distributions in the four cases. In the right panel, the figure shows the distribution from the integrated ISG15-AngII sample group obtained with the NORCOMBINE module. This demonstrates how the GIA assumption of a global biological variance is a valid approach to address the biological variability within this group. Red: null hypothesis (standard distribution); blue: experimental data.

Note also that the NORCOMBINE module employs a weighted averaging technique from multiple samples. Through its good fit to the null hypothesis (as illustrated in *Figure 51*), this approach enables accurate control over outliers. This unique approach allows the integration of protein values originating from unbalanced sample groups, distinct experiments, various mass spectrometers, and even different labeling techniques (refer to, for instance, [2]).

The module's task table specifies that samples were combined at the protein level using the *proteinall* level for normalization. In this process, log2 protein ratios are initially normalized by the grand mean before being integrated into an averaged protein value. This normalization compensates for differences in protein load into each iTRAQ channel. Importantly, it's worth noting that proteins could also be integrated at other levels (such as organelles, subcellular compartments, complexes, etc.) before being further integrated by NORCOMBINE, allowing for various types of normalization. Finally, the *lowerNorm* column indicates the file containing the normalized data, typically the *lowerNormV* files previously generated by the INTEGRATE module. For more detailed information, please refer to the iSanXoT documentation.

The protein averages derived from the four biological sample groups are then employed by the RATIOS basic module to calculate two ratios: *WT-AngIIvsWT-C*, where wild-type AngII-treated animals are compared to controls, and *ISG15-AngIIvsISG15-C*, where ISG15 AngII-treated animals are compared to ISG15 controls (*Figure 52*). In the "V method" column, users can specify the method used to assign a statistical weight to the log2ratios, with the default being the "max" method (for more details, please refer to the “RATIOS” module in the iSanXoT wiki: <https://github.com/CNIC-Proteomics/iSanXoT/wiki>).

A screenshot of a computer

Description automatically generated

*Figure 52. The RATIOS task table for workflow 2.*

The final basic module executed in workflow 2 is the SBT (*Figure 53*), which applies the SBT algorithm to the previously defined comparisons. The goal is to detect changes in functional categories resulting from the coordinated behavior of proteins, following the approach explained in workflow 1. The SBT module offers greater flexibility by allowing triangle operations on any level, not just proteins. In this case, the triangle is formed by the levels protein and category (*Figure 52*) and the corresponding grand mean.

A screenshot of a computer

Description automatically generated

*Figure 53. The SBT task table for workflow 2.*

Finally, the REPORT module (*Figure 54*) is used, as in Workflow 1, to generate tables with protein and category data. In this case, additional features of the REPORT module are utilized. The table “Npep2prot” is generated using an asterisk. This symbol serves as a wildcard character for iSanXoT, indicating that the results from all samples containing peptide-to-protein integration (i.e., ISG15-AngII-1, ISG15-AngII-2, ISG15-AngII-3, ISG15-AngII-4, ISG15-AngII, ISG15-C-1, ISG15-C-2, ISG15-C-3, ISG15-C-4, ISG15-C, and ISG15-AngIIvsISG15-C) are to be included in the table.

However, the “Npep2prot\_Quanprot\_ISG15\_filtered” and “Npep2prot\_Quanprot\_WT\_filtered” tables include protein changes (Zqa), the statistical significance (FDRqa) of these changes, and the number of peptides per protein only from the samples indicated in the *Sample folder(s)* column. The report for the ISG15 samples is filtered by Zqa to display the most extreme values (greater than 1 or less than -1) but only for the “ISG15-AngIIvsISG15-C” sample. Additional filters for the minimum number of peptides per protein are also applied to these tables. The tables containing category values are filtered by Zca (greater than or equal to 2 or less than or equal to -2) and/or by the number of proteins per category (between 5 and 100).

A screenshot of a computer

Description automatically generated

*Figure 54. The REPORT task table designed for workflow 2.*

The tables generated by REPORT can be used to generate heatmaps showing the most relevant protein abundance changes (*Figure 55*). As previously shown [5], iSanXoT analysis revealed a coordinated alteration of proteins implicated in cardiovascular function, extracellular matrix and remodeling, and vascular redox state in aortic tissue from AngII-infused ISG15-KO mice (*Figure 56A*). The coordinated protein behavior from some of the altered categories can be analyzed in the sigmoid plots (*Figure 56B*).

A screenshot of a diagram

Description automatically generated

*Figure 55. Differential abundance of functional proteins revealed by workflow 2. The heatmap (A) for proteins (Zqa) is based on the “Npep2prot\_Quanprot\_ISG15\_filtered” REPORT table. The heatmap (B) displays the proteins (Zqa) for the WT samples using the “Npep2prot\_Quanprot\_WT\_filtered” REPORT table.*

A screenshot of a graph

Description automatically generated

Figure 56. Functional category changes arising from coordinated protein behavior. A) Bar graph for functional categories (Zca) constructed from the “Nprot2cat\_Quancat\_filtered” REPORT table. B) The distributions of the standardized log2 protein ratios (Zqa) are shown for some of the functional categories that are significantly down-regulated (Left) or up-regulated (Right). The data to create the sigmoid curves are taken from the “Nprot2cat\_Quancat\_Quanprot\_filtered” REPORT table.

**Workflow 3: Quantification of posttranslationally modified peptides in a labeled experiment**

**Experimental**

This workflow was employed to quantify reversibly oxidized Cys peptides in mouse embryonic fibroblast (MEF) preparations subjected to chemical oxidation with diamide. The experiment aimed to illustrate the comparative performance of on-filter (FASILOX) and in-gel (GELSILOX) approaches for studying the thiol redox proteome [4]. These techniques involved differentially labeling Cys residues based on their oxidation state, resulting in two distinct populations of reduced and oxidized Cys-containing peptides. MEF samples were incubated with diamide (treated group) or PBS (control group), and the resulting peptides were isobarically labeled with iTRAQ 8-plex (four biological replicates per condition). The workflow is designed to detect statistically significant abundance changes in peptides containing modified Cys residues.

A diagram of a diagram of a dna

Description automatically generated with medium confidence

*Figure 57. Scheme of workflow 3 (quantification of posttranslationally modified peptides in a labeled experiment) showing module components: RELS CREATOR (A) and LEVEL CREATOR, LEVEL CALIBRATOR, INTEGRATE, and REPORT (B).*

**Workflow execution**

The workflow template and the required input files for executing this workflow can be downloaded from

<https://github.com/CNIC-Proteomics/iSanXoT/wiki/studies/cases/templates/WSPP_PTM.zip>

Please refer to the *Importing a Workflow Template* section below for detailed instructions.

**Workflow operation**

Workflow 3 comprises the basic modules LEVEL CREATOR, LEVEL CALIBRATOR, and INTEGRATE, as well as the RELS CREATOR and REPORT modules (*Figure 57*) and is very similar to workflow 2. LEVEL CREATOR was used to design the ratios and to generate the level files, sample folders and log2 ratios indicated in the corresponding task table (*Figure 58* and *Figure 59*). LEVEL CALIBRATOR was used to calibrate statistical weights (*Figure 60*, top) and INTEGRATOR to integrate from scan to peptide and from peptide to protein (*Figure 60*, bottom).



*Figure 58. The LEVEL CREATOR task table for workflow 3.*

A screenshot of a graph

Description automatically generated

*Figure 59. Excerpt from one of the* uscan *files generated by workflow 3 LEVEL CREATOR module showing element identifiers (left column), log2 ratios (center column) and statistical weights (right column).*

The only difference with workflow 2 lies in the INTEGRATE command used for the integration *peptide-to-protein*. INTEGRATE can use a modified version of the GIA algorithm for the quantitative analysis of posttranslational modifications (PTM) that includes a third column containing *tags* in the relation tables, as described [7]. In this workflow the advanced option of INTEGRATE was activated to display the *Tag* column, which is used to include only the peptides which are tagged in the relation table with the text “*Not modified”* when calculating the protein averages (*Figure 60*). An example of tagged *peptide2protein* relation table is shown in *Figure 61.* Proteins are thus quantified using only peptides which are not modified in Cys. However, although these Cys peptides do not contribute to protein averages, they are assigned a *Zpq* value, which serves to evaluate whether they deviate significantly from the expected distribution of peptides around their protein averages [4]. If the deviation is statistically significant it can be concluded that there is a change in abundance of the posttranslational modification in relation to the protein it comes from. This philosophy can be extended to any other kind of PTM.

A screenshot of a computer

Description automatically generated

*Figure 60. The LEVEL CALIBRATOR (Top) and INTEGRATE (Bottom) task tables for workflow 3.*

A screenshot of a computer

Description automatically generated

Figure 61. Excerpt from the peptide2protein relation table used to integrate peptides to proteins. Note the presence of a third column used to tag Cys-containing peptides, which will be excluded from the calculation of protein averages in the peptide-to-protein integration.

iSanXoT allows the automatic generation of relation tables containing tags, which are extracted from the “ID-q” table. To achieve this, RELS CREATOR employs a specific option (*Figure 62*). In this case, this option instructs RELS CREATOR to search the “ID-q” table for the column with the header *Modifications* and to translate its content into the third column of the *peptide2protein* relation table. In the “ID-q” table used in this instance, peptides containing modified Cys residues were labeled as either “Reduced-Cys peptides” or “Oxidized-Cys peptides” depending on the type of modification. RELS CREATOR locates these tags in the relation table (*Figure 61* and *Figure 62*). iSanXoT allows the use of any tag created by search engines or defined by the user, with the sole condition that the tag indicated in the INTEGRATE command must match the tag in the third column of the relation table (*Figure 60*, bottom).

A screenshot of a computer

Description automatically generated

Figure 62. The RELS CREATOR task table for workflow 3.

Finally, the REPORT module compiles the statistical variables generated by the *peptide-to-protein* integration for all the samples (*c1, c2, c3, c4, t1, t2, t3*, and *t4*), as indicated by the asterisk (*Figure 63A*). The REPORT commands mirror those employed in generating the protein tables in workflow 1, with the distinction that peptide values are tabulated instead of protein values, along with the inclusion of the number of scans per peptide instead of the number of peptides per protein. The false discovery rate (FDR) at the peptide level enables the detection of statistically significant changes in posttranslational modifications (PTM). This REPORT also generates a second filtered peptide table containing peptides with reduced Cys and the most pronounced abundance changes. This table was utilized to create a heatmap (*Figure 63B*).

A screenshot of a computer

Description automatically generated

Figure 63. (A) The task table for Workflow 3 is in the REPORT module. (B) Relative abundance of Cys-containing peptides in MEF samples is represented by peptide log2 ratios expressed in units of standard deviation corrected by the protein mean (Zpq). The data for the heatmap were derived from the “Nscan2pep\_Quanpepprot\_filtered” report table.

Of significance, the peptides integrated followed a standard distribution in all eight samples, as depicted in blue in *Figure 64* for *t1*, *t2*, *c1*, and *c2* samples. This evidences that the error distribution at the peptide level could be accurately modeled using the GIA algorithm. In addition, the treatment produced a generalized increase in the abundance of oxidized Cys-containing peptides (orange curves), with concomitant decrease in the abundance of reduced Cys-containing peptides (green curves). Consistently, the opposite changes were observed in the controls.

A graph of different colored lines

Description automatically generated

Figure 64. Distribution of the standardized variable at the peptide level (Zpq) in control MEF samples (t1, t2, c1 and c2) for all the peptides quantitated (blue) and the oxidized (orange) and reduced (green) Cys-containing peptide subpopulations. The theoretical normal distribution N(0,1) is shown in red. Positive/negative Zpq values indicate increased/decreased peptide abundance with respect to the average. These sigmoidal curves were created from the “Nscan2pep\_Quanpepprot” table generated by REPORT.

**Workflow 4: Label-free quantification**

**Experimental**

This workflow was employed to analyze quantitative data obtained from a multicenter study conducted in Data-dependent Acquisition (DDA) mode. The study utilized samples prepared exactly as described in the paper by Navarro *et al*. [6]. Two hybrid proteome test samples were generated in the study, comprising tryptic digests of human, yeast, and *Escherichia coli* proteins mixed in two distinct proportions, as detailed in Table S1.

In this workflow, quadruplicate peptide samples underwent analysis by LC-MS/MS. Following that, the MaxQuant [8] software was employed for peptide identification and quantification. For guidance on pre-processing data from MaxQuant and other software to adapt for use with iSanXoT, please consult the section described below.

Table S1. Proteome-hybrid samples A and B were prepared, each containing known quantities of peptide digestions of HeLa, Saccharomyces cerevisiae, and Escherichia coli. The samples were then mixed according to the procedure outlined in Navarro et al., Nature Biotech 2016.

|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | FOLD B/A |
| *E. coli* | 20% | 5% | 0.25 |
| *S. Cerevisiae* | 15% | 30% | 2 |
| *HeLa* | 65% | 65% | 1 |

A diagram of a protein source

Description automatically generated

Figure 65. Scheme of workflow 4 (label-free quantification) showing module components: RELS CREATOR (A) and LEVEL CREATOR, LEVEL CALIBRATOR, INTEGRATE, NORCOMBINE, and REPORT (B).

**Workflow execution**

The workflow template and necessary input files for executing this workflow can be downloaded from

<https://github.com/CNIC-Proteomics/iSanXoT/wiki/studies/cases/templates/WPP_LabelFree.zip>

Detailed instructions can be found in the *Importing a Workflow Template* section below.

**Workflow operation**

Workflow 4 includes the basic iSanXoT modules: LEVEL CREATOR, LEVEL CALIBRATOR, INTEGRATE, NORCOMBINE, and RATIOS, along with the REPORT and RELS CREATOR modules (*Figure 65)*. The starting module, LEVEL CREATOR, generates the level files, sample folders and log2 ratios indicated in the corresponding task (*Figure 66*) based on the quantitative data at the peptide level obtained with MaxQuant for replicate A- and B-type samples. In this example, we used the average of peptide intensities across all samples as the denominator of the log2 ratio. However, in this case, the averages of the four A-type and the four B-type samples are first calculated separately (as indicated by the square brackets), and then the average of the two averaged values is calculated (as indicated by the comma). This ensures that no log2 ratio is calculated when the four values are missing in either the A or the B sample group. This module generates uncalibrated files at the peptide level (*u\_peptide*) (Figure 67).

A screenshot of a computer

Description automatically generated

Figure 66. The LEVEL CREATOR task table for workflow 4.

A screenshot of a graph

Description automatically generated

Figure 67. Excerpt from one of the u\_peptide files generated by workflow 4 LEVEL CREATOR module showing element identifiers (left column), log2 ratios (center column) and uncalibrated statistical weights (right column).

The *u\_peptide* level files are subsequently calibrated using the LEVEL CALIBRATOR module through integration to the protein level (Figure 68*, Top*), resulting in calibrated *peptide* level files. The INTEGRATE module then executes *peptide-to-protein* and *protein-to-proteinall* integrations as specified in the module task table (Figure 68, Bottom).

In this example, it is important to note that the advanced option of INTEGRATE was activated to utilize the Tag column. Consequently, only proteins containing the *Homo sapiens* tag are employed in the *protein-to-proteinall* integrations (Figure 68*, Bottom*). The decision to restrict integration to human proteins serves two purposes: a) normalization is performed using the grand mean of human proteins, unaffected by the presence of yeast or E. coli proteins; b) estimation of the variance in the *protein-to-proteinall* integration relies solely on human proteins, mitigating the impact of yeast and E. coli proteins, which exhibit significant deviations from the mean. It is essential to recognize that this procedure does not remove yeast or E. coli proteins from the normalized files subsequently utilized by the NORCOMBINE module (as explained below).

A screenshot of a computer

Description automatically generated

Figure 68. The LEVEL CALIBRATOR (Top) and INTEGRATE (Bottom) task tables for workflow 4.

In this workflow, the protein level comprises numeric values derived from the “Protein group IDs” provided by *ID-q*, generated through the *modificationSpecificPeptides.txt* file from MaxQuant (refer to the *Preparing the ID-q file from MaxQuant output* section). Similar to workflow 3, the *protein-to-proteinall* relation table must include a third column that tags the species from which each protein originates (see Figure 69). The species names have been extracted from the *ID-q* file. It is important to note that the tag indicating human proteins aligns with the tag indicated in INTEGRATE (see Figure 68*, Bottom*).

A table with numbers and text

Description automatically generated

Figure 69. Excerpt from the protein2proteinall workflow 4 relation table that illustrate the linkage between proteins and a constant value representing the protein grand mean. The protein level is the "Protein group IDs" obtained from ID-q, generated by MaxQuant. It is noteworthy that a third column is employed to tag proteins with their respective species, facilitating later species-specific integration in the protein-to-proteinall context.

The *protein-to-proteinall* relation table is automatically generated by the RELS CREATOR module, which extracts information from the *ID-q* file created by the user. This file includes the relationship between protein group identifiers (under the *Protein group IDs* column header) and the corresponding species (under the *Species* column header).

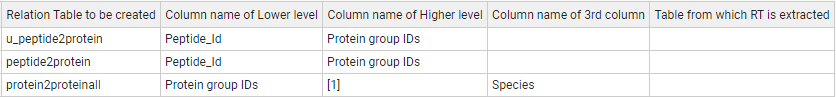


Figure 70. The RELS CREATOR task table for workflow 4.

Next, the normalized data at the *protein* level from the four replicates from each sample are combined into samples *A* and *B*, respectively, using the NORCOMBINE basic module (Figure 71, Top). To compare these two samples, new log2 ratios and statistical weights are calculated using the RATIOS basic module (Figure 71, Bottom). Finally, a *protein-to-proteinall* integration is carried out for the newly generated *B\_vs\_A* sample by the module INTEGRATE (Figure 68, Bottom), using again the *Homo sapiens* tag.

A screenshot of a computer

Description automatically generated

Figure 71. The NORCOMBINE (Top) and RATIOS (Bottom) task tables for workflow 4.

The REPORT module, employed in this workflow, compiles data at the protein level alongside the number of peptides per protein, like previous workflows (Figure 72). In this specific instance, the table encompasses Z and FDR, as well as the log2ratios of proteins from all samples (*Xinf* from the *protein-to-proteinall* integration, also denoted as *Xq*), the grand mean (*Xsup* from the *protein-to-proteinall* integration, or *Xa*), the statistical weights (*Vinf* or *Vq*), and the respective species (*tags*) for each protein. The grand mean, utilized for log2-ratio normalization, along with the statistical weights, can be employed in constructing plots, as illustrated in Figure 73 (see below).



Figure 72. The REPORT module task table for workflow 4.

It is noteworthy that variance modelling, normalization, standardization and statistical weighting, according to the GIA algorithm, are performed automatically, without data filtering, pre-processing or missing value imputation [6], even in a situation where numerous proteins have highly imbalanced data. Moreover, the sigmoid plots automatically generated in each one of the integrations performed by INTEGRATE clearly demonstrate that the GIA algorithm accurately predicts the distribution of peptide quantifications around their proteins (Figure 73A) and of protein quantifications around the grand mean (Figure 73B). These results demonstrate that this statistical model is very suitable for the analysis of label-free data.

A graph of a function

Description automatically generated

Figure 73. Distribution of the standardized variable at the peptide (Zpq) and protein (Zqa) levels for label-free data analyzed with iSanXoT. A) Zpq distribution for the eight individual A-type and B-type samples. B) Zqa distribution for the B-type vs A-type comparison. Red: null hypothesis (standard distribution); blue: experimental data.

The combined statistics *B\_vs\_A* also shows how human, yeast and bacterial proteins distribute around the expected 0, 1, and -2 log2- values (corresponding to 1-, 2- and 0.25-fold changes) (Figure 74), and how protein quantifications with higher statistical weights are more accurate. This plot also confirms how iSanXoT provides highly accurate quantitative results in a fully automated fashion.

A graph of different colored dots

Description automatically generated

Figure 74. Quantification of human (orange), yeast (grey) and bacterial (blue) proteins according to the combined statistics B\_vs\_A. Shown are log2-ratios normalized by the grand mean (Xq – Xa or Xinf - Xsup). This plot was generated from the table “Npep2prot\_Quanprot”.

We performed calculations to identify proteins exhibiting statistically significant changes between the two preparations. Proteins originating from E. coli and yeast were classified as true positives, while those from *Homo sapiens* were labeled as false positives. Subsequently, we computed key metrics, including the False Positive Rate (FPR), True Positive Rate (TPR), and False Discovery Rate (FDR), based on these classifications.

The same dataset was processed with iSanXoT following two distinct approaches. In the first approach, the integrated results of each sample for each condition (a1, a2, a3, a4 and b1, b2, b3, b4) were considered separately to obtain a standardized value for each protein. Subsequently, a conventional t-test was conducted on all proteins, and the results are presented in *Table S2*.

Table S2. Computation of False Positive Rate (FPR), True Positive Rate (TPR), and False Discovery Rate (FDR) based on statistically significant proteins from the integrated results of samples for each condition provided by iSanXoT and the conventional t-test. The FPR, TPR, and FDR percentages were calculated from the quantified proteins of iSanXoT and the identified proteins of the search engine.



Due to iSanXoT not inputting missing values and the t-test cannot be calculated in cases when there are fewer than two replicates with a quantitative value, the count of quantified proteins was lower compared to the identified proteins (*Table S2*). Nevertheless, iSanXoT achieved a significant True Positive Rate (TPR) for both E. coli and *Saccharomyces cerevisiae* while maintaining the False Positive Rate (FPR) below 5%.

In the second approach, the statistically significant proteins have been used from the comparison of conditions A and B, integrated with the grand mean of human proteins. As detailed above, these results were obtained through the combined use of the NORCOMBINE, RATIO, and INTEGRATE modules.

Table S3. Computation of False Positive Rate (FPR), True Positive Rate (TPR), and False Discovery Rate (FDR) based on significant changes determined by the iSanXoT modules (NORCOMBINE, RATIO, INTEGRATE). The FPR, TPR, and FDR percentages were calculated from the quantified proteins of iSanXoT and the identified proteins of the search engine.



As shown in *Table S3*, iSanXoT was able to quantify more proteins without imputing missing values compared to the t-test, achieving good TPR values while maintaining both FPR and FDR at very reasonable low levels.

# Importing a workflow template

In this section, we will provide instructions to execute the workflow examples and to import workflows that were previously created with iSanXoT to be reused in other projects. We will use the first workflow described in the previous section as an example.

Start by downloading the template for Workflow 1 and the input files from the iSanXoT wiki (<https://github.com/CNIC-Proteomics/iSanXoT/wiki/studies/workflows/WSPP-SBT.zip>). Then, extract the files included in the compressed archive to create a folder named WSPP-SBT. Check that the WSPP-SBT folder has been created in your file system. Proceed as follows:

* Open the iSanXoT application by double-clicking the application icon (*Figure 75*).



Figure 75. The iSanXoT startup message.

* Choose *New Project* from the *Project* menu (*Figure 76*).



Figure 76. Create New Project.

* Provide a name of your choice for the project folder and indicate a path to locate this folder, then click the *Submit* button (*Figure 76*).
* Choose *Import Workflow* from the *Project* menu (*Figure 77)* and select the folder WSPP-SBT created before (or any other iSanXoT project folder from which you want to import the workflow).



Figure 77. Importing a preexisting iSanXoT workflow to the newly-created project.

* Inspect the WSPP-SBT task table (in the Compound modules tab), the RELS CREATOR task table (in the Relation tables tab) and the REPORT task table (in the Reports tab) to check that the tables indicated in Fig. S2, S3 and S5 have been correctly loaded. Note that if a different template is imported, only the corresponding task tables will be loaded.
* Now, click on *Choose identification file* and select “ID-q.tsv” in the WSPP-SBT folder (*Figure 78)*. Alternatively, select the desired identification/quantification table with which this workflow is to be executed. Section 3 below shows how to prepare the “ID-q” file based on the output from a variety of proteomics pipelines. Bear in mind that the tasks defined in the LEVEL CREATOR and RELS CREATOR modules have to match the samples and column names from the specific “ID-q” file used.

A screenshot of a computer

Description automatically generated

Figure 78. Choosing the identification/quantification (ID-q) file for the newly-created project.

* Select *Save Project* from the *Project* menu to save the changes, or directly press the *Save and Run* button to both save and execute the current workflow.

# Creating the identification/quantification file from proteomics pipelines

iSanXoT necessitates an identification/quantification file in TSV format (*ID-q.tsv*), which should include, at a minimum, the quantified features alongside their corresponding quantitative values. Any TSV table can serve as an ID-q file, given that quantitative values are organized with column headers. In this arrangement, features (such as PSMs or peptides) are listed in rows, and their quantitative values are presented in columns, with each column representing a distinct sample. iSanXoT utilizes the column headers of the *ID-q* file to extract essential information.

Moreover, when the *ID-q* file encompasses features quantified in more than one experiment (e.g., different samples labeled with the same TMT-18plex tags), it must include an additional column with the header “Experiment” to specify the experiment assignment of the features.

Finally, iSanXoT requires additional information to generate the relation files necessary for integrating quantified features into higher levels. This information is typically found in the *ID-q* file. For example, iSanXoT can utilize columns containing scan and peptide identifiers to construct the *scan2peptide* relation table.

The majority of proteomics software tools generate tables that can be easily used for this purpose. In this section, we will describe how to prepare the *ID-q* file based on the output from the three most popular proteomics pipelines (*Table S4*).

Table S4. Output data from proteomics pipelines to be included in the ID-q.tsv file.

A table of data

Description automatically generated with medium confidence

## Preparing the *ID-q* file from Proteome Discoverer output

In the case of Proteome Discoverer version 2.5 [9], the way that quantitative data are adapted for use with iSanXoT depends on whether they originate from label-free or labelled experiments:

### Label-free experiments

In this scenario, quantitative data at the peptide level can be adapted for use with iSanXoT from the *\_PeptideGroups.txt* files obtained when the *Processing* workflow node *Minora Feature Detector* of Proteome Discoverer is employed. The following column headers of the *\_PeptideGroups.txt* files must be considered when preparing the *ID-q* file:

* *Sequence*: Amino acid sequence of the identified peptide;
* *Modifications*: Chemical or posttranslational modifications to the *Sequence* above;
* *Master Protein Accessions*: Accession code(s) for the protein(s) to which the peptide *Sequence* is ascribed;
* *Abundance: FX: Sample Type:* Peptide intensity in the RAW file identified with *FX* and tagged as *Sample Type* in the Proteome Discoverer *Input Files* tab.

The *peptide* level required for the *peptide-to-protein* integration with iSanXoT can be obtained by merging the *Sequence* and *Modifications fields* (see Section *Adapting the results from proteomics pipelines for iSanXoT* below).

### Labelled experiments

For labelled experiments (e.g., TMT- or iTRAQ-based), quantitative data at the scan level can be adapted for use with iSanXot from the *\_PSMs.txt* files generated when the *Processing* workflow node *Reporter Ions Quantifier* of Proteome Discoverer is used. The following column headers of the *\_PSMs.txt* files must be considered for preparing the *ID-q* file:

* *Spectrum File*: Name of the RAW file where the PSM was identified;
* *First Scan*: Spectrum (scan) number of the PSM in the RAW file;
* *Sequence*: Amino acid sequence of the identified peptide;
* *Modifications*: Chemical or posttranslational modifications to the *Sequence* above;
* *Master Protein Accessions*: Accession code(s) for the protein(s) to which the peptide *Sequence* is ascribed;
* *Abundance: Quan Channel*: Intensity of the reporter ion tagged as *Quan Channel* in the Proteome Discoverer *Samples* tab.

For the *scan to peptide* integration with iSanXoT, the *scan* level can be obtained by merging the *Spectrum File* and *First* *Scan* fields, and the *peptide* level by merging the *Sequence* and *Modifications* fields (see Section *Adapting the results from proteomics pipelines for iSanXoT* below; make sure *Max. Number of Peptides Reported* = 1 was selected in the *Input Data* section of the Proteome Discoverer *Processing* node used).

## Preparing the *ID-q* file from MaxQuant output

The way MaxQuant version 1.6.5.0 [8] data are adapted for use with iSanXoT depends on whether they originate from label-free or labelled proteomics experiments:

### Label-free experiments

In this case, the quantifications at the peptide level required to prepare the *ID-q* file can be found in the *modificationSpecificPeptides.txt* file, which is stored in the *“…combined/txt*” folder. The following column headers of the *modificationSpecificPeptides.txt* file must be considered for preparing the *ID-q* file:

* *Sequence*: Amino acid sequence of the identified peptide;
* *Modifications*: Chemical or posttranslational modifications to the *Sequence* above;
* *Proteins*: Identifier(s) of the protein(s) to which the peptide *Sequence* is ascribed;
* *Intensity Experiment*: Summed up extracted ion current of all isotopic clusters associated with the peptide Sequence identified across the raw files included in the *Experiment* as specified by the user in the MaxQuant *Raw data* tab.

The *peptide* level required for the *peptide* to protein integration with iSanXoT can be obtained by merging *Sequence* and *Modifications* fields (see Section *Adapting the results from proteomics pipelines for iSanXoT* below).

### Labelled experiments

When dealing with labelled experiments (e.g., iTRAQ- or TMT-based), the necessary quantitative data at the scan level can be found in the *modificationSpecificPeptides.txt* file, which is stored in the *“…combined/txt*” folder. The following column headers of the *modificationSpecificPeptides.txt* file must be considered for preparing the *ID-q* file:

* *Raw file*: Name of the RAW file where the PSM was identified;
* *Scan number*: Spectrum (scan) number of the PSM in the RAW file;
* *Modified Sequence*: Amino acid sequence of the identified peptide including chemical or posttranslational modifications. This parameter is nonblank only when identification was successful.
* *Proteins*: Identifier(s) of the protein(s) to which the peptide *Sequence* is ascribed;
* *Reporter intensity n*: Intensity of the reporter ion *n* as specified by the user in the MaxQuant *Group-specific parameters* tab.

For the *scan to peptide* integration with iSanXoT, the *scan* level can be obtained by merging the *Raw File* and *Scan number* fields (see Section *Adapting the results from proteomics pipelines for iSanXoT* below).

## Preparing the *ID-q* file from FragPipe output

The way that quantitative data from Fragpipe version 1.8.1 [10] are adapted for use with iSanXoT depends on whether they originate from label-free or labelled experiments:

### Label-free experiments

FragPipe *Quant (MS1)* module stores the quantifications at the peptide level necessary to prepare the *ID-q* file in a *combined\_modified\_peptide.tsv* file. The following column headers of the *modificationSpecificPeptides.txt* file must be considered for preparing the *ID-q* file:

* *Modified Sequence*: Amino acid sequence of the identified peptide;
* *Protein ID*: Identifier of the protein to which the *Modified* *Sequence* peptide is ascribed;
* *Experiment Intensity*: Summed up intensity of the *Modified Sequence* peptide in the RAW files included in the *Experiment* as specified by the user in the FragPipe *Workflow* tab.

### Labelled experiments

Fragpipe *Quant (Isobaric)* module generates a *psm.tsv* output file that contains the quantitative data at the scan level obtained from labelled experiments. The following column headers of the *psm.txt* file must be considered for preparing the *ID-q* file:

* *Spectrum*: Spectrum (scan) identifier of the PSM in the XML file;
* *Spectrum File*: Name of the XML file where the PSM was identified;
* *Modified Peptide*: Amino acid sequence of the identified peptide including chemical or posttranslational modifications;
* *Protein ID:* Identifier of the protein to which the *Modified Peptide* is ascribed;
* *Channel*: Intensity of the reporter ion *Channel* as specified by the user in the FragPipe TMT-Integrator table of the *Quant (Isobaric)* module.

The *scan* level required for the later *scan to peptide* integration with iSanXoT can be obtained by merging the *Spectrum* and *Spectrum File* fields (see Section *Adapting the results from proteomics pipelines for iSanXoT* below; make sure *Report top N = 1* was selected in the *Advanced Output Options* of the FragPipe *MSFragger* module).

# Adapting the results from proteomics pipelines for iSanXoT

iSanXoT requires an identification/quantification tab-separated values file (*ID-q.tsv*) containing at least the identified features along with their quantitative values (an experiment identifier is required if two or more experiments are included). Users can either manually compose this *ID-q* file (refer to the previous section for guidance on how to do this using data from the four most popular proteomics pipelines) or have it prepared by the iSanXoT Input Adaptor. The latter option is described in this section.

* Run the iSanXoT application and create a new project or open an existing project. A new window will appear asking for the ID-q file (*Figure 79*).
* If you already have a suitable *ID-q* file, click the *Select User-Provided* option, and then *Choose identification file* to select the file (*Figure 79*).

**A screenshot of a computer

Description automatically generated**

Figure 79. Selecting an ID-q file in the Input Adaptor main window.

* If you do not have an *ID-q* file, click *Select Adaptor from proteomics pipelines*. This option will launch the iSanXoT adapter to import your quantitative data. The adapter has been tested with recent versions of MaxQuant, Trans-Proteomic Pipeline, FragPipe, and Proteome Discoverer. Click on *Choose folder + Add anots* to select the folder containing your quantitative data (*Figure 80*). A three-panel window will pop-up (*Figure 81)*.



Figure 80. Having the iSanXoT Input Adaptor prepare the ID-q file.



Figure 81. Adapting results from a proteomics pipeline. In the top panel, several output files from Proteome Discoverer have been selected. These PSMs.txt files, which contain identification/quantification data, have been assigned an experiment name (Jurkat) in the middle panel. The bottom panel has been used to create identifiers by concatenating result file headers: ScanID (by concatenating Spectrum File, First Scan and Charge) and pepID (by concatenating Sequence and Modifications).

* The top panel displays the files included in the folder, allowing you to select one or more result files for consideration by the adapter. It's important to note that if several result files are chosen, they must have the same column headers.
* The middle panel is used to set the distribution of data items across experiments according to result filenames.
* The bottom panel allows to create identifiers by concatenating result file headers. It is composed of two interfaces:
  + The left side lists the headers found in the result files, with selected header names being added to the right-side interface.
  + The right-side interface displays the selected header names used to generate the identifier, along with the user-provided identifier name.
* Please note that the alphanumeric text that unambiguously identifies the items to be integrated is the only identifier that must be necessarily included in the *ID-q* file.
* Click the *Submit* button and the Input Adaptor will start generating the *ID-q.tsv* file.

# References

[1] Trevisan-Herraz M, Bagwan N, Garcia-Marques F, Rodriguez JM, Jorge I, et al. SanXoT: a modular and versatile package for the quantitative analysis of high-throughput proteomics experiments. Bioinformatics. 2019;35(9):1594-6.

[2] Navarro P, Trevisan-Herraz M, Bonzon-Kulichenko E, Nunez E, Martinez-Acedo P, et al. General statistical framework for quantitative proteomics by stable isotope labeling. J Proteome Res. 2014;13(3):1234-47.

[3] Garcia-Marques F, Trevisan-Herraz M, Martinez-Martinez S, Camafeita E, Jorge I, et al. A Novel Systems-Biology Algorithm for the Analysis of Coordinated Protein Responses Using Quantitative Proteomics. Mol Cell Proteomics. 2016;15(5):1740-60.

[4] Bonzon-Kulichenko E, Camafeita E, Lopez JA, Gomez-Serrano M, Jorge I, et al. Improved integrative analysis of the thiol redox proteome using filter-aided sample preparation. J Proteomics. 2020;214:103624.

[5] Gonzalez-Amor M, Garcia-Redondo AB, Jorge I, Zalba G, Becares M, et al. Interferon stimulated gene 15 pathway is a novel mediator of endothelial dysfunction and aneurysms development in angiotensin II infused mice through increased oxidative stress. Cardiovasc Res. 2021.

[6] Navarro P, Kuharev J, Gillet LC, Bernhardt OM, MacLean B, et al. A multicenter study benchmarks software tools for label-free proteome quantification. Nat Biotechnol. 2016;34(11):1130-6.

[7] Bagwan N, Bonzon-Kulichenko E, Calvo E, Lechuga-Vieco AV, Michalakopoulos S, et al. Comprehensive Quantification of the Modified Proteome Reveals Oxidative Heart Damage in Mitochondrial Heteroplasmy. Cell Rep. 2018;23(12):3685-97 e4.

[8] Tyanova S, Temu T, Cox J. The MaxQuant computational platform for mass spectrometry-based shotgun proteomics. Nat Protoc. 2016;11(12):2301-19.

[9] Orsburn BC. Proteome Discoverer-A Community Enhanced Data Processing Suite for Protein Informatics. Proteomes. 2021;9(1).

[10] Kong AT, Leprevost FV, Avtonomov DM, Mellacheruvu D, Nesvizhskii AI. MSFragger: ultrafast and comprehensive peptide identification in mass spectrometry-based proteomics. Nat Methods. 2017;14(5):513-20.