

# APP: an Automated Proteomics Pipeline for the analysis of mass spectrometry data based on multiple open access tools

Erik Malm

## Contents

<b>1</b>	<b>Project info</b>	<b>1</b>
<b>2</b>	<b>An introduction</b>	<b>1</b>
<b>3</b>	<b>Tasks</b>	<b>1</b>
<b>4</b>	<b>The server</b>	<b>2</b>
<b>5</b>	<b>The interface</b>	<b>3</b>
<b>6</b>	<b>Installation and use</b>	<b>3</b>
6.1	Installation . . . . .	3
6.1.1	Client/Server installation on Windows . . . . .	4
6.1.2	Client/Server installation on Ubuntu 12.04/14.04 . . . . .	5
6.1.3	Initial server/client setup using the GUI . . . . .	5
6.1.4	Connecting an additional worker using the GUI . . . . .	9
6.2	Configure on the command line . . . . .	13
6.3	User accounts and access keys . . . . .	14
6.3.1	Activate user accounts . . . . .	14
6.4	Add a client access key . . . . .	15
6.5	Test dataset search . . . . .	16
6.5.1	Interface installation Windows/Linux . . . . .	16
<b>7</b>	<b>Tutorials and sample datasets</b>	<b>16</b>
7.0.2	Starting up . . . . .	16
7.0.3	Build a multiple search engine task . . . . .	17
7.0.4	Monitor execution: . . . . .	27
7.0.5	Work with output from previous task: . . . . .	30
<b>8</b>	<b>Available Plugins</b>	<b>33</b>
8.1	General plugin function . . . . .	33
8.2	Raw data conversion plugins . . . . .	33
8.2.1	MSconvert plugin . . . . .	34
8.2.2	mzXML2Other . . . . .	36
8.2.3	PKL to MGF . . . . .	36
8.3	Database search engine plugins . . . . .	36
8.3.1	General Search Settings plugin . . . . .	36

8.3.2	X!Tandem . . . . .	38
8.3.3	Myrimatch . . . . .	39
8.4	Comet . . . . .	39
8.5	MSGFPlus . . . . .	40
8.5.1	InsPecT plugin . . . . .	42
8.5.2	OMSSA . . . . .	42
8.6	Spectral search engine plugins . . . . .	42
8.6.1	SpectraST library builder plugin . . . . .	43
8.6.2	SpectraST search plugin . . . . .	44
8.7	Data processing . . . . .	45
8.7.1	PeptideProphet plugin . . . . .	45
8.7.2	iProphet plugin . . . . .	46
8.8	Protein prophet plugin . . . . .	46
8.9	Smaller utilities . . . . .	46
8.9.1	Spectrum name fixer plugin summary . . . . .	46
8.9.2	Label Free Data Extractor . . . . .	47
8.9.3	LibraProteinRatioParser . . . . .	49
8.9.4	Spectractor . . . . .	49
8.9.5	Libra Normalizer . . . . .	49
8.9.6	Input file feeder . . . . .	50
8.9.7	Mascot2XML plugin . . . . .	50
8.9.8	IDConvert . . . . .	50

## 1 Project info

- Project homepage: <https://sourceforge.net/projects/automatedproteo>
- Discussion group: For support questions please contact:

## 2 An introduction

Automated Proteomics Pipeline (APP) is a working name given to our efforts to gather a large amount of proteomics functionality under one common interface. It builds on the work of many in the proteomics community to offer up an integrated user interface and distributed server infrastructure without many of the common difficulties of such setups. In a way it's a simplified grid computing implementation that is perfectly happy to run on whatever systems are available and focuses on wrapping the functionalities of external software. Our original aim when building APP was to provide simplified infrastructure for many of our own complex workflows, and to remove from the user the consideration of where parts of the task are executed. APP is open source under the GPL license.

Each implemented APP function is provided as a plugin, these plugins provide their own user interfaces and execution methodologies. A number of such plugins are then linked to provide an end-to-end processing workflow for the data. Common among the plugins is that they pass result files onward to plugins further down the line (though there are some exceptions, such as where a file is modified in place). The tasks are then submitted to an APP server which will handle organized execution of the task. Results are stored and accessible on the APP server. The server component can be in communication with other nodes (computers, virtual machines, or any other piece of hardware running java 7) on the network and will portion out tasks to these nodes for execution. If a task is successfully completed the results will be collected, if a task fails it will be attempted on other nodes. Nodes with a high success rate will be prioritized for tasks if several are available. Each node can have its own

set of plugins allowing machines specialized for specific tasks to be set up, such as a Windows machine using Vendor library enabled MsConvert in a network of Linux machines that perform database searches. The server can be configured and started either via a command-line interface or through a rich GUI. Configuration files are stored in XML formatted in a human readable fashion and simple text files. These can be directly edited or accessed through the interface for an easier setup.

### 3 Tasks

Anything worth doing in proteomics usually involves more than a few steps. In the simplest scenarios where one merely wants to analyze a specific gel-band this still involves several steps following MS analysis. Typically a work-flow will need to process mass-spectrometer raw-data and pick out features from MS scans, convert the data into a format that MS search engines can handle and then search the data using one or several database search engines. Any output is generally post-processed either manually or in an automated fashion to give a probability for all protein identifications. APP uses this model of the flow of data at its core, since tasks are defined as a network of plugins where each will push the resulting data into the next plugin. In the example given, data would be processed and converted using the MsConvert plugin (preferably running on Windows), searched using any combination of InsPecT, OMSSA, Comet, X!Tandem, Myrimatch and MSGFPlus search engines and the output would be analyzed with PeptideProphet, iProphet and finally ProteinProphet plugins. APP also automates many of the small steps to keep these different applications interoperable. Search settings for all of the mentioned search engines are handled from a single plugin, which produces a general settings file for feeding into all APP database search plugins where it is interpreted into search engine specific format. Functionality that is unique to any single search engine, such as X!tandem's ability to use a second refined set of search criteria, can be accessed through each individual plugins' settings. All plugin default settings aim to be ready to go though, so for general usage there should be no need to tweak. Other more housekeeping oriented functions are also automated, for example, MyriMatch, MSGFPlus and OMSSA will renumber spectra in their output. To keep analysis consistent spectrum references need to be corrected. This functionality is provided through a SpectrumFixer plugin. There are a number of templates available, and complex tasks can be reused to provide templates for future searches. There are examples of more complex work-flows in the tutorial section.

### 4 The server

The server handles execution of tasks and maintains a record of all tasks and their output. The server also communicates with users through an interface and also with other computers running either as a semi-server, referred to as a worker, that executes different parts of the task. The server feeds the different nodes information and files needed for a task automatically so no data-files need to be directly made available to the workers, though shared storage can also be used. See figure 1 for a schematic, or see table 6.1 for an overview of APP site content.

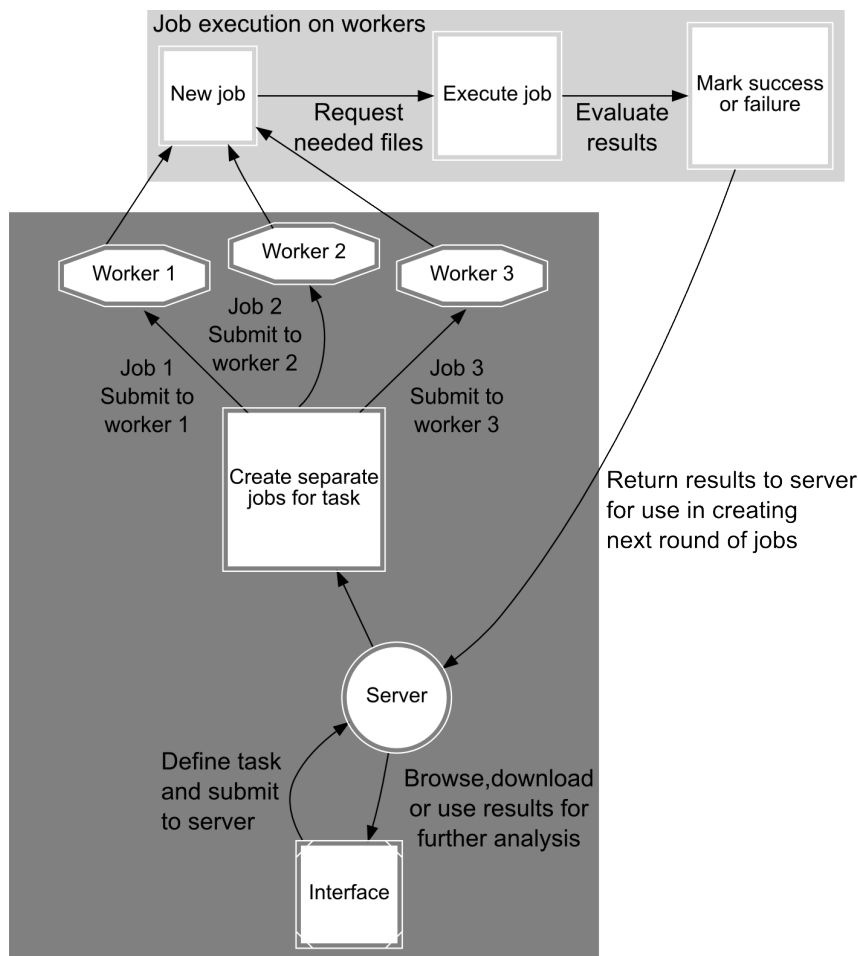


Figure 1: A schematic server setup showing three workers processing a job provided by the interface

## 5 The interface

Figure 1 shows a schematic server-worker structure. The APP interface is where all user interaction will take place, the interface allows users to build a proteomics task, mixing files stored on a local computer with files stored on the server and monitor the execution of any tasks currently running on the server. Please see the tutorial section for using the interface together with our provided tutorial data.

## 6 Installation and use

### 6.1 Installation

Installation differs slightly depending on if you wish to run a full server or merely add another processing node to an already running server. The default .zip bundles contain all files needed to add a database search engine node and will include executables for MSGFPlus, Myrimatch, X!Tandem, OMSSA and Comet. Included is also the open source components of proteowizard. For instructions on installation of the GUI see 6.5.1 and for usage examples see 7.0.2

Project files	
File-name	Description
APPWinBundle32.zip	Contains all files needed for a quick client setup on Windows. Also get as starting point for full <b>server</b> setup on Windows.
APPUbuntu14.04Bundle.zip	Contains all needed components for a full server or client install on linux.
APPInterface.zip	Packages the interface package of APP. This is also included in the Windows and linux bundles.
APPTutorialData.zip	Contains a set of test data for search in .mgf format.
APPSource.zip	Contains all the sources for APPs interface, APPs server components and all plugins. Also contains a list of needed dependencies.

Table 1: Listing of files available on the APP sourceforge page and their uses.

### 6.1.1 Client/Server installation on Windows

On Windows the easiest setup process involves three steps:

- **Client setup:** Download the **APPWindows32Bundle.zip** from our SourceForge page (<https://sourceforge.net/projects/automatedproteo>). This contains all plugins needed for standard database search and spectral search. While you can unzip it anywhere some software automated through APP has problems with folder-names containing spaces. For this reason it would be best to unzip in a folder other than your desktop (such as C:\APPServer), since for example Comet will fail if there is a space anywhere in the filepath. Once unzipped, double click APPServer.exe in the APPServer directory. If your current JVM is not at least 1.7.0 you will be directed to a download page. Please note that Oracle bundle Java with ASK toolbar, make sure to uncheck the selection for this during installation since it is unrelated bloatware.
- To run a full **APP server** it is best to also install the Trans Proteomic Pipeline.
- First download and install ActivePerl Community edition from <http://www.activestate.com/activeperl>. Download the appropriate (either 32 or 64 bit) version, and double click the downloaded .msi file to install.
- Download and install the latest TPP version from the Sashimi project page at <http://sourceforge.net/projects/sashimi/>.  
Note that SourceForge has been known to bundle bloatware inside installers, though this does not seem to have affected TPP so far users should pay attention to any check-boxes to avoid installing ad-ware.
- Start the server by double clicking APPServer.exe (if Windows hides the extension it should say "Application" as file-type). Now, your server should either start up or you will be taken to the correct webpage for downloading Java. In the second case, download and install Java (note that on Windows Java is also bundled with bloatware, say no to any other suggested programs) and then execute APPServer.exe again.
- See section 6.1.3 for further instructions, you now have the brain of your distributed computing network.
- *Note:* ProteoWizard (idconvert/msconvert) needs a .net installation to run properly. Most likely you have this already, but if you run into problems please reinstall .net from <http://www.microsoft.com>

### 6.1.2 Client/Server installation on Ubuntu 12.04/14.04

Download APPUbuntu14.04Bundled.zip from the sourceforge page, unpack the zip file. Using a terminal, invoke "sudo ./UbuntuInstaller.sh -install". Note that this will install a web-server, all linux compatible search engines, TPP4.7.1 rev 0 (will be updated each revision) and open port 80 and 1466 in iptables. You will be asked to confirm each of these steps at the command prompt.

To start the APP server tool, type APPServer at the terminal. To start the APP interface, type APPInterface at the terminal. TPP and APP directories can all be located under /usr/local/tp. To make APPServer auto start at boot-time, type "sudo update-rc.d APPServer defaults" at the terminal. If you have set APP up as an auto connection client node you can make the connection auto start when the computer turns on by running "sudo update-rc.d APPClient defaults".

### 6.1.3 Initial server/client setup using the GUI

The first thing that will greet you upon starting the server GUI is a blank status window (see Fig. 2). By default the outputs of APP are not sent here, but can be made to do so.

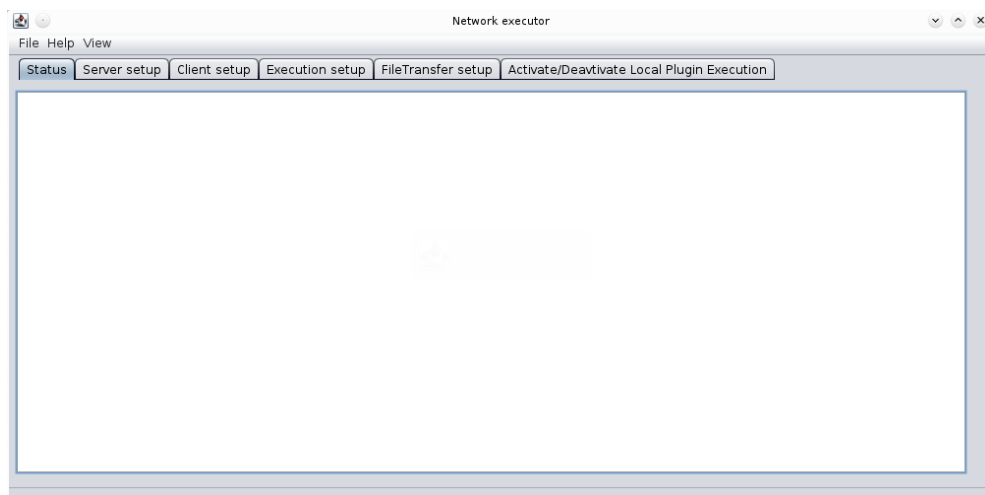


Figure 2: A blank server GUI

For now click on the "Execution setup" tab to set some standard options (see Fig.3). There are a number of options available here, most should be left alone.

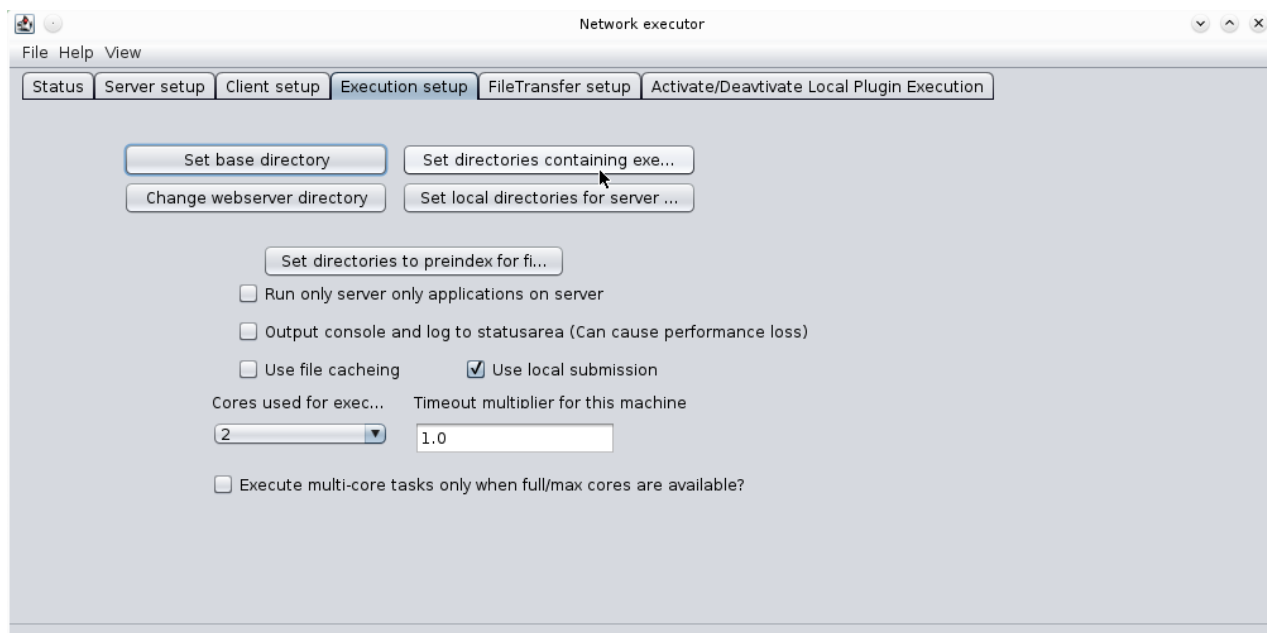


Figure 3: The execution setup tab

The options are:

- *Set base directory* - Sets APPs base directory, temporary files, tasks and other things will be stored under this directory. By default it is the directory containing APP but you can put this anywhere, such as on a drive with more space. On linux you should be aware of the folders permissions since the directory needs to be both readable and writeable.
- *Change webservice directory* - This directory is referenced by APP as the base folder for TPP derived web utilities, such as spectrum viewers and results viewers. If you installed using standard paths you can leave it alone, but this can also be set manually here.
- *Set directories containing executables* - The executable folders will be checked by APP for any needed binaries (such as search engines, data conversion utilities etc). Default should be fine but binaries can be stored anywhere, this should be set here.
- *Set local directories for server to index* - Add folders here that contain files that should be accessible to all users, examples of this are fasta database files (for use with search engines) or directories containing raw data from the mass spectrometer. These folders will be monitored for the addition of more files, which will then be presented to the interface when queried. For example, APP comes with a set of standard databases but will also display any other .fasta or .fa databases in the settings interface, if they are placed in a subfolder called dbases. Any other files will be available if you click "Browse files on server", allowing use of files on the server rather than the users desktop.
- *Set directories to preindex for files* - On a client commonly used files can be stored in pre-specified directories preventing the need to transfer databases or other such files. Any directory provided here will be indexed and files with a matching size and name to any transferred file will instead be retrieved from the local directory. Can also be used to map shared storage.
- *Run only "server only" applications on server* - Only relevant for the server, some plugins are tagged to not be distributed to clients. Examples of this is PeptideProphet, iProphet and ProteinProphet where all

files referenced in pep.xml files are expected to be accessible to the program. If this setting is checked, the server will execute ONLY such programs which can prevent long running searches from blocking relatively quick validation steps. **Leave this unchecked unless you plan to attach one or more clients to the server.**

- *Output console and log to status-area* - if running in GUI mode the standard output and logging operations of APP can be redirected to show in the "Status" tab. Check this box temporarily if you want to find out what is happening in the guts of the program.
- *Use file caching* - Only relevant to APP running as "**Client**". Will cache transferred files. This means that repeatedly transferred files such as databases or raw data files will not be deleted immediately but stored until a predetermined size limit is reached, preventing the need to transfer files multiple times. This should be enabled on most clients. Files below 5 megabytes are never cached.
- *Timeout multiplier* - If a client or server is slow it might need extra time to complete the jobs given to it. The timeout multiplier allows one to increase or decrease the total time available to each job between status updates. Setting the multiplier to 2 would give each job twice as long to run before a timeout and setting it to 0.5 would allow only half of the standard values.
- *Cores used for execution* - Determines the amount of cores that APP can access. On clients all assigned cores will be available to jobs, while one core will be reserved for the administrative functions on the server.
- *Execute multi-core tasks only when full/max cores are available?* - by default APP starts any available jobs as soon as at least one core is available. This option will prevent this and multi-core jobs will instead be prevented from starting until the maximum amount of cores they can utilize (or the max available to APP) are ready for use. Leave this alone unless you have reason to.

For now, if you have your executables in a non-standard location these should be added to the "Executable folders" list. Click on the "Set directories containing executables" button and then drag and drop any folders that contain programs useful to APP onto the window (Fig. 4)



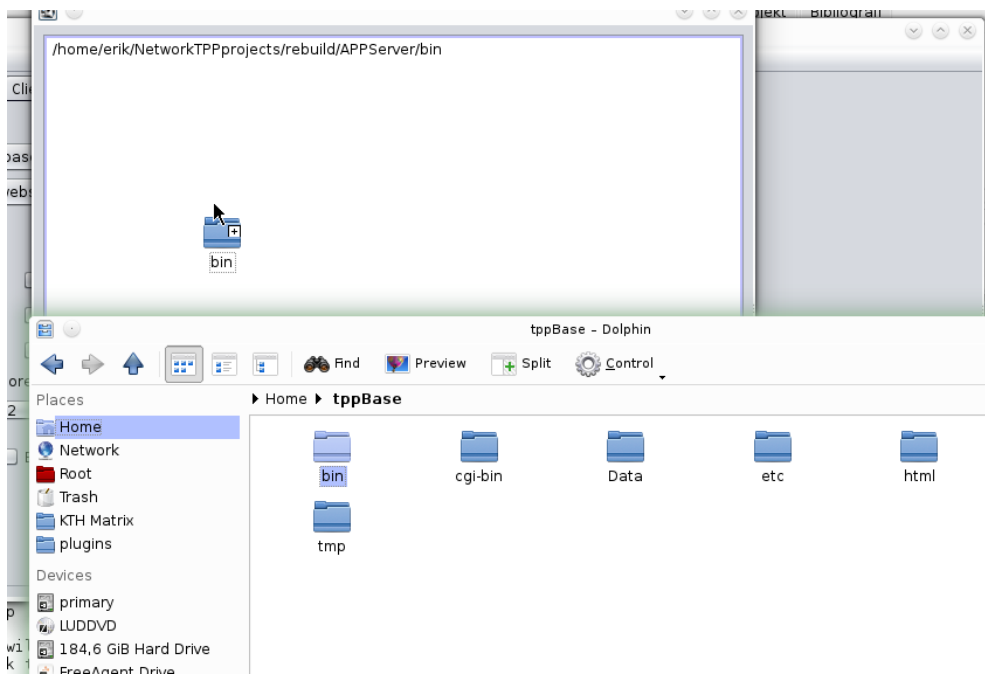


Figure 4: Drag and drop folders into the selection dialog to add executable folders or indexed folders

Do the same for any folders where you want the server to index files. Examples of folders to index are directories containing raw data or fasta databases. This will make them show up under the "Browse compatible files on server" when adding input to plugins later on. If you want databases to be shown in the "General search settings" plugin, put them in a subfolder called "dbase". Switch to the "Activate/Deactivate Plugin Execution", here you can decide which plugins should be executed by a client or server (see Fig.5).

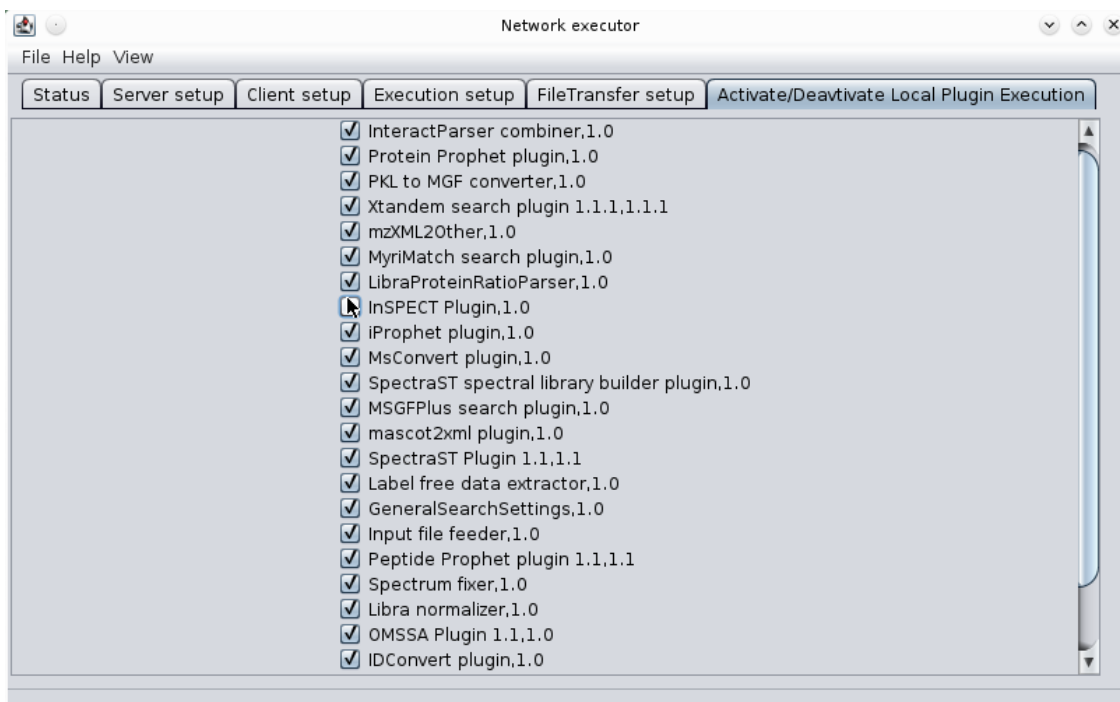


Figure 5: Activate/Deactivate plugin execution

Note that only the Windows version of MSconvert supports vendor formats (such as Waters/Thermo .raw files and Agilent .d files) and these will can not be converted on Linux. A suggested setup would be to allow linux computers to only run the Msconvert Open plugin, which disallows input of unsupported vendor data formats.

Usually all of this setup is optional, but available for tweaking. To start the server click the "Server setup" (see Fig. 6) tab and enter a name for your server, then click start server. It might take up to 3 minutes for a server to be detectable over the network but you should be able to connect almost instantly if you know the IP address of your server. You should now move on to the tutorial section (7.0.2) and process some data.

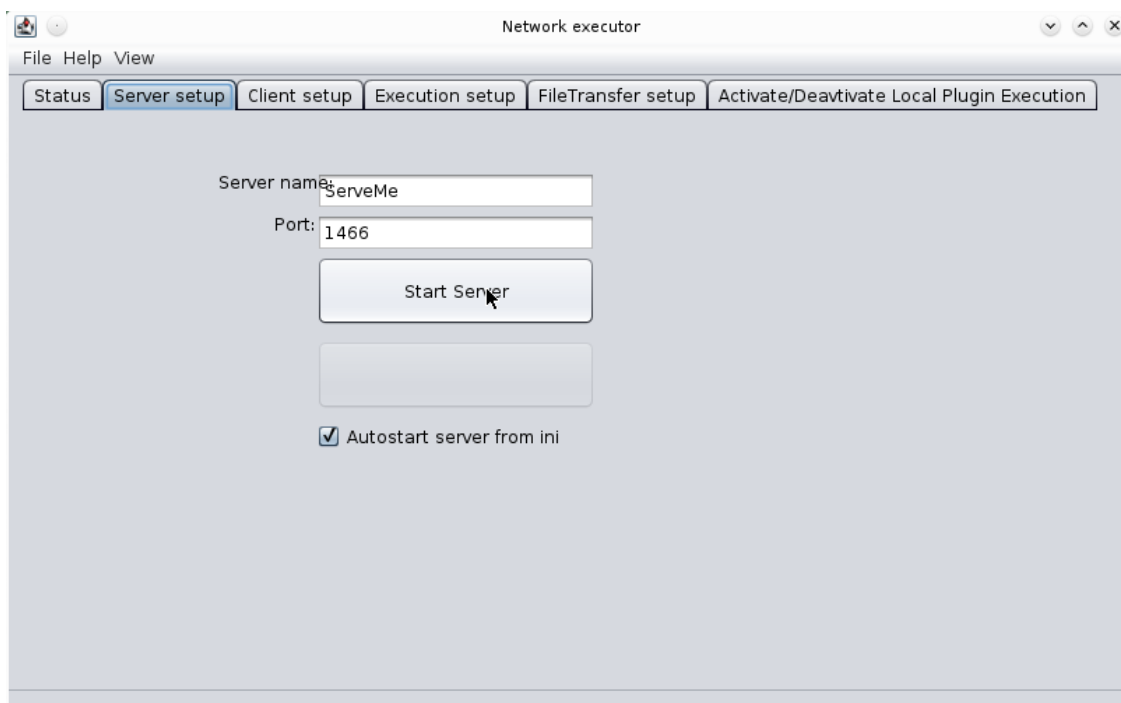


Figure 6: Starting the server

#### 6.1.4 Connecting an additional worker using the GUI

Connecting one or several additional worker nodes is relatively easy. A First run a client installation on the OS of your choice (for Windows this just means unzipping the distribution file,see Fig. 7). Once again start the program by using APPServer.exe or APPServer.jar.

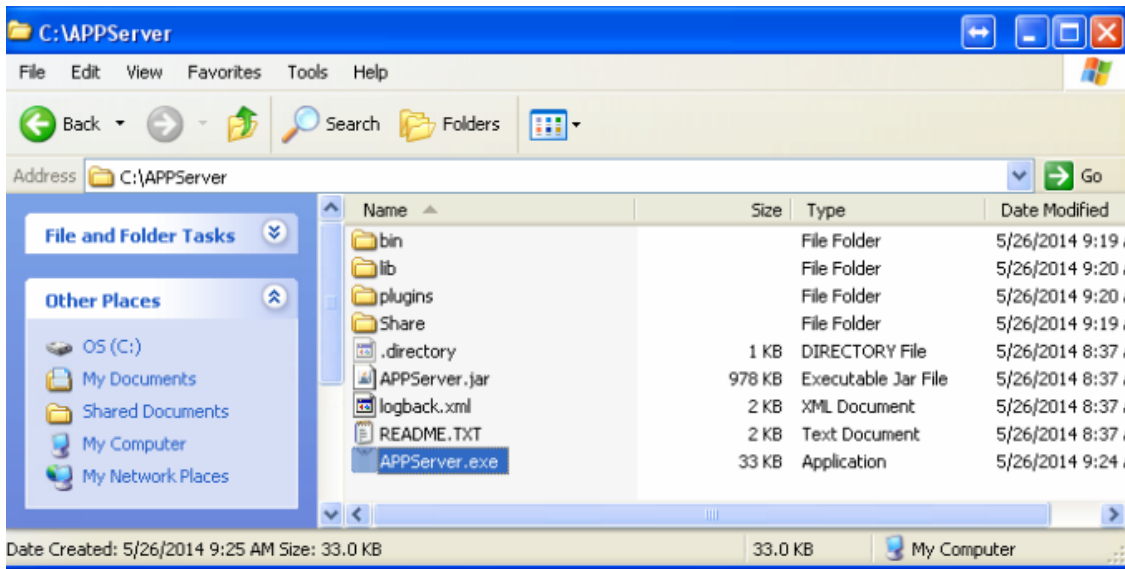


Figure 7: The unzipped client folder

When the graphical user interface opens you might first want to look at the "Execution setup" tab, for details see section 6.1.3. Once this is done click the "Client setup" tab (see Fig. 8).

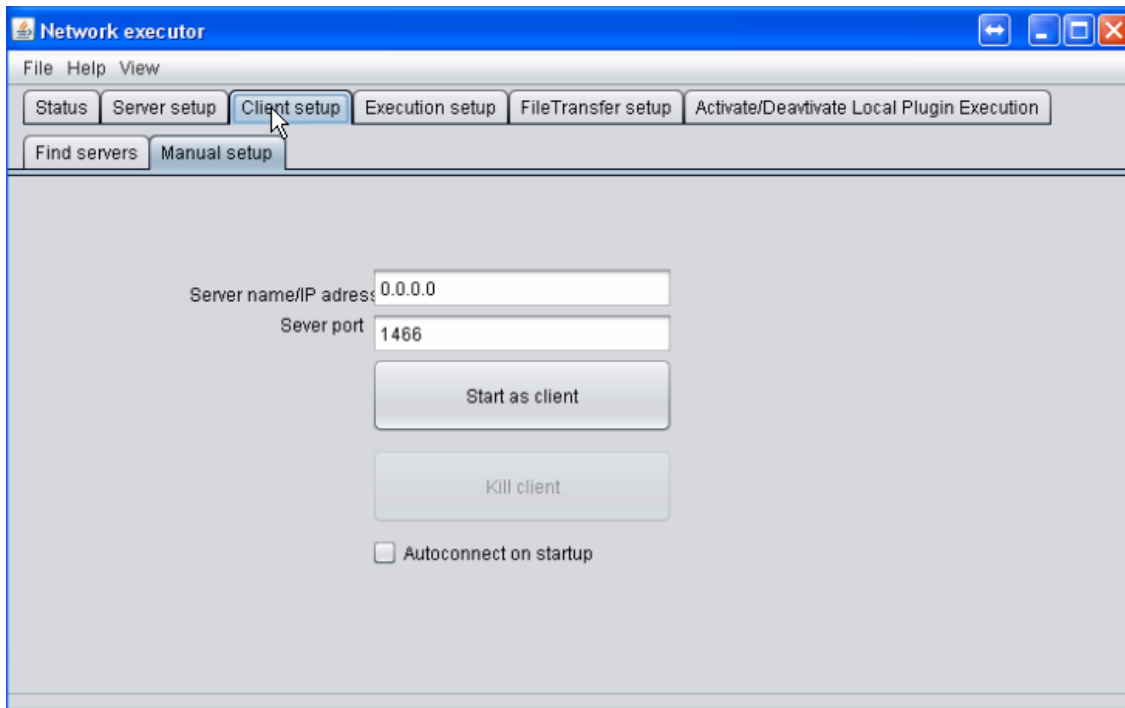
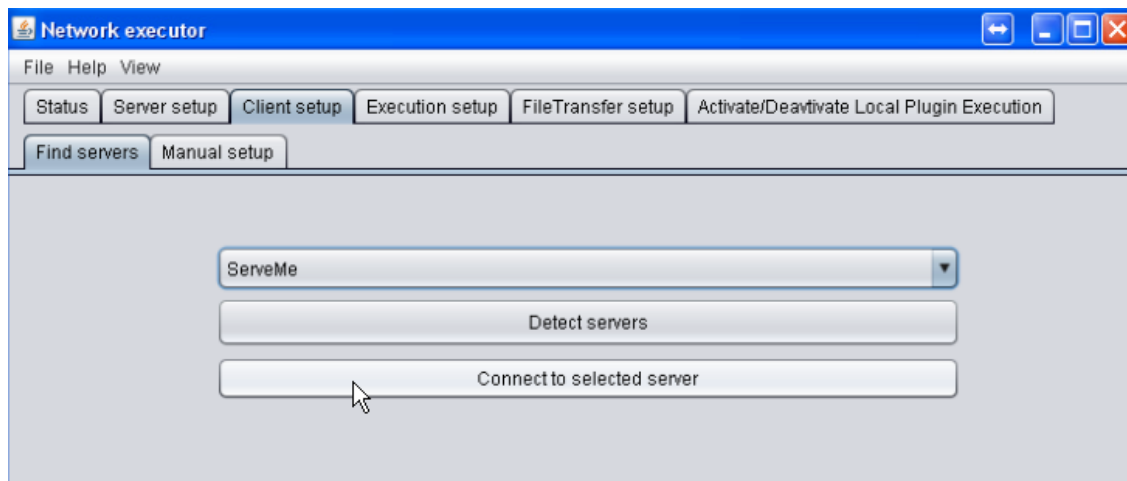


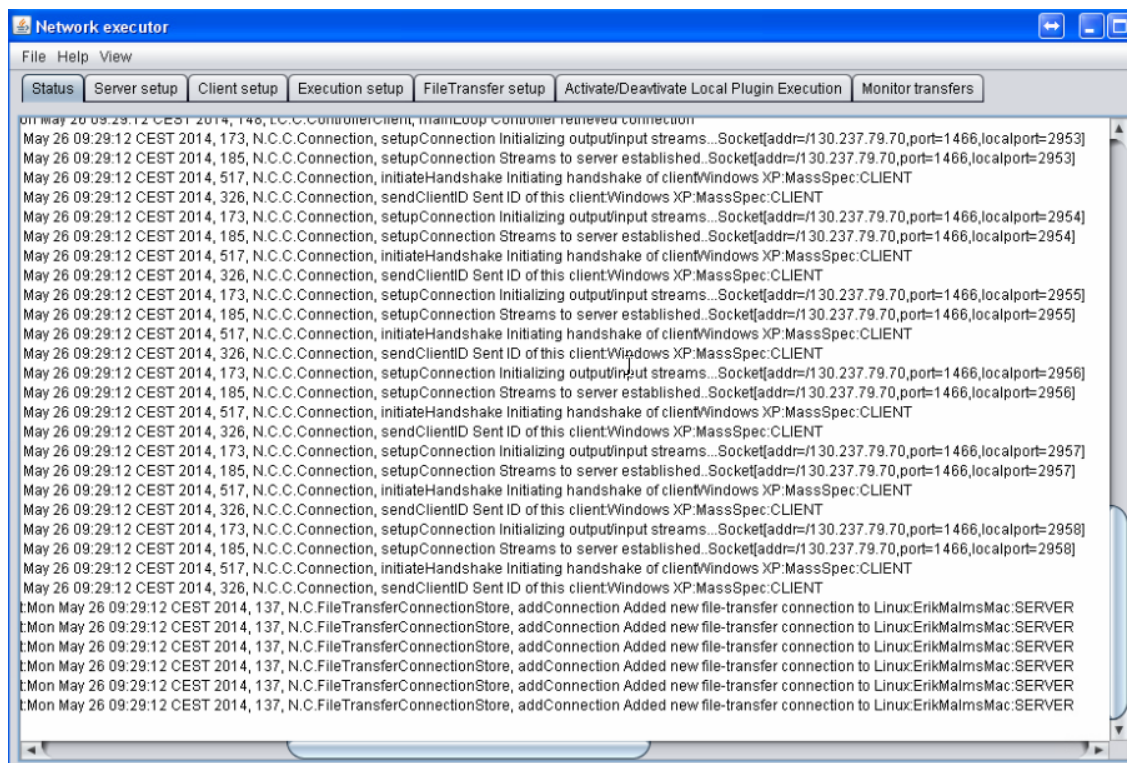
Figure 8: The client setup tab

This will show the manual setup tab. If you want to you can manually add the URL or IP-number of the server here. If you want this worker to auto-connect to your server, click the "Autoconnect on startup" checkbox. Once done, click the "Find server" tab.



Here you can either click the "Detect server" button, once the server has been detected you can click connect (see Fig. 9). If you entered the server URL on the manual setup tab you can click connect without detecting.

Your worker node is now connected to your server, note that if your server is not available the worker node will keep trying until a server is found. Forever. This means that if the server goes offline the worker clients will auto reconnect. If you clicked the "Output console and log to statusarea" option during "Execution setup" you can double check the connection progress by clicking the "Status" tab (Fig. 10).



A better way to check if your client is connected is via an APP interface (see section 7.0.2). Click the "Server actions" menu at the top of the window, and then the "Get server report" button, you will receive a list of

connected workers and their current workload (Fig. 11).

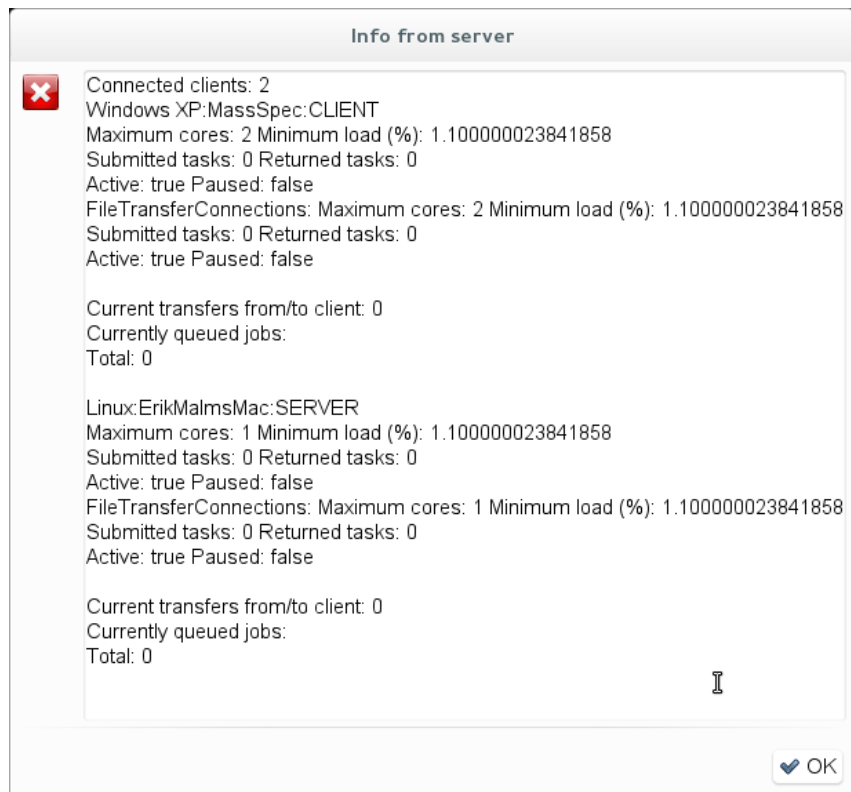


Figure 11: Server report received by interface

A final way to see which workers are currently connected is to use the connected clients dialog of the APPServer by toggling it's visibility in the "View" menu, see figure 12.

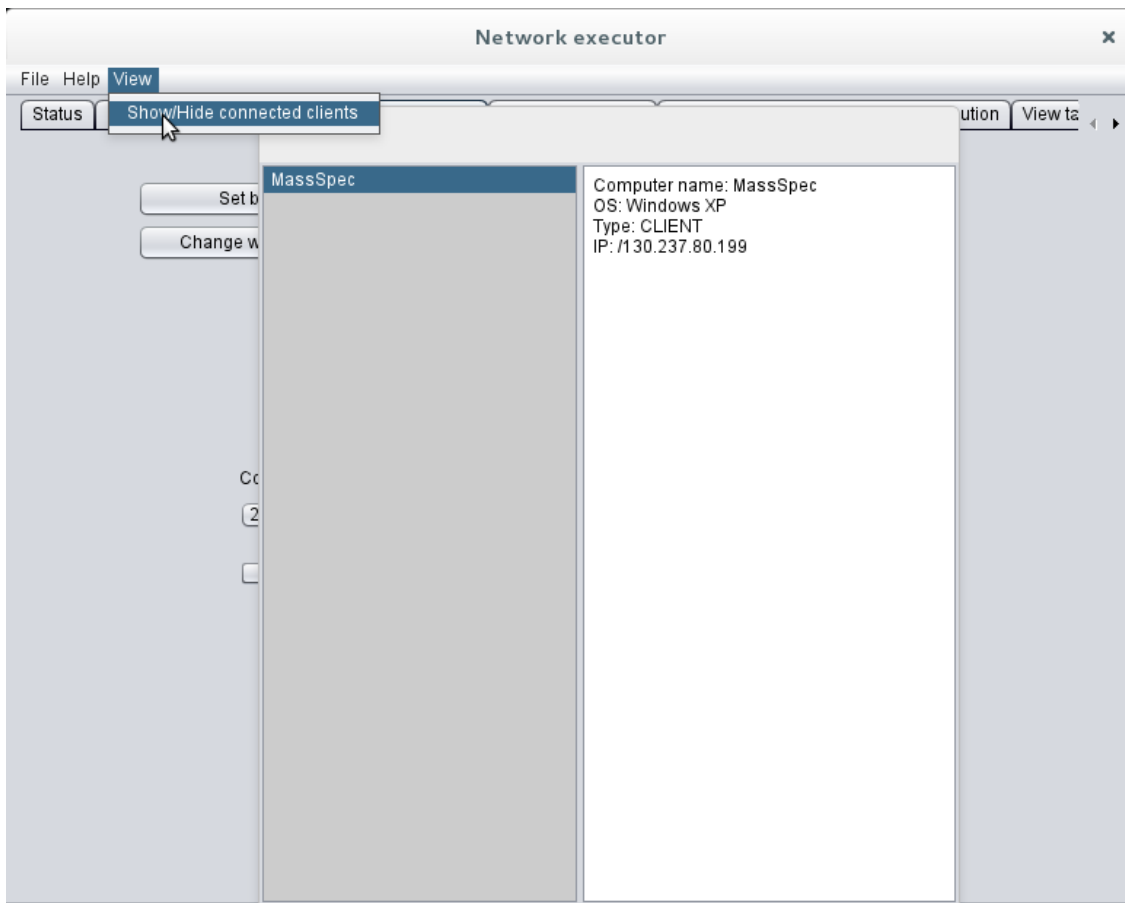


Figure 12: Connected clients dialog of server running with a GUI

APP is routinely tested with up to a hundred such nodes connected, and standard heavy usage is typically handled by around 30 clients connected to a server. However, for most tasks (depending on the power of the nodes in question) a few nodes is more than sufficient.

## 6.2 Configure on the command line

If you want to configure a server without using the GUI the corresponding settings are stored in the following files:

- Execution setup/Filetransfer setup: Setup.xml
- Client setup: client.ini
- Server setup: server.ini
- Activate/Deactivate plugins: Add a string of plugin name + version to the file blockedPlugins.ini in the base dir of the APP server component. A list of strings with plugins and versions can be retrieved by launching APPServer.jar with the -l option (or -help for more info) i.e. "java -jar APPServer.jar -l".

## 6.3 User accounts and access keys

### 6.3.1 Activate user accounts

By default APP does not protect access, small deployments are expected to be started when needed and killed when not. For sensitive or more permanent deployments, it is possible to restrict access to the server by activating user management under the execution setup tab, see fig 13. Two kinds of user accounts are provided by APP, administrators can add and remove users on the server, see and reset, delete and open any tasks. Non-administrator users can see, open reset and delete tasks they started. Each user also belongs to a group and members of a group can see and open each others task, but not reset delete etc.

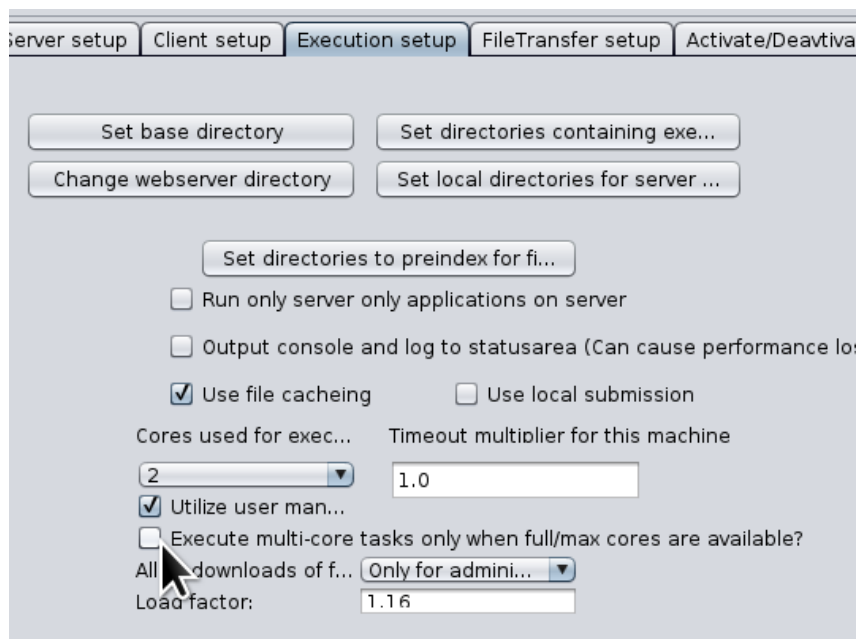


Figure 13: Enable user management

When first enabled a default user with the username and password of "root" is added. It is recommended to remove this default user after creating a new user with administrator privileges. This can be done in the user management tab, see fig. ???. Add a new user by clicking the "Add user" button and enter a new username and password. Then click the administrator checkbox to grant administrator rights. After this click "Apply changes" and when prompted to enter "root" "root" as username and password. After this feel free to delete the root user to prevent access.

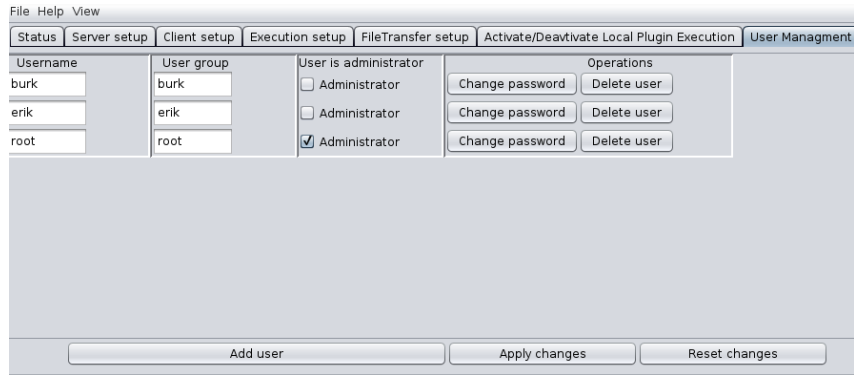


Figure 14: Enable user managment

## 6.4 Add a client access key

Computational nodes are not covered under the user accounts, however APP Servers can be made to only accept connections from clients with the same access key. To generate a secure access key click "Generate secure connection key" button in the server connection tab, see fig 15



Any client wishing to connect will need to be provided with an identical key in the connection tab. See fig 16

Figure 15: Generate an access key



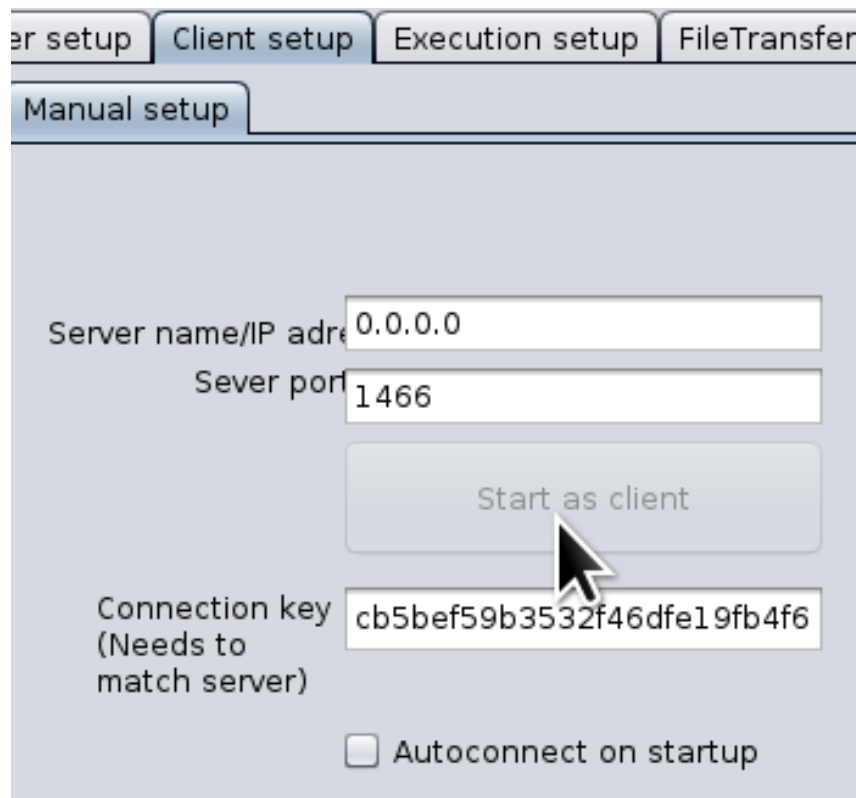


Figure 16: Add an access key to connecting client

## 6.5 Test dataset search

### 6.5.1 Interface installation Windows/Linux

On our SourceForge page (<https://sourceforge.net/projects/automatedproteo>) download APPInterface.zip. Unzip the archive using your preferred method. On Windows, double click APPInterface.exe (this will prompt a java installation if a JVM is missing or old version). On Ubuntu, right click the .jar file and check the "Executable" property. Then chose "Open with OpenJDK7" from the right click menu.

## 7 Tutorials and sample datasets

### 7.0.2 Starting up

Unpack the provided APPInterface.zip file. This will give you a directory contain APPInterface.jar, and for Windows users APPInterface.exe. You will need Java 7 to open the program, either OpenJRE7 (available in your local linux repository) or Oracle JRE7 ([www.java.com](http://www.java.com)). On Windows double click APPInterface.exe to run a Java check, auto set the reserved memory and start the interface. On linux, double click the launch.sh file and wait for the APP interface to load. A small connection dialog should pop up, with three tabs (see Fig. 17).

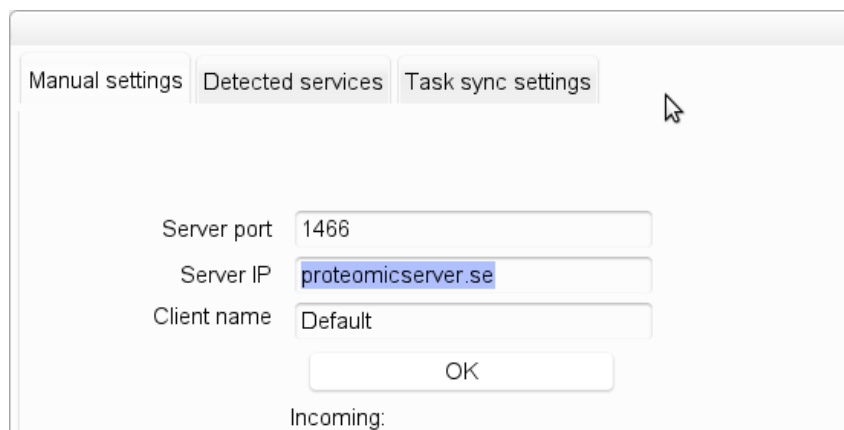


Figure 17: Connection dialog

If your server is on the local network simply click "Detect service" tab, then the "Detect server" button and wait for your server to be detected (see Fig. 18). Head back to the "Manual settings" tab and click connect. If your server is not detectable i.e. not on the local network you will have to enter the url manually into the connection dialog.



Figure 18: Detection tab of connection dialog

If user management is enabled you will now be prompted for a username and password combination.

Once connected the first thing that should happen is a syncing up of any new remote plugins of the server and the local plugins of the interface. This will be shown by a small sync gui. If there is such a sync event, it is recommended to **fully restart** the **interface** after the initial sync, this is done since plugins load at startup, as long as the plugins server-side are not changed the sync should not repeat.

### 7.0.3 Build a multiple search engine task

First download the APP tutorial data from our website (please see reference table 1, section 6.1). If you do not have plans to use at least two computing nodes, you might want to only use a few of the provided MGF files as input. A search in APP is made up of a series of connected modules. This means that a few parts need to be present to build a task:

The task needs actual database search engines such as X!Tandem and Comet, let's use those two for the tutorial, and to set their specific settings. The search engines also need the raw data in a format they can

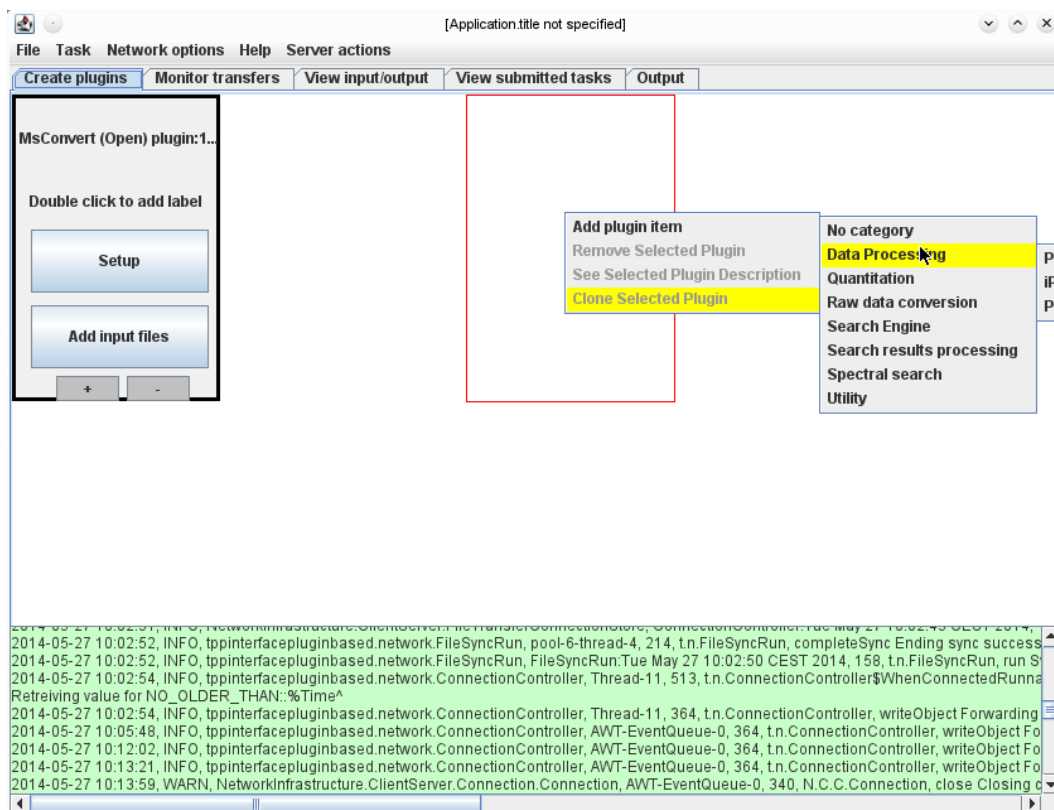


Figure 19: Task creation tab with popup-menu

understand, it is recommended to use mzML. Finally, search engines need info about what database, post-translational modifications and mass-tolerances to use. This is supplied via the "General Search Settings" plugin. In addition each search engine has a set of specific settings, which we will ignore for the time being (but read up under each specific plugin). Now set up a task to accommodate this. In the interface click on the "Create plugins" tab. You should now have an empty white window in front of you (Fig. 19). Right click in that window to bring up the plugin menu. The actions of the menu should not be too hard to grasp, allowing creation, deletion and cloning of plugins.

Try it now, right click the white space, then click "Add plugin item", then "Raw data conversion" and finally "MsConvert plugin" See Fig.20.

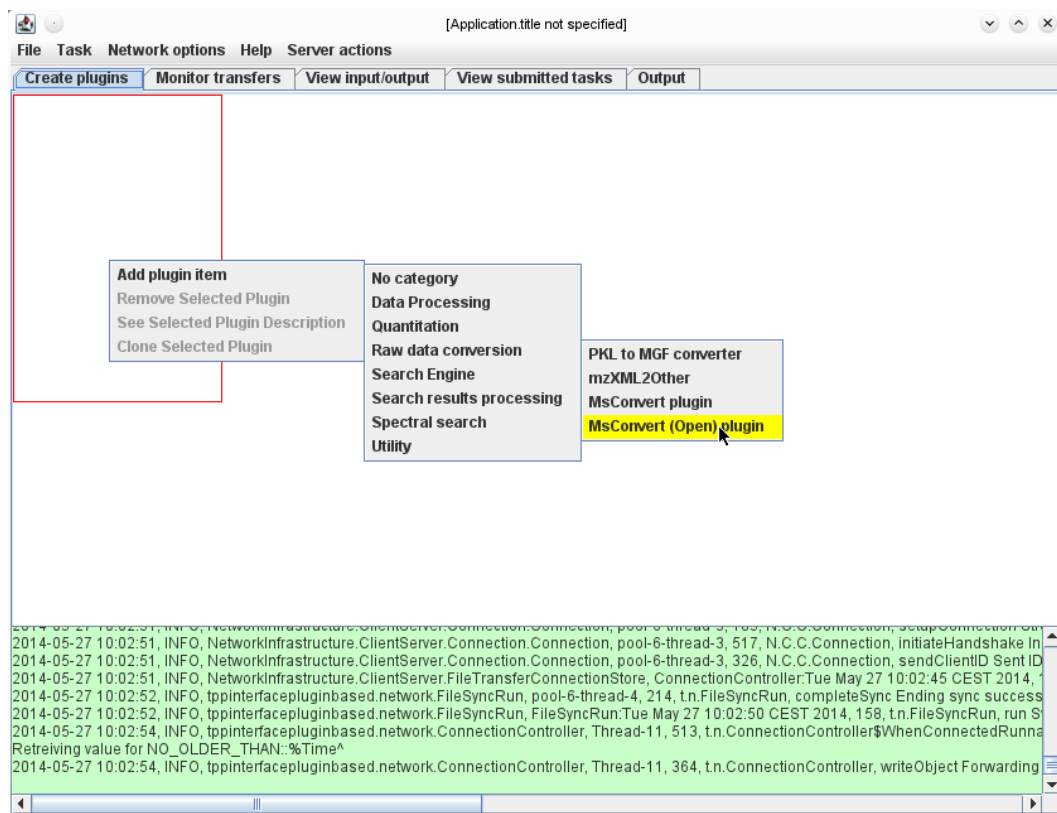


Figure 20: Create an MsConvert plugin

You should now have a small box with the words "Add input files" and "Setup" buttons on it. Move it around by clicking and dragging wherever there are no buttons, drop it somewhere to the right. Double click the "Double click to add label" to label specific plugins, this simplifies keeping track of specific plugins. Zoom in and out by using the mouse wheel, or chose one of the zoom options in the window "Task" menu. You can drag the entire field of view around by left clicking outside a plugin and dragging. Start by adding input files to our MsConvert plugin. Click on the "Add input files" button, then drag and drop files from your favourite file manager on to dialog and click "OK" (see Fig. 21).

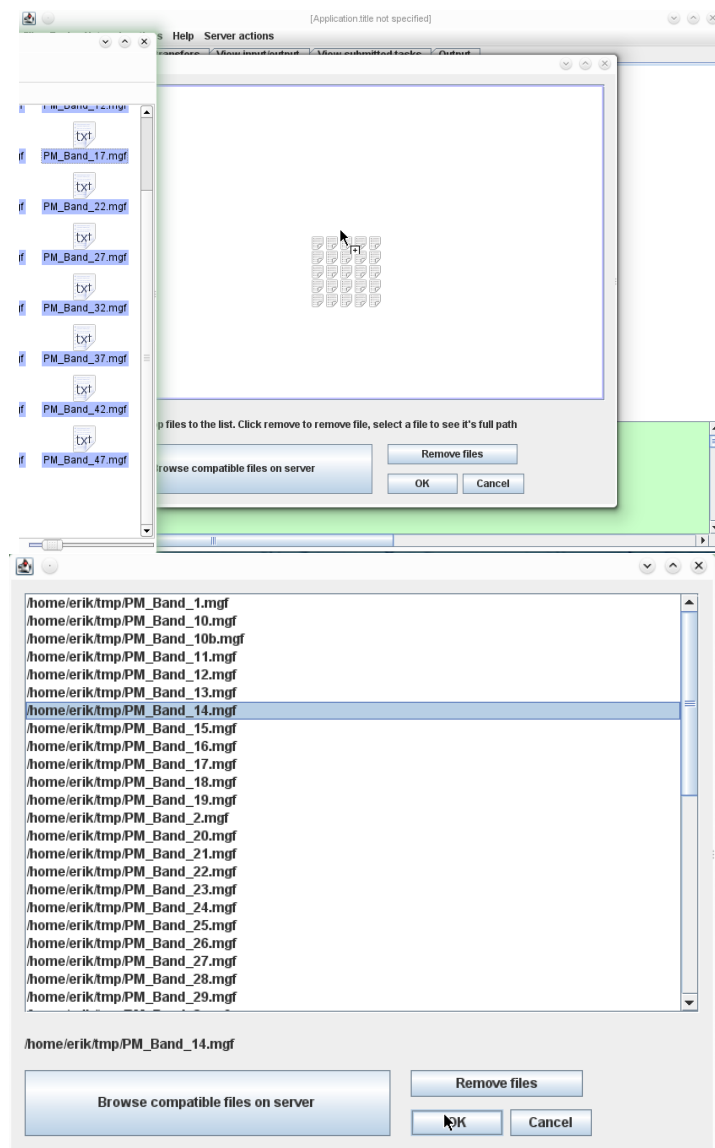


Figure 21: Adding input files to an existing plugin. Drag and drop.

You now have a plugin that will get input files and convert them into a different raw data format. Let's look at the details a bit later, for now add a few more plugins:

- Add one "X!Tandem" and one "Comet" plugin from the "Search engine" category. See figures 22 and 23

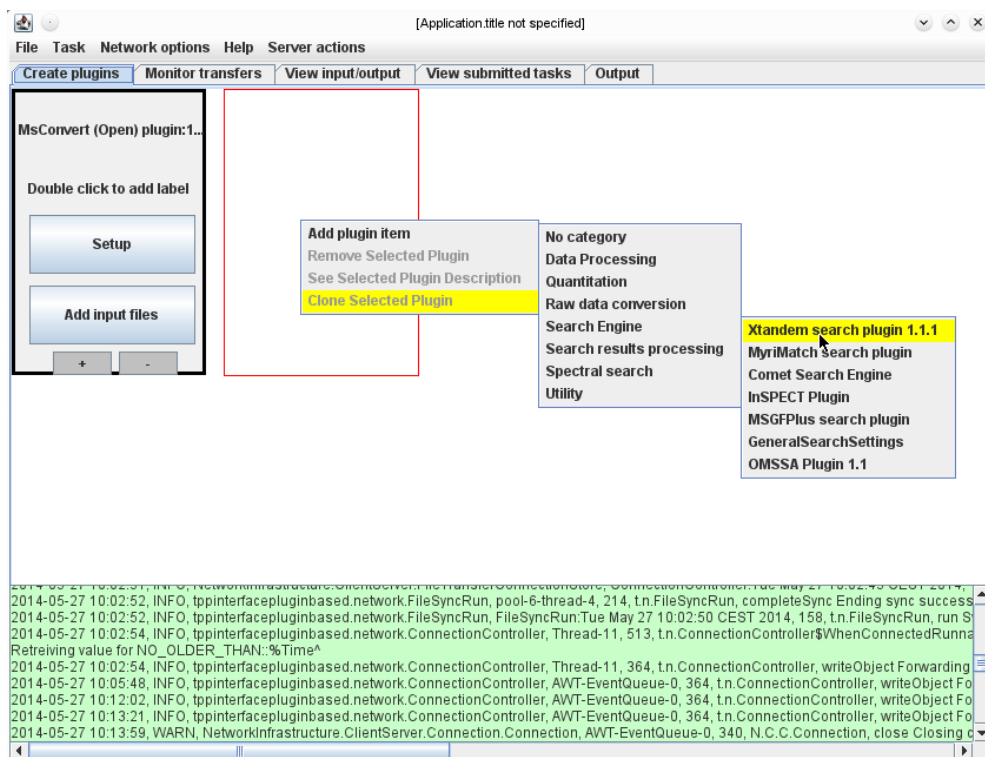


Figure 22: Adding an X!Tandem search engine plugin

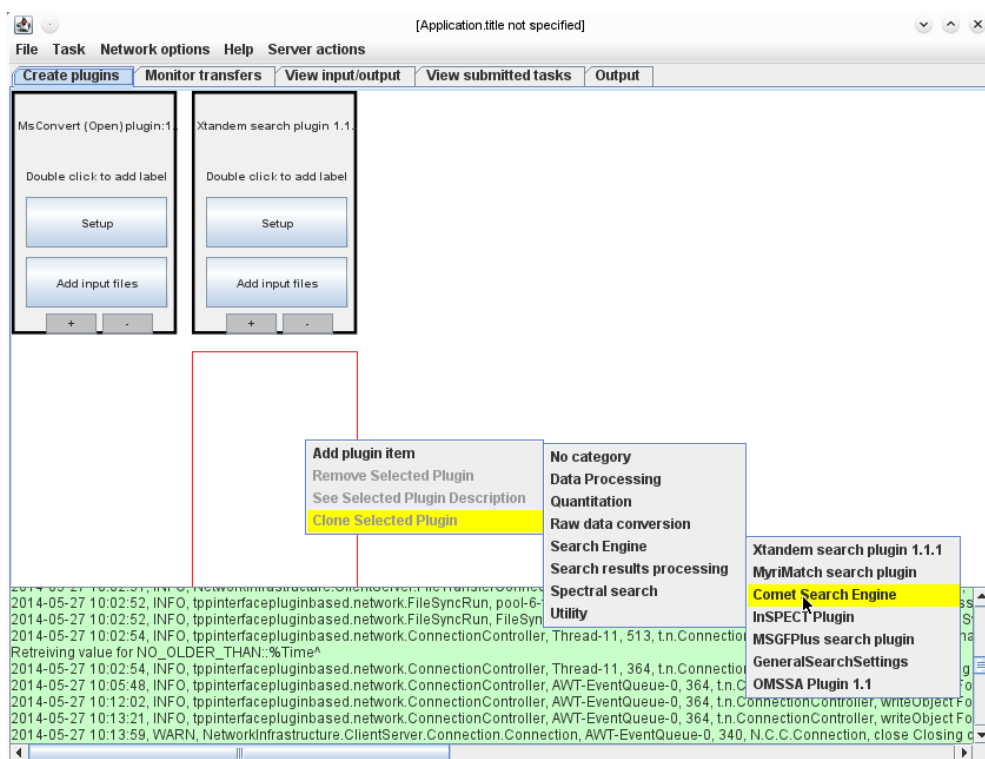


Figure 23: Adding a Comet search engine plugin

- Now to add the plugin that will provide search settings, add the GeneralSearchSettings plugin from the search engine category, see Fig.24

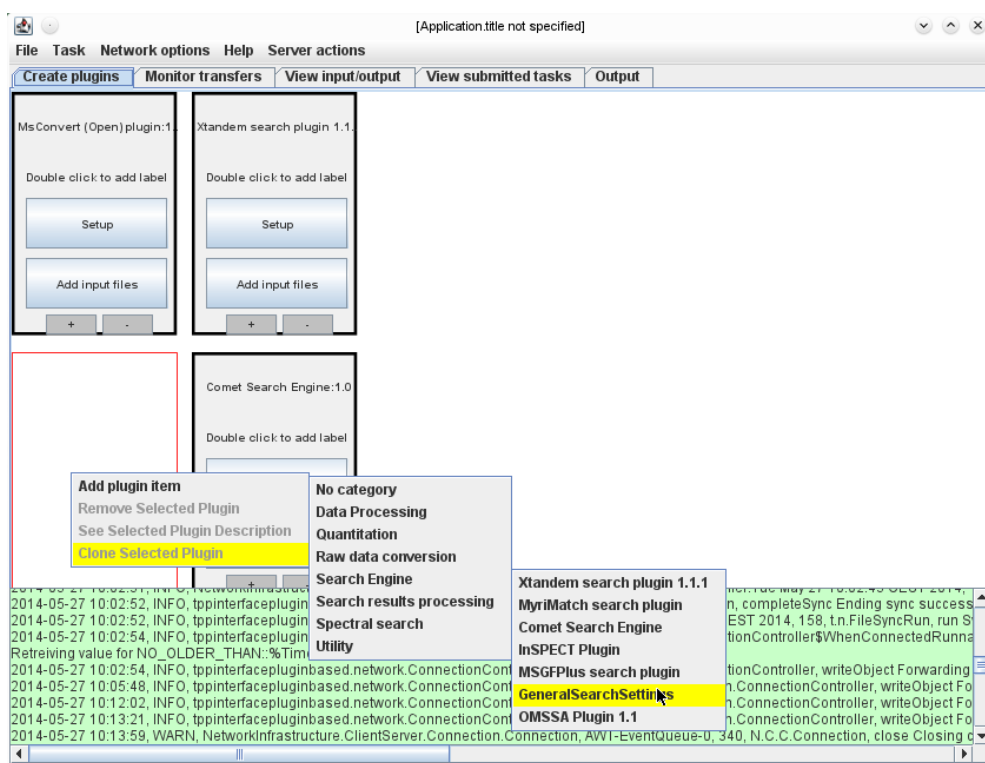


Figure 24: Adding a GeneralSearchSettings plugin

- Now on your own, from the *Data Processing* category add two *PeptideProphet* plugins to the right of the search engine plugins, then an *iProphet* plugin one step to the right of those and finally a *ProteinProphet* plugin to the right of that one. Drag and drop the plugins if you place them in the wrong place, when you're done your task should look like in Fig.25. Also try double clicking the "Double click to add label" text on the PeptideProphet plugin. Add a label to track which one is which.



Figure 25: Layout of the final task

Make sure you arrange the plugins in roughly the same order as pictured since each task maintains a flow from left to right, starting the leftmost plugin in a linked chain first.

- To direct a plugin to feed it's output into another, click the "+" sign button at the base of your plugin (see Fig.26) Click the "+" button on your MsConvert plugin. The "+" should turn red.

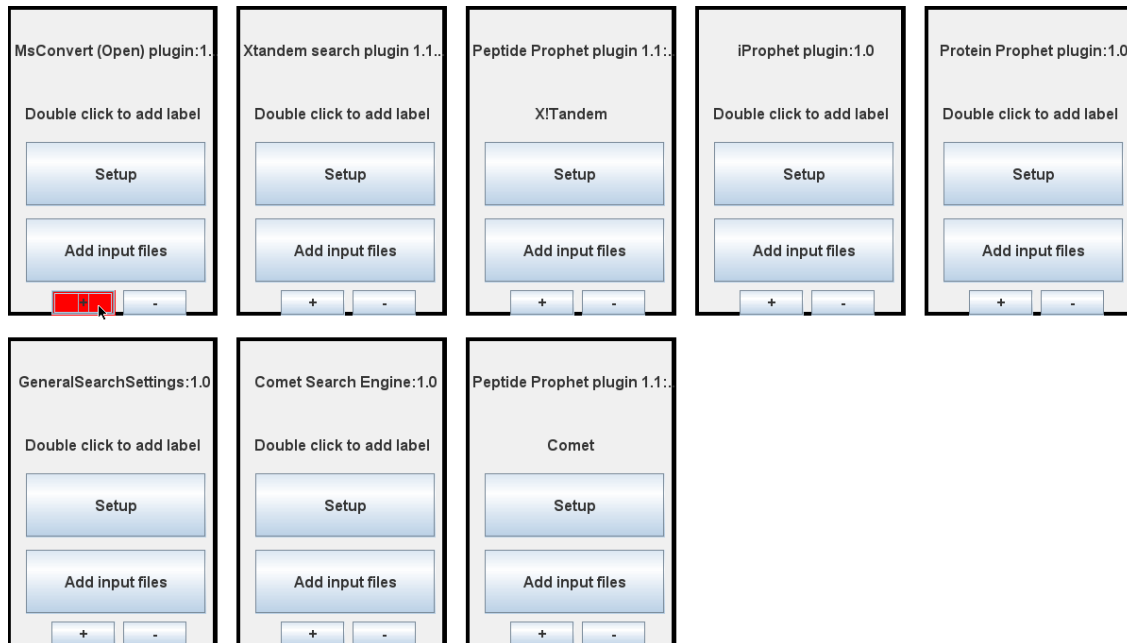


Figure 26: Initiating a plugin link

- Now directly after click anywhere on the X!Tandem plugin box. Do this until the "+" stops being red. Now when you hover your cursor over MsConvert X!Tandem should appear red. Vice versa, hover over X!Tandem and MsConvert should appear blue. Click the "+" on MsConvert again and then click the Comet plugin (Note: You can hold the "shift" key to allow linking or unlinking multiple plugins at once, i.e. click "+" or "-", hold shift and then click on any plugins).
- Click the plus button on the X!Tandem plugin and then click PeptideProphet.
- Hover your cursor over plugins to see their inputs and outputs, see Fig.27 for an example. Any *input* plugin will be highlighted as blue, whereas a plugin targeted for *output* will appear red.





Figure 27: Checking plugin input and output

- To link up the rest of the plugins, link "GeneralSearchSettings" to both "X!tandem" and Comet. Link Comet to the PeptideProphet plugin that does not have X!Tandem linked to it.
- Link both PeptideProphet plugins to iProphet, and finally link the iProphet plugin to the ProteinProphet plugin.

This is our chain, files will be converted from MGF to mzML in the MsConvert plugin, and then searched by X!Tandem and Comet using settings provided from GeneralSearchSettings. Finally output will be analyzed by PeptideProphet and will give vetted set of results. iProphet will take these and build a combined search result file which will then be processed by ProteinProphet into a final protein list.

The final step here is to setup each plugin and add the input files.

- Click setup on MSconvert and click "Perform peak picking" and make sure "Levels 1 and above" is the chosen option. This will mark peaks as centroided. Finally select mzML as the output format.

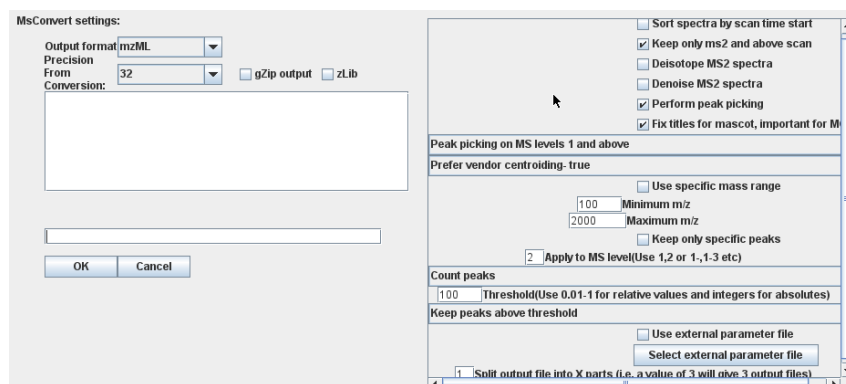


Figure 28: MsConvert settings for test data

- Click setup on "GeneralSearchSettings". This will initiate such a sync. available databases from the server. As such, if the database field stays empty for too long, please close the setup window and open it again which should also display any new databases available. Choose 50 ppm tolerance for MS1 and 0.2Da for MS2. Chose the *Populus* database with decoys, if you're using the base database set this is named "uniprot-organismPopulusTrichocarpaUniprotReferenceProteomeDecoyPrefixDc" from the dbase menu.

(see Fig.29). add a "Carbamidomethyl on C" fixed mod and an "Oxidation of M" variable mod(Hint: To get around quick press the first letter of the mod you're looking for).

The decoy database will allow use of PeptideProphet semi-parametric model which is necessary for validation of most supported search engines. If you only wish to run X!Tandem you can use one of the non-decoy databases.

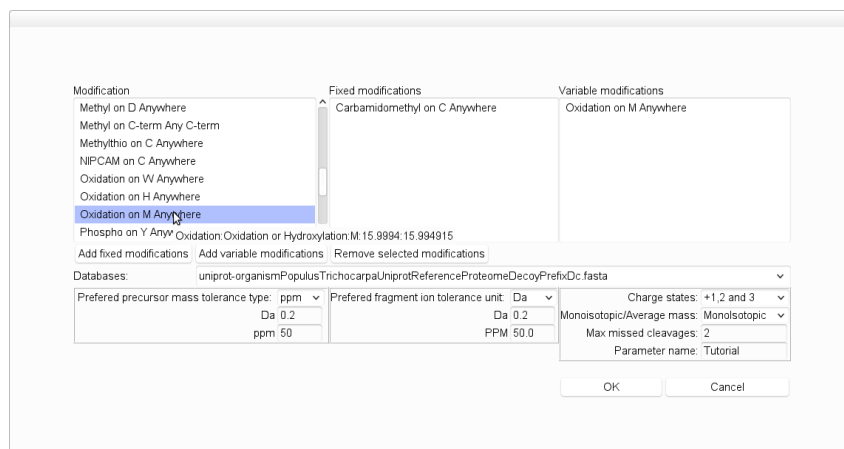


Figure 29: Setting up search parameters

- Click setup on the PeptideProphet plugin linked to "Comet" (hover your cursor over the comet plugin to see which one that is). Both Comet and X!Tandem can be analyzed without using peptide-prophets parametric model, but since Myrimatch,InsPecT and MSGFPlus all require decoys it's good to get into the habit of using them. To enable this click the "Use decoys to estimate correct peptides", see Fig.30 and enter "Dc" in the "Decoy label" field. Do the same thing for the PeptideProphet plugin linked to X!Tandem.

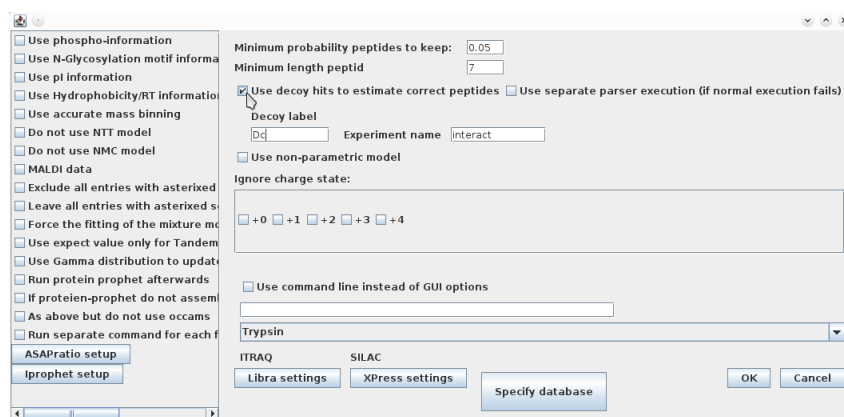


Figure 30: Add decoy option to PeptideProphet plugin settings

Finally, have a look at the search engine settings. Click "Setup" on the X!Tandem plugin, then make sure the defaultKScore parameters are selected (see Fig. 31, semi-trypic parameter files for tandem Native score and KScore are also provided, but these are more computationally intensive).

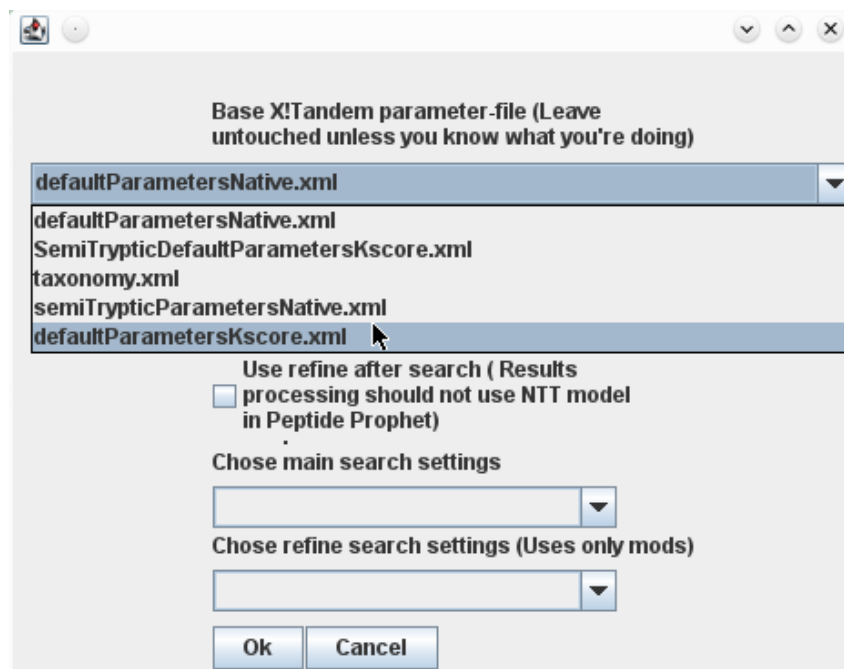


Figure 31: X!Tandem settings, choose the KScore parameter file.

There is no need to change any of the settings for iProphet. Instead click "Setup" on the ProteinProphet plugin and click the "Input is from iProphet" checkbox (Fig. 32).

You are now ready to submit your first APP task.

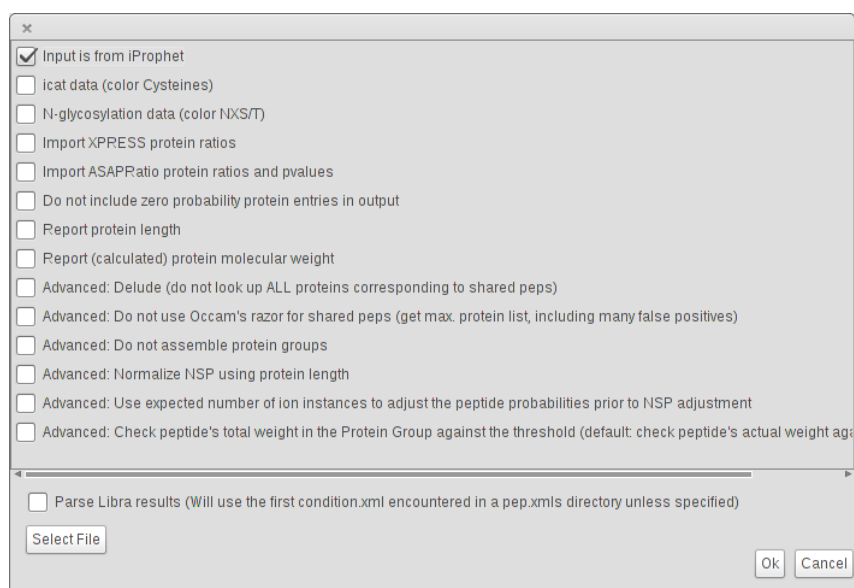


Figure 32: ProteinProphet plugin setup window

Save your task in the window menu "File". Pick a name and a description for your task, chose a file by clicking "Save as" and typing a file name. Finally, click "Save" (see Fig. 33 ).

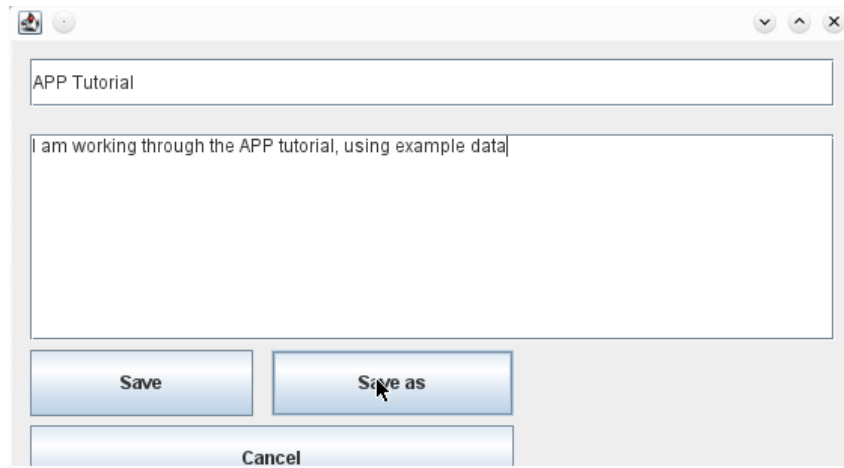


Figure 33: Save task dialog

Click on the "Task" window menu and chose "Submit current task". If you are fast you can look at the ongoing file transfers by looking at the "Monitor transfers" tab and clicking update (see Fig. 34). Once all the transfers are done your task should start processing.

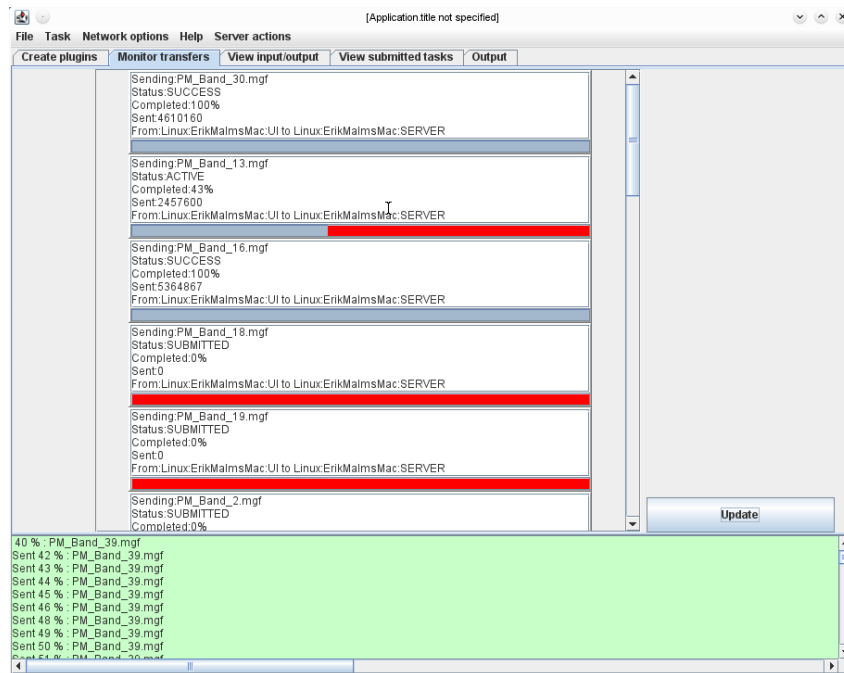


Figure 34: Monitor transfers going to and coming from the running interface

#### 7.0.4 Monitor execution:

Click on the "View submitted tasks" tab. Here all submitted tasks are shown in a tree form. Expand the tree and you will find a list of execution indexes, rising from 1. Double click to expand any of these nodes to see the state of your task, running tasks are updated roughly once every 40 seconds.

Under each plugin is a list of jobs performed by this plugin. Click on one to see any output so far. Click on the task or a plugin to get summary information. Wait for the task to show the status "complete" when you highlight it (to track ongoing task execution in detail, have a look at individual plugin nodes). Right click

anywhere on a task and choose "Open selected task" to see a more familiar representation of your task (see Fig. 35).

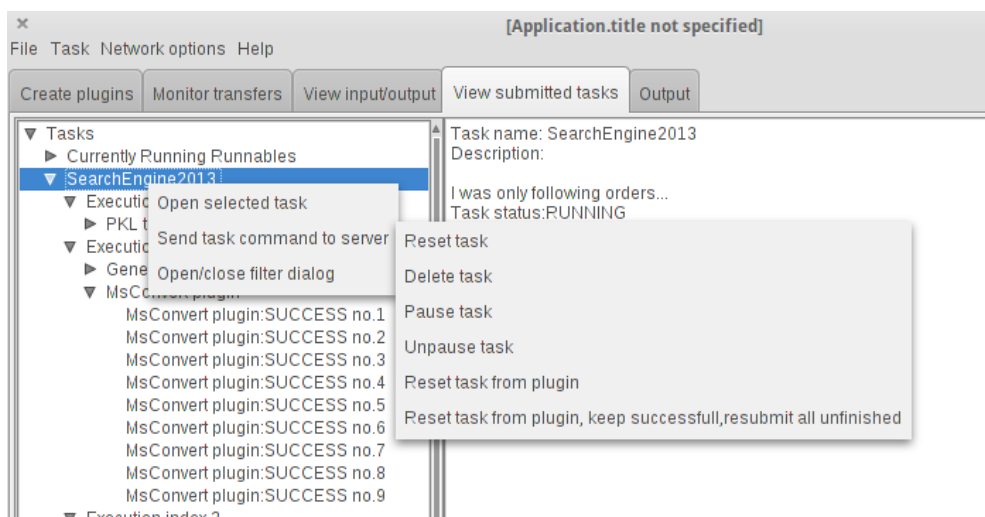


Figure 35: See all currently running tasks, by default only tasks from the last 30 days will be displayed.

Click browse jobs on any of the plugins to get detailed information on the execution (fig 36).

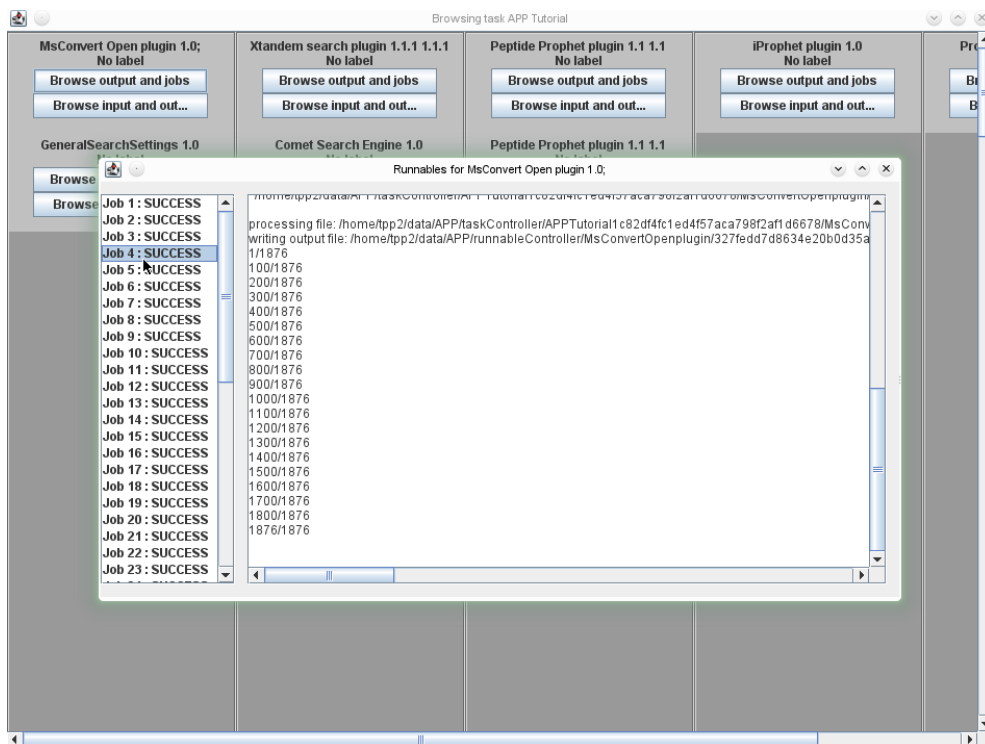


Figure 36: View the details of all operations performed by a plugin in the "Browse output and jobs" view.

Click "Browse input and output files" for the X!Tandem plugin. Highlight any of the files ending with pep.xml and click "Open" (fig 37).

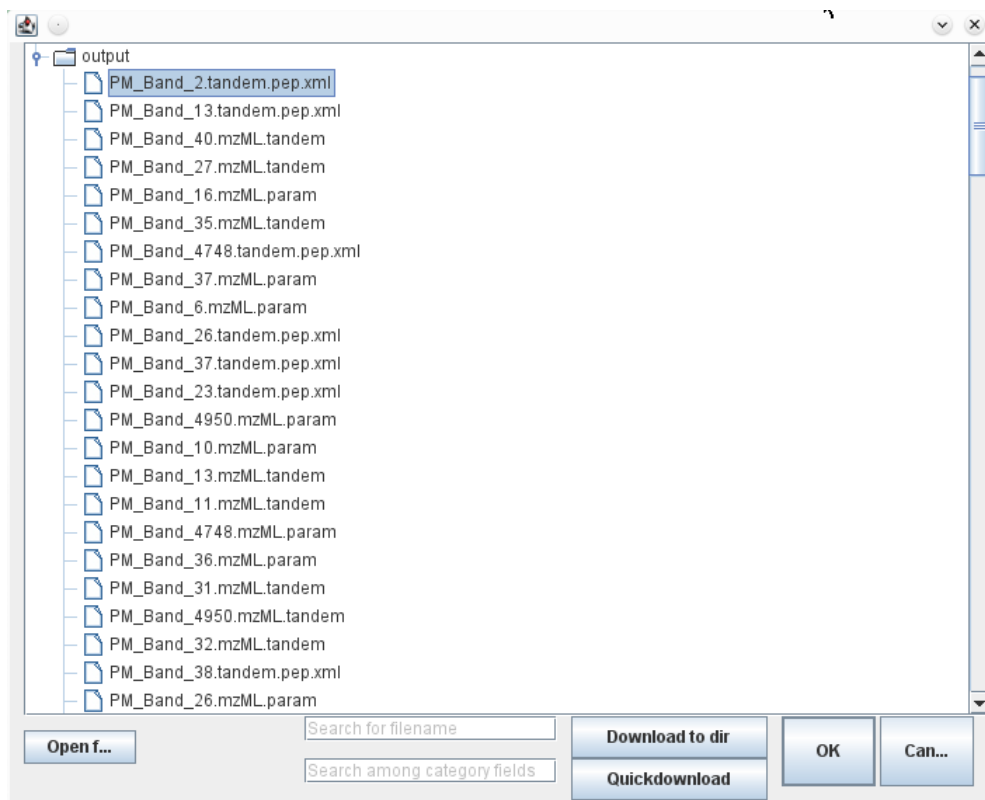


Figure 37: Any files created by a plugin can be seen in the "View files" dialog. Some file types, pep.xml, prot.xml and html files can be forwarded to your browser for display.

A popup window with a URL will show up, click "OK" and a browser should open showing your results (see Fig. 38).

Save to Mendeley

Peptide Viewer, 2006-03-15

Page 1 of 20

FIRST 1 2 3 4 5 6 11 NEXT LAST

SPECTRUM	HYPERSCORE	NEXTSCORE	EXPECT	IONS2	PEPTIDE	PROTEIN	CALC. MASS
PM_Band_2.00001.00001.2	504	374	0.42	12/18	R. KPDIIDGGITK, S <sup>+</sup>	trB9H5L8IB9H5L8_POPT <sup>+</sup>	1030.5297
PM_Band_2.00002.00002.2	329	328	7.1	9/22	R. ENGRSSNEGSAR, T <sup>+</sup>	trB9I6G1IB9I6G1_POPT <sup>+</sup>	1262.5597
PM_Band_2.00003.00003.2	278	264	3	10/22	K. KLTAASSPTGK, H <sup>+</sup>	trB9GME4IB9GME4_POPT <sup>+</sup>	1160.6397
PM_Band_2.00004.00004.2	268	255	4.8	11/22	K. NKLESDSGAASR, V <sup>+</sup>	trB9GJX7IB9GJX7_POPT <sup>+</sup>	1233.5947
PM_Band_2.00005.00005.2	295	272	5	8/30	K. LTPGSAASNNSGGAGK, N <sup>+</sup>	trA9PFH8IA9PFH8_POPT <sup>+</sup>	1387.6687
PM_Band_2.00006.00006.2	258	238	1.5	9/16	R. QRWRATSR, K <sup>+</sup>	Dc21644	1174.5957
PM_Band_2.00007.00007.2	325	272	0.14	11/20	K. ESANSDQPLQK, A <sup>+</sup>	trB9GW15IB9GW15_POPT <sup>+</sup>	1215.5731
PM_Band_2.00008.00008.2	230	225	12	12/16	K. AEGPMI47.04ALYK, G <sup>+</sup>	trB9IJB0IB9IJB0_POPT <sup>+</sup>	994.4797
PM_Band_2.00009.00009.2	234	209	5.8	10/24	K. TLAEVVGSEEEAR, K <sup>+</sup>	trB9I1V8IB9I1V8_POPT <sup>+</sup>	1388.6787
PM_Band_2.00010.00010.2	248	243	7.1	5/14	R. KVKNNMVR, A <sup>+</sup>	trIU5G5B4IU5G5B4_POPT <sup>+</sup>	944.5587
PM_Band_2.00011.00011.2	246	235	9.4	9/14	R. NNLTNTLK, D <sup>+</sup>	trIU5FXX9IU5FXX9_POPT <sup>+</sup>	916.4977
PM_Band_2.00012.00012.2	242	236	15	10/16	R. ADEAALSVR, E <sup>+</sup>	Dc29206 +1	930.4767
PM_Band_2.00013.00013.2	232	230	7.4	5/14	R. KGYKGFQK, E <sup>+</sup>	trB9GTK4IB9GTK4_POPT <sup>+</sup>	954.5287
PM_Band_2.00014.00014.3	387	365	7	16/44	- . ALAPISCI60.03NRC160.03DR, D <sup>+</sup>	Dc222021	1431.6707
PM_Band_2.00015.00015.4	571	554	6.3	23/102	R. APKSPQELKTASPPALK, Q <sup>+</sup>	Dc22539	1909.0677
PM_Band_2.00016.00016.3	416	401	1.5	16/80	R. AGSPSSAALAAATSAVDALR, T <sup>+</sup>	Dc15440	1872.9537

Figure 38: Output from X!Tandem search engine

Looking through the output you will find some entries with a protein name starting in "Dc". These are the

decoy spectrum matches and should be filtered out in the next step. Click "Browse files" on the ProteinProphet plugin and open ProteinProphet.prot.xml in the same way to get the final output (fig 39).

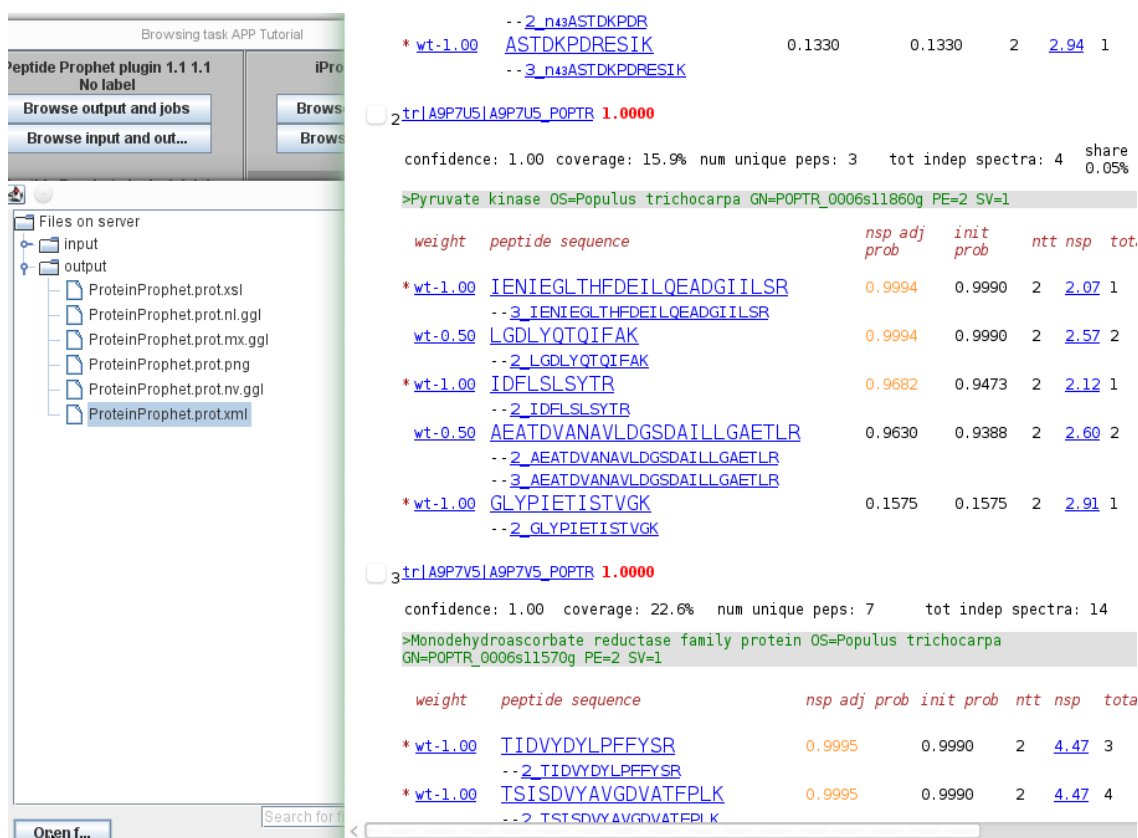


Figure 39: The final protein list from our test search can be retrieved from the "Protein Prophet" plugin.

### 7.0.5 Work with output from previous task:

Now, if you wanted to do a similar task again, all you would have to do is open your saved task and change the input data to the "MsConvert" plugin. If you want to use the LabelFreeQuantitation plugin to extract information about your last search (as an example) this can be done directly using the output of the previous task. This plugin extracts info about a search, including information that can be used for quantitation such as each proteins Spectral count or average Total Ion Current. It's also an easy way to extract info not generally displayed in the web based viewers such as Xpress-generated label-free quantities.

- Create a new task from the file menu.
- Add a "LabelFreeDataExtractor" plugin from the "Quantitation" category.

There is no need to change anything in the settings, but you can still have a look at the setup screen (Fig. 40).

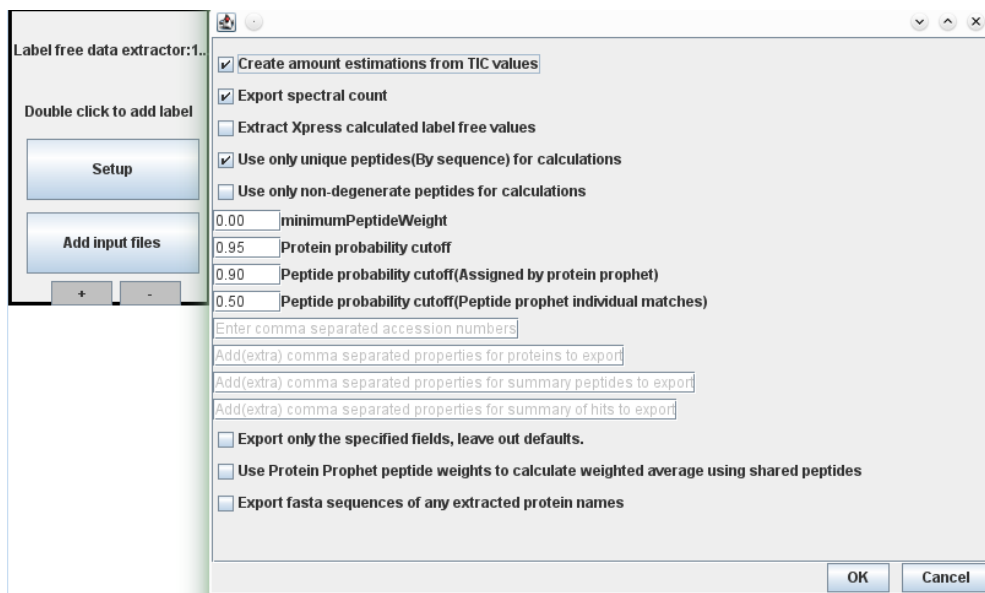


Figure 40: The "LabelFreeDataExtractor" plugin allows you to extract info from prot.xmls and pep.xmls into tabular formats.

Start by adding the files from your previous task to this plugin, as detailed below (or see Fig. 41).

- Click "Add input files" on the plugin.
- Click the "Browse compatible files on server" button. A sync of files available on the server should now commence from the server. Syncing info on available files from the server can take a few seconds, if the display of files is empty try closing the window and clicking the button again.
- A list should now appear as below (Fig. 41). If there are many files you can type the name of your task in the "Category filter" textfield or any part of the file name in the "File name" filter textbox. Now navigate to the output of ProteinProphet, highlight "ProteinProphet.xml" and click "Add vaultfiles" (Fig. 41).
- Save, name and submit your task as previously.



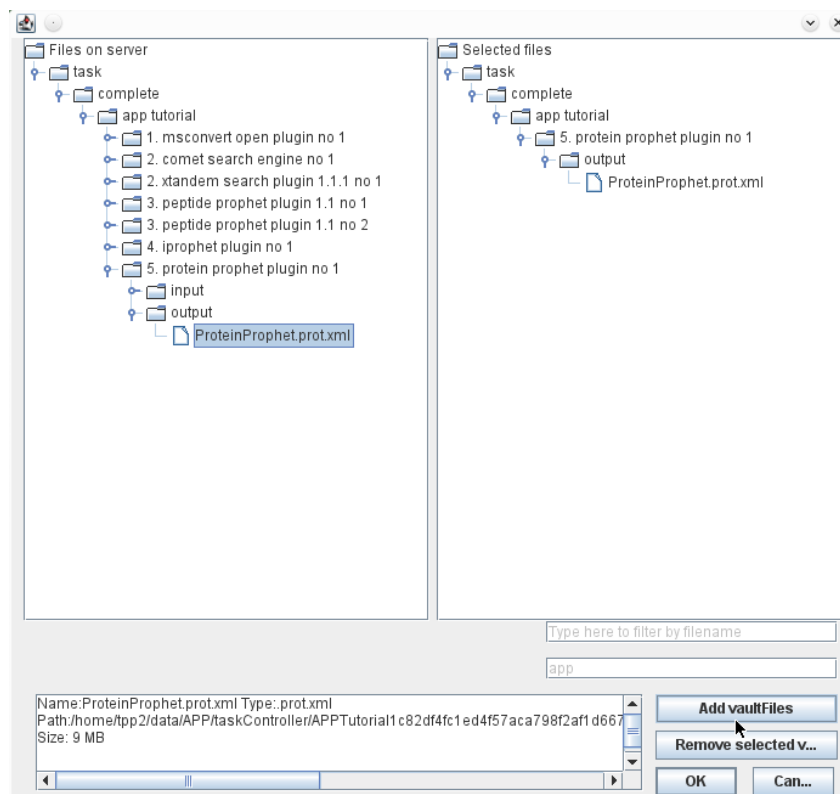


Figure 41: Files available from your previous tasks can be used as input for new tasks, directly on the server. Note the two filter fields for file- names and task names (categories).

Once the task has completed, right click on the task to open a menu, then click "open task". After that click open files on the LabelFreeDataExtractor plugin, highlight them all and click "Quick Download" (Fig. 42). Files download in the background so there is no need to keep the task window open. The result files will now be added to your local "Downloads" directory. For the most succinct info for the plugins, open ProteinProphet.proteins.txt in the spreadsheet editor of your choice to see all kinds of interesting information. These files are tab separated text files, as such avoid opening them in notepad.

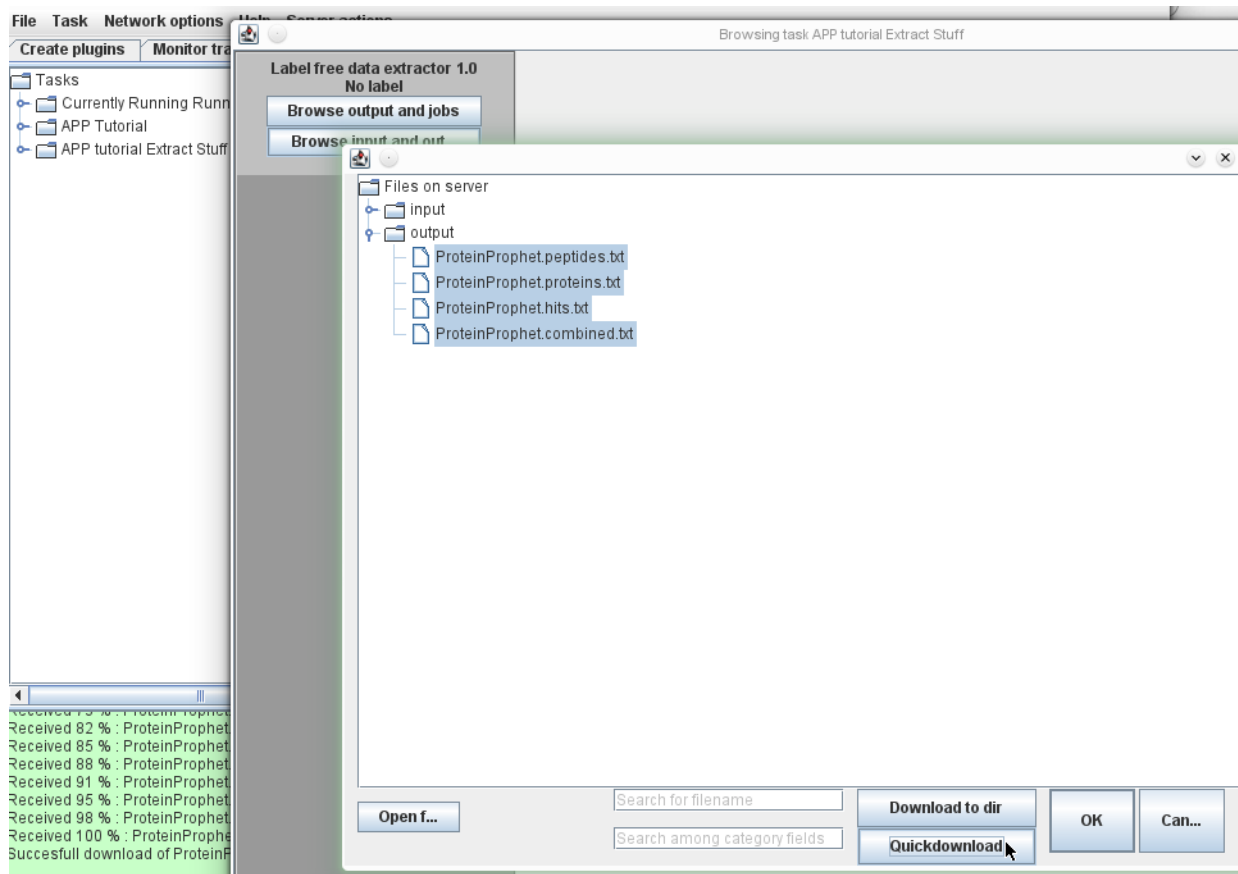


Figure 42: Quickdownload will download files to your Interfaces download directory. Files are downloaded a folder composed of Task name, plugin name, and any label applied to the plugin.

## 8 Available Plugins

An aim for APP plugins is to provide complete compatibility with the excellent Trans Proteomic Pipeline(TPP)<sup>1</sup> and the brunt of plugins aim to automate tasks done through TPP. A core distribution of APP also includes several tools not typically found in a vanilla TPP install along with plugins included for easier automation.

### 8.1 General plugin function

Plugins in general come with two direct ways of interaction: A setup button, and an add file button. All plugins behaviours are governed by their individual settings along with their input files. After completion the plugin then passes on a selection of files for further processing down the pipeline. This is not necessarily every file generated by the plugin, such as in the case of the PeptideProphet plugin where pep.xml outputs are passed on even if a prot.xml is also generated.

### 8.2 Raw data conversion plugins

Mass spectrometry data comes in a large variety of sizes and formats, in spite of many attempts to standardize. While a push towards mzML format can be seen throughout industry the most compatible format, in terms of tools utilizing it, is for the present still mzXML and this is the preferred format used by APP, mzML is still

<sup>1</sup><http://sourceforge.net/projects/sashimi>

available for the tools that handle it well. All plugins in the Raw Data conversion category are used to process, convert or otherwise modify mass spectrometry data. Individual plugins are further described below.

### 8.2.1 MSconvert plugin

Table 2: MsConvert plugin summary

Plugin name	Needed binaries	Accepted input formats	Available output formats
MsConvert plugin	MsConvert(.exe)	.raw (Windows only) .d (Windows only) .mzXML .mzML .mgf	mzXML mzML mgf

The msconvert plugin provides an interface to the excellent MSconvert utility created by the ProteoWizard<sup>2</sup> project. Msconvert can convert a number of vendor specific formats to the open mgf, mzXML or mzML formats. Additional functionality include a number of filters, such as keeping only the top peaks in an MS2 spectra and functions to denoise, deisotope and centroid peaks. Msconvert conversion can get very close to the results from vendor specific post-processing with a bit of tweaking. Options that can be set from the GUI include the following:

- *Output format*

Controls the mz-format of the output, options are mzXML (recommended for maximum compatability with APP tools.),mzML and MGF.

- *gzip output file/zlib peak list*

Compresses the output file or peak list to reduce space usage, though some search engines do not deal well with either gzip or zlib. As such it is recommended to be left off and to instead keep only MS2 and above spectra as well as perform deisotoping, denoising and peakpicking as this will also reduce output file size.

- *Precision from conversion*

Precision value in bits for both MS1 and MS2 data. 32 is recommended since some search engines has trouble handling 64 bit precision. For most tasks the lost precision does not affect accuracy.

- *Keep only MS2 and above*

Will filter out all MS1 scans. This will reduce data size and makes conversion much faster. If data is for database search or de-novo sequencing

- *Sort spectra by scan times*

Orders spectra in the produced file by scan time. Is set by default since it makes spectra comparisons between different converted files more consistent but has little effect on the workflow.

- *Deisotope MS2 spectra*

Deisotopes all peaks in MS2 spectra, usually is a great aid for MSMS search software but is left off by default.

---

<sup>2</sup><http://proteowizard.sourceforge.net/>

- *Denoise MS2 spectra*

Denoises MS2 spectra to remove random signal. Can greatly increase the number of hits for X!Tandem and other search engines.

- *Fix titles for Mascot*

If the MsConvert plugin is being used to generate data for Mascot search this option should be checked. It has no negative effect on other data and as such is left on by default.

- *Perform peak picking*

Msconvert will centroid the peaks, usually greatly boosting database search hits.

- *Peak picking levels*

Decides which MS levels centroiding applies to (If selected with the Peak Picking option). Default is to Centroid all MS levels from 1 and up.

- *Prefer vendor centroiding* Vendor centroiding algorithms is provided to the MSconvert team by various mass-spectrometer vendors and usually ensures the best possible centroid. This is not available for all vendors and if not present centroiding will act as if this option is set to false. In this case centroiding will be performed by a local-maximum seeking algorithm.

- *Use specific mass range*

This will keep only peaks in between the specified mz values specified in the linked options minimum m/z and maximum m/z.

- *Apply to MS level* Decides which MS levels will have peak filtering done, defaults to only MS2. i.e. the mass range and peak filtering options are only applied to MS levels specified here.

- *Keep only specific peaks* Decides if peak filtering will be performed at all. defaults to off.

Linked settings are:

Count peaks/ Count peaks after ties/ Absolute / Relative to top BPI / Relative to top TIC / Minimum tic Cutoff: All determine by what criteria should be used for filtering. Count is the default setting, in this case a certain number of peaks are kept in each spectra. Typically the top 40-100 peaks are kept. Count after ties is equivalent but also keeps all peaks that have equivalent intensities.

The relative values will keep all peaks that are close to the top intensity peaks, using either base-peak intensity or total ion current criteria. Absolute uses a certain intensity value cutoff and keeps peaks that pass it. Minimum TIC keeps peak above or below a certain TIC value.

Threshold: Determines the cutoff level. If using for an absolute such as the count, count after ties, absolute or minimum tic criteria, threshold should be set to a full number (i.e. setting threshold to 100 will keep the top 100 peaks) and if using a relative value set somewhere between 0.01-1. For example a threshold of 0.5 with a criteria of "Relative to top TIC" will keep any peaks that are within 50% of the top peak.

Keep above threshold/keep below threshold: Determines if peaks above or below the set criteria are kept. As such a "count" criteria with a threshold of 100 and the keep above option will keep the top 100 peaks in a spectra whereas if set to "Keep below threshold" it will keep the lowest 100 peaks.

- *Split into X parts* Will split mgf/mzML/mzXML files into multiple parts. All files contain a subset of spectra from the original files. A value of 3 will split input files into 3 parts. A file containing a 1000 MSMS spectra would thus generate two files containing 333 spectra and the final file would contain 334 spectra. Splitting files allow large files to be distributed more easily (MSGFPlus for example has limited

multi thread support but can easily be run as several parallel process on a split file). Output files have a label of "-part1", "part2" etc. And will be compatible with all TPP/APP analysis tools. Note that *SpectrumNameFixer* should be run on output from MSGFPlus/IDConvert and Myrimatch to ensure spectrum names are consistent with TPP expectations.

It is also possible to provide an MSconvert options file directly by uploading it, or to enter command line options directly into the text provided text field just as one would if running from command line.

### 8.2.2 mzXML2Other

Table 3: mzxml2other plugin summary

Plugin name	Needed binaries	Accepted input formats	Available output formats
MzXML2Other plugin	mzxml2other(.exe)	mzXML	pkl odta dta ms2

TPP provided conversion utility, offers conversion from mzxml to a number of other formats including MGF,dta and others. Conversion to mgf provides titling of the spectra, and these can then be searched through MASCOT and then remapping. This utility has mostly been replaced by MsConvert and will not be covered in great detail here, in general most options should be left as default.

### 8.2.3 PKL to MGF

Table 4: Pkl to MGF plugin summary

Plugin name	Needed binaries	Accepted input formats	Available output formats
PKL to MGF	None	PKL XML file processed by BioTools	mgf

PKL files are here converted to MGF typically for further conversion into mzXML and use in the pipeline. The interface offers no options and any tweaking of the raw data should be done in a separate MsConvert plugin step.

## 8.3 Database search engine plugins

Several MS search engines are provided with the default APP installation. All search engines take their basic input for mass tolerances, database to search, fixed and dynamic amino acid modifications etc. from a single plugin. As such all search engine plugins expect at least three input files: A fasta database to search, a .gsp settings file from the General Search Options plugin (described below) and also at least one data file in mzXML or mzML format (mzXML has broader compatability).

### 8.3.1 General Search Settings plugin

All search engine plugins take their basic input for mass tolerances, database to search, fixed and dynamic amino acid modifications etc. from a single plugin. This allows search settings to be described only once and then utilized for all search plugins in the current task.

The settings handled through the general search settings plugin are:

Table 5: General search settings plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
General search settings	None	None	gsp

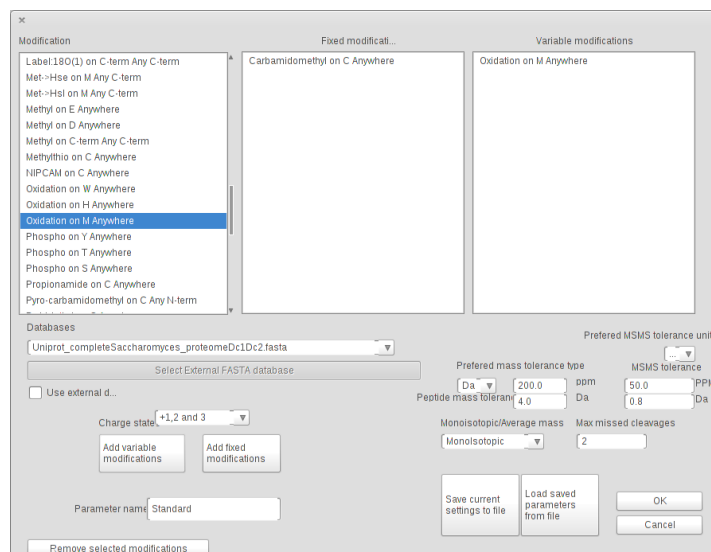


Figure 43: The general search settings setup window.

- Fixed/dynamic modifications

An export of the Unimod database is provided with the plugin, this provides a large set of modifications for use in search.

- MS1/MS2 mass tolerance, as well as preference for which to use In both PPM and Da. Since some search engines do not support PPM setting it is important to provide both Da and PPM tolerances. The search engine plugin will then use your preferred mass tolerance method if supported, and will fall back on Da if needed.
- Database, choose the search database to use. Files provided by the server are available through the dropdown menu. It is also possible to provide your own .fasta file, though this should be limited to smaller DBs.
- Charges, limit which charges are considered. This setting is not respected by all search engines (Though, X!Tandem and OMSSA do) and should usually be left alone.
- Name of the parameters, decides the name of the output. Has no effect on actual search but makes it easier to keep track of which set of parameters has been used.
- Monoisotopic or average mass switch. Most modern instruments should only consider using monoisotopic mass.
- Max missed cleavages, determines how many cleavage sites can be missed for a peptide. Higher numbers greatly increase search time.

### 8.3.2 X!Tandem

Provides access to X!Tandem search engine<sup>3</sup>. Minimum input is a database file, a search settings file and a raw data file in either mzXML (recommended for maximum compatability) or mzML. X!Tandem has a large range of options, dwarfing most other search engines. Among these is an ability to perform a search using one set of fixed and dynamic modifications and to then further refine the search by doing a second round of searching, usually using a larger number of dynamic modifications, against any protein models identified in the first run. To use this methodology a second settings file has to be provided, it is then possible to choose which set of modifications to use for the second round of searches. The gui provides options for the following:

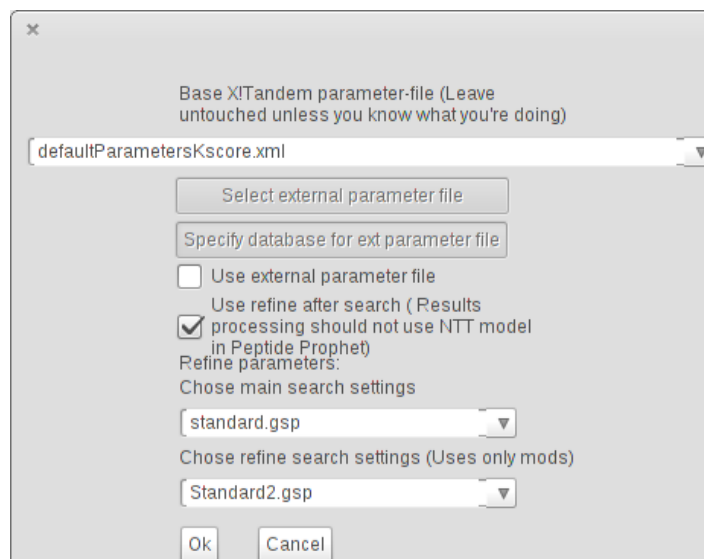


Figure 44: X!tandems setup window.

Table 6: X!Tandem search plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
X!Tandem plugin	tandem(.exe)	mzML/mzML	pep.xml
	tandem2xml(.exe), on server only	fasta	
		gsp	

- *X!Tandem base parameter file*, to deal with X!Tandems extensive range of esoteric options such as spectrum conditioning or variants of scoring the plugin uses several standard X!Tandem parameter files as it's base. Base variants are provided for using kscore, tandem native scoring with either tryptic or semi-tryptic search variants.
- *Use external parameter file* If a different set of base options are needed for the search, this can be provided by uploading an X!Tandem parameter file.
- *Use refine search* This demands the input of at least two different parameter files. By selecting a main and secondary .gsp file it is possible to perform a refine search using a second set of parameters and mass tolerances.

<sup>3</sup><http://www.thegpm.org/TANDEM>

### 8.3.3 Myrimatch

Provides a plugin for the excellent Myrimatch<sup>4</sup> search engine. Most settings are provided by a .gsp file from a settings plugin. The produced pep.xml file often have scrambled spectrum references, this can be corrected through *SpectrumNameFixer* plugin. The gui provides options for the following:

Table 7: Myrimatch plugin overview

Plugin name	Needed binaries	Accepted input formats	Available output formats
MyriMatch plugin	myrimatch(.exe)	mzML/mzML fasta gsp	pep.xml MZident

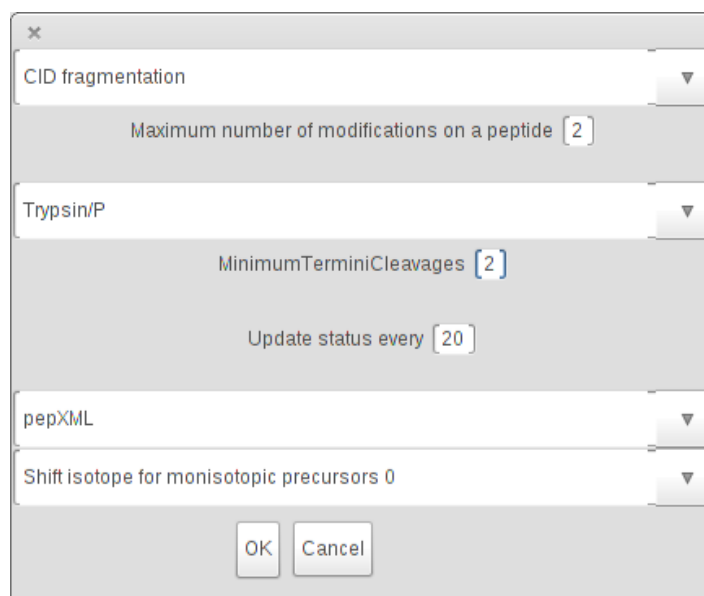


Figure 45: Myrimatch setup window

## 8.4 Comet

Comet is fully featured search engine, it provides a wealth of spectrum processing options along with very fast speed.

Table 8: Comet search plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
Comet plugin	comet.linux.exe (only on Linux 64 bit) comet.win32.exe (only on Windows 32 bit) comet.win64.exe (only on Windows 64 bit)	mzML/mzML fasta gsp	pep.xml

The comet interface provides two separate settings groups, one for comets standard settings, along with a range of advanced settings. The basic settings are outlined here, but for a full understanding of the advanced options users should refer to documentation found on Comets homepage: <http://comet-ms.sourceforge.net/>

<sup>4</sup><http://fenchurch.mc.vanderbilt.edu/lab/software.php>



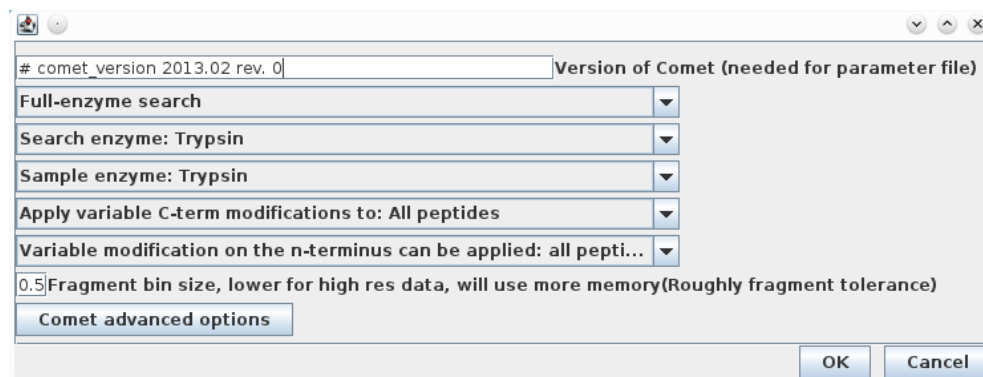


Figure 46: Comet basic setup window

- *Comet version* This header is needed for generation of parameter files, unless you upgrade comet, don't touch.
- *Full enzyme search/Semi-tryptic* Defines which peptides are considered, choices include fully tryptic (default), fully semi-tryptic or consider semi-tryptic cleavage only in the direction of the N or C terminus.
- *Search-enzyme* Defines which enzyme is used to define cleavage rules for Comet
- *Sample-enzyme* This enzyme will be reported in output. For example it is possible to perform the search with "No enzyme" but present trypsin in the output.
- *Apply variable C-term modifications* Determines if Comet considers c-term variable modifications on every peptide or only for specific protein derived peptides.
- *Apply variable N-term modifications* Determines if Comet considers N-term variable modifications on every peptide or only for specific protein derived peptides.
- *Apply variable C-term modifications* Bin size determines in how fine partitions the MS2 spectra are treated during a comet search. This roughly translates into fragment tolerance and lower numbers will require a higher amount of memory to be available. A more full explanation is available on the comet home page.
- *Comet advanced options* Comet advanced options provide access to the search engines full range of options, these should not be tweaked without first consulting: <http://comet-ms.sourceforge.net/>

## 8.5 MSGFPlus

MSGFPlus features powerful models for detecting phosphorylated peptides and an innovative edge-scoring algorithm. MSGFPlus outputs exclusively mzID files, these need to be converted into pep.xml using the **IDConvert** plugin when included in a project. To ensure correct spectrum names **IDConvert** output should then be fed into the **SpectrumNameFixer** plugin.

Table 9: MSGFPlus search plugin summary

Plugin name	Needed binaries	Accepted input formats	Available output formats
MSGFPlus plugin	MSGPlus.jar	mzML/mzML fasta gsp	mzid

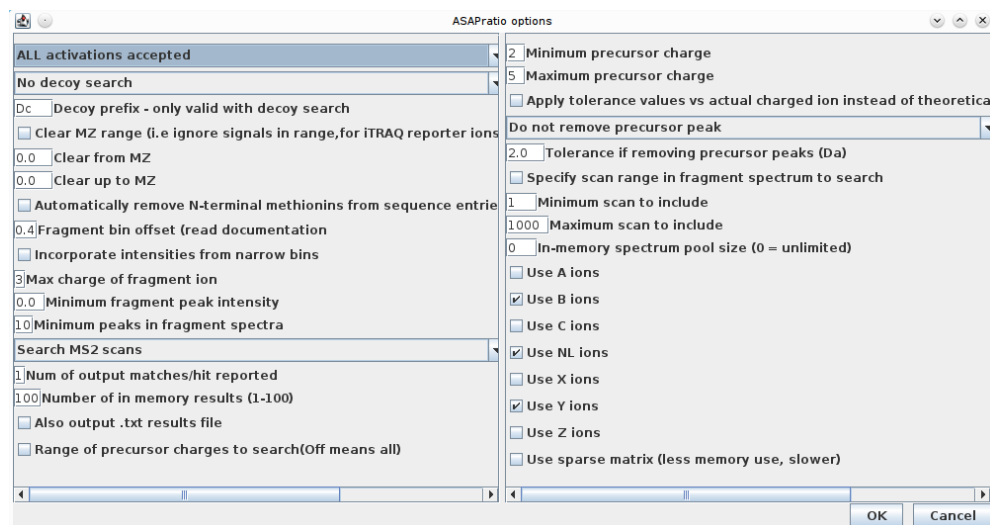


Figure 47: Comets advanced settings, experts can tweak away.

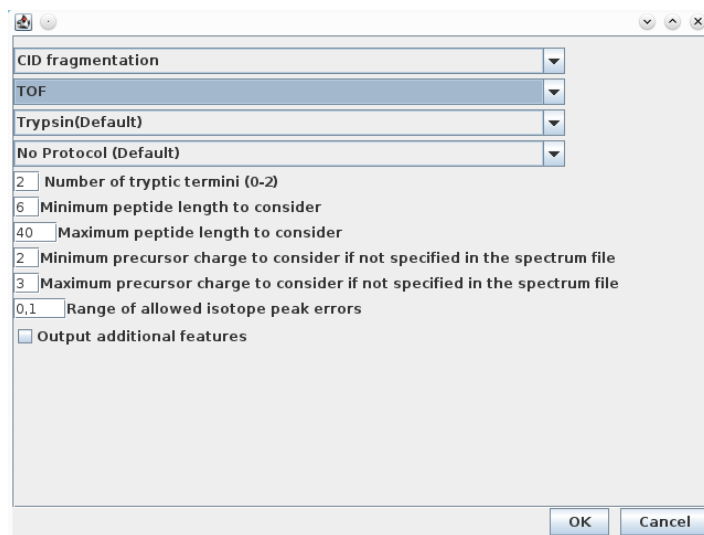


Figure 48: MSGFPlus search settings

- *Fragmentation type* Collision induced dissociation and Electron induced dissociation models are supported.
- *Instrument rule* Supported rules include high and low sensitivity ion trap instruments, along with Time of Flight instruments.
- *Enzyme used* Determines MSGFPlus expected cleavage rules.
- *Protocol setting* Protocols allow focused detection of various features such as phosphorylated peptides or iTRAQ tags.
- *Number of tryptic termini* Determines if non-enzymatic peptides are considered.

Table 10: InsPecT search plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
InsPecT plugin	InsPecT.exe msconvert.exe (Unless input is 32bit, non-zlibbed,non-gzipped mzXML)	mzML/mzML fasta gsp	pep.xml

### 8.5.1 InsPecT plugin

InsPecT integration does not yet implement unrestricted search.<sup>5</sup> search engine. Most settings are provided by a .gsp file from a settings plugin. !todo

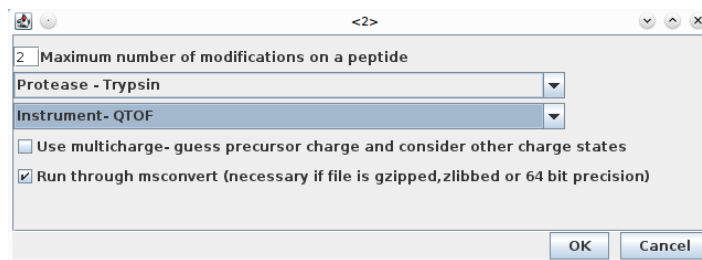


Figure 49: InsPecT setup window

### 8.5.2 OMSSA

Provides a plugin for the Open Mass Spectrometry Search Algorithm <sup>6</sup> search engine. Most settings are provided by a .gsp file from a settings plugin. OMSSA needs to be run using an MGF file and will be converted to such before being searched with the plugin, alternatively an MGF may be directly provided to the plugin.

Table 11: OMSSA search plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
OMSSA plugin	omssacl(.exe) msconvert(.exe) or mzxml2other(.exe) (If file is not a .mgf)	mzML/mzML/mgf fasta gsp	pep.xml

## 8.6 Spectral search engine plugins

**Spectral search** offers a complementary function to database search. It allows a department to store all identified peptides in spectral libraries, along with their identified spectra. These spectra can then be used in spectral searches, this is much faster than database-searches and incredibly sensitive. As such tracking of specific peptides through a multitude of experiments is greatly facilitated by using spectral libraries and spectral search can be performed with a wider mass-tolerance and will find any modified peptides from previously identified experiments.

<sup>5</sup><http://proteomics.ucsd.edu/Software>

<sup>6</sup>[http://www.ncbi.nlm.nih.gov/pubmed/15473683\(downloadpagenolongerup?\)](http://www.ncbi.nlm.nih.gov/pubmed/15473683(downloadpagenolongerup?))

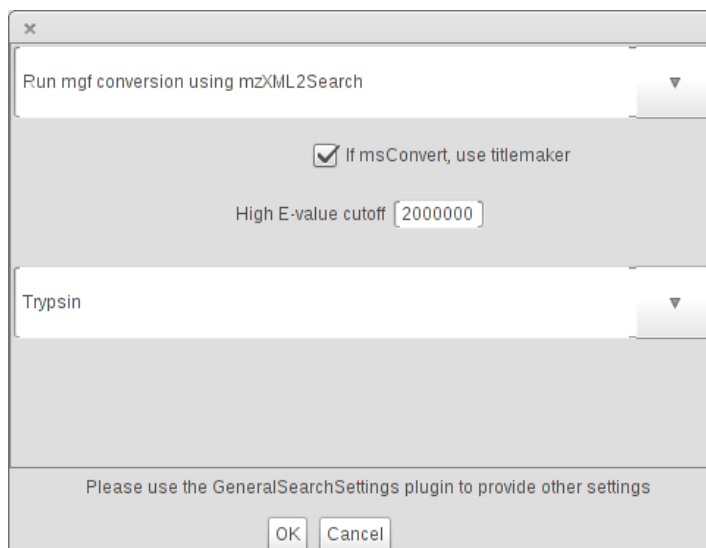


Figure 50: OMSSA setup window

### 8.6.1 SpectraST library builder plugin

SpectraST<sup>7</sup> is a powerful spectral search engine that offers a large array of options. For this purpose all options for creating and maintaining a set of spectral libraries have been split of into a separate plugin from the core search plugin. The plugin can handle creation of Spectral libraries from pep.xml files, if these are uploaded

Table 12: SpectraST spectral library builder plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
SpectraST spectral library builder plugin	spectrast(.exe)	pep.xml fasta gsp mzML/mzML	splib
Co			

from a remote source they need to have raw data uploaded along with them in mzML or mzXML format. Additionally, the plugin handles other splib files, allowing one to use SpectraST to combine multiple spectral libraries into a larger one, and to generate spectral decoys in the libraries. This makes it easier to utilize several previously established spectral libraries created in a multitude of experiments for a single spectral search.

Inputs to plugin: The plugin needs a pep.xml as input, from this it will attempt to retrieve the location of matching mzXML/mzML files. The GSP file is used to define modifications for inclusion in the spectral library, to ensure tha no exotic modifications are excluded. The .fasta file is used to map spectra to a database.

Inputting mzML/mzXML files directly to this plugin will mean these files are used rather than any files of matching names referenced in the pep.xml file and is an easy way to compensate for a broken reference in a file.

- *gsp file* From the general seach settings plugin. Any mods defined will be used to create a custom modification file and ensure they get imported.
- *Fasta database file* Used to map spectra to protein names
- *Pep.xml files* Need to already have been processed using either Peptide Prophet or iProphet. These will be used to create spectral libraries from any spectra exceeding the minimum probability cutoff.

<sup>7</sup><http://sourceforge.net/projects/sashimi>

- *Splib spectral library* All input spectral libraries will be combined into a single splib, along with any additional freshly created spectral libraries.

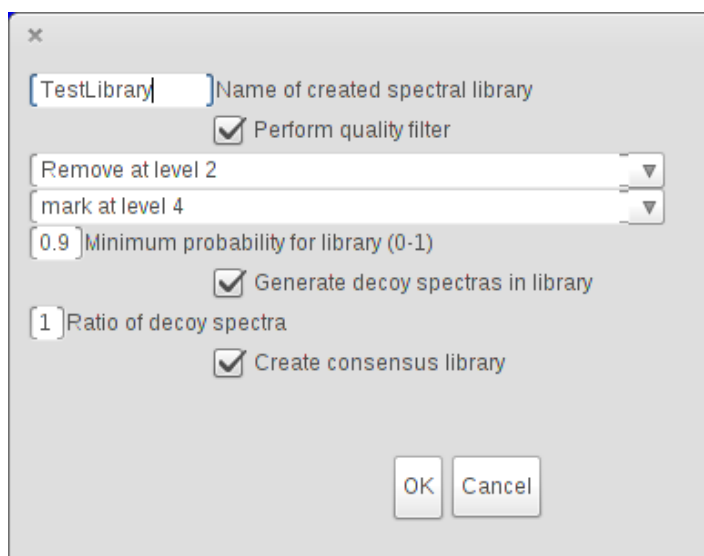


Figure 51: Spectral library builder setup window

- *Name of created spectral library:* The base name of the spectral library
- *Perform quality filter:* Run a quality filtering step on the created spectral library
- *Remove at level:* Spectra with a quality score at or below this level (1-5, with 5 being best) will be removed from the created spectral library.
- *Mark at level:* Spectra with a quality level will be marked in the library, this is a mean of keeping track of less than perfect spectra.
- *Minimum probability for library:* Only spectra at or above this Peptide Prophet or iProphet probability will be included into the library.
- *Generate decoy spectra in library:* The final library will have decoy spectra generated. These spectra will have an associated protein name starting with "DECOY".
- *Ratio of decoy spectra:* Ratio of decoys to real spectra in the final splib.
- *Create a consensus library:* Will create a consensus library, keeping a merged spectra from all available spectra for each peptide. This is necessary for decoy generation.

### 8.6.2 SpectraST search plugin

SpectraST search plugin performs searches against created spectral libraries.

Table 13: SpectraST spectral search plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
SpectraST spectral search plugin	spectrast(.exe)	fasta gsp mzML/mzML splib	pep.xml

Table 14: SpectraST spectral search plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
PeptideProphet plugin	xinteract(.exe)	pep.xml	pep.xml
	InteractParser(.exe)	fasta (remaps pep.xml)	
	PeptideProphetParser(.exe)	mzXML/mzML(remaps pep.xml)	
	InterProphetParser(.exe)		
	ProteinProphetParser(.exe)		
	Additional TPP provided parsers		

## 8.7 Data processing

### 8.7.1 PeptideProphet plugin

The name is somewhat misleading since the plugin actually interacts with the xinteract executable and as such can invoke several other tools, including Libra (iTRAQ), Peptide Prophet, Protein Prophet, PTMProphet etc. Most of these have their own settings under the Peptide Prophet plugin GUI.

This plugin is the first step in processing search result from any of the search engines supported by APP. The options for Xinteract are extensive, as such the best place for info on all options is at the TPP wiki at: [http://tools.proteomecenter.org/wiki/index.php?title=Main\\_Page](http://tools.proteomecenter.org/wiki/index.php?title=Main_Page). The most common options available are covered here.

- *Minimum probability peptides to keep:* Peptide Specrum Matches with a lower probability than this will be filtered out of resulting pep.xml files.
- *Minimum lenght peptide:* As above but but for peptide length (No. of amino acids)
- *Use decoy hits to estimate correct peptides:* Peptide prophet will use hits from known decoys to calibrate it's internal null distribution. For most search engines this should be on.
- *Use non-parametric model:* Peptide prophet will use disregard it's preset parameters totally and rescore from decoy hits.
- *Experiment name:* Experiment names tag spectra and are used by iProphet in the "Number of replicate experiments" model. As such samples that are considered unique should have different experiment names if there is interest in combining them later. Note that the same data searched with different search engines should retain the same experiment name.
- *Ignore charge states:* Peptide prophet will ignore spectra that have precursor in these charge states analysing data.
- *Use command line instead of GUI options:* This will feed the contents of the text field below directly to the xinteract command. It allows use of more advanced xinteract options if not available through the GUI.

- *Enzyme list (default Trypsin)*: Provides interactParser with info on which enzyme is used, this is also used for later analysis.
- *Libra settings*: Brings up settings for iTRAQ analysis. For details see tutorial section.
- *ASAPRatio settings*: Settings for analysing isotopically labeled samples.
- *XPRESS settings*: Secondary utility to handle isotopically labelled samples, simpler settings but less advanced than ASAPRatio.
- *iProphet options*: Used to enable iProphet analysis and more options. Also set options for using PTM-Prophet on data, for more info on these options see the iProphet and PTMProphet plugins.

### 8.7.2 iProphet plugin

iProphet<sup>8</sup> provides a powerful meta analysis tool, allowing combination and analysis of results from several experiments, search engines or samples. Settings are used to enable or disable iProphet's scoring models, for more info on these please see the iProphet source paper. This plugin is the fir

Table 15: iProphet plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
iProphet plugin	InterProphetParser(.exe)	pep.xml	pep.xml

## 8.8 Protein prophet plugin

Utilize the Protein Prophet<sup>9</sup> tool to create protein lists for finished experiments. Can also import results from LIBRA,XPRESS or ASAPRatio to show protein level quantification.

Table 16: ProteinProphet plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
ProteinProphet plugin	ProteinProphet(.exe)	pep.xml	prot.xml

## 8.9 Smaller utilities

### 8.9.1 Spectrum name fixer plugin summary

Table 17: Spectrum name fixer plugin

Plugin name	Needed binaries	Accepted input formats	Available output formats
Spectrum name fixer	None	pep.xml mzxml/mzml	pep.xml

Spectrum name fixer will handle differences with regards to search engine output not reporting the same spectrum names, such as in the case of *Myrimatch* and *MSGFPlus* where spectrum Index is reported rather

<sup>8</sup><http://sourceforge.net/projects/sashimi>

<sup>9</sup><http://sourceforge.net/projects/sashimi>

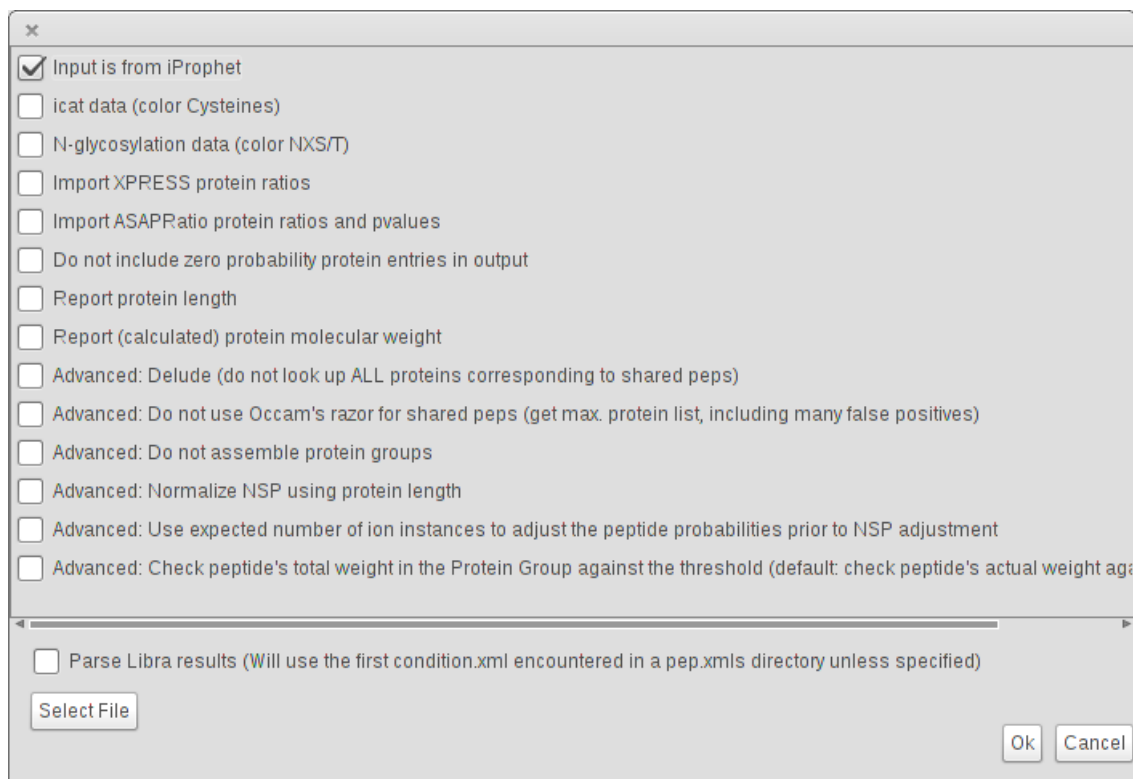


Figure 52: ProteinProphet plugin settings window

than Spectrum number. As such any pep.xml output from MSGFPlus and Myrimatch need to be processed through this plugin before using iProphet or any quantitation tools other than spectral counting. Raw data files are parsed to match index to spectrum number, if data files have been moved since the search they can be set as input for the plugin. The plugin has no settings, merely feed it a pep.xml file from IDConvert (MSGFPlus) or from Myrimatch.

### 8.9.2 Label Free Data Extractor

The label free data extractor is a utility for extracting information from pep.xml/prot.xml files. The plugin performs basic quantitation, such as spectral counting and extraction of Total Ion Current for proteins. The plugin has a number of filter settings to determine what proteins, peptides and peptide spectrum matches get included in amount estimations.

Plugin name	Needed binaries	Accepted input formats	Available output formats
Label Free Data Extractor plugin	None	pep.xml	proteins.txt
		prot.xml	peptides.txt
		pep.xml	hits.txt
			combined.txt

- *Create amount estimations from TIC values* Will parse source mzXML/mzML files to get the "Total Ion Current" for each spectra. This will be used to calculate a total and an average for each protein, giving a label free way of estimating amount.
- *Export spectral count* Will count all spectra that pass filtering for each peptide and protein.



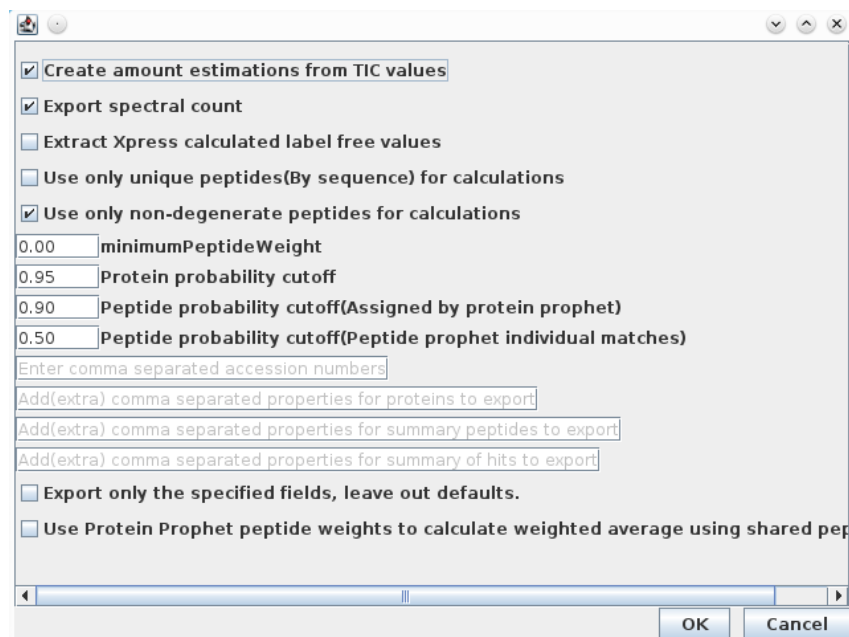


Figure 53: Label Free Data Extractor plugin settings window

- *Extract Xpress calculated label free values* If the file has been analyzed using Xpress label free settings, you can extract the values into a more convenient text file.
- *Use only unique peptides* Will keep only peptides with a unique unmodified sequence, i.e. NO shared peptides will be included in calculations.
- *Use only non-degenerate peptides* Filter out all peptides that are marked as non-degenerate by Protein-Prophet.
- *Minimum peptide weight* Keep only peptides assigned above this peptide weight.
- *Minimum protein probability* Keep only peptides and PSMs from proteins with this or higher Protein-Prophet probability.
- *Minimum peptide probability* Keep only peptides with a ProteinProphet assigned probability over this value.
- *Minimum peptide probability (Peptide prophet individual matches)* Minimum iProphet/Peptide Prophet probability for individual spectrum matches to be included.
- *Comma separated accesssion numbers/protein names* Only proteins matching these will be considered.
- *Protein properties* Any property stored in the prot.xml can be exported by adding it here. Add the names with comma separation.
- *Extra properties to export, these are properties stored in the prot.xml*
- *Extra properties for hits to export* Properties of PSMs from peptide prophet, i.e. "precursormass" or "charge" etc.
- *Export only extra fields* Will ignore standard fields such as spectrum,protein name etc and instead only export properties specified by the user.

- *Use protein prophet weights to calculate weighted averages for shared peptides* Will use Proteinprophet assigned peptide weights to distribute spectrum counts/TIC between proteins with shared peptides.

### 8.9.3 LibraProteinRatioParser

Allows running LibraProteinRatioParser on prot.xml files, calculating protein ratios.

Plugin name	Needed binaries	Accepted input formats	Available output formats
LibraProteinRatioParser plugin	LibraProteinRatioParser(.exe)	prot.xml	prot.xml

- *Condition file* The condition file to use when calculating ratios.

### 8.9.4 Spectractor

The Spectractor plugin uses wkhtml2pdf and the Comet spectrum viewer (NOT the search engine) to extract PDF images of spectra from a prot.xml.

Plugin name	Needed binaries	Accepted input formats	Available output formats
Spectractor plugin	wkhtmltopdf(.exe) comet spectrum viewer (on server)	prot.xml	pdf

- *Condition file* The condition file to use when calculating ratios.

filtering options match those of Label Free Data Extractor with a few addition. Only the extra options will be covered here:

- *cgi bin directory prefix on server* Typically will be tpp-bin for a server running on Windows and cgi-bin for a server on linux.
- *Keep only top spectrum per peptide* Will only export a pdf of the top ranking PSM for each peptide.
- *Extract only proteins with one peptide* Many publications demand the submission of spectra from single peptide hit, this allows an easy way to extract only spectra from such proteins.

### 8.9.5 Libra Normalizer

Libra Normalizer will normalize all Libra iTRAQ channels to contain exactly the same total intensity. This is a good way to compensate for pipeting errors and other sample preparation errors. Typically the normalization factors will be very close to 1, larger or smaller factors indicate often indicate a problem somewhere during sample preparation.

Plugin name	Needed binaries	Accepted input formats	Available output formats
LibraNormalizer plugin	non	pep.xml	normalized.pep.xml

Normalized pep.xmls can then be used together with the ProteinProphet and LibraProteinRatioParser plugins to generate final normalized quantities.

Plugin name	Needed binaries	Accepted input formats	Available output formats
Input file feeder plugin	None	ANYTHING	THE SAME ANYTHING

### 8.9.6 Input file feeder

The simplest of plugins, any file input into ***Input file feeder*** plugin will merely be output. As such it can be used to give the same input to a large amount of other plugins by linking them. Useful to provide the same data to multiple search engines for example.

### 8.9.7 Mascot2XML plugin

Mascot2XML is a TPP provided converter which allows conversion from Mascots .dat files into pep.xml files. A typical usage looks like this:

1. Convert data into mzXML/mzml
2. Use MsConvert (with titlemaker option) or mzxml2other to create mascot compatible MGF files.
3. Search files on mascot
4. Download .dat file from mascot.
5. Create a Mascot2XML plugin and provide it with the following: the .dat file, a .fasta database file, all the original mzXML/mzML files.
6. feed the output pep.xml files to a PeptideProphet plugin.

### 8.9.8 IDConvert

IDConvert is a ProteoWizard provided tool for conversion between MSMS results formats pep.xml/prot.xml and MZIdent, it also provides output as it's own internal text format. The main use of IDConvert within APP is providing a way to convert MSGFPlus derived mzID files into usefule pep.xml files.

Plugin name	Needed binaries	Accepted input formats	Available output formats
IDConvert	idconvert(.exe)	pep.xml prot.xml  mzid	mzid pep.xml txt

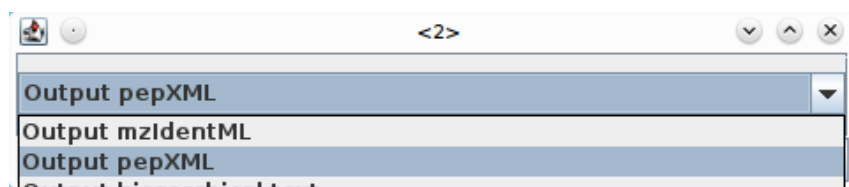


Figure 54: The single setting of idconvert

Note that conversions for searches done with mzXMLs will not work, and some conversions from mzXMLs converted into mzMLs also fails. It is best to keep the whole pipeline using mzML if possible.