



ScaffoldBatch 3 User Documentation
Proteome Software, Inc.

Version 1.8 September, 2011

ScaffoldBatch version 3.2

© 2011 Proteome Software, Inc

Table of Contents

File Reference Nomenclature

ScaffoldBatch 3 Functionality

Typical Workflow

Installation

Running ScaffoldBatch 3

Return Codes

Error log files

Monitoring progress

ScaffoldBatch 3 and FASTA databases

ScaffoldBatch 3 and X! Tandem

Sample XML Driver File

Annotated XML Driver File

ScaffoldBatch 3 protXML

protXML Example

File Reference Nomenclature

Throughout this manual, you will see file types referred to. For example you might read *.sf3 as a Scaffold 3 file. Please note that in this manual this same file type can be written as SF3 as well. Similarly, a Mascot file could be written either *.dat or DAT. These are synonymous.

We refer to a ScaffoldBatch XML driver file as that file that runs ScaffoldBatch when called from the command line. In Scaffold 3, you can export a Scaffold XML driver file, and this is called a *.scafml (SCAFML) file. Therefore, a Scaffold XML driver file is synonymous with a SCAFML file in this manual.

ScaffoldBatch 3 Functionality

ScaffoldBatch 3 is a batch version of Scaffold 3. It can load and analyze the same data that Scaffold 3 does, but in a batch rather than interactive environment.

Batch mode means that ScaffoldBatch 3 can be run on the command line or called from a batch script. The intended use is for organizations that want to integrate Scaffold 3 into a proteomics pipeline. ScaffoldBatch 3 can be used as one component of an automated proteomics workflow. For organizations that want a turnkey proteomics pipeline, Genomics has integrated Scaffold 3 into their Proteus Analytics package.

ScaffoldBatch 3 is an extended version of Scaffold 3. When you install ScaffoldBatch 3, a copy of the interactive version of Scaffold 3 is automatically installed as well. Like Scaffold 3, ScaffoldBatch 3 is locked to one computer by a software key.

As a command line batch program, ScaffoldBatch 3 is intended to be called from a batch script. In the Microsoft world this might be a *.bat file. In the Linux world this might be a *.sh file. ScaffoldBatch 3 is driven by an XML file. A sample driver XML file is shown in the next section and the parameters for the XML file are defined following that.

This first version of ScaffoldBatch 3 has inherited some of the restrictions of Scaffold 3. Since parameter files and fasta database indexes may be changed as ScaffoldBatch 3 runs, it is not safe to run more than one copy of ScaffoldBatch 3 at one

time on one computer. To prevent clashes, ScaffoldBatch 3 will exit immediately upon startup if it detects that another version of Scaffold 3 is running. There are a few features which can be accessed only by Scaffold 3, not ScaffoldBatch 3. For this reason it is wise to install ScaffoldBatch 3 on a Linux, Windows or Apple computer that can also run the interactive version of Scaffold 3.

ScaffoldBatch 3 requires at least 1GB of RAM to run. With this much memory ScaffoldBatch 3 can process approximately 1,000,000 to 2,000,000 spectra (depending on the users system). If the spectra were searched against multiple search engines, say Sequest and X! Tandem, then ScaffoldBatch 3 can deal fewer spectra than if only one search engine is involved. Datasets having over 250,000 spectra in a single MuDPIT sample also stress ScaffoldBatch 3. Some installations wanting to run larger datasets have run Scaffold 3 using more than 4GB of RAM.

ScaffoldBatch 3 also creates temporary files in the system TEMP directory. These files may be 1GB or larger.

This documentation assumes that you are already familiar with the interactive version of Scaffold 3. Many of ScaffoldBatch 3's features and options are documented here by referring to the corresponding feature in Scaffold 3. For an introduction to the interactive version of Scaffold 3, refer to the Scaffold 3 User Guide available from http://www.proteomesoftware.com/proteome_software_Scaffold_sample_data.html. The details of the options in Scaffold 3 are available from the on-line help. You can read the online help by running Scaffold 3, the interactive version on the computer where ScaffoldBatch 3 is installed, or by installing and running the free Scaffold 3 viewer on any computer.

Typical Workflow

A typical proteomics workflow takes data from a 1) mass spectrometer, 2) extracts the fragment peaks, 3) optionally processes the data for noise reduction and deisotoping, 4) searches the spectra against a FASTA database with Sequest, Mascot or X! Tandem, 5) analyzes the data with Scaffold 3 to validate the proteins identified, and then 6) loads the data into a database. Step five in this workflow, the ScaffoldBatch 3 program, also creates a *.sf3 file that can be used to further analyze and view the data.

Anyone who wants to automate the proteomics workflow will want to create a system that performs all these steps. In particular for the Scaffold 3 portion of the workflow, they will have to create a program that

- 1) creates the XML file that drives ScaffoldBatch 3,
- 2) schedules ScaffoldBatch 3 to run only when all its input files are available,
- 3) waits until no other copy of Scaffold 3 or ScaffoldBatch 3 is running before launching ScaffoldBatch 3,
- 3) monitors and reports the progress of ScaffoldBatch 3, and
- 4) checks the return code of ScaffoldBatch 3 and handles any errors.

The *.sf3 files created by ScaffoldBatch 3 can be read by Scaffold 3 or by the free Scaffold 3 Viewer. Anyone can download the Scaffold 3 viewer from http://www.proteomesoftware.com/Proteome_software_prod_Scaffold3_download-main.html.

The protXML files created by ScaffoldBatch 3 are intended to be loaded into a database or other software tool. Scaffold 3's version of protXML is an extension of the XML developed at the Institute for System Biology as part of their Trans Proteomic Pipeline. ProtXML is documented at the website www.proteomecenter.org and Scaffold 3's extensions are documented below.

Installation

To install ScaffoldBatch 3, begin by downloading the Scaffold 3 Program from:
http://www.proteomesoftware.com/Proteome_software_prod_Scaffold3_download-main.html

Click on the *Install_Scaffold 3.exe* file to install both Scaffold 3 and the ScaffoldBatch 3 program. ScaffoldBatch 3 offers you an opportunity to choose the destination directory for installation. The default location in Windows XP is the Program Files directory.

Opening and installing Scaffold 3 requires a special software key. To receive your batch key, email info@proteomesoftware.com or call (800)-944-6027. Note that ScaffoldBatch 3 will not work if you have a Scaffold 3 evaluation key or a regular purchased Scaffold 3 key.

The software key can be installed by starting Scaffold 3 in the interactive mode, or by running ScaffoldBatch 3 at the command line. Follow the screen directions to insert the ScaffoldBatch 3 key and complete installation. The ScaffoldBatch 3 key will supercede all Scaffold 3 keys that are previously installed on that computer. Since the ScaffoldBatch 3 key unlocks Scaffold 3 as well, there is no conflict.

Running ScaffoldBatch 3

The following file types are standard inputs and outputs from ScaffoldBatch 3:

Inputs – Scaffold 3 *.sf3 (or *.sfd)

SEQUEST® = *.dta and *.out, *.ms2 and *.sqt, and *.srf (BioWorks); Mascot® = *.dat; X! Tandem® = *.xml; Phenyx = *.scaffold-pidres.xml; OMSSA = *.omx; Waters® IdentityE = *.xml (Note: These files must be created from our plugin in the new version of Waters PLGS); Proteome Discoverer = *.msf

FASTA database files

Outputs – Scaffold 3 *.sf3 file

protXML.xml file

An XML driver file specifies these files and the parameters used to run ScaffoldBatch 3. Example XML driver files and a specification for these files are given later.

To call ScaffoldBatch 3 from the command line or from a batch program:

ScaffoldBatch3.exe [-f] [-q] [-keypath PATH] [xmlDriverFile]

xmlDriverFile	This parameter is an XML file with commands which tell ScaffoldBatch 3 which files to load. See the sections below “Sample XML Driver File” and “Annotated XML Driver File”.
---------------	--

Runtime options for the command line

-f	Force mode, don't request answers to questions from command line. This option will normally be used in all batch operations to keep ScaffoldBatch 3 from stopping to wait for input.
-q	Quiet mode, write minimal information to standard output
-keypath PATH	Keypath is the path to the license key. Here “keypath” is the keyword and PATH is the fully qualified directory path to the license key file. This is only needed if the license key is not in the default Scaffold 3 directory. An instance where this might be useful is for grid systems.

The normal sample command line will look like this---

ScaffoldBatch3.exe -f driver1.xml

ScaffoldBatch 3 prints hundreds or thousands of lines to the system standard output. It may be useful to pipe this to a monitoring program. See the Monitoring Progress section below.

Allocating Memory

By default, ScaffoldBatch 3 uses a minimum of 512MB of RAM. This is often not enough memory to handle large datasets. If more memory is available on the computer, you can tell ScaffoldBatch 3 to use the more memory by setting the java virtual memory options. To set the virtual memory available for Scaffold 3, edit the file "ScaffoldBatch 3.vmoptions". This file is found in the Scaffold 3 install directory.

The ScaffoldBatch 3.vmoptions file will have a single line that looks like

```
-Xmx512m
```

This tells java to allocate 512 megabytes to Scaffold 3.

To allocate one gigabyte, use "-Xmx1g" or "-Xmx1024m".

To allocate four gigabytes, use "-Xmx4g". To allocate 8 gigabytes, use "-Xmx8g".

Scaffold 3 will not run reliably on less than 512 MB. If you allocate more than 2GB, the default 32-bit java runtime environment (JRE) will not be able to use the memory (See Java Runtime Environment below). When allocating memory to Scaffold 3 remember to leave some memory for the operating system and other applications.

Java Runtime Environment

The java runtime environment (JRE) shipped with Scaffold 3 is the 32-bit version. If you want to use more than 2GB of RAM, then you may want to download and install the 64-bit version of Scaffold 3. You must have a 64-bit computer with a 64-bit operating system to use Scaffold 3 64-bit. Scaffold 3 will use this RAM only if you have set the memory allocation – see the section above "Allocating Memory".

Return codes:

ScaffoldBatch 3 returns a return code of 0 if successful and 1 if unsuccessful.

Error log files:

As ScaffoldBatch 3 executes it echoes the output log and the error log to the standard system console or standard out. It also writes this information to output_log.txt and error_log.txt. These files can be found in the directory where Scaffold 3 and ScaffoldBatch 3 are installed. On a Windows installation this is typically in the C:\Program Files directory.

The output_log.txt file is a record of the most recent execution of Scaffold 3 or ScaffoldBatch 3. It includes numerous lines monitoring the progress of Scaffold 3. It is overwritten each time Scaffold 3 or ScaffoldBatch 3 starts.

The error_log.txt file is a cumulative record of all errors that Scaffold 3 or ScaffoldBatch 3 have encountered. For debugging purposes, most of these errors are recorded with the java system stack trace. Stack traces can make a small error seem rather fierce.

Monitoring progress:

As ScaffoldBatch 3 executes it prints its progress to the standard out file. Lines of the form

%9.47%

%9.59%

%9.7%

%9.8%

%9.91%

%10.01%

%10.12%

%10.24%

This output is provided so that the calling pipeline program can monitor this output stream and report the progress to the users. The numbers are the percent complete.

ScaffoldBatch 3 and FASTA databases

When ScaffoldBatch 3 runs, one of the parameters tells it which FASTA database to use to supply the protein sequence coverage and to use if ScaffoldBatch 3 runs X! Tandem. ScaffoldBatch 3 also needs to access an INDEX file for each FASTA database it uses. This INDEX may be present already or it may be created when ScaffoldBatch 3 runs.

This index may be created in the interactive Scaffold 3 (Scaffold Desktop GUI) by using the menu option accessed by selecting Edit > Edit FASTA databases to get the Configure Database Parser or it may be created by specifying parse rules in the XML driver file for ScaffoldBatch 3.

The behavior of ScaffoldBatch 3 when dealing with FASTA databases and INDEX files is as follows:

- if the XML driver contains parse rules, they will be used both in reading the data and in applying the protein sequences. If there is already an INDEX file, it will be replaced.
- if the XML driver contains no parse rules but the database has been previously indexed (either in Batch or GUI) the existing index will be used in both phases.
- if no parse rules are included in the XML driver and the database has not been indexed already, Scaffold will make its best guess at a correct parse rule or use a generic rule to index the FASTA database.

If the FASTA file has non-standard header lines, you must specify parsing rules for the accession number and protein description. These parsing rules can be created and used in the interactive version of Scaffold 3.

The parsing rules can also be specified in the XML driver file for ScaffoldBatch 3. The format of these parsing rules is that of regular expressions. An example is

<FastaDatabase

```
id="swissprot"
```

```
path="/Users/<user>/Documents/WorkFiles/control_sprot.fasta"
```

```
name="Generic"
```

```
databaseAccessionRegEx=">([ ^ ]*)"
```

```
databaseDescriptionRegEx=">[ ^ ]* (.*)"
```

```
/>
```

The use of back slashes to escape characters in the version of regular expression parsing is different than that used in the Mascot regular expression parser. Here are the parsing rules that the GUI version of Scaffold 3 provides as options.

name	databaseAccessionRegEx	databaseDescriptionRegEx
EST/NR (NCBI)	>(gi [0-9]*)	>[]* (.*)
IPI (EBI)	>IPI:([^\]*)	>[]* Tax_Id=[0-9]* (.*)
Swiss-Prot (SIB/EBI)	>([]*)	>[]* \([]*\) (.*)
UniProt/Swiss-Prot (UniProtKB)	>[]*\([]*	>[]*\([]* (.*)
UniRef/NREF (UniProt)	>UniRef100_([]*)	>[]* (.*)
Ensembl (EMBL/EBI)	>(ENS[]*)	>[]* (.*)
MSDB (Proteomics Group)	>([]*)	>[]* (.*)
Swiss-Prot Alt. Accession (SIB/EBI)	>[]* \([]*\)	>[]* \([]*\) (.*)
UniProt/Swiss-Prot Alt. Accession (UniProtKB)	>([^\]*)	>[]*\([]* (.*)
Generic	>([]*)	>[]* (.*)

Sequest uses the parsing rules listed here as "Generic". The parsing specified does NOT override the parsing in an existing index file.

ScaffoldBatch 3 and X! Tandem

Some Scaffold 3 users find it useful to search their Sequest or Mascot files a second time using X! Tandem. X! Tandem can be run within ScaffoldBatch 3 or externally as a standalone program.

There are two advantages to running X! Tandem within ScaffoldBatch 3. The first is that it is convenient. The second is that since Scaffold 3 controls X! Tandem it can easily merge the X! Tandem results back into the Sequest or Mascot results already loaded.

The main reason to run X! Tandem separately is that X! Tandem standalone can process many files simultaneously on a multi-threaded computer or computer cluster. A secondary reason is that this avoids some X! Tandem crashes that occur occasionally when X! Tandem is called from within Scaffold 3. You can download and install X! Tandem for free from www.thegpm.org. If you want to do a complex installation of X! Tandem you may want to contract for support with Beavis Informatics.

The main disadvantage of running X! Tandem standalone is that this makes ScaffoldBatch 3 work much harder to match the results from X! Tandem with the corresponding results from Sequest or Mascot. This is a very RAM intensive part of ScaffoldBatch 3 and may be a bottleneck in analyzing your data.

If you do decide to run X! Tandem standalone, there are certain parameters that you need to set in X! Tandem to make it compatible with Scaffold 3.

The main things you need are extra specifications (tags) in the default.xml/input.xml file. These XML files tell X! Tandem how to analyze the dataset. The extra tags we require revolve around a) printing out publication information and b) reporting at least one match for every spectrum for statistical analysis. As such, the following tags are required:

This tag requires X! Tandem to include enough information so that we can line up spectra from different search engines:

1. <note type="input" label="spectrum, dynamic range">1000.0</note>

These tags force X! Tandem to search every spectrum and to not modify the spectrum:

1. `<note type="input" label="spectrum, use noise suppression">no</note>`
2. `<note type="input" label="spectrum, minimum parent m+h">0.0</note>`
3. `<note type="input" label="spectrum, minimum fragment mz">0.0</note>`
4. `<note type="input" label="spectrum, minimum peaks">0</note>`

This tag requires X! Tandem to score everything it "identified":

1. `<note type="input" label="scoring, minimum ion count">0</note>`

These tags make X! Tandem put out the publication information and parameters we require:

1. `<note type="input" label="output, parameters">yes</note>`
2. `<note type="input" label="output, performance">yes</note>`

This tag requires X! Tandem to include the MS/MS spectra:

1. `<note type="input" label="output, spectra">yes</note>`

And this tag requires X! Tandem to provide results for every single spectrum (necessary for statistical analysis):

1. `<note type="input" label="output, results">all</note>`
2. `<note type="input" label="output, maximum valid expectation value">1000</note>`

For more information on the X! Tandem API see <http://www.thegpm.org/TANDEM/api/index.html>.

Sample XML driver file

ScaffoldBatch 3 requires an XML driver file to define its operational parameters. This documentation provides two sample XML driver files and an annotated list of the tags in the XML driver file.

Here is an example of a minimal Scaffold 3 XML file:

```
<Scaffold>
<Experiment name="sb_test_1">
<FastaDatabase id="swissprot" path="C:\sprot.fasta"/>
<BiologicalSample database="swissprot" >
    <InputFile>C:\control_01.dat</InputFile>
</BiologicalSample>
<BiologicalSample database="swissprot">
<InputFile>C:\control_02.dat</InputFile>
</BiologicalSample>
    <DisplayThresholds name="DTs" id="thresh" proteinProbability="0.95"
        minimumPeptideCount="2" peptideProbability="0.95"/>
    <Export type="sfd" thresholds="thresh"
        path="C:\sb_test_1.sfd"/>
    <Export type="protxml" thresholds="thresh"
        path="C:\sb_test_1_protxml.xml"/>
</Experiment>
</Scaffold>
```

Here is a more complete Scaffold 3 XML driver file:

```
<Scaffold>
<Experiment name="Demo"
    description="An Example Experiment"
    load="Demo_existing.sfd"
    peakListGeneratorName="Bioworks"
    peakListGeneratorVersion="3.2"
    peakListDeisotoped="false"
    showStatisticsPane="false"
    password="foo"
    protectThresholds="true"
    protectDisplaySettings="true"
    protectExportSpectra="true"
    analyzeWithTandem="true"
    condenseDataWhileLoading="true">
<FastaDatabase
    id="swissprot"
    name="swissprot"
    databaseAccessionRegEx=">([ ^ ]*)"
    databaseDescriptionRegEx=">[ ^ ]* (.*)"
    path=" ../Sprout_human.fasta"/>
<Modification unimodName="HNE"/>
<Modification
    fullName="testMod"
    referenceName="a test mod"
    monoMass="42"
    averageMass="42"
    minoAcidsModified="A"/>
<BiologicalSample
    analyzeAsMudpit="true"
    database="swissprot"
    name="Demo_01"
    category="Category 1">
    <InputFile>Demo_01.dat</InputFile>
    <InputFile>Demo_01.tar.gz</InputFile>
</BiologicalSample>
```

```

<BiologicalSample
  analyzeAsMudpit="true"
  database="swissprot"
  name="Demo_02">
  <InputFile>Demo_02.dat</InputFile>
  <InputFile>Demo_02.mgf.xml</InputFile>
  <InputFile>Demo_02.tar.gz</InputFile>
</BiologicalSample>
<BiologicalSample
  analyzeAsMudpit="true"
  database="swissprot"
  name="Demo_03">
  <InputFile>Demo_03.dat</InputFile>
  <InputFile>Demo_03.mgf.xml</InputFile>
  <InputFile>Demo_03.tar.gz</InputFile>
</BiologicalSample>
<DisplayThresholds
  name="Some Thresholds"
  id="thresh"
  proteinProbability="0.95"
  minimumPeptideCount="2"
  peptideProbability="0.95"
  minimumNTT="1"
  useCharge="true,true,true"/>
<Export
  type="sfd"
  thresholds="thresh"
  saveOnlyIdentifiedSpectra="true"
  saveNoSpectra="true"
  path="."/>
<Export
  type="protxml"
  thresholds="thresh"
  path="Demo_foo.protxml"/>
</Experiment>
</Scaffold>

```

Here is an example of a file merge xml driver file:

```
</Scaffold>
  <Experiment
    name="control_experiment"
    description="An Example Experiment"
    file="/Scaffold 3/Scaffold 3 Analysis/11-19-07/2_out.sfd"
    merge="/Scaffold 3/Scaffold 3 Analysis/11-19-07/1_out.sfd">
  </Experiment>
</Scaffold>
```

(Merge's 11-19-07/2_out.sfd file with 11-19-07/1_out.sfd)

See below for a fully annotated ScaffoldBatch XML driver file (SCAFML example file).

Annotated XML Driver File

The following XML driver file includes references to the interactive version of Scaffold 3 for most tags and parameters. To learn more about the functionality associated with the XML instructions, please refer to the referenced screens in Scaffold 3, and the available user documentation included with your Scaffold 3 installation.

XML Code	Associated Functionality
<pre data-bbox="167 451 969 1449"> <Scaffold> <Experiment name="Demo" description="An Example Experiment" load="Demo_existing.sfd" peakListGeneratorName="Bioworks" peakListGeneratorVersion="3.2" peakListDeisotoped="false" showStatisticsPane="false" </pre>	<p data-bbox="985 451 1451 688">The pair of tags <Scaffold> to </Scaffold> brackets all the other specifications in the xml file. It is common to add the version of Scaffold here, e.g., <Scaffold version="Scaffold_3_00_04"></p> <p data-bbox="985 737 1451 821">The pair of tags <Experiment> to </Experiment> brackets the rest of the specification.</p> <p data-bbox="985 869 1338 926">Menu Experiment / Edit Experiment</p> <p data-bbox="985 1024 1240 1052">Menu File / Open</p> <p data-bbox="985 1150 1256 1178">Publications Page</p> <p data-bbox="985 1360 1354 1388">Menu Edit / Preferences</p>
<pre data-bbox="167 1486 969 1701"> condenseDataWhileLoading = "true" password="foo" protectThresholds="true" protectDisplaySettings="true" </pre>	<p data-bbox="985 1486 1451 1543">Sample load wizard / Condense data while loading checkbox</p> <p data-bbox="985 1591 1354 1619">Menu Edit / Preferences</p>

<pre> trancheUsername="username" tranchePassword="password" analyzeWithTandem="true" analyzeWithSubsetDB="true" </pre>	<p>Publish view, upload file to tranche</p> <p>New Biological Sample Wizard</p>
<pre> highMassAccuracyScoring="true" useIndependentSampleGrouping="true" connectToNCBI="true" annotateWithGOA="true"> </pre>	<p>Turn on High Mass Accuracy Scoring</p> <p>Turn on Independent Sample Grouping. Default is "false"</p> <p>Menu Experiment / Add NCBI Annotations (Note: This option is set by default to "false")</p> <p>Menu Experiment / Add GO Annotations, which annotates with GOA database</p>
<pre> Merge="/location of sfd file/file.sfd" <FastaDatabase id="swissprot" name="Generic" databaseAccessionRegex=">[^]*" databaseDescriptionRegex=">[^]* (.*)" decoyProteinRegex="REVERSE RANDOM -R ##" path="../Sprot__human.fasta"/> </pre>	<p>Menu File / Merge with SFD.</p> <p>If the XML driver contains parse rules, they will be used both in reading the data and in applying the protein sequences;</p> <p>If the XML driver contains no parse rules but the database has been previously indexed (either in Batch or GUI) the existing index will be used in both phases;</p> <p>And, finally, if no parse rules are included in the XML driver and the database has not been indexed already, Scaffold will make its best guess at a correct parse rule or use a generic rule.</p>

```
<Modification unimodName="HNE"/>
```

Select a predefined modification.

Load Biosample Wizard

Only need to specify extra modifications beyond those in the loaded Sequest or Mascot file. These extra modifications will be added onto the modifications used by Sequest or Mascot. You only need to specify modifications when doing X! Tandem analysis. If not doing X! Tandem analysis, then no extra modifications need be specified. All modifications specified will apply to all data processed by X! Tandem.

The list of unimodNames for standard modifications in the UNIMOD list can be seen in the Load BioSample wizard.

```
<Modification fullName="testMod"
  referenceName="a test mod"
  monoMass="42"
  averageMass="42"
  aminoAcidsModified="A"/>
```

Create a new modification not in unimod. For use with X! Tandem search when the modification is not one of the standard ones in the UNIMOD list.

From the "Analyze Data with X! Tandem" screen, choose the New button

For N- and C-termini modifications, use lower case "n" and "c. "

```
<BiologicalSample
  analyzeAsMudpit="true"
  database="swissprot" >
  name="Demo_01"
  category="Category 1">
```

The pair of tags <BiologicalSample> to </BiologicalSample> enclose the input files that are to be loaded into this biological sample.

The database is required. It is defined with the <FASTADatabase> tag.

The database="swissprot" term attaches the database to the file being loaded.

```
<QuantitativeModel type="[iTRAQ 4-Plex,  
iTRAQ 8-Plex, TMT 6-Plex]">
```

```
<QuantitativeSample name="name"  
category="category" description="description"  
primary="is reference sample" reporter="114" />
```

```
<PurityCorrection>
```

```
0.0,0.01,0.98,0.01,0.0
```

```
0.0,0.01,0.98,0.01,0.0
```

```
0.0,0.01,0.98,0.01,0.0
```

```
0.0,0.01,0.98,0.01,0.0
```

```
</PurityCorrection>
```

```
</QuantitativeModel>
```

```
<InputFile>Demo_01.dat</InputFile>
```

```
<InputFile>Demo_01.tar.gz</InputFile>
```

```
</BiologicalSample>
```

```
<BiologicalSample analyzeAsMudpit="true"
```

```
database="swissprot" name="Demo_02">
```

```
<InputFile>Demo_02.dat</InputFile>
```

```
<InputFile>Demo_02.mgf.xml</InputFile>
```

```
<InputFile>Demo_02.tar.gz</InputFile>
```

```
</BiologicalSample>
```

```
<BiologicalSample analyzeAsMudpit="true"
```

```
database="swissprot"
```

This command is specific to Scaffold Q+. These commands specify the quantitative model type, name, and purity correction for the labels.

Use one for each reporter ion, e.g., 114, 115, etc.

Options available during Scaffold GUI load data process. Options are also available under Quant in Scaffold's taskbar.

One for each reporter ion, blocks of 5.

Multiple input files can be loaded.

ScaffoldBatch 3 recognizes files from

Sequest (*.dta and *.out),

Sequest batch files (*.ms2 and *.sqt),

zipped folders of Sequest files (either *.zip or *.tgz),

Mascot files (*.dat) and

X! Tandem files (*.xml).

These tags have to be enclosed within a pair of tags <BiologicalSample> to </BiologicalSample>

Additional Biological Sample

Additional Biological Sample

```

name="Demo_03">
<InputFile>Demo_03.dat</InputFile>
<InputFile>Demo_03.mgf.xml</InputFile>
<InputFile>Demo_03.tar.gz</InputFile>
</BiologicalSample>

```

```

<DisplayThresholds name="Some Thresholds"
id="thresh"
proteinProbability="0.95"
minimumPeptideCount="2"
peptideProbability="0.95"
minimumPeptideLength="0"
useDeltaMassTolerance="true"
deltaMassTolerance="100"
useAMU="false"/>

```

```

<DisplayThresholds
name="Some Thresholds"
id="thresh"
useCharge="true,true,true"
minimumNTT="1"
useCharge="true,true,true"/>

```

Menu Edit / Edit Peptide Thresholds

The terms proteinProbability, minimumPeptideCount and peptideProbability correspond to the filter thresholds at the top of the Samples and Proteins pages. These thresholds can be applied to the exported data. See the <Export> tags.

By itself <DisplayThresholds.../> filters the proteins displayed.

The id term defined here is applied to the "thresholds=" term of the <Export> tag.

If the <DisplayThresholds> term is missing or no parameters are defined, the thresholds default to proteinProbability="0.99", peptideProbability="0.95", and minimumPeptideCount="2".

minimumPeptideLength is a "custom" threshold that doesn't show peptides shorter than "X."

useDeltaMassTolerance: "custom" threshold to enable setting delata mass tolerances at the parent ion level.

deltaMassTolerance is also a "custom" threshold to set the actual mass tolerance used in filtering.

useAMU is a "custom" threshold to specify the units of the deltaMassTolerance: true for AMU or false for PPM.

The pair of tags <DisplayThresholds> to </DisplayThresholds> bracket the optional custom thresholds for specific search engines.

These are equivalent to selecting the "Use Individual Program Thresholds" option on the Configure Peptide Thresholds screen on the menu

<pre><MascotThresholds ionMinusIdentityScore="0" ionScores="20.0, 20.0, 30.0, 40.0"/></pre>	<p>Edit / Edit Peptide Thresholds.</p> <p>The other terms are for custom thresholds that can be set from the Configure Peptide Thresholds panel.</p>
<pre><SequestThresholds xCorrs="1.8, 2.5, 3.0, 3.5" deltaCn="0.1"/></pre> <pre><TandemThresholds logExpectScores="2.0, 2.0, 2.0, 2.0"/></pre>	<p>Note that scores for each charge state are listed separately.</p> <p>Note that scores for each charge state are listed separately.</p>
<pre><ZCoreThresholds zcoreScores="100,100,100,100"/></pre> <pre><PhenyxThresholds peptideProbability="90,90,90,90" zScore="5,5,5,5"/></pre> <pre><OmsaTresholds peptideProbability="95,95,95,95" logExpectScore="2,2,2,2"/></pre> <pre><ProteinLynx peptideProbability="95,95,95,95" score="30,30,30,30"</pre> <pre></DisplayThresholds></pre> <pre><Export type="sf3" thresholds="thresh" saveOnlyIdentifiedSpectra="true"</pre>	<p>Note that scores for each charge state are listed separately.</p> <p>Note that scores for each charge state are listed separately.</p> <p>Note that scores for each charge state are listed separately.</p> <p>Note that scores for each charge state are listed separately.</p> <p>Menu File / Save</p> <p>This saves the data loaded into ScaffoldBatch 3 as a *.sf3 file. This file can be read by Scaffold 3 or by the free Scaffold 3 Viewer.</p> <p>The term thresholds="..." refers back to the threshold ID defined in</p>

```
saveNoSpectra="true"  
discardBelowThresholds="false"  
discardSpectraWithPeptidesCount="2"  
saveFrozen="true"  
path="."/>
```

```
<Export  
  type="protxml"  
  thresholds="thresh"  
  path="Demo_foo.protxml"/>
```

```
<Export  
  type="mzIdentML"  
  threshold="thresh"  
  path="Demo_foo.mzid"/>
```

```
<Export  
  type="peptide-report"  
  thresholds="thresh"  
  path="."/>
```

<DisplayThresholds>.

saveOnlyIdentifiedSpectra="true"
saves only the identified spectra in your Scaffold SF3 file.

saveNoSpectra="true" saves the SF3 file without spectra.

discardBelowThresholds="false" will discard un-needed spectra. Equivalent to "save as condensed."

discardSpectraWithPeptideCount: discards spectra on proteins with more than "X" peptides, to save file space. MCP requires that all one-hit-wonders have spectra available, but other proteins don't need spectra to back up the identification. This is equivalent to setting of "1."

The saveFrozen setting saves the file as frozen, where users can't change thresholds.

This saves the data loaded into ScaffoldBatch 3 as a protXML file. This is a data format for loading into a database.

This saves an mzIdentML file (with extension *.mzid) for use in other software. This format is becoming more widely used in the proteomics community.

Exports a peptide report to Excel. The following are additional reports Scaffold can export:

"spectrum-report" - Spectrum report "new" style

"peptide-report" - Peptide report

"protein-report" - Protein report

"publication-report" - Report of publication information

"isoform-report" - Sample report with isoform information

"spectrum-counting-report" -

```

    <Export
      type="experiment-report"
      thresholds="thresh"
      displayBioSamples="true"
      experimentDisplayType="Number of
Unique Peptides"

      experimentDisplayType="Quantitative
Value"

      experimentDisplayType="Percentage of
Total Spectra"
      path="."/>

```

```

    <Export
      saveFrozen="true"
      uploadToTranche="true"
      trancheTitle="title name"
      trancheFilePassword="password"/>

  </Experiment>
</Scaffold>

```

Spectrum count report

"accession-report" -
Accession number report

"experiment-report" -

- a) "displayBioSamples" with options of "true" and "false" toggles between Bio vs MS/MS view. The default is "false" (MS/MS).
- b) "experimentDisplayType" with options of "Protein Identification Probability", "Percentage of Total Spectra", "Number of Assigned Spectra", "Number of Unique Peptides", "Number of Unique Spectra", "Percent Coverage", "Unweighted Spectrum Count", and "Quantitative Value", which do the same as the "Display Type" GUI-dropdown in the Samples view. The default is "Protein Identification Probability".

Uploads the file to Tranche. GUI options in Publish view.

These tags close down an experiment and the Scaffold 3 file.

ScaffoldBatch 3 protXML

The XML format for protXML was developed by The Institute for Systems Biology for their Trans Proteomics Pipeline. They have provided a very complete set of documentation including XSL specifications for protXML. You can see this documentation at

http://sashimi.sourceforge.net/software_tpp.html

Note that proteins have peptides specified in protXML but not spectra. The peptide probabilities given in protXML is the probability of the best spectrum matching that peptide.

This may be confusing. An example may clarify it. Suppose a spectrum in sample_A has a probability of 80% of matching peptide YNGVFQECCQAEDK. Further suppose that in a second sample, sample_B, a different spectrum has a 95% probability of matching YNGVFQECCQAEDK. In the protXML there will be only one probability for the peptide YNGVFQECCQAEDK and that probability will be 95%.

Other details at the spectrum level such as the experimental measured mass or delta mass or Sequest xcorr are also not available in protXML.

Extensions to protXML

Scaffold 3 has extended protXML by providing the tags <identified_samples> and <identified_sample>. We will only document these extensions here.

The <identified_samples> tag is placed within the <peptide>...</peptide> tags. The <identified_samples> tag is a container that can hold one or more <identified_sample> tags.

Each <identified_sample> tag tells one sample where the peptide was identified. If the sample was identified in multiple samples, there will be multiple <identified_sample> tags. Each tag has three required parameters: sample_name, biological_sample_name and initial_probability.

The sample_name is the name of the MS/MS sample run which identified the peptide.

The biological_sample_name is the name of the biological sample that was analyzed in the MS/MS sample run.

The initial_probability is the peptide identification probability from Scaffold 3. The maximum of the initial_probability scores from a peptide's identified_samples is the value of the initial_probability in the peptide tag.

Example

```
<identified_samples>
  <identified_sample
    sample_name="Mudpit_control_071904_01.concat.dtas.txt (control_01)"
    biological_sample_name="Biosample_A"
    initial_probability="1.0"/>
</identified_samples>
```

MuDPIT samples

When a sample is analyzed as a MuDPIT experiment, that is when analyzeAsMudpit="true" in the <BiologicalSample> tag of the driver xml file, the several MS/MS samples in the biological sample are combined and treated as a single sample. This is usually used for shotgun proteomics experiments where the proteins are digested before any separation is done.

Since several MS/MS samples are combined in a Mudpit, ScaffoldBatch 3 uses not the name of the sample, but the word "Mudpit" followed by the name of the first MS/MS sample. This means that the sample_name in the protxml <identified_sample> tag is not the name enclosed in <InputFile> tags in the xml driver file. However, the biological_sample_name in the protxml <identified_sample> tag is the same as the "name" in the xml driver file <BiologicalSample> tag.

protXML Example

```
<protein_summary summary_xml="C:\Scaffold 3_test \sb_test_1_protxml.xml">
  <protein_summary_header
    reference_database="C:\Scaffold 3_testcontrol_sprot.fasta"
    sample_enzyme="Trypsin"
    source_files="C:\Scaffold 3_test \sb_test_1.sfd"

    source_files_alt="control_01.dat,control_02.dat"
min_peptide_probability="0.95"
    initial_min_peptide_prob="0.95"
    min_peptide_weight="0.5"
    num_predicted_correct_prot="25.5"
    num_input_1_spectra="315"
    num_input_2_spectra="768"
    num_input_3_spectra="335"
    num_input_4_spectra="0"
    total_no_spectrum_ids="1418">
  <program_details analysis="Scaffold 3"
    time="Thu Nov 30 12:59:28 PST 2006" version="HEAD">
</program_details>
</protein_summary_header>
<protein_group group_number="1" probability="1.000">
  <protein
    group_sibling_id="a"
    protein_name="ALBU_BOVIN"
    n_indistinguishable_proteins="1"
    probability="1.000"
    percent_coverage="0.644"
    total_number_peptides="56"
    pct_spectrum_ids="6.4">
  <annotation
    protein_description="Serum albumin precursor (Allergen Bos d
6) "/>
  <peptide
    peptide_sequence="TVMENFVAFVDK"
    charge="2"
    initial_probability="0.950"
    is_nondegenerate_evidence="Y"
```

```
n_enzymatic_termini="2"
n_sibling_peptides="1.900"
is_contributing_evidence="Y"
n_instances="2"
calc_neutral_pep_mass="1398.6533">
  <identified_samples>
    <identified_sample
      sample_name="control_02.dat (control_02)"
initial_probability="0.950"/>
    <identified_sample
      sample_name="control_01.dat (control_01)"
initial_probability="0.950"/>
  </identified_samples>
  <peptide_parent_protein
    protein_name="ALBU_BOVIN"/>
</peptide>
</protein>
</protein_group>
</protein_summary>
```

<< End of document ScaffoldBatch 3 Manual >>

© 2011 Proteome Software, Inc