massXpert
the massist's program

Simulating and analyzing
flying species

massXpert version 2.0.3 **User Manual**

This User Manual is distributed at
http://www.massxpert.org

Filippo Rusconi, Ph.D.

Chargé de recherches au CNRS

Centre national de
la Recherche scientifique

UMR CNRS 5153 - UR INSERM 565 - USM MNHN 0503
Muséum national d'Histoire naturelle
43, rue Cuvier
F-75231 Paris Cedex 05
France

***massXpert* User Manual**

http://www.massxpert.org

Revision History

* **april 2009 the 1st** Although not visible in the documentation, I wanted to publically extend my warm thanks to Lionel Élie Mamane, who helped me along these last months with the Debian packaging of *massXpert*. Note that this work also proved useful for other areas in the project.

* **february 2009 the 20th** Updated the *XpertEdit* chapter to show how to configure the options about the number of decimals to be used for display of numerals in the program;

* **february 2009 the 5th** Updated the *XpertEdit* chapter to show the simplified polymer sequence editing feature whereby editing of the sequence might be performed by clicking on monomer items in the list of all the monomers defined in the polymer chemistry definition. Fixed small bug in the documentation of about multi-region selection behaving as oligomers or residual chains.

* **december 2008 the 11th** Updated the *XpertEdit* chapter to show the feature by which it is now possible to force the calculation engine to take into account the left/right end modification(s) when calculating the masses of a sequence region that does not encompass the left/right end of the polymer sequence. This new feature was essential in trying to perform full simulations of the molecular heterogeneity of the telokin protein (Rusconi

et al. 1997 *Biochemistry*). Added a paragraph about max count of chemical modifications of a given monomer at once in the *XpertDef* chapter and another one in the *XpertEdit* chapter to explain its working;

* **september 2008 the 11th** Finally indexed the whole document. Performed some minor modifications so that the documentation system does not produce HTML files anymore (the HTML production was not really worth it anyways);

* **august 2008 the 5th** Updated the user manual with a bunch of updated screen shots;

* **july 2008 the 30th** Updated the *XpertMiner* chapter to illustrate the m/z–z mass list matching feature;

* **july 2008 the 8th** Updated the *XpertEdit* chapter to illustrate the new multi-cleavage feature;

* **july 2008 the 1st** Updated the *XpertEdit* chapter to illustrate the new multi-region and multi-selection features;

* **may 2008 the 29th** Changed the install instructions for the *Mac OS X* system;

* **may 2008 the 27th** Added a chapter about *XpertMiner* to document the new features in that module. Some fixes here and there.

* **may 2008 the 26th** Modified the documentation to reflect switch to version 3 of the GNU General Public License;

* **april 2008 the 25th** The installation chapter was updated to illustrate the installation of the software in the *Mac OS X* system;

* **april 2008 the 24th** The installation chapter was updated to reflect the changes in the way the package might be installed (the package is now relocatable, provided the user indicates where the directories are located);

* **april 2008 the 2nd** The *XpertDef* chapter was updated to detail the new way of defining fragmentation specifications where the side chain is decomposed in the gas-phase. The section about fragmentations is now much better documented;

* **march 2008 the 30th** The *XpertEdit* chapter was updated to include a description of the new fragmentation/mass searching data in-place filtering. A section is now devoted to data filtering;

* **march 2008 the 25th** The *XpertEdit* chapter was updated to include a description of the new sequence cleavage data in-place filtering;

* **march 2008 the 18th** The *XpertEdit* chapter was updated to include a description of the find sequence motif feature;

* **march 2008 the 13th** The *XpertEdit* chapter was updated to include a new paragraph about monomer cross-linking as this is now implemented in the software;

* **february 2008 the 21th** The *XpertEdit* chapter was updated to include a new figure of the polymer modification procedure and to describe the enhanced modification procedure;

* **february 2008 the 15th** The *XpertEdit* chapter was updated to include a new figure of the monomer modification procedure and to describe the enhanced modification procedure;

* **december 2007 the 17th** The chapter about installation of *massXpert* has been rewritten to reflect changes in the building of *massXpert* and in the installation of *Debian GNU/Linux* and *Fedora core GNU/Linux* packages;

* **december 2007 the 16th** The chapter about polymer chemistry definitions has been refactored to reflect the rewriting of the corresponding code. Added a small section about m/z ratio calculation, that was missing, although the feature was added a long time ago;

* **september 2007 the 22nd** The new multi-charged cleavage and fragmentation oligomers have been documented;

* **august 2007 the 19th** Switched back to version 2 of the GPL in the Appendices chapters, as *massxpert* cannot be licensed otherwise: the Qt libraries are licensed using version 2 of the GPL without the "or any later version, at your option" wording;

* **july 2007 the 28th** Updated the *XpertDef* chapter (modifications) to show the new "targets" feature. Updated the *XpertEdit* chapter to show the new monomer modification dialog.

* **july 2007 the 19th** Added explanation on the arbitrary formula-based polymer sequence ends modification.

* **july 2007 the 13th** Revision of the whole document for a better printed output;

* **july 2007 the 10th** Added a section to the *XpertEdit* chapter about the data mining mass list lab feature added recently. Mentioned the installation of *Debian* packages;

* **june 2007 the 30th** Switched the project and all accompanying documentation and other data files to GPL version 3;

* **mid-june 2007** Start of the writing by taking inspiration of the GNU polyxmass manual.

To MARIA CECILIA,

*To all the admirable people acting in the "*Free Software Movement*"*
for a better and more ethical computing world,

To all involved in the development of the K Desktop Environment (KDE),

To all the readers who helped me with this manual...

Contents

List of Figures

List of Tables

1

Preface

This manual is about the *massXpert* mass spectrometric software suite, a software program that aims at letting users predict/analyze mass spectrometric data on (bio)polymers. As such, this manual is intended for people willing to learn how to install and use this software package.

Mass spectrometry has gained popularity across the past ten years or so. Indeed, developments in polymer mass spectrometry have made this technique appropriate to accurately measure masses of polymers as heavy as many hundreds of kDa, and of any chemical type.

There are a number of utilities—sold by mass spectrometer constructors with their machines, usually as a marketing "plus"—that allow predicting/analyzing mass spectrometric data obtained on polymers. These programs are usually different from a constructor to another. Also, there are as many mass spectrometric data prediction/analysis computer programs as there are different polymer types. You will get a program for oligonucleotides, another one for proteins, maybe there is one program for saccharides, and so on. Thus, the biochemist/massist, for example, who happens to work on different biopolymer types will have to learn to use several different software packages. Also, if the software user does not own a mass spectrometer, chances are he will need to buy all these software packages.

The *massXpert* mass spectrometric software is designed to provide *free* solutions to all these problems by:

* Allowing *ex nihilo* polymer chemistry definitions (in the *XpertDef* module that is part of the *massXpert* program);
* Allowing simple yet powerful mass computations to be made in a mass desktop calculator that is both polymer chemistry definition-aware and fully programmable (that's the *XpertCalc* module also part of the *massXpert* program);

* Allowing highly sophisticated editing of polymer sequences on a polymer chemistry definition-specific basis, along with chemical reaction simulations, finely configured mass spectrometric computations... (all taking place in the *XpertEdit* module that is the main module of the *massXpert* program);
* Allowing customization of the way each monomer will show up graphically during the program operation (in the *XpertEdit* module);
* Allowing polymer sequence editing with immediate visualization of the mass changes elicited by the editing activity (in the *XpertEdit* module);
* Unlimited number of polymer sequences opened at any given time and of any given polymer chemistry definition type (in the *XpertEdit* module).

This manual will progressively introduce all these functionalities in a timely and clear manner.

Project History

This is a brief history of *massXpert*.

* **1998–2000** The name *massXpert* comes from a project I started while I was a post-doctoral fellow at the École Polytechnique (Institut Européen de Chimie et Biologie, Université Bordeaux 1, Pessac, France).

  The *massXpert* program was published in *Bioinformatics* (Rusconi, F. and Belghazi, M. *Desktop prediction/analysis of mass spectrometric data in proteomic projects by using massXpert˙ Bioinformatics*, 2002, 644–655).

  At that time, *MS-Windows* was at the *Windows NT 4.0* version and the next big release was going to be "you'll see what you'll see" : *MS-Windows 2000*.

  When I tried *massXpert* on that new version (one colleague had it with a new machine), I discovered that my software would not run normally (the editor was broken). The Microsoft technical staff' would advise to "buy a new version of the compiler environment and rebuild". This was a no-go: I did not want to continue paying for using something I had produced.

* **2001–2006**

  During fall 1999, I decided that I would stop using Microsoft products for my development. At the beginning of 2000 I started as a CNRS research staff in a new laboratory and decided to start fresh: I switched to GNU/Linux (I never looked back). After some months of learning, I felt mature to start a new development project that would eventually become an official GNU package: GNU polyxmass.

  The GNU polyxmass software, much more powerful than what the initial *massXpert* software used to be, was published in *BMC Bioinformatics* in 2006 (Rusconi, F., *GNU polyxmass: a software framework for mass spectrometric simulations of linear (bio-)polymeric analytes. BMC Bioinformatics*, 2006,226).

Following that publication I got a lot of feedback (very positive, in a way) along the lines: —*"Hey, your software looks very interesting; only it's a pity we cannot use it because it runs on GNU/Linux, and we only use MS-Windows and MacOSX!"*

* **2007**–

  In december 2006, I decided to make a full rewrite of GNU polyxmass. The software of which you are reading the user manual is the result of that rewrite. I decided to "recycle" the *massXpert* name because this software is written in C++, as was the first *massXpert* software. Also, because the first *MS-Windows*-based *massXpert* project is not developped anymore, taking that name was kind of a "revival" which I enjoyed. However, the toolkit I used this time is not the Microsoft Foundation Classes (first *massXpert* version) but the Trolltech Qt framework (see the "About Qt" help menu in *massXpert*).

  Coding with Qt libraries has one big advantage: it allows the developer to code once and to compile on the three main platforms available today: *GNU/Linux*, *MacOSX*, *MS-Windows*. Another advantage is that Qt libraries are wonderful software, technically and philosophically (Free Software).

Typographical Conventions

Throughout the book the following typographical conventions are used:

* *emphasized text* is used each time a new term or concept is introduced
* `shell-prompt $` shows the prompt at which a command should be entered as non-root
* `shell-prompt #` shows the prompt at which a command should be entered as root
* `this typography` applies to commands that the user enters at the shell prompt along with eventual options
* ↩ symbolizes pressing the Enter key
* `this typography` applies to an output resulting from entering a command at the shell prompt
* `emacs` or `libQtCore` names of a program or of a library
* `KDE`, `The Gimp` is the name of a generic software (not a specific executable file)
* `/usr/local/share/massxpert`, `/usr/bin/massxpert` are names of a directory or of a file
* `http://www.gnu.org` is an URL (Uniform Resource Locator)

Program Availability, Technicalities

The ancestor of *massXpert*, GNU polyxmass, was initially developed on a *GNU/Linux* system (RedHat distribution versions successively 6.0, 7.0, 7.2, 7.3, 8.0, 9.0) using software from the Free Software Foundation (FSF[1]). The main libraries used were `libglib`, `libgobject`, `libxml2` and `libgtk+`. Since mid-2002, the development was performed on a *Debian GNU/Linux* system (`http://www.debian.org`), which I find to be the ultimate highly-configurable easy-to-use distribution on earth. *massXpert* is still developed using the *Debian GNU/Linux* system, using Free Software libraries that allow cross-platform computer program development with unprecedented ease (Qt libraries from the Trolltech company; `http://www.trolltech.com`). Developing for *GNU/Linux* has been utterly exciting and extremely efficient.

Organization Of This Manual

After having quickly described the installation of *massXpert*, this manual aims at providing the required conceptual toolset for understanding what to expect from a computer program like *massXpert* and how to use it. Thus, the general organization of this book is:

* Installation of the *massXpert* software program;
* The basics of polymer chemistry;
* The basics of mass spectrometry;
* Generalities about *massXpert*;
* The *XpertDef* module (definition of atoms and of new polymer chemistries);
* The *XpertCalc* module (polymer chemistry-aware programmable calculator);
* The *XpertEdit* module (sequence editor, biochemical/mass spectrometric simulations);
* The *XpertMiner* module (data mining calculations);
* The data customization that *massxpert* is designed to not only make possible but also to foster;
* Appendices.

[1] For an in-depth coverage of the philosophy behind the FSF, specifically creating a *free operating system*, you might desire to visit `http://www.gnu.org`.

massXpert's Licensing

The front matter of this manual contains a Copyright statement. I retain the copyright to *massXpert* and all related writings (source and configuration files, programmer's documentation, user manual...) I encourage others to make copies of the work, to distribute it freely, to modify the work and redistribute that derivative work according to the GNU General Public License version 3. The aim of this licensing is to favor spread of knowledge to the widest public possible. Also, it encourages interested hackers[2] to change the code, to improve it and to send patches to the author so that their improvements get into the program to the benefit of the widest public possible. For an in-depth study of the FREE SOFTWARE philosphy I kindly urge the reader to visit http://www.gnu.org/philosophy.

[2] *Hacker* is a specialized term to design the programmer who codes programs; this term should *not* be mistaken with *cracker* who is a person who uses computer science knowledge to break information systems' security barriers.

Contacting The Author

massXpert is the fruit of years of work on my part.[3] While I've put a lot of energy into making this program as stable and reliable a piece of software as possible, *massXpert* comes with no warranty of any kind.

The general policy for directing questions, comments, feature requests, *massXpert* program and/or *massXpert* documentation bug reports should be self-explanatory by looking at the addresses below:

massxpert-maintainer@massxpert.org
massxpert-bugs@massxpert.org
massxpert-webmaster@massxpert.org
massxpert-request@massxpert.org

To direct any comment(s) to the author through snail mail, use the following address:

D^r Filippo Rusconi

Chargé de recherches au CNRS
Centre national de
la Recherche scientifique

UMR CNRS 5153 - UR INSERM 565 - USM MNHN 0503
Muséum national d'Histoire naturelle
43, rue Cuvier
F-75231 Paris Cedex 05
France

[3] As said earlier, *massXpert* is the successor to the GNU polyxmass project of which it inherits all the original features, while still integrating new interesting developments.

2

massXpert Installation

In this chapter, the installation process will be described, for *GNU/Linux* systems, for the *MS-Windows* system and for the *Mac OS X* system.

> Note that the *massXpert* software package is built to be located in certain places on the destination computer's filesystem on the disk. However, beginning with version 1.7.5, the software package is relocatable. Please read section 2, page 11.

The *GNU/Linux* Platform

The installation of *massXpert* can be performed using the source code tarball, the binary tarball or binary distribution-specific packages. At the moment the only distribution-specific packages being prepared are the *Debian GNU/Linux* and *Fedora core* packages. The naming of the packages are according to the following schema:

```
massxpert-2.0.3-bin.tar.gz
```

The "-bin" suffix indicates that the package is a binary package. Source packages would use the "-src" suffix (or none at all), exactly the same way:

```
massxpert-2.0.3-src.tar.gz
```

Install From A Binary Tarball

To install a binary tarball, simply issue the following command as root:

```
shell-prompt # tar xvzf massxpert-version-bin.tar.gz -C /↩
```

This command installs the package to directory `/usr/local`, which means that the program is now available for all to use. Version 2.0.3 of the *massXpert* software installs the following files and directories:

- `/usr/local/bin/massxpert` this is the binary (executable) program itself;
- `/usr/local/share/massxpert/pol-chem-defs` polymer chemistry definition files;
- `/usr/local/share/massxpert/pol-seqs` polymer sequence files;
- `/usr/local/share/massxpert/locales` Qt linguist-based translation files;
- `/usr/local/share/doc/massxpert/COPYING` license file of the *massXpert* software;
- `/usr/local/share/doc/massxpert/usermanual` user manual (HTML- and PDF-formatted files);

Upon installation, if all the dependencies are already installed on the system, the user might start the `massxpert` program right away by executing the `/usr/local/bin/massxpert` file.

The following are the dependencies for *massXpert*, as obtained using the `ldd` command:

```
linux-gate.so.1 =>  (0xffffe000)
libQtSvg.so.4 => /usr/lib/libQtSvg.so.4 (0xb7ebf000)
libQtGui.so.4 => /usr/lib/libQtGui.so.4 (0xb77a9000)
libpng12.so.0 => /usr/lib/libpng12.so.0 (0xb7786000)
libSM.so.6 => /usr/lib/libSM.so.6 (0xb777d000)
libICE.so.6 => /usr/lib/libICE.so.6 (0xb7766000)
libXi.so.6 => /usr/lib/libXi.so.6 (0xb775e000)
libXrender.so.1 => /usr/lib/libXrender.so.1 (0xb7756000)
libXrandr.so.2 => /usr/lib/libXrandr.so.2 (0xb7750000)
libXcursor.so.1 => /usr/lib/libXcursor.so.1 (0xb7747000)
libXinerama.so.1 => /usr/lib/libXinerama.so.1 (0xb7743000)
libfreetype.so.6 => /usr/lib/libfreetype.so.6 (0xb76d4000)
libfontconfig.so.1 => /usr/lib/libfontconfig.so.1 (0xb76a9000)
libXext.so.6 => /usr/lib/libXext.so.6 (0xb769b000)
libX11.so.6 => /usr/lib/libX11.so.6 (0xb75af000)
libQtXml.so.4 => /usr/lib/libQtXml.so.4 (0xb7556000)
libQtNetwork.so.4 => /usr/lib/libQtNetwork.so.4 (0xb74c4000)
libQtCore.so.4 => /usr/lib/libQtCore.so.4 (0xb734e000)
libz.so.1 => /usr/lib/libz.so.1 (0xb7339000)
libpthread.so.0 => /lib/i686/cmov/libpthread.so.0 (0xb7322000)
libdl.so.2 => /lib/i686/cmov/libdl.so.2 (0xb731e000)
libm.so.6 => /lib/i686/cmov/libm.so.6 (0xb72f9000)
libstdc++.so.6 => /usr/lib/libstdc++.so.6 (0xb720d000)
libgcc_s.so.1 => /lib/libgcc_s.so.1 (0xb7202000)
```

```
libc.so.6 => /lib/i686/cmov/libc.so.6 (0xb70ba000)
libaudio.so.2 => /usr/lib/libaudio.so.2 (0xb70a4000)
libXt.so.6 => /usr/lib/libXt.so.6 (0xb7054000)
libXfixes.so.3 => /usr/lib/libXfixes.so.3 (0xb704f000)
libgthread-2.0.so.0 => /usr/lib/libgthread-2.0.so.0 (0xb704a000)
librt.so.1 => /lib/i686/cmov/librt.so.1 (0xb7041000)
libglib-2.0.so.0 => /usr/lib/libglib-2.0.so.0 (0xb6fa1000)
libexpat.so.1 => /usr/lib/libexpat.so.1 (0xb6f81000)
libXau.so.6 => /usr/lib/libXau.so.6 (0xb6f7d000)
libXdmcp.so.6 => /usr/lib/libXdmcp.so.6 (0xb6f78000)
/lib/ld-linux.so.2 (0xb7f2b000)
libpcre.so.3 => /usr/lib/libpcre.so.3 (0xb6f58000)
```

As visible on the first lines of the output above, the main dependency that might not be available on your system, especially if not running the KDE environment, are the libQt* libraries. These should be very easily installable, as they constitute the very core of a highly popular desktop environment used on *GNU/Linux* computers called KDE (*"Kommon Desktop Environment"*).

Installing From A Source Tarball

The source is built using the CMake program (http://www.cmake.org). The build of the software takes place in another directory than the source directory. The steps are easy:

* Unpack the source tarball with the following command as normal user:

  ```
  shell-prompt $ tar xvzf massxpert-2.0.3-src.tar.gz ↩
  ```

  This command unpacks the tarball to the current directory in a subdirectory named massxpert-2.0.3;

* Now create a directory called massxpert-build with the following command:

  ```
  shell-prompt $ mkdir massxpert-build ↩
  ```

* Change to that directory:

  ```
  shell-prompt $ cd massxpert-build ↩
  ```

  and run the cmake configuration command:

  ```
  shell-prompt $cmake ../massxpert-2.0.3 ↩
  ```

* Build the software:

  ```
  shell-prompt $make ↩
  ```

* If the build runs fine, then simply become root and issue the following command:

  ```
  shell-prompt $ make install ↩
  ```

At this point the software should have installed in the destination tree (`/usr/local` prefix). The program should be callable immediately if the destination directory is in the path, otherwise it will first be required to use the full pathname to call it, like for example:

```
shell-prompt $ /home/rusconi/myprogs/bin/massxpert ↩
```

Installing A Binary Package

Debian GNU/Linux Package

To install a *Debian GNU/Linux* package just issue the following command:

```
shell-prompt # dpkg -i massxpert_2.0.3-1_i386.deb ↩
```

If the documentation is needed, the `massxpert-doc_2.0.3-1_i386.deb` might be installed also using the same command line.

Fedora core GNU/Linux Package

To install a *Fedora core GNU/Linux* package just issue the following command:

```
shell-prompt # rpm -ivh massxpert_2.0.3-1_i386.deb ↩
```

Note that the *Fedora core GNU/Linux* package *does* install the documentation along with the binary, so there is no need to as for the installation of a doc package.

The *Mac OS X* Platform

The *Mac OS X* package that is provided does not require the installation of libraries, as these frameworks are included in the application bundle.

The user gets a disk image file (format *dmg*) which he double-clicks in the Finder. This will open up the contents of the image file. After reading the `COPYING` license file, the user drops the `massXpert.app` bundle anywhere on the disk. Double-clicking that `massXpert.app` bundle will launch the program.

Note that by using the "Show Package Contents" Finder menu on that `massXpert.app` bundle, the user may browse the bundle's contents and peruse the user manual that is located in the `Contents>Doc>UserManual` directory.

The *MS-Windows* Platform

In this system also, it is possible to install software in two manners: by installing a binary package or by building-installing the software from source.

Installing A Binary Package

To install the binary package (a file typically in the autoinstaller *exe* format named something like `massxpert-2.0.3-setup.exe`) just double-click onto the

file icon in the file manager. The program is automatically made available in the system menu.

Installing From A Source Tarball

The building of the *massXpert* software package is in two steps: first make sure that the system has the Qt libraries installed along with MinGW32. The packages can be installed by browsing the following link:

http://trolltech.com/developer/downloads/qt/windows

Be sure to select from one of the repositories a package that also contains the MinGW package:

`qt-win-opensource-4.3.0-mingw.exe`

When the installer asks if MinGW should be installed, say **Yes**. Once the installation of the Qt libraries has been performed, the system menu will have a menu *Qt by Trolltech⟶Qt 4.2.3 (Build Debug Libraries)*. Select that menu and answer yes to the question that is asked in a console window. This will build the libraries in the Debug mode, so that it will be possible to compile *massXpert* later. Once that compilation is finished, continue with the build steps for *massXpert*.

To unpack the source tarball, use the 7zip package (http://www.7-zip.org/download.html; Free Software, GNU LGPL license) to extract the source to any location of your choice.

At this point, the steps to make the software are similar to what described above using the CMake program for *MS-Windows*, which is, by the way graphical and not command line. To install the software, you should probably become administrator and issue the following command:

```
make install ↩
```

At this point the software should be installed. Note that in this case no shortcut to the program is installed, the user might want to do that manually.

Software Package Relocation

As mentioned earlier, the *massXpert* software package might be relocated by copying its system directories in other places than the ones it was built for.

When the *massXpert* software program is run, the first thing it does is check whether it can find all its configuration data in the system directories where they belong (configuration data, chemistry definition data, plugins, localization files (translation of the software). If any of its attempts fails, the user is provided with the dialog window shown in Figure 2.1, where he is invited to locate all the system directories that are part of the *massXpert* software distribution. Once the settings are saved, the program can continue its execution successfully. The directories that are checked upon startup of the program are:

* The data directory (where the polymer chemistry definition files—in `pol-chem-defs`—and the polymer sequence files—in `pol-seqs`—are located);

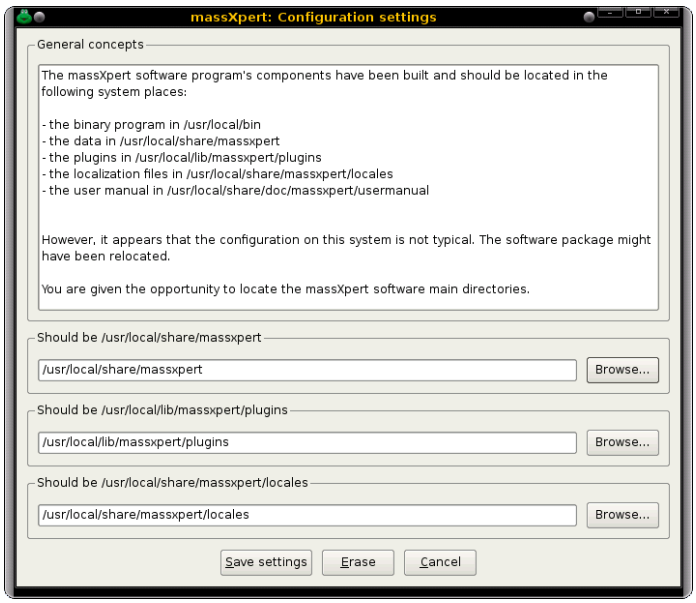


Figure 2.1: ***massXpert* configuration settings.** Upon running the *massXpert* software program, it might detect that its system directories are not located where they were planned to be (the package directories might have been moved, for example). In this case, the user is provided with the dialog window shown here, where he is invited to locate all the system directories that are part of the *massXpert* software package. Once the settings are saved, the program can continue its execution successfully.

* The `plugins` directory (where the plugins—which are dynamically linked libraries—are located);
* The `locales` directory (where the translation files are located).

The directories listed above might be moved on the filesystem, however *their contents might not be changed*. If there is a need to customize the data, that should be done carefully, by changing the contents of the files, *but not the structure of the directories*. For example, it is perfectly safe to add new polymer chemistry definitions or to change files belonging to any polymer chemistry definition, but it is not safe to move files around without knowing what you do.

The User Manual

Whatever the package you used to install *massXpert* binary or source, the documentation got installed on your system (look for a `usermanual` directory somewhere in the installed material). The documentation that was installed is nothing but the document you are reading now. It is available both in the form of HTML files and of one PDF file.

3

Basics in Polymer Chemistry

This chapter will introduce the basics of polymer chemistry. The way this topic is going to be covered is admittedly biased towards mass spectrometry and biological polymers. Moreover, the aim of this chapter is to provide the reader with the specialized words that will later be used to describe and explain the (inner) workings of the *massXpert* program. This manual is not a "crash course" in biochemistry.

Polymers? Where? Everywhere!

Indeed, polymers are everywhere. If you ask somebody to show you something polymeric, he/she will point you at the first plastic object in the vicinity. Right, plastic materials are made of hydrocarbon polymers. We also have many different polymers in our body. Proteins are polymers, complex sugars are polymers, DNA (the so-called "molecule of heredity" is a *huge* polymer. There are polymers in wine, in wood... Where? Everywhere!

The *Oxford Advanced Learner's Dictionary of Current English* gives for *polymer* the following definition: *natural or artificial compound made up of large molecules which are themselves made from combinations of small simple molecules.*

A polymer is indeed made by covalently linking small simple molecules together. These small simple molecules are called *monomer*s, and it is immediate that a *polymer* is made of a number of monomers. A general term to describe the process that leads to the formation of a polymer is *polymerization*. It should be noted that there are many ways to polymerize monomers together. For example, a polymer might be either linear or branched. A polymer is linear if the monomers that are polymerized can be joined at most two times. The first junction links the monomer to an elongating polymer (thus making it the new end of the elongating polymer which, by the way, is longer than before by one unit) and the second junction links the new elongating polymer's end to another monomer. This process goes on until the reaction is stopped, the point at which the polymer reaches its *finished state*. A branched polymer is a polymer in which at least one monomer is able to contract more than two bonds. It is thus clear that a single monomer linked three times to other monomers will yield a "T-structure", which is nothing but a branched structure.

In the following sections we'll describe a number of different kinds of polymers. Each time, they will be described by initially detailing the structure of their constitutive monomers; next the formation of the polymer is described. At each step we shall try to set forth each polymer characteristics in such a manner as to introduce the way *massXpert* "thinks polymers" and to introduce specialized terminologies. Once the basic chemistries (of the different polymers) have all been described, we will enter a more complex subject that is of enormous importance to the mass spectrometry specialist: polymer chain disrupting chemistry. We shall see that this terminology actually involves two kinds of chemistries: cleavage, on the one hand, and fragmentation, on the other hand.

While *massXpert* is basically oriented to linear single-stranded polymer chemistries, it can also be used to simulate highly complex polymer chemistries. Biological polymers are the main focus of this manual, however all the concepts described here may be applied with no modification to synthetic polymer chemistries.

Various Biopolymer Structures

Biopolymers are amongst the most sophisticated and complex polymers on earth and it certainly is not a mistake to take them as examples of how monomers (be these complex or not) can assemble covalently into life-enabling polymers. In this section we will visit three different polymers encountered in the living world: proteins, nucleic acids and polysaccharides. We shall be concerned with 1) the monomers' structure, 2) the polymerization reaction and 3) the final end-capping reaction responsible for putting the polymer in its finished state.

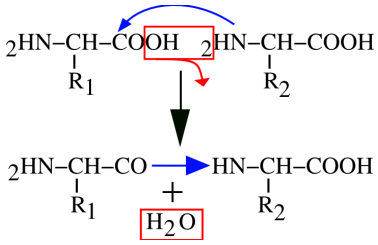


Figure 3.1: **Peptidic bond formation by condensation.** The left end monomer R_1 is condensed to the right end monomer R_2 to yield a peptidic bond. A water molecule is lost during the process.

Proteins

These biopolymers are made of amino acids. There are twenty major amino acids in nature, and each protein is made of a number of these amino acids. The combinations are infinite, providing enormous diversity of proteins to the living world.

A protein is a polar polymer: it has a left end and a right end, and polymerization actually occurs from left to right (from N-terminus to C-terminus, see below). Figure 3.1 shows that the chemical reaction at the basis of protein synthesis is a *condensation*. A protein is the result of the condensation of amino acids with each other in an orderly polar fashion. A protein has a left end, called *N-terminus; amino terminal end* and a right end, called *C-terminus; carboxyl terminal end*. The left end is an amino group ($_2$HN $-$ $-$) corresponding to the non-reacted amino group of the amino acid. Upon condensation of a new amino acid onto the first one, the carboxyl group of the first amino acid reacts with the amino group of the second amino acid. A water molecule is released, and the formation of an amide bond between the two amino acids yields a dipeptide. The right end of the dipeptide is a carboxyl group (–COOH) corresponding to the un-reacted carboxyl group of the last amino acid to have "polymerized in".

The bond formed by condensation of two amino acids is an amide bond, also called—in protein chemistry—a *peptidic bond*. The elongation of the protein is a simple repetition of the condensation reaction shown in Figure 3.1, granted that the elongation *always* proceeds in the described direction (a new monomer arrives to the right end of the elongating polymer, and elongation is done from left to right).

Now we should point at a protein chemistry-specific terminology issue: we have seen that a protein is a polymer made of a number of monomers, called amino acids. In protein chemistry, there is a subtlety: once a monomer is polymerized into a protein it is no more called a monomer, it is called a *residue*. We may say that a residue is an amino acid less a water molecule.

From what we have seen until now, we may define a protein this way: —"*A protein is a chain of residues linked together in an orderly polar fashion, with the residues being numbered starting from 1 and ending at n, from the first residue on the left end to the last one on the right end*". This definition is still partly inexact, however. Indeed, from what is shown in Figure 3.2, there is still a problem with the extremities of the residual chain: what about the amino

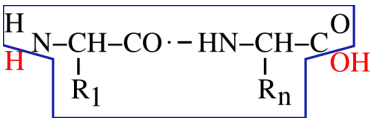


Figure 3.2: **End capping chemistry of the protein polymer.** A protein is made of a chain of residues and of two caps. The left cap is the N-terminal proton and the right cap is the C-terminal hydroxyl. Altogether, the residual chain (enclosed here in the blue polygon) and both the H and OH red-colored caps do form a complete protein polymer in its finished state.

group on the left end of a protein (the amino group sits right onto the first amino acid of the protein), and what about the carboxyl group of the right end of a protein (the carboxyl group sits right onto the last amino acid of the protein)? Because these groups lie at the extremities of the residual chain, they remained unreacted during the polymerization process. But because we are simulating a residual chain using residues and not amino-acids, we still need to put the residual chain in its finished state: by *capping* the left end with a proton *cap* (so as to complete the amino group) and the right end with a hydroxyl cap (so as to complete the carboxyl group). The capping of the residual chain extremities ensures that the polymer is in its finished state, and that it cannot be elongated anymore. The proton is the *left cap* of the protein polymer and the hydroxyl is the *right cap* of the protein polymer.

Now comes the question of unambiguously defining the structure of a protein. It is commonly accepted that the simple ordered sequence of each residue code in the protein, from left to right, constitutes an unambiguous description of the protein's primary structure (that is its sequence). Of course, proteins have three-dimensional structures, but this is of no interest to a program like *massXpert*, which is aimed at calculating masses of polymers. To enunciate unambiguously the sequence of a protein, one would use a symbology like this:

using the 3-letter code of the amino acids:
Ala Gly Trp Tyr Glu Gly Lys
or, using the 1-letter code of the amino acids:
A G W Y E G K
Alanine is thus the residue 1 and Lysine is the last residue (n = 7).

Nucleic Acids

These biopolymers are more complex than proteins, mainly because they are composed of monomers *(nucleotides)* that have three different chemical parts, and because those parts differ in DNA and RNA. A nucleotide is the nucleic acid's brick: *a nucleotide consists of a nitrogenous base combined with a ribose/deoxyribose sugar and with a phosphate group.* There are two different kinds of nucleic acids: deoxyribonucleic acid (DNA, the sugar is a deoxyribose) and ribonucleic acid (RNA, the sugar is a ribose). DNA is most often found in its double stranded form, while RNA is most often found in single strand form. There are four nitrogenous bases for each: Adenine, Thymine, Guanine, Cytosine for DNA; in RNA only one of these bases changes: Thymine is replaced by

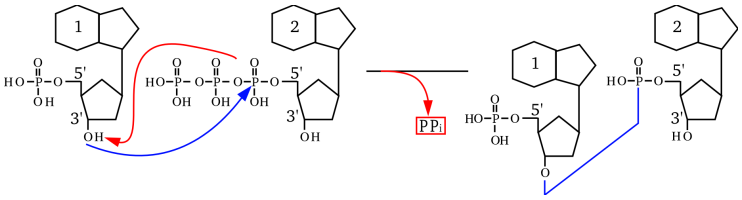


Figure 3.3: **Phosphodiester bond formation by esterification.** The arriving monomer (on the right) has its triphosphate on the 5' carbon of the sugar esterified by nucleophilic attack of the first phosphorus by the alcohol function beared by the 3' carbon of the (deoxy)ribose sugar ring of the left monomer. The bond that is formed is a phosphodiester bond, with release of a pyrophosphate group (PP_i). Note that the sugar and nitrogenous bases are schematically represented in this figure.

Uracile. As for proteins, nucleic acids are polar polymers: the polymerization process is polar, from left to right (sometimes left is up and right is down in certain vertical representations found mainly in textbooks).

This manual is not to teach biochemistry, which is why the structure of the monomers is not described in atomic detail. However, since it is important to understand how the polymerization occurs, Figure 3.3 represents the polymerization reaction mechanism between a nucleotide and another one, to yield a dinucleotide. That reaction is a *trans-esterification*. A nucleic acid has a left end—*5' end; often this end is phosphorylated*—and a right end—*3' end; hydroxyl end*. The trans-esterification reaction is the attack of the phosphorus of the new (deoxy)nucleotide triphosphate by the 3'OH of the right end of the elongating nucleotidic chain. Upon trans-esterification, an *inorganic pyrophosphate* (PP_i) is released, and the formation of a phosphodiester bond between the two nucleotides yields a dinucleotide. The elongation of the nucleic acid polymer is a simple repetition of this esterification reaction so that the chain growth is always in the 5'$\Longrightarrow$3' direction. This is achieved in the living cells by what is called the *5'$\Longrightarrow$3' polymerase enzymatic activity*.

The conventional representation of a nucleic acid involves showing the 5' end on the left, and the 3' end on the right, horizontally. Sometimes, to clearly indicate that the left end is phosphorylated, while the right end is not, the ends are indicated as "5'P" and "3'OH". Figure 3.4 shows a simple way to formalize what a nucleic acid polymer is. The molecule represented on the left is the "monomer" in the sense that the polymer is made of n monomers. On the right side of that figure, the polymer made of n monomers is shown as a residual chain (inside the blue polygon box) that got capped with OH on its left end and H on its right end (red-colored atoms). Thus, in the case of the nucleic acid polymers, the left cap is a hydroxyl and the right cap is a proton. This anecdotically happens to be the exact converse of what was described earlier for proteins.

Now comes the question of unambiguously defining the structure of a nucleic acid. It is commonly accepted that the listing of the named nitrogenous bases in the nucleic acid—from left (5' end) to right (3' end)—constitutes an unambiguous description of the nucleic acid sequence. To enunciate the sequence of

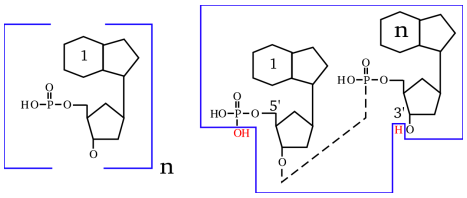


Figure 3.4: **End capping chemistry of the nucleic acid polymer.** A nucleic acid is made of a chain of nucleotides (left formula) and of two caps. The left cap is the hydroxyl group that belongs to the terminal phosphate of the 5' carbon of the sugar. The right cap is the proton that belongs to the hydroxyl group of the 3' carbon of the sugar ring (right formula). Altogether, a finished nucleic acid polymer is made of the nucleotidic chain (enclosed here in the blue polygon), made of the repetitive elements (one of which is shown on the left), and of the two caps (red-colored OH and H, out of the box on the right).

a gene, one would use a symbology like this:

for a DNA, using the 1-letter code of the nitrogenous bases: A T G C A G T C
for an RNA, using the 1-letter code of the nitrogenous bases: A U G C A G U C
Adenine is thus the base 1 and Cytosine is the last base (n = 8).

Saccharides

These biopolymers are certainly amongst the most complex ones in the living world. This is mainly due to the fact that saccharides are usually heavily modified in living cells with a huge variety of chemical modifications. Furthermore, the ramifications in the polymer structure are more often the normal situation than not. Interestingly, these molecules are first thought of as the "fuel" for the cell, which is certainly far from being total nonsense, but it is also undoubtful that their structural role is extremely important (often in combination with proteinaceous material). Another interesting aspect of their ability to form complex structures is their use as "key" systems for identification processes: a number of complex sugars are located on the cell walls and provide "recognition patterns" for the other cells to deal with...

Nonetheless, the general picture is not that complex, if the way monomers are polymerized together is the only concern (which is the case in this manual). As far as we are concerned, in fact, the polymerization mechanism is a simple condensation (much like what has been described for proteins), yielding a sugar bond. Indeed, some people use the same terminology: a monomeric sugar becomes a residue once polymerized in the saccharidic chain. There are two main different kinds of sugars: *pentoses* (in C_5) and *hexoses* (in C_6); it should be noted, however, that there is a variety of other common molecules, like *sialic acids*, *heptoses*...

Like already seen for proteins and nucleic acids, a saccharidic polymer is polar: it has a left end and a right end. The terminology regarding the ends of a saccharidic polymer is rather unexpected at first sight: the left end is said to be the *non-reducing end* while the right end is said to be the *reducing*

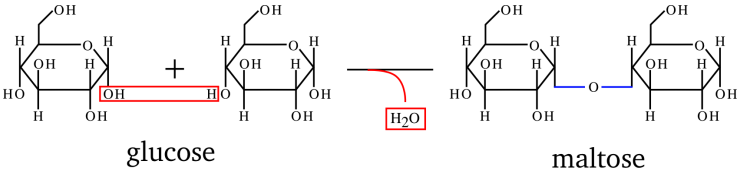


Figure 3.5: **Osidic bond formation by condensation.** The two monomers are subject to condensation with loss of one molecule of water.

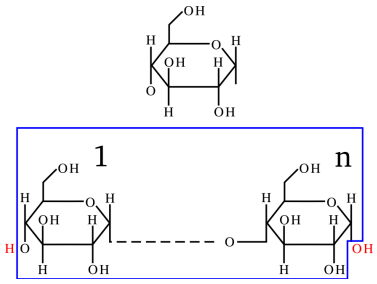


Figure 3.6: **End capping chemistry of the polysaccharidic polymer.** A polysaccharide is made of a chain of osidic residues (blue-boxed formula) and of two caps (red-colored atoms). The left cap is the proton group that belongs to the non-reducing end of the polymer. The right cap is the hydroxyl group that belongs to the reducing end of the polymer.

end. Historically this was observed with monosaccharides (also called *monoses*), which reduced cupric (Cu^{2+}) ions, thus getting oxydized themselves on the carbonyl (when in the open ring aldehydic form).

Figure 3.5 shows the polymerization reaction between a sugar and another one (2 glucose monomers, actually), to yield a maltose disaccharide. The polymerization mechanism is a simple condensation. The elongation of the saccharidic polymer is a simple repetition of this condensation reaction so that the chain growth is always in the same orientation, from the non-reducing end to the reducing end. The conventional representation of a polysaccharide involves showing the non-reducing end on the left, and the reducing end on the right, horizontally. Figure 3.6 shows a simple way to formalize what a saccharidic polymer is. The top formula is the representation of the monomer. The bottom formula represents a polysaccharide, with the repetitive elements boxed (there are n monomers polymerized). The atoms shown in red (outside the boxed repetitive elements) are the saccharidic polymer caps. Thus, we see clearly that in the case of polysaccharides, the left cap is a proton and the right cap is a hydroxyl. This anecdotically happens to be identical to proteins and the exact converse of what we described previously for nucleic acids.

Now comes the question of unambiguously defining the structure of a saccharidic polymer. It is commonly accepted that the simple ordered sequence of the named monoses in the saccharidic polymer, from left (non-reducing end) to right (reducing end), constitutes an unambiguous description of the glycan sequence. To enunciate the sequence of a glycan, one would use a symbology

polymer	name	code	formula	left cap	right cap
protein				H	OH
	Glycine	G	$C_2H_3O_1N_1$		
	Alanine	A	$C_3H_5O_1N_1$		
	Tyrosine	T	$C_9H_9O_2N_1$		
nucleic acid				OH	H
	Adenine	A	$C_{10}H_{12}O_5N_5P_1$		
	Cytosine	C	$C_9H_{12}O_6N_3P_1$		
saccharide				H	OH
	Arabinose	Ara	$C_5H_8O_4$		
	Heptose	Hep	$C_7H_{12}O_8$		
Note: LC=left cap; RC= right cap					

Table 3.1: **Quick comparison of three biopolymers with examples of monomers**

like this:
using a 3-letter code:
Ara Gal Xyl Glc Hep Man Fru
Arabinose is thus the monose 1 and Fructose is the last monose (n = 7).

Incidentally, this is where the ability of *massXpert* to handle monomer codes of non-limited length comes in handy!

To Sum Up

We made a rapid overview of the three major polymers in the living world. A great many other polymers exist around us. Table 3.1 tries to sum up all the informations gathered so far. Note that the formulæ given for the monomers are the "residual" ones. For example, the formula of the glycyl residue corresponds to the formula of the Glycine monomer less one molecule of water. Many synthetic polymers are much simpler than the ones we have rapidly reviewed, and it should be clear that, if *massXpert* can deal with the complex biopolymers described so far, it certainly will be very proficient with less complex synthetic polymers. Describing the formation of polymers is one thing, but we also have to describe how to disrupt polymers. This is what we shall do in the next section.

Polymer Chain Disrupting Chemistry

The "polymer chain disrupting chemistry" was mentioned earlier as a complex subject that was of *enormous* importance to the mass spectrometrist. This is why that subject will be treated in a pretty thorough manner. First of all it should be noted that a chemical modification of a polymer does not necessarily involve the perturbation of the chain structure of the polymer. Here, however, we are concerned specifically with a number of chemical modifications that yield a polymer chain perturbation; *cleavage* and *fragmentation*:

A CLEAVAGE IS A CHEMICAL PROCESS by which a cleaving agent will act directly on the polymer chain making it fall into at least two separated pieces (the *oligomers*). As a result of the cleavage reaction, groups originating in the cleaving molecule remain attached to the polymer at the precise cleavage location;

A FRAGMENTATION IS A CHEMICAL PROCESS by which the polymer structure is disrupted into separated pieces (the *fragments*) mainly because of energy-dependent electron doublet rearrangements leading to bond breakage.

Polymer Cleavage

We said above that, upon cleavage of a polymer, the cleaving molecule reacts with it, and by doing so directly or indirectly "*dissolves*" an inter-monomer bond. A polymer cleavage always occurs in such a way as to generate a set of *true* polymers (smaller in size than the parent polymer, evidently, which is why they are called *oligomers*). Indeed, let us take the example shown in Figure 3.7, where a tripeptide (a very little protein, containing a methionyl residue at position 2) is submitted either to a water-mediated cleavage (hydrolysis, upper panel) or to a cyanogen bromide-mediated cleavage (lower panel). The two cases presented in this figure are similar in some respects and different in others:

* In the first case the molecule that is responsible for the cleavage is water, while in the second case it is cyanogen bromide;
* In both cases the bond that is cleaved is the inter-monomer bond (in protein chemistry this is a peptidic bond);
* In both cases the Oligomer 2 has the same structure;
* The structures of the Oligomer 1 species differ when produced using water or cyanogen bromide as the cleaving molecule.

The difference between hydrolysis and cyanogen bromide cleavage is in the generation of the Oligomer 1 species: the cyanogen bromide cleavage has a side effect of generating a homoserine as the right end monomer of Oligomer 1, while hydrolysis generates a genuine methionine monomer. This is because water reverses in a very symmetrical manner what polymerization did (hydrolysis is the converse of condensation), while cyanogen bromide did some chemical modification onto the generated Oligomer 1 species.

Nonetheless, the reader might have noted that—interestingly—all the four oligomers do effectively have their left cap (a proton) and their right cap (the hydroxyl). This means that in both water- and cyanogen bromide-mediated cleavages, all the generated oligomers are indeed true polymers in the sense that: 1) they are a chain of monomers (modified or not) and 2) they are correctly capped (*i.e.* they are polymers in their finished state). This is important because it is the basis on which we shall make the difference between a cleavage process and a fragmentation process. Thus, the *massXpert* definition of an oligomer might be: *an oligomer is a polymer (of at least one monomer) in its finished state that was generated upon cleavage of a longer polymer.*

When the polymer cleavage reaction precisely reverses the reaction that was performed for the same polymer's synthesis, there is no special difficulty. But

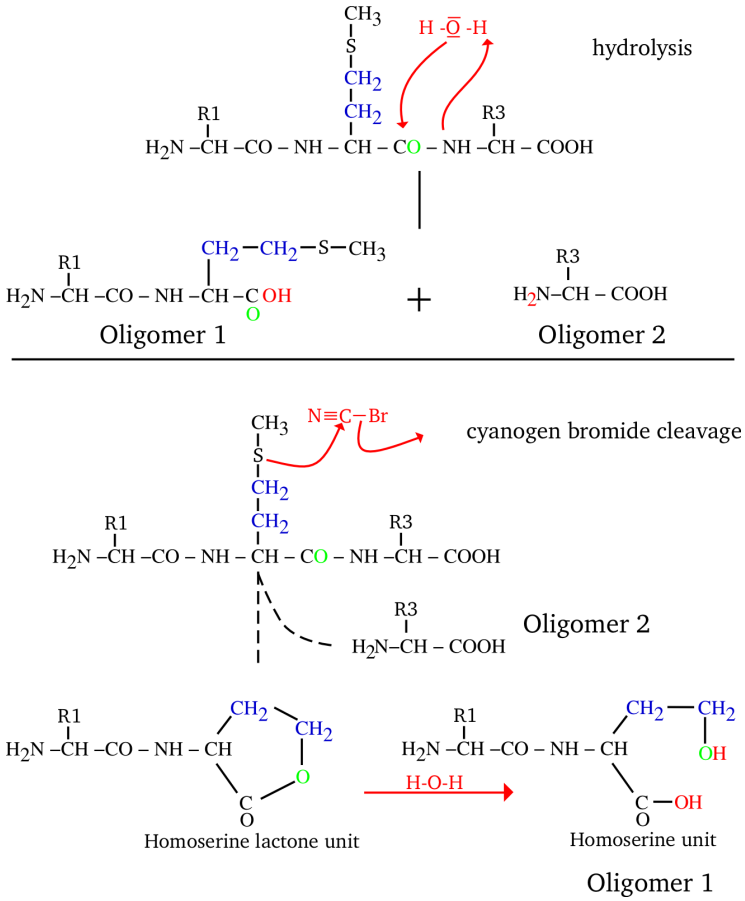


Figure 3.7: **Protein cleavage by water and cyanogen bromide.** A tripeptide is cleaved at position 1 either by hydrolysis (top) or by cyanogen bromide (bottom). Cyanogen bromide cleaves specifically on the right of a methionine monomer. Upon cleavage, the methionyl monomer gets converted into homoserine by the cyanogen bromide reagent.

when the cleavage reaction modifies the substrate, then this should be carefully modelled. How? To answer this question we might start by comparing the two different Oligomer 1 species that were yielded upon the water-mediated and the cyanogen bromide-mediated cleavage reactions: "the hydrolysis-generated Oligomer 1 is equal to the cyanogen bromide-generated Oligomer 1 +S1 +C1 +H2 -O1"; this is a big difference! The observations we did so far might be worded this way: *Whenever a protein undergoes a cyanogen bromide-mediated cleavage, the "-C1H2S1+O1" chemical reaction should be applied to the resulting oligomers if and only if they have a methionine monomer at their right end.* In *massXpert*'s jargon, this logical condition is called a *cleavage rule* (described later; see page 50).

Well, all this sounds reasonable. But what about the "normal" case, when the cleavage is done using water? Nothing special: the mass of the oligomer is calculated by summing the mass of each monomer in the oligomer (since the monomers are not modified, this is easily done) and the masses corresponding to the left and right caps (these are defined in the polymer chemistry definition; in our present case it would be a proton on the left end, and a hydroxyl on the right end). In this way, the oligomer complies with its definition, which states that it is a faithful polymer made of monomers and that it is in its finished state.

Yes, but then how will *massXpert* manage to calculate the mass of the modified oligomer, like our Oligomer 1 in the case of the cyanogen bromide-mediated cleavage? Simple enough: in a first step it does exactly the same way as for the unmodified oligomer. Next, each oligomer is checked for presence or absence of a methionine residue on its right end. If a methionine is found, the mass corresponding to the "-C1H2S1+O1" chemical reaction is applied. And that's it.

In the previous cyanogen bromide example, the logical condition was involving the identity of the oligomers' right end monomer, but other examples can involve not the right end monomer, but the left end monomer, if some chemical modification was to occur to the monomer sitting right of the cleavage location. In this case the user would have to analyse the situation and provide *massXpert* with the proper chemical reaction by stating something analog to: *if and only if they have a Xyz monomer at their left end.* This introduction to polymer cleavage abstraction should be enough to later delve into the cleavage specification definition as *massXpert* conceives it and that is thoroughly detailed at page 50.

Polymer Fragmentation

In a fragmentation process, the bond that is broken is not necessarily the inter-monomer bond. Indeed, fragmentations are oft-times high energy chemical processes that can affect bonds that belong to the monomers' internal structure. This is one of the reasons why fragmentations do differ from cleavages: they are specific of the polymer type in which they occur. Hydrolyzing a protein and an oligosaccharide is just the same process, from a chemical point of view. But fragmenting a protein or an oligosaccharide are truly different processes because the way that the fragmentation happens in the polymer sequence is so much dependent on the nature of each monomer that makes it.

Another peculiarity of the fragmentations, compared with the cleavages that

were described above, is the fact that there is no cleaving molecule starting the process. Instead, a fragmentation process is often initiated by an intra molecular electron doublet rearragement that propagates more or less in the polymer structure to eventually break it. Fragmentations are mainly a gas phase process, not some reaction that happens in solution as a result of putting in contact the polymer and some reagent. It is precisely because no cleaving molecule is involved in the fragmentation process that the fragments are not necessarily capped like a normal polymer should be; and this is another really important difference between cleavage and fragmentation. The following examples should illustrate these concepts: protein and nucleic acid fragmentation.

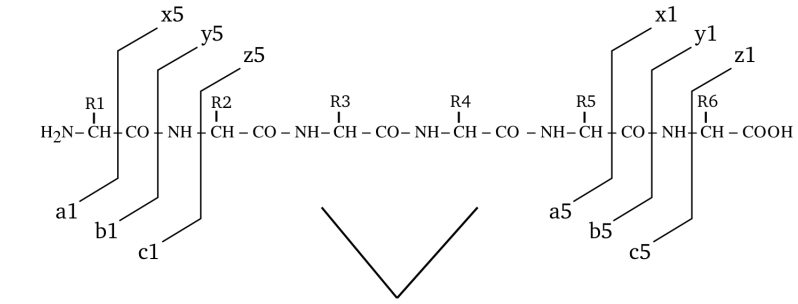
Protein Fragmentation

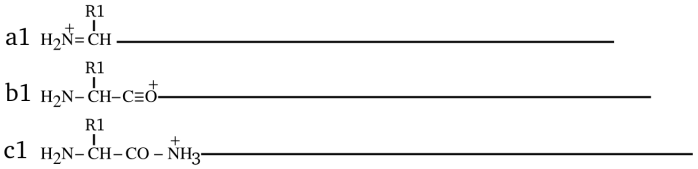
There is a pretty important number of different kinds of fragments that can be generated upon fragmentation of peptides. We are going to detail the most common ones; the user is invited to use the *massXpert*' fragmentation-specification grammar to add less frequent (or newly discovered) fragmentation types.

As can be seen from Figure 3.8, the fragmentations do generate fragments of three categories: the ones that include the left end of the precursor polymer (a, b, c), the ones that include the right end of the precursor polymer (x, y, z), and finally the special case in which the fragment is an *internal fragment*, like the immonium ions. When looking at the fragmentations described in the figure it becomes immediately clear why a fragmentation cannot be mistaken for a cleavage: the ionization of the fragment is not necessarily due to the captation of a proton by the fragment. Furthermore, we can also see that a fragmentation is not a cleavage because the fragment that is generated is *absolutely* not necessarily what we call a polymer, in the sense that the fragment might not be capped the same way as the precursor polymer is (that is, the fragment is not in its finished polymerizaton state).

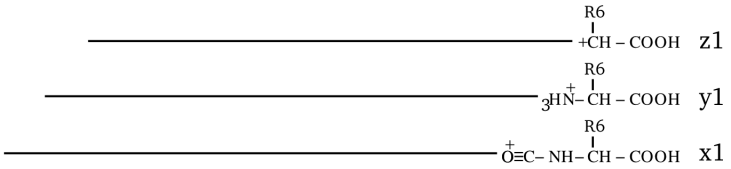
The two observations above should make clear to the reader that calculating masses for fragments is a more difficult process than what was described above for the oligomers. Indeed, while it was simple to calculate the mass of an oligomer (by simply adding the masses of its constitutive monomer units, plus the left and right caps, plus ionization), here there is no chemical formalism generally applicable to all the fragment types. This is why the specification of the fragmentation is left to the user's responsibility.

By looking at Figure 3.8, the reader should have noticed that the fragment naming scheme takes into consideration the fact that the fragment bears the left or the right end of the precursor polymer (or none, also). Indeed, the numbering of fragments holding the left end of the precursor polymer sequence begins at the left end, and for fragments that hold the right end, at the right end. Thus the third fragment of series *a*—*a3*—would involve monomers [1→3]; and the third fragment of series *y*—*y3*—would involve monomers [6→4] (in the figure, these left-to-right and right-to-left directions are symbolized using arrows). Therefore, it should appear to the reader how important—when specifying a fragmentation—it is to clearly indicate from which end of the precursor polymer the fragment is generated (in *massXpert*'s jargon this is "LE" for left end, "RE" for right end and "NE" for no end). *massXpert* knows what action it should take when it encounters one of these three specifications; for example, if a "LE"





m (a1) = m (monomer 1) - C1O1 + left cap (H1)
m (b1) = m (monomer 1) + left cap (H1)
m (c1) = m (monomer 1) + N1H3 + left cap (H1)



✲ m (z1) = m (monomer 6) -N1H1 + right cap (O1H1) (variant: +H1)
m (y1) = m (monomer 6) + H2 + right cap (O1H1)
m (x1) = m (monomer 6) + C1O1 + right cap (O1H1)

✲ Note how a z fragment is identical to a [y -NH3] fragment.
In some cases (high CID energy) the z fragment is often seen as a species of mass z+1

canonical monomer immonium ions

Figure 3.8: **Protein fragmentation patterns most widely encountered.** An hexapeptide is fragmented in the seven most widely encountered manners, such as to generate a, b, c, x, y, z and immonium fragment ions. The figure illustrates the position of the cleavage for each kind of fragment (exemplified using the case of the smallest fragment possible) and the mass calculation method is described for each fragment kind; consider that each fragment bears only *one positive* charge.

specification is found for a given fragmentation specification, *massXpert* adds to the fragment's mass the mass corresponding to the left cap of the precursor polymer.

***a* fragment series** If we take the *a* fragment series, the Figure 3.8 indicates that the fragments include the left end and that their last monomer lacks its carbonyl group (see, on top of Figure 3.8, that the *a1* arrow goes between the CαH and the CO of monomer 1?). So we would say that each fragment of the *a* series should be challenged with the following chemical treatments: 1) addition of the mass corresponding to the left cap (proton), 2) removal of the mass corresponding to the lacking CO group. This way we have the mass of fragment *a1*. If we were interested in the fragment *a4* we would have summed the masses of monomers 1 to 4, added the mass of the left cap, and finally removed the mass of a CO. The mass calculation is thus mathematically expressed

$$a_i = LC + \sum_{1}^{i} M_i - CO$$

***b* fragment series** Similarly, the mass calculation is mathematically expressed

$$b_i = LC + \sum_{1}^{i} M_i$$

***c* fragment series** The mass calculation is mathematically expressed

$$c_i = LC + \sum_{1}^{i} M_i + NH_3$$

***x* fragment series** For this series of fragments we do not add the left cap anymore, but replace it with the right cap, since the fragments hold the right end of the precursor polymer. Note also that the numbering of the monomers using the variable i in the following mathematical expressions goes from right to left (contrary to what happened for the *a, b, c* fragment series. All the fragments that hold the precursor polymer right end are numbered this way, so this applies to fragments *x, y, z*. The mass calculation is mathematically expressed

$$x_i = RC + \sum_{1}^{i} M_i + CO$$

***y* fragment series** The calculation is mathematically expressed

$$y_i = RC + \sum_{1}^{i} M_i + H_2$$

***z* fragment series** In low energy CID, the z fragments are expressed this way:

$$z_i = RC + \sum_{1}^{i} M_i - NH$$

which is equivalent to y-NH_3; in high energy CID an additional proton is often measured:

$$z_i = RC + \sum_{1}^{i} M_i - NH + H$$

***immonium* fragment series** These fragments are internal fragments in the sense that they do not hold neither of the two precursor polymer's ends. *massXpert* understands that the user is speaking of this kind of fragment when the "from which end" piece of data –in the fragmentation specification– states "NE" instead of "LE" or "RE" (see page 52). The mass calculation for these fragments does not take into account the monomers surrounding the one for which the calculation is done. The mass for an immonium ion –at position i in the precursor polymer– will be the mass of the monomer at position i, less the mass of a CO, plus the mass of a proton. The mass calculation for these special internal fragments is expressed

$$imm_i = M_i + H - CO$$

Nucleic Acid Fragmentation

The fragmentations that can be obtained with nucleic acids are numerous and it is more complicated than with proteins to describe them fully. The main reason for this is that there are a big number of fragmentation combinations because of the loss of nitrogenous bases from the skeleton. The mechanisms by which this loss happens are fairly complex, and I am not going to detail any of them. Figure 3.9 on the next page shows the most common fragmentations (without taking into consideration the potential loss of bases). An example of fragment is given for each fragment series (pretty the same way as we did before for proteins). Note that the fragment representations are aimed at helping the reader to figure out what the product ion is, not taking into account where the negative charge lies on the fragment, since this charge can float around at every de-protonatable group. All the fragments shown bear one and one only negative charge.

The reader might have noticed at the bottom of Figure 3.9 on the following page that a provision is made in the case the fragmented molecular species are not 5' end-phosphorylated but 5' end-hydroxylated. Indeed, the canonical monomer is such that, upon polymerization and left capping, the 5' end is phosphorylated. However, oft-times the oligonucleotides are synthesized chemically without the 5' end phosphate group, thus ending in hydroxyl. This special case should be accounted for by applying to all the fragments that bear the left end of the precursor polymer the following chemical reaction: $-HPO_3$. This chemical reaction should be applied *in addition* to the chemical reaction that yields the fragment *per se*.

Exactly as done earlier for the protein fragments, the mathematical expressions used to calculate the mass of different series of nucleic acid fragments are

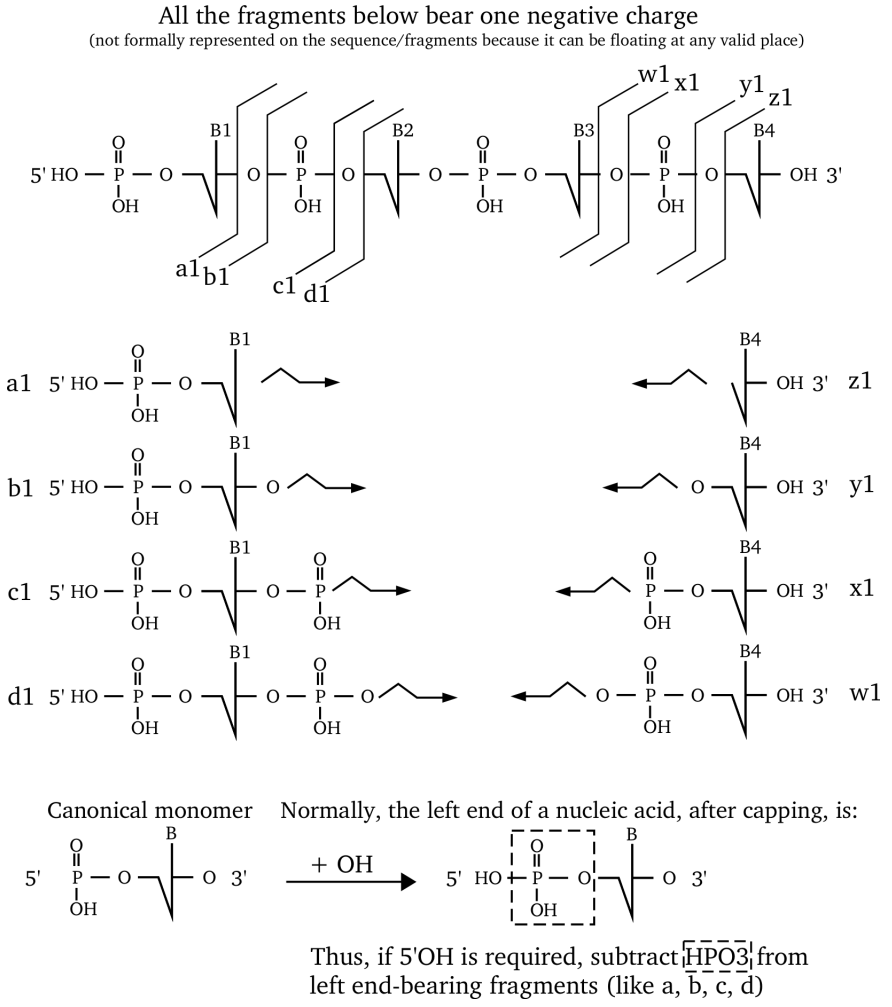


Figure 3.9: **DNA fragmentation patterns most widely encountered.** A short DNA sequence is fragmented in the eight most widely encountered manners, such as to generate a, b, c, d, w, x, y, z fragment ions. The figure illustrates the position of the cleavage for each kind of fragment (exemplified using the case of the smallest fragment possible). and the mass calculation method is described for each fragment kind; considering that each fragment is protonated only once (+1).

provided; in these calculations it is assumed that the left end of the precursor polymer is phosphorylated (5'P) and the reader should bear in mind that this precise phosphate might itself be expelled by the fragmentation. The fragment naming schemed detailed earlier for proteins applies to nucleic acids in the very same manner.

***a* fragment series** These fragments most often appear with base loss.

$$a_i = LC + \sum_{1}^{i} M_i - O$$

***b* fragment series**

$$b_i = LC + \sum_{1}^{i} M_i$$

***c* fragment series**

$$c_i = LC + \sum_{1}^{i} M_i - HPO_2$$

***d* fragment series**

$$d_i = LC + \sum_{1}^{i} M_i - HPO_3$$

***w* fragment series**

$$w_i = RC + \sum_{1}^{i} M_i + O$$

***x* fragment series**

$$x_i = RC + \sum_{1}^{i} M_i$$

***y* fragment series**

$$y_i = RC + \sum_{1}^{i} M_i - HPO_2$$

***z* fragment series**

$$z_i = RC + \sum_{1}^{i} M_i - HPO_3$$

There are also a variety of fragments for which a base is lost.

More Complex Patterns Of Fragmentation

Before finishing with fragmentations, it is necessary to describe a powerful feature of the fragmentation specification grammar available in *massXpert*. This feature was required for the fragmentation of oligosaccharides and also sometimes for proteins. When the fragmentation (the bond breakage reaction itself) occurs at the level of certain monomers, it might be necessary to be able to specify some particular chemistry that would arise on the monomer in question.

We have seen in the cleavage documentation that, upon cleavage of a protein sequence with cyanogen bromide, for example, a particular chemical reaction had to be applied to the oligomers that were generated with a methionine monomer as their right end monomer. Well, in a fragmentation specification it is possible to apply comparable chemical reactions but in a more thorough manner. Indeed, while in the cleavage it was possible to say something like "*apply a given chemical reaction to the oligomer if the right end monomer is Xyz*", in the fragmentation the logical condition can be bound not only to the identity of the currently fragmented monomer, but also (optionally) to the identity of the previous and/or next monomer in the precursor polymer sequence. For example: —"*Apply a given chemical reaction if fragmentation occurs at the level of "Xyz" monomer only if it is preceded by a "Yxz" monomer and followed by a "Zyx" monomer*".

These logical conditions are called *fragmentation rules*. A *fragmentation specification* can hold as many rules as necessary. All of this is described in great detail at page 52.

To Sum Up

To sum up all what we have seen so far with polymer chain disrupting chemistries:

- A polymer sequence gets cleaved into oligomers when a chemical reaction occurs in it at the level of one or more inter-monomer bond(s); monomer-specific chemical reactions can be modelled into the cleavage specification using at most one leftrighrule;
- A polymer sequence gets fragmented into fragments when a bond breakage occurs, without the help of any exterior molecule, at any level of the polymer structure, with no limitation to the inter-monomer bond; monomer-specific chemical reactions can be modelled into the fragmentation specification using any number of fragrules;
- Oligomers are automatically capped—*on both ends*—using the rules described in the precursor polymer's definition;
- Fragments are capped automatically only—*on the end they hold, if any*—using the rules described in the precursor polymer's definition;
- Oligomers are automatically ionized (if required by the user) using the rules described in the precursor polymer's definition;
- Fragments are never ionized automatically; ionization (gain/loss of a charged group) is necessarily integrated in the fragmentation specification.

4

Basics in Mass Spectrometry

Mass spectrometry has become a "buzz word" in the field of structural biology. While it has been used for long to measure the molecular mass of little molecules, its recent developments have brought it to the center of the analytical arsenal in the field of structural biology (also of "general" polymer science). It is now current procedure to use mass spectrometry to measure the mass of polypeptides, oligonucleotides (even complete transfer RNAs!) and saccharides, amongst other complex biomolecules.

A mass spectrometer is usually described by giving to its three main different "regions" a name suggestive of their function:

* the source, where production of ionized analytes takes place;
* the analyzer, where the ions are electrically/magnetically "tortured";
* the detector, where the ions arrive, are detected and counted.

Ion Production: The Source

Mass spectrometry can do nothing if the molecule to analyze (the *analyte*) is not in an electrically charged state. The process of creating an ion from an uncharged analyte is called *ionization*. Most of the times the ionization is favored by adapting the sample's pH to a value lower than the isoelectric pH of the analyte, which will elicit the appearance of charge(s) onto it. In cases where the analyte cannot be charged by simple pH variations (small molecule that does not bear any ionizable chemical group), the ionization step might require—on the massist's part—use of starker ionization techniques, like electronic impact ionization or chemical ionization. In biopolymer mass spectrometry, the pH strategy is usually considered the right way to proceed. The ionization process might involve complex charge transfer mechanisms (not fully understood yet, at least for certain ionization/desorption methods) which tend to ionize the analyte in a way not predictable by looking at the analyte's chemical structure.

Ion production should not be uncoupled from one important feature of mass spectrometry: solvent evaporation–*desolvation*, in case of liquid sample delivery to the mass spectrometer–and sample *desorption*—in case of solid state sample introduction. The general idea is that mass spectrometry works on gas phase ions. This is because it is of crucial importance, for a correct mass measurement to take place, that the analyte be *totally* freed of its chemical immediate environment. That is, it should be "naked" in the gas phase. Equally important is the fact that ions must be capable of travelling long distances without ever encountering any other molecule in their way. This is achieved by pumping very hard in the two regions called "analyzer" and "detector". In this respect, the source is a special region because, depending on the design of the mass spectrometer, it might be partially at the atmospheric pressure during mass spectrometer operation. It is not the aim of this manual to provide insights into mass spectrometer design topics, but the general principle is that mass spectrometry involves working on gas phase ions. This is why a mass spectrometer is usually built on extremely reliable pumping technology aimed at maintaining for long periods of time (with no sudden interruption, otherwise the detector might suffer seriously) a good vacuum in the conduit in which ions must flow during operation.

The Analyzer

Once an ion has been generated in the gas phase, its mass should be measured. This is a complex physical process. Depending on the mass spectrometer design, the mass measurement is based on more or less complex physical events. Magnetic mass spectrometers are usually thought of as pretty complex devices; this is also the case for the Fourier transform ion cyclotronic resonance devices. An analyzer like the *time of flight* analyzer is much simpler. I will refrain from trying to explain the physics of the mass measurement, just limit myself to saying that—at some stage of the mass measurement process—forces are exerted on the ions by electric/magnetic fields (incidentally, this explains why it is so important that an analyte be ionized, otherwise it would not be subject to these fields). The ionized analytes submitted to these forces have their trajectory modified

in such a way that the detector should be able to quantify this modification. Roughly, this is the measurement process.

What Is Really Measured?

Prior to entering into some detail, it seems necessary to make a few definitions[1]:

- unified mass scale (u): IUPAC & IUPAP (1959-1960) agreed upon scale with 1 u equal to 1/12 the mass of the most abundant form of carbon; the dalton is taken as identical to u (but not accepted as standard nomenclature by IUPAC or IUPAP), it is abbreviaed in Da.

- a former unit was "a.m.u." (*i.e.* "atomic mass unit"). It should be considered obsolete, since based on an old 1/16 of ^{16}O standard;

- the mass of a molecule (also "molecular mass") is expressed in daltons. The symbol commonly used is "M" (not "m"), as in "M+H" or "M+Na"... Symbol "m" is already employed for ion mass (as in "m/z");

- the mass-to-charge ratio ("m/z") of an ion is the ion's mass (in daltons) divided by the number (z) of elementary charges. Hence "m/z" is "mass per charge" and units of "m/z" are "daltons per charge";

- nominal mass: the integral sum of the nucleons in an atom (it is also the atomic mass number);

- exact (also known as accurate) mass: the sum of the masses of the protons and neutrons plus the nuclear binding energy;

In the previous sections I used to say that a mass spectrometer's task is to measure masses. Well, this is not 100 % exact. A mass spectrometer actually allows to measure something else: it measures the *m to z ratio* of the analyte, which is denoted m/z. What is this "*m to z ratio*" all about? Well, we said above that a mass spectrometer has to exert forces on the ions in order to determine their m/z. Now, let us say that we have an electric field of constant value, E. We also have two ions of identical masses, one bearing one charge (q) and the other one bearing two charges ($2q$)—positive or negative, no matter in this discussion. These two ions, when put in the same electric field E, will "feel" two different forces exerted on them: F_1 and F_2. It is possible to calculate these forces ($F_1 = qE$ and $F_2 = 2qE$). Evidently, the ion that bears two charges is submitted to a force that is twice as intense as the one exerted on the singly charged ion.

What does this mean? It means simply that the numeric result provided by the mass spectrometer is not going to be the same for both ions, since the physics of the mass spectrometer takes into account the charge level of each different analyte. Our two ions weigh exactly the same, but the mass spectrometer simply can not know that; all it knows is how a given ion reacts to the electric field it is put in. And our two ions, evidently, will react differently.

[1] Interesting posting signed by Ken I. Mitchelhill in the ABRF mailing list at http://www.abrf.org/archives, and a document published by the California Institute of Technology.

When we say that a mass spectrometer measures a m/z ratio, the z in this ratio represents the sum of all the charges (this is a net charge) that sit onto the analyte. But what does the m stand for? The molecular mass? No! The m stands for the mass of the whole analyte ion, which is—in a word—the *measured mass*. This is not the molecular mass (which would be M), it is the molecular mass *plus/less* the mass of the chemical entity that brings the charge to the analyte. When ionizing a molecule, what happens is that something brings (or removes) a charge. In biopolymer chemistry, for example, often the ionization is a simple protonation/deprotonation. If it is a protonation, that means that an electronic doublet (on some basic group of the analyte) captures a proton. This brings the mass of a proton to the biopolymer ($\simeq$ 1 Da). Conversely, if it is a deprotonation (loss of a proton by some acidic group, say a carboxylic that becomes a carboxylate) the polymer looses the mass of a proton. Of course, if the ionization involves a single electron transfer the mass difference is going to be so feeble as to be un-measurable on a variety of mass spectrometers. Let us try to formalize this in a less verbose manner by using a sweet amino acid as an example:

- the non-ionized analyte (Glycine) has the following formula: $C_2H_5O_2N_1$; the molecular mass is thus M = 75.033 Da;
- the analyte gets protonated in the mass spectrometer:

$$C_2H_5O_2N_1 + H \rightharpoonup C_2H_6O_2N_1$$

  the measured mass of the ion is thus m = 75.033 + 1.00782 Da and the charge beared by the ion is thus $z = +1$.

- the peak value read on the mass spectrum for this analyte will thus be (with z = +1):

$$\text{value} = \frac{\text{m}}{\text{z}} = \frac{\text{M} + 1.00782}{\text{z}} = 76.04$$

We see here that the label on the mass spectrum does not correspond to the nominal molecular mass of the analyte: the ionizing proton is "weighed" along with the Glycine molecule. Imagine now that, by some magic, this same Glycine molecule just gets protonated a second time. Let's do exactly the same type of calculation as above, and try to predict what value will be printed onto the mass spectrum:

- the un-ionized analyte (Glycine) has the following formula: $C_2H_5O_2N_1$; the molecular mass is thus M = 75.033 Da;
- the analyte gets protonated in the mass spectrometer *two times*:

$$C_2H_5O_2N_1 + 2H \rightharpoonup C_2H_7O_2N_1$$

  the measured mass of the ion is thus m = 75.033 + 2.01564 Da and the charge beared by the ion is thus z = +2.

- the peak value read on the mass spectrum for this analyte will thus be (with z = +2):

$$\text{value} = \frac{\text{m}}{\text{z}} = \frac{\text{M} + 2.01564}{\text{z}} = 38.52$$

At this point it is absolutely clear that a m/z is not a molecular mass. By the way, if the Glycine happened to be ionized *negatively* the calculation would have been analogous to the one above, but instead of *adding* the mass of the proton(s) we would have *removed* it. Summing up all this in a few words: an ionization involves one or more charge transfer(s) and in most cases (at least in biopolymer mass spectrometry) also involves matter transfer(s). It is crucial *not* to forget the matter transfer(s) when ionizing an analyte. This means that when an ionization process is described, its description ought to be complete, clearly stating three different pieces of information:

- the matter transfer (optional; usually a formula like "+H1");
- the charge transfer (net charge that is brought by the ionization agent);
- the ionization level (the number of ionization event; 0 means "no ionization"; usually this would be 1 for a single ionization, but might be as large as 30 if, for example, a protein was ionized by electrospray. In this case the m/z value would be computed this way (with z = +30):

$$\text{value} = \frac{\text{m}}{\text{z}} = \frac{\text{M} + 30 \cdot 1.00782}{30} = \frac{16959 + 30.2346}{30} = 566.30$$

In the next chapters of this manual *massXpert* will be described so as to let the user take advantage of its powerful capabilities. In a first chapter some general concepts around the way the program behaves will be presented. Next, in the remaining part of this manual, a chapter will be dedicated to each important *massXpert* function or characteristic.

5

massXpert Generalities

In this chapter, I wish to introduce some general concepts around the *massXpert* program.

General *massXpert* Concepts

The *massXpert* mass spectrometry software suite has been designed to be able to "work" with every linear polymer. Well, in a certain way this is true... A more faithful account of the *massXpert*'s capabilities would be: "*The massXpert software suite works with whatever polymer chemistry the user cares to define; the more accurate the polymer chemistry definition, the more massXpert will be accurate*".

For the program to be able to cope with a variety of possibly very different polymers, it had to be written using some *abstraction layer* in between the mass calculations engine and the mere description of the polymer sequence. This abstraction layer is implemented with the help of "polymer chemistry definitions", which are files describing precisely how a given polymer type should behave in the program and what its constitutive entities are. The way polymer chemistry definitions are detailed by the user is the subject of a chapter of this book (see menu *XpertDef* of the program). However, in order to give a quick overview, here is a simple situation: a user is working on two polymer sequences, one of chemistry type "protein" and another one of chemistry type "dna". The protein sequence reads "ATGC", and the dna sequence reads "CGTA". Now imagine that

the user wants to compute the mass of these sequences. How will *massXpert* know what formula (hence mass) each monomer code corresponds to? There must be a way to inform *massXpert* that one of the sequences is a protein while the other is a DNA oligonucleotide: this is done upon creation of a polymer sequence; the programs asks of what chemistry type the sequence to be created is. Once this "chemical parentage" has been defined for each sequence, *massXpert* will know how to handle both the graphical display of each sequence and the calculations for each sequence.

On Formulæ And Chemical Reactions

Any user of *massXpert* will inevitably have to perform two kinds of chemical simulations:

* Define the formula of some chemical entity;
* Define a given chemical reaction, like a protein monomer modification, for example.

While the definition of a formula poses no special difficulty, the definition of a chemical reaction is less trivial, as detailed in the following example. The lysyl residue has the following formula: $C_6H_{12}N_2O$. If that lysyl residue gets acetylated, the acetylation reaction will read this way:—"*An acetic acid molecule will condense onto the ϵ amine of the lysyl side chain*". This can also read:—"*An acetyl group enters the lysyl side chain while a hydrogen atom leaves the lysyl side chain; water is lost in the process*". The representation of that reaction is:

$$R - NH_2 + CH_3COOH \rightleftharpoons R - NH - CO - CH_3 + H_2O$$

When the user wants to define that chemical reaction, she can use that representation: "$-H_2O + CH_3COOH$", or even the more brief but chemically equivalent one: "$-H + CH_3CO$". In *massXpert*, the chemical reaction representation is considered a valid formula.

The *massXpert* Framework Data Format

All the data dealt with in *massXpert* are stored on disk as *XML*-formatted files. *XML* is the *eXtensible Markup Language*. This "language" allows to describe the structure of a document. The structure of the data is first described in a section of the document that is called the *Document Type Definition, DTD*, and the data follow in the same file. One of the big advantages of using such *XML* format in *massXpert* is that it is a text format, and not a binary one. This means that any data in the *massXpert* package is human-readable (even if the *XML* syntax makes it a bit difficult to read data, it is actually possible). Try to

read one of the polymer chemistry definition *XML* files that are shipped with this software package, and you'll see that these files are pure text files (the same applies for the `*.mxp` *XML* polymer sequence files. The advantages of using text file formats, with respect to binary file formats are:

* The data in the files are readable even without the program that created them. Data extraction is possible, even if it costs work;
* Whenever a text document gets corrupted, it remains possible to extract some valid data bits from its uncorrupted parts. With a binary format, data are chained from bit to bit; loosing one bit lead to automatic corruption of all the remaining bits in the file;
* Text data files are searchable with standard console tools (`sed, grep...`, which make it possible to search easily text patterns in any text file or thousands of these files in one single command line. This is not possible with binary format, simply because reading them require the program that knows how to decode the data and the powerful console-based tools would prove useless.

General Chemical Entity Naming Policy

Unless otherwise specified, the user is *strongly* advised *not* to insert any non-alphanumeric-non-ASCII characters (space, %, #, $\$\ldots$) in the strings that identify polymer chemistry definition entities. This means that, for example, users must refrain from using non-alphanumeric-non-ASCII characters for the atom names and symbols, the names, the codes or the formulæ of the monomers or of the modifications, or of the cleavage specifications, or of the fragmentation specifications... Usually, the accepted delimiting characters are '-' and '_'. It is important not to cripple these polymer data for two main reasons:

* So that the program performs smoothly (some file-parsing processes rely on specific characters (like '#' or '%', for example) to isolate sub-strings from larger strings);
* So that the results can be easily and clearly displayed when time comes to print all the data.

6

XpertDef: Definition Of Polymer Chemistries

After having completed this chapter the reader will be able to accomplish the very first steps needed to use *massXpert*'s features at best: the normal workflow, indeed, is to first make a polymer chemistry definition, in order to be able to edit polymer sequences of that specific definition. The *XpertDef* module is made available in *massXpert* by pulling down the *XpertDef* menu item from the program's menu. It is possible to start a new polymer chemistry definition from scratch, but it is certainly usually easier to first duplicate a polymer chemistry definition shipped with *massXpert* and then open that copy and edit it. Please, refer to chapter 10, page 111 for an explanation of how this is safely done.

To open a polymer chemistry definition, the user may either select one that is already registered with the system, and that appears listed in the drop-down list widget shown in Figure 6.1 on the next page or click the Cancel button so as to open one definition file by browsing the filesystem. In the polymer chemistry

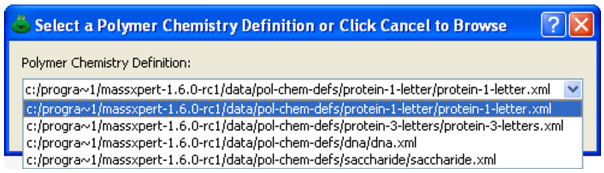


Figure 6.1: **Select one polymer chemistry definition file.** It is possible to immediately select a polymer chemistry definition already registered with the system, or open an arbitrary file by browsing the filesystem (click the Cancel button, hidden in this figure, if so desired).

definition window that shows up, the user accomplishes two different tasks:

* Define the name of the polymer chemistry definition;
* Define "singular" data like the left cap and the right cap of the polymer, the ionization rule governing the default ionization of the polymer sequence;
* Define the atoms needed to operate the different polymer chemistry entities (these are "plural" data) ;
* Define all the polymer chemistry entities needed to work on polymer sequences (all these are also "plural data") .

The definition of the atoms and of all the chemical entities belonging to a given polymer chemistry are collectively called a *polymer chemistry definition*. The polymer chemistry definition window that shows up is shown in Figure 6.2 on the facing page.

The Atoms

The definition of the atoms is performed through the user interface shown in Figure 6.3 on page 46 (Atoms button in the polymer chemistry definition window). In this dialog, the user defines chemical elements (atoms) as entities made of isotopes (at least one isotope per atom, logically).

The design of this dialog window follows the general design for all the dialog windows related to the definition of plural data in the polymer chemistry definition. The leftmost list widget (Atoms) lists the final object as defined and available in the polymer chemistry definition (in this case the atoms), while the second list widget (Isotopes) lists the objects that are defined in order to actually make the selected object in the first list widget (thus, atoms are made of isotopes). We see that two isotopes were defined in order to create the *Carbon* atom.

To add a new atom, the user clicks the Add button below Atoms list widget, which triggers the insertion of a new row in the list widget. The Details groupbox on the right side of the dialog window now shows *Type name* as the name of the atom and *Type symbol* as its symbol. The list of isotopes is empty, because we still did not define any. First thing to do is to actually give the atom a name

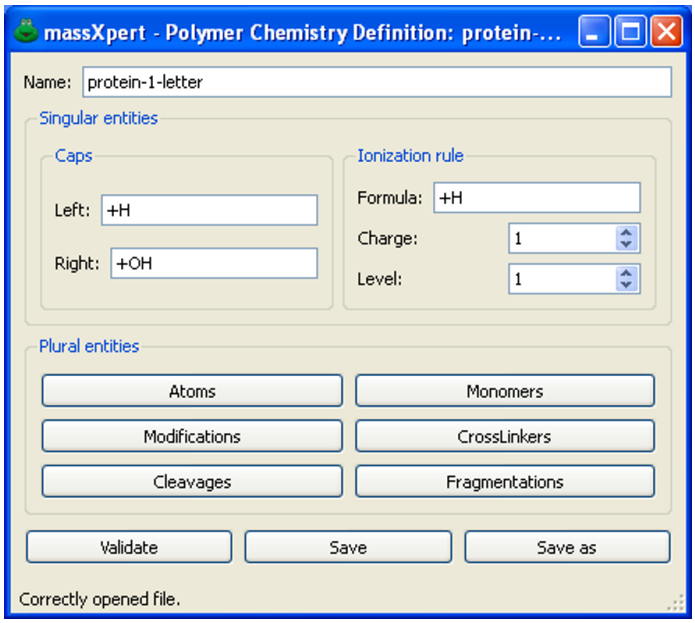


Figure 6.2: ***XpertDef* polymer chemistry definition window.** All the polymer chemistry entities are defined in this window. The different buttons dealing with atoms, monomers, modifications, cross-linkers, cleavage and fragmentation specifications open up specific dialogs (see below).

and a symbol. The are no length limitations to any of the new data, but a reasonable limit is 3 characters for the symbol, the first being uppercase and all the remaining ones lowercase. Use only alphabetic characters (that is [a-zA-Z]). Once these two data are set, click on to the Apply button; the list widget item will be updated to reflect the new atom name.

To add a new isotope, first select the atom to which it should be added. Click on the Add button below the Isotopes list widget. A new item will be added to the list widget with text *0.0000000000*. Enter the mass/abundance data in the Isotope groupbox and click Apply. The corresponding item in the list widget will be updated (the mass of the isotope is displayed in the list widget). Each time a modification is performed in the list of isotopes of a given atom, the monoisotopic and average masses are updated in the Atom groupbox. Recalculation of the average mass is automatic as soon as something is modified in the list of isotopes.

Other buttons, like Move up or Move down, are self-explanatory. Before moving on, please, validate the atom definitions by clicking onto the Validate button.

The Polymer Chemical Entities

Once the atoms have been properly defined (note that such atoms are already available in the distributed package), it is possible to start entering data for

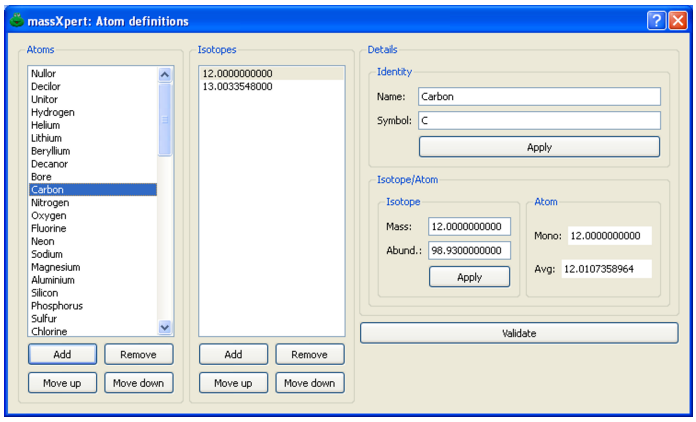


Figure 6.3: ***XpertDef* atom definition.** Each chemical element must contain at least one isotope, otherwise it does not have any *"raison d'être"*.

the other polymer chemical entities. These are often defined using chemical formulas, which explain why it is necessary to first define the atoms.

The following are the data that need to be entered so as to obtain a usable polymer chemistry definition:

* The polymer chemistry definition Name *protein-1-letter* Name of the polymer chemistry definition;
* Caps Chemical capping reactions that should happen on the left end (Left) and on the right end (Right) of the polymer sequence:
  - Left $+H$ Left capping of the polymer sequence;
  - Right $+OH$ Right capping of the polymer sequence;
* Polymer Ionization rule This rule describes the manner in which the polymer sequence should be ionized by default, when the mass is calculated. This rule actually holds two elements:
  - Formula $+H$ Chemical reaction that ionizes the polymer sequence. In the example, all the polymer sequences of polymer chemistry definition "protein-1-letter" are protonated by default;
  - Charge *1* Charge that is brought by the chemical agent ionizing the polymer (the formula above). In the example, a protonation reaction brings a single positive charge.
  - Level *1* Number of times that the ionization must be performed by default on any polymer sequence of this chemistry definition. In this example, monoprotonation is set as the default ionization rule.

At this point, time has come to deal with "plural" data. The first chemical entities to deal with are monomers.

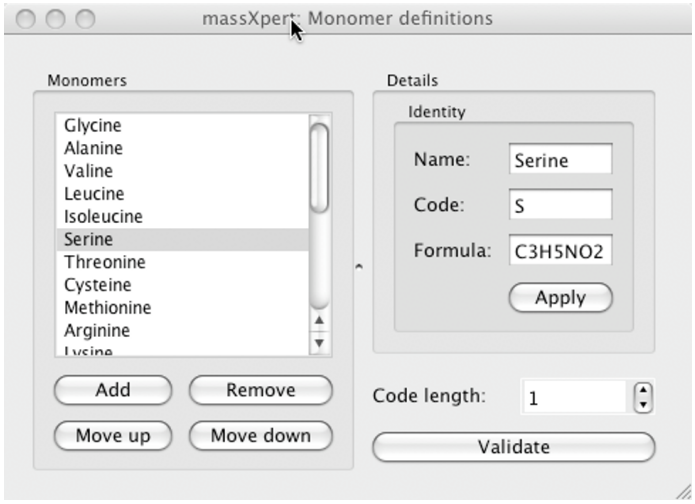


Figure 6.4: ***XpertDef* monomers definition.** Each monomer is defined using a name, a code and a chemical formula.

The Monomers

The monomers are the constitutive blocks of the polymer sequence. In the *massXpert*'s jargon, "monomer" stands *not* for the molecule that may be used to perform a polymer synthesis; it stands for this molecule *less* the chemical group(s) that were eliminated upon polymerization. If this concept is not familiar to the reader, it might be useful to read chapter 3 on page 15 for an overview of polymer chemistry.

Click onto the Monomers button, which triggers the opening of the dialog window shown in Figure 6.4.

The way this dialog is operated is similar to what was described for the atom, unless it is simpler, because monomers are non-deep objects: there are no contained objects. One data element is critical: the number of characters that might be used to define the code of the element cannot be greater than the value entered in Code length spinbox widget[1]. The fundamental rule is the following:

> *"The first character of a monomer code must be uppercase, while the remaining characters (if any) must be lowercase."* That means that—if Code length is *3*—'A', "Al", "Ala" would be perfectly fine, while "Alan", "AL", 'a', "AlA" would be wrong.

After addition of the monomers it is always a good idea to validate them by clicking onto the Validate button.

[1] Allowing more than one letter to craft monomer codes might seem trivial at first. But that design decision triggered the requirement for non-trivial algorithms throughout all the code of the of program. This is easily understandable at least in the polymer sequence editor: how are monomer codes keyed-in if 'A' and "Ala" are valid monomer codes in a polymer chemistry definition? The magic is described in the chapter about *XpertEdit* (see chapter 8 on page 67)

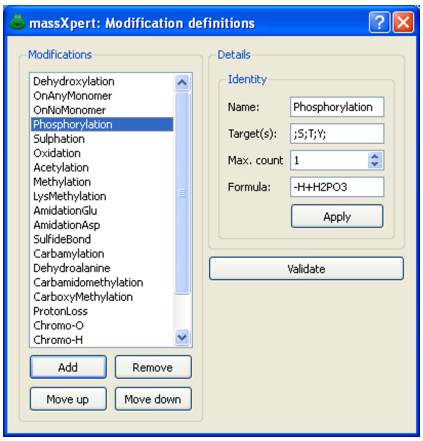


Figure 6.5: ***XpertDef* modifications definition.** Each modification is defined using a name, a targets specification and a chemical formula.

The Modifications

Polymer are often either chemically or biochemically modified. In nature, biopolymers are modified more often than not. One of the more common modifications in the protein reign are phosphorylation or acetylation. Nucleic acids are modified with a sheer number of chemical modifications, saccharides also. The *massXpert* software provides entire freedom to define any number of intelligent modifications, that is modifications with any chemical formula but also that are knowledgeable of what monomers they can modifiy. Indeed, it would make no sense to phosphorylate a glycyl residue in a protein, for example.

Click onto the Modifications button, which triggers the opening of the dialog window shown in Figure 6.5. In the example shown, the *Phosphorylation* modification is being defined. A modification is defined by a Name, a list of monomer codes that might be modified by this modification: Targets[2], a Max. count describing the maximum number of times that modification can be applied to the target monomers[3] , and finally a Formula. The formula is actually a chemical reaction, as explained in section 5, chapter 5, page 39. The *Phosphorylation* reaction can thus be read like this: —*"The polymer looses a proton and gains H2PO3"*. The *Phosphorylation* is being defined as having *S;T;Y* targets only, that means that when the user will try to modify non-seryl or non-threonyl or non-tyrosinyl monomers, the program will complain that these monomers are not targets of *Phosphorylation*. There is, however, and for maximum flexibility, the possibility to override these target-limiting data when modifying monomers. When the polymer is modified with this modification, its masses will change according to the net mass of this *Phosphorylation* "reaction".

[2] A Targets datum is made of monomer codes separated by ';' separators.

[3] This feature is essential when working on methylation of proteins, for example, with arginyl and lysyl residues being multi-methylated.

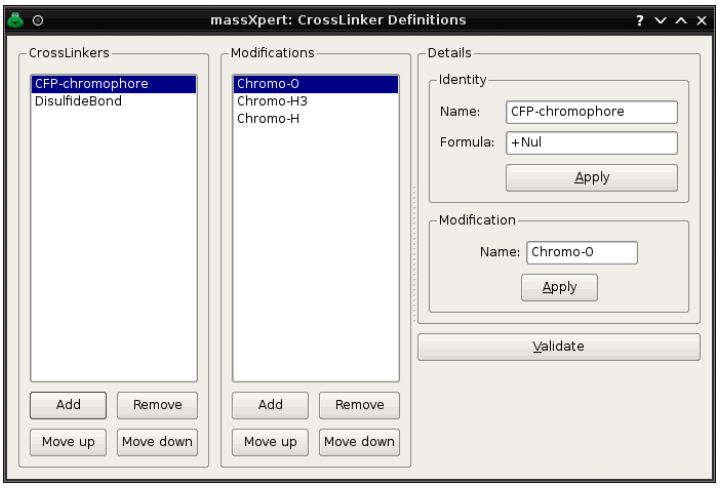


Figure 6.6: ***XpertDef* cross-linkers definition.** Each cross-linker is defined using a name, a formula and either no modification or as many modifications as there are monomers involved in the formation of the cross-link.

The Cross-linkers

Polymers are often either chemically or biochemically modified by interconnecting monomers from the same polymer sequence. In the protein reign, one classical example of intra-sequence cross-linking is the formation of disulfide bonds. Another wonderful example is the formation of the fluorophore in the fluorescent proteins: there is a chemical reaction involving the side chains of three consecutive residues going on, resulting in the formation of a complex intra-sequence cross-link. Each side chain of the three monomers involved are chemically modified.

Cross-linkers are defined in the dialog window shown in Figure 6.6. This dialog window is opened by clicking onto the CrossLinkers button.

The formation of cross-link between one or more monomers often involves chemical reactions to occur at the level of the engaged monomers. Cross-linkers defined in *massXpert*should refer to these modifications as modification objects already available in the polymer chemistry definition. Note that, in some cases, it is not necessary to define modifications to occur at the level of the cross-linked monomers. The example described in Figure 6.6, corresponds to the cross-linking reaction involved in the formation of the chromophore of the cyan fluorescent protein. That reaction involves the three following monomers: ^{65} Threonyl, ^{66} Tyrosinyl, ^{67} Glycyl. Each monomer undergoes a distinct chemical modification: "-0", "-H3" and "-H", respectively. Three modifications were thus defined: *Chromo-0*, *Chromo-H3* and *Chromo-H*, in that specific order, as these modifications are going to be sequentially applied to their corresponding monomer in the cross-linking reaction.

Note that the formula of the *CFP-Chromophore* cross-linker is *+Nul*, that is there is no chemical reaction defined for the cross-linker *per se*. When modifications are defined, their number must match the number of monomers involved, and their order must match the order with which the monomers are cross-linked.

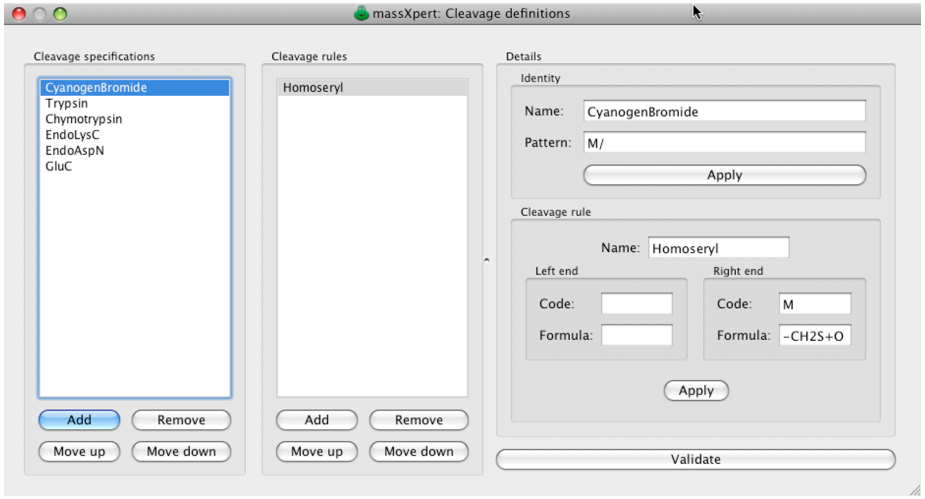


Figure 6.7: ***XpertDef* cleavage specifications definition.** Each cleavage specification is defined using a name, a cleavage pattern and any number of cleavage rules.

If no modification is defined, then, the chemical reaction that occurs upon cross-linking might be defined in the formula of the cross-linker.

The Cleavage Specifications

It is common practice—in biopolymer chemistry, at least—to cut a polymer into pieces using molecular scissors like the following:

* proteases, for proteins;
* nucleases, for nucleic acids;
* glycosidases, for saccharides...

For each different polymer type, the molecular scissors are specific. Indeed, a protease will not cleave a polysaccharide. This is why cleavage specifications belong to polymer chemistry definitions. In the example of Figure 6.7, the definition of the *CyanogenBromide* cleavage specification is detailed (this organic reagent cleaves right of methionyl residues). The *CyanogenBromide* cleavage specification is qualified as so:

* Name *CyanogenBromide* Name of the cleavage agent;
* Pattern *M/* Sequence specificity of the cleavage agent. In this case, the cleavage agent cleaves the protein right after *M*ethionyl residues;
* Cleavage rule This groupbox allows the definition of the cleavage rules that might be added to the cleavage specification:

- Left Code and Left Formula (Empty) This is a special case for those cleavage agents that not only cut a polymer sequence (usually it is a hydrolysis) but that also modify the substrate in such a way that must be taken into account by *massXpert* so that it computes correct molecular masses for the resulting oligomers. These rules are optional. However, if Left Code is filled with something, then it is compulsory that Left Formula be filled with something valid also, and conversely;
- Right Code and Right Formula *M* and *-CH2S+O3*, respectively. Same explanation as above. This cleavage rule stipulates that upon cleavage of a protein using cyanogen bromide, the methionyl residue that gets effectively cleaved must be converted to a homoseryl residue. See below for a detailed explanation.

Here are some examples of more complex cleavage patterns:

* **Trypsin = K/;R/;-K/P** *"Trypsin cuts right of a 'K' and right of a 'R'. But it does not cut right of a 'K' if this K is immediately followed by a P"*;
* **EndoAspN = /D** *"EndoAspN cuts left of a D"*;
* **Hypothetical = T/YS; PGT/HYT; /MNOP; -K/MNOP** *"Hypothetical cuts after 'T' if it is followed by YS and also cuts after 'T' if preceded by PG and followed by HYT. Also, Hypothetical cuts prior to 'M' if 'M' is followed by NOP and if 'M' is not preceded by K"*.

Please, *do* note that the letters in the examples above correspond to monomer codes and *not* to monomer names. If, for example, we were defining a "Trypsin" cleavage specification pattern—in a protein polymer chemistry definition with the standard 3-character monomer codes—we would have defined it this way: "Trypsin = Lys/;Arg/;-Lys/Pro".

Now comes the time to explain in more detail what the Left Code and Left Formula (along with the Right siblings) are for. For this, we shall consider that we have the following polymer sequence (1-character monomer codes): THISMWILLMBECUTMANDTHATMALSO. If that sequence had been cleaved using "CyanogenBromide" and if the cleavage had been total,[4] that would have generated the following oligomers: THISM WILLM BECUTM ANDTHATM ALSO. But if there had been partial cleavages, one or more of the following oligomers would have been generated: THISMWILLM BECUTMANDTHATM ALSO WILLMBECUTM ANDTHATMALSO and so on...

Now, the biochemist knows that when a protein is cleaved with cyanogen bromide, the cleavage occurs effectively right of monomer 'M' (this we also know already) *and* the 'M' monomer that underwent the cleavage is changed from a methionyl residue to an homoseryl residue (this chemical change involves this formula: "-CH2S+O"). Amongst all the oligomers generated above, there are two oligomers that should not undergo the cleavage rule "-CH2S+O": ALSO and ANDTHATMALSO. Indeed, these two oligomers were generated by the "CyanogenBromide" cleavage, but were not actually cleaved at the right side of a methionyl

[4] Cleavage occurs at every possible position, right of each monomer 'M'.

residue, because they correspond the the right end terminal part of the protein sequence (even if one them does contain a 'M' residue; the cleavage did not *occur* at that residue).

This example should clarify why the definition clearly stipulates—in the cleavage specification for "CyanogenBromide"—that the oligomers resulting from this cleavage should *"undergo the '-CH2S+O' formula only if they have a 'M' as their right end monomer code"*. These *cleavage rule*s need to be defined in a very careful way: imagine that—in some cyanogen bromide experiments—that reagent would cleave right of 'C' (cysteine) residues, but with no chemical modification of the 'C' monomer.[5] In this case, it would be suitable to put the flexibility of *massXpert* at work by specifying that the generated oligomers should "undergo the '-CH2S+O' formula" only if they have a 'M' as their right end monomer, so that 'C'-terminated oligomers are not chemically modified. Thus the cleavage pattern might be safely defined: "M/;C/"...

The Fragmentation Specifications

As previously discussed (chapter 3, section 3 on page 25), specifying the fragmentation specifications of a polymer chemistry definition is not a trivial task. In this section three different cases will be described, from simple to more complex.

Simple fragmentation patterns

One simple example of polymer chain fragmentation is the formation of *a* fragments with a nucleic acid (DNA, in this example). The fragments obtained by *a*-type fragmentation are described in Figure 3.9 on page 30. Bond cleavage occurs right before the sugar-carbon-linked oxygen of the phosphoester bond linking one deoxyribonucleotide to the next. Thus, the molecular weight of the fragment corresponds, as illustrated, to the sum of the monomer masses from the left end of the polymer up to and including the monomer being decomposed *less* one oxygen. Thus, the formula of the *a* fragmentation pattern is "-O". Therefore, the definition of the *a* DNA fragmentation pattern is as described in Figure 6.8 on the facing page, were wee see that the Name of the fragmentation specification for *a* fragments is a, that the Formula is *-O*, that the fragments encompass the *LE* (for "left end") End of the polymer chain. The Side chain value is set to *0*, which will be explained later.

More complex fragmentation patterns

In nucleic acids gas-phase chemistry, it often happens that not only fragmentation occurs at the level of the phospho-ribose skeleton, but also at the level of the nucleic base. These fragmentation patterns are called abasic patterns. The decomposition of the base occurs at the monomer position where the fragmentation occurs. For example, if a "ATGC" oligonucleotide is fragmented according to pattern *a* but with nucleic base decomposition, and that fragmentation occurs at position 1, then the computation of the mass should occur like represented in

[5] This is a purely hypothetical situation that I never observed personally.

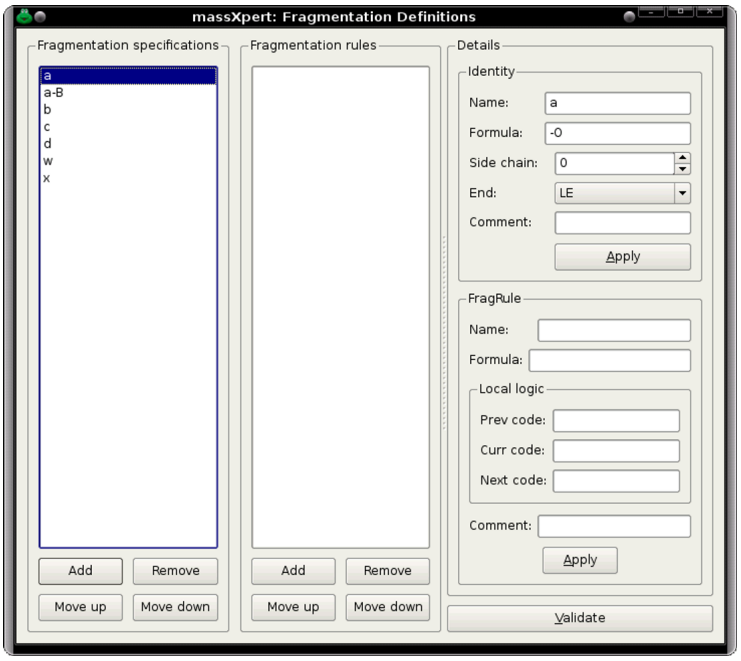


Figure 6.8: ***XpertDef* fragmentation rules definition.** Each fragmentation rule is defined using a name, a formula and a local logic, that is a set of logial conditions which must be verified for the fragmentation rule to be applied to the fragment.

Figure 6.9 on the next page. This figure illustrates a number of things, amongst which some known basics. The panel on the top right hand side shows the constituents of the DNA polymer chemistry definition: the caps are OH on the left end and H on the right end; the circled formula is the skeleton (also called backbone) and the base attached to the deoxyribose ring singularizes the nucleotide. That base might be adenine, guanine, cytosine, thymine. In the "dna" polymer chemistry definition, the monomers are made of the skeleton (formula C5H8O5P) plus the formula of the base, which is understandable.

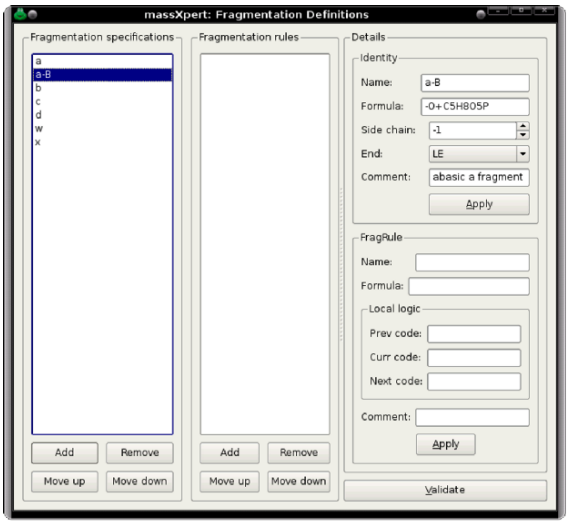
Using a generalizable specification Now, if we want to compute the mass of the a-B#1 fragment, that is fragmentation occurs according to pattern *a* right after the 'A' monomer *plus* decomposition of the base (in our case this is an Adenine) we have to:

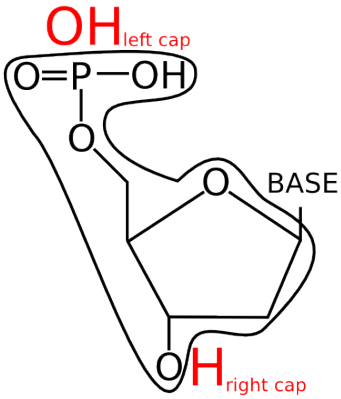
* Apply the specification for *a* fragments (that is, remove one oxygen, the *-O* component of the formula);
* Remove one *full monomer* with Side chain set to *-1* (this equals to the removal of both the skeleton and the side chain—the adenine, here);
* Add back the skeleton (the *+C5H8O5P* component of the formula);
* As for *a* fragments, the end of the polymer sequence that gets included in the fragment is the *LE* ("left end").

The advantage of working this way is that we need not specify a fragmentation rule for each different monomer in the sequence (see below, for how this might be done). Indeed, by specifying Side chain to be *-1*, we indicate—without knowing the monomer identity—to the mass calculation engine that once the fragmentation has occurred in the polymer chain, the mass of the monomer that got fragmented should be subtracted from the fragment mass. That subtraction removes, however too much material, as we do not want to loose the skeleton, we only want to loose the base (adenine, in our example). This is why we ask in the fragmentation specification formula that the skeleton be added (the *+C5H8O5P* component of the formula). Because the skeleton does not change along the polymer chain, even if the base itself changes, this computation method is generalizable, and because of this the polymer chemistry definition works.

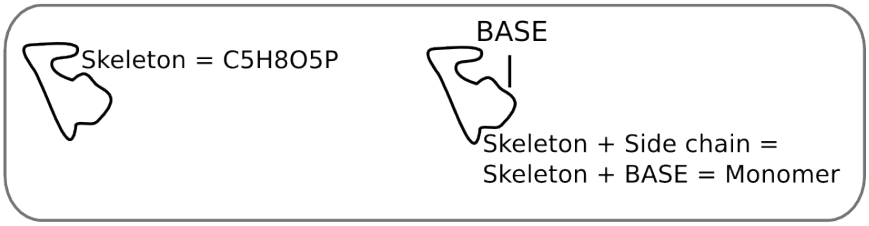
This whole process of defining a fragmentation pattern that needs to "know" what monomer is being fragmented so as to compute the fragment masses correctly, can be performed by using fragmentation rules. This is described below.

Using a monomer-specific specification Another way of achieving what was described above is by using fragmentation rules, whereby the fragment's mass computation is made conditional to one or more conditions that should be verified. Figure 6.10 on page 56 shows how the *a-B* fragmentation pattern might be defined using fragmentation rules.
The *a-B* fragmentation specification comprises 4 rules, one rule for each available monomer in the "dna" polymer chemistry definition: 'A', 'T', 'G' and 'C'. The figures illustrates the definition of the fragmentation specification *a-B* which stipulates that the mass of the fragment should be computed this way:






```
Adenine:  C5H4N5   (monomer: C10H12N5O5P) mono: 313.058
Guanine:  C5H4N50  (monomer: C10H12N5O6P) mono: 329.053
Cytosine: C4H4N30  (monomer: C9H12O6N3P)  mono: 289.040
Thymine:  C5H5N2O2 (monomer: C10H13N2O7P) mono: 304.050
```




```
a#1= monomer - O + left cap =
     monomer - O + OH = 313 + 1 = 314

a-B#1= monomer - O + left cap - 1x(side chain) =
       monomer - O + OH - monomer + skeleton =
       314 -16 + 17 - 314 + C5H8O5P =
       1 + 179 = 180 (This is the right result)
```

Figure 6.9: ***XpertDef* fragmentation specifications definition.** Each fragmentation specification is defined using a name, a formula, the fragmented monomer side chain contribution, the end of the polymer that is contained in the fragment and any number of fragmentation rules.

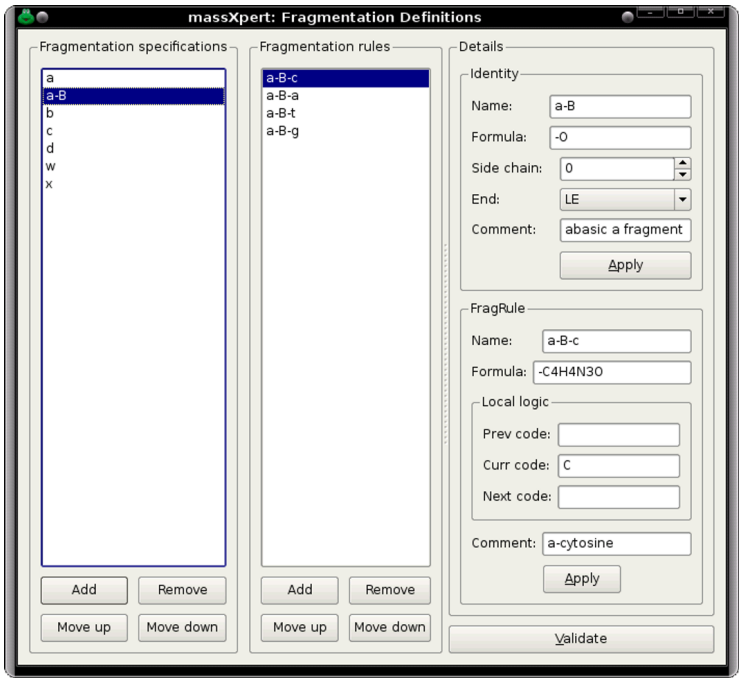


Figure 6.10: ***XpertDef* fragmentation rules definition.** Each fragmentation rule is defined using a name, a formula and a local logic, that is a set of logical conditions which must be verified for the fragmentation rule to be applied to the fragment.

* For the fragmentation specification part, everything is like for fragments of type *a*, that is, the formula is merely -*O* and the end is *LE* (see above, for explanations);
* But there is one rule (*a-B-c*) which adds some Local logic for the fragmentation specification: the formula -*C4H4O3N* should be applied upon calculation of the fragment's masses if the monomer at which the fragmentation actually occurs is of Curr code *C*, that is if it is a Cytosine. The -*C4H4O3N* formula is the formula of Cytosine (the base, not the monomer).
* The other rules (for Curr code *A*, *T* and *G* are identical to the *a-B-c* one unless the Curr code is 'A', 'T' or 'G' and the formula to be removed is the formula of the corresponding DNA base.

The fragmentation rule-based definition of fragmentation pattern *a-B* yields identical results as for the more generalizable method described earlier.

Even more complex fragmentation patterns

Note that in saccharide chemistry, the fragmentation patterns are extremely complex, and often totally depend on the nature of the monomers local to the fragmentation site. For example, the fragmentation behaviour at position 'E' in a sequence "DEAR" might be different than in a sequence "DERA". *massXpert* had to be able to model these complex situations, and this is done using fragmentation rules where the local logic involves defining the Prev code and/or the Next code for a given Curr code at which the fragmentation occurs. For example, one specific fragmentation pattern for fragmentation at 'E' in sequence "DEAR" might be defined this way:

* Prev code: *D*;
* Curr code: *E*;
* Next code: *A*.

In stead of that fragmentation rule, one would have for fragmentation at 'E' in sequence "DERA" the following rule:

* Prev code: *D*;
* Curr code: *E*;
* Next code: *R*.

Note the change for Next code, from *A* to *R*. Also, be aware that the "Prev", "Curr" and "Next" notions are polar, that is, they depend on the value of End (that is *LE* or *RE*). For example, if we wanted to model the fragmentation pattern at 'E' for a fragment of End *RE*, similar to what was done above with sequences "DEAR" and "DERA", we would have set the local logical like this:

For sequence "DEAR":

* Prev code: *A*;
* Curr code: *E*;
* Next code: *D*.

For sequence "DERA":

* Prev code: *R*;
* Curr code: *E*;
* Next code: *D*.

This highly flexible fragmentation specification allows for definition of highly complex fragmentation behaviours of biopolymers.

Saving The Definition

Once the polymer chemistry definition is completed, the user can save it to an *XML* file. Prior to actually writing to the file, the program checks the validity of all the chemical entities in the definition. This check can be triggered manually by clicking onto the Validate button. If an error is found, it is reported so that the user may identify the problem and fix it.

The location where the file should be saved, and the manner that it may be made available to *massXpert* is to be described in a later chapter. It is, in fact, very important that *massXpert* knows where to find newly defined polymer chemistries so as to be able to use them when sequences of that polymer chemistry are created or used.

7

XpertCalc: A Powerful Mass Calculator

After having completed this chapter you will be able to perform sophisticated polymer chemistry-aware mass calculations.

XpertCalc Invocation

The *XpertCalc* module is easily called by pulling down the *XpertCalc* menu item from the *massXpert* program's menu. The user is presented with a window to select the polymer chemistry definition that should be used for the calculations (Figure 7.1).

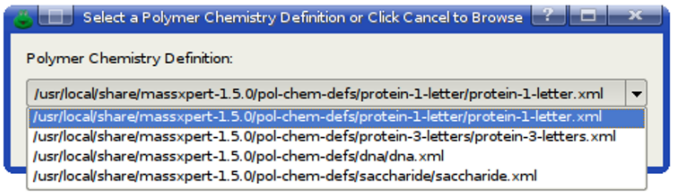


Figure 7.1: **Selecting a polymer chemistry definition for use with *XpertCalc*.** This figure shows that the user can either select one already registered polymer chemistry definition (listed in the drop-down widget) or browse the filesystem to select one polymer chemistry definition file. Choosing a polymer chemistry definition allows to take advantage of all the chemical entities defined therein during the mass calculations.

An Easy Operation

Once the polymer chemistry definition has been correctly selected, it is parsed by the *XpertCalc* module and its entities are automatically made available in the calculator window, as shown in Figure 7.2. The way *XpertCalc* is operated is very easy. This is partly due to the very self-explanatory graphical user interface of the module, which is illustrated in Figure 7.2. *XpertCalc* can handle a number of items that are reviewed below:

* Seed Masses The user may (is not obliged to) seed the calculation by setting masses manually in these line edit widgets (the left line edit is for mono and the right one for avg; both monoisotopic and average m/z values need to be entered). For example, imagine that a mass spectrum analysis session ends up like this: —*"There is a peak with m/z 1000.55, z=1 and another one roughly 80 Da more. Is it possible that the analyte showing up at m/z 1000.55 is phopshorylated?"* The massist would seed the calculator with mass *1000.55* and ask that one *Phosphorylation* modification be added to it by setting *1* in front of the corresponding drop-down widget. Clicking onto Apply triggers the calculation, with the resulting masses being displayed in the Result Masses line edit widgets. We can see that the phosphorylation of our analyte shifts its m/z value from *1000.55* to *1080.5163*.

  Note that each time a calculation is triggered by clicking onto Apply, the values already present in the Result Masses line edit widgets are transferred to the Seed Masses line edit widgets. This provides a 1-level undo;

* Result Masses Each time a calculation is triggered by clicking the Apply button (or the chemical pad's buttons; see below), the newly obtained masses are displayed in these line edit widgets. The values that were displayed there previously are transferred to the Seed Masses line edit widgets, thus providing a 1-level undo.

* Formula This group box widget contains two widgets: a line edit widget where the formula is typed and a count spin box widget where the user

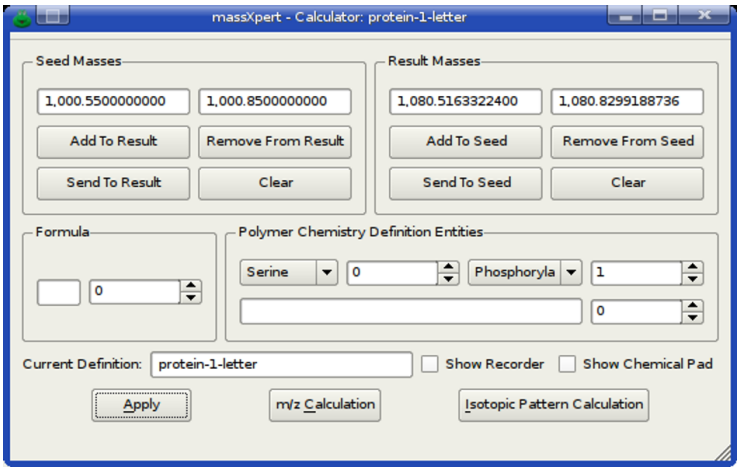


Figure 7.2: **Interface of the *XpertCalc* module.** This figure shows that the *XpertCalc* polymer chemistry definition-aware module can handle atoms, formulæ, monomers, modifications and even polymer sequences for computing masses.

sets the number of times that the formula should be applied. Setting the formula to *H2O* and the count to *2* would hydrate the analyte twice.

* Polymer Chemistry Definition Entities This group box widget contains two drop-down widgets and a line edit widget. The drop-down widget on the left lists all the monomers defined in the *protein-1-letter* polymer chemistry definition; the drop-down widget on the right lists all the modifications defined in the *protein-1-letter* polymer chemistry definition. Each drop-down widget has its corresponding count spin box widget. In the example, the user asked that one (*1*) *Phosphorylation* modification be applied during the calculation. The line edit widget below the first row of widgets is the polymer sequence widget where the user might enter a sequence of monomers. It is possible to apply many times the sequence by setting the count spin box widget value to something greater than 1 (either positive or negative);

It is possible to perform a set of calculations in one go, that is, the user may ask for a formula, a monomer, a modification, a sequence to be accounted in one single calculation operation. Other prominent features of *XpertCalc* are described in the following sections.

The Programmable Calculator

For the scientists who work on molecules that are often modified in the same usual ways, *XpertCalc* features a built-in mechanism by which they can easily program their calculator. This programming involves the definition of how a *chemical pad* (or *chempad*) may be arranged, exactly the same way as a desktop calculator would display its numerical keypad.

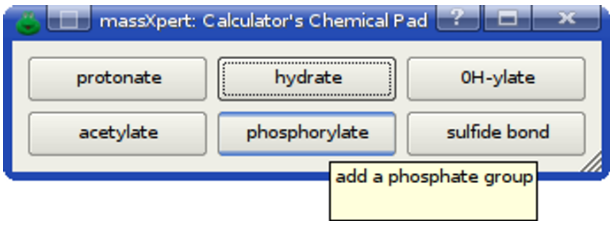


Figure 7.3: **Interface of the chemical pad.** This figure shows that the chemical pad is very similar to what a numerical calculator would display. Here, the user has programmed a number of chemical reactions.

The chemical pad can be shown/hidden by using the Show Chemical Pad check box widget. An example of such a chemical pad is shown in Figure 7.3, where a "protein-1-letter" polymer chemistry definition-associated chempad is featured. As shown, the user has programmed a number of chemical reactions that may be applied to the masses in the *XpertCalc* calculator window by simply clicking on their respective button. The configuration of the chemical pad is very easy, as shown in the code below:

```
chempad_columns$3

chempadkey=protonate%+H1%adds a proton
chempadkey=hydrate%+H2O1%adds a water molecule
chempadkey=0H-ylate%+O1H1%adds an hydroxyl group
chempadkey=acetylate%-H1+C2H3O1%adds an acetyl group
chempadkey=phosphorylate%-H+H2PO3%add a phosphate group
chempadkey=sulfide bond%-H2%oxydizes with loss of hydrogen
```

What this text file says is very simple:

* That the buttons should be arranged in rows of three columns;
* Follows the description of a number of buttons (chempad keys) to be laid out in the chempad (each line describes one button).

Each button is defined in a line that begins with the text `chempadkey=`. Let's look at one button definition, the "phosphorylate" button. The `phosphorylate` text string after the `=` character is the label that will decorate the button that is being configured. The `-H+H2PO3` text string is the formula that should be applied to the result masses in the *XpertCalc* main window when this button is clicked; that's a chemical reaction, in fact. The `add a phosphate group` is a text string that is displayed as a tooltip when the mouse cursor stays for a number of milliseconds over the button (see Figure 7.3).

The LogBook Recorder

Each time an action that is chemically relevant—from a molecular mass point of view—is performed, the program dumps the calculations to the *XpertCalc*

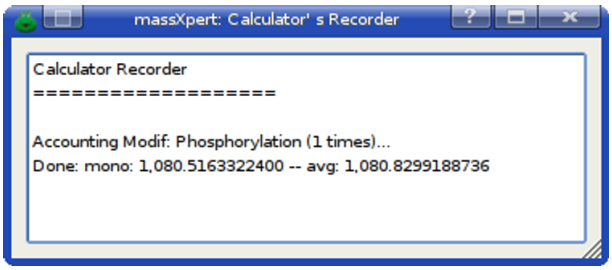


Figure 7.4: **The *XpertCalc* recorder window.** This figure shows that the recorder window is a simple text edit widget that records all the mass-significant operations in the *XpertCalc* calculator. The text in the recorder may be selected and later used in an electronic logbook or printed.

recorder window (Figure 7.4). The recorder can be shown/hidden by using the Show Recorder check box widget. The text in the recorder window is editable for the user to edit the *XpertCalc* output, and selectable also, so that pasting to text editors or word processors is easy *via* the clipboard.

The m/z Ratio Calculator

It very often happens that the massist doing electrospray analyzes is faced with a challenging task: to compute by mind all the m/z ratios for a given family of charge peaks. To ease that daunting task, *XpertCalc* contains a m/z ratio calculator that is called by clicking onto the m/z Calculation button. This action pops up a window that is shown in Figure 7.5.

In order to compute the m/z ratios requested by the user, the program needs to have some seeding data, which have to be entered in the Initial Status frame widget. Of course, initial m/z values have to be entered (both monoisotopic and average m/z values need to be entered). The user must inform the calculator about how these m/z values were calculated, that is, what was the ionization status of the analyte when these m/z values were measured (either virtually or effectively). These ionization data are entered in the Ionization Rule frame, which contains one line edit widget and two spin box widgets. The Formula line edit widget lets the user indicate the ionization agent (for us it is a protonation). The Charge and Level widgets let the user indicate what is the charge brought by the Formula and the number of such ionization event. In the example, the protonation brings one (*1*) positive charge, and the m/z value corresponds to a mono-protonation of the analyte.

With all these data, the m/z ratio calculator can "reverse-compute" the *molecular* mass of the analyte (not the ion mass). That molecular mass will then be used to perform the requested m/z ratio calculations (Target Ionization Status frame, which behaves identically to the one described above). The computed m/z ratios are displayed in a treeview widget (Ion Charge Family).

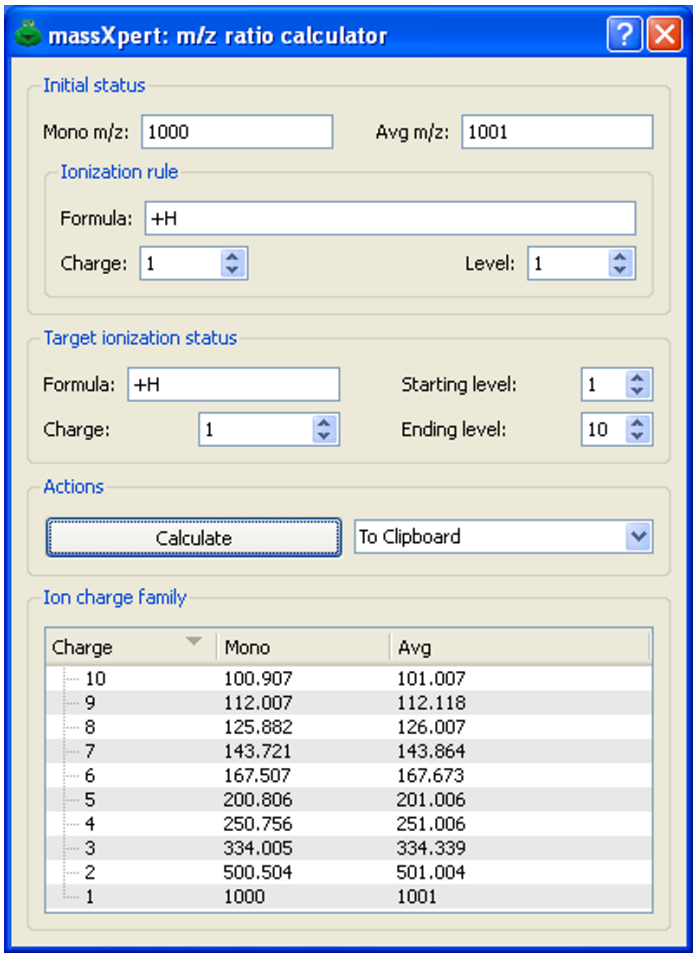


Figure 7.5: **The m/z ratio calculator.** The m/z calculator is rather straight forward to use. Given some initial parameters, the results are displayed in the Ion Charge Family treeview widget.

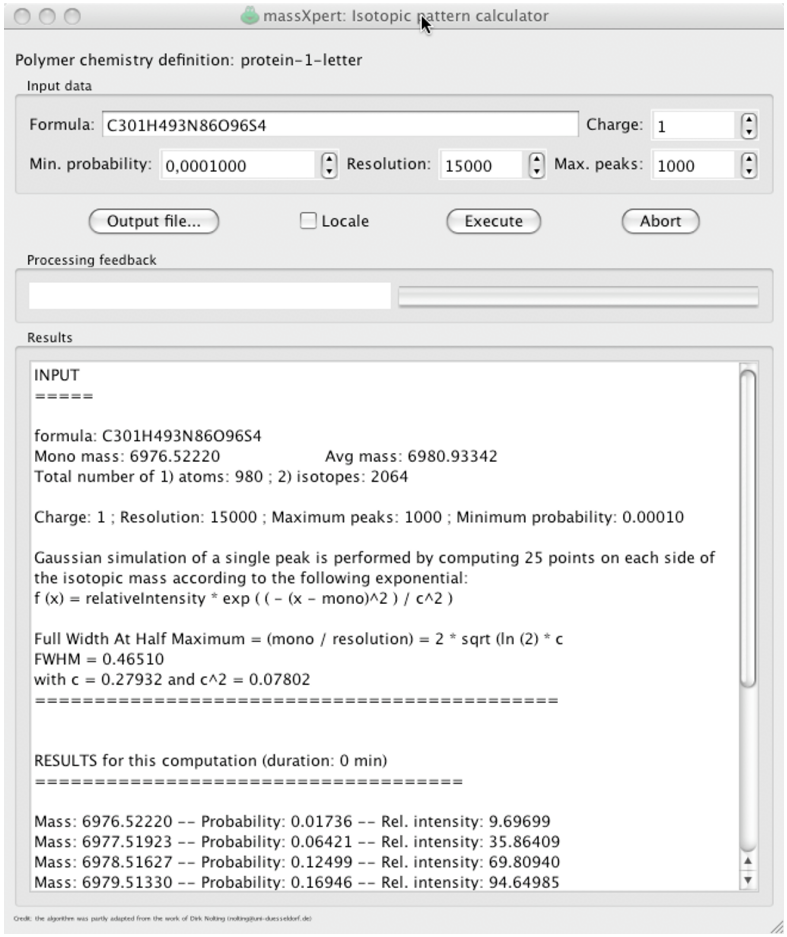


Figure 7.6: **The isotopic pattern calculator.** The isotopic pattern calculator is rather straight forward to use. Given some initial parameters, the results are displayed in the Results text edit widget.

The Isotopic Peaks Calculator

It is sometimes useful to predict (or calculate *a posteriori*) the isotopic peaks pattern of a given analyte. This calculation takes a number of parameters, as shown in Figure 7.6:

* Formula Formula of which the isotopic pattern calculation must be performed. This formula might correspond to a peptide, for example;
* Charge The charge of the analyte;
* Min. Probability The minimum probability value to find a given m/z peak in the isotopic pattern. This allows a degree of optimization when calculations are too long to perform, by removing all isotopic peaks for which the probability of occurrence is lower than the set value;

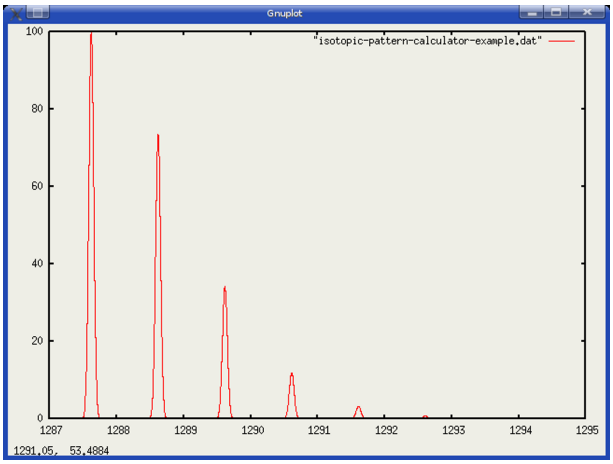


Figure 7.7: **An isotopic pattern calculator output example.** The graph shows the isotopic pattern that should be expected to be obtained by performing mass spectrometric analysis of the analyte of formula C61H87N14O15S1.

* Resolution Resolution of the mass spectrometer. Should be of a compatible value with respect to the m/z of the analyte;

* Max. Peaks Maximum number of peaks in the isotopic pattern. This allows a degree of optimization when calculations are too long to perform by limiting the number of isotopic peaks in the pattern to the set value (the number of peaks in the isotopic peaks pattern increases exponentially with the numer of atoms);

* Output File... Button to click so as to choose a file in which all the data are to be stored for later plotting of the isotopic peaks pattern graph;

* Locale If checked, the results should be displayed (or written to file) using the current locale. It might be useful *not* to check this check box widget in case the plotting program does not understand numerical values as produced by the currrent locale. For example, some plotting programs do not understand values like *140,000.00* (that is one hundred and fourty thousands with a comma separating thousands and dot as the decimal separator).

The Results text edit widget is updated at the end of the calculation so as to contain both the input data and the result data. Note that if an Output File name was set (see above), the (x,y) coordinates of the isotopic peaks pattern graph are not displayed in the Results text edit widget. The results for the given example are graphed using `gnuplot` and shown in Figure 7.7.

8

XpertEdit: A Powerful Editor and Simulation Center

After having completed this chapter you will be able to perform sophisticated polymer chemistry simulations on polymer sequences—that can be edited in place—along with automatic mass recalculations.

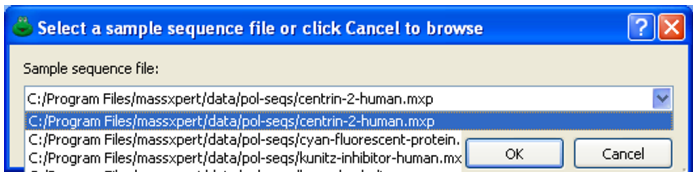


Figure 8.1: **Selection of a sample polymer sequence.** *massXpert* ships with a number of sample polymer sequences which are designed to allow easy demonstration of the *XpertEdit* features. This selection dialog lists all the polymer sequence files that were shipped along with *massXpert*.

XpertEdit Invocation

The *XpertEdit* module is easily called by pulling down the *XpertEdit* menu item from the *massXpert* program's menu. The user may start the *XpertEdit* module by:

* Opening a sample polymer sequence;
* Creating a new polymer sequence;
* Loading a polymer sequence from disk.

XpertEdit Operation: *In Medias Res*

The first manner to start an *XpertEdit* session is by opening a sample sequence out of the list of sequences that were shipped along with *massXpert*. The *XpertEdit*⟶*Open Sample Sequence* menu item opens the dialog box shown in Figure 8.1. The drop-down widget in this dialog window lists all the polymer sequence files that were shipped along with *massXpert*. Simply select one item and click OK. To select another polymer sequence file, click Cancel, which will trigger the system's file selection dialog to open for you to browse to the location where the polymer sequence file is stored. The process is identical to the normal polymer sequence file opening (see below).

The second way to start an *XpertEdit* session is by creating a new polymer sequence (*XpertEdit*⟶*New Sequence* menu). The program immediately asks to select a polymer chemistry definition, as shown in Figure 8.2. The drop-down widget lists all the polymer chemistry definitions currently registered on the system. If the polymer chemistry definition is not listed, clicking onto Cancel will let the user browse the disk in search for a polymer chemistry definition file.[1] Once the polymer chemistry definition has been selected and successfully parsed by the program, the user is presented with an empty sequence editor.

The third way to start an *XpertEdit* session is by opening an existing polymer sequence file. Once the sequence file has been opened, the user is presented with a sequence editor as represented in Figure 8.3. At this point, when the

[1] Note that once the sequence is saved, the polymer chemistry definition file *must* be registered or the sequence file will not be loadable. This is described in a later chapter.

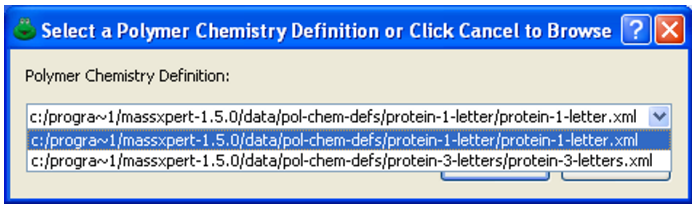


Figure 8.2: **Selection of the polymer chemistry definition.** When creating a new polymer sequence, it is necessary to first indicate of what polymer chemistry definition the polymer sequence will be. This window lists all the polymer chemistry definition currently available on the system.

user starts editing a sequence, the characters entered at the keyboard, or pasted from the clipboard, will be interpreted using the polymer chemistry definition that was selected in the initialization window described above.

Now, of course, editing a polymer sequence is not enough for a mass spectrometric-oriented software suite; what we want is *compute masses!* The mass calculation process is immediately visible on the right hand side of the sequence editor shown in Figure 8.3. The Masses frame box widget contains two items:

* Whole Sequence A frame box widget displaying the Mono and Avg masses of the whole polymer sequence, irrespective of the current selection;
* Selected Sequence A frame box widget displaying the Mono and Avg masses of the currently selected region of the polymer sequence.

The user may change the mass calculation engine configuration at any point in time using the widgets in the Calculation Engine tool box that contains the following configurable parameters:

* Polymer
  - Left Cap If checked, the left cap of the polymer sequence will be taken into account;
  - Right Cap If checked, the right cap of the polymer sequence will be taken into account. Note that if Force is checked also, then the modification is taken into account even when selecting a region of the sequence that does not encompass the left end monomer;
  - Left Modif If checked, the modification of the polymer sequence's left end will be taken into account. Note that if Force is checked also, then the modification is taken into account even when selecting a region of the sequence that does not encompass the right end monomer;
  - Right Modif Same as above, but for the right end modification;
* Selections and regions
  - Multi-region If checked, the sequence editor allows more than one region to be selected at any given time (no limitation on the number of selected regions;

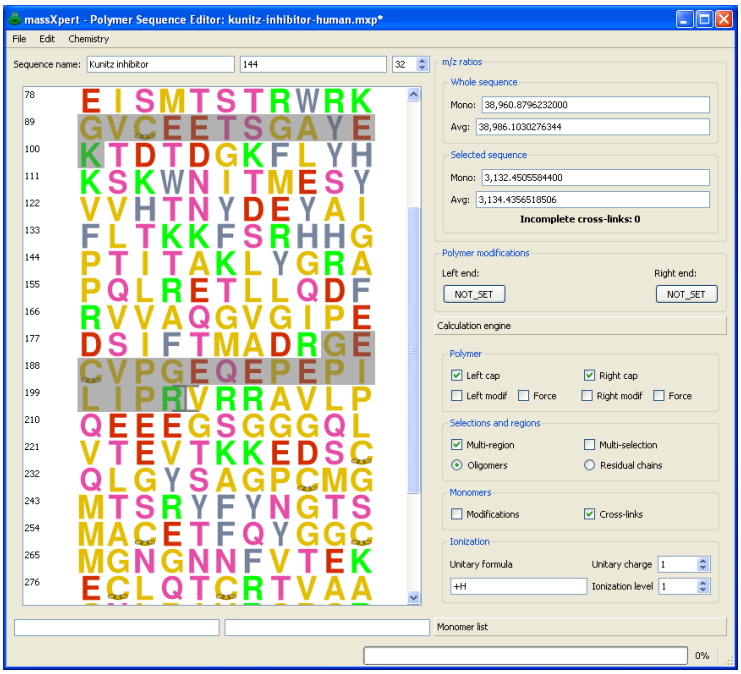


Figure 8.3: **The *XpertEdit* module.** This figure shows a polymer sequence displayed in an *XpertEdit*or window.

- ✦ Multi-selection If checked, the sequence editor allows not only the selection of multiple regions at any given time, but also the selection of totally or partially overlapping regions.
- ✦ Oligomers When multiple regions are selected, each selected region behaves like an oligomer, that is, it gets its left and right end caps added (if the corresponding calculation engine configuration item is activated);
- ✦ Residual chains When multiple regions are selected, the different regions behave like residual chains: the left and end caps are added only once (if the corresponding calculation engine configuration item is activated).

✱ Monomers

- ✦ Modifications If checked, the monomer modifications will be taken into account;
- ✦ Cross-links If checked, the cross-links in the polymer sequence will be taken into account. Note that *only cross-links fully encompassed by the selected sequence region(s)* will be taken into account for the Selected sequence mass calculations. If any number of cross-links are not fully encompassed by the currently selected sequence region, then that number is displayed along with the following label visible in the Selected sequence group box : Incomplete cross-links:.

- Ionization
  - *+H* This formula represents the ionization agent formula (that is, a protonation);
  - Unitary charge *1* Charge brought by the ionization agent. In the example, a protonation brings a positive charge;
  - Ionization level *1* Level of the ionization requested. In the example, a single ionization is requested, that is a monoprotonation.

When any parameter listed above is changed, the recalculation of the masses—for both the Whole sequence and the Selected sequence—is triggered and the new masses are updated in their respective line edit widgets, described earlier. The fact that the user can specify ionization rules should make it clear that the values that are displayed are actually m/z ratios (as long as one ionization is required).

The Editor Window Menu

The menu bar in the polymer sequence editor displays a number of menu items, reviewed below:

- *File* (Figure 8.4)
  - *File*⟶*Close* Closes the sequence;
  - *File*⟶*Save* Saves the sequence. If the sequence has no filename yet, the user is invited to select a filename;
  - *File*⟶*Save As* Save the sequence in a new file;
  - *File*⟶*Import Raw* Opens a text file and tries to import the sequence. If invalid monomer code characters are found, the user is given a chance to revise the imported sequence;
  - *File*⟶*Export to Clipboard* Copies the sequence and all the data (masses and calculation options) to the clipboard, in the form of simple text;
  - *File*⟶*Export to File* Writes to file the sequence and all the data (masses and calculation options) to the clipboard, in the form of simple text (if a filename was already selected, otherwise the user is invited to select a file into which the data are to be written);
  - *File*⟶*Select export file* Invites the user to select a file into which the data are to be written).
- *Edit*
  - *Edit*⟶*Copy* Copies the current selected region(s) (if any) to the clipboard. If there are more than one region currently selection, then the user is informed that the copied sequence will correspond to these two sequences joined together. *Be aware, that the order in which the region sequences are joined is the order in which the regions were selected, and not the order in which the sequences appears in the whole polymer sequence*;

- *Edit⟶Cut* Copies the current selection (if any) to the clipboard and removes it from the sequence. Note that it is not yet possible to cut more than one selected region in one single operation;;
- *Edit⟶Paste* Pastes the sequence from the clipboard into the sequence at point (that is the current cursor location). If the pasted sequence is found to contain characters not valid for the current polymer chemistry definition, the user is given a chance to revise the pasted sequence. If one sequence region was selected, it is replaced with the pasted sequence. If more than one sequence region was selected, the operation cannot be performed and the user is informed;
- *Edit⟶Find Sequence* Finds a sequence motif in the polymer sequence.

* *Chemistry* (Figure 8.5)

  - *Chemistry⟶Modify Monomer(s)* Modify (or unmodify) one or more monomers in the polymer sequence;
  - *Chemistry⟶Modify Polymer* Set (or unset) the left (or right, or both) modification of the polymer sequence;
  - *Chemistry⟶Cross-link Monomers* Set cross-links to monomers of the polymer sequence;
  - *Chemistry⟶Cleave* Perform a chemical/enzymatical cleavage of the polymer sequence;
  - *Chemistry⟶Fragment* Perform the gas phase fragmentation of the currently selected oligomer;
  - *Chemistry⟶Mass Search* For any sequence having a mass matching the searched mass;
  - *Chemistry⟶Compute m/z Ratios* Starting from a given m/z ratio and a given ionization status, calculate a range of m/z ratios with a given ionization agent;
  - *Chemistry⟶Determine Compositions* Calculate the monomeric/element composition of the whole polymer sequence or of the current selection;
  - *Chemistry⟶pKa pH pI* Perform acidity, pH and isoelectric point calculations on the whole sequence or on the current selection.

* *Options*

  - *Options⟶Decimal places* Set the number of decimal places to be used to display the numerical values.

Editing Polymer Sequences

As described earlier, in the chapter about the *XpertDef* module, a polymer chemistry definition may allow more than one character to qualify the codes of

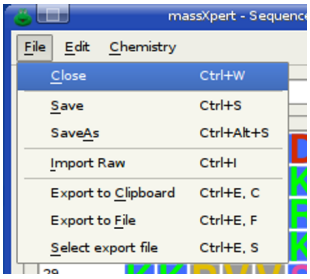


Figure 8.4: **The *XpertEdit* window File menu.** This figure shows the File menu as dropped-down menu in the polymer sequence window.

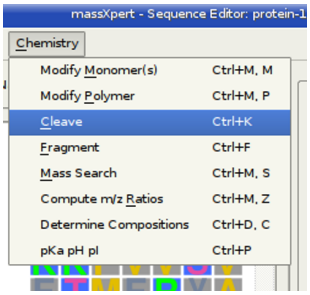


Figure 8.5: **The *XpertEdit* window Chemistry menu.** This figure shows the Chemistry menu as dropped-down menu in the polymer sequence window.

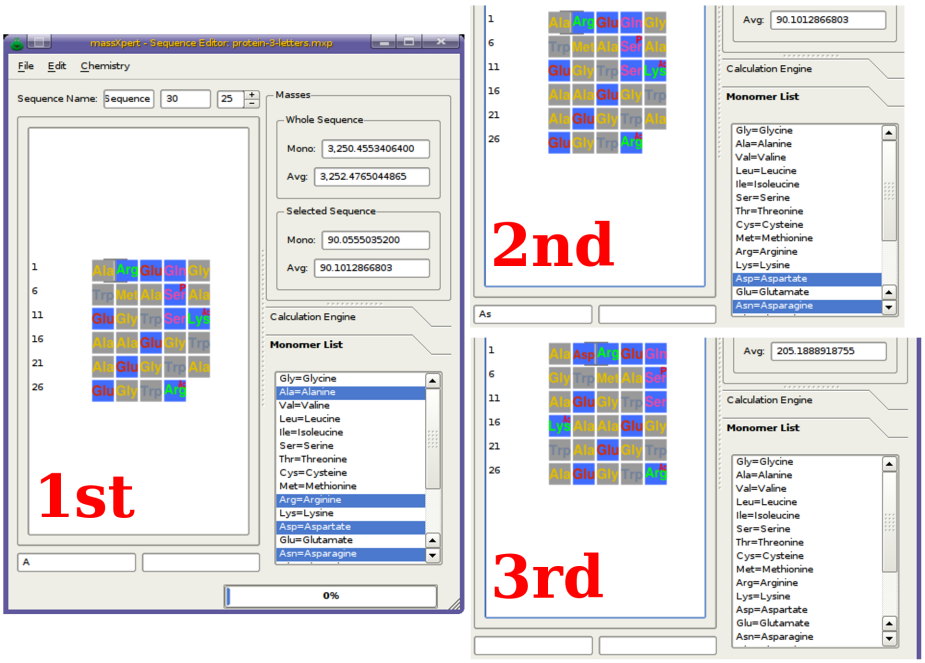


Figure 8.6: **Multi-character code sequence editing in *XpertEdit*.** This figure shows the process by which it is made possible to edit polymer sequences with a monomer code set that allows more than one character per code.

the monomers (see chapter 6, section 6 on page 47). It was noted also that it is not because the number of allowed characters is 3, for example, that all the monomer codes of the polymer chemistry definition must be defined using three characters: 3 is the *maximum* number of characters that may be used.

Multi-Character Monomer Codes

This section deals with the editing of a polymer sequence for which monomer codes can be made of more than one character. Figure 8.6 shows the case of a polymer sequence for which the polymer chemistry definition allows three characters to define monomer codes. The example is based on the following real-world situation: the user wants to edit the sequence by insertion—at the cursor point—of a new "Aspartate" monomer, of which the user knows only that its code starts with an 'A'. The cursor is located after the first "Ala" monomer at position 1 (panel 1st).

After keying-in A (panel 1st), no sequence modification is visible in the sequence editor. Instead, an 'A' character is now displayed in the left line edit widget under the sequence. The reason of this apparently odd behaviour is that the polymer chemistry definition allows up to 3 characters to describe a monomer code. If no monomer vignette is displayed in the polymer sequence, that means that more than one monomer code start with an 'A' character: *XpertEdit* cannot figure out which monomer code was actually meant by the

user when keying-in A.

There is a way, called *code completion*, to know which monomer code(s)—in the current polymer chemistry definition—do start with the keyed-in character(s) (currently, 'A'). The user can always enter the *code completion mode* by hitting the ENTER key. This is what is shown in the panel 1st, right hand side Monomer List listview widget (click on that Monomer List label to show that list if it is not already visible). We see that, in the current polymer chemistry definition, four monomer codes start with an 'A' character, and these are "Ala", "Arg", "Asp" and "Asn" (as highlighted in the code completion monomer list).

Because we now know that the code we are to key-in is "Asp", we key-in a s. The result is shown in panel 2nd. What we see here is that, this time also, nothing changed in the polymer sequence. What changed is that the character string in the left line edit widget below the sequence is now "As". Let's key-in once more the ENTER key. This time, only two items are highlighted: "Asp" and "Asn" in the code completion monomer list (panel 2nd). This is easy to understand: there are only two monomer codes that start with the two letters 'A' and 's' ("As") that we have keyed-in so far. At this time, we key-in a last character: p. At this point, the monomer is effectively inserted in the polymer sequence, as the "Asp" monomer left of the cursor, as shown in panel 3rd.

Unambiguous Single-/Multi-Character Monomer Codes

Let's imagine that we have a polymer chemistry definition that allows up to 3 characters for the definition of monomer codes, but that we have one of these monomer codes (let's say the one for the "Glutamate" monomer) that is one-letter-long: 'E'. This monomer code 'E' is the only one in the polymer chemistry definition to start with an 'E' character. In this case, when we key-in E, we'll observe that the monomer code is immediately validated and that its corresponding monomer vignette is also immediately inserted in the polymer sequence. This is because, *if there is no ambiguity, XpertEdit will immediately validate the code being edited.*

The mechanism described above means that the user is absolutely free to define *only single-character monomer codes* in a polymer chemistry definition; the behaviour of the program is thus to behave exactly as if the multi-character code feature was inexistent in the program: each time a new uppercase letter is keyed-in, it is automatically validated and the corresponding monomer is created in the sequence.

Erroneous Monomer Codes

The typing error detection system triggers immediate alerts whenever the code beign keyed-in is incorrect. This is described in Figure 8.7 on the following page. If the user enters an uppercase character not matching any monomer code currently defined in the polymer chemistry definition, or a lowercase character as the first character of a monomer code, the program immediately complains

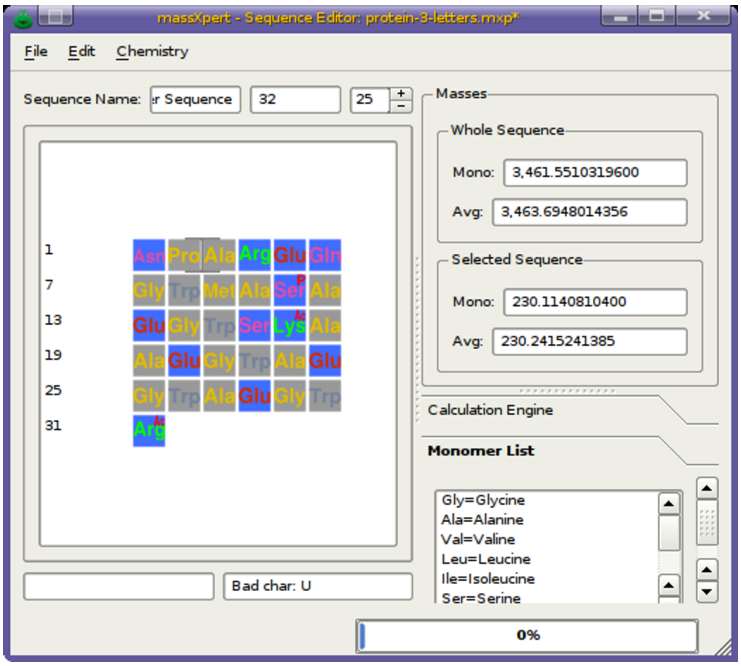


Figure 8.7: **Bad code character in *XpertEdit* sequence editor.** This figure shows the feedback that the user is provided by the code editing engine, when a bad character code is keyed-in.

in the right line edit widget below the sequence. In this case, the monomer code is not put into the left text widget, which means it is simply ignored.

If the user starts keying-in valid monomer character codes, like for example we did earlier with "As", and that she wants to erase these characters because she changed her mind, she *must not* use the BACKSPACE key, because this key will erase the monomer left of the cursor point in the polymer sequence! The way that the user has to remove the characters currently displayed in the left line edit widget below the sequence, is to key-in the Esc key once for each character. For example, let's say you have already keyed-in A and s. In this case the left line edit widget displays these two characters: "As". Now, if the user changes his mind, not willing to enter "Asp" monomer code anymore, but "Gly" instead, all she has to do is to key-in the Esc key once for the 's' character (which disappears) and once more to remove the remaining 'A' character. At this point it is possible to start fresh with the "Gly" monomer code by keying-in sequentially G, l and finally y.

Simplified Editing

When the monomer codes of a given polymer chemistry definition are too numerous or too long to remember, one simplified editing strategy is by using the list of available monomers located on the right side of the sequence editor (wid-

Figure 8.8: **Finding a sequence motif in the polymer sequence.** The first iteration should be performed by clicking onto the Find button, and each following iterations should be performed using the Next button.

get labelled Monomer list). The items in the list are active: if double-clicked, an item will see its corresponding monomer code inserted in the sequence at the current cursor location. This list thus makes it easy to "visually" edit the polymer sequence without having to remember all the codes in the polymer chemistry definition.

Finding sequence motifs

Finding sequence motifs in the polymer sequence is performed by selecting the *Edit*⟶*Find Sequence* menu item. The dialog window is shown in Figure 8.8. When performing the first search in a polymer sequence, the Find button should be used. This will trigger a search starting at the beginning of the polymer sequence. For each successive search, the Next button should be used.

Each searched sequence motif will be stored in a history list that is made available by dropping down the combo box widget where the sequence motif is entered. The Clear history button will erase all the searched sequence motifs from the history, thus resetting it.

Importing Sequences

Very often, the user will make a sequence search on the web and be provided with a polymer sequence that is crippled with non-code characters. That web output might either be saved in a text file for future reference or copied to the clipboard for immediate use in *massXpert*. The two cases are reviewed below.

Importing From The Clipboard

XpertEdit provides a convenient way to spot non-valid characters in a text and to let the user "purify" the imported sequence. A clipboard-imported sequence is systematically parsed. When invalid characters are found, the window depicted in Figure 8.9 on the following page is presented to the user for her to

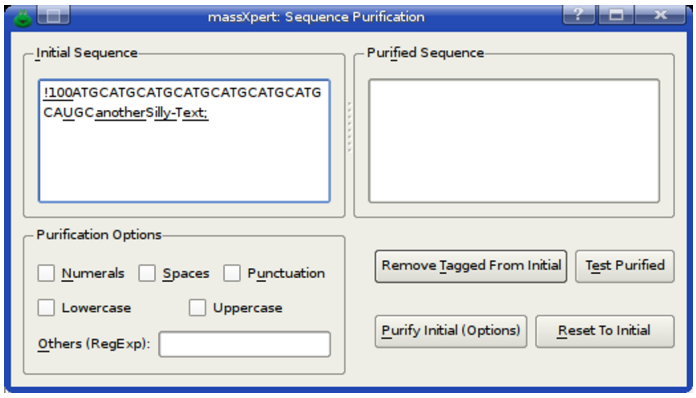


Figure 8.9: **Clipboard-imported sequence error-checking.** If a sequence that is imported through the clipboard to the *XpertEdit* sequence editor contains invalid characters, the user is provided with a facility to "purify" the sequence. This facility is provided to the user through the window depicted in this figure.

make appropriate adjustments (in this example we tried to copy from clipboard the following sequence: "`!100 ATGCATGC ATGCATGC ATGCATGC ATGCAUGC anotherSilly-Text;`").

As soon as a character does not correspond to any valid monomer code, it is tagged, and the sequence is presented to the user in a text edit widget (Initial Sequence) with the all the improper characters tagged by underlining. At that point, if the user clicks the Remove Tagged From Initial button, all the tagged characters will be automatically removed and the purified sequence will show up in the Purified Sequence text edit widget.

Also, the user is provided with automatic "purification" procedures whereby it is possible to remove one or more classes of characters from the imported sequence (Purification Options frame widget). Checking one or more of the Numerals or Spaces or Punctuation or LowerCase or Uppercase checkbuttons, or even entering other user-specified regular expressions in the Other (RegExp) line edit widget, will elicit their removal from the imported sequence after the user clicks the Purify Initial (Options) button.

When the user is confident that almost all the erroneous characters have been removed (Figure 8.10 on the next page), she can click the Test Purified button, which will trigger a "re-reading" of the sequence in the Purified Sequence text edit widget. If erroneous characters are still found, they are tagged.

Note that, for maximum flexibility, the user is allowed an immediate and direct editing of the purified sequence in the Purified Sequence text edit widget (that is, that text edit widget is *not* read-only).

Once the sequence if finally depured from all the invalid characters, the user can select it in the text edit widget and paste it in the *XpertEdit* sequence editor. This time, the paste operation will be error-free. Note that if any sequence portion is currently selected, it will be replaced by the one that is being pasted into the editor.

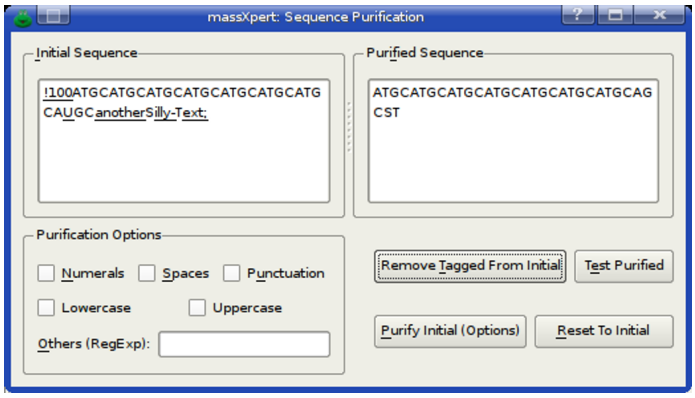


Figure 8.10: **Clipboard-imported sequence purification.** There are a number of ways to purify a sequence. Here the Remove Tagged From Initial button was clicked. The purified sequence shows up in the Purified Sequence text edit widget.

Importing From Raw Text Files

It might be of interest to be able to import a sequence from a raw file. To this end, the user is provided the menu *File⟶Import Raw* that opens up a file selection window from which to choose the file to import. The program then iterates in the lines of that file and checks their contents for validity. If errors are found, then the same process as described earlier for clipboard-imported sequences is started. The user can then purify the sequence imported from the file and finally integrate that sequence in the polymer sequence currently edited. Note that if any sequence portion is currently selected, it will be replaced by the one that is being imported.

Multi-region Selections

massXpert implements a sophisticated multi-region selection model. Two selection modes are available:

- *Multi-region selection mode:* In this mode, it is possible to select more than one region in the polymer sequence. In all cases below, make sure that the Multi-region checkbutton is checked in Selections and regions group box. This is how these selections are performed:
  - *With the mouse:* Left-click and drag to make the first selection. Go with the mouse cursor at the beginning of new selection, hold the Ctrl key down while left-clicking and dragging to perform the second region selection. Continue as may times as necessary;
  - *With the keyboard:* Position the cursor at the beginning of the first region to be selected, hold the Ctrl+Shift keys down while moving

the cursor with the direction keys (←, →, ↑, ↓). Hold the Ctrl key down and use the direction keys to go to the beginning of the new region selection, press the Shift key and hold it down while moving the cursor with the direction keys to actually perform the region selection.

* *Multi-selection region mode:* In this mode (which requires the multi-region selection mode to be enabled), it is possible to perform selections that overlap. For example, one could select the sequence "MAMISGM" and then select the sequence "SGMSGRKAS". The overlapping sequence is thus "SGM".

Being able to select multiple regions and/or to select multiple times the same region involves some configurations, as far as calculating relevant masses is concerned. Indeed, whatever the selection mode that is enabled, each time one selection (overlapping with another or not) is added or removed, masses are recalculated for the current selection.[2] The way the multi-region selections and the multi-selection regions are handled, from the mass calculation standpoint, is configured as follows:

* *Regions are oligomers:* In this configuration, each selection behaves as an oligomer, and thus should normally be capped on both its left and right ends. This is typically the situation when the user wants to simulate the formation of a cross-linked species arising from the cross-linking of two oligomers: each oligomer is capped on both its ends;

* *Regions are residual chains:* In this configuration, each selection behaves as a residual chain, and thus the oligomer resulting from the multi-region selections is capped on its left and right ends only once. This situation is typically encountered when simulating partial cleavages by first selecting an oligomer, checking its mass and then continuing selection to simulate a longer oligomer resulting from a partial cleavage. Also, the situation might be encountered when there are multiple repeated sequence motifs in a polymer sequence and mass data are difficult to analyze.

Polymer Sequence Modification

It very much often happens that the (bio) chemist uses chemical reactions to modify the polymer sequence she is working on. Mass spectrometry is then often used to check if the reaction proceeded properly or not. Further, in nature, chemical modifications of biopolymer sequences are very often encountered. For example, protein sequences get often modified as a means to regulate their function (phophorylations, for example, or acetylations, methylations...). Nucleic acid sequences are very often and extensively modified with modifications such as methylation...

[2] "Selection", here, is thus used to collectively represent all multi-region selections and multi-selection regions at any given time in the polymer sequence editor.

It is thus crucial that *massXpert* be able to model with high precision and flexibility the various chemical reactions that can be either made in the chemistry lab or found in nature. The *massXpert* program provides two different chemical modification processes:

* A process by which monomers belonging to the polymer sequence can be individually modified;

* A process by which the whole polymer sequence can be modified, either on its left end or on its right end or even on both ends.

Selected Monomer(s) Modification

There are a number of manners in which monomers can be modified in a polymer sequence. Figure 8.11 on the following page shows the simplest manner: the user first selects the monomer vignette to be modified and calls the *Chemistry⟶Modify Monomer(s)* menu. A window shows up where all the modifications currently available in the polymer chemistry definition are listed. Because a monomer vignette was initially selected in the editor window, the Selected Monomer target radiobutton is on by default.[3] It is then simply a matter of choosing the right modification from the Available modifications list and clicking onto the Modify button. The target(s) of a given modification (as selected in the Target frame widget) can be identified according to:

* The Selected Monomer frame will display data in its two line edit widgets if a single monomer vignette was selected at the time the monomer modification action was invoked (exactly as in Figure 8.11 on the next page). Only the monomer of which the code and the position are displayed will be modified (even if it is no more selected or if the sequence has changed and the monomer at the displayed position is not the same anymore);

* The Current Selection radiobutton widget indicates that the modification should be performed on all the monomers that are *currently* selected, that is, if the selection changed after the modification window was displayed, the new selection is modified, not the old one;

* The Monomers Of Same Code If a monomer code is displayed in the Selected Monomer frame, all the monomers in the sequence that have that code are modified;

* Monomers From The List All the monomers in the polymer sequence having a code corresponding to any code selected in the Available Monomers list are modified;

* All Monomers All the monomers of the polymer sequence are modified;

[3] Note that if a sequence was selected when the monomer modification task was started, then selecting Current selection would be required to modify all the monomers in the selection. Alternatively, if this is not what is required, re-selecting the right monomer in the sequence and selecting Current selection will ensure the modification applies only on the currently selected monomer.

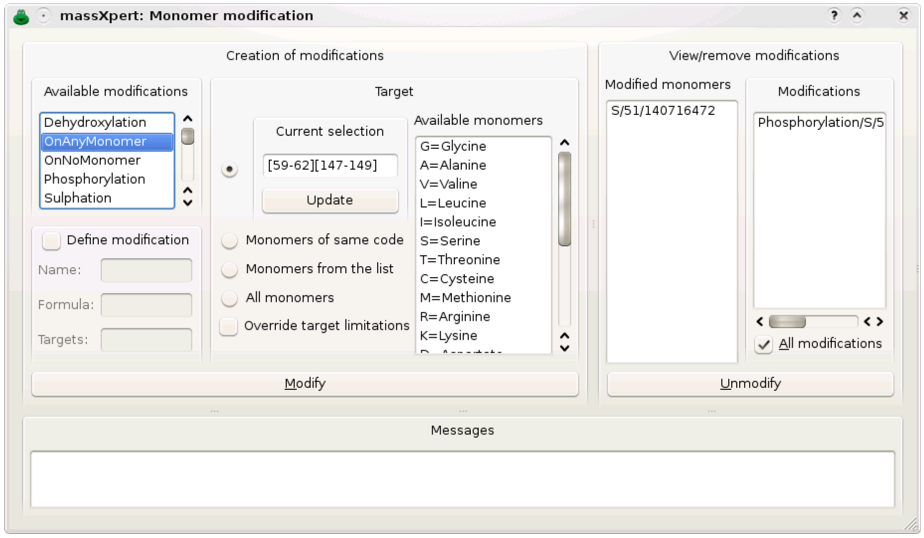


Figure 8.11: **Modification of a monomer in a polymer sequence.** This figure shows how the chemical modification of monomer(s) can be performed.

Note that there is one checkbox widget (Override target limitations) that requires explanation. In the chapter about the definition of polymer chemistries (chapter 6 on page 43) the definition of modifications was detailed, and the target notion was explicited. If, during a monomer modification, *massXpert* detects that the user is trying to modify a monomer that is not a target of the modification at hand, it will complain, as shown in the Messages text edit widget of Figure 8.11). In this example, indeed, the user tried to modify monomer *Isoleucine* with *Phosphorylation*, which is not possible because modification *Phosphorylation* has been defined a not having monomer *Isoleucine* as any of its targets. Another situation where target limitations might show up, is when trying to modify a monomer more than authorized by the Max. count number of times that monomer might be modified at once with that modification. For example, when working of methylation of proteins, it might happen that lysyl residues get methylated more than one at a time (tri-methylation occurs often in histones). If the chemical modification was defined in *XpertDef* with a max count of 2 and a third chemical modification is asked on a given target monomer, then the program refuses to perform the modification. To override this limitation, check the Override target limitations checkbox widget.

The general concept about this is : the Override target limitations checkbox widget is unchecked by default so that the user does not do mistakes without knowing. However, flexibility is desirable, and the Override target limitations checkbox widget can be checked if required.

As a result of the monomer modification, the monomer vignette gets modified. Figure 8.11 shows one phosphorylated Seryl residue at position 8: a transparent graphics object (a red 'P') was overlaid onto the corresponding seryl monomer vignette. If the user modifies a monomer with a modification that has no corresponding *svg* file defined for its graphical rendering in file `modifica-`

Figure 8.12: **Rendering of a monomer modification in a polymer sequence.** This figure shows how the chemical modification of monomer(s) is graphically rendered. The 'K' residue is modified using an "Acetylation" modification. The 'S' residue is modified with a modification that has no associated graphical vignette. The default vignette is thus used.

`tion_dictionary`, then a default modification rendering is used.

The user is responsible for correctly reading the messages that might be published in the Messages text edit widget. It is important to understand that, when a monomer is modified, its previous modification (if any) is overwritten with the new one. The user is invited to experiment a bit with the monomer modification process, so as to be confident of the results that she is going to obtain when real polymer chemistry work is to be modelled in *massXpert*.

If the modification to be applied is not readily available in the list of modifications defined in the polymer chemistry definition, then it is possible, by checking the Define modification check button widget to manually define a modification. This procedure leads to the modification of the target monomer(s) exactly as if the modification had been selected from the list of available modifications. But, because the modification has a name not known to the polymer chemistry definition, the editor cannot modify the monomer vignette with a predefined transparent raster image. Thus, as seen on Figure 8.12, the modified residue gets visually modified using the default transparent raster image (4 interrogation marks, one at each corner of the monomer vignette square).

It is perfectly feasible to modify a single monomer more than once (with the same modification or not ; for example a tri-methylation with a methylation modification). This is why when the window depicted in Figure 8.11 shows up, the two lists at the right hand side show the monomers currently modified and the modification(s) that are currently set to these modified monomers. Selecting one item from the Modified monomers list will show only the modifications set to that monomer in the Modifications list. If all the modifications in the polymer sequence are to be displayed then, checking the All modifications check box widget will trigger the display of all the modifications set to any monomer in the whole polymer sequence.

Unmodification of monomers is easily performed by selecting any number of items from the Modifications list and clicking the Unmodify button.

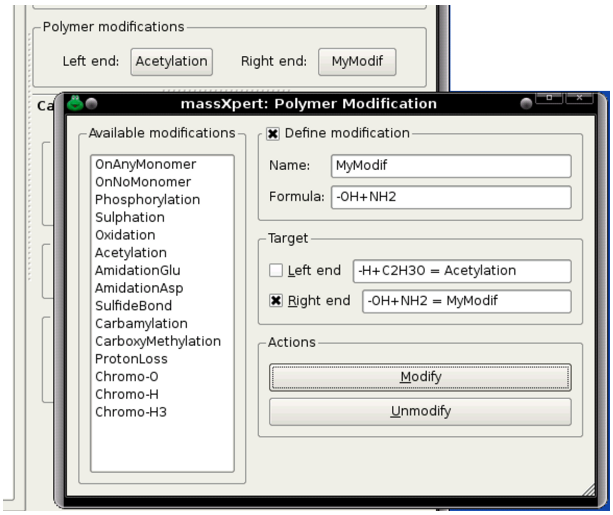


Figure 8.13: **Modification of the left end of a polymer sequence.** This figure shows how simple it is to permanently modify a polymer sequence on either or both its left/right ends.

It should be noted that once a monomer modification dialog window has been opened, the polymer sequence should not be edited. This is because the modification/unmodification process takes for granted that the polymer sequence still is identical to what it was when the monomer modification dialog was opened. Mecanisms are there to ensure that the irreparable does not happen, but this warning is in order.

Whole Sequence Modification

As described above, it is possible to modify any monomer in the polymer sequence; whhen any modified monomer is removed, the modification associated to it disappears also. The modifications that we describe here are not of this kind. They can be applied to either the left end of the polymer sequence or its right end (or both ends at any given time). But these modifications do belong to the polymer sequence *per se* and are not removed from it—even if the polymer sequence is edited by removing the left end monomer or the right end monomer. This is why these modifications are *polymer modifications* and not monomer modifications.

The way in which a polymer sequence is modified using *polymer modifications* is much easier than the previous *monomer modifications* case. The modification window is opened by choosing the *Chemistry*⟶*Modify Polymer* menu. The Figure 8.13 shows that window. The modification is absolutely easy to perform, with a clear feedback provided to the user (by listing the permanent modifications in two line edit widgets located in front of the Target checkbuttons Left End and Right End.

Note that, as a convenience for the user, it is possible to modify the polymer sequence using an arbitrary modification in the form of a combination of a name

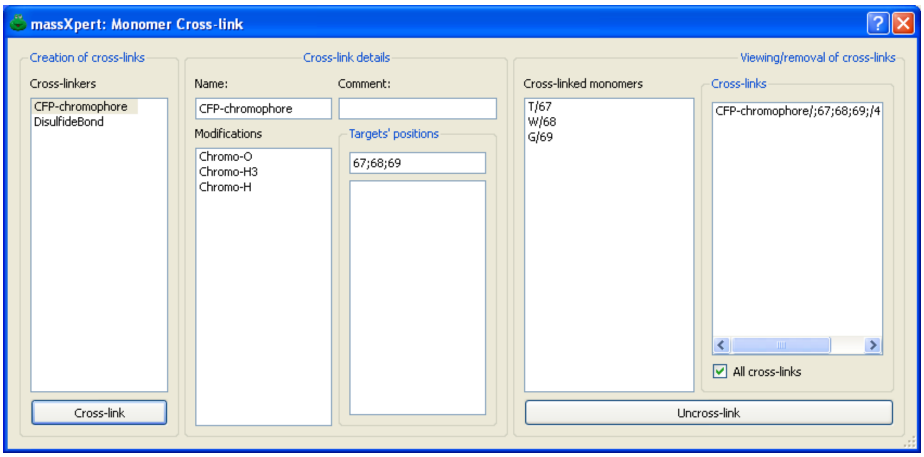


Figure 8.14: **Cross-linking of monomers.** This figure shows the window in which monomers can be cross-linked together. A cross-link (as defined in the current polymer chemistry definition) is selected and the targets are specified in the Targets' positions text line edit widget in the form of monomer positions separated by ';' semicolumns.

and a formula (check the Define modification checkbox, to that effect). The modification object used is created on-the-fly by the program and gets saved in the file as if the user had selected a modification out of the list of available modifications. In the example (Figure 8.13 on the facing page), the polymer sequence was modified on its left end using the "Acetylation" modification available in the polymer chemistry definition and was amidated (formula *-OH+NH2*) with a manually-defined modification called *MyModif*. The polymer sequence editor window displays the left end and right end modifications as labels of buttons located in the Polymer modifications groupbox.

Monomer Cross-linking

A cross-link is a covalent bond that links a monomer with one or more other monomer. A monomer might be cross-linked more than once. The dialog window in which the user might define cross-links is shown in Figure 8.14.

Cross-linkers were defined in the section about *XpertDef* (see page 49). A cross-linker might either define no modification to be applied to the cross-linked monomers or the same number of modifications as there are monomers cross-linked. For example, fluorescent proteins have a chromophore that is made by reaction of three residues (Threonyl [or Seryl]–Tryptophanyl [or Tyrosinyl or Phenylalanyl]–Glycyl), as shown in Figure 8.15. When cross-linking with the fluorescent protein cross-linker, there must be three monomers involved as these are three modifications defined in the cross-linker.

When any monomer involved in a cross-linker is edited off a polymer sequence, the cross-link(s) it was involved in are automatically dissolved and destroyed. Destruction of a cross-link might be performed by selecting the cross-

Figure 8.15: **Graphical rendering of cross-linked monomers.** This figure shows the three monomers (TWG) from cyan fluorescent protein cross-linked together.

link in the Cross-links list widget at the right hand side of the dialog window depicted in Figure 8.14 and by clicking the Uncross-link button.

Sequence Cleavage

It happens very often that polymer sequences get cleaved in a sequence-specific manner. These specific cleavages do occur very often in nature, and are made by enzymes that do cleave biopolymer sequences, like the glycosidases (cleaving saccharides), the proteases (cleaving proteins) or the nucleases (cleaving nucleic acids). But the scientist also uses purified enzymes or chemicals to perform such cleavages in the test tube. *massXpert* must be able to perform those cleavages *in silico*.

It is a matter of having a polymer sequence opened in an editor window and selecting the *Chemistry*⟶*Cleave* menu. The user is provided with a window where a number of cleavage specifications are listed (Figure 8.16, page 87). These cleavage specifications are listed by looking into the polymer chemistry definition corresponding to the polymer sequence to be cleaved. The program knows, for example, that the polymer sequence to be cleaved is of the "protein-1-letter" chemistry type, and thus will list all the cleavage specifications that were defined in that polymer chemistry definition.

The user selects the cleavage specification of interest and also sets the number of partial cleavages that the cleaving agent may yield. In our example, 0 was entered, which means that the cleavage reaction will yield the set of oligomers corresponding to a total cleavage (no missed cleavages=partial cleavages 0). Also the user might indicate that the oligomers computed during the cleavage should be ionized according to the current ionization rule (displayed in the main window) and in the specified range. The calculation process is extremely rapid, so the user may enter rather high values here.

The user might want to generate oligomers for different kinds of cleavages. For example, it might be interesting to have in the same tree view widget the oligomers generated using first trypsin and then cyanogen bromide. In order to add new oligomers to pre-existing one, it is simply required to check the Stack oligomers check button widget prior to clicking the Cleave button again with the new cleavage settings.

The Details frame widget at the bottom of the window displays a number of

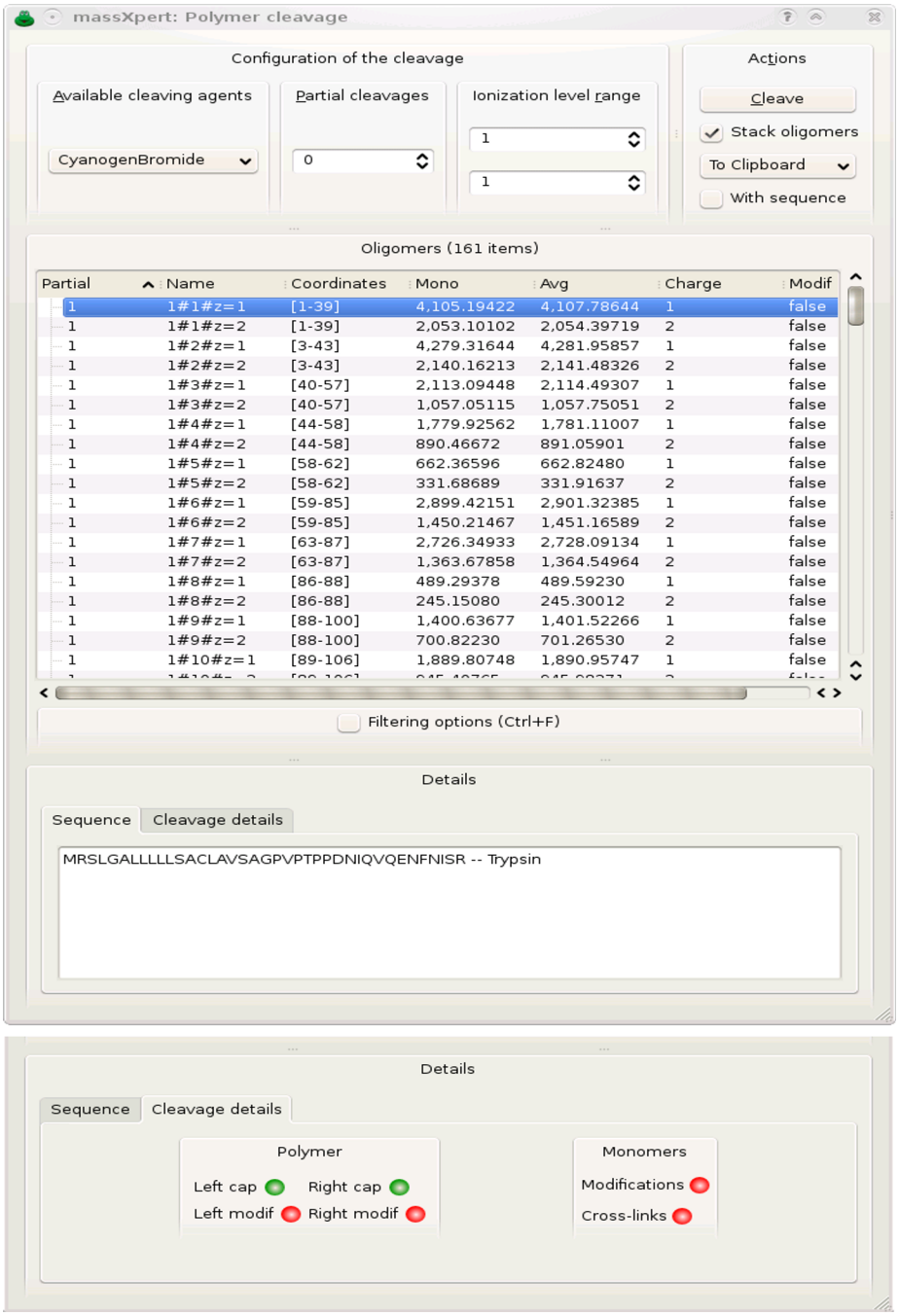


Figure 8.16: **Polymer sequence cleavage window.** This figure shows the window in which polymer sequence cleavage is performed. One cleavage specification is selected and the number of allowed partial cleavages is set. The results are displayed in the same window. It is possible to stack oligomers from different cleavage simulation in the same window.

informative data. In particular, the Sequence tab widget displays the sequence of the oligomer currently selected in the Oligomers treeview widget along with the name of the cleavage agent which it arose from. The Cleavage Details tab widget displays the mass calculation engine configuration at the time the *last*Âăcleavage was performed (one red led means that the related feature was off, conversely a green led means that the feature was on). In our example, the mass calculation for the oligomers did not account for the monomer modifications nor for the left/right ends of the polymer, nor for the cross-links.

When the user triggers a cleavage, the mass calculation engine configuration currently set in the sequence editor is used for the calculation of the mass of the oligomers obtained *per* the cleavage. This process allows an easy change in the mass calculation engine configuration between one cleavage and another so as to allow comparison of masses obtained for the same cleavage but with different mass calculation engine configurations.

Finally, one last note: if the list of monoisotopic or average masses are desired in the form of a text list, right-clicking onto the treeview widget will allow copying to the clipboard either the monoisotopic or the average masses. Also, it is possible to either export the data to the clipboard or to a file or even to drag the displayed oligomer items in a text editor. Only the selected items in the tree view widget will be exported.

For oligomer data filtering, please refer to section 8, page 90.

Oligomer Fragmentation

It happens very often that polymer sequences need to be fragmented in the gas phase (in the mass spectrometer) so that structure characterizations may be performed. For protein chemistry, this happens very often in order to get sequence information for a given peptide ion selected in the gas phase. *massXpert* must be able to perform those fragmentations *in silico*. Let's see how an oligomer can be fragmented using *massXpert*.

It is a matter of having a polymer sequence opened in an editor window and selecting the sequence region to be fragmented. Once this is done, the user selects the *Chemistry*⟶*Fragment* menu. The user is provided with a window where a number of fragmentation specifications are listed (Figure 8.17 on the next page). As detailed for the cleavage of polymers, these fragmentation specifications are listed by looking into the polymer chemistry definition corresponding to the polymer sequence of which an oligomer is to be fragmented.

The user selects the fragmentation specification(s) of interest, set the ionization range required for the generated fragment oligomers (the same as for polymer cleave) and clicks the Fragment button. Upon successful termination of the fragmentation reaction, the generated fragments are displayed in the Oligomers treeview widget.

As detailed for the cleavage of polymer sequences, the Details frame widget displays data about the fragments generated and the way masses were calculated for them.

Finally, one last note: if the list of monoisotopic or average masses are desired in the form of a text list, right-clicking onto the treeview widget will allow copying to the clipboard either the monoisotopic or the average masses.

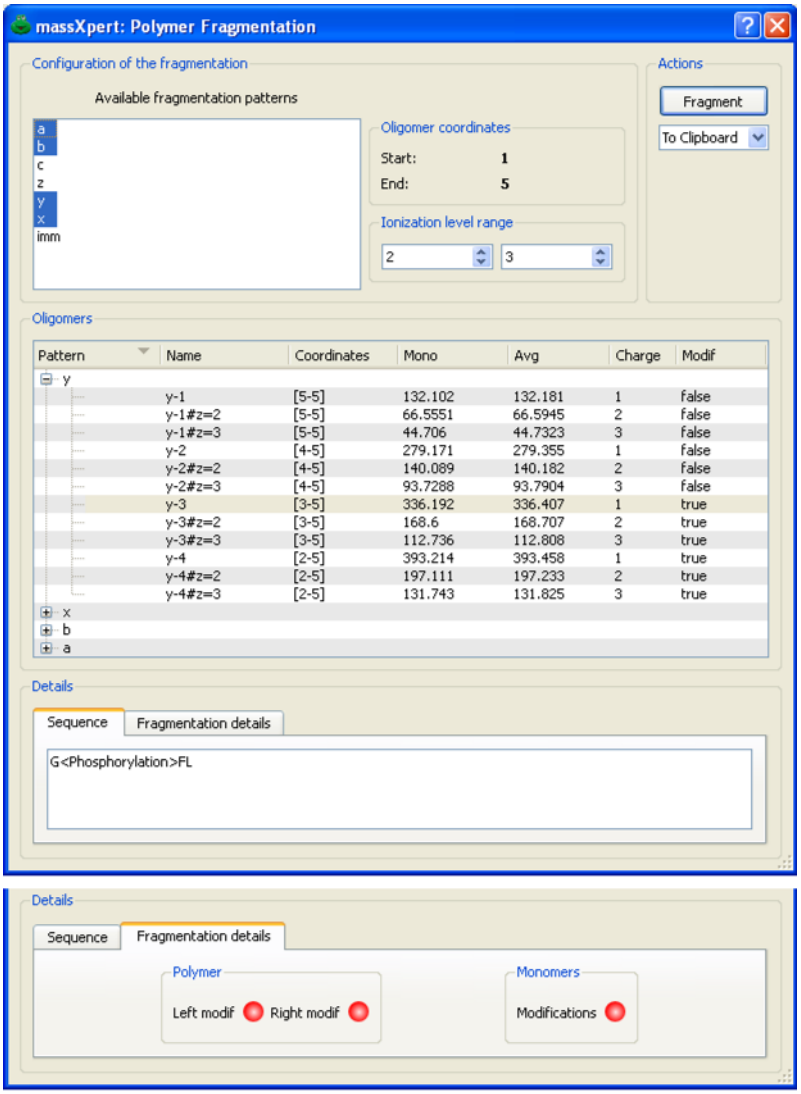


Figure 8.17: **Oligomer fragmentation window.** This figure shows the window in which oligomer fragmentation is performed. One or more fragmentation patterns might be selected in one fragmentation step.

Also, it is possible to either export the data to the clipboard or to a file or even to drag the displayed oligomer items in a text editor.

For oligomer data filtering, please refer to section 8, page 90.

Mass Searching

It may happen that the scientist needs to know if some arbitrary sequence region would have a given mass. *massXpert* allows for mass searching operations in the polymer sequence. This is done by using the menu *Chemistry*⟶*Mass Search*. The window illustrated in Figure 8.18 on the facing page shows up and the user enters masses to search for. A number of parameters are to be detailed:

* Targets The masses should be searched for in the whole sequence or in the currently selection region?

* Ionization When calculating masses for the potential oligomers matching the searched mass, should different levels of ionization be calculated. For example, one find in an electrospray ionization experiment mass spectrum a peak at m/z 1245. It is not possible to know the ionization level for that ion. On could imagine that this value is for a monopronotonated or for a multiprotonated species. If we wanted to asses this, we might ask that the mass be searched for by computing a range of possible ionization levels between Start *1* and End *4* (admitting that for that experiment this is what one would expect).

Once the masses have been searched for, if results are found they are displayed in the same window in the Oligomers treeview widgets (the left one for the mono masses and the right one for the avg masses).

Finally, one last note: if the list of monoisotopic or average masses are desired in the form of a text list, right-clicking onto the treeview widget will allow copying to the clipboard either the monoisotopic or the average masses. Also, it is possible to either export the data to the clipboard or to a file or even to drag the displayed oligomer items in a text editor.

For oligomer data filtering, please refer to section 8, page 90.

Oligomer Data Filtering

Oligomer-generating simulations, like polymer sequence cleavages or fragmentations or mass searches, produce a very large amount of data. It is often desirable to be able to filter quickly some specific data out of these bunch of data...

In all three simulations mentioned above, the results that are displayed in the corresponding dialog windows are easily filtered using the mechanism illustrated in Figure 8.19.

Filtering on the data is easily performed by entering the options in the Filtering options group box (Figure 8.19, page 92). For any filtering operation, only one criterium can be used, that is, for example, filtering can occur only on the basis of the monoisotopic mass or of the average mass, but not on both masses. For example, if one wanted to filter a huge set of data against a specific

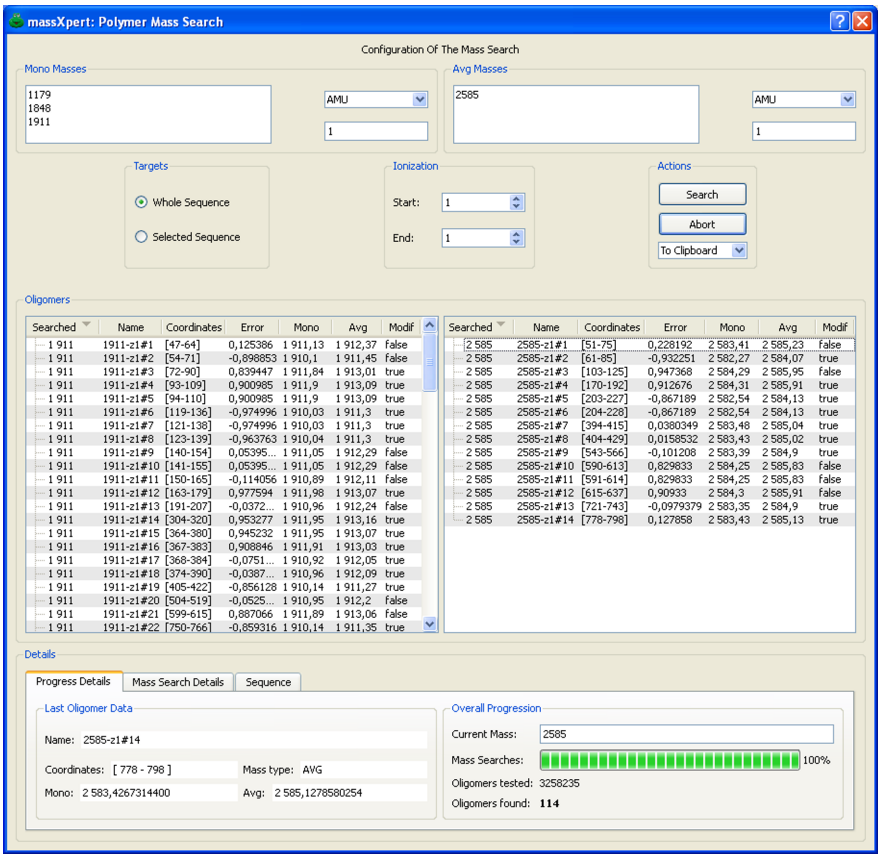


Figure 8.18: **Searching masses in a polymer sequence.** This figure shows the window in which to search for masses in a polymer sequence.

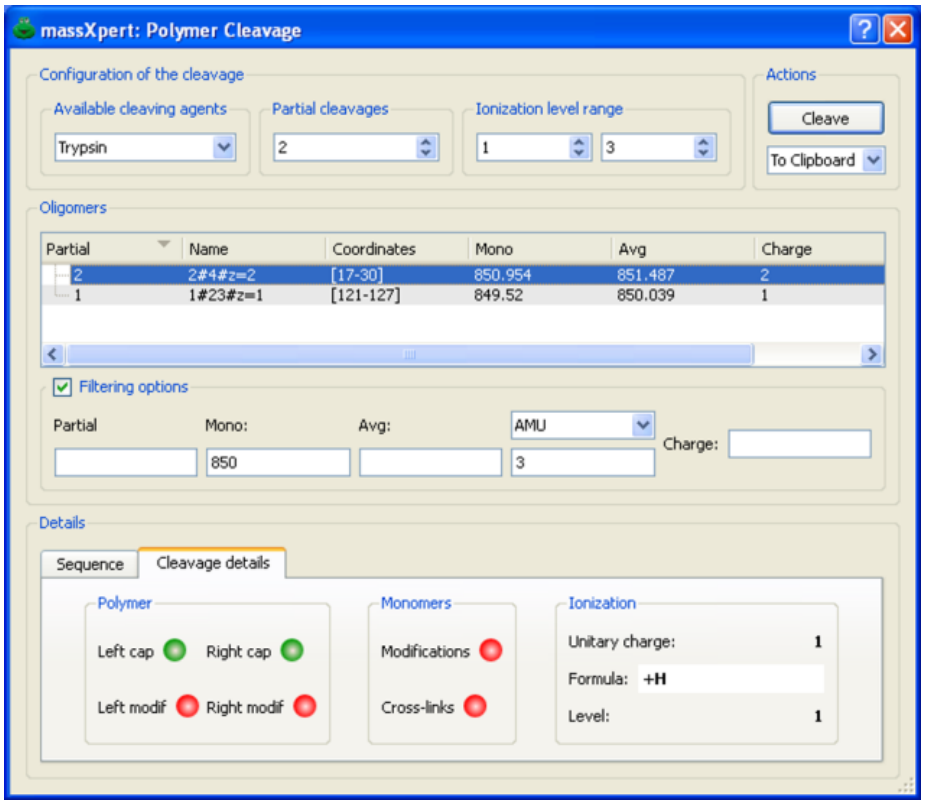


Figure 8.19: **Oligomer data filtering.** This figure shows how oligomer data can be filtered. The Filtering options group box contains four line edit widgets where filtering might be triggered: Partial, Mono, Avg, Charge. The filtered data are displayed in the same window (this examlple for polymer sequence-cleavage oligomer data.

monoisotopic mass of 850 plus or minus 3 atomic mass units, it would simply be a matter of setting the monoisotopic mass to be *850* with a tolerance of *3 AMU* in the corresponding line edit widgets contained in the Filtering options group box. To perform that filtering action, first set the tolerance value (*3*) in its line edit widget and next set the monoisotopic mass value to be *850* in the corresponding line edit widget. While the cursor *is still* in the Mono line edit where *850* was entered, press the keyboard key combination Ctrl+ENTER. The filtering will be immediate and the treeview will show the data that passed the filter. Note that the combo box widget holding the unit of the tolerance (in the example, that unit is AMU, that is "atomic mass unit") and the line edit widget where the tolerance value proper is set (*3* in the example) do not trigger any filtering by themselves; these widgets are only useful in conjunction with other oligomer data : Mono, Avg, Error line edit widgets (depending on the dialog window the filtering occurs: cleavage, fragmentation or mass search). In our example, thus, the filtering would be spoken like this: —*"Only show the oligomers for which the monoisotopic mass is 850 plus or minus 3 atomic mass units"*.

To exit the data filtering mode, simply uncheck the Filtering options check box, and all the initial data will be displayed, irrespective of any data in the line edit boxes described above.

m/z Ratio Calculation

In electrospray ionization, a given polymer sequence might be charged a large number of times. The tool shown in Figure 8.20 on the next page shows how to compute a range of m/z ratios starting from one m/z value for a given charge and a given ionization agent. It is also possible to switch ionization agent on-the-fly.

Monomeric And Elemental Compositions

The *Chemistry⟶Determine Compositions* menu triggers the window shown in Figure 8.21. The elemental composition is determined using the calculations engine configuration currently set in the polymer sequence editor window.

pKa, pH, pI and Charges

When preparing biochemical experiments, very often users need to know how many charges a given polymer sequence will bear at any given pH. Equally important is the ability to know at which pH value the polymer sequence will have a net charge near to zero. The pH value for which a given polymer sequence has a net charge near to zero (typically this means that the number of positive charges equals the number of negative charges) is called the isoelectric point—the pI.

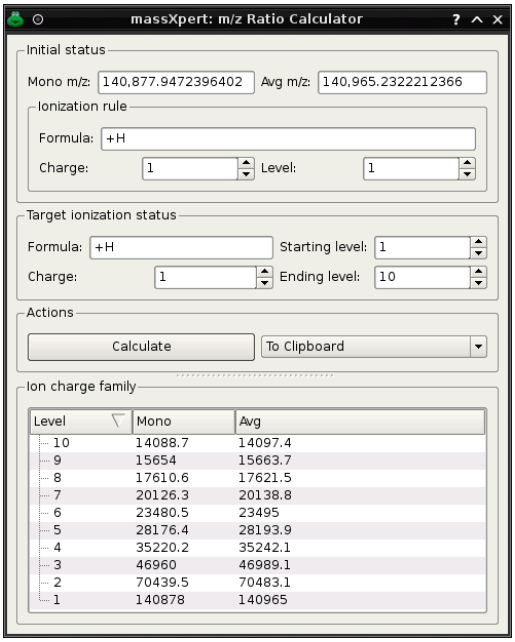


Figure 8.20: **Calculation of ranges of m/z ratios.** This figure shows the window in which to perform the calculation of different m/z ratios starting from one m/z value with a given ionization agent.

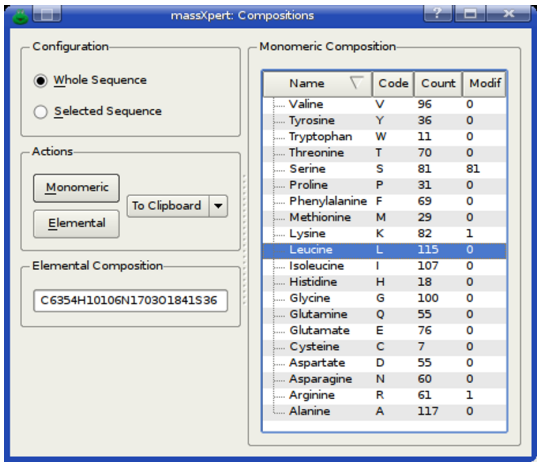


Figure 8.21: **Determination of the compositions.** This figure shows how to determine the monomeric and elemental compositions for the whole sequence or the current selection.

Such computations are pretty computer-intensive and require a very precise knowledge of the chemical structure of the different monomers that take part in the definition of the polymer chemistry. A file, called `pka_ph_pi.xml` is located in the polymer chemistry definition directory. This file lists all the chemical groups that are possibly charged; each monomer of the polymer definition is represented by a `<monomer>` element in which data are defined for any chemical group of that monomer that might bear a charge at any given pH. You can find the listing of the `pka_ph_pi.xml` file in chapter 11 on page 119. We'll discuss any aspect of this file's contents in the next sections with enough detail that the user will be able to write one such file for her specific polymer chemistry.

At the moment, two entities in the polymer chemistry definition might have chemical groups bearing charges: monomers and modifications. We will first review monomers, and modifications next.

Ionized Group(s) In Monomers

Monomers are the building blocks of polymer sequences. These blocks must have at least two reactive groups so that they can be polymerized into a polymer sequence thread. Reactive groups are often chargeable groups; for example, the amino group of amino-acids is such that it gets protonated (positively charged) at a pH inferior to its pKa. Similarly, the carboxylic acid group of amino-acids is deprotonated (negatively charged) at physiological pH.

Some Theory First

For the non-biochemist reader, amino-acids involved in the formation of proteins have always at least two chemical groups that are of inverted electrical charge, at physiological pH values (see Figure 8.22):

* The amino group (called αNH_2) has a typical pKa value of 9.6. This means that, at physiological pH values (between 6.5 and 7.5), the amino group will find the environment rather acidic, and will thus be protonated, leading to a positively-charged species (αNH_3^+);

* The carboxylic group (called $\alpha COOH$) has a typical pKa value of 2.35. This means that, at physiological pH values, the carboxylic group will be in a rather basic environment, and will thus be deprotonated, leading to a negatively-charged species (αCOO^-).

It should be clear that, at physiological pH values the two α chemical groups have a net charge of 0. But proteins are charged, and this is because some of the twenty common amino-acids have other chemical groups beyond the two others already described. Indeed, some amino-acids have lateral chains that bear groups that might be charged depending on the pH: seryl residues have an alcohol group that has a pKa of 13, for example; that means that it is almost always uncharged (form ROH at physiological pH values). The lateral chain of lysine has a pKa of 10.53, which means that at pH values below this pKa value, the ϵNH_2 gets protonated, introducing a positive charge in the protein. Similarly, amino-acids glutamate and aspartate do have a lateral chain ended

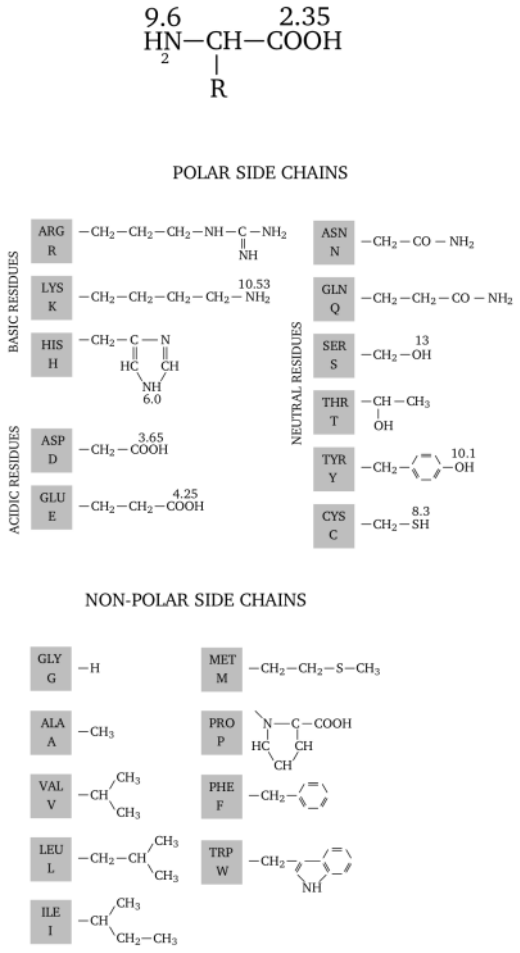


Figure 8.22: **Different pKa values for a number of amino-acids' chemical groups.** All of the twenty amino-acids are represented here, which each amino-acid's lateral chain fully represented. Above each chemical group—for which the value makes sense from a biological perspective—the pKa value is indicated.

with a γCOOH and a βCOOH, respectively. Their pKa values are below 4.5, and thus the groups are negatively charged a physiological pH values.

When the net charge of a polymer sequence has to be computed for a given pH condition, the program iterates in the sequence, and for each monomer will check which one of its chemical group(s) is possibly charged. For this to happen, it is required that a number of data be known for each monomer's chemical group that might play a role in the determination of the polymer sequence's electrical charge. Thus, for each chemical group a number of data should be listed in the `pka_ph_pi.xml` file (please, see that file in the chapter 11 on page 119):

- the chemical group's `<name>` element is required. Examples: "αNH_2" or "ϵNH_2" or "αCOOH";
- the chemical group's `<pka>` element is optional, but is the basis for the charge calculation. Examples: 9.6 for the "αNH_2" or 2.35 for "αCOOH";
- the `<acidcharged>` element is required if the <pka> element is given. This element is responsible for telling if the chemical group is charged (positively) when the pH is lower than pKa (that is when the medium is acidic with respect to the pKa). Examples: an amine is positively charged when it is in its acidic form (protonated); a carboxylic acid is *not* charged when it is in its acidic form;
- there can be none, one or more `<polrule>` element(s) for each chemgroup. The `<polrule>` element gives informations about the way the chemical group at hand might be "trapped" (or not) in the formation of inter-monomer bonds (while the monomer is polymerized into the polymer sequence). The value "left_trapped" means that the chemical group ceases to be involved in charge calculations as soon as it has a monomer at its left end. The value "right_trapped" means the same as above, but when a monomer is polymerized at its right end. For a chemical group that is "left_trapped", we understand that it is only effectively evaluated if it is at the left end of the polymer sequence, since in this case it does not have a monomer at its left side. Conversely, a chemical group that has a `<polrule>` element with value "right_trapped", will be evaluated only if the monomer is actually the right end monomer in the polymer sequence. Finally, the typical lateral chains of amino-acids have a `<polrule>` element with a value "never_trapped", as these chemical groups do not take part in the formation of the inter-monomer bond;
- there can be none, one or more `<chemgrouprule>` element(s) for each chemgroup. A chemgrouprule element should contain the following:
  - there must be an `<entity>` element that indicates what is the chemical entity being dealt with in the current chemgroup element. Valid values for this element are "LE_PLM_MODIF", "RE_PLM_MODIF" or "MNM_MODIF";
  - there must be a `<name>` element naming the chemical entity properly;
  - there must be an `<outcome>` element telling what action should be taken when encountering the `<entity>` on the chemgroup. Valid values are either "LOST" or "PRESERVED".

Understanding By Example

Let us take some examples in order to make sure we actually understand the process of describing how an electrical net charge is calculated for a given polymer sequence and at any given pH value.

Let us see the example of the aspartate amino-acid, of which the lateral chain is nothing but CH_2COOH:

```
<monomer>
  <code>D</code>
  <mnmchemgroup>
    <name>N-term NH2</name>
    <pka>9.6</pka>
    <acidcharged>TRUE</acidcharged>
    <polrule>left_trapped</polrule>
    <chemgrouprule>
      <entity>LE_PLM_MODIF</entity>
      <name>Acetylation</name>
      <outcome>LOST</outcome>
    </chemgrouprule>
  </mnmchemgroup>
  <mnmchemgroup>
    <name>C-term COOH</name>
    <pka>2.36</pka>
    <acidcharged>FALSE</acidcharged>
    <polrule>right_trapped</polrule>
  </mnmchemgroup>
  <mnmchemgroup>
    <name>Lateral COOH</name>
    <pka>3.65</pka>
    <acidcharged>FALSE</acidcharged>
    <polrule>never_trapped</polrule>
    <chemgrouprule>
      <entity>MONOMER_MODIF</entity>
      <name>AmidationAsp</name>
      <outcome>LOST</outcome>
    </chemgrouprule>
  </mnmchemgroup>
</monomer>
```

We see that the code of the monomer for which acid-basic data are being defined is 'D' and that this monomer has three chemical groups that might bring electrical charges. These chemical groups are described by three `<mnmchemgroup>` elements that we will review in detail below (see Figure 8.22 on page 96).

The first `<mnmchemgroup>` element is related to the αNH_2 amino group of the amino-acid:

* `<name>N-term NH2</name>` The name of the chemical group is not immediately useful, but will be used when reports are to be prepared for the calculation;

- `<pka>9.6</pka>` This element is optional. However, of course, if the chemical group might be electrically charged, the pKa value will be essential in order to compute the charge that is brought by this chemical group at any given pH;

- `<acidcharged>TRUE</acidcharged>` This element is also optional, however, if the previous element is given, then this one is compulsory. Telling if the conjugated acid form is charged (that is protonated) is essential in order to know what sign the charge has to be when the chemical group is ionized. The value "TRUE" indicates that when the pH is lower than the pKa, the chemical group is charged, thus protonated (in the form NH_3^+). Consequently, if the pH is higher than the pKa, then the chemical group is neutral (in the form NH_2);

- `<polrule>left_trapped</polrule>` This element indicates that the chemical group should only be taken into account in the eventuality that the monomer bearing it (code 'D') is the left end monomer of the polymer sequence. This can easily be understood, as this chemical group is responsible for the establishment of the inter-monomer bond towards the left end of the polymer sequence;

- `<chemgrouprule>` This element provides further details on the chemistry that this chemical group might be involved in:

  - `<entity>LE_PLM_MODIF</entity>` This element indicates that the supplementary data in the current `<chemgrouprule>` element are pertaining to the αNH_2 chemical group *only* in case the polymer sequence is left end-modified (that is with a permanent left end modification) and the monomer (code 'D') is located at the left end of the polymer sequence (that is: it is the first monomer of the sequence for which the electrical charge—or pI—calculation is to be performed).

  - `<name>Acetylation</name>` This element goes further in the detail of the potential chemistry of the αNH_2 chemical group: if the left end permanent modification is "Acetylation", then the current chemgrouprule element can be further processed, otherwise it should be abandoned;

  - `<outcome>LOST</outcome>` This element actually indicates what should be done with the chemical group for which the chemgrouprule is being defined. What we see here is: —*"If the αNH_2 chemical group, belonging to a 'D' monomer located at the left end of a polymer sequence, is modified permanently with an "Acetylation" left end modification, it should not be taken into account when computing the charge that it could bring to the polymer sequence."*

The second `<mnmchemgroup>` element is related to the $\alpha COOH$ carboxylic group of the amino-acid:

- `<name>C-term COOH</name>` Same remark as above;

- `<pka>2.36</pka>` Same remark as above;

- `<acidcharged>FALSE</acidcharged>` Same remark as above. However, as we can see, the value indicates that the acid conjugate (form COOH) does not bring any charge. This means that when the basic conjugate is predominant (that is when pH > pKa), it brings a negative charge: the form is COO^-;

- `<polrule>right_trapped</polrule>` The chemical group should not be evaluated if a monomer is linked to it at its right side. That means that the current chemical group is only evaluated if the monomer bearing it is located at the right end of the polymer sequence. This is easily understood, as the αCOOH chemical group is involved in the formation of the inter-monomer bond towards the right end of the polymer sequence.

The third `<mnmchemgroup>` element is related to the βCOOH carboxylic group of the amino-acid:

- `<name>Lateral COOH</name>`;

- `<pka>3.65</pka>`;

- `<acidcharged>FALSE</acidcharged>`;

- `<polrule>never_trapped</polrule>` This element indicates that, whatever the position of the monomer bearing the chemical group in the polymer sequence (left end, right end or middle), the chemical group is to be evaluated;

- `<chemgrouprule>` This element provides further details on the chemistry that the chemical group at hand (βCOOH) might be involved in:

  - `<entity>MONOMER_MODIF</entity>` This element indicates that the supplementary data in the current `<chemgrouprule>` element are pertaining to the βCOOH chemical group *only* in case the monomer bearing the chemical group is chemically modified;

  - `<name>AmidationAsp</name>` This is the modification by which the monomer should be modified in order to have the `<chemgrouprule>` element effectively evaluated;

  - `<outcome>LOST</outcome>` This element actually indicates that if the monomer bearing the chemical group is modified with an "AmidationAsp" chemical modification, then the chemical group should not be evaluated any more for the electrical charge —or pI— calculations, since reacting a carboxylate group with an amino group produces an amide group which is not easily chargeable at physiological pH values.

At this point we should have made it clear how the charge calculations can be configured for the different monomers in the polymer chemistry definition. As usual, the more the polymer chemistry definition is sophisticated, the more sophisticated the computations are allowed.

Ionized Group(s) In Modifications

In the excerpt from the `pka_ph_pi.xml` file below, we see that chemical modifications can also bring charges. The example of the chemical modification "Phosphorylation" shows that when a monomer is phosphorylated, two chemical groups are brought in: the first has a pKa value of 1.2 (that is it will always be deprotonated at physiological pH values), the second has a pKa value of 7 (that is it will be divided by half in a protonated (not charged) form and in an un-protonated (negatively charged) form, leading to a net electrical charge of −0.5.

```
<modif>
  <name>Phosphorylation</name>
  <mdfchemgroup>
    <name>none_set</name>
    <pka>1.2</pka>
    <acidcharged>FALSE</acidcharged>
  </mdfchemgroup>
  <mdfchemgroup>
    <name>none_set</name>
    <pka>6.5</pka>
    <acidcharged>FALSE</acidcharged>
  </mdfchemgroup>
</modif>
```

At this point we should be able to study the way computations are actually performed in the *XpertEdit* module.

pH, pI and Charge Calculations

The user willing to compute charges (positive, negative, net) or the isoelectric point for the current polymer sequence uses the menu *Chemistry⟶pKa pH pI* which triggers the appearance of the window shown in Figure 8.23 on the next page.

This figure shows that the user can calculate the charges (positive, negative and net) beared by the polymer sequence (either the whole sequence or the current selection) by setting the pH value at which the computation should take place. It is also possible to calculate the isoelectric point by clicking onto the Isoelectric Point button.

Note that the computations might involve the permanent left/right modifications of the polymer sequence, as well as the monomer chemical modifications. To configure the way net charge—or pI—calculations are performed, use the calculations engine configuration of the sequence editor window.

General Options

One of the options that are valued most by users is to be able to set the number of decimal places used to diplay numbers. The settings should apply in a distinct

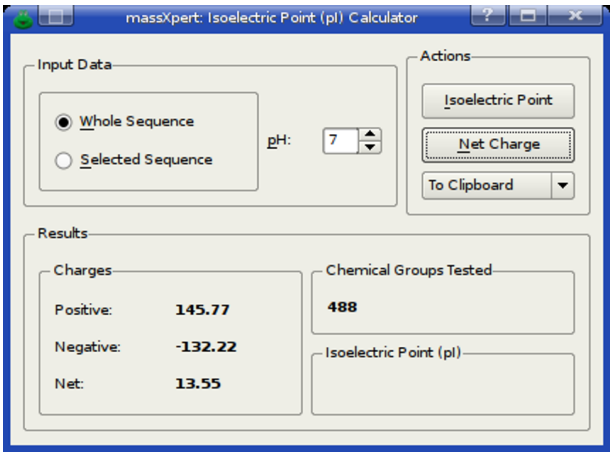


Figure 8.23: **Acido-basic computations: net charges.** This figure shows the options that can be set for the calculation of the charges beared by the polymer sequence.

manner depending on the different entities for which numerical values are to be displayed. The following are the default values (and recommended ones):

* Atoms (and all related entities (isotopic masses, isotopic abundances): 10;
* pKa, pH, pI: 2;
* Oligomers (obtained *via* mass searches, polymer cleavages, oligomer fragmentations): 5;
* Polymers : 3;

Note that modifying these values will allow immediate change of the way numerals are displayed, without needing to restart the program. Only triggering a new cleavage or a new fragmentation will update the data display according to the new options set. These options are stored on the disk and are permanent.

9

XpertMiner: A Data Miner

XpertMiner is a module that has been conceived as a repository of functionalities aimed at analyzing mass data—data which might originate in the *massXpert*-based simulations and/or in the mass spectrometer. *massXpert*, as of version 1.7.9, only contains one "miner" functionality: *mzLab*.

XpertMiner Invocation

The *XpertMiner* module is easily called by pulling down the *XpertMiner* menu item from the *massXpert* program's menu. Clicking on *XpertMiner*⟶*mzLab* will the *mzLab* window, as represented in Figure 9.1 on the following page.

mzLab: Mining m/z ratios

The kinds of data on which the features available in this laboratory will operate is lists of m/z values in the form of a (m/z , z) pair. The mass of the ion is represented by m, while z is the charge of the ion. With the two data in the pair, the m/z ratio and the z charge, and knowing the ionization rule that ionized the analyte in the first place, it is possible to perform any mass calculation on the (m/z , z) pair.

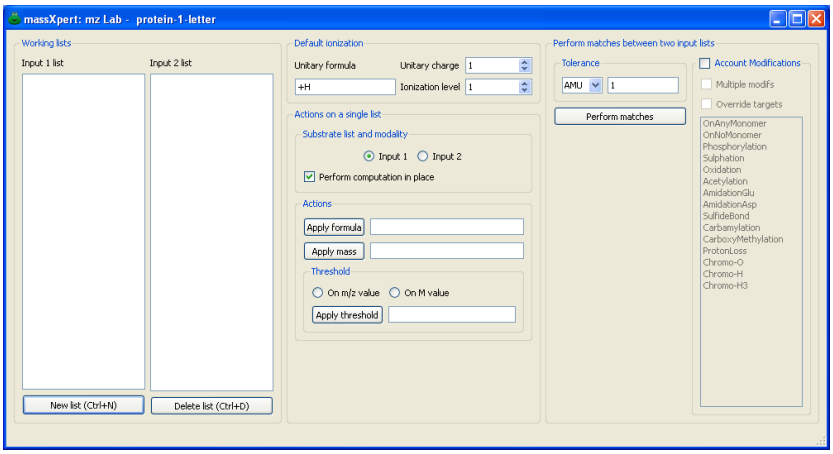


Figure 9.1: ***mzLab* window.** The *mzLab* window is the central location of the laboratory. From there it is possible to open any number of m/z list dialog windows. See text for details.

The *mzLab* window is represented in Figure 9.1. This window is divided into three distinct parts:

- The left part (Working lists) contains two list widgets which will hold the names of the different working m/z lists;
- The central part contains:
  - A group box widget (Default ionization) where the ionization rule for the current polymer chemistry definition is detailed;
  - A group box widget (Actions on a single list) with a number of actions that might be performed on one list of m/z ratios.
- The right part contains a group box widget (Perform matches between two input lists) in which the user may perform matches between lists of m/z ratios.

Creating A New Input m/z List

In order to be able to use the *mzLab* feature, it is necessary to create at least one list of (m/z , z) pairs, which is referred to by "input m/z list", for short. That kind of list is actually a tree view widget that is embedded in a dialog window. The first column of the tree view widget holds the m/z value, and the second column, the z value. To create a new input m/z list, the user clicks onto the button labelled New list. This will trigger the opening of an input dialog window where the user enters an unambiguous name for the new input m/z list. The new input m/z list dialog window shows up empty like in Figure 9.2 on the facing page.

Note that upon creation of a new input m/z list, its name is used to refer to it in the two list widgets on the left of the *mzLab* window. This way, it will be

Figure 9.2: ***mzLab*'s empty input m/z list dialog window.** The *mzLab*'s input m/z list dialog window that shows up when the user creates a new input m/z list is empty. Filling that list is performed by drag-and-drop operations.

possible later to refer to the various input m/z lists by their name. Therefore, it might make sense to use a meaninfgul name for the lists.

Filling Of The Data In Input m/z Lists

Once a new input m/z list has been named and created, it is necessary to fill it with (m/z , z) pairs. This is performed *via* drag-and-drop operations. There might be a number of different data sources to be used for the dragging of data, as detailed below:

- Data from the various simulations available in *massXpert*. These simulations are cleavages, fragmentations and mass searches, which all produce oligomers that are displayed in tree view widgets, as shown in Figure 8.16 on page 87 or Figure 8.17 on page 89 or Figure 8.18 on page 91. Dragging data from these tree view widgets is performed simply by selecting the items of interest in the tree view widget and dragging them to the input m/z list to be filled. Figure 9.3 on the following page shows the data in the input m/z list right after a data drop;

- Data in the form of textual lines, like will be the case when importing a (m/z) or a (m/z , z) pair list from the mass spectrometer's program. There are two cases:

  - If the data dropped are in the form of a list of m/z data, without the z value (one m/z ratio value per line), then the z value will be considered to be the charge that would result from the ionization of the analyte using the ionization rule detailed in the Default ionization group box widget (see above). This case is represented in Figure 9.4 on page 107.

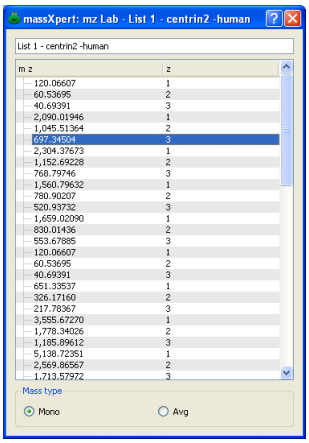


Figure 9.3: ***mzLab*'s data-filled input m/z list dialog window.** The *mzLab*'s input m/z list dialog window fills with data when result items are dragged and dropped onto it.

- If the data dropped are a list of (m/z , z) pairs (one pair per line, like "`1234.567 2`"), then the z value will be read from the dropped data (2, in this example). This case is represented in Figure 9.5 on the facing page.

Imposing The Mass Type: Mono Or Avg

When dropping data—either from results windows (cleavage, fragmentation or mass search) or from textual data—it is necessary to inform the input m/z list of what type the mass of interest is. That is, when dropping a line like `1234.56 1`, is m/z 1234.56 a monoisotopic m/z or an average m/z? The type of the masses dropped in an input m/z list is governed by the two radio buttons labelled Mono and Avg. The one of the two radiobuttons that is checked at the moment the drop occurs determine the type of the m masses that are dropped. It will be possible to check the other radio button widget once a first data drop occurred, but then the user will be alerted about doing so.

Imposing The Oligomer Kind

When dropping data, it is required that the laboratory know if the oligomers are cleavage, mass search or fragmentation oligomers. Indeed, the way the calculations are performed is dependent on the kind of the oligomers used: fragmentation oligomers are not equivalent to cleavage oligomers, because cleavage oligomers are not charged by themselves, while fragmentation oligomers *are* charged by themselves (the charge that the fragmentation oligomer gets upon its creation is intrinsic to it thanks to the fragmentation pattern that gave rise to it). It is thus of crucial importance for the faithfulness of the computations that the laboratory be fed with identified oligomers. This is the reason why the

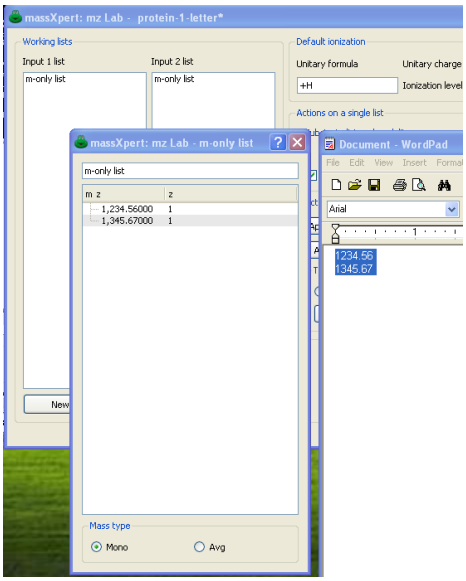


Figure 9.4: ***mzLab*'s (m/z) textual data-filled input m/z list dialog window.** See text for details.

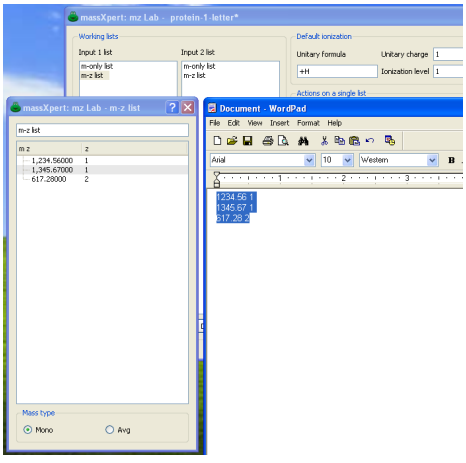


Figure 9.5: ***mzLab*'s (m/z , z) textual data-filled input m/z list dialog window.** See text for details.

input m/z list dialog windows have a Fragments check box widget that the user must check if dealing with fragmentation data in the form of textual data. It is not necessary to check this check box when dropping in the input m/z list data obtained by fragmenting a sequence in *massXpert*, because in that case the program knows that the oligomers are actually fragmentation oligomers and the check box gets checked automatically.

Working On One Input m/z List

Once an input m/z list has been filled with data, it becomes possible to perform simulations on these data. Because there might be any number of input m/z lists open at any given time, it is necessary to identify the input m/z list onto which to perform these simulations. The selection of the input m/z list is performed in two steps: first, the user indicates which list of input m/z lists will contain the input m/z list of interest (select either Input 1 or Input 2 in the Substrate list and modality group box widget) ; second, in that list of input m/z lists, select the input m/z list by its name. If no list is selected, then no simulation might be performed.

Available Calculations

There are a number of operations that might be performed, all of which are selectable in the Actions on a single list group box widget. The simulations are organized into two groups:

* Formula-based actions which involve processing the input m/z lists with formulas (that is, chemical entities represented using formulas):
  - Apply formula will perform the same as above but starting from a formula. This is where it is crucial that the mass type be set correctly, because the type of the mass calculated for the formula will be of the same type as the type of the data;
  - Increment charge by will iterate in all the items present in the list and apply the charge increment to them. One item in the list that is charged 1 will be deionized and reionized to 2 (this calculation involves the ionization rule of the oligomer, and thus its ionization formula);
  - Reionization will iterate in all the items present in the list and apply the new ionization rule, defined in this group box widget.
* Mass-based actions which involve processing the input m/z lists with numerical data representing masses:
  - Apply mass will iterate in all the items present in the list and apply the entered mass to them;
  - Apply threshold will remove all data items in the list for which m/z or M is less than the value set.

Output Of The Calculations

Simulations performed on a single input m/z list produce a m/z list that is identical to the input list, unless for the m and/or z values, which might have changed. This means that it is perfectly possible to:

* Overwrite the initial data with the newly obtained ones (this is performed by checking the Perform computation in place check button widget);

* Create a new list with the newly obtained data. As a convenience for the user, the new list will be an input m/z list in which it will be possible to perform ulterior simulations. This is useful when the simulations that need to be performed are sequential in kind. To have a new list created uncheck Perform computation in place check button widget.

Internal Workings

When an operation is performed on the items of an input m/z list, say we want to make sodium adducts (that would be a formula "$-H+Na$") of all the items in the list, the process involves the following steps, as detailed below for one single item of the list (which has data pair $(334.341\,,3)$ and *protonation* as ionization agent).

* Convert the tri-protonated analyte into a non-ionized analyte, thus getting M=1000; [1]

* Compute the mass of the "$-H+Na$" formula: 21.98 Da;

* Add 1000+21.98;

* Reionize to the initial charge state: $(341.67\,,3)$.

Working On Two Input m/z Lists

It is possible to perform calculations on two input m/z lists. These calculations are called matches. The $(m/z\,,z)$ pairs of two different input m/z lists might be matched. Typically, a match operation would involve data from the mass spectrometer and data from a *massXpert*-based simulation (cleavage or fragmentation, for example). In order to perform a match operation, the first input m/z list (the data from the mass spectrometer) should be selected by its name in the Input 1 List list and the second input m/z list (the data from the simulation) should be selected by its name in the Input 2 List list. Note that if the two input m/z lists are not of the same type (one is mono and the other is avg), the user will be alerted about this point.

[1] Note that if the oligomer is a fragmentation oligomer, the tri-protonated analyte is converted to the canonical oligomer bearing a single charge.

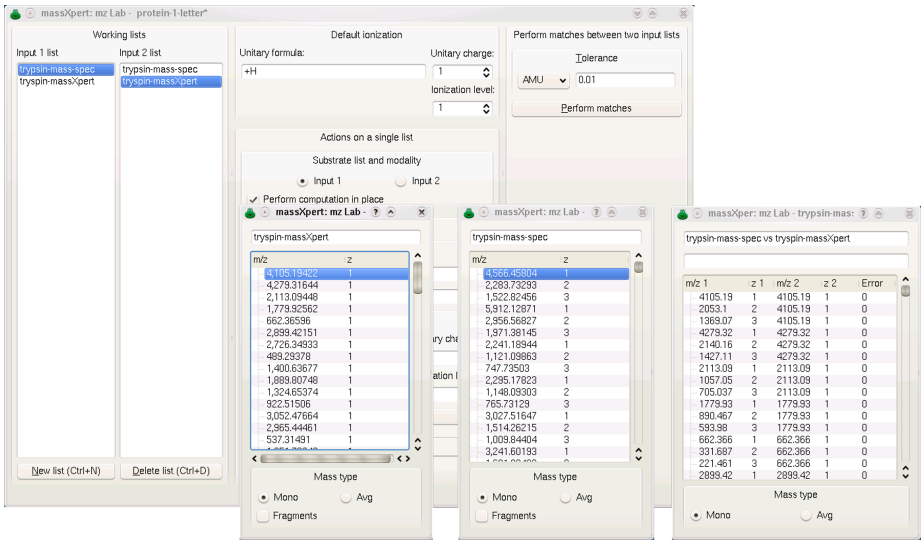


Figure 9.6: ***mzLab*'s match operation output list dialog window.** See text for details.

Output Of The Calculations

Calculations involving matches between two input lists produce an output that is displayed in an output m/z list, which is different from an input m/z list. Figure 9.6 shows the results after having performed a match operation between an input m/z list obtained from the mass spectrometer (Input 1 list) and an input m/z list obtained by simulating a cleavage with trypsin (Input 2 list). The output m/z list dialog window holds all the matches along with the original data and the error.

Tracing The Data

When the data used for filling an input m/z list come from a *massXpert*-based simulation it is possible to trace back the (m/z , z) pair items to the corresponding sequence in the polymer sequence editor that gave rise to these oligomers in the first place. This is only possible if:

* The way the data were fed into the input m/z list was by dragging oligomers from the tree view widgets, as described earlier;
* The polymer sequence window is still opened when the tracing back is tried.

In order to trace back any given item in an input or in an output m/z list to its corresponding polyemr sequence, just activate the item while having a look at the polymer sequence whence the oligomers initially originated. Each time an item is selected, its corresponding sequence region will be selected in the polymer sequence.

10

Data Customization

In this chapter, the user will be walked trough an example of how new polymer chemistry definition data can be generated and included in the automatic "data detection system" of *massXpert* (that is how new polymer chemistry definitions should be registered with the system).

Customization is typically performed by the normal user (not the Administrator nor the Root of the machine) and as such new data are typically stored in the user's "home" directory. On *UNIX* machines, the "home" directory is usually the `/home/username` directory, where username is the logging user name. On *MS-Windows*, that directory is typically the `C:/Documents and Settings/username`[1], once again with username being the logon user name.

In the next sections we will refer to that "home directory" (be it on *UNIX* or *MS-Windows* machines) as the $HOME directory, as this the standard environment variable describing that directory in *GNU/Linux*.

When *massXpert* is executed, it automatically tries to read data configuration files from the home directory (in the `.massxpert` directory). Once this is done, it reads all the data configuration files in the installation directory (typically, on *GNU/Linux* that would be the configuration data in the `/usr/local/share/massxpert` directory or, on *MS-Windows*, the `c:/Program Files/massxpert` directory).

[1] Although *MS-Windows* pathnames use a back slash, in this book these are composed using forward slashes for a number of valid reasons. The reader only needs to replace back slashes with the forward variety.

We said above that *massXpert* tries to read the data configuration files from the home directory. But upon its very first execution, right after installation, that directory does not exist, and in fact *massXpert* creates that directory for us to populate it some day with interesting new data.

The `$HOME/.massxpert` directory should have a structure mimicking the one that was created upon installation of the software, that is, it should contain the following two directories:

* `pol-chem-defs`
* `plugins`

Those are the directories where the user is invited to store her personal data. In order to start a new definition, one might simply copy there one of the polymer chemistry definitions that are shipped with *massXpert*. What should be copied? An entire polymer chemistry definition directory, like for example the following:

`/usr/local/share/massxpert/pol-chem-defs/protein-1-letter`

or

`C:/Program Files/massxpert/data/pol-chem-defs/protein-1-letter`

Once that polymer chemistry definition is copied, one may start studying how it actually works. This directory contains the following kinds of files:

* `protein-1-letter.xml`: the polymer chemistry definition file. This is the file that is read upon selection of the corresponding polymer chemistry definition name in *XpertDef*. If the polymer chemistry definition is not yet registered with the system (described later), then open that file by browsing to it by clicking the Cancel button.[2];
* *svg* files: *scalar vector graphics* files used to render graphically the sequence in the sequence editor. For example, `arginine.svg` contains the graphical representation of the arginine monomer. There are such graphics files also for the modifications (like, for example, the `sulphation.svg` contains the graphical representation of the sulphation modification. Figure 10.1 shows two examples of *svg* files belonging to two distinct polymer chemistry definitions;
* `chem_pad.conf`: configuration file for the chemical pad in the *XpertCalc* module;
* `monomer_dictionary`: file establishing the relationship between any monomer code of the polymer chemistry definition and the graphical *svg* file to be used to render graphically that monomer in the sequence editor;
* `modification_dictionary`: file establishing the relationship between any monomer modification[3] and the graphical *svg* file to be used to render graphically that modification onto the modified monomer in the sequence editor;

[2] See chapter 6, page 43.
[3] See section 8, page 81.

* `cross_linker_dictionary`: file establishing the relationship between any cross-link[4] and the graphical *svg* file to be used to render graphically that cross-link onto the cross-linked monomers in the sequence editor;

* `pka_ph_pi.xml`: file describing the acido-basic data[5] pertaining to ionizable chemical groups in the different entities of the polymer chemistry definition;

The polymer sequence editor is not a classical editor. There is no font in this editor: when the user starts keying-in a polymer sequence in the editor, the small *svg* graphics files are rendered into raster *vignettes* at both the proper resolution and screen size and displayed in the sequence editor. The user is totally in charge of designing the *svg* graphics files for each of the monomers defined in the polymer sequence editor. Of course, reusing material is perfectly possible. There is one constraint: that the `monomer_dictionary` file lists with precision "what code goes with what *svg* graphics file". That file has the following contents, for example, for the "protein-1-letter" polymer chemistry definition, as shipped in the *massXpert* package:

```
# This file is part of the massXpert project.

# The "massXpert" project is released ---in its entirety--- under the
# GNU General Public License and was started (in the form of the GNU
# polyxmass project) at the Centre National de la Recherche
# Scientifique (FRANCE), that granted me the formal authorization to
# publish it under this Free Software License.

# Copyright (C) 2006,2007 Filippo Rusconi

# This is the monomer_dictionary file where the correspondences
# between the codes of each monomer and their graphic file (pixmap
# file called "image") used to graphicallly render them in the
# sequence editor are made.

# The format of the file is like this :
# -------------------------------------

# A%alanine.svg

# where A is the monomer code and alanine.svg is a
# resolution-independent svg file.

# Each line starting with a '#' character is a comment and is ignored
# during parsing of this file.

# This file is case-sensitive.
```

[4] See section 8, page 85.
[5] See section 8, page 93.

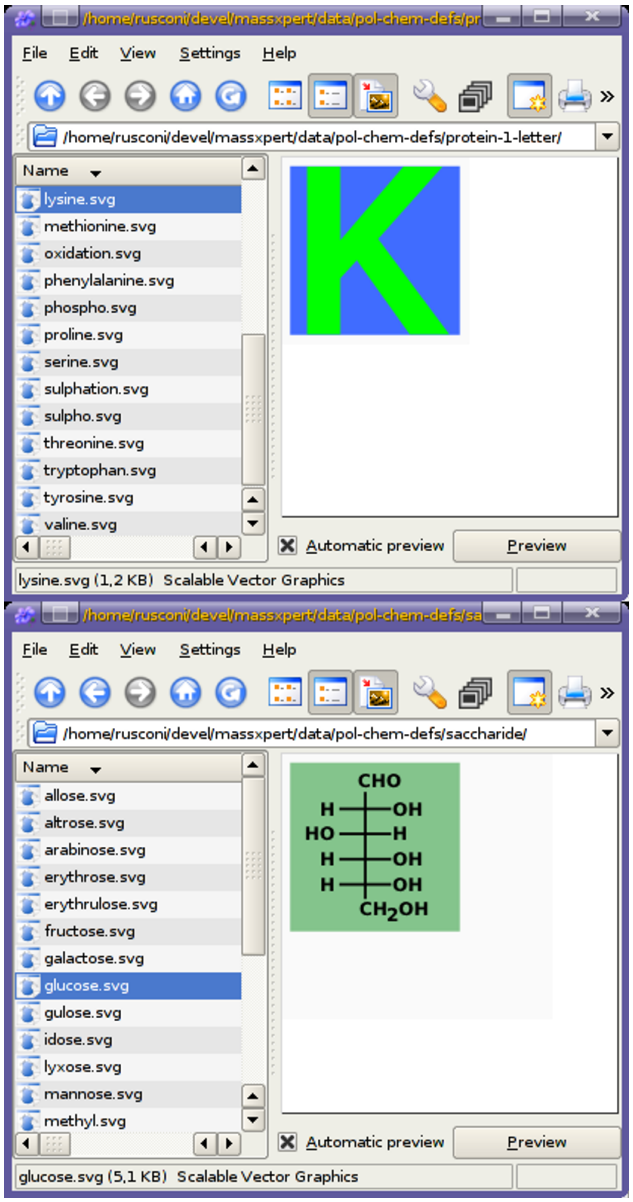


Figure 10.1: **The polymer chemistry definition directory.** Each monomer of the polymer chemistry definition ought to have a corresponding *svg* file with which it has to be rendered graphically should that monomer be inserted in the polymer sequence. This example shows two *svg* files corresponding to two monomers each belonging to a different polymer chemistry definition.

```
A%alanine.svg
C%cysteine.svg
D%aspartate.svg
E%glutamate.svg
F%phenylalanine.svg
G%glycine.svg
H%histidine.svg
I%isoleucine.svg
K%lysine.svg
L%leucine.svg
M%methionine.svg
N%asparagine.svg
P%proline.svg
Q%glutamine.svg
R%arginine.svg
S%serine.svg
T%threonine.svg
V%valine.svg
W%tryptophan.svg
Y%tyrosine.svg
```

What one sees from the contents of the file is that each monomer code has an associated *svg* file. For example, when the user has to key-in a valine monomer, she keys-in the code V and *XpertEdit* knows that the monomer vignette to show has to be rendered using the `valine.svg` file.

For the monomer modification graphical rendering, the situation is somewhat different, as seen in the `modification_dictionary` file:

```
# This file is part of the massXpert project.

# The "massXpert" project is released ---in its entirety--- under the
# GNU General Public License and was started (in the form of the GNU
# polyxmass project) at the Centre National de la Recherche
# Scientifique (FRANCE), that granted me the formal authorization to
# publish it under this Free Software License.

# Copyright (C) 2006,2007 Filippo Rusconi

# This is the modification_dictionary file where the correspondences
# between the name of each modification and their graphic file (pixmap
# file called "image") used to graphicallly render them in the
# sequence editor are made. Also, the graphical operation that is to
# be performed upon chemical modification of a monomer is listed ('T'
# for transparent and 'O' for opaque). See the manual for details.

# The format of the file is like this :
# -------------------------------------
```

```
# Phosphorylation%T%phospho.svg

# where Phosphorylation is the name of the modification. T indicates
# that the visual rendering of the modification is a transparent
# process (O indicates that the visual rendering of the modification
# is a full image replacement 'O' like opaque). phospho.svg is a
# resolution-independent svg file.


# Each line starting with a '#' character is a comment and is ignored
# during parsing of this file.

# This file is case-sensitive.

Phosphorylation%T%phospho.svg
Sulphation%T%sulpho.svg
AmidationAsp%O%asparagine.svg
Acetylation%T%acetyl.svg
AmidationGlu%O%glutamine.svg
Oxidation%T%oxidation.svg
```

There are two ways to render a chemical modification of a monomer:

* **Opaque** rendering: the initial monomer vignette is replaced using the one listed in the file for the modification. This is visible in the `AmidationGlu\%O\%glutamine.svg` line: when a monomer is (typically that would be a Glu monomer) is amidated, the graphical representation of the modification process should involve the *replacement* of the old vignette in the sequence editor with the new one (in the example, the new vignette should be rendered using the `glutamine.svg` file. In other words, the process involves an "**O**paque" overlay of the vignette for unmodified Glu with a vignette rendered by using the `glutamine.svg` file.

* textbfTransparent rendering: the initial monomer vignette is overlaid with one new vignette that is rendered using a *svg* file that is transparent (unless for the graphical motif to be made visible, of course). One example is the "Phosphorylation" modification (line `Phosphorylation%T%phospho.svg`), for which the monomer being phosphorylated has its vignette in the sequence editor overlaid with a "**T**ransparent" one which only shows a small red 'P' and that is rendered using the `phospho.svg` file.

The way new *svg* files might be edited is using the following programs:

* `Inkscape`: on *GNU/Linux* and *MS-Windows*;

* `Karbon`: on *GNU/Linux*;

In general, the best thing to do is to convert text to path, so that the rendering is absolutely perfect.

It is absolutely essential, for the proper working of the sequence editor, that the *svg* files be square (that is, width = height).

Once the new polymer chemistry has been correctly defined, it is time to register that new definition to the system. To recap: all the files for that definition should reside in a same directory, exactly the same way as the files pertaining to a given polymer chemistry definition are shipped in *massXpert* altogether in one directory. The name of the new polymer chemistry definition should be unambiguous, with respect to other registered polymer chemistry definitions.

The way a polymer chemistry definition is registered is by created a personal polymer chemistry definition catalogue file, which must comply with two requirements:

* Be named `xxxxx-pol-chem-defs-cat`, with `xxxxx` being a discretionary string (this might well be your name, for example). The requirement is that `-pol-chem-defs-cat` be the last part of the filename. Please *DO NOT USE* spaces, punctuation or diacritical signs in your filenames. *RESTRICT* yourself to ASCII characters between [a-z], [0-9], '_' and '-'.[6]

* Be located in the `$HOME/.massxpert/pol-chem-defs` directory and have the following format:

  `dna=/path/to/definition/directory/dna/dna.xml`. In this example, the "dna" polymer chemistry definition is being registered as a file `dna.xml` located in the `dna` directory, itself located in the `/path/to/definition/directory` directory;

Note that if a new polymer chemistry definition should be made available system-wide, then it is logical that its directory be placed along the ones shipped with *massXpert* and a new local catalogue file might be created to register the new polymer chemistry definition.

At this point the new polymer chemistry definition might be tested. Typically, that involves restarting the *massXpert* program and creating a brand new polymer sequence of the new definition type. The first step is to check if the new definition is successfully registered with the system, that is, it should show up a an available definition upon creation of the new polymer sequence. If not, then that means that the catalogue file could not be found or parsed correctly.

When problems like this one occurs, the first thing to do is to ensure that the console window (on *MS-Windows* it is systematically started along with the program; on *GNU/Linux* the way to have it is to start the program from the shell) so as to look with attention at the different messages that might help understanding what is failing.

Please, do not hesitate to submit bug reports (see the first pages of this manual for the address where to post bug reports).

[6] This is actually something very general as a recommendation in order to not suffer from severe headaches when you expect it less...

11

Appendices

The Protein Chemistry Definition File

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- DTD for polymer definitions, used by the
'massXpert' mass spectrometry application.
Copyright 2006,2007 Filippo Rusconi - Licensed under the GNU GPL -->
<!DOCTYPE polchemdefinition [
<!ELEMENT polchemdefinition (atomdefdata,polchemdefdata)>
<!ATTLIST polchemdefinition version NMTOKEN #REQUIRED>
<!ELEMENT atomdefdata (atom+)>
<!ELEMENT atom (name,symbol,isotope+)>
<!ELEMENT symbol (#PCDATA)>
<!ELEMENT isotope (mass , abund)>
<!ELEMENT mass (#PCDATA)>
<!ELEMENT abund (#PCDATA)>
<!ELEMENT polchemdefdata (name,leftcap,rightcap,codelen,ionizerule,monomers,modifs,crosslinkers
<!ELEMENT ionizerule (formula,charge,level)>
<!ELEMENT monomers (mnm*)>
<!ELEMENT modifs (mdf*)>
<!ELEMENT crosslinkers (clk*)>
<!ELEMENT cleavespecs (cls*)>
<!ELEMENT fragspecs (fgs*)>
<!ELEMENT mnm (name,code,formula)>
<!ELEMENT mdf (name,formula,targets)>
<!ELEMENT clk (name,formula,modifname*)>
```

```
<!ELEMENT cls (name,pattern,clr*)>
<!ELEMENT fgs (name,end,formula,comment?,fgr*)>
<!ELEMENT clr (name,(le-mnm-code,le-formula)?,(re-mnm-code,re-formula)?)>
<!ELEMENT fgr (name,formula,prev-mnm-code?,curr-mnm-code?,next-mnm-code?,comment?)>
<!ELEMENT leftcap (#PCDATA)>
<!ELEMENT rightcap (#PCDATA)>
<!ELEMENT codelen (#PCDATA)>
<!ELEMENT charge (#PCDATA)>
<!ELEMENT level (#PCDATA)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT modifname (#PCDATA)>
<!ELEMENT code (#PCDATA)>
<!ELEMENT formula (#PCDATA)>
<!ELEMENT targets (#PCDATA)>
<!ELEMENT pattern (#PCDATA)>
<!ELEMENT end (#PCDATA)>
<!ELEMENT le-mnm-code (#PCDATA)>
<!ELEMENT re-mnm-code (#PCDATA)>
<!ELEMENT le-formula (#PCDATA)>
<!ELEMENT re-formula (#PCDATA)>
<!ELEMENT comment (#PCDATA)>
<!ELEMENT prev-mnm-code (#PCDATA)>
<!ELEMENT curr-mnm-code (#PCDATA)>
<!ELEMENT next-mnm-code (#PCDATA)>
]>
<polchemdefinition version="3">
  <atomdefdata>
    <atom>
      <name>Nullor</name>
      <symbol>Nul</symbol>
      <isotope>
        <mass>0.0000000000</mass>
        <abund>100.0000000000</abund>
      </isotope>
    </atom>
    <atom>
      <name>Decilor</name>
      <symbol>Dic</symbol>
      <isotope>
        <mass>0.1000000000</mass>
        <abund>100.0000000000</abund>
      </isotope>
    </atom>
    <atom>
      <name>Unitor</name>
      <symbol>Uno</symbol>
      <isotope>
        <mass>1.0000000000</mass>
        <abund>100.0000000000</abund>
      </isotope>
```

```
</atom>
<atom>
  <name>Hydrogen</name>
  <symbol>H</symbol>
  <isotope>
    <mass>1.0078250400</mass>
    <abund>99.9885000000</abund>
  </isotope>
  <isotope>
    <mass>2.0141017900</mass>
    <abund>0.0115000000</abund>
  </isotope>
</atom>
<atom>
  <name>Helium</name>
  <symbol>He</symbol>
  <isotope>
    <mass>3.0160293000</mass>
    <abund>0.0001400000</abund>
  </isotope>
  <isotope>
    <mass>4.0026032500</mass>
    <abund>99.9998600000</abund>
  </isotope>
</atom>
<atom>
  <name>Lithium</name>
  <symbol>Li</symbol>
  <isotope>
    <mass>6.0151232000</mass>
    <abund>7.5900000000</abund>
  </isotope>
  <isotope>
    <mass>7.0160045000</mass>
    <abund>92.4100000000</abund>
  </isotope>
</atom>
<atom>
  <name>Beryllium</name>
  <symbol>Be</symbol>
  <isotope>
    <mass>9.0121825000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Decanor</name>
  <symbol>Dac</symbol>
  <isotope>
    <mass>10.0000000000</mass>
```

```
      <abund>100.0000000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Bore</name>
    <symbol>B</symbol>
    <isotope>
      <mass>10.0129380000</mass>
      <abund>19.9000000000</abund>
    </isotope>
    <isotope>
      <mass>11.0093053000</mass>
      <abund>80.1000000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Carbon</name>
    <symbol>C</symbol>
    <isotope>
      <mass>12.0000000000</mass>
      <abund>98.9300000000</abund>
    </isotope>
    <isotope>
      <mass>13.0033548000</mass>
      <abund>1.0700000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Nitrogen</name>
    <symbol>N</symbol>
    <isotope>
      <mass>14.0030740000</mass>
      <abund>99.6320000000</abund>
    </isotope>
    <isotope>
      <mass>15.0001090000</mass>
      <abund>0.3680000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Oxygen</name>
    <symbol>O</symbol>
    <isotope>
      <mass>15.9949146000</mass>
      <abund>99.7570000000</abund>
    </isotope>
    <isotope>
      <mass>16.9991306000</mass>
      <abund>0.0380000000</abund>
    </isotope>
```

```
  <isotope>
    <mass>17.9991594000</mass>
    <abund>0.2050000000</abund>
  </isotope>
</atom>
<atom>
  <name>Fluorine</name>
  <symbol>F</symbol>
  <isotope>
    <mass>18.9984032000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Neon</name>
  <symbol>Ne</symbol>
  <isotope>
    <mass>19.9924391000</mass>
    <abund>90.4800000000</abund>
  </isotope>
  <isotope>
    <mass>20.9938453000</mass>
    <abund>0.2700000000</abund>
  </isotope>
  <isotope>
    <mass>21.9913837000</mass>
    <abund>9.2500000000</abund>
  </isotope>
</atom>
<atom>
  <name>Sodium</name>
  <symbol>Na</symbol>
  <isotope>
    <mass>22.9897697000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Magnesium</name>
  <symbol>Mg</symbol>
  <isotope>
    <mass>23.9850450000</mass>
    <abund>78.9900000000</abund>
  </isotope>
  <isotope>
    <mass>24.9858392000</mass>
    <abund>10.0000000000</abund>
  </isotope>
  <isotope>
    <mass>25.9825954000</mass>
```

```
    <abund>11.0100000000</abund>
  </isotope>
</atom>
<atom>
  <name>Aluminium</name>
  <symbol>Al</symbol>
  <isotope>
    <mass>26.9815413000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Silicon</name>
  <symbol>Si</symbol>
  <isotope>
    <mass>27.9769284000</mass>
    <abund>92.2297000000</abund>
  </isotope>
  <isotope>
    <mass>28.9764964000</mass>
    <abund>4.6832000000</abund>
  </isotope>
  <isotope>
    <mass>29.9737717000</mass>
    <abund>3.0872000000</abund>
  </isotope>
</atom>
<atom>
  <name>Phosphorus</name>
  <symbol>P</symbol>
  <isotope>
    <mass>30.9737634000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Sulfur</name>
  <symbol>S</symbol>
  <isotope>
    <mass>31.9720718000</mass>
    <abund>94.9300000000</abund>
  </isotope>
  <isotope>
    <mass>32.9714591000</mass>
    <abund>0.7600000000</abund>
  </isotope>
  <isotope>
    <mass>33.9678677000</mass>
    <abund>4.2900000000</abund>
  </isotope>
```

```
  <isotope>
    <mass>35.9670790000</mass>
    <abund>0.0200000000</abund>
  </isotope>
</atom>
<atom>
  <name>Chlorine</name>
  <symbol>Cl</symbol>
  <isotope>
    <mass>34.9688527000</mass>
    <abund>75.7800000000</abund>
  </isotope>
  <isotope>
    <mass>36.9659026000</mass>
    <abund>24.2200000000</abund>
  </isotope>
</atom>
<atom>
  <name>Argon</name>
  <symbol>Ar</symbol>
  <isotope>
    <mass>35.9675456000</mass>
    <abund>0.3365000000</abund>
  </isotope>
  <isotope>
    <mass>37.9627322000</mass>
    <abund>0.0632000000</abund>
  </isotope>
  <isotope>
    <mass>39.9623831000</mass>
    <abund>99.6003000000</abund>
  </isotope>
</atom>
<atom>
  <name>Potassium</name>
  <symbol>K</symbol>
  <isotope>
    <mass>38.9637079000</mass>
    <abund>93.2581000000</abund>
  </isotope>
  <isotope>
    <mass>39.9639988000</mass>
    <abund>0.0117000000</abund>
  </isotope>
  <isotope>
    <mass>40.9618254000</mass>
    <abund>6.7302000000</abund>
  </isotope>
</atom>
<atom>
```

```
  <name>Calcium</name>
  <symbol>Ca</symbol>
  <isotope>
    <mass>39.9625907000</mass>
    <abund>96.9410000000</abund>
  </isotope>
  <isotope>
    <mass>41.9586218000</mass>
    <abund>0.6470000000</abund>
  </isotope>
  <isotope>
    <mass>42.9587704000</mass>
    <abund>0.1350000000</abund>
  </isotope>
  <isotope>
    <mass>43.9554848000</mass>
    <abund>2.0860000000</abund>
  </isotope>
  <isotope>
    <mass>45.9536890000</mass>
    <abund>0.0040000000</abund>
  </isotope>
  <isotope>
    <mass>47.9525320000</mass>
    <abund>0.1870000000</abund>
  </isotope>
</atom>
<atom>
  <name>Scandium</name>
  <symbol>Sc</symbol>
  <isotope>
    <mass>44.9559136000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Titanium</name>
  <symbol>Ti</symbol>
  <isotope>
    <mass>45.9526327000</mass>
    <abund>8.2500000000</abund>
  </isotope>
  <isotope>
    <mass>46.9517649000</mass>
    <abund>7.4400000000</abund>
  </isotope>
  <isotope>
    <mass>47.9479467000</mass>
    <abund>73.7200000000</abund>
  </isotope>
```

```
  <isotope>
    <mass>48.9478705000</mass>
    <abund>5.4100000000</abund>
  </isotope>
  <isotope>
    <mass>49.9447858000</mass>
    <abund>5.1800000000</abund>
  </isotope>
</atom>
<atom>
  <name>Vanadium</name>
  <symbol>V</symbol>
  <isotope>
    <mass>49.9471613000</mass>
    <abund>0.2500000000</abund>
  </isotope>
  <isotope>
    <mass>50.9439625000</mass>
    <abund>99.7500000000</abund>
  </isotope>
</atom>
<atom>
  <name>Chromium</name>
  <symbol>Cr</symbol>
  <isotope>
    <mass>49.9464630000</mass>
    <abund>4.3450000000</abund>
  </isotope>
  <isotope>
    <mass>51.9405097000</mass>
    <abund>83.7890000000</abund>
  </isotope>
  <isotope>
    <mass>52.9406510000</mass>
    <abund>9.5010000000</abund>
  </isotope>
  <isotope>
    <mass>53.9388822000</mass>
    <abund>2.3650000000</abund>
  </isotope>
</atom>
<atom>
  <name>Manganese</name>
  <symbol>Mn</symbol>
  <isotope>
    <mass>54.9380463000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
```

```
  <name>Iron</name>
  <symbol>Fe</symbol>
  <isotope>
    <mass>53.9396121000</mass>
    <abund>5.8450000000</abund>
  </isotope>
  <isotope>
    <mass>55.9349393000</mass>
    <abund>91.7540000000</abund>
  </isotope>
  <isotope>
    <mass>56.9353957000</mass>
    <abund>2.1190000000</abund>
  </isotope>
  <isotope>
    <mass>57.9332778000</mass>
    <abund>0.2820000000</abund>
  </isotope>
</atom>
<atom>
  <name>Cobalt</name>
  <symbol>Co</symbol>
  <isotope>
    <mass>58.9331978000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Nickel</name>
  <symbol>Ni</symbol>
  <isotope>
    <mass>57.9353471000</mass>
    <abund>68.0769000000</abund>
  </isotope>
  <isotope>
    <mass>59.9307890000</mass>
    <abund>26.2231000000</abund>
  </isotope>
  <isotope>
    <mass>60.9310586000</mass>
    <abund>1.1399000000</abund>
  </isotope>
  <isotope>
    <mass>61.9283464000</mass>
    <abund>3.6345000000</abund>
  </isotope>
  <isotope>
    <mass>63.9279680000</mass>
    <abund>0.9256000000</abund>
  </isotope>
```

```
</atom>
<atom>
  <name>Copper</name>
  <symbol>Cu</symbol>
  <isotope>
    <mass>62.9295992000</mass>
    <abund>69.1700000000</abund>
  </isotope>
  <isotope>
    <mass>64.9277924000</mass>
    <abund>30.8300000000</abund>
  </isotope>
</atom>
<atom>
  <name>Zinc</name>
  <symbol>Zn</symbol>
  <isotope>
    <mass>63.9291454000</mass>
    <abund>48.6300000000</abund>
  </isotope>
  <isotope>
    <mass>65.9260352000</mass>
    <abund>27.9000000000</abund>
  </isotope>
  <isotope>
    <mass>66.9271289000</mass>
    <abund>4.1000000000</abund>
  </isotope>
  <isotope>
    <mass>67.9248458000</mass>
    <abund>18.7500000000</abund>
  </isotope>
  <isotope>
    <mass>69.9253249000</mass>
    <abund>0.6200000000</abund>
  </isotope>
</atom>
<atom>
  <name>Gallium</name>
  <symbol>Ga</symbol>
  <isotope>
    <mass>68.9255809000</mass>
    <abund>60.1080000000</abund>
  </isotope>
  <isotope>
    <mass>70.9247006000</mass>
    <abund>39.8920000000</abund>
  </isotope>
</atom>
<atom>
```

```
  <name>Germanium</name>
  <symbol>Ge</symbol>
  <isotope>
    <mass>69.9242498000</mass>
    <abund>20.8400000000</abund>
  </isotope>
  <isotope>
    <mass>71.9220800000</mass>
    <abund>27.5400000000</abund>
  </isotope>
  <isotope>
    <mass>72.9234639000</mass>
    <abund>7.7300000000</abund>
  </isotope>
  <isotope>
    <mass>73.9211788000</mass>
    <abund>36.2800000000</abund>
  </isotope>
  <isotope>
    <mass>75.9214027000</mass>
    <abund>7.6100000000</abund>
  </isotope>
</atom>
<atom>
  <name>Arsenic</name>
  <symbol>As</symbol>
  <isotope>
    <mass>74.9215955000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Selenium</name>
  <symbol>Se</symbol>
  <isotope>
    <mass>73.9224771000</mass>
    <abund>0.8900000000</abund>
  </isotope>
  <isotope>
    <mass>75.9192066000</mass>
    <abund>9.3700000000</abund>
  </isotope>
  <isotope>
    <mass>76.9199077000</mass>
    <abund>7.6300000000</abund>
  </isotope>
  <isotope>
    <mass>77.9173040000</mass>
    <abund>23.7700000000</abund>
  </isotope>
```

```
  <isotope>
    <mass>79.9165205000</mass>
    <abund>49.6100000000</abund>
  </isotope>
  <isotope>
    <mass>81.9167090000</mass>
    <abund>8.7300000000</abund>
  </isotope>
</atom>
<atom>
  <name>Bromine</name>
  <symbol>Br</symbol>
  <isotope>
    <mass>78.9183361000</mass>
    <abund>50.6900000000</abund>
  </isotope>
  <isotope>
    <mass>80.9162900000</mass>
    <abund>49.3100000000</abund>
  </isotope>
</atom>
<atom>
  <name>Krypton</name>
  <symbol>Kr</symbol>
  <isotope>
    <mass>77.9203970000</mass>
    <abund>0.3500000000</abund>
  </isotope>
  <isotope>
    <mass>79.9163750000</mass>
    <abund>2.2800000000</abund>
  </isotope>
  <isotope>
    <mass>81.9134830000</mass>
    <abund>11.5800000000</abund>
  </isotope>
  <isotope>
    <mass>82.9141340000</mass>
    <abund>11.4900000000</abund>
  </isotope>
  <isotope>
    <mass>83.9115064000</mass>
    <abund>57.0000000000</abund>
  </isotope>
  <isotope>
    <mass>85.9106140000</mass>
    <abund>17.3000000000</abund>
  </isotope>
</atom>
<atom>
```

```
  <name>Rubidium</name>
  <symbol>Rb</symbol>
  <isotope>
    <mass>84.9117996000</mass>
    <abund>72.1700000000</abund>
  </isotope>
  <isotope>
    <mass>86.9091836000</mass>
    <abund>27.8300000000</abund>
  </isotope>
</atom>
<atom>
  <name>Strontium</name>
  <symbol>Sr</symbol>
  <isotope>
    <mass>83.9134280000</mass>
    <abund>0.5600000000</abund>
  </isotope>
  <isotope>
    <mass>85.9092732000</mass>
    <abund>9.8600000000</abund>
  </isotope>
  <isotope>
    <mass>86.9088902000</mass>
    <abund>7.0000000000</abund>
  </isotope>
  <isotope>
    <mass>87.9056249000</mass>
    <abund>82.5800000000</abund>
  </isotope>
</atom>
<atom>
  <name>Yttrium</name>
  <symbol>Y</symbol>
  <isotope>
    <mass>88.9058560000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Zirconium</name>
  <symbol>Zr</symbol>
  <isotope>
    <mass>89.9047080000</mass>
    <abund>51.4500000000</abund>
  </isotope>
  <isotope>
    <mass>90.9056442000</mass>
    <abund>11.2200000000</abund>
  </isotope>
```

```
  <isotope>
    <mass>91.9050392000</mass>
    <abund>17.1500000000</abund>
  </isotope>
  <isotope>
    <mass>93.9063191000</mass>
    <abund>17.3800000000</abund>
  </isotope>
  <isotope>
    <mass>95.9082720000</mass>
    <abund>2.8000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Niobium</name>
  <symbol>Nb</symbol>
  <isotope>
    <mass>92.9063780000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Molybdenum</name>
  <symbol>Mo</symbol>
  <isotope>
    <mass>91.9068090000</mass>
    <abund>14.8400000000</abund>
  </isotope>
  <isotope>
    <mass>93.9050862000</mass>
    <abund>9.2500000000</abund>
  </isotope>
  <isotope>
    <mass>94.9058379000</mass>
    <abund>15.9200000000</abund>
  </isotope>
  <isotope>
    <mass>95.9046755000</mass>
    <abund>16.6800000000</abund>
  </isotope>
  <isotope>
    <mass>96.9060179000</mass>
    <abund>9.5500000000</abund>
  </isotope>
  <isotope>
    <mass>97.9054050000</mass>
    <abund>24.1300000000</abund>
  </isotope>
  <isotope>
    <mass>99.9074730000</mass>
```

```
      <abund>9.6300000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Hector</name>
    <symbol>Hec</symbol>
    <isotope>
      <mass>100.0000000000</mass>
      <abund>100.0000000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Rutenium</name>
    <symbol>Ru</symbol>
    <isotope>
      <mass>95.9075960000</mass>
      <abund>5.5400000000</abund>
    </isotope>
    <isotope>
      <mass>97.9052870000</mass>
      <abund>1.8700000000</abund>
    </isotope>
    <isotope>
      <mass>98.9059371000</mass>
      <abund>12.7600000000</abund>
    </isotope>
    <isotope>
      <mass>99.9042175000</mass>
      <abund>12.6000000000</abund>
    </isotope>
    <isotope>
      <mass>100.9055810000</mass>
      <abund>17.0600000000</abund>
    </isotope>
    <isotope>
      <mass>101.9043480000</mass>
      <abund>31.5500000000</abund>
    </isotope>
    <isotope>
      <mass>103.9054220000</mass>
      <abund>18.6200000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Rhodium</name>
    <symbol>Rh</symbol>
    <isotope>
      <mass>102.9055030000</mass>
      <abund>100.0000000000</abund>
    </isotope>
```

```
</atom>
<atom>
  <name>Palladium</name>
  <symbol>Pd</symbol>
  <isotope>
    <mass>101.9056090000</mass>
    <abund>1.0200000000</abund>
  </isotope>
  <isotope>
    <mass>103.9040260000</mass>
    <abund>11.1400000000</abund>
  </isotope>
  <isotope>
    <mass>104.9050750000</mass>
    <abund>22.3300000000</abund>
  </isotope>
  <isotope>
    <mass>105.9034750000</mass>
    <abund>27.3300000000</abund>
  </isotope>
  <isotope>
    <mass>107.9038940000</mass>
    <abund>26.4600000000</abund>
  </isotope>
  <isotope>
    <mass>109.9051690000</mass>
    <abund>11.7200000000</abund>
  </isotope>
</atom>
<atom>
  <name>Silver</name>
  <symbol>Ag</symbol>
  <isotope>
    <mass>106.9050950000</mass>
    <abund>51.8390000000</abund>
  </isotope>
  <isotope>
    <mass>108.9047540000</mass>
    <abund>48.1610000000</abund>
  </isotope>
</atom>
<atom>
  <name>Cadmium</name>
  <symbol>Cd</symbol>
  <isotope>
    <mass>105.9064610000</mass>
    <abund>1.2500000000</abund>
  </isotope>
  <isotope>
    <mass>107.9041860000</mass>
```

```
      <abund>0.8900000000</abund>
    </isotope>
    <isotope>
      <mass>109.9030010000</mass>
      <abund>12.4900000000</abund>
    </isotope>
    <isotope>
      <mass>110.9041820000</mass>
      <abund>12.8000000000</abund>
    </isotope>
    <isotope>
      <mass>111.9027610000</mass>
      <abund>24.1300000000</abund>
    </isotope>
    <isotope>
      <mass>112.9044010000</mass>
      <abund>12.2200000000</abund>
    </isotope>
    <isotope>
      <mass>113.9033610000</mass>
      <abund>28.7300000000</abund>
    </isotope>
    <isotope>
      <mass>115.9047580000</mass>
      <abund>7.4900000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Indium</name>
    <symbol>In</symbol>
    <isotope>
      <mass>112.9040560000</mass>
      <abund>4.2900000000</abund>
    </isotope>
    <isotope>
      <mass>114.9038750000</mass>
      <abund>95.7100000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Tin</name>
    <symbol>Sn</symbol>
    <isotope>
      <mass>111.9048230000</mass>
      <abund>0.9700000000</abund>
    </isotope>
    <isotope>
      <mass>113.9027810000</mass>
      <abund>0.6600000000</abund>
    </isotope>
```

```
  <isotope>
    <mass>114.9033440000</mass>
    <abund>0.3400000000</abund>
  </isotope>
  <isotope>
    <mass>115.9017430000</mass>
    <abund>14.5400000000</abund>
  </isotope>
  <isotope>
    <mass>116.9029540000</mass>
    <abund>7.6800000000</abund>
  </isotope>
  <isotope>
    <mass>117.9016070000</mass>
    <abund>24.2200000000</abund>
  </isotope>
  <isotope>
    <mass>118.9033100000</mass>
    <abund>8.5900000000</abund>
  </isotope>
  <isotope>
    <mass>119.9021990000</mass>
    <abund>32.5800000000</abund>
  </isotope>
  <isotope>
    <mass>121.9034400000</mass>
    <abund>4.6300000000</abund>
  </isotope>
  <isotope>
    <mass>123.9052710000</mass>
    <abund>5.7900000000</abund>
  </isotope>
</atom>
<atom>
  <name>Antimony</name>
  <symbol>Sb</symbol>
  <isotope>
    <mass>120.9038240000</mass>
    <abund>57.2100000000</abund>
  </isotope>
  <isotope>
    <mass>122.9042220000</mass>
    <abund>42.7900000000</abund>
  </isotope>
</atom>
<atom>
  <name>Tellurium</name>
  <symbol>Te</symbol>
  <isotope>
    <mass>119.9040210000</mass>
```

```
      <abund>0.0900000000</abund>
    </isotope>
    <isotope>
      <mass>121.9030550000</mass>
      <abund>2.5500000000</abund>
    </isotope>
    <isotope>
      <mass>122.9042780000</mass>
      <abund>0.8900000000</abund>
    </isotope>
    <isotope>
      <mass>123.9028250000</mass>
      <abund>4.7400000000</abund>
    </isotope>
    <isotope>
      <mass>124.9044350000</mass>
      <abund>7.0700000000</abund>
    </isotope>
    <isotope>
      <mass>125.9033100000</mass>
      <abund>18.8400000000</abund>
    </isotope>
    <isotope>
      <mass>127.9044640000</mass>
      <abund>31.7400000000</abund>
    </isotope>
    <isotope>
      <mass>129.9062290000</mass>
      <abund>34.0800000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Iodine</name>
    <symbol>I</symbol>
    <isotope>
      <mass>126.9044770000</mass>
      <abund>100.0000000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Xenon</name>
    <symbol>Xe</symbol>
    <isotope>
      <mass>123.9061200000</mass>
      <abund>0.0900000000</abund>
    </isotope>
    <isotope>
      <mass>125.9042810000</mass>
      <abund>0.0900000000</abund>
    </isotope>
```

```
  <isotope>
    <mass>127.9035310000</mass>
    <abund>1.9200000000</abund>
  </isotope>
  <isotope>
    <mass>128.9047800000</mass>
    <abund>26.4400000000</abund>
  </isotope>
  <isotope>
    <mass>129.9035100000</mass>
    <abund>4.0800000000</abund>
  </isotope>
  <isotope>
    <mass>130.9050760000</mass>
    <abund>21.1800000000</abund>
  </isotope>
  <isotope>
    <mass>131.9041480000</mass>
    <abund>26.8900000000</abund>
  </isotope>
  <isotope>
    <mass>133.9053950000</mass>
    <abund>10.4400000000</abund>
  </isotope>
  <isotope>
    <mass>135.9072190000</mass>
    <abund>8.8700000000</abund>
  </isotope>
</atom>
<atom>
  <name>Caesium</name>
  <symbol>Cs</symbol>
  <isotope>
    <mass>132.9054330000</mass>
    <abund>100.0000000000</abund>
  </isotope>
</atom>
<atom>
  <name>Barium</name>
  <symbol>Ba</symbol>
  <isotope>
    <mass>129.9062770000</mass>
    <abund>0.1060000000</abund>
  </isotope>
  <isotope>
    <mass>131.9050420000</mass>
    <abund>0.1010000000</abund>
  </isotope>
  <isotope>
    <mass>133.9044900000</mass>
```

```
      <abund>2.4170000000</abund>
    </isotope>
    <isotope>
      <mass>134.9056680000</mass>
      <abund>6.5920000000</abund>
    </isotope>
    <isotope>
      <mass>135.9045560000</mass>
      <abund>7.8540000000</abund>
    </isotope>
    <isotope>
      <mass>136.9058160000</mass>
      <abund>11.2320000000</abund>
    </isotope>
    <isotope>
      <mass>137.9052360000</mass>
      <abund>71.6980000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Lanthanium</name>
    <symbol>La</symbol>
    <isotope>
      <mass>137.9071140000</mass>
      <abund>0.0900000000</abund>
    </isotope>
    <isotope>
      <mass>138.9063550000</mass>
      <abund>99.9100000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Gold</name>
    <symbol>Au</symbol>
    <isotope>
      <mass>196.9665600000</mass>
      <abund>100.0000000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Mercury</name>
    <symbol>Hg</symbol>
    <isotope>
      <mass>195.9658120000</mass>
      <abund>0.1500000000</abund>
    </isotope>
    <isotope>
      <mass>197.9667600000</mass>
      <abund>9.9700000000</abund>
    </isotope>
```

```
    <isotope>
      <mass>198.9682690000</mass>
      <abund>16.8700000000</abund>
    </isotope>
    <isotope>
      <mass>199.9683160000</mass>
      <abund>23.1000000000</abund>
    </isotope>
    <isotope>
      <mass>200.9702930000</mass>
      <abund>13.1800000000</abund>
    </isotope>
    <isotope>
      <mass>201.9706320000</mass>
      <abund>29.8600000000</abund>
    </isotope>
    <isotope>
      <mass>203.9734810000</mass>
      <abund>6.8700000000</abund>
    </isotope>
  </atom>
  <atom>
    <name>Lead</name>
    <symbol>Pb</symbol>
    <isotope>
      <mass>203.9730370000</mass>
      <abund>1.4000000000</abund>
    </isotope>
    <isotope>
      <mass>205.9744550000</mass>
      <abund>24.1000000000</abund>
    </isotope>
    <isotope>
      <mass>206.9758850000</mass>
      <abund>22.1000000000</abund>
    </isotope>
    <isotope>
      <mass>207.9766410000</mass>
      <abund>52.4000000000</abund>
    </isotope>
  </atom>
</atomdefdata>
<polchemdefdata>
  <name>protein-1-letter</name>
  <leftcap>+H</leftcap>
  <rightcap>+OH</rightcap>
  <codelen>1</codelen>
  <ionizerule>
    <formula>+H</formula>
    <charge>1</charge>
```

```
    <level>1</level>
  </ionizerule>
  <monomers>
    <mnm>
      <name>Glycine</name>
      <code>G</code>
      <formula>C2H3NO</formula>
    </mnm>
    <mnm>
      <name>Alanine</name>
      <code>A</code>
      <formula>C3H5NO</formula>
    </mnm>
    <mnm>
      <name>Valine</name>
      <code>V</code>
      <formula>C5H9NO</formula>
    </mnm>
    <mnm>
      <name>Leucine</name>
      <code>L</code>
      <formula>C6H11NO</formula>
    </mnm>
    <mnm>
      <name>Isoleucine</name>
      <code>I</code>
      <formula>C6H11NO</formula>
    </mnm>
    <mnm>
      <name>Serine</name>
      <code>S</code>
      <formula>C3H5NO2</formula>
    </mnm>
    <mnm>
      <name>Threonine</name>
      <code>T</code>
      <formula>C4H7NO2</formula>
    </mnm>
    <mnm>
      <name>Cysteine</name>
      <code>C</code>
      <formula>C3H5NOS</formula>
    </mnm>
    <mnm>
      <name>Methionine</name>
      <code>M</code>
      <formula>C5H9NOS</formula>
    </mnm>
    <mnm>
      <name>Arginine</name>
```

```
  <code>R</code>
  <formula>C6H12N4O</formula>
</mnm>
<mnm>
  <name>Lysine</name>
  <code>K</code>
  <formula>C6H12N2O</formula>
</mnm>
<mnm>
  <name>Aspartate</name>
  <code>D</code>
  <formula>C4H5NO3</formula>
</mnm>
<mnm>
  <name>Glutamate</name>
  <code>E</code>
  <formula>C5H7NO3</formula>
</mnm>
<mnm>
  <name>Asparagine</name>
  <code>N</code>
  <formula>C4H6N2O2</formula>
</mnm>
<mnm>
  <name>Glutamine</name>
  <code>Q</code>
  <formula>C5H8N2O2</formula>
</mnm>
<mnm>
  <name>Tryptophan</name>
  <code>W</code>
  <formula>C11H10N2O</formula>
</mnm>
<mnm>
  <name>Phenylalanine</name>
  <code>F</code>
  <formula>C9H9N1O</formula>
</mnm>
<mnm>
  <name>Tyrosine</name>
  <code>Y</code>
  <formula>C9H9N1O2</formula>
</mnm>
<mnm>
  <name>Histidine</name>
  <code>H</code>
  <formula>C6H7N3O</formula>
</mnm>
<mnm>
  <name>Proline</name>
```

```
      <code>P</code>
      <formula>C5H7N1O1</formula>
    </mnm>
  </monomers>
  <modifs>
    <mdf>
      <name>OnAnyMonomer</name>
      <formula>+Hec</formula>
      <targets>*</targets>
    </mdf>
    <mdf>
      <name>OnNoMonomer</name>
      <formula>+Hec</formula>
      <targets>!</targets>
    </mdf>
    <mdf>
      <name>Phosphorylation</name>
      <formula>-H+H2PO3</formula>
      <targets>;S;T;Y;</targets>
    </mdf>
    <mdf>
      <name>Sulphation</name>
      <formula>-H+HSO3</formula>
      <targets>;S;T;Y;</targets>
    </mdf>
    <mdf>
      <name>Oxidation</name>
      <formula>+O</formula>
      <targets>;M;</targets>
    </mdf>
    <mdf>
      <name>Acetylation</name>
      <formula>-H+C2H3O</formula>
      <targets>;K;</targets>
    </mdf>
    <mdf>
      <name>AmidationGlu</name>
      <formula>-OH+NH2</formula>
      <targets>;E;</targets>
    </mdf>
    <mdf>
      <name>AmidationAsp</name>
      <formula>-OH+NH2</formula>
      <targets>;D;</targets>
    </mdf>
    <mdf>
      <name>SulfideBond</name>
      <formula>-H2</formula>
      <targets>;C;</targets>
    </mdf>
```

```
  <mdf>
    <name>Carbamylation</name>
    <formula>-H+CONH2</formula>
    <targets>;K;</targets>
  </mdf>
  <mdf>
    <name>CarboxyMethylation</name>
    <formula>-H+CH2COOH</formula>
    <targets>;C;</targets>
  </mdf>
  <mdf>
    <name>ProtonLoss</name>
    <formula>-H</formula>
    <targets>;C;</targets>
  </mdf>
  <mdf>
    <name>Chromo-O</name>
    <formula>-O</formula>
    <targets>;T;</targets>
  </mdf>
  <mdf>
    <name>Chromo-H</name>
    <formula>-H</formula>
    <targets>;G;</targets>
  </mdf>
  <mdf>
    <name>Chromo-H3</name>
    <formula>-H3</formula>
    <targets>;Y;</targets>
  </mdf>
</modifs>
<crosslinkers>
  <clk>
    <name>DisulfideBond</name>
    <formula>+Nul</formula>
    <modifname>ProtonLoss</modifname>
    <modifname>ProtonLoss</modifname>
  </clk>
</crosslinkers>
<cleavespecs>
  <cls>
    <name>CyanogenBromide</name>
    <pattern>M/</pattern>
    <clr>
      <name>Homoseryl</name>
      <re-mnm-code>M</re-mnm-code>
      <re-formula>-CH2S+O</re-formula>
    </clr>
  </cls>
  <cls>
```

```
      <name>Trypsin</name>
      <pattern>K/;R/;-K/P</pattern>
    </cls>
    <cls>
      <name>Chymotrypsin</name>
      <pattern>W/;V/</pattern>
    </cls>
    <cls>
      <name>EndoLysC</name>
      <pattern>K/</pattern>
    </cls>
    <cls>
      <name>EndoAspN</name>
      <pattern>/D</pattern>
    </cls>
    <cls>
      <name>GluC</name>
      <pattern>E/</pattern>
    </cls>
  </cleavespecs>
  <fragspecs>
    <fgs>
      <name>a</name>
      <end>LE</end>
      <formula>-C1O1</formula>
      <fgr>
        <name>a-fgr-1</name>
        <formula>+H2O0</formula>
        <prev-mnm-code>E</prev-mnm-code>
        <curr-mnm-code>D</curr-mnm-code>
        <next-mnm-code>F</next-mnm-code>
        <comment>comment here!</comment>
      </fgr>
      <fgr>
        <name>a-fgr-2</name>
        <formula>+H1O0</formula>
        <prev-mnm-code>F</prev-mnm-code>
        <curr-mnm-code>D</curr-mnm-code>
        <next-mnm-code>E</next-mnm-code>
        <comment>comment here!</comment>
      </fgr>
    </fgs>
    <fgs>
      <name>b</name>
      <end>LE</end>
      <formula>-H0</formula>
    </fgs>
    <fgs>
      <name>c</name>
      <end>LE</end>
```

```
        <formula>+N1H2+H1</formula>
        <comment>that's just a comment</comment>
      </fgs>
      <fgs>
        <name>z</name>
        <end>RE</end>
        <formula>-N1H1</formula>
        <comment>Not in CID high En. frag</comment>
      </fgs>
      <fgs>
        <name>y</name>
        <end>RE</end>
        <formula>+H2</formula>
      </fgs>
      <fgs>
        <name>x</name>
        <end>RE</end>
        <formula>+C1O1</formula>
        <fgr>
          <name>x-fgr-1</name>
          <formula>+H100</formula>
          <prev-mnm-code>E</prev-mnm-code>
          <curr-mnm-code>D</curr-mnm-code>
          <next-mnm-code>F</next-mnm-code>
          <comment>comment here!</comment>
        </fgr>
        <fgr>
          <name>x-fgr-2</name>
          <formula>+H200</formula>
          <prev-mnm-code>F</prev-mnm-code>
          <curr-mnm-code>D</curr-mnm-code>
          <next-mnm-code>E</next-mnm-code>
          <comment>comment here!</comment>
        </fgr>
      </fgs>
      <fgs>
        <name>imm</name>
        <end>NE</end>
        <formula>-C1O1+H1</formula>
      </fgs>
    </fragspecs>
  </polchemdefdata>
</polchemdefinition>
```

One Example Polymer Sequence File

The sequence below is not biologically relevant. It is only shown here for didactic purposes. Indeed, no one has never seen a phosphorylated Glycyl residue!

```
<?xml version="1.0"?>
<!-- DTD for polymer sequences, used by the
'massXpert' mass spectrometry application.
Copyright 2006,2007 Filippo Rusconi - Licensed under the GNU GPL -->
<!DOCTYPE polseqdata [
<!ELEMENT polseqdata (polchemdef_name,name,code,author,datetime,polseq,le_modif?,re_m
<!ELEMENT polchemdef_name (#PCDATA)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT code (#PCDATA)>
<!ELEMENT author (#PCDATA)>
<!ELEMENT datetime (#PCDATA)>
<!ELEMENT polseq (codes|monomer)*>
<!ELEMENT le_modif (#PCDATA)>
<!ELEMENT re_modif (#PCDATA)>
<!ELEMENT codes (#PCDATA)>
<!ELEMENT monomer (code, prop*)>
<!ELEMENT prop (name, data+)>
<!ATTLIST data type (str | int | dbl) "str">
<!ELEMENT data (#PCDATA)>
]>
<polseqdata>
  <polchemdef_name>protein-1-letter</polchemdef_name>
  <name>NOT_SET</name>
  <code>NOT_SET</code>
  <author>rusconi</author>
  <datetime>2007-05-20:43:40</datetime>
  <polseq>
    <codes>YG</codes>
    <monomer>
      <code>G</code>
      <prop>
        <name>MODIF</name>
        <data>Phosphorylation</data>
      </prop>
    </monomer>
    <codes>FL</codes>
  </polseq>
  <le_modif>Acetylation</le_modif>
  <re_modif>Acetylation</re_modif>
</polseqdata>
```

The pka_ph_pi.xml File

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<!-- DTD for polymer elements' pka data, used by the
'massXpert' of mass spectrometry program.
Copyright 2003 2007 Filippo Rusconi - Licensed under the GNU GPL -->
<!DOCTYPE pkaphpidata [
<!ELEMENT pkaphpidata (monomers,modifs*)>
<!ELEMENT monomers (monomer*)>
<!ELEMENT modifs (modif*)>
<!ELEMENT monomer (code,mnmchemgroup*)>
<!ELEMENT modif (name,mdfchemgroup*)>
<!ELEMENT mnmchemgroup (name,pka,acidcharged,polrule,chemgrouprule*)>
<!ELEMENT mdfchemgroup (name,pka,acidcharged)>
<!ELEMENT chemgrouprule (entity,name,outcome)>
<!ELEMENT pka (#PCDATA)>
<!ELEMENT value (#PCDATA)>
<!ELEMENT code (#PCDATA)>
<!ELEMENT outcome (#PCDATA)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT entity (#PCDATA)>
<!ELEMENT acidcharged (#PCDATA)>
<!ELEMENT polrule (#PCDATA)>
]>
<pkaphpidata>
  <monomers>
    <monomer>
      <code>A</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>C</code>
      <mnmchemgroup>
```

```
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>Lateral SH2</name>
<pka>8.3</pka>
<acidcharged>FALSE</acidcharged>
<polrule>never_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>D</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.36</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>Lateral COOH</name>
<pka>3.65</pka>
<acidcharged>FALSE</acidcharged>
<polrule>never_trapped</polrule>
<chemgrouprule>
  <entity>MONOMER_MODIF</entity>
  <name>AmidationAsp</name>
```

```
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>E</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.36</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>Lateral COOH</name>
<pka>4.25</pka>
<acidcharged>FALSE</acidcharged>
<polrule>never_trapped</polrule>
<chemgrouprule>
  <entity>MONOMER_MODIF</entity>
  <name>AmidationGlu</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>F</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
```

```
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>G</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>H</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.36</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>In-ring NH+</name>
<pka>6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>never_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>I</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
```

```
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>K</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.36</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>Lateral NH2</name>
<pka>10.53</pka>
<acidcharged>TRUE</acidcharged>
<polrule>never_trapped</polrule>
<chemgrouprule>
  <entity>MONOMER_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>L</code>
      <mnmchemgroup>
```

```
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>M</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>N</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
```

```
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>P</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>Q</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>R</code>
      <mnmchemgroup>
```

```
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.36</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>Lateral guanidinium</name>
<pka>12.48</pka>
<acidcharged>TRUE</acidcharged>
<polrule>never_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>S</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>Lateral alcohol</name>
<pka>13</pka>
<acidcharged>FALSE</acidcharged>
<polrule>never_trapped</polrule>
<chemgrouprule>
  <entity>MONOMER_MODIF</entity>
  <name>Phosphorylation</name>
```

```
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>T</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>Lateral alcohol</name>
<pka>13</pka>
<acidcharged>FALSE</acidcharged>
<polrule>never_trapped</polrule>
<chemgrouprule>
  <entity>MONOMER_MODIF</entity>
  <name>Phosphorylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
    </monomer>
   <monomer>
      <code>V</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
```

```
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>W</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.35</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
    </monomer>
    <monomer>
      <code>Y</code>
      <mnmchemgroup>
        <name>N-term NH2</name>
<pka>9.6</pka>
<acidcharged>TRUE</acidcharged>
<polrule>left_trapped</polrule>
<chemgrouprule>
  <entity>LE_PLM_MODIF</entity>
  <name>Acetylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>C-term COOH</name>
<pka>2.36</pka>
<acidcharged>FALSE</acidcharged>
<polrule>right_trapped</polrule>
      </mnmchemgroup>
      <mnmchemgroup>
        <name>Lateral phenol</name>
<pka>10.1</pka>
<acidcharged>FALSE</acidcharged>
<polrule>never_trapped</polrule>
<chemgrouprule>
```

```
  <entity>MONOMER_MODIF</entity>
  <name>Phosphorylation</name>
  <outcome>LOST</outcome>
</chemgrouprule>
      </mnmchemgroup>
    </monomer>
  </monomers>
  <modifs>
    <modif>
      <name>Phosphorylation</name>
      <mdfchemgroup>
        <name>none_set</name>
<pka>1.2</pka>
<acidcharged>FALSE</acidcharged>
      </mdfchemgroup>
      <mdfchemgroup>
        <name>none_set</name>
<pka>6.5</pka>
<acidcharged>FALSE</acidcharged>
      </mdfchemgroup>
    </modif>
  </modifs>
</pkaphpidata>
```

GNU General Public License Text

GNU GENERAL PUBLIC LICENSE
Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program–to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

Terms and Conditions

0. Definitions.

   "This License" refers to version 3 of the GNU General Public License.

   "Copyright" also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

   "The Program" refers to any copyrightable work licensed under this License. Each licensee is addressed as "you". "Licensees" and "recipients" may be individuals or organizations.

   To "modify" a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a "modified version" of the earlier work or a work "based on" the earlier work.

   A "covered work" means either the unmodified Program or a work based on the Program.

   To "propagate" a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

   To "convey" a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

   An interactive user interface displays "Appropriate Legal Notices" to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

   The "source code" for a work means the preferred form of the work for making modifications to it. "Object code" means any non-source form of a work.

A "Standard Interface" means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The "System Libraries" of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A "Major Component", in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The "Corresponding Source" for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

   No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

   When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

   You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

   You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

   You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

   (a) The work must carry prominent notices stating that you modified it, and giving a relevant date.

   (b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to "keep intact all notices".

   (c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

   (d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an "aggregate" if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

   You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

   (a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

   (b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

   (c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

   (d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

   (e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A "User Product" is either (1) a "consumer product", which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, "normally used" refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

"Installation Information" for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

"Additional permissions" are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as

though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

(a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

(b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

(c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

(d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

(e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

(f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered "further restrictions" within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

   You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

   However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

   Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

   Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

   You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

    Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

    An "entity transaction" is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

    You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose

a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A "contributor" is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's "contributor version".

A contributor's "essential patent claims" are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, "control" includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a "patent license" is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To "grant" such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. "Knowingly relying" means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is "discriminatory" if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-

exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

    If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

    Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

    The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License "or any later version" applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

    If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public state-

ment of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

End of Terms and Conditions

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion

of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

> <one line to give the program's name and a brief idea of what it does.>
>
> Copyright (C) <year> <name of author>
>
> This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.
>
> This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
>
> You should have received a copy of the GNU General Public License along with this program. If not, see http://www.gnu.org/licenses/.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

> <program> Copyright (C) <year> <name of author>
>
> This program comes with ABSOLUTELY NO WARRANTY; for details type `show w`. This is free software, and you are welcome to redistribute it under certain conditions; type `show c` for details.

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an "about box".

You should also get your employer (if you work as a programmer) or school, if any, to sign a "copyright disclaimer" for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see http://www.gnu.org/licenses/.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read http://www.gnu.org/philosophy/why-not-lgpl.html.

Index