



Structure Validation and Normalization

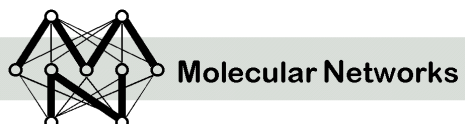
User Manual

Version 1.0

for program version 1.0 (or higher)

November 2003

Molecular Networks GmbH – Computerchemie
Nägelsbachstraße 25, 91052 Erlangen, Germany



Molecular Networks GmbH

Computerchemie

Nägelsbachstr. 25

91052 Erlangen

Germany

Phone: +49-(0)9131-815668

Fax: +49-(0)9131-815669

Email: info@mol-net.de

WWW: www.mol-net.de

This document is copyright © 2002 by Molecular Networks GmbH Computerchemie. All rights reserved. Except as permitted under the terms of the Software Licensing Agreement of Molecular Networks GmbH Computerchemie, no part of this publication may be reproduced or distributed in any form or by any means or stored in a database retrieval system without the prior written permission of Molecular Networks GmbH Computerchemie.

The software described in this document was originally designed and implemented for the CACTVS system by W. D. Ihlenfeldt. It is not part of the standard CACTVS toolkit distribution; it is furnished under a license and may be used and copied only in accordance with the terms of such license. For licensing this software please contact Molecular Networks GmbH, Nägelsbachstr. 25, 91052 Erlangen, Germany.

Product names and company names may be trademarks or registered trademarks of their respective owners, in the Federal Republic of Germany and other countries. All rights reserved.

Table of Contents

1.	General Information about MN.CHECK	5
2.	Installation	6
2.1.	Requirements.....	6
2.2.	Installation Steps for UNIX Operating Systems (IRIX, Solaris, Linux)	6
2.3.	Installation Steps for Microsoft Windows Operating Systems (NT4/2000/XP).....	6
3.	Uninstallation.....	6
3.1.	Uninstallation Steps for UNIX Operating Systems (IRIX, Solaris, Linux).....	6
3.2.	Uninstallation Steps for Microsoft Windows Operating Systems (NT4/2000/XP) ..	6
4.	Problems and Help!.....	7
5.	Release Notes	8
5.1.	Version 1.0	8
6.	Getting Started.....	9
6.1.	UNIX operating systems.....	9
6.2.	Microsoft Windows operating systems.....	9
7.	Program Use	10
7.1.	Synopsis	10
7.2.	General Program Features.....	11
7.3.	Supported file formats for input files.....	11
7.4.	Program Features in More Detail	11
	-format <abbreviation of the output format>	11
	-outfile <filename.extension>	12
	-directory <dirname>	12
	-feedback 0/n	12
	-pedantic	13
	-skiperrors	13
	-unique	13
	-hydrogens stripall/strip/asis/add/hetero/striplight	14
	-nitrostyle ionic/asis/penta	14
	-chargebalance	14
	-uncharge.....	15
	-desalt	15
	-saltfile <filename>.....	16
	-stat	17
	-version	17
	-h or -help.....	17
8.	Extended Features Only Available for the UNIX Operating Systems.....	18
9.	Frequently Asked Questions (FAQ)	19
10.	Error Messages	19
11.	Known Problems and Limitations	19
12.	Technical Support	20
	The MN.CHECK Web Site	20

	Reporting Problems.....	20
	Updates.....	20
	Contact Information	20
13.	Report Form.....	21
14.	Index.....	22

1. General Information about MN.CHECK

Like in the laboratory, compounds can be electronically specified in many different ways. Structures can be stored in different ionization states, in salt form and with solvent molecules. For some applications such as ligand-based and structure-based modeling or when intrinsic physicochemical properties of compounds are computed, this variability is undesirable. In this case, the state of each structure needs to be first normalized. In addition, especially when obtained from different external sources, the integrity of each structure needs also to be checked.

MN.CHECK performs high throughput structure integrity checks and can be used to normalize the state of each compound. Integrity check is performed on atomic valence, hybridization state, and ionization state of a molecule. The state of each structure can be controlled by specifying its ionization state, by removing salts and solvents or by adding missing hydrogen atoms.

The program MN.CHECK

- adds or strips hydrogen atoms in various modes
- can produce a neutral form of the structure
- controls ionization state of nitro groups (ionic or uncharged)
- processes datasets with 99.9% conversion rate
- handles datasets of hundreds of thousands of chemical structures
- supports the SDF and SMILES file format for reading input files and saving output files

2. Installation

2.1. Requirements

MN.CHECK is available for common UNIX platforms (x86 Linux, Sun Solaris, SGI IRIX, DEC AlphaStation). It is also available for Microsoft Windows NT4/2000/XP.

The program runs in a batch mode.

2.2. Installation Steps for UNIX Operating Systems (IRIX, Solaris, Linux)

- 1.) Create a subdirectory, e.g., `mn_check`
(for system administrators when installing software locally, e.g. `/usr/local/bin/mn_check`).
- 2.) Copy the file `mn_check_<version>.<os>.gz` to the subdirectory `mn_check`
- 3.) Unpack the distribution by executing the `gunzip` command:
`gunzip mn_check_<version>.<os>.gz`
- 4.) Rename the file `mn_check_<version>.<os>` to `mn_check`.
Please note: `mn_check_<version>.<os>` is a binary file.
- 5.) Add the `mn_check` subdirectory name to the environment variable `PATH` in your `.login` or `.cshrc` files (`.profile` or `.bashrc`).

Launch MN.CHECK with the command

```
mn_check -version or /usr/local/bin/mn_check/mn_check -version
```

2.3. Installation Steps for Microsoft Windows Operating Systems (NT4/2000/XP)

Although administrator privileges are not necessary, we recommend logging in as administrator. Double-click on the executable setup program and follow the instructions on the screen.

After successful installation there is no need to reboot your PC.

3. Uninstallation

3.1. Uninstallation Steps for UNIX Operating Systems (IRIX, Solaris, Linux)

Log in as root and delete the file `mn_check` in your installation directory carefully (default path during installation was `/usr/local/bin/mn_check/`).

3.2. Uninstallation Steps for Microsoft Windows Operating Systems (NT4/2000/XP)

Log in as administrator, launch the uninstaller and follow the on-screen instructions.

4. Problems and Help!

If you have any difficulties with the installation of MN.CHECK or if any problems occur while running MN.CHECK, please send all your inquiries to the following address:

Molecular Networks GmbH Computerchemie
Nägelsbachstr. 25
91052 Erlangen
Germany,

or contact us by email
or by fax

support@mol-net.de,
+49-(0)9131 - 81 56 69.

Please mention the program version of MN.CHECK (`mn_check -version`), include your input file and the output file on an MS/DOS diskette (3½”) or send it to us by email. These files will help us to analyze the problem; if your system displays any error messages, please add them to your report.

You can also use the report form at the end of this manual.

5. Release Notes

5.1. Version 1.0

First release of MN.CHECK

6. Getting Started

6.1. UNIX operating systems

The example file `nitrobenzene.sdf` submitted with the distribution contains the structure information of the molecule nitrobenzene in SD format. Copy this example file into your working directory and type the following command:

```
mn_check -hydrogen addall -nitrostyle ionic nitrobenzene.sdf
```

MN.CHECK now creates the output file named `nitrobenzene.mdl` written to the same directory where the file `nitrobenzene.sdf` is located. The resulting molecule now contains hydrogen atoms and an ionic nitro group representation. Figure 1 shows the depiction of the input and output file.

6.2. Microsoft Windows operating systems

The example file `nitrobenzene.sdf` submitted with the distribution contains the structure information of the molecule nitrobenzene in SD format. Copy this example file into your working directory and open a DOS shell. Change the working directory to the directory where you installed `mn_check` by using the `cd` command then type the following command:

```
mn_check -hydrogen addall -nitrostyle ionic nitrobenzene.sdf
```

MN.CHECK now creates the output file named `nitrobenzene.mdl` written to the same directory where the file `nitrobenzene.sdf` is located. The resulting molecule now contains hydrogen atoms and an ionic nitro group representation. Figure 1 shows the depiction of the input and output file.

If you have no permission writing to the directory in which the program was installed, set the **-directory** option for specifying another directory:

```
UNIX: mn_check -hydrogen addall -nitrostyle ionic -directory  
/tmp nitrobenzene.sdf
```

```
Windows: mn_check -hydrogen addall -nitrostyle ionic -directory  
C:/temp nitrobenzene.sdf
```



Figure 1: 2D depictions of the input and output file

7. Program Use

7.1. Synopsis

The general synopsis for using MN.CHECK is:

```
mn_check [ -option(s) ] [ infile ]
```

An overview of the various options is given in Table 1 and in a more detailed one in the following chapter. `Infile` is the input file name. If no file name is given, the program reads from standard input.

[-chargebalance]	produces a neutral, uncharged form of the structure
[-desalt]	removes counter ions from the input file
[-directory dirname]	specifies the output directory
[-feedback 0/n]	prints a control message after processing a block of n items
[-format fmt]	specifies the output format name
[-h] or [-help]	shows a brief help message about the usage of the program
[-hydrogens stripall/strip/asis/addall/addhetero/striplight]	adapt the set of hydrogen atoms
[-nitrostyle ionic/asis/penta]	controls the encoding of nitro groups
[-outfile filename]	defines the name of the output file
[-pedantic]	produces error messages for various non-critical errors in the input file
[-saltfile filename]	specifies a file name containing fragments that should be discarded when removing counterions
[-skiperrors]	resumes processing after recording a syntax error in the input file
[-stat]	writes statistical information about the number of successfully processed records and processing failures
[-uncharge]	neutralizes a compound, e.g. removing excess protons or add them
[-unique]	removes duplicates
[-version]	prints version and licensing information

Table 1: Overview of all options

Executing the program without any option will write all readable structures to the output file without adding or stripping hydrogen atoms and without modifying the style of nitro groups.

7.2. General Program Features

The file type of the input file is automatically recognized. If no input file is specified, or the file name „-“ is used, the program reads from standard input.

If you are running MN.CHECK under a UNIX operating system, there are some more features reading input files (see chapter 8 “Extended Features Only Available for the UNIX Operating Systems” for more details).

The file name of the output file is either explicitly set with the **-outfile** option or automatically derived from the input file and the given output file format (**-format**). The special filename `stdout` can be used to direct output to the standard output channel.

7.3. Supported file formats for input files

The program will automatically detect the file format of the input files. Three standard exchange formats are supported. Thus, there is no need for a parameter specifying the input format.

The supported file formats are listed in the table below.

Full Format Name	Default Input-Extension	Read	Comment
MDL Molfile	mol	Yes	
MDL SDF	sdf	Yes	
SMILES	smi	Yes	

Table 2: Overview of the supported input file formats

7.4. Program Features in More Detail

-format <abbreviation of the output format>

The parameter **format** is specified for selecting the output format.

The supported file formats are listed in the table below.

Full Format Name	Default Output-Extension	Write	Comment
MDL Molfile	mol	Yes	
MDL SDF	mdl	Yes	
SMILES	smi	Yes	

Table 3: Overview of the supported output file formats

Please use the abbreviation of the format names for specifying your desired file format using the **-format** option. If no output file is specified, the output has the same name (but with an updated suffix) and is written in the same directory as the input file. The extension of the resulting output file is sometimes different to the given abbreviation (see the previous table).

If the output file is specified explicitly with the **-outfile** parameter, this file name including the chosen suffix, will be used.

Default value:

Parameter without a default value

Example:

Generating a MDL SD-file:

```
mn_check -format sdf ./examples/nitrobenzene.sdf
```

Remarks:

If this option is not used, an attempt is made to guess the output file format from its suffix by the given **-outfile** parameter.

-outfile <filename.extension>

The parameter **outfile** defines the name of the output file. MN.CHECK automatically recognizes the desired output format, thus in most cases it is not necessary to specify the output format.

If you are using MN.CHECK on a UNIX operating system the output file name can also be an anonymous ftp URL.

Default value:

Parameter without a default value

Example:

Generating a MDL SD-file and adding all hydrogen atoms:

```
mn_check -outfile nitrobenzene_H.sdf -hydrogen addall
./examples/nitrobenzene.sdf
```

-directory <dirname>

This parameter sets the target directory. If the directory does not yet exist, it will be created.

Default value:

The directory of the output files is the same as of the corresponding input files, or the current directory, if the input file names do not contain directory information.

Example:

Generating a MDL SD-file saved in a given directory:

```
UNIX: mn_check -outfile nitrobenzene_H.sdf -directory /tmp
-hydrogen addall ./examples/nitrobenzene.sdf
```

```
Windows: mn_check -outfile nitrobenzene_H.sdf -directory C:/Temp
-hydrogen addall ./examples/nitrobenzene.sdf
```

-feedback 0/n

If the parameter **feedback** is set to a value larger than zero, a control message is printed after processing a block of n structures. The current record number and the object name are printed on the standard error channel. Only structures which are actually written out are counted.

Default value:

It is not active by default.

Example:

```
Generating a MDL SD-file printing dots for every one hundred records:  
mn_check -outfile maybridge_checked.sdf -feedback 100  
./examples/maybridge.sdf
```

-pedantic

If this parameter is set, error messages are produced for various non-critical errors in input files.

Default value:

By default this option is off.

Example:

```
Generating a MDL SD-file:  
mn_check -outfile maybridge_checked.sdf -pedantic  
./examples/maybridge.sdf
```

-skiperrors

If this parameter is set, records which could not be read due to syntax errors in the input file will be skipped and processing resumes with the next record after the defective entry.

Default value:

By default, the processing of such a file is stopped when an error occurs.

Example:

```
mn_check -outfile alkanes1_12_minus7.sdf -skiperrors  
./examples/alkanes1_12_error7.sdf
```

Remarks:

Note that it is difficult in some file formats to re-synchronize to the beginning of a new record if an error was encountered, so sometimes correct records following the corrupted entry may still not be readable. Also, in extreme cases a faulty record among the first few records may prevent processing altogether, if the problem prevents the file format recognition routines from working properly. Simple ASCII file formats such as SMILES or SDF are the most robust to use with this feature.

-unique

If this parameter is set, records which were already encountered in any file processed during the current run are discarded. Stereo isomers are considered as different compounds.

Tautomers are always considered as structurally different from the standard form.

Default value:

This parameter is generally not set.

Example:

```
Generating a MDL SD-file without duplicates (do not distinguish between  
stereoisomers):  
mn_check -outfile unique.sdf -unique ./examples/not_unique.sdf  
input file: four structures (methane, ethane, propane, ethane)
```

output file: three structures (methane, ethane, propane)

-hydrogens stripall/strip/asis/add/hetero/striplight

The parameter **hydrogens** can adapt the set of hydrogen atoms. This option can be used to remove all hydrogen atoms (*stripall*), all hydrogen atoms which are usually not drawn in structure plots (*strip*), keep them as they are (*asis*), or add all hydrogen atoms as required by the valency rules (*add*) or just to those positions where they are usually drawn, but remove them from other locations such as hetero atoms (*hetero*). If isotope atoms of hydrogen (deuterium and tritium) should not be removed choose the parameter *striplight*.

Default value:

asis (keep the hydrogens as they are)

Example:

Generating a MDL SD-file without any hydrogen atom:

```
mn_check -outfile nitrobenzene_H.sdf -hydrogens addall
./examples/nitrobenzene.sdf
```

-nitrostyle ionic/asis/penta

This parameter controls the encoding of nitro groups in the output file.

If the parameter is set to *ionic* all nitro groups and similar functional units are re-coded as charge pairs (with a tetravalent, positively charged nitrogen atom, and a negatively charged oxygen atom or another ligand atom). If, however, the parameter is set to *penta* the uncharged variant with an octet expansion on the nitrogen atom will be written to the output file. If the option is set to *asis*, no processing takes place.

Default value:

asis (keep the nitrostyle as it is)

Example:

Generating a MDL SD-file with ionic nitro groups:

```
mn_check -outfile nitrobenzene_ionic.sdf -nitrostyle ionic
./examples/nitrobenzene.sdf
```

Remarks:

Similar functional groups are processed in the same way as nitro groups.

-chargebalance

If this parameter is set, the program attempts to produce a neutral, uncharged form of the structure.

Default value:

This flag is deactivated by default.

Example:

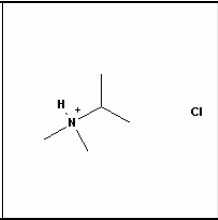
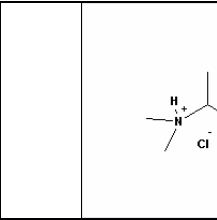
Generating a MDL SD-file writing an uncharged form of the structure:

```
mn_check -outfile hydrochloride.sdf -chargebalance
./examples/wrong_hydrochloride.sdf
```

Remarks:

This option does not have an effect on data files which contain either explicit atom charges, or a global ensemble charge.

Pictures:

	
wrongly coded hydrogenchloride	result of the used chargebalance option Note, that the chloride now has a negative charge.

-uncharge

If this parameter is set, the program will attempt to neutralize a compound, for example by removing excess protons or adding them to negatively charged atoms.

Default value:

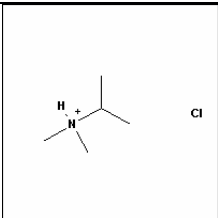
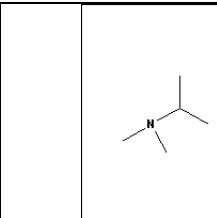
This flag is deactivated by default.

Example:

Generating a MDL SD-file writing a structure file after neutralizing a compound:

```
mn_check -outfile uncharged.sdf -uncharge
./examples/wrong_hydrochloride.sdf
```

Pictures:

	
wrongly coded hydrogenchloride	result of the used uncharged option Note, that both molecules now have no charge.

-desalt

If this parameter is set, counterions are removed from the input file. For removing counterions, currently two methods are available based on the molecule size or the specification of given counterions, respectively. The default algorithm splits each input record into ions and discards all but the largest ion, as determined by the atom count, including hydrogen atoms.

Larger counterions can be selectively removed with the **-saltfile** option. Molecules found in that file have precedence for deletion from the input record, even if they are larger than their counterparts.

Default value:

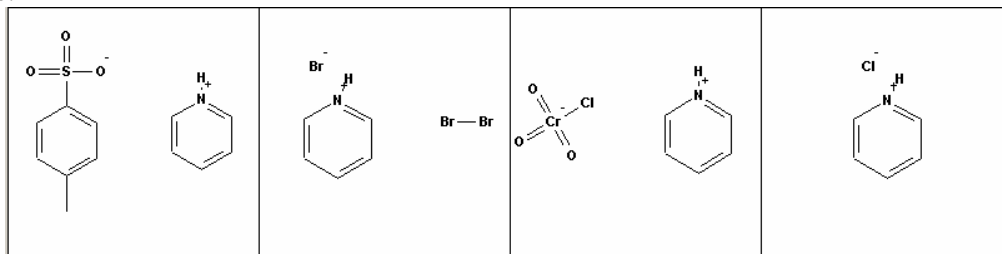
It is not active by default.

Example:

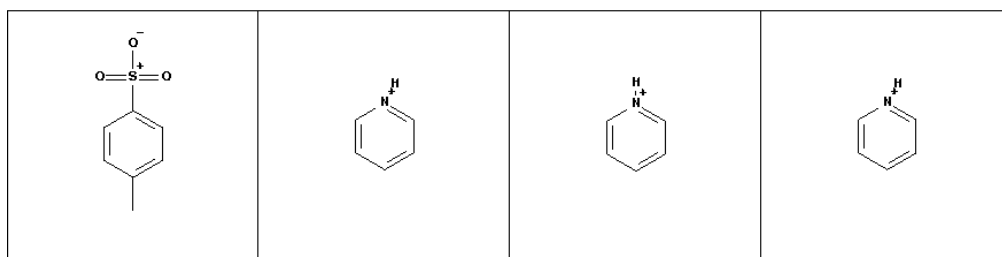
Generating a MDL SD-file removing counterions based on the molecule size:

```
mn_check -outfile unsalted.sdf -desalt ./examples/salts.sdf
```

Pictures:



Structures of the input file



Structures of the output file

-saltfile <filename>

This parameter is used to specify the name of a multi-record file in MDL or SMILES format with fragments that should be discarded when removing counterions.

Default value:

Parameter without a default value

Examples:

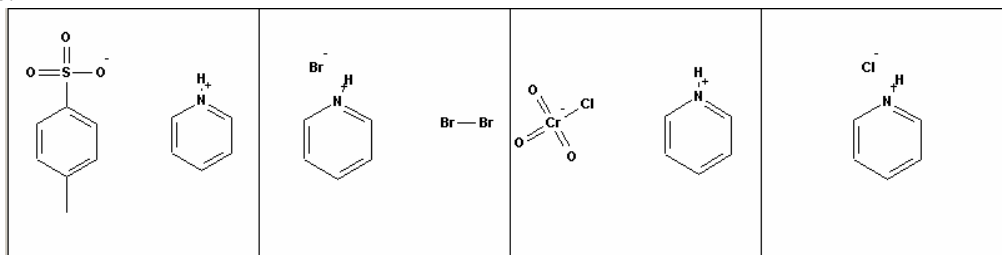
Generating an MDL SD-file by removing counterions based on a given file containing counterions which should be removed in the input file:

```
mn_check -outfile unsalted.sdf -desalt -saltfile  
./examples/pyridinium.smi ./examples/salts.sdf
```

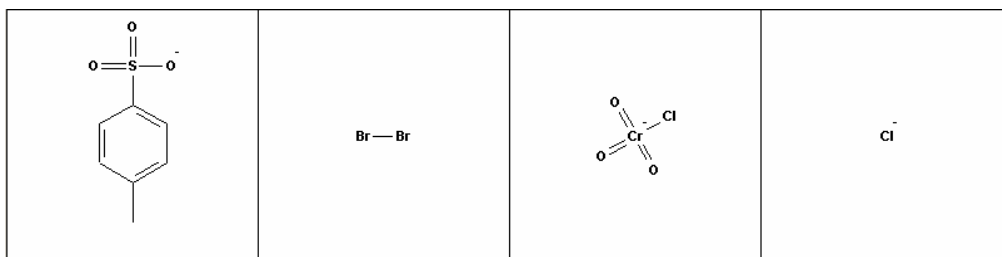
Remarks:

If the **-desalt** option is not set, this parameter has no effect.

Pictures:



Structures of the input file



Structures of the output file

-stat

If this flag is set, statistical information about the number of successfully processed records and conversion failures is written to the standard error channel.

Default value:

This flag is deactivated by default.

Example:

Generating a MDL SD-file showing the statistical information of the conversion:

```
mn_check -outfile maybridge_checked.sdf -stat
./examples/maybridge.sdf
```

Output:

```
Convert file ./examples/maybridge.sdf
Successfully read 62622 records, failed 0
Successfully wrote 62622 records, failed 0.
```

-version

If this flag is set, the version and licensing information is printed.

Default value:

This flag is deactivated by default.

Example:

Showing the program version:

```
mn_check -version
```

-h or -help

If this flag is set, a brief help message about the usage of the program is shown.

Default value:

This flag is deactivated by default.

Example:

Show the help message:

```
mn_check -h or mn_check -help
```

8. Extended Features Only Available for the UNIX Operating Systems

Input files can be processed in compressed or gzip-ed form without prior unpacking. The input file name arguments may each be a local file, an URL (http, ftp, gopher, file) or an email message file containing the structure data in the main body or as one or more attachments. URL retrieval and compression can be combined.

9. Frequently Asked Questions (FAQ)

10. Error Messages

MN.CHECK does not process the single input structure although a valid SMILES string is given

Running MN.CHECK under Windows the very last line of the input structure must be empty.

11. Known Problems and Limitations

12. Technical Support

The MN.CHECK Web Site

If you have problems while running MN.CHECK please have a look at the Support- and FAQ web site of MN.CHECK. The pages are available at <http://www.mol-net.de>

Reporting Problems

If your problem is not listed in these web pages please report it to the MN.CHECK team at Molecular Networks. Please make sure to provide us with all important data for replicating your problem on our machines. Therefore please use the report form on the next page.

Updates

If you have licensed the program MN.CHECK with maintenance you will automatically receive updates every time a new release is launched.

Contact Information

Distribution and Maintenance for MN.CHECK is handled by Molecular Networks Computerchemie, Erlangen, Germany.

Molecular Networks GmbH
Computerchemie
Nägelsbachstraße 25
91052 Erlangen
Germany

e-Mail: support@mol-net.de

Tel. +49 9131/815668

Fax +49 9131/815669

13.Report Form

In case of problems occurring during installation or running MN.CHECK, please complete the following form and send it or fax it to

Molecular Networks GmbH Computerchemie
Nägelsbachstraße 25
91052 Erlangen
Germany
FAX: +49-(0)9131-815669

User:

MN.CHECK program and version number (mn_check -version):

Command line to run MN.CHECK:

Error and warning messages by MN.CHECK:

System messages:

Short description:

Please include the input file and output file generated by MN.CHECK on a 3½" diskette written in MS/DOS format or send an e-mail to support@mol-net.de attaching these files. These files will help us to analyze your problems. All data will be treated confidentially.

14.Index

inputfile
 alkanes1_12_error7.sdf 13
 maybridge.sdf 13, 17
 nitrobenzene.sdf 9, 12, 14
 not_unique.sdf 13
 salts.sdf 16
 wrong_hydrochloride.sdf 14, 15

option
 chargebalance 14
 desalt 15
 directory 12
 feedback 12
 format 11
 h 17
 help 17
 hydrogens 14
 nitrostyle 14
 outfile 12
 pedantic 13
 saltfile 16
 skiperrors 13
 stat 17
 uncharge 15
 unique 13
 version 17

outputfile
 alkanes1_12_minus7.sdf 13
 hydrochloride.sdf 14
 maybridge_checked.sdf 13, 17
 nitrobenzene.sdf 12
 nitrobenzene_H.sdf 12, 14
 nitrobenzene_ionic.sdf 14
 uncharged.sdf 15
 unique.sdf 13
 unsalted.sdf 16

referencefile
 pyridinium.smi 16