Software

SubMat

Generation of
Binary Substructure Descriptors
for Chemical Structures

User Guide

Version 3.1

Laboratory for **C**hemo**M**etrics

Institute of Chemical Engineering
Vienna University of Technology, Austria

June 2004

AUTHORS

Heinz **Scsibrany** and Kurt **Varmuza**

Laboratory for ChemoMetrics
Vienna, Austria

CORRESPONDENCE

Prof. Dr. Kurt Varmuza

Laboratory for ChemoMetrics
c/o Institute of Chemical Engineering
Vienna University of Technology
Getreidemarkt 9/166-2
A-1060 Vienna, Austria

kvarmuza@email.tuwien.ac.at
www.lcm.tuwien.ac.at

LIMITED WARRANTY

No warranties are made by the authors that the Program or User Guide are free of errors. The user relies on the results of the Program solely at his/her own risk. The authors are not liable to any damages caused by bugs or ambiguities in the Program, the delivered data files or the User Guide.

COPYRIGHT

Laboratory for ChemoMetrics, c/o Vienna University of Technology, Austria

SubMat-UserGuide-3c.doc 2004-06-11

CONTENTS

			page
1	Introduction		
	1.1	Overview	4
	1.2	Example	4
	1.3	Operating modes	5
	1.4	Structure files	5
2	Installation		6
3	Menu for interactive mode		
	3.1	File	6
	3.2	View	7
	3.3	Calculation	7
4	Chemical structures		
	4.1	Format	8
	4.2	Isomorphism examples	8
	4.3	Creation of a structure file	9
	4.4	Limits	9
5	Remote mode		
	5.1	Calling SubMat	10
	5.2	Command file	10
	5.3	Communication files	12
	5.4	Error codes	12
6	Appendix		
	6.1	Example of an element file	13
	6.2	Calling SubMat from a Matlab program	15

1 Introduction

Software SubMat is a scientifically oriented tool for computer chemistry and chemometrics. SubMat is running under the operating systems Microsoft Windows 95/98/ME/NT/2000/XP.

1.1 Overview

Main function of SubMat is the generation of a matrix containing binary substructure descriptors (*substructure isomorphism matrix*, ***S***).

Input files file with molecular structures (SDF/Molfile format)
file with substructures (SDF/Molfile format)

Output file file with the generated binary substructure descriptors
(TXT format, format controlled by a few input parameters)

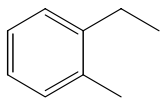
SubMat does not consider 3D or stereo data.
SubMat does not include tools for an evaluation of the generated descriptors; however, the output file can be easily imported by other software.

Binary substructure descriptors - as calculated by SubMat - can be used for a variety of problems in computer chemistry:

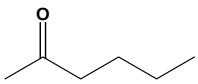
+ Substructure searches
+ Structure similarity searches
+ Search for structure duplicates
+ Characterization of the diversity of structural libraries
+ Cluster analysis of chemical structures
+ Modeling structure-property relationships (together with other structural descriptors)
+ Topological hierarchy of structures, as described in
Varmuza K., Scsibrany H.: *J. Chem. Inf. Comput. Sci.* **40**, 308 (2000)

1.2 Example

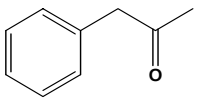
Input file with molecular structures contains 5 structures



1

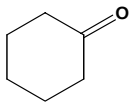


2



3

4

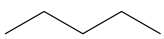


5

Input file with substructures contains 3 structures

1

2



3

The generated output file may look as follows:

```
0 1 0
1 0 1
1 1 0
1 0 0
1 0 1
```

Each molecular structure corresponds to a row.
Each substructure corresponds to a column.
"1" indicates the presence of a substructure in a molecular structure,
"0" indicates its absence.

1.3 Operating modes

SubMat can be used in two different operating modes.

- **Interactive** mode with a typical Windows user interface.

- **Remote** mode by calling SubMat from another program.
  A command file is used to transfer parameters to SubMat.
  So called semaphore files are used for communication between the programs (error messages, interrupt, etc.).
  In this mode no window is opened by SubMat.

1.4 Structure files

- The files used in the example in section 1.2 are provided with SubMat:

Demo-Molecules.SDF	5 molecular structures,
Demo-Substructures.SDF	3 substructures.

- A file (Substructures-500-a2.sdf) containing 500 *general-purpose* substructures is provided with the full version of SubMat.

Look at the README.TXT file.

2 Installation

No special requirements for the personal computer and no special installation are necessary.

Look at the README.TXT file.

(1) Create (or select) a folder (or several folders) for the use of SubMat.

(2) Copy the program file SUBMAT.EXE and the provided data files into this (these) folder(s).

(3) Start the program by a double-click at SUBMAT.EXE.
Eventually create an icon for SubMat by a method available in the used operating system.

Typical computing time (Pentium IV 2.6 GHz):

1000 molecular structures and 200 substructures	1 s
per descriptor value	5 - 7 μs

3 Menu for interactive mode

3.1 File

Load Substructures ...
Select a file containing the substructures in Molfile format (SDF-file).

Load Molecular Structures ...
Select a file containing the molecular structures in Molfile format (SDF-file).

Read Element file ...
All necessary data about elements are already implemented in SubMat. Optionally, another element file can be loaded which for instance may contain user-created data. The format of an element file can be seen from the example given in section 6.1.

Exit

3.2 View

View Substructures

Display of the loaded substructures in a separate window.

View Molecular Structures

Display of the loaded molecular structures in a separate window.

View Elementfile

Display of element data in a separate window.

3.3 Calculation

S-Matrix ...

Calculates the *substructure isomorphism matrix, **S***, containing the generated binary descriptors for the loaded molecular structures and the loaded substructures.

Auto S-Matrix ...

Calculates the ***auto*** *substructure isomorphism matrix, **S***.
In this application the same structures are used as substructures as well as molecular structures.
Only the structures loaded as substructures are used.

The resulting auto-***S***-matrix is quadratic. It contains for instance information about the topological hierarchy of the loaded structures.
When "Check Structures for Identity" was marked it contains information about duplicate structures.

Parameter for the output file (*S*-matrix or auto-*S*-matrix)

(1) File name

(2) Selection of the delimiting character to be placed between the columns (comma, or blank, or no delimiter).

(3) Optional selections

- ☐ Two header lines containing the number of used substructures and the number of used molecular structures.
- ☐ A header line containing identification numbers of the substructures.
- ☐ A column containing identification numbers of the molecular structures.
- ☐ Check Structures for Identity.
  If this box is marked the structures are matched for topological identity. A "1" in the resulting matrix means identical structures.
  See section 4.

4 Chemical Structures

4.1 Format

Input format Molfile format (not all features are supported)

http://www.mdli.com/downloads/ctfile/ctfile_subs.html
Dalby A. et al: *J. Chem. Inf. Comput. Sci.* **32**, 244 (1992)

Hydrogen atoms Structures are usually given without hydrogen atoms; however, explicit hydrogen atoms can be used. Free valences in molecular structures are interpreted as H-substituted. Free valences in substructures can be attached to any atom.

Atoms Elements hydrogen ... uranium, or as contained in a user-defined element table,
Q (any element except hydrogen),
A (hetero atom, any element except carbon or hydrogen).

Bonds

single	double	triple	aromatic	query

Not used isotope masses,
atom charges,
3D or stereo data.

4.2 Isomorphism examples

Substructure Q=Q is contained in molecular structure C=C.
Substructure C=C is NOT contained in molecular structure Q=Q.
The same applies for hetero atoms, A.

C=O is a substructure of $H_2C=O$ (defined with two explicit H-atoms!) and vice versa. However, if the switch "Check Structures for Identity" is activated these two structures are not found to be identical.

Aromatic rings can be defined either by aromatic bonds (Molfile bond code 4) or by alternating single and double bonds. SubMat automatically checks for aromatic rings. The criteria applied for aromatic rings are

- ring size = 4n + 2 (n = 1, 2, 3, ...),
- alternating (re-locatable) double and single bonds.

All bonds in detected aromatic rings are converted into aromatic bonds before substructure searches start.

A **query bond** (Molfile bond code 8, bond type "any") means that the bond type is not defined. For example substructure C~~C (~~ is used here for a query bond) is for instance contained in the molecular structures C-C and C=C. However, the substructures C-C or C=C are for instance NOT contained in molecular structure C~~C.

4.3 Creation of a structure file

Use a structure editor to draw structures and export each of them in Molfile format (*.MOL). Concatenate the MOL-files to a SDF-file; for instance by the DOS command **`COPY *.MOL NEWFILE.SDF`**.

SDF-files must contain only structural data. Any structure must be terminated by a line containing "**`M  END`**" or (!) "**`$$$$`**". After this line the begin of the next structure (or end of file) is expected. See the provided example files.

4.4 Limits

Maximum number of atoms in a structure:	127
Maximum number of bonds in a structure:	255
Maximum number of bonds at an atom:	11

The maximum number of molecular structures and substructures is limited by the available memory. With 256 Mbyte RAM ca 10,000 molecular structures and 1,000 substructures should be possible.

5 Remote mode

SubMat can be executed by calling it from another program (for instance from a Matlab program). A command file (in TXT format) is used to transfer parameters to SubMat. So called semaphore files are used for a simple communication between the calling program and SubMat (error messages, progress, interrupt, termination). In remote mode usually no window is opened by SubMat.

Experiences reported here are mainly based on applications within Win 2000 and Matlab version 6.

5.1 Calling SubMat

- From a DOS batch file
  <path1>SUBMAT.EXE <path2><command_file>

- From a Basic program
  SHELL "<path1>SUBMAT.EXE <path2><command_file>"

- From a Matlab program (see section 6.2)
  dos('<path1>submat.exe <path2><command_file> &')

  Remarks for the Matlab command:

  * exe in lower case avoids opening of a new window for the started SubMat
  * EXE in upper case opens a new window for the started SubMat; this window has to be closed manually.
  * **&** causes the Matlab program to continue after the start of SubMat

<command_file>	command file in text format with all parameters and filenames necessary (described in section 5.2)
<path1>	path for SubMat exe-file
<path2>	path for command file

5.2 Command file

The command file transfers to SubMat: names of files to be used or created, parameters.

The command file is a text file; it consists of data lines. Each line has the format
<keyword> = <data>

Some keywords contain blanks; these blanks must not be omitted; lower or/and upper case characters may be used. Blanks before and after the "=" are allowed.

keyword	data

mandatory

Substr File =	path and name of file containing the substructures
Structure File =	path and name of file containing the molecular structures
Result File =	path and name of output file containing results (the substructure isomorphism matrix)
Progress File =	path and name of the progress file containing the progress (%) of computation or eventually an error code
Sem End File =	path and name of the semaphore end file that indicates the end of SubMat execution
Stop File =	path and name of file that causes a stop of SubMat

optional

Elementtable =	path and name of a user-defined element file
S-Matrix Param =	parameter for computation and for the result file (use the sum of the selected codes): 1 write two header lines containing the number of substructures and the number of molecular structures 2 write a header line containing substructure identification numbers 4 write a first column containing molecular structure identification numbers 16 delimiter is comma, default: no delimiter 32 delimiter is blank if code 16 is not used, default: no delimiter 512 Check Structures for Identity

Example for a command file

```
Substr File     = d:\submat\tests\demo-substructures.sdf
Structure File  = d:\submat\tests\demo-molecules.sdf
Result File     = d:\submat\tests\demo$results.txt
Progress File   = d:\temp\progress-file.txt
Sem End File    = d:\temp\finished.txt
Stop File       = d:\temp\stop.txt
S-Matrix Param  = 17
```

5.3 Communication files

Progress file
SubMat creates this file and writes to it the progress of computation (%, a single integer number).
If an error during the execution of SubMat occurs a negative error code is written to the progress file (see section 5.4).

Semaphore end file
SubMat creates this file after closing the result file (and immediately before termination). This file is empty. Existence of this file tells the calling program that the result file is ready to be read.

Stop file
This file can be used to terminate the execution of SubMat. The file has to be created by the calling program. The content of the stop file determines how to finish SubMat:

"0" no result file is made
"1" a result file is made from the data available at the time of interrupt

Attention
After SubMat has finished the calling program has to delete three files (eventually after check of their existence):

progress file,
semaphore end file,
stop file.

5.4 Error codes

- Errors detected by SubMat during execution are reported - if possible - via the progress file by a negative error code (instead of % progress).

error code	error
-1	substructure file wrong
-2	molecular structure file wrong
-4	result file wrong
-8	stop file wrong
-16	end semaphore file wrong
-32	element file wrong
-10001 to -26380	substructure with number (-error code + 10000) wrong
-30001 to -46380	molecular structure with number (-error code + 30000) wrong

File "wrong" may mean:
- file name not given in the command file,
- file not found.

- In some error cases, the progress file cannot be opened by SubMat and the error is reported in an error message box. The calling program cannot (easily) check such error messages; the error message box has to be closed by the user.

error message	error
Invalid command file name	name of command file wrong or file not found
Invalid progress file name	name of progress file wrong or file cannot be created

6 Appendix

6.1 Example of an element file

Necessary data about elements are already implemented in SubMat. Optionally, another element file can be loaded which for instance may contain user created data. The text format of an element file is demonstrated by the following example. Explanations are given below.

```
*User created element file (example)
*Data in this examples are default values.
H
Wasserstoff
Hydrogen
1, 2, 1.00797
1, 99.985, 7825
2, 0.01492, 14102
*
He
Helium
Helium
2, 2, 4.0026
3, 0.000137, 16030
4, 100, 2603
*
Li
Lithium
Lithium
3, 2, 6.939
6, 7.42, 15123
7, 92.58, 16005
*
```

... and so on ...

The first two lines contain comments (starting with "*****").

Each element contains 5 + *n* lines (with *n* being the number of isotopes):

(1) Element symbol
(2) Element name in any language (German in this example)
(3) Element name in English
(4) 3 numbers: atomic number,
number of isotopes,
averaged atomic mass
(5) 3 numbers for first isotope:
integer mass,
abundance in %,
mass defect given as (exact mass - integer mass)*10^6
for instance for ^{1}H: (1.007825 - 1)* 10^6 = 7825,
for instance for ^{35}Cl: (34.986854 - 35)* 10^6 = -31146
(.) eventual data for further isotope(s)

(5+n) "*****" as a terminator

6.2 Calling SubMat from a Matlab program

A simple source code in Matlab and some comments (in blue color) are given for calling SubMat, and then checking the progress file and finally terminating SubMat. No paths for files are used in this example.

```
% prepare   command file (command.txt),
%           SDF-file with substructures,
%           SDF-file with molecular structures
% define variables   progressFile     name of progress file
%                    endFile          name of end semaphore file

% start submat
dos('submat.exe command.txt &')            % exe in lower case !

% loop during execution of SubMat

endloop = -1;
while endloop == -1

   pause(1);          % pause of 1 second (check every second the
                      % progress file and the end semaphore file)

   % check if progress file exists
   % if yes then read the progress file
   progressFileExist = exist(progressFile,'file');
   if progressFileExist == 2
      fidProgress = fopen(progressFile,'rt');         % open file
      lin = fgetl(fidProgress);                       % read file
      progressNumber = str2num(lin);          % convert to number
      % if number read is positive then continue
      % else terminate loop because of an error
      if  progressNumber > 0
            % output progress in percent ...
      else
            % output error message ...
          endloop = 1;
      end
      fclose(fidProgress)                           % close file
   end

   % check if end semaphore file exists
   % if yes then terminate loop
   endExist = exist(endFile,'file');
   if endExist == 2
      delete(endFile);
      endloop = 1;
   end

end
delete(progressFile)

% read result file
```

End of User Guide.