Open Babel

Access and interconvert chemical information

Noel M. O’Boyle

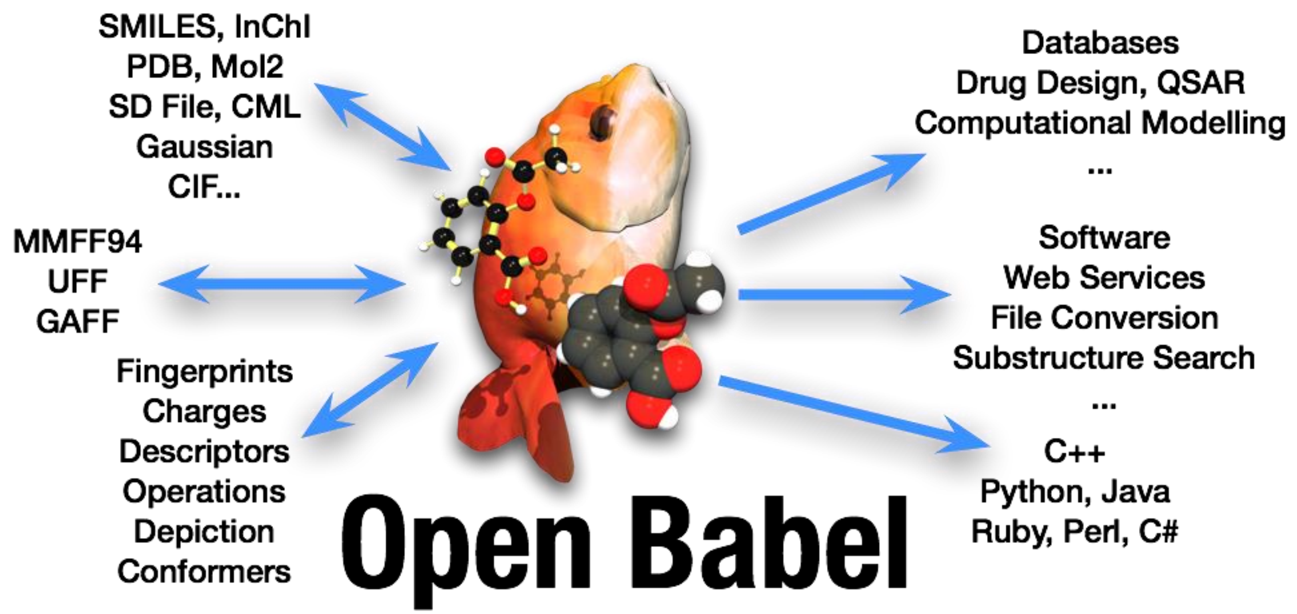
Open Babel development team and NextMove Software, Cambridge, UK

Dec 2013

EMBL-EBI/Wellcome Trust Course: Resources for Computational Drug Discovery

Image credit: AJ Cann (AJC1 on Flickr)

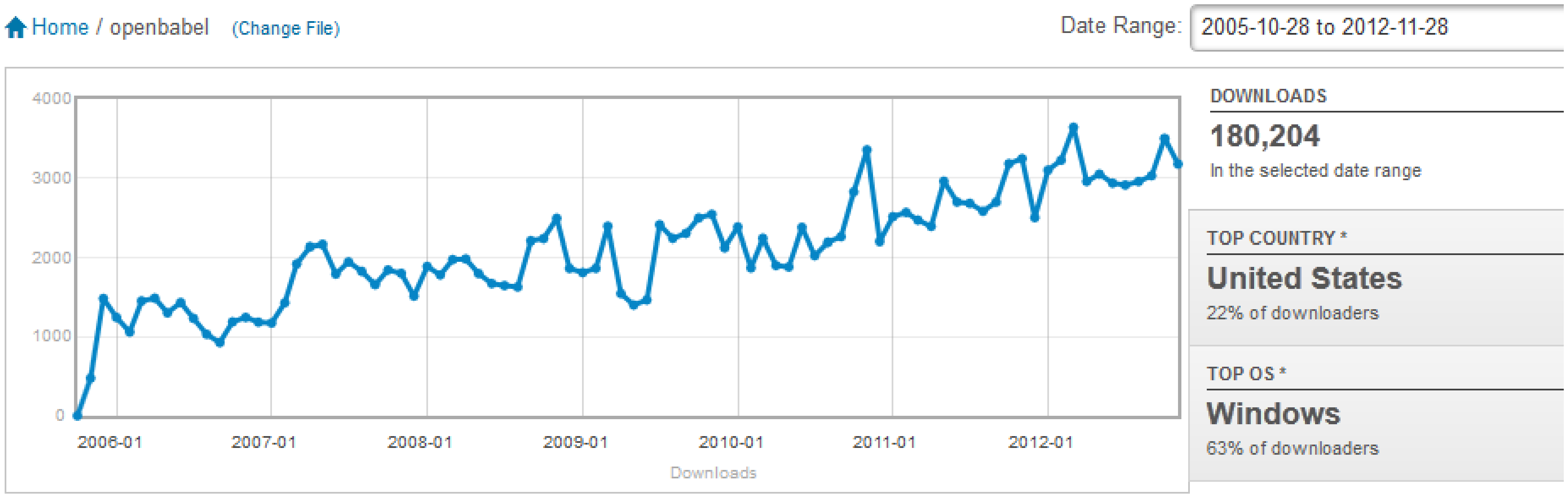
Image credit: Jon Osborne (jonno101101 on Flickr)



- Volunteer effort, an open source success story
  - Originally a fork from OpenEye's OELib in 2001
  - Lead is Geoff Hutchison (Uni of Pittsburgh)
  - 4 or 5 active developers – I got involved in late 2005

- http://openbabel.org
- Associated paper: (Open Access)
  - **Open Babel: An open chemical toolbox**, J. Cheminf., 2011, 3, 33.

Does anyone else use Open Babel?


Home / openbabel (Change File)

Date Range: 2005-10-28 to 2012-11-28

DOWNLOADS
180,204
In the selected date range

TOP COUNTRY *
United States
22% of downloaders

TOP OS *
Windows
63% of downloaders

- 40K downloads (from SF) in last 12 months
  - 1.4K downloads of Windows Python bindings
- Paper #1 most accessed in last year
  - Cited 60 times in 1 year
- In short, very widely-used

Features

- Multiple chemical file formats (+ options) and utility formats
- 2D coordinate generation and depiction (PNG and SVG)
- 3D coordinate generation, forcefield minimisation, conformer generation
- Binary fingerprints (path-based, substructure-based) and associated “fast search” database
- Bond perception, aromaticity detection and atom-typing
- Canonical labelling, automorphisms, alignment
- Plugin architecture
- Several command-line applications, but also a software library
- Written in C++ but bindings in several languages

obabel and file conversion

- Basic usage:

  obabel infile.extn –O outfile.extn

- Can also read from *stdin*, write to *stdout*, read from a SMILES string, specify the input and output file formats, specify conversion options, and format specific options
  - Or ask for help (obabel –H)…online docs better!

- Note: **obabel** has replaced the older **babel**

Conversion options

- Handle multimolecule files
  join/m, sort, C
- Handle multicomponent molecules
  r, separate
- Filter
  filter, smallest/largest, s/v, f/l, unique
- Manipulate structure or atom order
  addpolarh, align, b, c, canonical, d, h, gen2d/3d
- Forcefield
  minimize, conformer, energy
- Conformers
  readconformer, writeconformers
- Manipulate SDF properties and title
  add, addfilename, addindex, addoutindex, addtotitle, append, delete, property, title

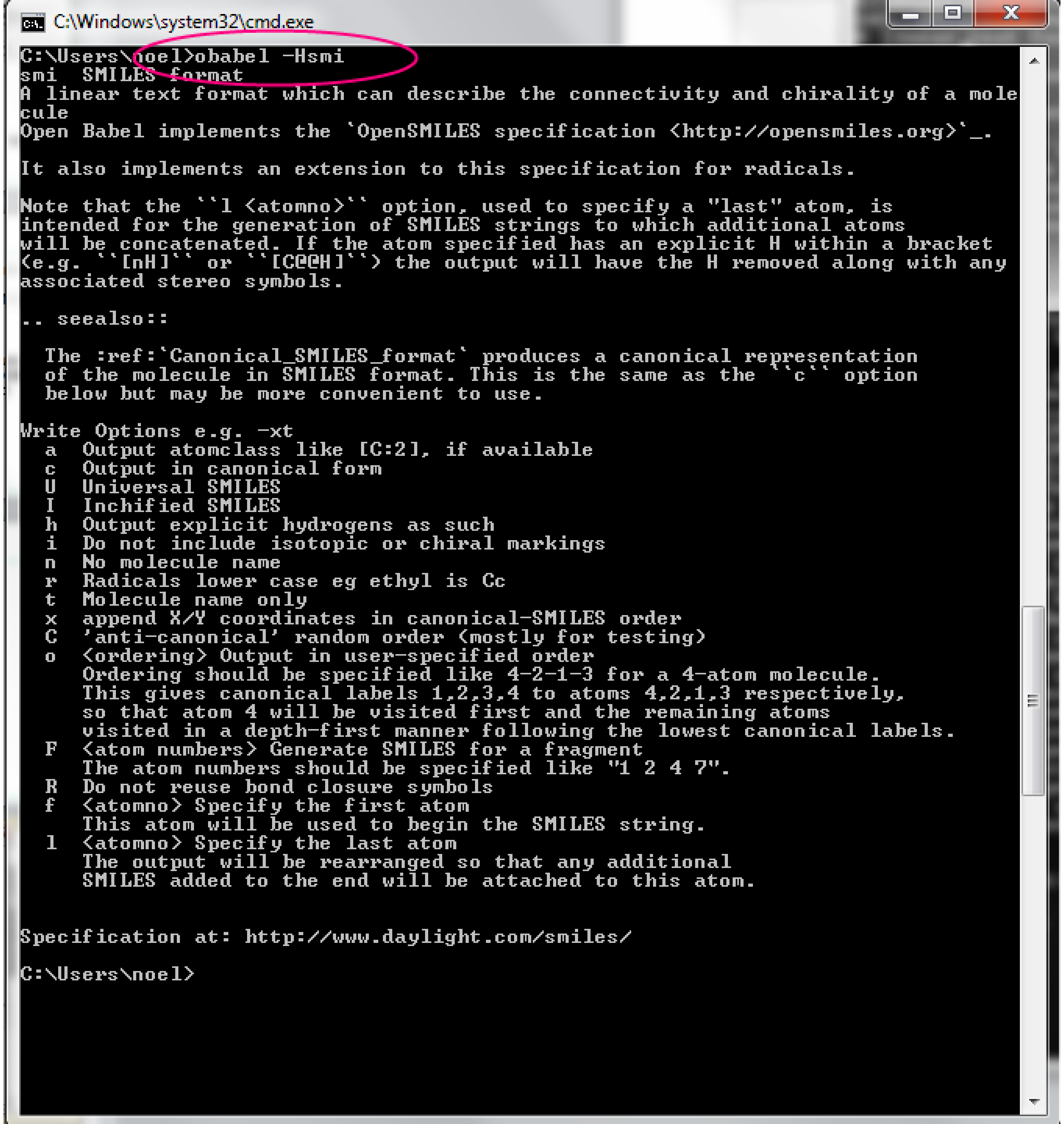
See ***http://openbabel.org/docs***

File-format options

- Particular file formats may have their own specific input or output options
  - To provide or handle different flavours of the file format
  - To specify additional information to include
  - To provide additional functionality

- Options are listed in the help text for a format (see next slides)

- To use:
  - specify read options with **–a** (e.g. –ar)
  - specify write options with **–x** (e.g. –xi)



```
C:\Users\noel>obabel -Hsmi
smi  SMILES format
A linear text format which can describe the connectivity and chirality of a mole
cule
Open Babel implements the `OpenSMILES specification <http://opensmiles.org>`_.

It also implements an extension to this specification for radicals.

Note that the ``l <atomno>`` option, used to specify a "last" atom, is
intended for the generation of SMILES strings to which additional atoms
will be concatenated. If the atom specified has an explicit H within a bracket
(e.g. ``[nH]`` or ``[C@@H]``) the output will have the H removed along with any
associated stereo symbols.

.. seealso::

  The :ref:`Canonical_SMILES_format` produces a canonical representation
  of the molecule in SMILES format. This is the same as the ``c`` option
  below but may be more convenient to use.

Write Options e.g. -xt
  a  Output atomclass like [C:2], if available
  c  Output in canonical form
  U  Universal SMILES
  I  Inchified SMILES
  h  Output explicit hydrogens as such
  i  Do not include isotopic or chiral markings
  n  No molecule name
  r  Radicals lower case eg ethyl is Cc
  t  Molecule name only
  x  append X/Y coordinates in canonical-SMILES order
  C  'anti-canonical' random order (mostly for testing)
  o  <ordering> Output in user-specified order
     Ordering should be specified like 4-2-1-3 for a 4-atom molecule.
     This gives canonical labels 1,2,3,4 to atoms 4,2,1,3 respectively,
     so that atom 4 will be visited first and the remaining atoms
     visited in a depth-first manner following the lowest canonical labels.
  F  <atom numbers> Generate SMILES for a fragment
     The atom numbers should be specified like "1 2 4 7".
  R  Do not reuse bond closure symbols
  f  <atomno> Specify the first atom
     This atom will be used to begin the SMILES string.
  l  <atomno> Specify the last atom
     The output will be rearranged so that any additional
     SMILES added to the end will be attached to this atom.


Specification at: http://www.daylight.com/smiles/

C:\Users\noel>
```

SMILES format (smi, smiles)

A linear text format which can describe the connectivity and chirality of a molecule

Open Babel implements the OpenSMILES specification.

It also implements an extension to this specification for radicals.

Note that the `l <atomno>` option, used to specify a "last" atom, is intended for the generation of SMILES strings to which additional atoms will be concatenated. If the atom specified has an explicit H within a bracket (e.g. `[nH]` or `[C@@H]`) the output will have the H removed along with any associated stereo symbols.

See also

The *Canonical SMILES format (can)* produces a canonical representation of the molecule in SMILES format. This is the same as the `c` option below but may be more convenient to use.

Write Options

`-a`	*Output atomclass like [C:2], if available*
`-c`	*Output in canonical form*
`-U`	*Universal SMILES*
`-I`	*Inchified SMILES*
`-h`	*Output explicit hydrogens as such*
`-i`	*Do not include isotopic or chiral markings*
`-n`	*No molecule name*
`-r`	*Radicals lower case eg ethyl is Cc*
`-t`	*Molecule name only*
`-x`	*append X/Y coordinates in canonical-SMILES order*
`-C`	*'anti-canonical' random order (mostly for testing)*
`-o <ordering>`	*Output in user-specified order* Ordering should be specified like 4-2-1-3 for a 4-atom molecule. This gives canonical labels 1,2,3,4 to atoms 4,2,1,3 respectively, so that atom 4 will be visited first and the remaining atoms visited in a depth-first manner following the lowest canonical labels.
`-F <atom numbers>`	
	Generate SMILES for a fragment The atom numbers should be specified like "1 2 4 7".
`-R`	*Do not reuse bond closure symbols*
`-f <atomno>`	*Specify the first atom*

12.1.7 SMILES format (smi, smiles)

A linear text format which can describe the connectivity and chirality of a molecule

Open Babel implements the OpenSMILES specification.

It also implements an extension to this specification for radicals.

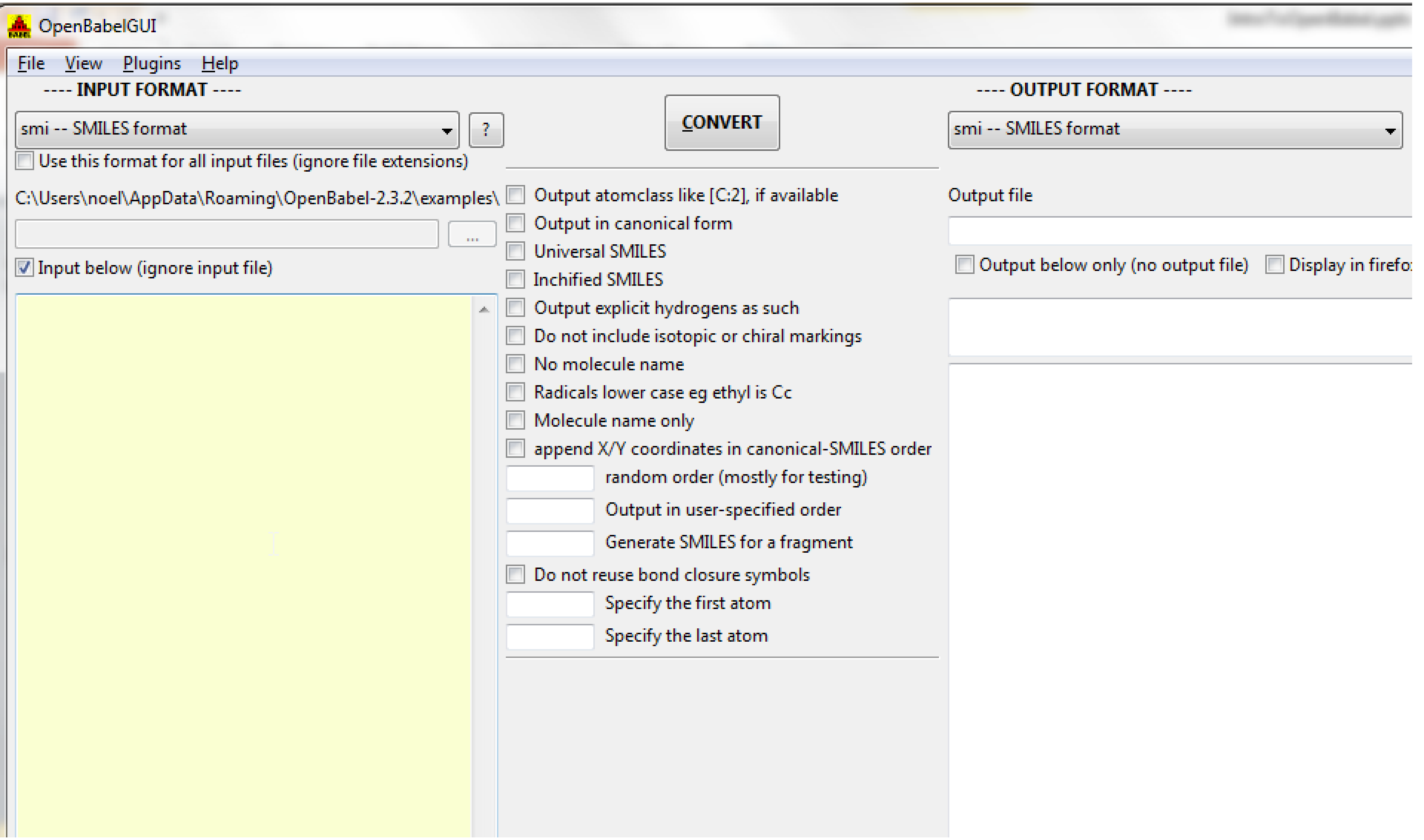
Note that the `l <atomno>` option, used to specify a "last" atom, is intended for the generation of SMILES strings to which additional atoms will be concatenated. If the atom specified has an explicit H within a bracket (e.g. `[nH]` or `[C@@H]`) the output will have the H removed along with any associated stereo symbols.

See Also:

The *Canonical SMILES format (can)* produces a canonical representation of the molecule in SMILES format. This is the same as the `c` option below but may be more convenient to use.

Write Options

a	*Output atomclass like [C:2], if available*
c	*Output in canonical form*
h	*Output explicit hydrogens as such*
i	*Do not include isotopic or chiral markings*
n	*No molecule name*
r	*Radicals lower case eg ethyl is Cc*
t	*Molecule name only*
x	*append X/Y coordinates in canonical-SMILES order*
C	*'anti-canonical' random order (mostly for testing)*
f <atomno>	*Specify the first atom* This atom will be used to begin the SMILES string.
l <atomno>	*Specify the last atom* The output will be rearranged so that any additional SMILES added to the end will be attached to this atom.



smi -- SMILES format

- [ ] Use this format for all input files (ignore file extensions)

C:\Users\noel\AppData\Roaming\OpenBabel-2.3.2\examples\

- [✓] Input below (ignore input file)

- [ ] Output atomclass like [C:2], if available
- [ ] Output in canonical form
- [ ] Universal SMILES
- [ ] Inchified SMILES
- [ ] Output explicit hydrogens as such
- [ ] Do not include isotopic or chiral markings
- [ ] No molecule name
- [ ] Radicals lower case eg ethyl is Cc
- [ ] Molecule name only
- [ ] append X/Y coordinates in canonical-SMILES order
- random order (mostly for testing)
- Output in user-specified order
- Generate SMILES for a fragment
- [ ] Do not reuse bond closure symbols
- Specify the first atom
- Specify the last atom

smi -- SMILES format

Output file

- [ ] Output below only (no output file)
- [ ] Display in firefo

SMILES output options

Note that atom order is preserved

Make atom 3 the first atom…

…and atom 1 the last

```
> obabel -:CC(=O)Cl -osmi
CC(=O)Cl
> obabel -:CC(=O)Cl -osmi -xh -h
[CH3]C(=O)Cl
> obabel -:CC(=O)Cl -osmi -xf 3
O=C(C)Cl
> obabel -:CC(=O)Cl -osmi -xf 3 -xl 1
O=C(Cl)C
> obabel -:CC(=O)Cl -:CC(=O)Cl -osmi -xC
ClC(=O)C
O=C(Cl)C
> obabel -:CC(=O)Cl -osmi -xF "2 4"
CCl
```

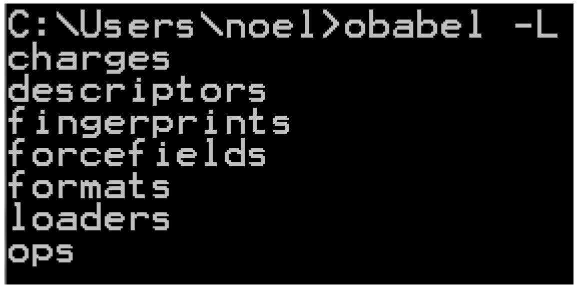
1. Add explicit Hs
2. Show them in the output

Random order

Fragment SMILES for the fragment composed of atoms 2 and 4

Take home message: Look through the list of options for file formats which you frequently use (and request new options!)

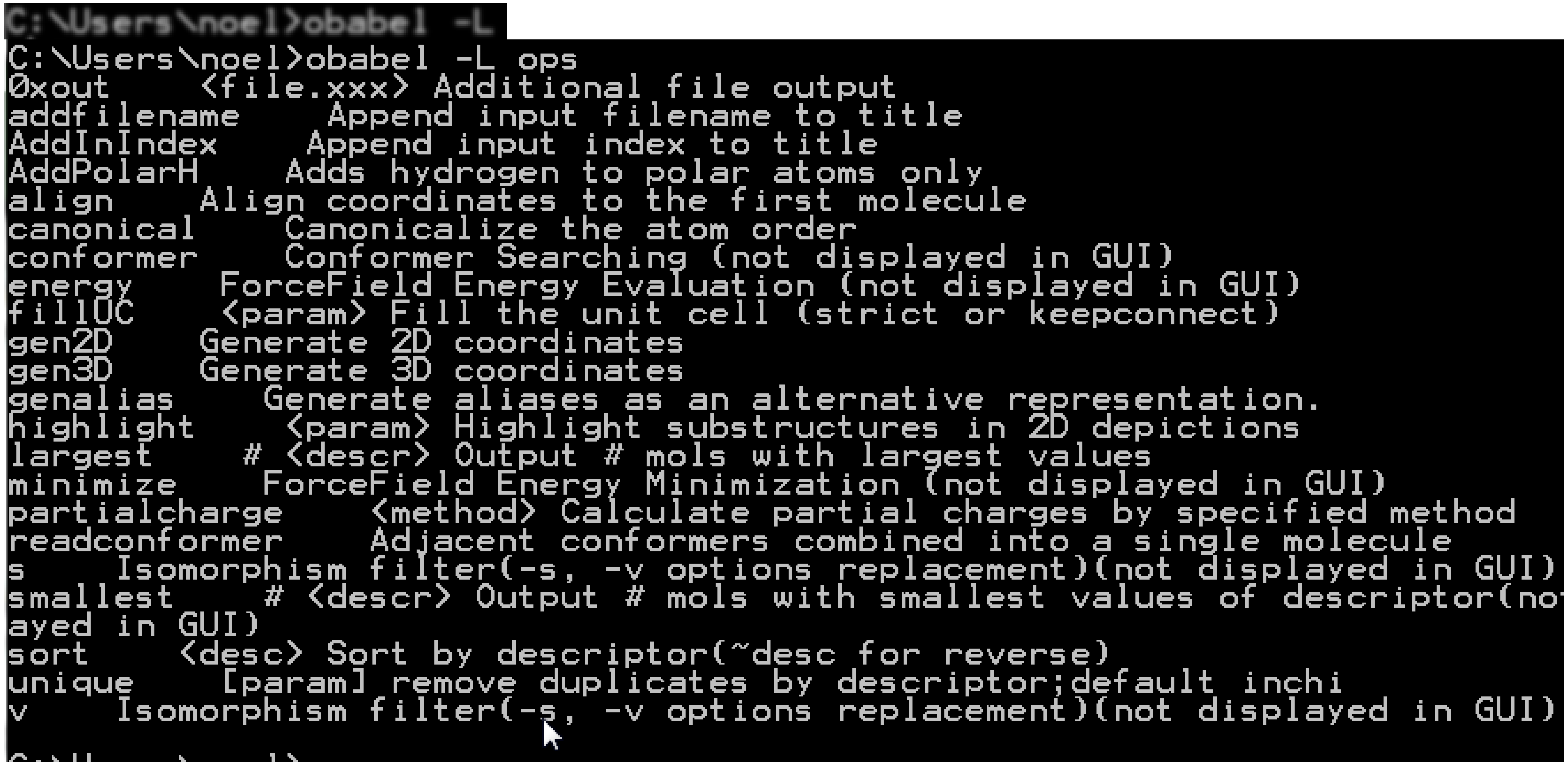
Pro tip #1 “obabel –L” is your friend



```
C:\Users\noel>obabel -L
charges
descriptors
fingerprints
forcefields
formats
loaders
ops
```

Information on plugins and plugin options.

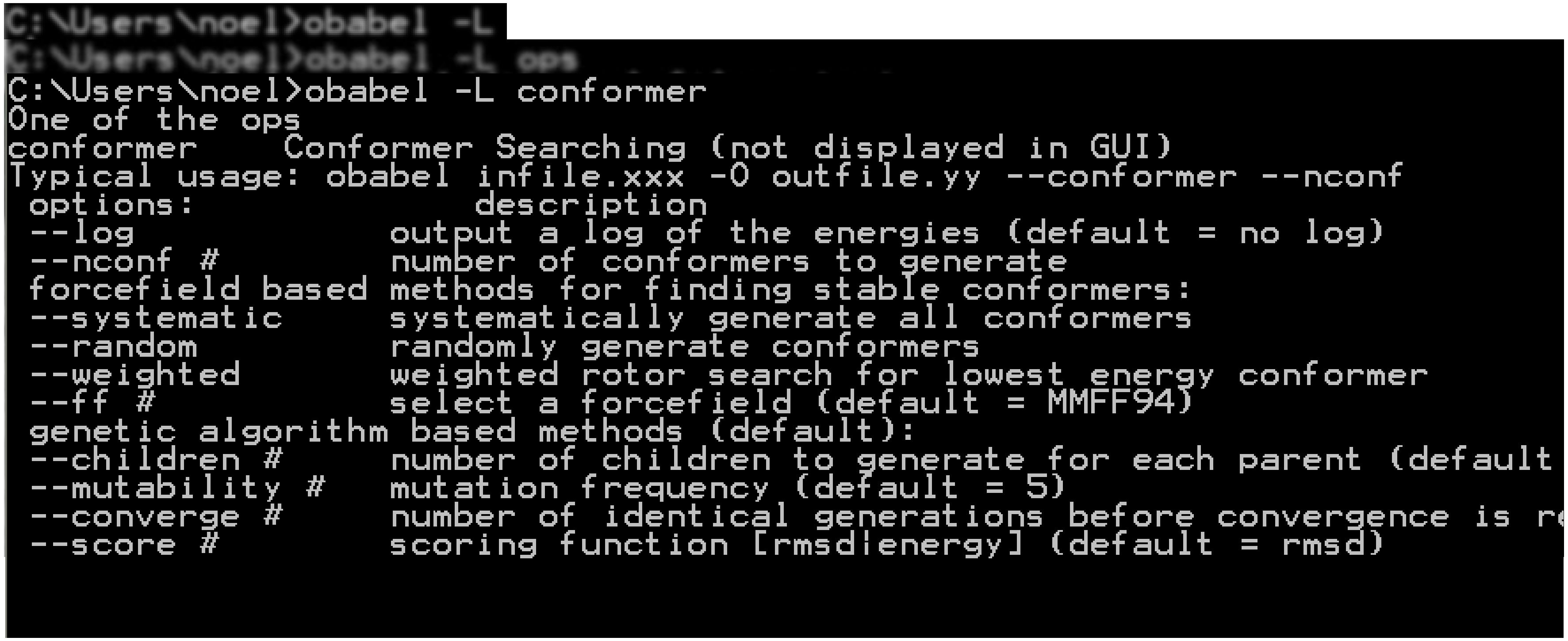
Pro tip #1 “obabel –L” is your friend



```
C:\Users\noel>obabel -L
C:\Users\noel>obabel -L ops
0xout    <file.xxx> Additional file output
addfilename    Append input filename to title
AddInIndex    Append input index to title
AddPolarH    Adds hydrogen to polar atoms only
align    Align coordinates to the first molecule
canonical    Canonicalize the atom order
conformer    Conformer Searching (not displayed in GUI)
energy    ForceField Energy Evaluation (not displayed in GUI)
fillUC    <param> Fill the unit cell (strict or keepconnect)
gen2D    Generate 2D coordinates
gen3D    Generate 3D coordinates
genalias    Generate aliases as an alternative representation.
highlight    <param> Highlight substructures in 2D depictions
largest    # <descr> Output # mols with largest values
minimize    ForceField Energy Minimization (not displayed in GUI)
partialcharge    <method> Calculate partial charges by specified method
readconformer    Adjacent conformers combined into a single molecule
s    Isomorphism filter(-s, -v options replacement)(not displayed in GUI)
smallest    # <descr> Output # mols with smallest values of descriptor(no
ayed in GUI)
sort    <desc> Sort by descriptor(~desc for reverse)
unique    [param] remove duplicates by descriptor;default inchi
v    Isomorphism filter(-s, -v options replacement)(not displayed in GUI)
```

Information on plugins and plugin options.

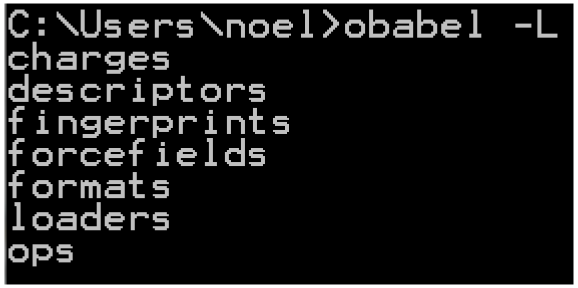
Pro tip #1 “obabel –L” is your friend



```
C:\Users\noel>obabel -L
C:\Users\noel>obabel -L ops
C:\Users\noel>obabel -L conformer
One of the ops
conformer    Conformer Searching (not displayed in GUI)
Typical usage: obabel infile.xxx -O outfile.yy --conformer --nconf
 options:             description
 --log               output a log of the energies (default = no log)
 --nconf #           number of conformers to generate
 forcefield based methods for finding stable conformers:
 --systematic        systematically generate all conformers
 --random            randomly generate conformers
 --weighted          weighted rotor search for lowest energy conformer
 --ff #              select a forcefield (default = MMFF94)
 genetic algorithm based methods (default):
 --children #        number of children to generate for each parent (default
 --mutability #      mutation frequency (default = 5)
 --converge #        number of identical generations before convergence is re
 --score #           scoring function [rmsd|energy] (default = rmsd)
```

Information on plugins and plugin options.

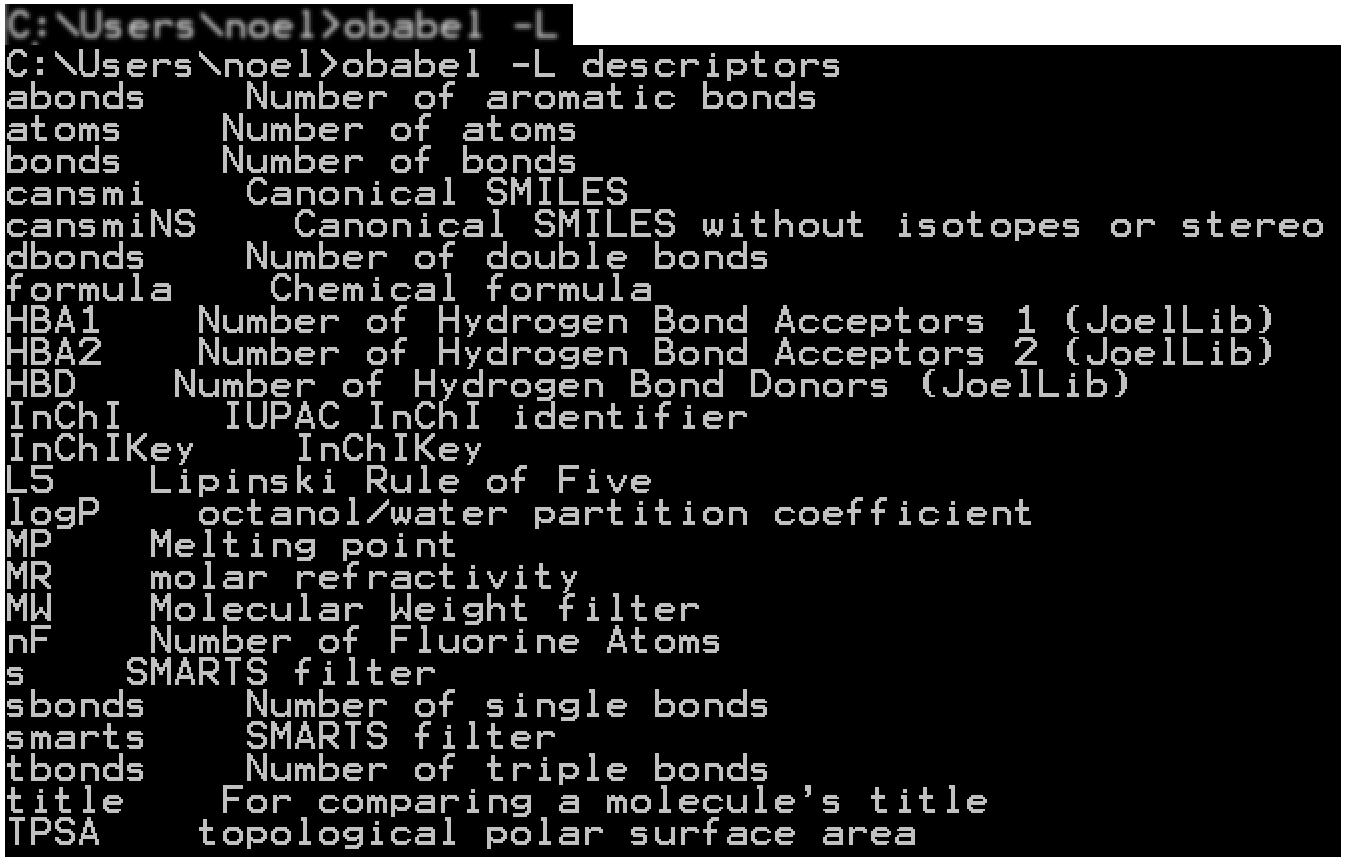
Pro tip #1 “obabel –L” is your friend



```
C:\Users\noel>obabel -L
charges
descriptors
fingerprints
forcefields
formats
loaders
ops
```

Information on plugins and plugin options.

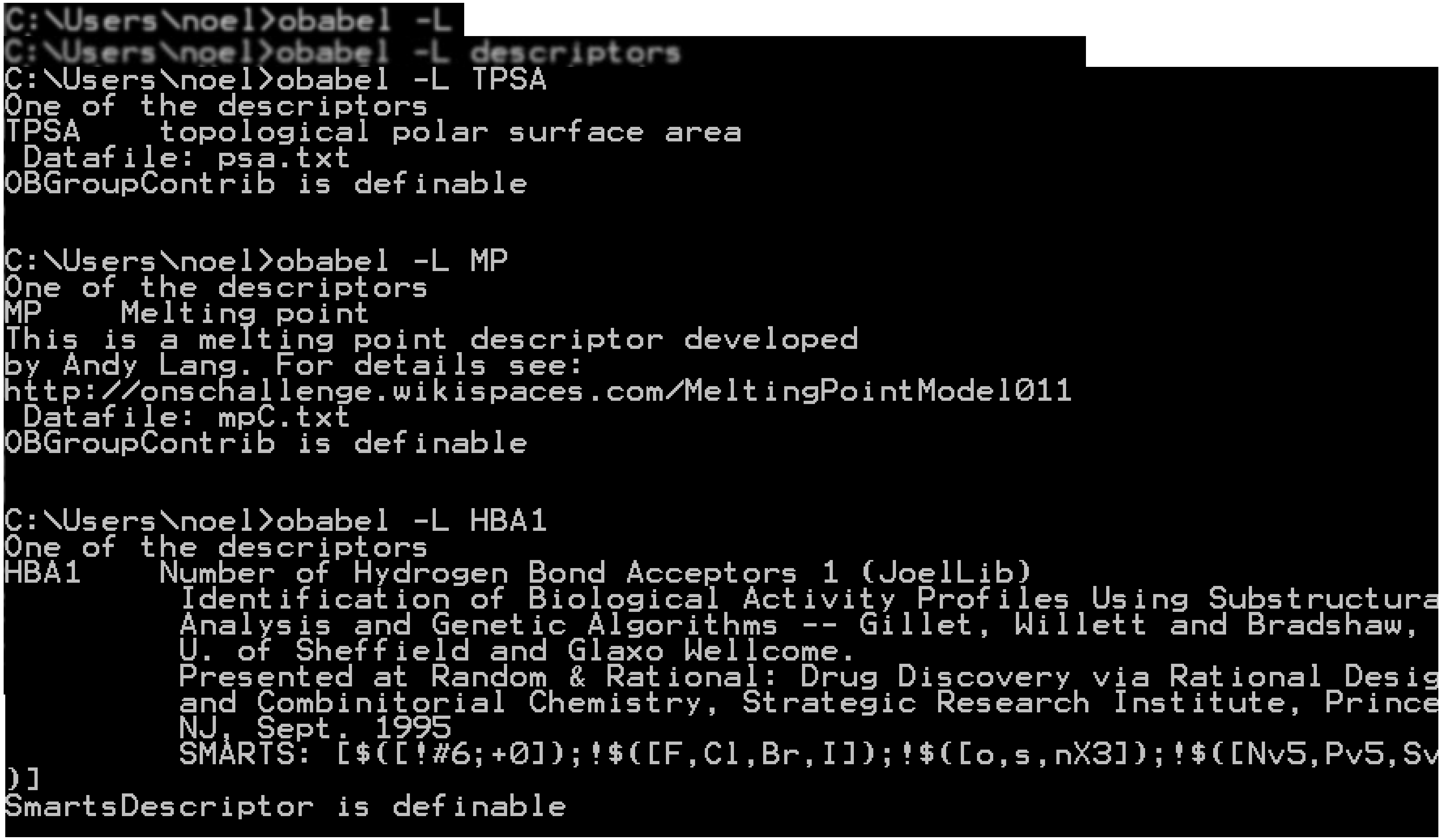
Pro tip #1 “obabel –L” is your friend



```
C:\Users\noel>obabel -L
C:\Users\noel>obabel -L descriptors
abonds     Number of aromatic bonds
atoms     Number of atoms
bonds     Number of bonds
cansmi     Canonical SMILES
cansmiNS     Canonical SMILES without isotopes or stereo
dbonds     Number of double bonds
formula     Chemical formula
HBA1     Number of Hydrogen Bond Acceptors 1 (JoelLib)
HBA2     Number of Hydrogen Bond Acceptors 2 (JoelLib)
HBD     Number of Hydrogen Bond Donors (JoelLib)
InChI     IUPAC InChI identifier
InChIKey     InChIKey
L5     Lipinski Rule of Five
logP     octanol/water partition coefficient
MP     Melting point
MR     molar refractivity
MW     Molecular Weight filter
nF     Number of Fluorine Atoms
s     SMARTS filter
sbonds     Number of single bonds
smarts     SMARTS filter
tbonds     Number of triple bonds
title     For comparing a molecule's title
TPSA     topological polar surface area
```

Information on plugins and plugin options.

Pro tip #1 “obabel –L” is your friend



```
C:\Users\noel>obabel -L
C:\Users\noel>obabel -L descriptors
C:\Users\noel>obabel -L TPSA
One of the descriptors
TPSA    topological polar surface area
 Datafile: psa.txt
OBGroupContrib is definable


C:\Users\noel>obabel -L MP
One of the descriptors
MP    Melting point
This is a melting point descriptor developed
by Andy Lang. For details see:
http://onschallenge.wikispaces.com/MeltingPointModel011
 Datafile: mpC.txt
OBGroupContrib is definable


C:\Users\noel>obabel -L HBA1
One of the descriptors
HBA1    Number of Hydrogen Bond Acceptors 1 (JoelLib)
         Identification of Biological Activity Profiles Using Substructura
         Analysis and Genetic Algorithms -- Gillet, Willett and Bradshaw,
         U. of Sheffield and Glaxo Wellcome.
         Presented at Random & Rational: Drug Discovery via Rational Desig
         and Combinitorial Chemistry, Strategic Research Institute, Prince
         NJ, Sept. 1995
         SMARTS: [$([!#6;+0]);!$([F,Cl,Br,I]);!$([o,s,nX3]);!$([Nv5,Pv5,Sv
)]
SmartsDescriptor is definable
```

Information on plugins and plugin options.

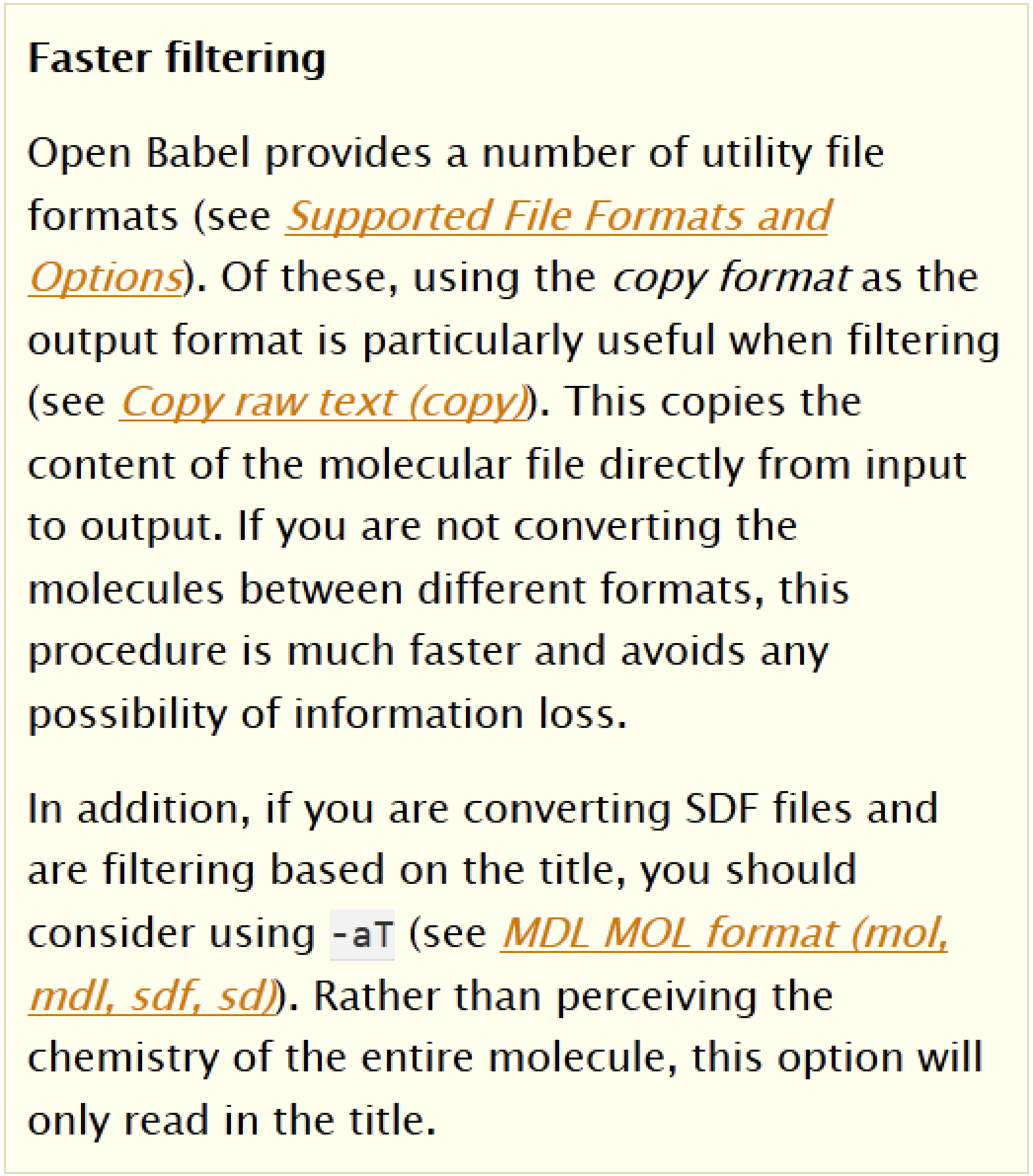
What can be done with descriptors and SDF properties?

- Filter based on value or True/False
  --filter "MW<130 & My_Property < 12"
- Sort and reverse sort --sort ~logP
- Take the N largest or smallest (or everything but)
  --largest 5 MW
- Add SDF properties --add MW
- Add to title (useful for depictions) --addtotitle MW
- Remove duplicates --unique cansmi

- Create more descriptors!
  - Group contribution, SMARTS descriptors or compound descriptors are easily added via text files*

* http://open-babel.readthedocs.org/en/latest/WritePlugins/AddNewDescriptor.html

Pro Tip #2 Faster filtering



Faster filtering

Open Babel provides a number of utility file formats (see *Supported File Formats and Options*). Of these, using the *copy format* as the output format is particularly useful when filtering (see *Copy raw text (copy)*). This copies the content of the molecular file directly from input to output. If you are not converting the molecules between different formats, this procedure is much faster and avoids any possibility of information loss.

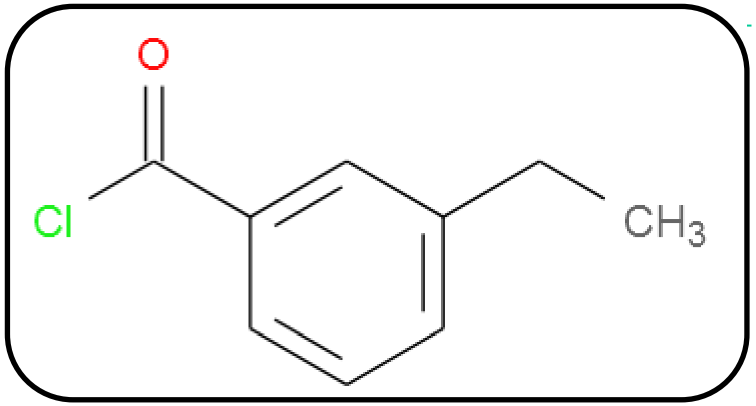
In addition, if you are converting SDF files and are filtering based on the title, you should consider using `-aT` (see *MDL MOL format (mol, mdl, sdf, sd)*). Rather than perceiving the chemistry of the entire molecule, this option will only read in the title.

Also –aP if filtering based on SDF properties

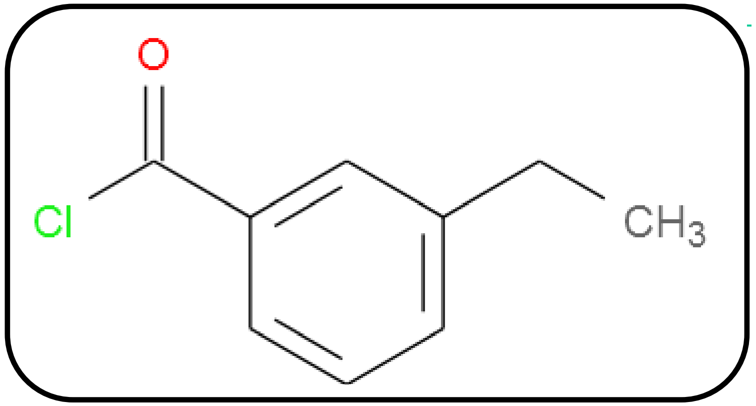
Pro tip #3 (Ab)use the title output format

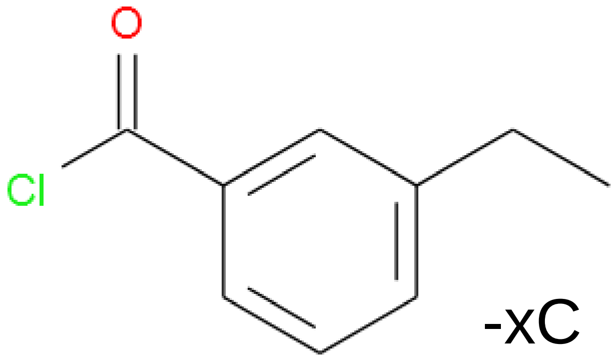
- obabel myfile.sdf –o txt
  - List the titles of all of the molecules
- obabel myfile.sdf –otxt --title “” --append MW
  - List the molecular weights of all of the molecules
- obabel myfile.sdf –otxt --title “” --append My_Property
  - List the property value for all of the molecules

PNG Depiction

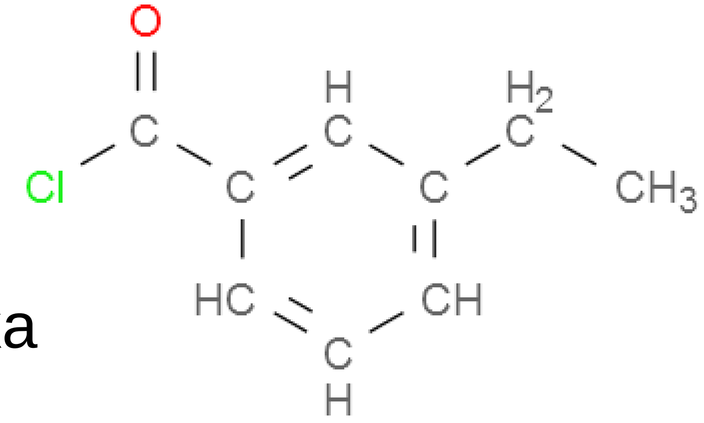


PNG Depiction

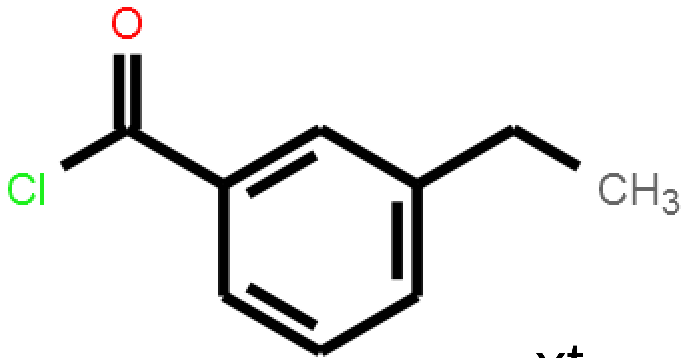




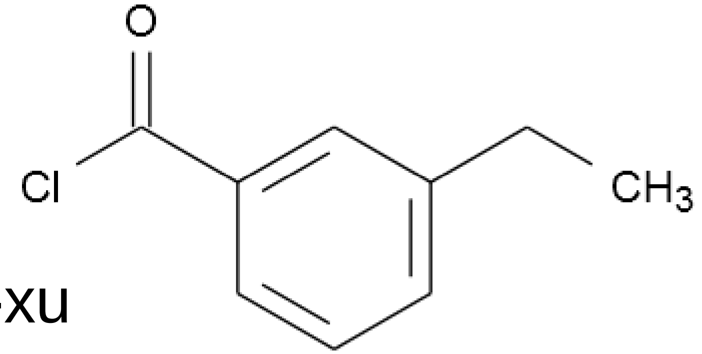
-xC



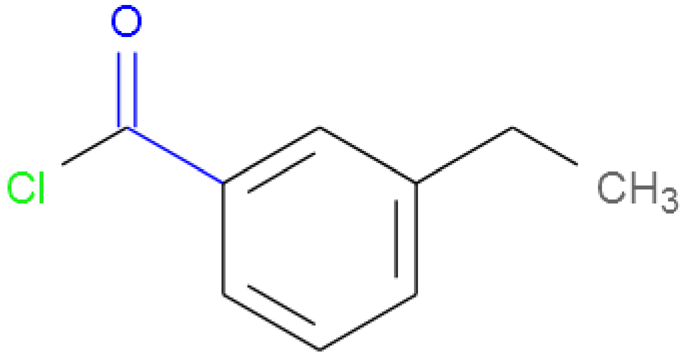
-xa



-xt

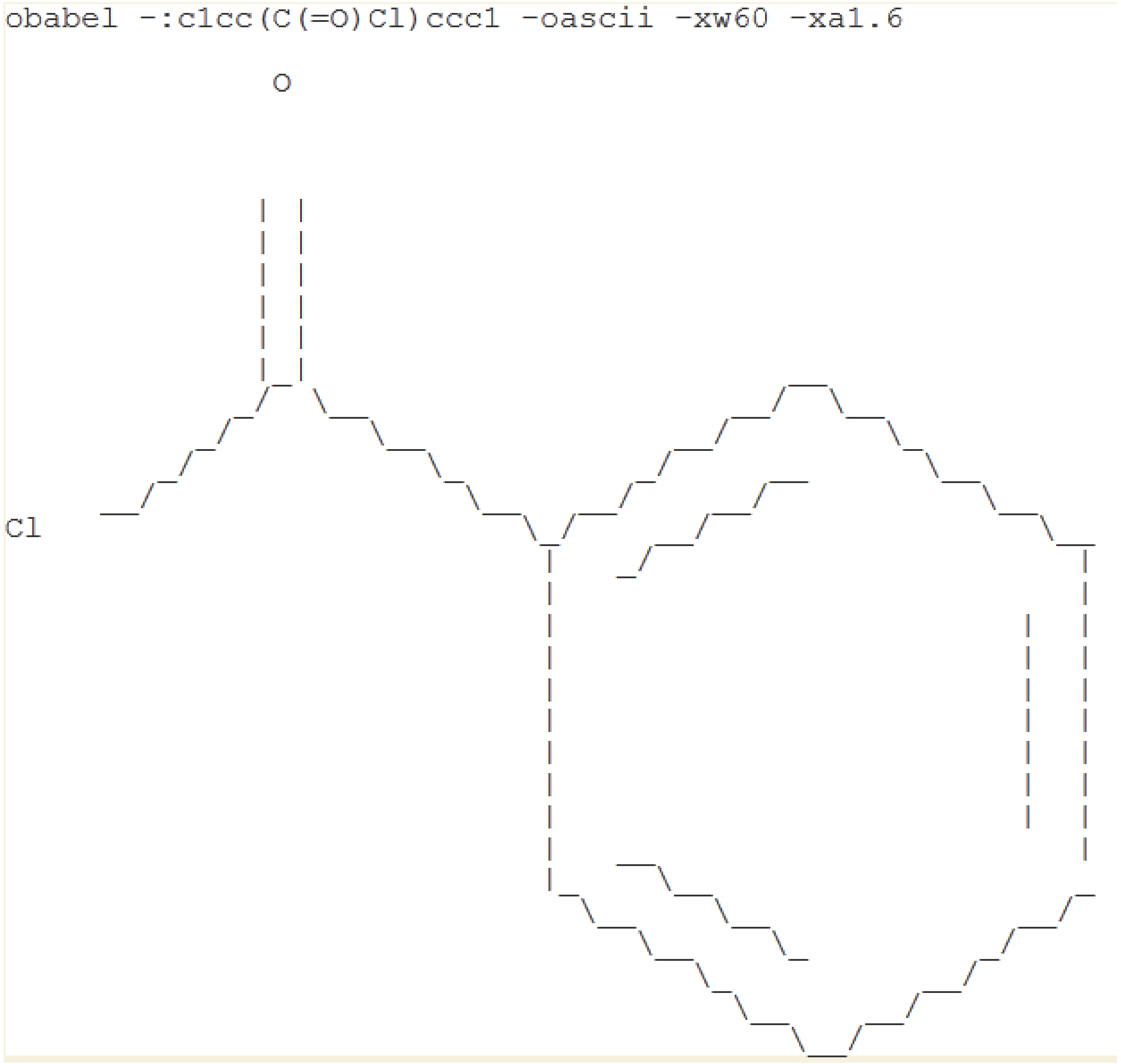


-xu



--highlight “cCO blue”

Ascii Depiction

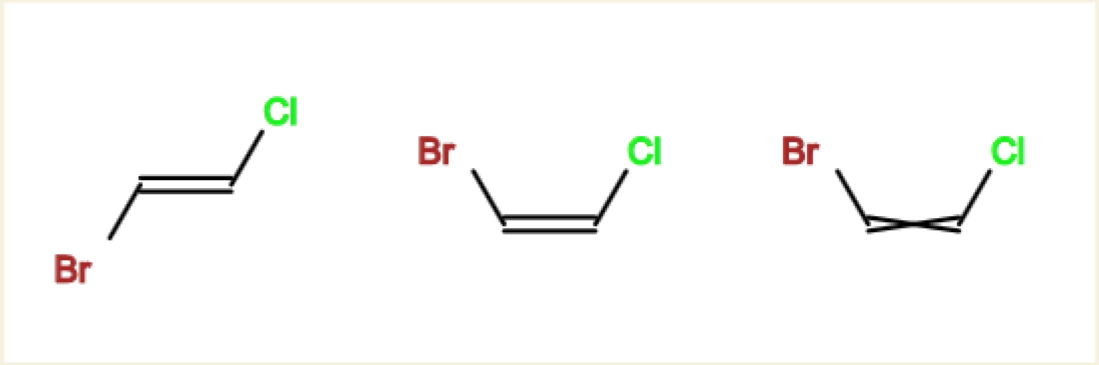


```
obabel -:c1cc(C(=O)Cl)ccc1 -oascii -xw60 -xa1.6
```

Pro Tip #4 SVG + Firefox = User interface

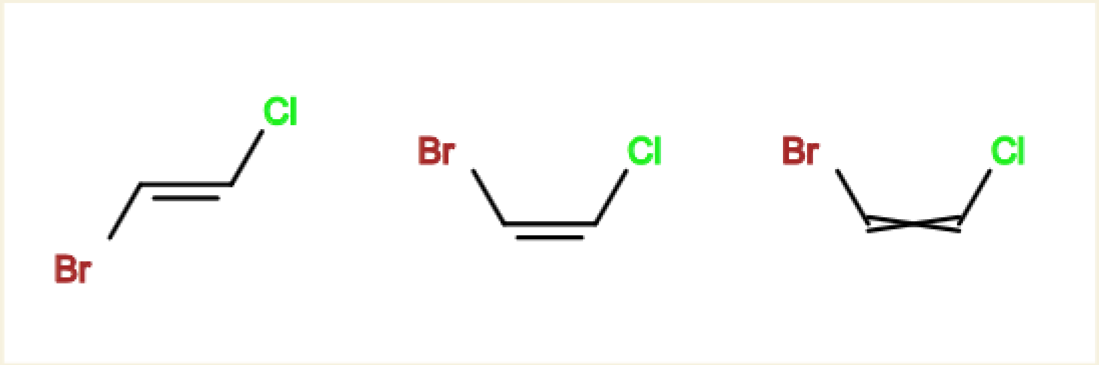
- SVG has same options as PNG…
- …but drag-and-drop onto Firefox and you have a zoomable user interface
  - particularly useful for visualising multimolecule files
  - Demo showing a 1000 molecule file (only 3MB): http://baoilleach.blogspot.co.uk/2011/06/molecular-zooming-with-open-babel-svg.html
- You could create a navigation interface for an entire database (sponsorship opportunity!)
  - E.g. make each of 1000 molecules link to another SVG with 1000 molecules
- Multimolecule depictions can be aligned based on substructure (also PNG)
  - Demo: http://baoilleach.blogspot.co.uk/2012/02/portrait-of-molecule-as-green.html

```
obabel -:"Cl/C=C/Br" -:"Cl/C=C\Br" -:"ClC=CBr"
        -O tmp.svg -xr 1
```



...and with an asymmetric double bond for extra pizazz:

```
obabel -:"Cl/C=C/Br" -:"Cl/C=C\Br" -:"ClC=CBr"
        -O tmp.svg -xr 1 -xs
```



Credits: Twisted double bond by me. Everything else of depiction by Chris Morley and Tim Vandermeersch. Structure layout by Sergei Trepalin.

Pro Tip #5 Automatic conversion

On Windows, create a file sdf.bat on your Desktop with the following text:

```
@obabel.exe %1 -O "%~ndp1.%~n0"
```

If you drag-and-drop a chemical file onto this, the file will be converted to an SDF file.

(Rename to mol2.bat for mol2 files, etc.)

Alignment

- Open Babel does not have any code to determine the maximum common substructure (MCS)
  - Sponsorship opportunity ahoy!

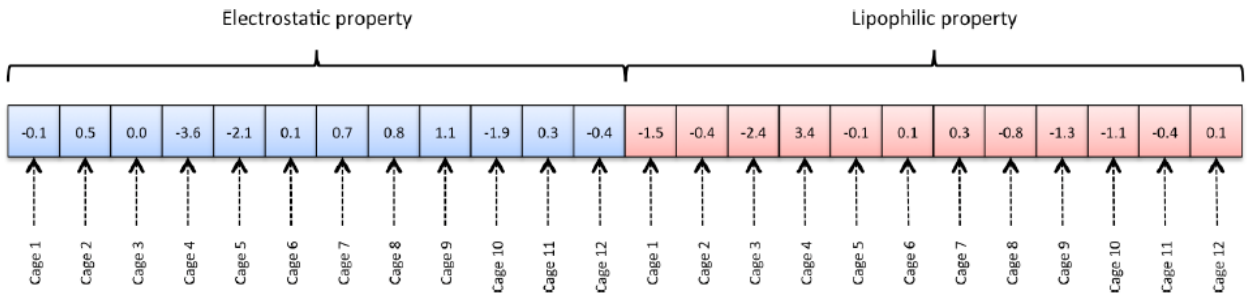
- 2D and 3D alignment is supported –align
  - Based on Kabsch alignment (minimised RMSD)
  - You either have to align the whole molecule (atoms should be in same order) or else a specified substructure (SMARTS)

- When aligning 3D structures I find it useful to --join the results into a single structure and view in 3D viewer (e.g. Avogadro)

Spectrophores

- Donated by Silicos-it, http://silicos-it.com/
- Usage: obspectrophore –i myfile.extn
- Requires 3D structure
  - Note: it does not complain if you give it a 2D structure
  - 3D conformation dependent, but orientation independent

- 48-value descriptor based on electrostatic, lipophilic and electrophilic property values at points on a grid (or cage) and the atomic shape deviation



- Typically, four properties per SPECTROPHORE™, hence 48 points per SPECTROPHORE™

- 1. Electrostatic partial charges
  2. Atomic lipophilic potential
  3. Atomic shape deviations
  4. Atomic electrophilicity potential

Spectrophores

- Donated by Silicos-it, http://silicos-it.com/
- Usage: obspectrophore –i myfile.extn
- Requires 3D structure
  - Note: it does not complain if you give it a 2D structure
  - 3D conformation dependent, but orientation independent

- 48-value descriptor based on electrostatic, lipophilic and electrophilic property values at points on a grid (or cage) and the atomic shape deviation

- Custom code require to use spectrophores for similarity
- Silicos-it have previously trained Self-Organising Maps (SOMs) using spectrophores for known classes of compounds and used them to predict novel compounds for a particular class

Progamming with Open Babel

- Sometimes the GUI or command-line interface does not do exactly what you want
  - You can write your own applications or scripts
- Choice of C++, Python, Java, .NET, Perl
  - But C++ and Python best supported
- Python is well-established in chemistry
  - Relatively easy to learn
  - Small number of commands
  - Can do a lot in a few lines
- Since the full Open Babel library is quite large, to make it easy to get started we provide a Python module **Pybel**
  - Makes it easy to do the most common operations
  - Very small number of classes and functions
  - The full library is still available under-the-hood
- Google “Open Babel Python”

Using the Python Bindings

```
import pybel

# Read a molecule
inputfile = pybel.readfile(“mol”, “tmp.mol”)
mol = next(inputfile)

print(mol.molwt) # Show molecular weight
```

Using the Python Bindings

```
import pybel

# Loop over multiple molecules
inputfile = pybel.readfile(“sdf”, “tmp.sdf”)
for mol in inputfile:
        # Show molecular weight
        print(mol.molwt)
```

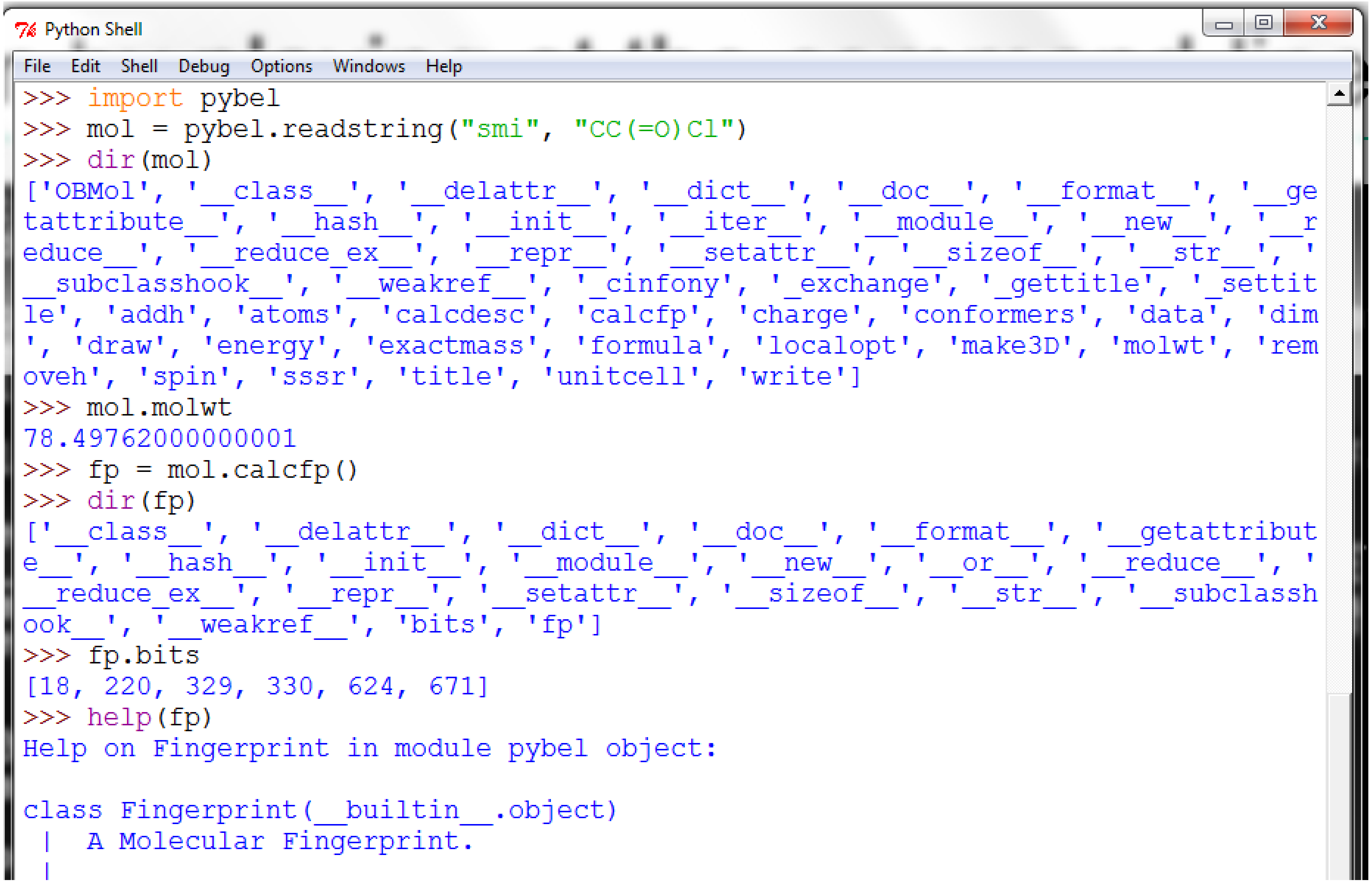
Using the Python Bindings

```
import pybel

# Loop over multiple molecules
inputfile = pybel.readfile(“sdf”, “tmp.sdf”)
for mol in inputfile:
        if (mol.title.endswith(“_active”) and
            mol.wt > 100 and “S” in mol.formula):
                # Show molecular weight
                print(mol.molwt)
```

Using the Python Bindings

```
import pybel

# Loop over multiple molecules
inputfile = pybel.readfile(“sdf”, “tmp.sdf”)
outputfile = pybel.Outputfile(“smi”, “tmp.smi”)
for mol in inputfile:
        if (mol.title.endswith(“_active”) and
            mol.wt > 100 and “S” in mol.formula):
                # Add the molecule to the output file
                outputfile.write(mol)
```

Learn by playing at the command-line



```
>>> import pybel
>>> mol = pybel.readstring("smi", "CC(=O)Cl")
>>> dir(mol)
['OBMol', '__class__', '__delattr__', '__dict__', '__doc__', '__format__', '__ge
tattribute__', '__hash__', '__init__', '__iter__', '__module__', '__new__', '__r
educe__', '__reduce_ex__', '__repr__', '__setattr__', '__sizeof__', '__str__', '
__subclasshook__', '__weakref__', '_cinfony', '_exchange', '_gettitle', '_settit
le', 'addh', 'atoms', 'calcdesc', 'calcfp', 'charge', 'conformers', 'data', 'dim
', 'draw', 'energy', 'exactmass', 'formula', 'localopt', 'make3D', 'molwt', 'rem
oveh', 'spin', 'sssr', 'title', 'unitcell', 'write']
>>> mol.molwt
78.49762000000001
>>> fp = mol.calcfp()
>>> dir(fp)
['__class__', '__delattr__', '__dict__', '__doc__', '__format__', '__getattribut
e__', '__hash__', '__init__', '__module__', '__new__', '__or__', '__reduce__', '
__reduce_ex__', '__repr__', '__setattr__', '__sizeof__', '__str__', '__subclassh
ook__', '__weakref__', 'bits', 'fp']
>>> fp.bits
[18, 220, 329, 330, 624, 671]
>>> help(fp)
Help on Fingerprint in module pybel object:

class Fingerprint(__builtin__.object)
 |  A Molecular Fingerprint.
 |
```

A cry for help

Like mailing lists?

- openbabel-discuss@lists.sf.net

Like forums?

- http://forums.openbabel.org

Like to email a developer directly?

- We will ask you to email the list :-)

Don't forget to read the docs first and Google it

- **http://openbabel.org/docs**

Image: Tintin44 (Flickr)