Subject: Faulty SDF import
Posted by richards99 on Mon, 19 Oct 2020 09:30:37 GMT
View Forum Message <> Reply to Message

Hi,
There are still issues with regards the import of SDF files and its interpretation of the bonding.
This time it is not down to specified aromatic bonds but simply kekulised bonds.

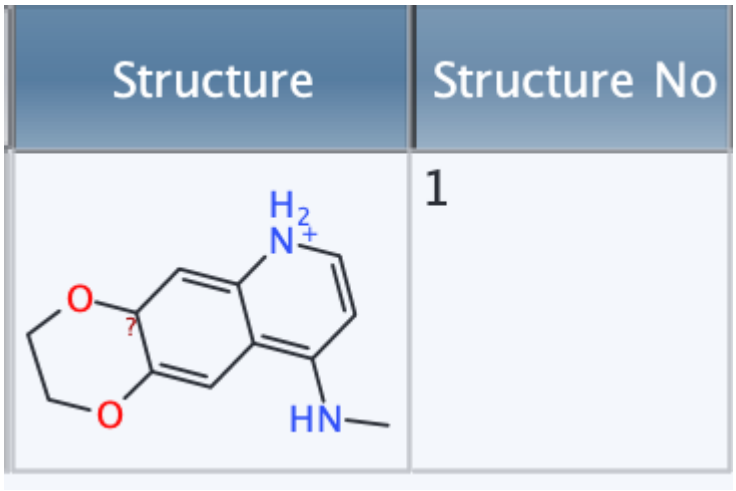
Take for instance this SMILES string of a protonated heterocycle.
CNC1=C2C=C3OCCOC3=CC2=[NH+]C=C1

When this is saved as an SDF file using Marvin for instance, and then imported into DataWarrior the bonding is completely wrong.
I have attached the SDF of this smiles string above.

The output it provides is attached

File Attachments

1) Screenshot 2020-10-19 at 10.28.58.png, downloaded 1071 times



2) untitled_molecule_file.sdf, downloaded 640 times

Subject: Re: Faulty SDF import
Posted by richards99 on Mon, 19 Oct 2020 09:32:14 GMT
View Forum Message <> Reply to Message

I am also using the dev version of DataWarrior where you mentioned there were fixes regard SDF importing, but the issue remains.

Subject: Re: Faulty SDF import
Posted by nbehrnd on Wed, 21 Oct 2020 07:15:38 GMT
View Forum Message <> Reply to Message

My observation differs:

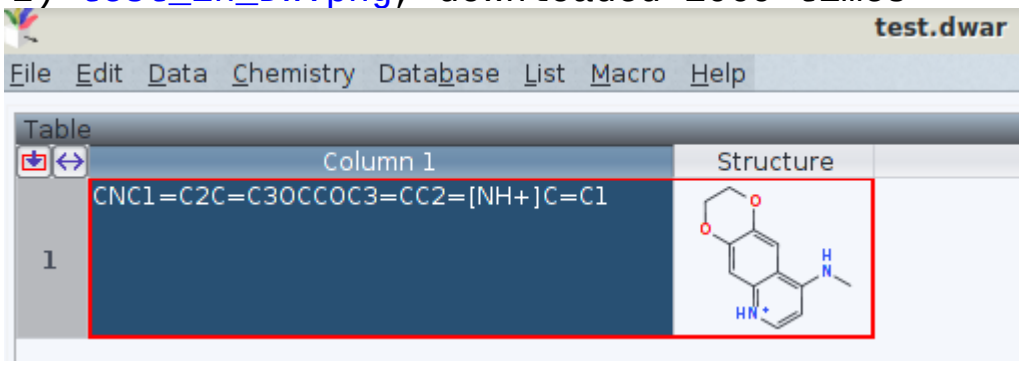
1) First I pasted the SMILES string into DW (5.2.1, natively Linux version [used in Debian 10 / unstable]) and obtained a pyridinium:

2) Then the .sdf provided by you (as generated by Marvin) was read by a separate, newly started instance of DW, again yielding a pyridinium:

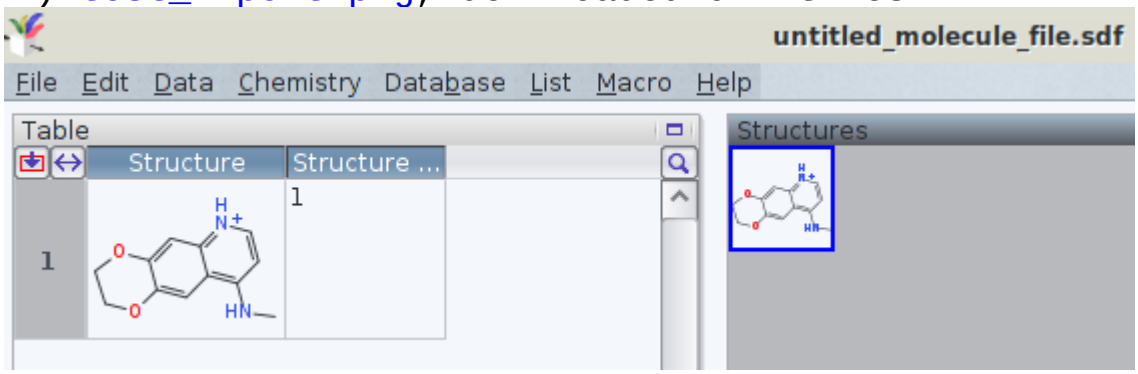
which seems chemically plausible, matches (by constitution) both the former and the visual output when requesting OpenBabel (3.1.0) to visualize the .sdf as a .png:

File Attachments

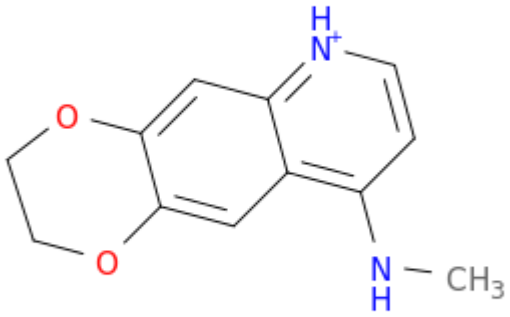
1) test.dwar, downloaded 603 times
2) test_in_DW.png, downloaded 1069 times



3) untitled_molecule_file.dwar, downloaded 617 times
4) test_import.png, downloaded 944 times



5) untitled_molecule_file.png, downloaded 1149 times



Subject: Re: Faulty SDF import
Posted by richards99 on Wed, 21 Oct 2020 09:54:34 GMT
View Forum Message <> Reply to Message

It seems the standard download of DW does display it correctly, but the DEV release of DW is where the structure is displayed incorrectly.
This post (http://openmolecules.org/forum/index.php?t=msg&th=385&am p;start=0&)mentions SDF importing has been fixed in the DEV release, but I think this has introduced a new error.

Subject: Re: Faulty SDF import
Posted by nbehrnd on Thu, 22 Oct 2020 10:28:02 GMT
View Forum Message <> Reply to Message

Affirmative. The pristine installation of DW for Linux using the stable release (last edit 23 Feb 2020) loads your .sdf file and then displays the single protonated pyridinium, as described in my initial post. But the substitution of elder / original datawarrior.jar file by the one obtained from the development branch for Linux (1), deposit into the /opt/datawarrior directory (last edit 13 Sep 2020), then yields the questionable double-protonation on the pyridine; which matches 1:1 your initial observation.

Because rebuilding DW today (22 Okt 2020) from its public sources (2, last edit 13 Sep 2020) in Linux equally yields an executable displaying again the erroneous «double pyridine protonation», possibly the issue spreads out to the development version for Windows,(3) too.

Norwid

(1) http://www.openmolecules.org/datawarrior/dw521x.zip

(2) http://www.openmolecules.org/datawarrior/dw521source.tar.gz
(3) http://www.openmolecules.org/datawarrior/dw521win.zip

Subject: Re: Faulty SDF import
Posted by thomas on Fri, 23 Oct 2020 08:13:40 GMT
View Forum Message <> Reply to Message

Dear All,

there has been a bug in OpenChemLib regarding open valences of charge atoms in certain situations, which seems to have caused the issue. DataWarrior stores structures in a canonical representation. Therefore, it needs to resolve delocalized bonds into alternating single/double bonds when displaying. Because of the valence bug the charged nitrogen was not seen as having the needed open valence. The bug was fixed Sep. 30 in OpenChemLib, but took till yesterday to materialize in a DataWarrior build.

Thank you all for reporting.

Thomas
