General requirements.

Python™ 3.0 or higher is required to be installed on your computer (freely available from www.python.org).

Input file.

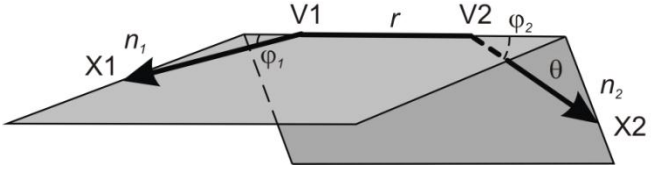
An input file for the program is SDF containing the following information for each representative of the library (see example provided):

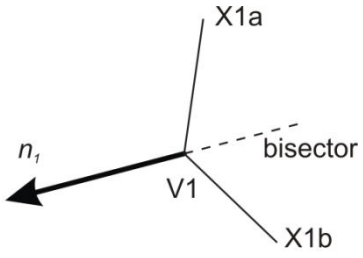
1. Structure in MDL Molfile format containing 3D coordinates of atoms.
2. Compound Id
3. Numbers of atoms V1, V2 (two variation points) and X1, X2 (substituents attached to them) in the above-mentioned MDL Molfile. These atoms are used to define vectors $\boldsymbol{n_1}$ and $\boldsymbol{n_2}$ necessary to calculate EVP parameters.

   There are two ways to define the direction of vector $\boldsymbol{n_1}$ (the same for $\boldsymbol{n_2}$):

   a) It can be colinear to vector **V1X1**, where X1 is exocyclic atom directly attached to the variation point V1 of the scaffold. This approach is commonly used if V1 is carbon or any other atom which is configurationally stable. In this case, the field <X1> of SDF contains a single value – the number of X1.

   b) They can be colinear to the bisector of the angle X1a–V1–X1b, where X1a and X1b– two atoms of the scaffold directly connected to the variation point V1. This approach is commonly used if V1 is nitrogen or other atom which is configurationally unstable. Therefore, the vector $\boldsymbol{n_1}$ defines an average orientation of the substituent attached to the variation point. In this case, the field <X1> of SDF contains two values separated by semicolon (;) – numbers of atoms X1a and X1b.





These data can be generated using any common chemoinformatics software. In particular, to generate example file, we have imported MOL files with 3D coordinates for each compound into Instant JChem database,

and then filled the fields Id, V1, V2, X1, X2 manually using built-in atom numbering tools. Note that order of fields is important.

See also an example provided.

Running the program.

1. Copy the file *EVP.py* into the directory where your SDF input is located.
2. Double click on the file *EVP.py*.
3. Enter the name of SDF with extension (*SDF_example.sdf*)
4. Enter the number of compounds in SDF (*9*).
5. Enter the name of the field containing Id of the compound (*Id*).
6. The program creates file *Results.txt* with results.

Calculation protocol.

For each compound, the program does the following:

1. Extracts 3D coordinates (*x*, *y*, *z*) of atoms V1, V2, X1 and X2 (or X1a,b and X2a,b, see above) using information available in SDF.
2. Calculates coordinates of vector **V1V2** (same as vector ***r***) as following:

$$x_r = x_{V2} - x_{V1}$$

$$y_r = y_{V2} - y_{V1}$$

$$z_r = y_{V2} - y_{V1}$$

3. Calculates coordinates of vector $\boldsymbol{n_1}$ as following:

   a) if a single value is present in the field <X1>, then

$$x_{n1} = x_{X1} - x_{V1}$$

$$y_{n1} = y_{X1} - y_{V1}$$

$$z_{n1} = y_{X1} - y_{V1}$$

   b) if two values are present in the field <X1>, then

$$x_{V1X1a} = x_{X1a} - x_{V1}$$

$$y_{V1X1a} = y_{X1a} - y_{V1}$$

$$z_{V1X1a} = z_{X1a} - z_{V1}$$

$$x_{V1X1b} = x_{X1b} - x_{V1}$$

$$y_{V1X1b} = y_{X1b} - y_{V1}$$

$$z_{V1X1b} = z_{X1b} - z_{V1}$$

$$x_{n1} = -\frac{x_{V1X1a}}{\sqrt{(x^2_{V1X1a} + y^2_{V1X1a} + z^2_{V1X1a}}} - \frac{x_{V1X1b}}{\sqrt{(x^2_{V1X1b} + y^2_{V1X1b} + z^2_{V1X1b}}}$$

$$y_{n1} = -\frac{y_{V1X1a}}{\sqrt{(x^2_{V1X1a} + y^2_{V1X1a} + z^2_{V1X1a}}} - \frac{y_{V1X1b}}{\sqrt{(x^2_{V1X1b} + y^2_{V1X1b} + z^2_{V1X1b}}}$$

$$z_{n1} = -\frac{z_{V1X1a}}{\sqrt{(x^2_{V1X1a} + y^2_{V1X1a} + z^2_{V1X1a}}} - \frac{z_{V1X1b}}{\sqrt{(x^2_{V1X1b} + y^2_{V1X1b} + z^2_{V1X1b}}}$$

4. Calculates coordinates of vector $\boldsymbol{n_2}$ by the same equations as described above for $\boldsymbol{n_1}$ (except V2 and X2 or X2a,b are used instead of V1 and X1 or X1a,b).

5. Calculates parameter *r* using the following equation:

$$r = \sqrt{x_r^2 + y_r^2 + z_r^2}$$

6. Calculates parameters φ_1 and φ_2:

$$\varphi_1 = 180° - \arccos \frac{x_{n1}x_r + y_{n1}y_r + z_{n1}z_r}{r \cdot \sqrt{x^2_{n1} + y^2_{n1} + z^2_{n1}}}$$

$$\varphi_2 = \arccos \frac{x_{n2}x_r + y_{n2}y_r + z_{n2}z_r}{r \cdot \sqrt{x^2_{n2} + y^2_{n2} + z^2_{n2}}}$$

7. Calculates boolean parameter $\varphi_1\_\varphi_2$, which is true (1) if $\varphi_1 \geq \varphi_2$ and false (0) if $\varphi_1 < \varphi_2$.

8. Calculates parameter θ:

$$x_{A1} = x_{n1} - \frac{(x_r x_{n1} + y_r y_{n1} + z_r z_{n1}) \cdot x_r}{r^2}$$

$$y_{A1} = y_{n1} - \frac{(x_r x_{n1} + y_r y_{n1} + z_r z_{n1}) \cdot y_r}{r^2}$$

$$z_{A1} = z_{n1} - \frac{(x_r x_{n1} + y_r y_{n1} + z_r z_{n1}) \cdot z_r}{r^2}$$

$$x_{A2} = x_{n2} - \frac{(x_r x_{n2} + y_r y_{n2} + z_r z_{n2}) \cdot x_r}{r^2}$$

$$y_{A2} = y_{n2} - \frac{(x_r x_{n2} + y_r y_{n2} + z_r z_{n2}) \cdot y_r}{r^2}$$

$$z_{A2} = z_{n2} - \frac{(x_r x_{n2} + y_r y_{n2} + z_r z_{n2}) \cdot z_r}{r^2}$$

$$\theta = \arccos \frac{x_{A1}x_{A2} + y_{A1}y_{A2} + z_{A1}z_{A2}}{\sqrt{(x_{A1}^2 + y_{A1}^2 + z_{A1}^2)(x_{A2}^2 + y_{A2}^2 + z_{A2}^2)}}$$

9. Calculates the sign_θ parameter (since the above calculation gives absolute value), sign_θ = 1 if $\theta > 0$; 0 if $\theta = 0$; –1 if $\theta < 0$.

$$\text{sign\_}\theta = \text{sign}((z_{A2}y_{A1} - y_{A2}z_{A1})\cdot x_r + (x_{A2}z_{A1} - z_{A2}x_{A1})\cdot y_r + (y_{A2}x_{A1} - x_{A2}y_{A1})\cdot z_r)$$

Output format.

The calculated data are exported to the TXT table with the following format of each line:

Id r φ_1 φ_2 θ φ_1_φ_2 sign_θ

The first line is a header of the table.

NOTE: the results are sorted by ID value.