py.Aroma 4

A Multi-Functional Tool for Theoretical Analyses of Aromaticity

Program User Manual

Version 4.0.0, build 3116.

– Developer –
Zhe Wang, *Ph.D.*

[Homepage] https://wongzit.github.io/program/pyaroma

1 Overview **1**
1.1 About 1
1.2 Running Program 1
1.3 Startup 2

2 General **4**
2.1 File System 4
2.2 Program Setting Panel 5
2.3 Calculation Setup Panel 5

3 Functions **7**
3.1 Bond Length Alternation (BLA) 7
3.1.1 Theory 7
3.1.2 Usage 7
3.2 Calculate HOMA Index 8
3.2.1 Theory 8
3.2.2 Usage 9
3.3 Calculate POAV Index 10
3.3.1 Theory 10
3.3.2 Usage 10
3.4 NICS Calculation 11
3.4.1 Theory 11
3.4.1.1 Single Point NICS 11
3.4.1.2 NICS Scan 11
3.4.1.3 2D and 3D NICS Analysis 12
3.4.1.4 Integral NICS 13
3.4.2 Create Input File for NICS Calculation 14
3.4.2.1 Single Point NICS 14
3.4.2.2 NICS Scan 14
3.4.2.3 2D and 3D NICS 14
3.4.2.4 INICS 15
3.4.3 Process Output File of NICS Calculation 15
3.4.3.1 Single Point NICS 15
3.4.3.2 NICS Scan 15
3.4.3.3 2D NICS 16
3.4.3.4 3D NICS 16
3.4.3.5 INICS 16
3.5 NMR Analysis 17

3.5 Generate Computational Supporting Material 18

4 Tutorials and Examples 19

4.1 BLA Analysis 19

4.1.1 Basic Information 19

4.1.2 BLA Analysis of 1,3,5,7-Octatetraene 20

4.1.3 BLA Analysis of Oligo(biindenylidene) 21

4.2 HOMA Analysis 22

4.2.1 Basic Information 22

4.2.2 HOMA Analysis of Triphenylene 23

4.3 POAV Analysis 25

4.3.1 Basic Information 25

4.3.2 POAV Analysis of Fullerene C_{60} 25

4.4 Create Input File for NICS 27

4.4.1 Basic Information 27

4.4.2 Example: DR-4CPP 28

4.5 Create Input File for NICS Scan 30

4.5.1 Basic Information 30

4.5.2 Example: Fluorene 30

4.6 Create Input File for 2D NICS 32

4.6.1 Basic Information 32

4.6.2 Example: AWA Molecule 32

4.7 Create Input File for 3D NICS 34

4.8 Create Input File for INICS 35

4.8.1 Basic Information 35

4.8.2 Example: Terphenyl 35

4.9 Analyze Output Files for NICS Calculation 37

4.9.1 Basic Information 37

4.9.2 Single Point NICS 37

4.9.3 NICS Scan 39

4.9.4 2D NICS/ICSS 40

4.9.5 3D NICS/ICSS 41

4.9.6 INICS 41

4.10 NMR Plotting 43

4.10.1 Main Function 43

4.10.2 NMR Spectrum 44

4.11 Generate Computational Supporting Material 45

Appendix **47**

I Colormaps in *Matplotlib* 47

II Colors in *Matplotlib* 49

III Accuracy of Integral NICS 51

IV Lorentzian Line Broadening in NMR Spectrum 52

Preface

As a researcher, I found it is important to make some contribution. py.**Aroma 4** is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

If you found any problem or bug, it would be helpful if you can contact me via E-mail (wang.zhe.dr@gmail.com). I am glad to help on your problems and your bug report will help me to make py.**Aroma 4** better.

For bug report, please include following information as much as possible:

(1) OS type and version.

(2) Source code mode?

(2) Is the bug/error reproducible? How to reproduce the bug?

(3) Did the bug make program crashing or give wrong/unnatural result?

(4) If possible, please attach the file(s) when the bug occurred.

I am also very happy to accept request for new functions and add them to new version of py.**Aroma**. Please feel free to contact me via E-mail.

If py.**Aroma 4** is utilized in your work, or the code is implied in your own code, please consider citing following contents:

(1) Yuki Miyazawa, Zhe Wang, Misaki Matsumoto, Sayaka Hatano, Ivana Antol, Eiichi Kayahara, Shigeru Yamago, Manabu Abe, *Journal of the American Chemical Society*, **2021**, *143*(*19*), 7426–7439.

(2) Zhe Wang, py.Aroma 4, A https://wongzit.github.io/program/pyaroma (accessed *data*, *month*, *year*).

[illegible]

Zhe WANG, *Ph.D.*
WPI-iCeMS, Kyoto University, Japan.
Dec. 2023

1 Overview

1.1 About

py.**Aroma 4** is a multi-functional tool for aromaticity analysis. It supports BLA, HOMA, POAV and NICS analysis. In addition, py.**NMR**[1] and **CSIgen**[2] have been introduced to py.**Aroma 4**, users can also plot NMR spectrum and generate computational supporting information.

Source code of py.**Aroma 4** including follows file:

`pyaroma_main.py`: Main code of py.**Aroma 4**.
`CONSTANT.py`: Including some constants like atom radius, HOMA parameters, *etc*.
`readFile.py`: Read input files and extract geometry.
`geomAnalyzer.py`: Find connectivity and cyclic unit in molecules.
`NICSInp.py`: Create NICS input files.
`NICSout.py`: Extract shielding tensors from NICS output files.
`homaCalc.py`: Compute HOMA values.
`pathCreator.py`: Create path for NICS scan.
`poav.py`: Compute POAV1 and POAV2.
`pynmr.py`: Calculate chemical shift and plot NMR spectrum.
`config.ini`: Including program setting parameters.
`assets`: A folder including necessary images.

1.2 Running Program

py.**Aroma 4** is developed using Python 3.11 and utilizes PyQt6 for the GUI. In order to launch py.**Aroma 4**, it is necessary to have third-party libraries: NumPy, Matplotlib, NetworkX, and OpenPyXL installed. The source code has been tested on macOS 14.1, Microsoft Windows 11 Pro (Update 23H2) and Red Hat Enterprise Linux 8.8.

To run py.**Aroma 4** from source code, users will need to download all the files in the "*src*" folder to computer (*e.g.*, the "*src*" folder is located at "*/home/wangzhe/py.Aroma/src*") and execute the command `python  /home/wangzhe/py.Aroma/src/pyaroma_main.py` in the *Terminal* or *PowerShell*, *etc*. Before running py.**Aroma 4** in source-code mode for the first time, ensure that Python[3] (3.10 or newer is recommended) and all necessary libraries are already installed on your computer[4].

Pre-packaged executable files for macOS, Microsoft Windows and Linux (tested on RHEL) are

[1] https://github.com/wongzit/pyNMR
[2] https://github.com/wongzit/CSIgen
[3] Free available from Python homepage: python.org
[4] Run command `pip install pyqt6 numpy matplotlib networkx openpyxl` in terminal.

available from py.**Aroma 4** homepage.

For macOS users, please save the *py.Aroma 4.app* in py.Aroma-4.0.0-macOS.dmg to `/Applications` folder, otherwise the program may not work normally. For mac computers powered by Apple Silicon CPUs, please force py.**Aroma 4** run in Rosetta 2 mode as following procedure:

(1) Find py.**Aroma 4** in `/Applications` folder.

(2) Select py.**Aroma 4**, and then press *Command+I* (or right-click/user the File menu and select *Get Info.*). This will open an Info window with details about py.**Aroma 4**.

(3) In the Info window, look for a checkbox labeled "Open using Rosetta". Check the box.

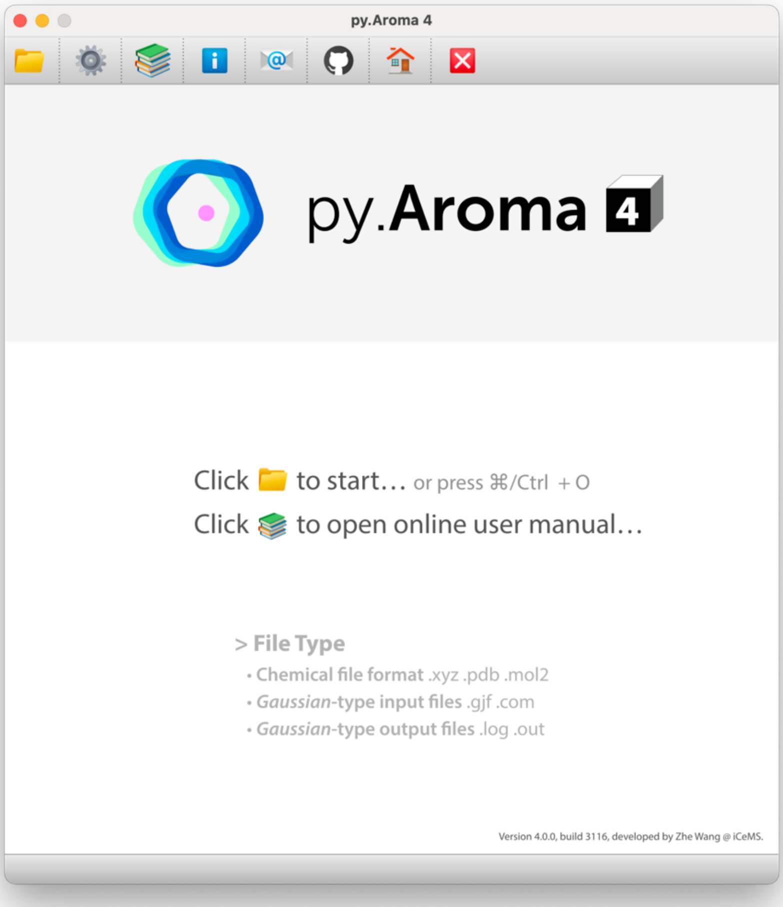
(4) Close the Info window.

(5) If you are already running py.**Aroma 4**, quit and relaunch.

For Microsoft Windows users, please launch the ***pyAroma-4.exe*** file by double-clicking. As for Linux, run the executable file ***py.Aroma-4*** in the folder.

1.3 Startup

Once py.**Aroma 4** is launched, users will see the follows window, which includes a task bar and a brief introduction about "how to quick start".



Click 📁 to start... or press ⌘/Ctrl + O
Click 📚 to open online user manual...

> File Type
· Chemical file format .xyz .pdb .mol2
· *Gaussian*-type input files .gjf .com
· *Gaussian*-type output files .log .out

Version 4.0.0, build 3116, developed by Zhe Wang @ iCeMS.

Open: Browse a file to open and process by py.**Aroma 4**.
Settings: Open py.**Aroma 4** setting panel.
Documents: Open online user manual (need Internet connection).
About: About py.**Aroma 4**.
Contact: Contact developer via e-mail.
GitHub: Check the GitHub page of developer.
Homepage: Open the homepage of py.**Aroma 4**.
Quit: Quit py.**Aroma 4**.

2 General

This chapter introduces some basic information about py.**Aroma 4**.

2.1 File System

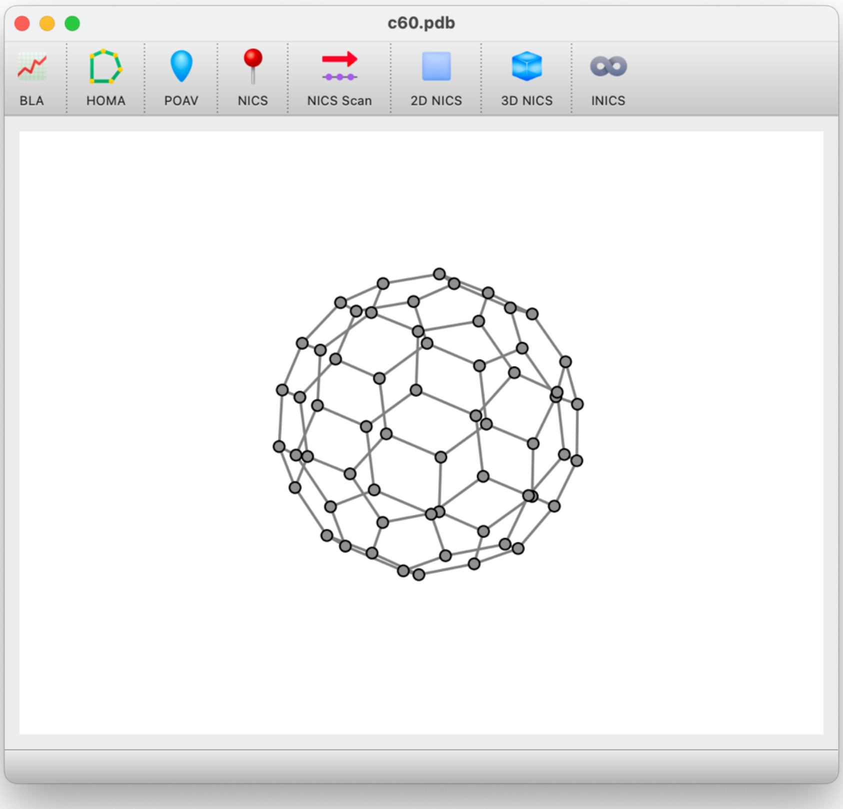
Three types of files could be read by py.**Aroma 4**:

Type-I: Chemical file format: *.pdb*, *.xyz*, *.mol2*; *Gaussian*-type input file: *.gjf*, *.com*.

Type-II: *Gaussian*-type output file *.log* or *.out* without ghost atom(s).

Type-III: *Gaussian*-type output file *.log* or *.out* with ghost atom(s).

Different functions would be active when reading different files.



Functions	Icon	Type-I	Type-II	Type-III
Bond Length Alternation (BLA)	📈	○	○	○
HOMA	⬠	○	○	○
POAV	💧	○	○	○
Create input file for single point NICS	📍	○	○	–
Create input file for NICS scan	➡	○	○	–
Create input file for 2D NICS/ICSS	🟦	○	○	–

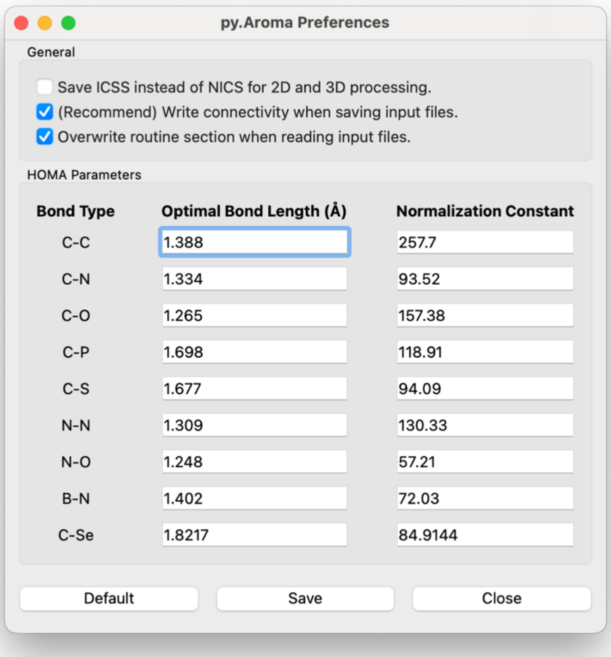
Continue

Functions	Icon	Type-I	Type-II	Type-III
Create input file for 3D NICS/ICSS	🧊	○	○	–
Create input file for INICS	∞	○	○	–
Plot NMR Spectrum	🧪	–	△[a]	○
Generate computational supporting information	📝	–	○	–
Analyze output of single point NICS	🔍	–	–	△[b]
Analyze output of NICS scan	🔍	–	–	△[b]
Analyze output of 2D NICS/ICSS	🔍	–	–	△[b]
Analyze output of 3D NICS/ICSS	🔍	–	–	△[b]
Analyze output of INICS	🔍	–	–	△[b]

[a]Need *Gaussian* output files of NMR calculation. [b]Depend on file name.

2.2 Program Setting Panel

The program setting panel offers common options and HOMA parameters. If the settings have been changed and saved, they will not revert to the default even if the py.**Aroma 4** is restarted. To return to the default settings, the user needs to click "Default" and "Save".



General

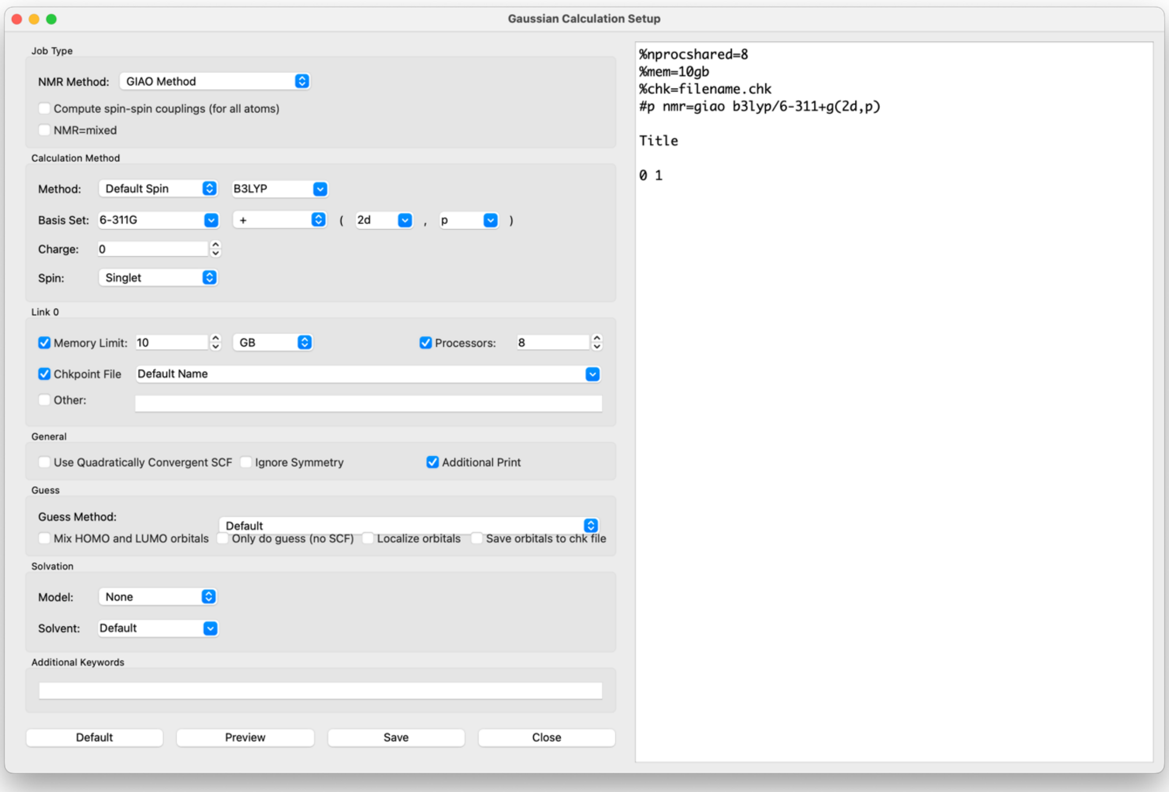
- [ ] Save ICSS instead of NICS for 2D and 3D processing.
- [✓] (Recommend) Write connectivity when saving input files.
- [✓] Overwrite routine section when reading input files.

HOMA Parameters

Bond Type	Optimal Bond Length (Å)	Normalization Constant
C-C	1.388	257.7
C-N	1.334	93.52
C-O	1.265	157.38
C-P	1.698	118.91
C-S	1.677	94.09
N-N	1.309	130.33
N-O	1.248	57.21
B-N	1.402	72.03
C-Se	1.8217	84.9144

2.3 Calculation Setup Panel

py.**Aroma 4** provides a calculation setup panel, which can be accessed when creating a *Gaussian*-type input file by clicking the "Calculation Setup" button.



Gaussian Calculation Setup

Job Type

NMR Method: GIAO Method

☐ Compute spin-spin couplings (for all atoms)

☐ NMR=mixed

Calculation Method

Method: Default Spin | B3LYP

Basis Set: 6-311G | + | (2d , p)

Charge: 0

Spin: Singlet

Link 0

☑ Memory Limit: 10 GB | ☑ Processors: 8

☑ Chkpoint File: Default Name

☐ Other:

General

☐ Use Quadratically Convergent SCF ☐ Ignore Symmetry ☑ Additional Print

Guess

Guess Method: Default

☐ Mix HOMO and LUMO orbitals ☐ Only do guess (no SCF) ☐ Localize orbitals ☐ Save orbitals to chk file

Solvation

Model: None

Solvent: Default

Additional Keywords

```
%nprocshared=8
%mem=10gb
%chk=filename.chk
#p nmr=giao b3lyp/6-311+g(2d,p)

Title

0 1
```

In the calculation setup panel, users can specify the NMR method, computational level, Link 0 section, solvation model, and more. The functionality of this calculation setup panel is similar to the "Calculation Setup" in *GaussView*. Users can preview the routine section by clicking the "Preview" button.

Note: If users want to check the input files created via py.**Aroma 4** by *GaussView*, to prevent *GaussView* from automatically creating bonds with ghost atoms, it is strongly recommended to check the "Write connectivity when saving input file" option in the program setting panel (*Section 2.2*), otherwise it might take extremely longer to open the input file by *GaussView*.

3 Functions

This chapter provides a detailed introduction to all the functions of py.**Aroma 4**. Different functions require different types of input files (**Type-I**, **Type-II** and **Type-III**), thus, please choose the proper type of files according to *Section 2.1*.

3.1 Bond Length Alternation (BLA)

☞Supported file types: **Type-I**, **Type-II** and **Type-III**

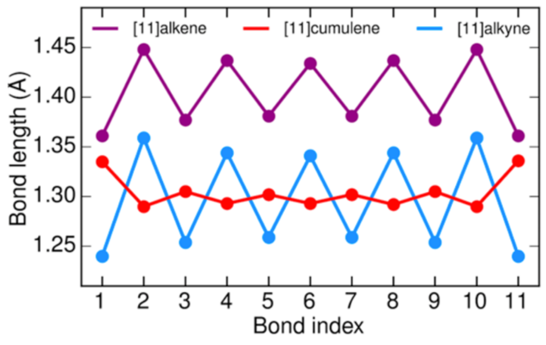
3.1.1 Theory

In conjugated systems, the bond lengths in the conjugation chain exhibit alternant character. Bond length alternation (BLA) is a crucial quantity in the study of conjugated molecules. Generally, BLA is defined as equation:

$$\mathrm{BLA} = \bar{R}_{\mathrm{even}} - \bar{R}_{\mathrm{odd}}$$

as the difference between the average length of even bonds and odd bonds. A smaller BLA magnitude implies better electron conjugation along the selected path.

A more effective method for investigating BLA is by plotting bond length against bond index: the smaller the change in bond length, the smaller the BLA value. To plot a bond length versus bond index graph, the bond sequence in the conjugated chain must be provided. An example from *J. Phys. Chem. C*, **2018**, *122*, 26777:



3.1.2 Usage

To utilize this function, a bond sequence must be defined. Upon entering the BLA function of py.**Aroma 4**, the user will be prompted to input the indices of the bonds that make up the sequence. Subsequently, based on the sequence, py.**Aroma 4** will automatically calculate the bond length for those specified bonds and plot a bond length versus bond index graph in the BLA window. Meanwhile, the BLA value will also be calculated and displayed on the graph. The graph can be saved as a *.png* file, and the data can be exported to *.txt* and Excel *.xlsx* formats, which can then be imported into data processing software such as *Origin* and *Prism*, *etc*. The *.png*, *.txt*, and *.xlsx* files will be saved with the same file name as the input file, with the addition of the "_BLA" suffix.

An example of using BLA function in py.**Aroma 4** is given in *Section 4.1.2* and *4.1.3*.

3.2 Calculate HOMA Index

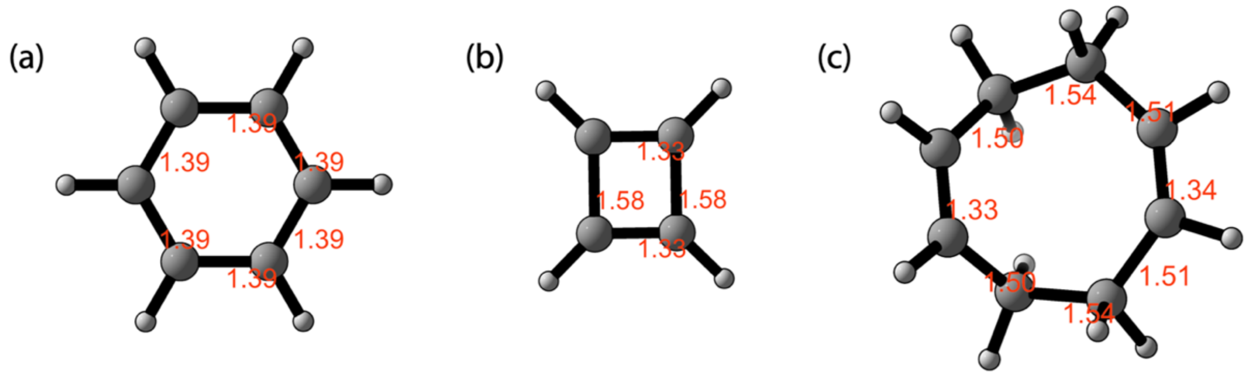
☞Supported file types: **Type-I**, **Type-II** and **Type-III**

3.2.1 Theory

Harmonic oscillator model of aromaticity (HOMA) is one of the most popular indexes for evaluating aromaticity. This quantity was originally proposed in *Tetrahedron Lett.*, **1972**, *13*, 3839, and then the generalized form was given in *J. Chem. Inf. Comput. Sci.*, **1993**, *33*, 70. The generalized HOMA can be written as:

$$\text{HOMA} = 1 - \sum_i \frac{\alpha_{i,j}}{N}\left(R_{\text{opt}} - R_{i,j}\right)^2$$

where N is the total number of the atoms (bonds) considered, j denotes the atom next to atom i, R_{opt} and α are pre-calculated constants for each type of atom pair. If HOMA equals to 1, that means length of each bond is identical to the optimal value R_{opt} and thus the ring is fully aromatic. While if HOMA equals to 0, that means the ring is completely nonaromatic. If HOMA is a significant negative value, the ring shows antiaromatic character. For example, HOMA indexes of benzene, cyclobutadiene and cyclooctadiene are 0.99603 (aromatic), –3.96177 (antiaromatic) and –2.40591 (nonaromatic), respectively.



*Geometries were optimized at (R)B3LYP/6-311+G(2d,p) level of theory, orange values indicate bond lengths in Å.

The default R_{opt} and α parameters are sourced from *Chem. Rev.,* **2014**, *114*, 6383. Users have the option to modify these parameters from the program setting panel (*Section 2.2*). Please note than only the follows bond types are supported.

Bond Type	R_{opt}	α	**Bond Type**	R_{opt}	α
C–C	1.388	257.7	N–N	1.309	130.33
C–N	1.334	93.52	N–O	1.248	57.21
C–O	1.265	157.38	B–N	1.402	72.03
C–P	1.698	118.91	C–Se	1.8217	84.9144
C–S	1.677	94.09			

3.2.2 Usage

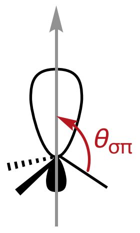
Upon entering the HOMA function, py.**Aroma 4** will automatically calculate the HOMA indexes for all chord less monocyclic structures in the input geometry. Users can also obtain the HOMA index for a specified cycle by inputting a sequence of atomic indices, with the input order being consistent with atom connectivity. An example of using the HOMA function in py.**Aroma 4** is provided in *Section 4.2.2*.

3.3 Calculate POAV Index

☞Supported file types: **Type-I**, **Type-II** and **Type-III**

3.3.1 Theory

The π-orbital axis vector (POAV) analysis offers a comprehensive understanding of the electronic structure of nonplanar conjugated molecules. The POAV1 theory provides $\theta_{\sigma\pi}$, θ_p (= $\theta_{\sigma\pi} - 90^\circ$), *m* and *n* values:



$$m = \frac{2\cos^2\theta_{\sigma\pi}}{1 - 3\cos^2\theta_{\sigma\pi}}$$

$$n = 3m + 2$$

where *m* indicates the s-orbital content ($s^m p$) of the π-orbitals and *n* indicates the p-orbital (sp^n) content of the σ-orbitals.

The POAV2 hybridization provides $\theta_{1\pi}$, $\theta_{2\pi}$, $\theta_{3\pi}$, n_1, n_2, n_3 and *m*. More details about the POAV1 and POAV2 theory can be found in *J. Am. Chem. Soc.*, **1986**, *108*, 2837 and *J. Phys. Chem. A*, **2001**, *105*, 4164. The one-center POAV1 and POAV2 function in py.**Aroma 4** is based on the three sigma bond angles.

For an ideal sp^2 hybridized carbon atom, $\theta_{1\pi} = \theta_{2\pi} = \theta_{3\pi} = 120^\circ$, $\theta_{\sigma\pi} = 90^\circ$, $\theta_p = 0^\circ$, $m = 0$ and $n = n_1 = n_2 = n_3 = 2$.

3.3.2 Usage

Calculating POAV1 and POAV2 parameters using py.**Aroma 4** is simple. Users can input the index of the target atom, and the program will automatically compute and display the POAV1 and POAV2 parameters in the window. It's important to note that the inputted atom MUST have and ONLY have three bonded neighboring atoms. An example of using POAV function in py.**Aroma 4** is given in *Section 4.3.2*.

3.4 NICS Calculation

3.4.1 Theory

This section will provide some basic information about NICS calculation.

3.4.1.1 Single Point NICS

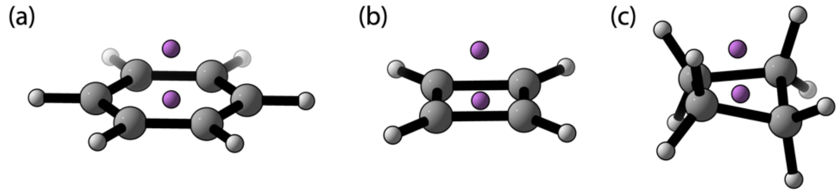
The nucleus-independent chemical shift (NICS) is a widely used index for evaluating aromaticity through theoretical computation. A negative NICS index indicates an aromatic system, while a positive index indicates an antiaromatic system. A NICS index close to 0 signifies nonaromaticity or weak aromaticity/antiaromaticity. Several studies, such as the one published in *Org. Lett.*, **2006**, *8*, 863, have shown that NICS(0)$_{ZZ}$ or NICS(1)$_{ZZ}$ is a better index than the original definition of NICS, now known as NICS(0).

The most common method to obtain NICS values is to add ghost atoms (in *Gaussian*: Bq atom) at the target positions and then submit to NMR computation. The shielding tensors of all atoms are computed and summarized in the output file. A typical output for the shielding tensors of ghost atoms is shown below:

```
   9  Bq   Isotropic =    -0.1133   Anisotropy =     1.6782
XX=     0.0988   YX=     0.5330   ZX=     0.3438
XY=     0.5353   YY=     0.1032   ZY=     0.3474
XZ=     0.7234   YZ=     0.7264   ZZ=    -0.5420
```

The NICS index represents the reversed value of the shielding tensor. In the example aboved, NICS is 0.1133 (indicated by “Isotropic =”) and NICS$_{ZZ}$ is 0.5420 (indicated by “ZZ=”).

For instance, NICS indexes calculated at (R)B3LYP/6-311+G(2d,p) for benzene, cyclobutadiene, and cyclobutane are shown:

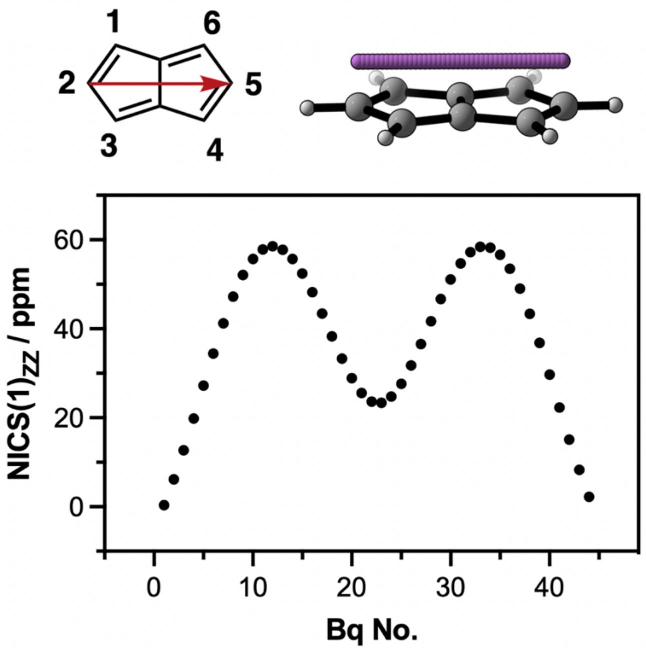


Molecule	NICS(0)	NICS(1)	NICS(0)$_{ZZ}$	NICS(1)$_{ZZ}$
benzene	–7.6064	–9.9993	–14.9722	–29.6808
cyclobutadiene	27.0495	17.7196	110.7864	55.6224
cyclobutane	0.3766	1.2447	56.7631	3.1377

3.4.1.2 NICS Scan

NICS is often studied at specific points, such as the center of a ring. To provide a more detailed understanding of the induced magnetic field of a molecule, a method called NICS Scan has been proposed, which involves using a series of NICS probes. One popular approach is the "NICS-*XY*-Scan" (See: *Chem. Eur. J.*, **2014**, *20*, 5673), where the molecule is positioned on the XY plane, the trajectory of NICS probes is parallel to the XY plane, and the probes are placed at 0.1 Å intervals along a line that spans the length of the system.

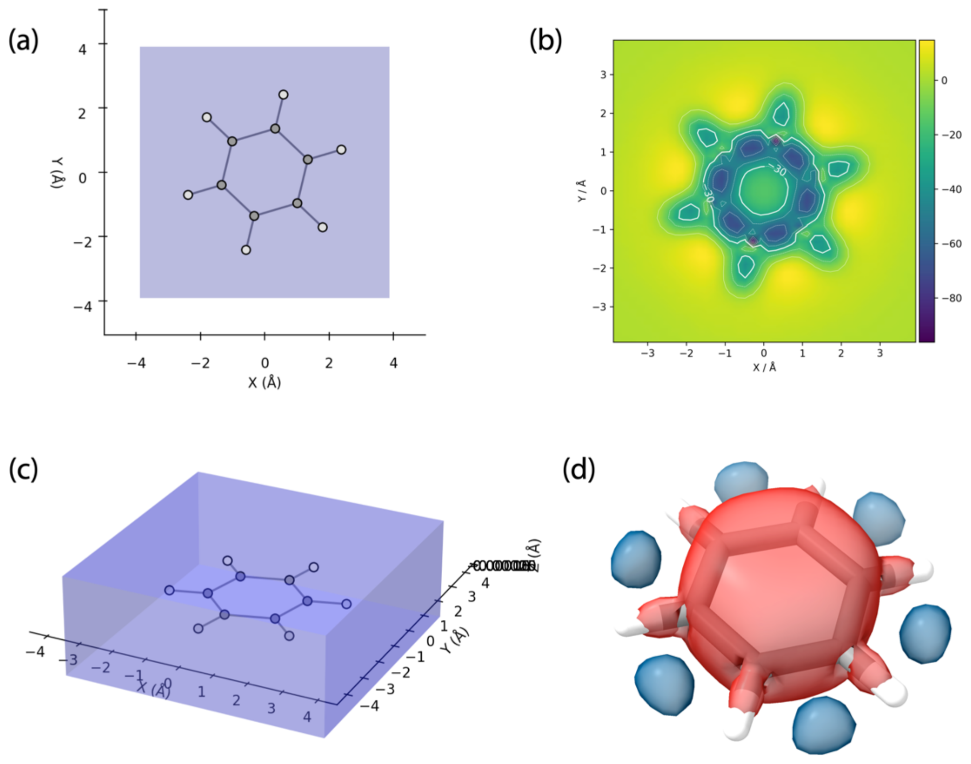
An example of a NICS Scan of pentalene, calculated at the (R)B3LYP/6-311+G(2d,p) level of theory, is illustrated. In this example, the NICS probes are aligned along the C2 to C5 with 0.1 Å intervals and are positioned 1 Å above the pentalene plane.



3.4.1.3 2D and 3D NICS Analysis

Similar to the NICS Scan, the 2D and 3D NICS provide a more intuitive representation of aromaticity. In some cases, the 2D/3D NICS is also referred to as 2D/3D ICSS, which stands for iso-chemical shielding surface. The values used in 2D/3D ICSS are derived from shielding tensors, rather than the reversed values used in NICS analysis. Consequently, both 2D/3D ICSS and 2D/3D NICS can be analyzed from the same output file, and they share the same absolute values but with different signs.

Examples of 2D and 3D NICS analyses of benzene are depicted with the NMR computations performed at the (R)B3LYP/6-311+G(2d,p) level of theory.



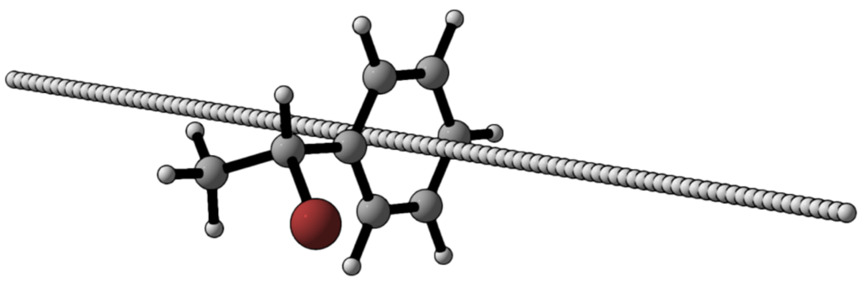
*Illustrates of (a) 2D NICS and (c) 3D NICS, the violet regions indicate the positions of ghost atom. (b) give a 2D NICS(0)$_{ZZ}$ heatmap and the contour line of –30 ppm is emphasized in bold line. (d) show the iso-surface of –25 ppm (red) and 10 ppm (blue) from the data of 3D NICS$_{ZZ}$.

3.4.1.4 Integral NICS

Integral NICS (INICS, ∫NICS) has been suggested to evaluate aromaticity of nonsymmetric system, *i.e.*: NICS(l) ≠ NICS(–l). Since the center of the induced ring current may be a little off the geometrical center. This difference is expected to be negligible at distances ≥ 5 Å. See *J. Phys. Chem. A* , **2019**, *123*, 3922 and *J. Phys. Chem. A* , **2022**, *126*, 3433, and Appendix III The INICS index is defined as follows:

$$\mathrm{INICS}_{\alpha} = \int_{-l}^{l} \mathrm{NICS}_{\alpha}(z)\mathrm{d}z$$

For example, (1-bromoethyl)benzene has different chemical environment on the two sides of phenyl ring, thus, NICS(0)$_{ZZ}$ = –9.40, NICS(1, –CH_3)$_{ZZ}$ = –14.22, NICS(–1, Br)$_{ZZ}$ = –14.52, INICS(–10, 10) = –65.61.



3.4.2 Create Input File for NICS Calculation

☞Supported file types: **Type-I** and **Type-II**

Users can generate *Gaussian* input files for NICS calculations using py.**Aroma 4**. Upon opening the **Type-I** or **Type-II** input file, users can select “NICS”, “NICS Scan”, “2D NICS”, “3D NICS”, or “INICS” based on their specific requirements.

In the input file creation window, users have the option to edit the *Gaussian* calculation keyword via the “Calculation Setup” button. However, when reading a *.gjf* or *.com* file as a py.**Aroma 4** input, the “Calculation Setup” is only applicable when the “Overwrite routine section when reading input files” option is enabled in the program setting window (*Section 2.2*). This option is checked by default.

3.4.2.1 Single Point NICS

Upon entering the single point NICS window, py.**Aroma 4** will automatically detect all chord less monocycles in the geometry, allowing users to add ghost atoms for all monocycles at once. If a sequence of atom indices (requiring more than three atoms) is provided, the ghost atom will be added at the center of the specified atoms. Users also have the option to adjust the height. If a height value greater than 0 is specified, two ghost atoms will be added above and below the plane.

py.**Aroma 4** also allows users to add ghost atom at center of geometry and center of mass.

The *.gjf* files for NICS calculations will be saved with the same file name as the input file, followed by the “_NICS” suffix.

For an example of creating an input file for single point NICS calculation with py.**Aroma 4**, please refer to *Section 4.4.2*.

3.4.2.2 NICS Scan

To create a path for a NICS scan, a sequence of knots (at least two knots) is required. The knots will be added at the center of the specified atoms. Once more than two knots are specified, the path will be displayed on the screen, and the figure can be saved as a *.png* image with the same file name as the input file, followed by the “NICS_Scan” suffix.

py.**Aroma 4** also supports NICS scan on the YZ and XZ planes, which can be selected in the parameter section, including intervals and height above the plane. The input file will be saved with the same name as the *.png* file. An example of creating an input file for a NICS scan calculation with py.**Aroma 4** is provided in *Section 4.5.2*.

3.4.2.3 2D and 3D NICS

Creating the input files for 2D/3D NICS calculation with py.**Aroma 4** is straightforward. Users only need to specify the region for adding ghost atoms, which can be checked by clicking the “Preview” button.

The *.gjf* files for 2D/3D NICS calculation will be saved with the same file name as the read file, followed by the “_2D(3D)_NICS” suffix. Examples of creating input files for 2D NICS scan calculation with py.**Aroma 4** are provided in *Section 4.6.2*, and for 3D NICS, please refer to *Section 4.7*.

3.4.2.4 INICS

INICS requires adding NICS probe from distance *l* below the ring plane to distance *l* above the ring plane. py.**Aroma 4** provide a simple protocol to create input files for INICS calculations via *Gaussian*. Users need to specify the range *l* and interval *j* of NICS probe path. The ghost atoms will be added in the range from –*l* Å to *l* Å, in every *j* Å. Same as NICS module described in *Section 3.4.2.1*, the chord less monocycles could be automatically detected, thus, users can and NICS probe for all monocycles at once. Also, users can specify the cyclic structure by inputting atom indexes.

The NICS probes will be updated automatically in the preview window. Users can easily confirm the position of ghost atoms. The *.gjf* files will be saved with the same file name as the file read, with "_INICS" suffix. An example of creating an input file for a INICS calculation with py.**Aroma 4** is provided in *Section 4.5.2*.

3.4.3 Process Output File of NICS Calculation

☞Supported file types: **Type-III**

Users can process output files of NICS calculations with py.**Aroma 4**. For analyzing output files of NICS scan, 2D NICS, 3D NICS and INICS, the output files must be generated by the input files created by py.**Aroma 4**. For 2D and 3D NICS, by default, the NICS indexes (*i.e.*, reversed values of shielding tensors) will be used. If 2D and 3D ICSS is preferred, please check the "Save ICSS instead of NICS for 2D and 3D processing" option in the program setting panel (*Section 2.2*).

3.4.3.1 Single Point NICS

py.**Aroma 4** will automatically find the shielding tensors from the output file and display the NICS values on the screen. Users can choose the component of the shielding tensors. The results can be saved as a *.txt* file with the same file name as the *Gaussian* output, followed by the "_NICS_output" suffix.

For those rings which are not parallel to the Cartesian planes (*i.e.*, XY, YZ, XZ), py.**Aroma 4** can compute pseudo-NICS$_{ZZ}$, *i.e.*, NICS$_{\perp}$, for those rings, based on the shielding tensors and normal vector of fitted plane. An example can be found in *Section 4.9.2*.

3.4.3.2 NICS Scan

py.**Aroma 4** will automatically find the shielding tensors from the output file, display the NICS values, and plot the NICS values versus indices of ghost atoms on the screen. Users can also choose the component of the shielding tensors, and the results will be updated automatically. The results can be saved as a *.png* and *.xlsx* with the "_NICS_Scan_output" suffix. An example can be found in *Section 4.9.3*.

3.4.3.3 2D NICS

When reading output files for 2D NICS, the heatmap will be automatically displayed on the screen. Users can choose the component of the shielding tensors, and the heatmap will be updated aotumatically. The heatmap can be saved as a *.png* image with the "_2D_NICS_output" or "_2D_ICSS_output" suffix in the file name. The NICS (or ICSS) indexes can also be saved as a *.csv* file for further investigation. An example can be found in *Section 4.9.4*.

3.4.3.4 3D NICS

After entering 3D NICS window, user could choose the component of shielding tensors and save them to a *Gaussian*-type *.cube* (also known as *.cub*) file, with "_3D_NICS_output" or "_3D_ICSS_output" suffix in file name, which could be visualized by *GaussView*, *VMD*, *ChimeraX*, *etc*.

A preview of the heatmap for a 2D slice will be displayed on the screen. Users can edit it in the same way as 2D NICS module. An example can be found in *Section 4.9.5*.

3.4.3.5 INICS

After entering INICS window, the geometry and a plot of NICS index versus $\mathrm{NICS}_{\mathrm{ZZ}}$ will be displayed. If there is more than one monocycle in the geometry, the ring No. 1 will be displayed in the plot. Users can choose the NICS component and ring No. for the research interest. The INICS index will be summarized in the right part of the window.

Users can save the NICS plot as *.png* file by "Save *.png*" button, and save the NICS values in a *.xlsx* file by "Save *.xlsx*" button. An example can be found in *Section 4.9.6*.

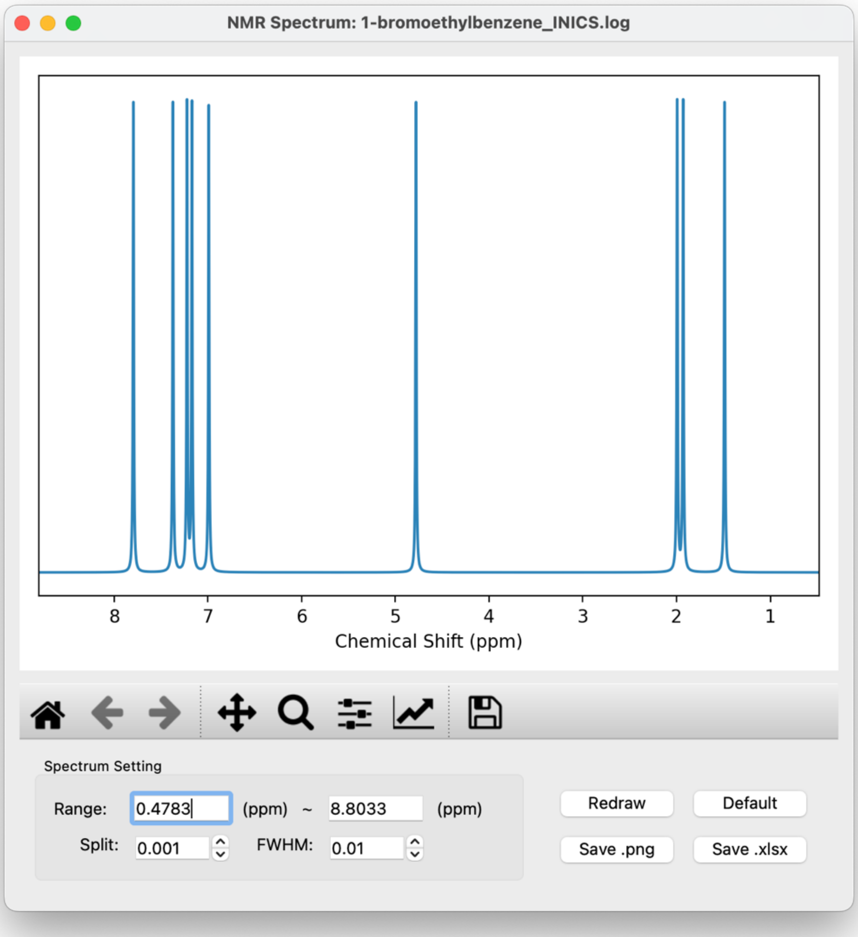
3.5 NMR Analysis

☞Supported file types: **Type-II** (NMR output only), **Type-III**

The Python tool for applying scaling factor to calculated NMR chemical shift, **py.NMR**, has been combined into py.**Aroma 4**, with further improved functions. When a NMR output file is read by py.**Aroma 4**, this function will be available.

After entering the NMR window, user can choose the element for plotting NMR spectrum. Three different modes are available: (1) Use isotropic shielding tensors directly from output file; (2) Set reference and compute chemical shift refer to it; (3) Apply scaling factor[5].

An example for using **py.NMR** module in py.**Aroma 4** could be found in *Section 4.10*.



Range: 0.4783 (ppm) ~ 8.8033 (ppm)
Split: 0.001 FWHM: 0.01

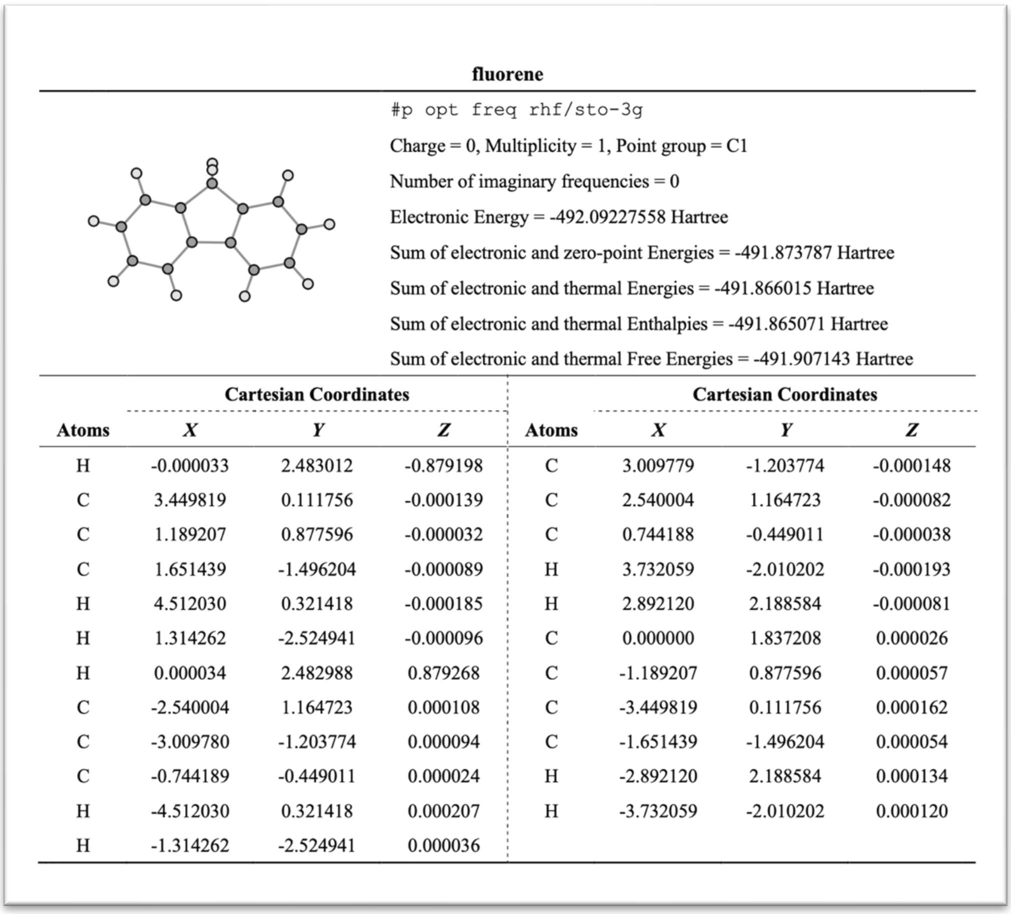
Note: **py.NMR** is no longer being maintained by the developer since the basic function has been combined in py.**Aroma** from version 4.0.0, but it can still be downloaded from the archived GitHub page (https://github.com/wongzit/pyNMR).

[5] http://cheshirenmr.info

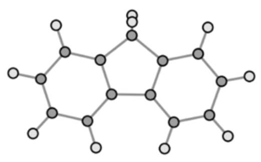
3.6 Generate Computational Supporting Material

☞Supported file types: **Type-II**

As an additional function, **CSIgen**, which is an open-source Python script for generating computational supporting information/materials, is integrated into py.**Aroma 4**. When a **Type-II** file is read by py.**Aroma 4**, this function becomes available.



fluorene



```
#p opt freq rhf/sto-3g
```

Charge = 0, Multiplicity = 1, Point group = C1

Number of imaginary frequencies = 0

Electronic Energy = -492.09227558 Hartree

Sum of electronic and zero-point Energies = -491.873787 Hartree

Sum of electronic and thermal Energies = -491.866015 Hartree

Sum of electronic and thermal Enthalpies = -491.865071 Hartree

Sum of electronic and thermal Free Energies = -491.907143 Hartree

	Cartesian Coordinates				**Cartesian Coordinates**		
Atoms	***X***	***Y***	***Z***	**Atoms**	***X***	***Y***	***Z***
H	-0.000033	2.483012	-0.879198	C	3.009779	-1.203774	-0.000148
C	3.449819	0.111756	-0.000139	C	2.540004	1.164723	-0.000082
C	1.189207	0.877596	-0.000032	C	0.744188	-0.449011	-0.000038
C	1.651439	-1.496204	-0.000089	H	3.732059	-2.010202	-0.000193
H	4.512030	0.321418	-0.000185	H	2.892120	2.188584	-0.000081
H	1.314262	-2.524941	-0.000096	C	0.000000	1.837208	0.000026
H	0.000034	2.482988	0.879268	C	-1.189207	0.877596	0.000057
C	-2.540004	1.164723	0.000108	C	-3.449819	0.111756	0.000162
C	-3.009780	-1.203774	0.000094	C	-1.651439	-1.496204	0.000054
C	-0.744189	-0.449011	0.000024	H	-2.892120	2.188584	0.000134
H	-4.512030	0.321418	0.000207	H	-3.732059	-2.010202	0.000120
H	-1.314262	-2.524941	0.000036				

Upon entering the window, the user can select which information to be included in the supporting information and save it as a *.txt* or *.xlsx* file. An example of using the **CSIgen** module in py.**Aroma 4** can be found in *Section 4.11*.

Note: **CSIgen** is no longer being maintained by the developer since the basic function has been combined in py.**Aroma** from version 3.0.0, but it can still be downloaded from the archived GitHub page (https://github.com/wongzit/CSIgen).

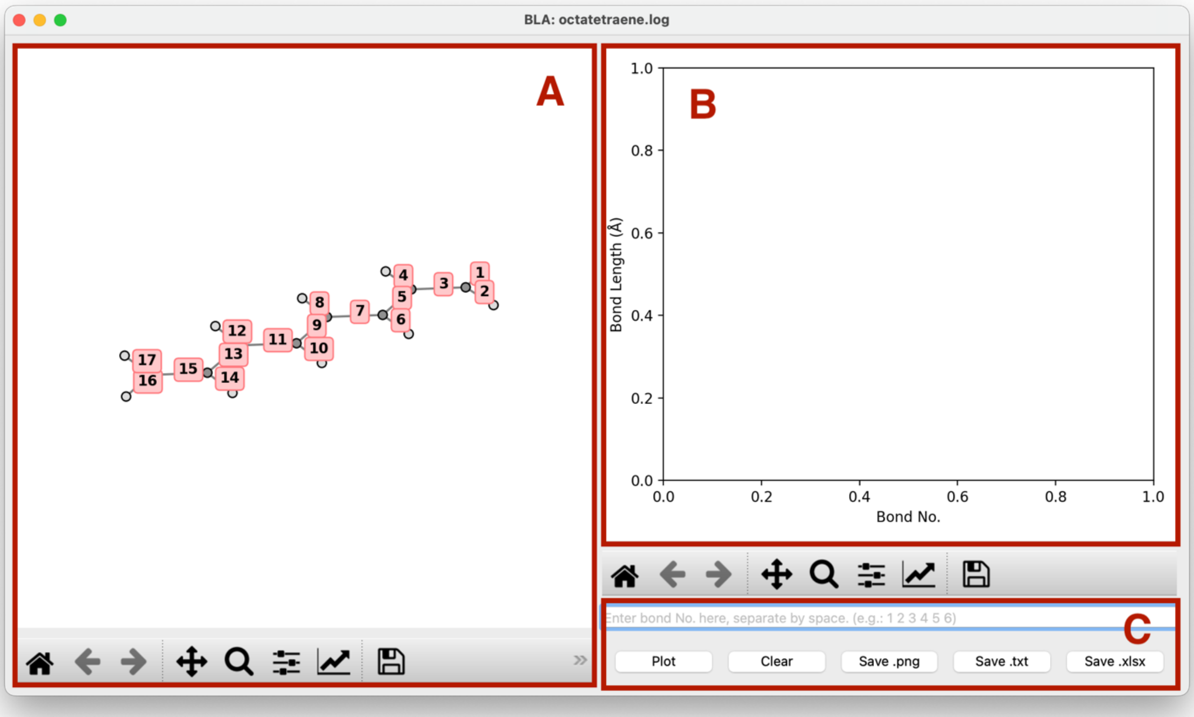
4 Tutorials and Examples

This chapter includes detail tutorials and some examples for using py.**Aroma 4** to carry out aromaticity analyses.

4.1 BLA Analysis

4.1.1 Basic Information

The BLA function can be accessed by clicking the BLA button after opening a file. The main window of the BLA function includes 3 areas. The molecular geometry from the input file is displayed in **area A**, with bond indexes at the center of each bond. In **area B**, a preview of the bond length versus bond index graph is shown. However, there is no preview unless the user presses the "Plot" button in **area C**. **Area C** contains a group of functions, including a text bar for entering bond indexes, a "Plot" button for previewing the graph and BLA value in **area B**, a "Clear" button to clear the current graph in **area B**, a "Save *.png*" button to save the current graph in **area B**, and "Save *.txt*" / "Save *.xlsx*" buttons to save bond data to *.txt* / *.xlsx* files.



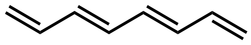
In order to plot the BLA graph, the user needs to input a sequence of bond indexes separated by spaces in the text bar in **area C**, for example: 3 5 7 9 11 13 15. It is important NOT to use other separators such as ",". Some examples of bad inputs:

- 3,5,7,9,11,13,15 (*illegal separator*)
- 3.5.7.9.11.13.15 (*illegal separator*)

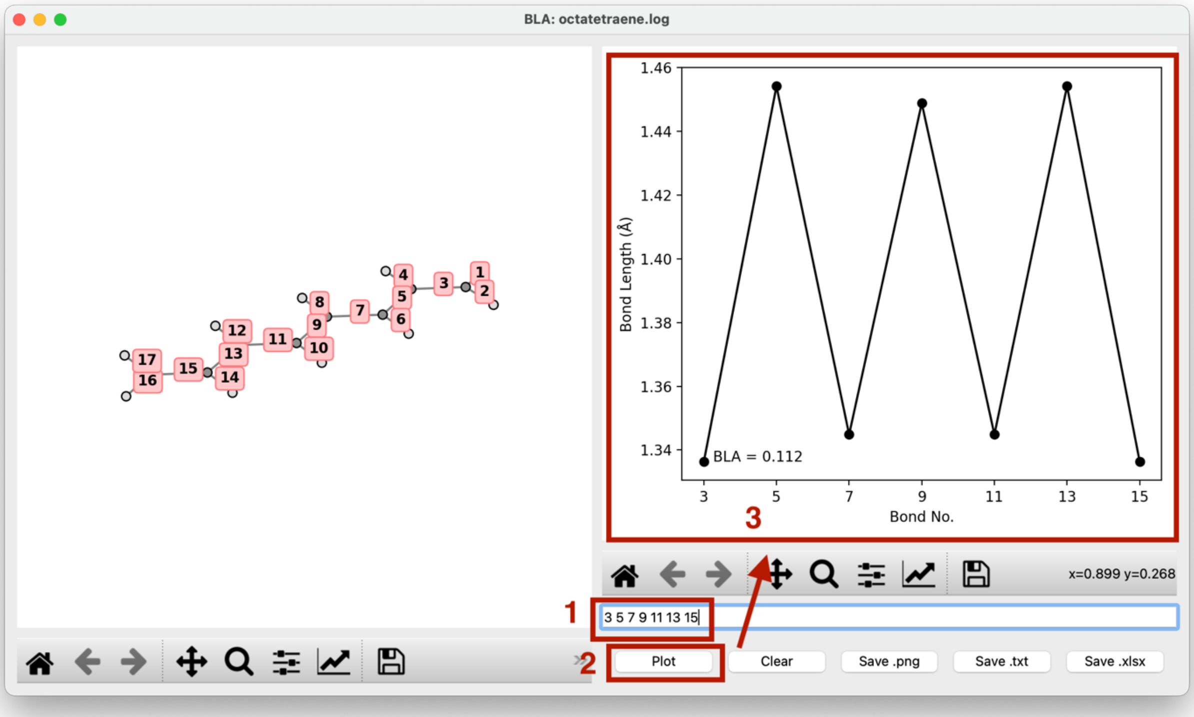
- 3 5 7 9 11 13 22 (*bad bond index, no bond No. 22 in molecule*)
- 3 5 7 9 11 13 1t (*only input number*)

4.1.2 BLA Analysis of 1,3,5,7-Octatetraene

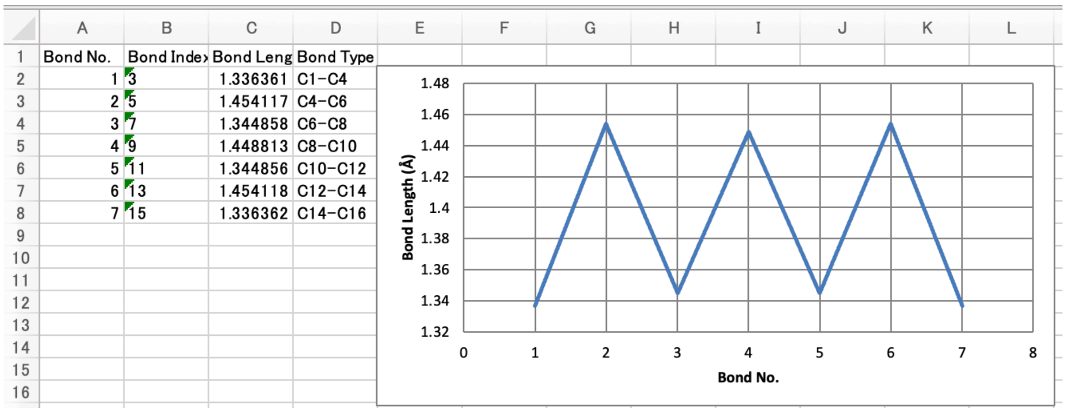
Structure optimization of 1,3,5,7-octatetraene was performed at (R)ωB97X-D/6-31G(d) level of theory. The output file "*octatetraenene.log*" was read by py.**Aroma 4**.



Input sequence of bond indexes "3 5 7 9 11 13 15" into the text bar in **area C**. By clicking the "Plot" button, a preview of the BLA graph is displayed in **area B**.



Next, if users want to save the graph and/or data, simply click on "Save *.png*", "Save *.txt*", and "Save *.xlsx*" as needed. The files will be saved as "*octatetraene_**BLA**.png*", "*octatetraene_**BLA**.txt*" and "*octatetraene_**BLA**.xlsx*". For example, a screenshot of the .xlsx file generated by py.**Aroma 4** is shown follows.

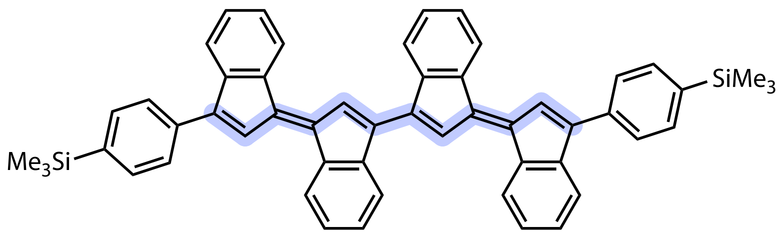


	A	B	C	D
1	Bond No.	Bond Inde	Bond Leng	Bond Type
2	1	3	1.336361	C1−C4
3	2	5	1.454117	C4−C6
4	3	7	1.344858	C6−C8
5	4	9	1.448813	C8−C10
6	5	11	1.344856	C10−C12
7	6	13	1.454118	C12−C14
8	7	15	1.336362	C14−C16

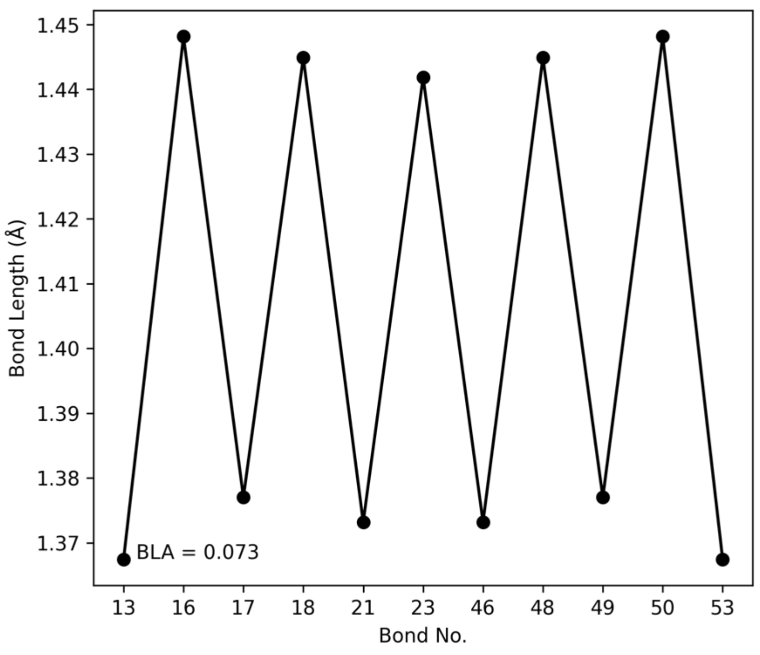
***Hint*:** Once the bond sequence is inputted, it is possible to directly click on "Save *.png*", "Save *.txt*", and "Save *.xlsx*" without clicking the "Plot" button, as the preview module of py.**Aroma 4** will be active before saving the *.png*, *.txt*, and *.xlsx* files.

4.1.3 BLA Analysis of Oligo(biindenylidene)

Follows example is from *Nat. Commun.*, **2023**, *14*, 2741. The structural coordinates were extracted from supporting information and saved as "natcom_4b.*xyz*" and then read by py.**Aroma 4**.



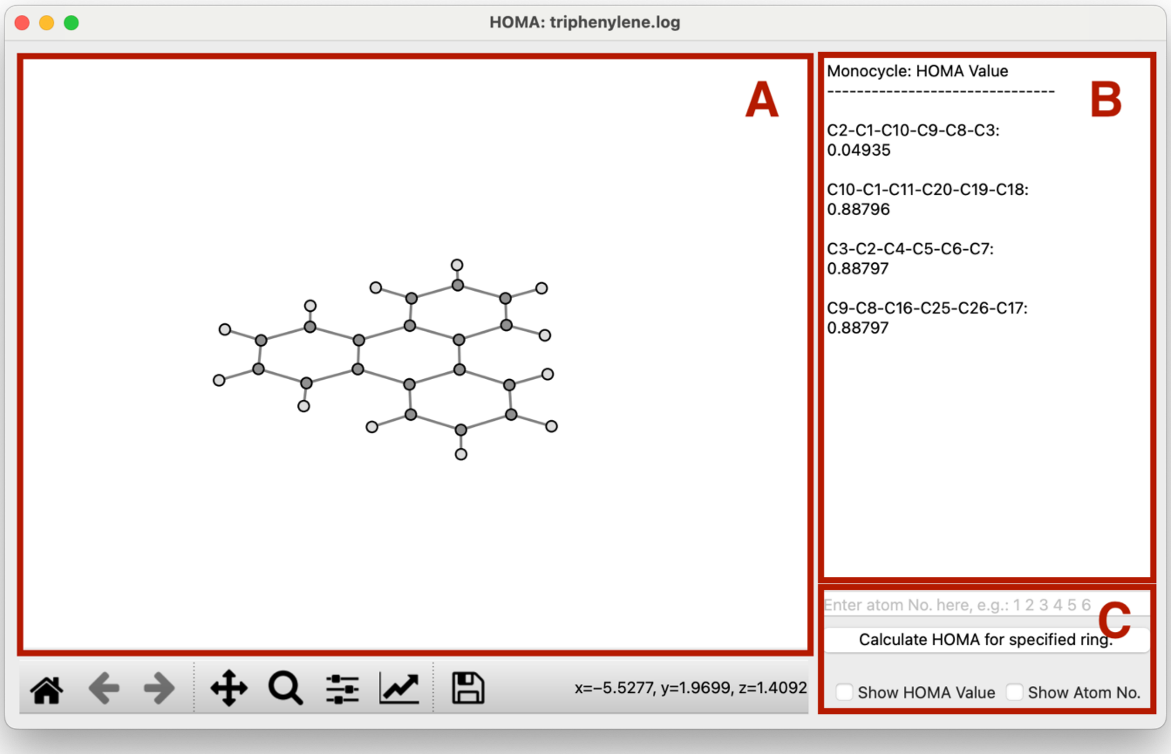
The bond indexes "13 16 17 18 21 23 46 48 49 50 53" were inputted and the BLA graph was plotted. As shown in the follows, the Figure 4b in the original paper is well reproduced by py.**Aroma 4**.



4.2 HOMA Analysis

4.2.1 Basic Information

The HOMA function can be accessed by clicking the HOMA button after opening a file. **Area A** displays a 3D geometry of the inputted molecule. The HOMA values for all monocycles are automatically computed and displayed in **area B**. In **area C**, there is a text bar asking for a sequence of atom indices if the user has cyclic structures for which the HOMA values are needed. After inputting the atom indices, the user can obtain the HOMA value by clicking the "Calculate HOMA for specified ring" button in **area C**. The check boxes in **area C**: "Show HOMA Value" will display the HOMA values at the center of each cycle in **area A**; "Show Atom No." will display atom indices on each atom in **area A**, which is useful when specifying rings in the text bar.


HOMA: triphenylene.log
A
B
C
Enter atom No. here, e.g.: 1 2 3 4 5 6
Calculate HOMA for specified ring.
Show HOMA Value
Show Atom No.
x=−5.5277, y=1.9699, z=1.4092

```
Monocycle: HOMA Value
------------------------------
C2-C1-C10-C9-C8-C3:
0.04935

C10-C1-C11-C20-C19-C18:
0.88796

C3-C2-C4-C5-C6-C7:
0.88797

C9-C8-C16-C25-C26-C17:
0.88797
```

The R_{opt} and α parameters used for HOMA calculations are defined in program setting panel. User can modify the parameters if necessary. More information about program setting, please refer to *Section 2.2*.

In **area B**, the atom indexes of cyclic structure and its HOMA values would be displayed. For example:

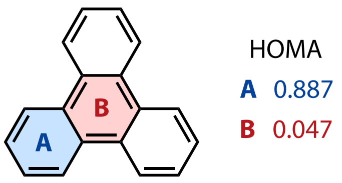
```
C10-C18-C19-C20-C11-C1:
0.88796
```

means the HOMA value of ring "C10-C18-C19-C20-C11-C1" is 0.88796.

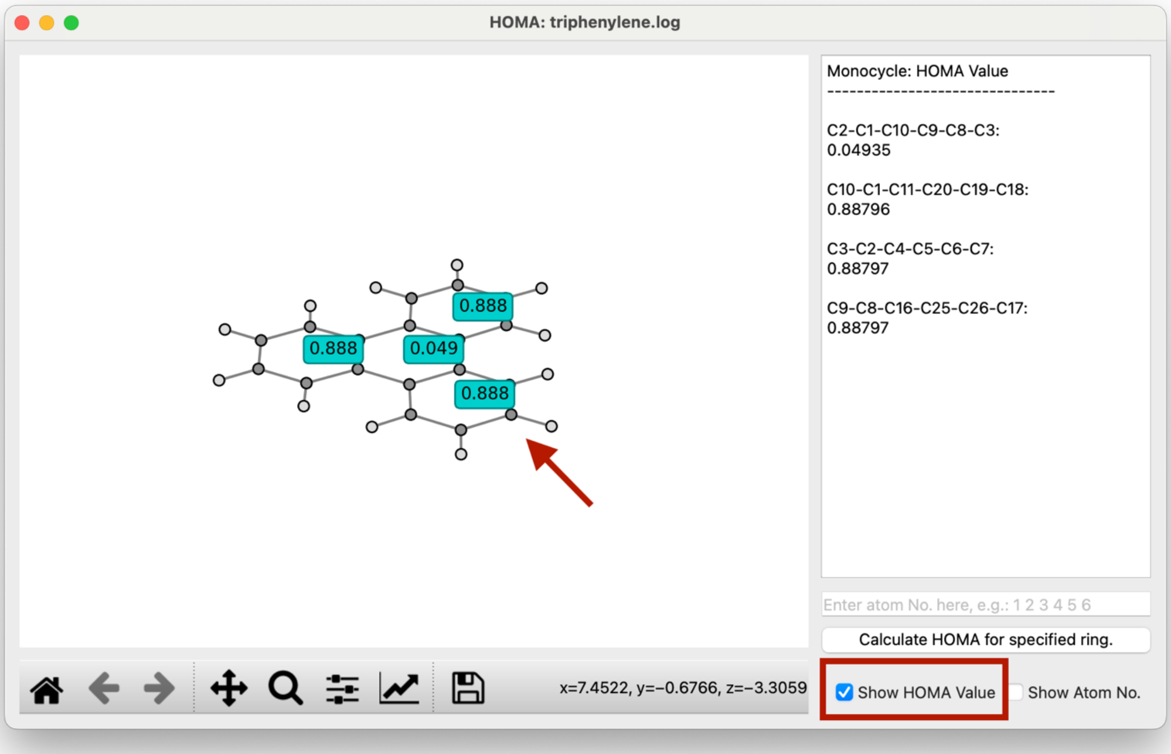
Same as the BLA function, the atom indexes in the text bar of **area C** should also be separated by spaces, *e.g.*: 10 18 19 20 11 1, and the order should be the same as bond connectivity. For example, "*10 19 18 20* 11 1" is not valid since there is no connection between 10 and 19, 18 and 20. The sequence MUST include more than three atom indexes to construct a cyclic structure.

4.2.2 HOMA Analysis of Triphenylene

Structure optimization of triphenylene was performed at (R)B3LYP/6-31G(d) level of theory. The optimized triphenylene adapted D_{3h} point group and the HOMA values of ring A and ring B have been reported to be 0.887 and 0.047, respectively, from *Org. Lett.*, **2014**, *16*, 6116.

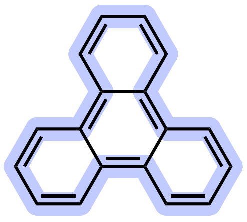


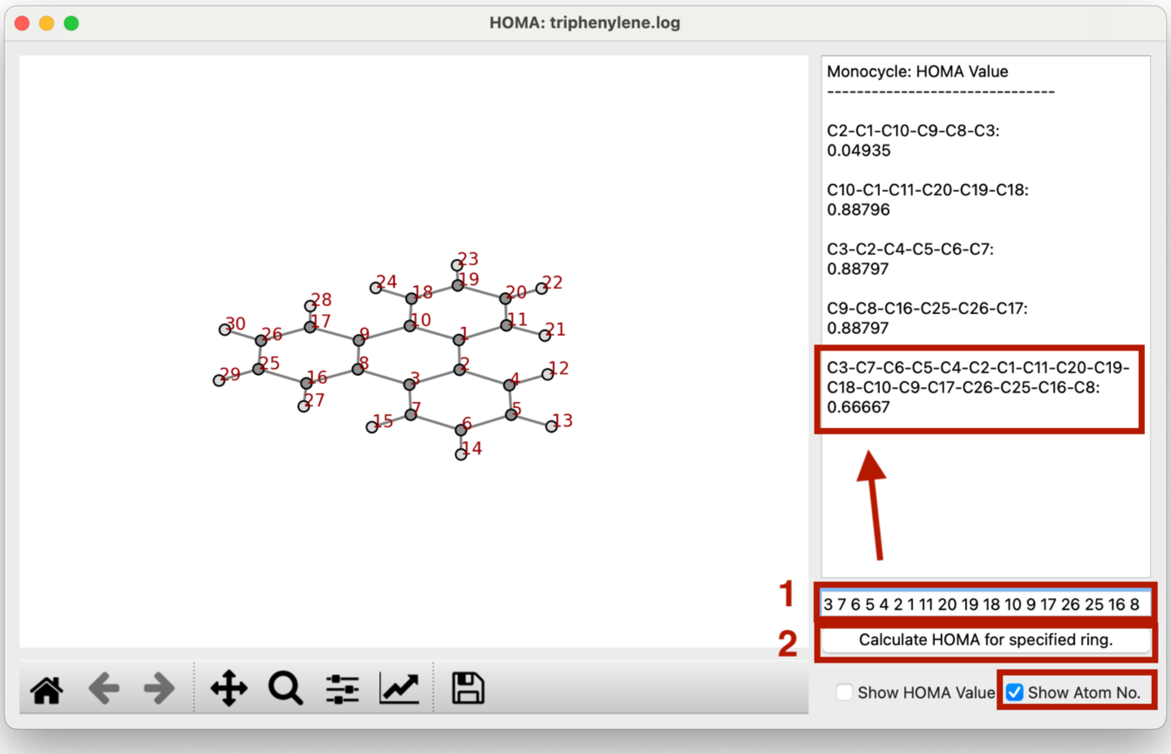
The optimized geometry was opened by py.**Aroma 4** for computing HOMA values. The HOMA values for all monocycles automatically displayed in **area B**, and the results are well reproduced. By checking the "Show HOMA Value" option, the HOMA values are displayed on the center of each monocycle.


HOMA: triphenylene.log
0.888
0.888
0.049
0.888
x=7.4522, y=−0.6766, z=−3.3059
Enter atom No. here, e.g.: 1 2 3 4 5 6
Calculate HOMA for specified ring.
Show HOMA Value
Show Atom No.

```
Monocycle: HOMA Value
--------------------------------

C2-C1-C10-C9-C8-C3:
0.04935

C10-C1-C11-C20-C19-C18:
0.88796

C3-C2-C4-C5-C6-C7:
0.88797

C9-C8-C16-C25-C26-C17:
0.88797
```

In this case, to investigate the HOMA value of the outer edge, a sequence of atom indexes (3 7 6 5 4 2 1 11 20 19 18 10 9 17 26 25 16 8) is entered into the text bar, followed by clicking the "Calculate HOMA for specified ring" button. Subsequently, the HOMA value will be displayed in **area B**.





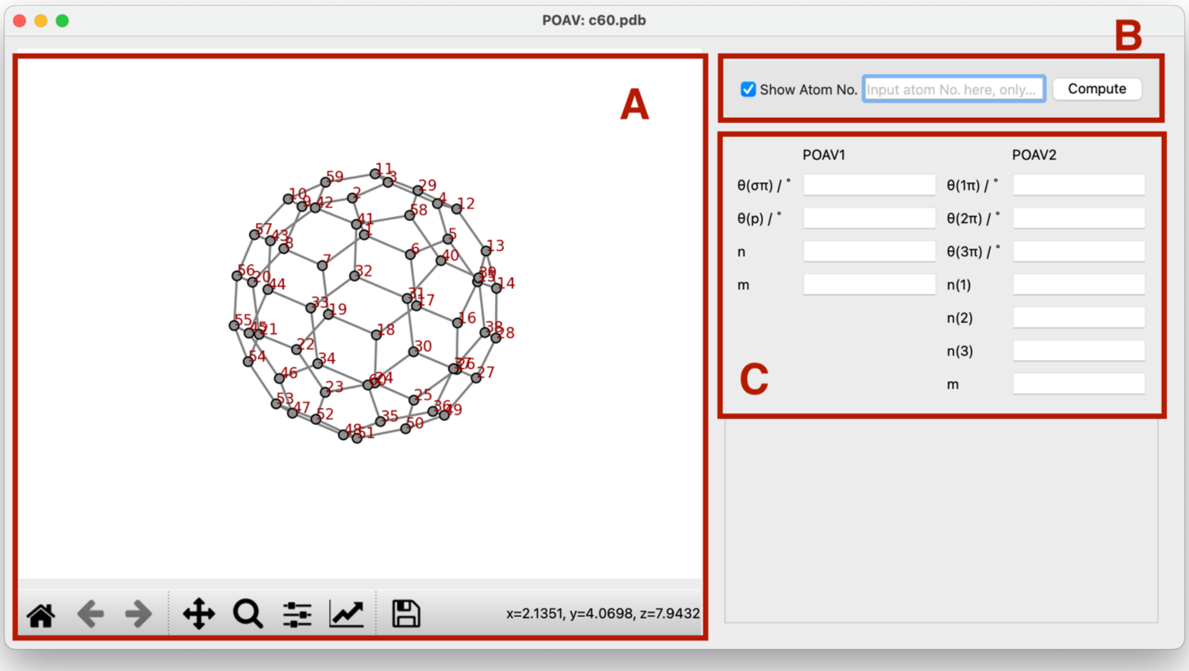
```
Monocycle: HOMA Value
--------------------------------

C2-C1-C10-C9-C8-C3:
0.04935

C10-C1-C11-C20-C19-C18:
0.88796

C3-C2-C4-C5-C6-C7:
0.88797

C9-C8-C16-C25-C26-C17:
0.88797

C3-C7-C6-C5-C4-C2-C1-C11-C20-C19-
C18-C10-C9-C17-C26-C25-C16-C8:
0.66667
```

```
3 7 6 5 4 2 1 11 20 19 18 10 9 17 26 25 16 8
```

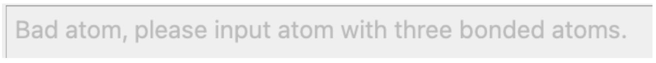
4.3 POAV Analysis

4.3.1 Basic Information

Users can access the POAV function by clicking the POAV button after opening a file. **Area A** displays the input geometry with atom indexes. **Area B** contains basic functions, including a check box labeled "Show Atom No." that allows users to show or hide atom indexes in **Area A**. There is also a text bar near the check box where you can enter the target atom index for the POAV analysis. Once user inputs the atom index, click the "Compute" button, the POAV1 and POAV2 results will then be displayed in **Area C**, and the target atom will be highlighted in **Area A**.

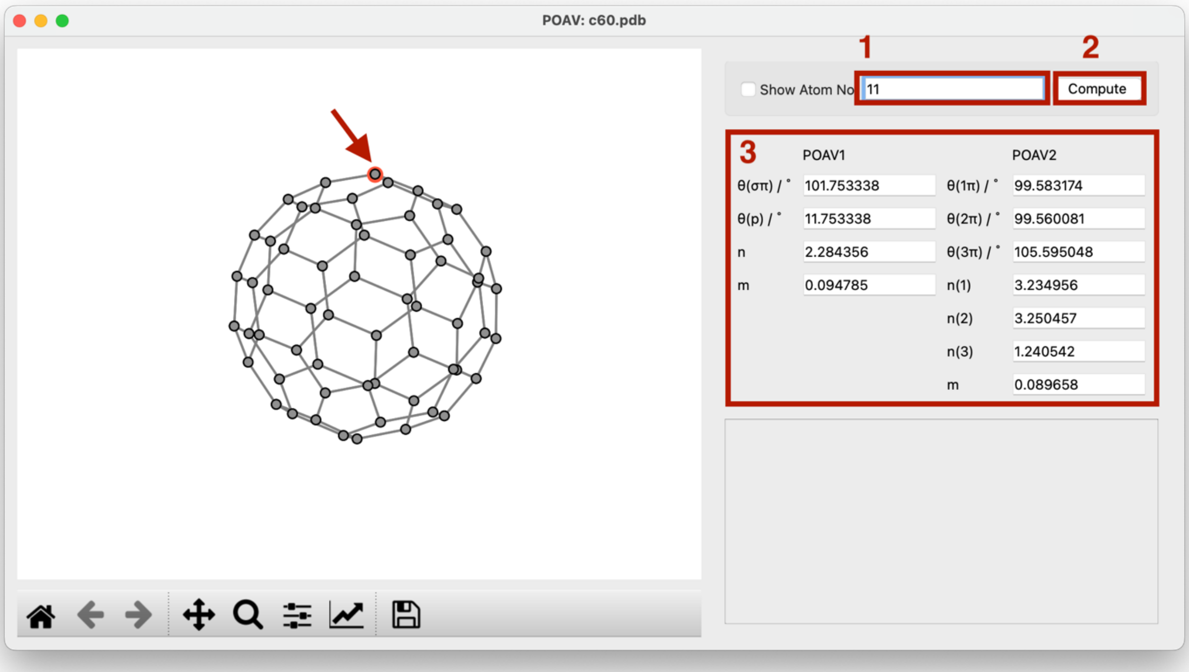


The text bar in **area B** could only accept one atom index, and the target atom MUST has and ONLY has three bonded atoms. Otherwise, and error message would be displayed.


Bad atom, please input atom with three bonded atoms.

4.3.2 POAV Analysis of Fullerene C_{60}

Fullerene C_{60} is a hot molecule used as example of POAV analysis. As example, we want to know the POAV parameters of atom C11, so, we just need to input "11" in the text bar and click "Compute". The atom C11 is highlighted and the POAV parameters are displayed.



11

	POAV1		POAV2
θ(σπ) / °	101.753338	θ(1π) / °	99.583174
θ(p) / °	11.753338	θ(2π) / °	99.560081
n	2.284356	θ(3π) / °	105.595048
m	0.094785	n(1)	3.234956
		n(2)	3.250457
		n(3)	1.240542
		m	0.089658

4.4 Create Input File for NICS 📍

4.4.1 Basic Information

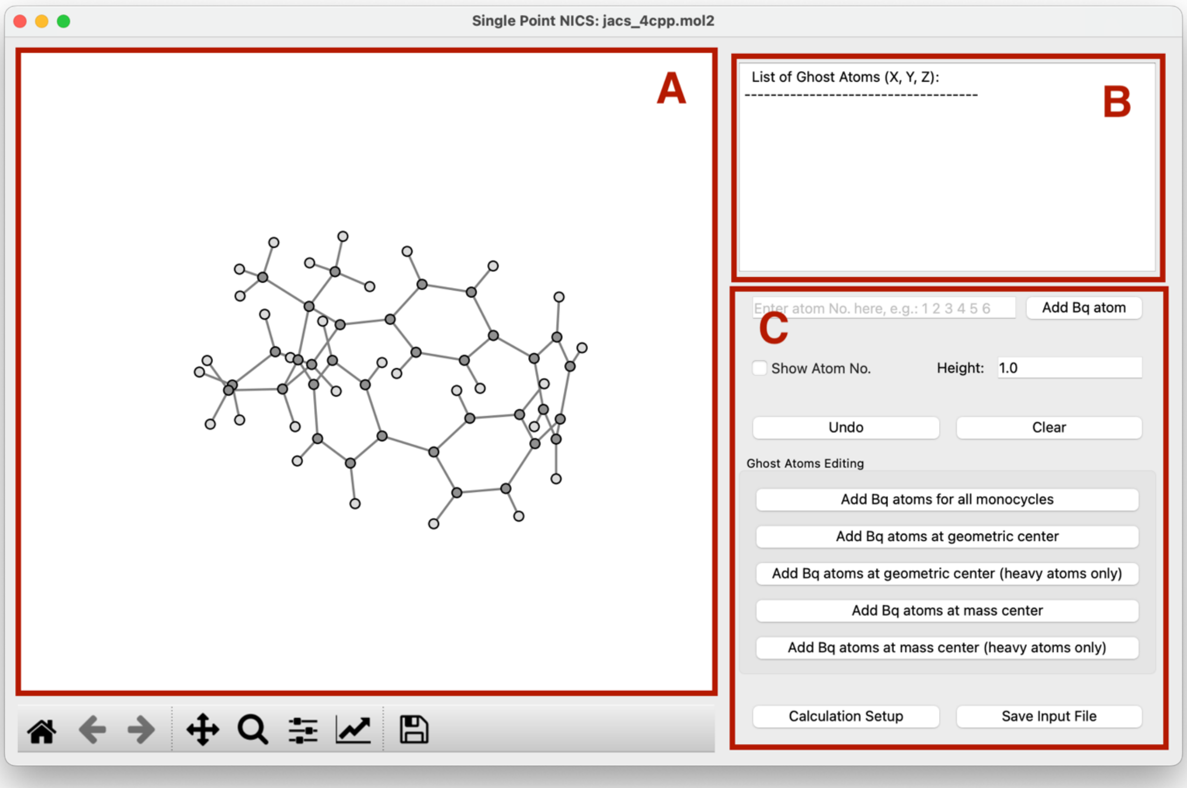
The single point NICS function can be accessed by clicking the NICS button after opening a file. The molecular geometry is displayed in **area A**. The monocycles in the molecule were automatically detected once after loading the geometry. By clicking the "Add Bq atoms for all monocycles" button in **area C**, py.**Aroma 4** will add ghost atoms at a specified height for all monocycles (1 Å by default, but the height can be modified in the "Height:" text bar in **area C**). Furthermore, users can customize the position of the ghost atom by follows these steps:

(0) Check "Show Atom No.", although this is not compulsory, it is convenient.

(1) Input the atom indexes in the text bar in **area C** (*e.g.*: 1 2 3 4 5 6).

(2) Set the height in the "Height:" text bar.

(3) Click the "Add Bq atom" button. The ghost atom(s) will then be added in the center of atoms 1, 2, 3, 4, 5 and 6, at the specified height in Å.


Single Point NICS: jacs_4cpp.mol2
A
B
C
List of Ghost Atoms (X, Y, Z):
Enter atom No. here, e.g.: 1 2 3 4 5 6
Add Bq atom
Show Atom No.
Height: 1.0
Undo
Clear
Ghost Atoms Editing
Add Bq atoms for all monocycles
Add Bq atoms at geometric center
Add Bq atoms at geometric center (heavy atoms only)
Add Bq atoms at mass center
Add Bq atoms at mass center (heavy atoms only)
Calculation Setup
Save Input File

py.**Aroma 4** will ignore monocycles larger than 10-membered rings. If your compound includes large cycles and you want to compute NICS for them, please add ghost atoms by specifying the atom indexes in **area C**.

Users can delete ghost atoms using the "Undo" and "Clear" buttons in **area C**. "Undo" will delete ghost atoms one by one, starting with the newest added atom. "Clear" will delete all ghost atoms that have been added. The Cartesian coordinates of the ghost atoms will be displayed in **area B**.

To specify the atom indexes in **area C**, at least three atoms are needed to define a plane. The order is insensitive, for example, both “1 2 3 4 5 6” and “1 5 3 6 4 2" are acceptable.

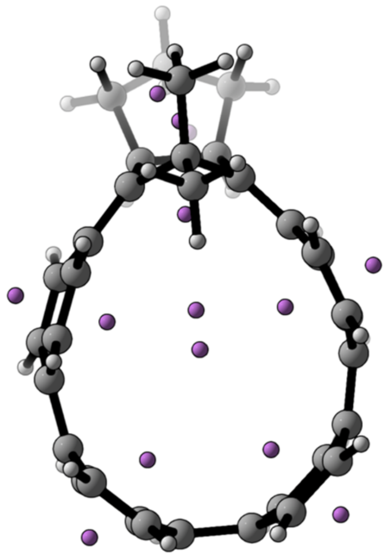
If height is not 0 (*i.e.*: NICS(*l*), *l* > 0), two ghost atoms would be added above and below the cycle.

Users can also add ghost atom at center of geometry and/or center of mass, with contribution from all atoms or only heavy atoms. These functions could be accessed from the button groups in **area C**.

To save *Gaussian* input file, user need to click the “Save Input File” button. Before saving, user can modify the calculation setup, more information about calculation setup, please refer to *Section 2.3*.

4.4.2 Example: DR-[4]CPP

To give a better understanding of how to use NICS module in py.**Aroma 4**, a step-by-step procedure using **DR-4CPP** (*J. Am. Chem. Soc.*, **2021**, *143*, 7426.) would be described. The target is: (*a*) NICS(1) for all monocyclic rings; (*b*) NICS(0) at the center of macrocycle; (*c*) NICS at center of mass (heavy atom only).



< Procedure >

(0) Open *jacs_4cpp.mol2* in py.**Aroma 4**.

(1) Click “Add Bq atoms for all monocycles”, ghost atoms for NICS(1) will be added. Target (*a*) done.

(2) Check “Show Atom No.”

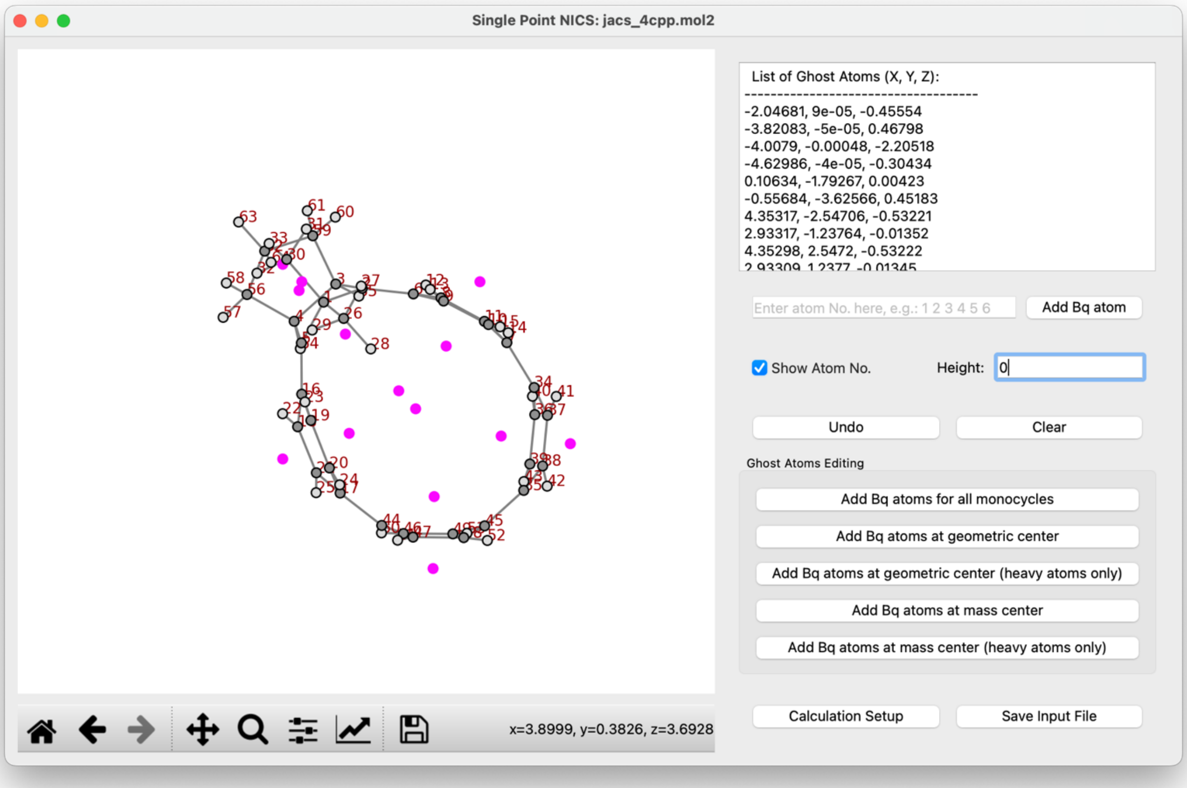
(3) Change the “Height” value to 0.

(4) Input the atom sequence in the text bar: 2 5 6 7 34 35 44 45 16 17. In this example, the equatorial atoms are used to define the plane.

(5) Click “Add Bq Atom”, the NICS(0) of the macrocycle will be added. Target (*b*) done.

(6) Click “Add Bq atoms at mass center (heavy atom only)”. Target (*c*) done.

(7) Click “Save Input File”, finished!

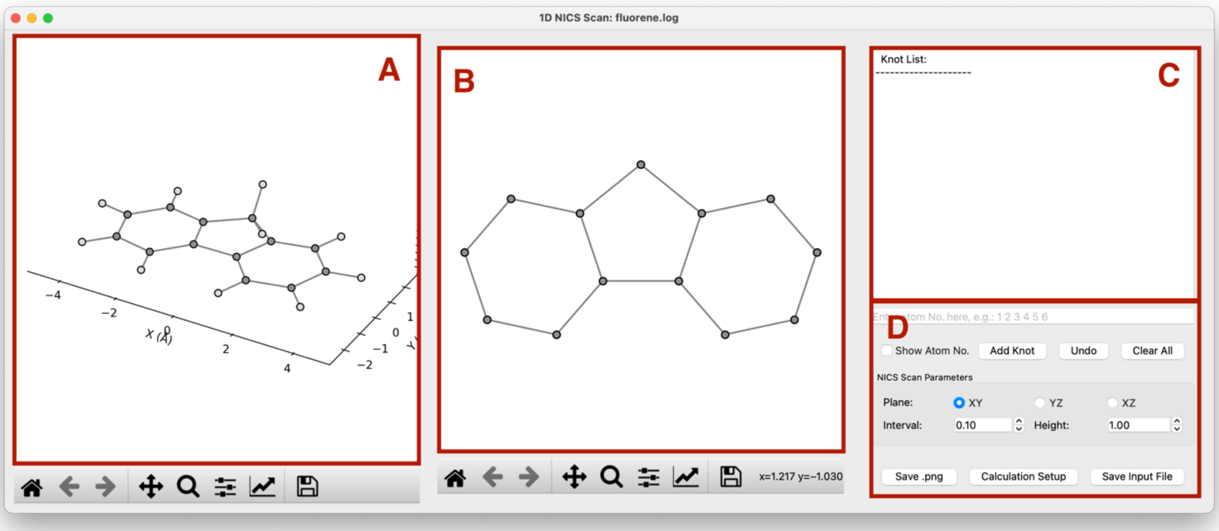

Single Point NICS: jacs_4cpp.mol2
Enter atom No. here, e.g.: 1 2 3 4 5 6
Add Bq atom
Show Atom No.
Height: 0
Undo
Clear
Ghost Atoms Editing
Add Bq atoms for all monocycles
Add Bq atoms at geometric center
Add Bq atoms at geometric center (heavy atoms only)
Add Bq atoms at mass center
Add Bq atoms at mass center (heavy atoms only)
Calculation Setup
Save Input File
x=3.8999, y=0.3826, z=3.6928

```
List of Ghost Atoms (X, Y, Z):
----------------------------------
-2.04681, 9e-05, -0.45554
-3.82083, -5e-05, 0.46798
-4.0079, -0.00048, -2.20518
-4.62986, -4e-05, -0.30434
0.10634, -1.79267, 0.00423
-0.55684, -3.62566, 0.45183
4.35317, -2.54706, -0.53221
2.93317, -1.23764, -0.01352
4.35298, 2.5472, -0.53222
2.93309, 1.2377, -0.01345
```

4.5 Create Input File for NICS Scan

4.5.1 Basic Information

The NICS scan function can be accessed by clicking the NICS scan button after opening a file. Upon entering the NICS scan function, the 3D geometry of the molecule will be displayed in **area A**, and its projection on the XY plane will be automatically displayed in **area B**. Users can change the plane of projection by editing the "Plane" radio button (XY, YZ, XZ) in the "NICS Scan Parameters" section in **area D**.



Plane: [✓ XY] YZ XZ
Interval: 0.10
Height: 1.00

A path is necessary for the 1D NICS scan, and users can add knots by follows this procedure to create a path:

(0) Check "Show Atom No.", which is not compulsory but convenient.

(1) Input the atom indexes in the text bar in **area D** (*e.g.*: 1 2 3 4 5 6).

(2) Click "Add Knot", and the knot will be added at the center of the inputted atoms. The knot information will be updated in **area C**. (***Note***: If users inputted one atom index, the atom will be the knot.)

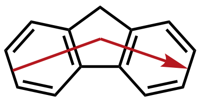
(3) Repeat steps (1) and (2) until the full path is finished.

Users can delete all knots by clicking "Clear All" and delete the last knot by using the "Undo" button. To save a *Gaussian .gjf* file, users can simply click the "Save Input File" button. py.**Aroma 4** will add ghost atoms at a specified height (1 Å by default, but the height can be modified in the "Height:" text bar in **area D**) along the path of knots. The intervals of neighboring ghost atoms are 0.1 Å by default, but users can modify it using the "Interval:" option in **area D**.

The figure in **area B** can be saved as a *.png* file by clicking the "Save *.png*" button in **area D**.

4.5.2 Example: Fluorene

Follows is a step-by-step tutorial for how to use NICS scan module in py.**Aroma 4**, with fluorene as example. The fluorene molecule is placed on XY plane, we aim to put the NICS probes on the trajectory as shown below, on the height (Z axis) of 1 Å, with 0.2 Å interval of ghost atoms.



< Procedure >

(0) Open *fluorene.log* in py.**Aroma 4**.

(1) Check “Show Atom No.”.

(2) Input “16 17” in the text bar in **area D** and click “Add Knot”.

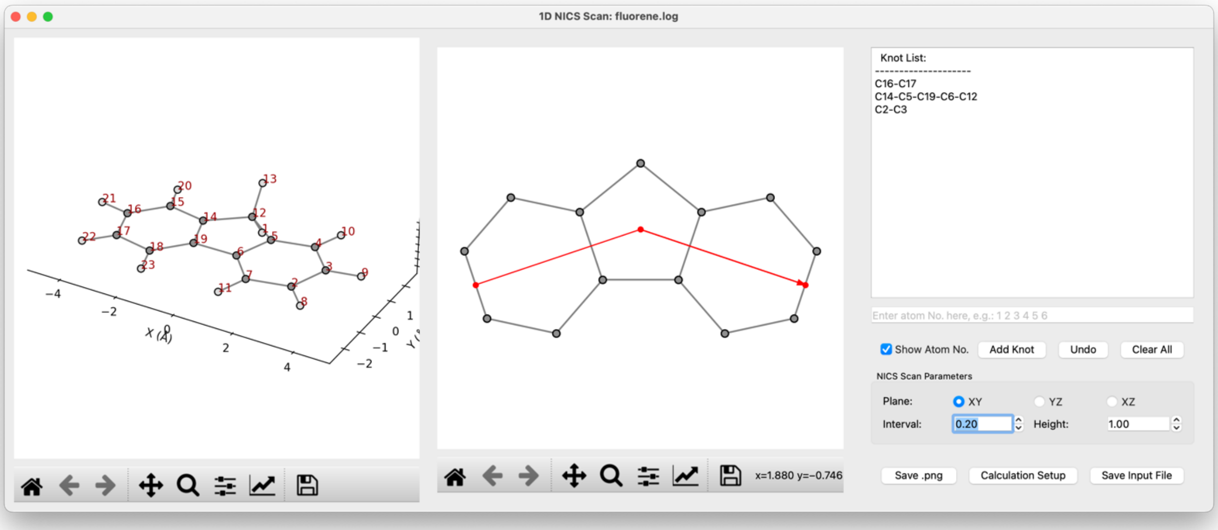
(3) Input “14 5 19 6 12” in the text bar in **area D** and click “Add Knot”.

(4) Input “2 3” in the text bar in **area D** and click “Add Knot”.

(5) Change the value of “Interval” to 0.2.

(6) Click “Save Input File”.

(7) (option) Click “Save .png”. Finished!


1D NICS Scan: fluorene.log

```
Knot List:
--------------------
C16-C17
C14-C5-C19-C6-C12
C2-C3
```

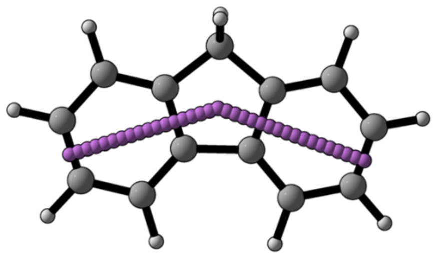
[✓ Show Atom No.]

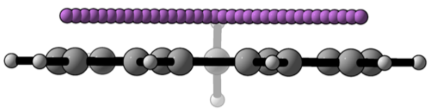
NICS Scan Parameters

Plane: [✓ XY] YZ XZ

Interval: 0.20 Height: 1.00

x=1.880 y=−0.746

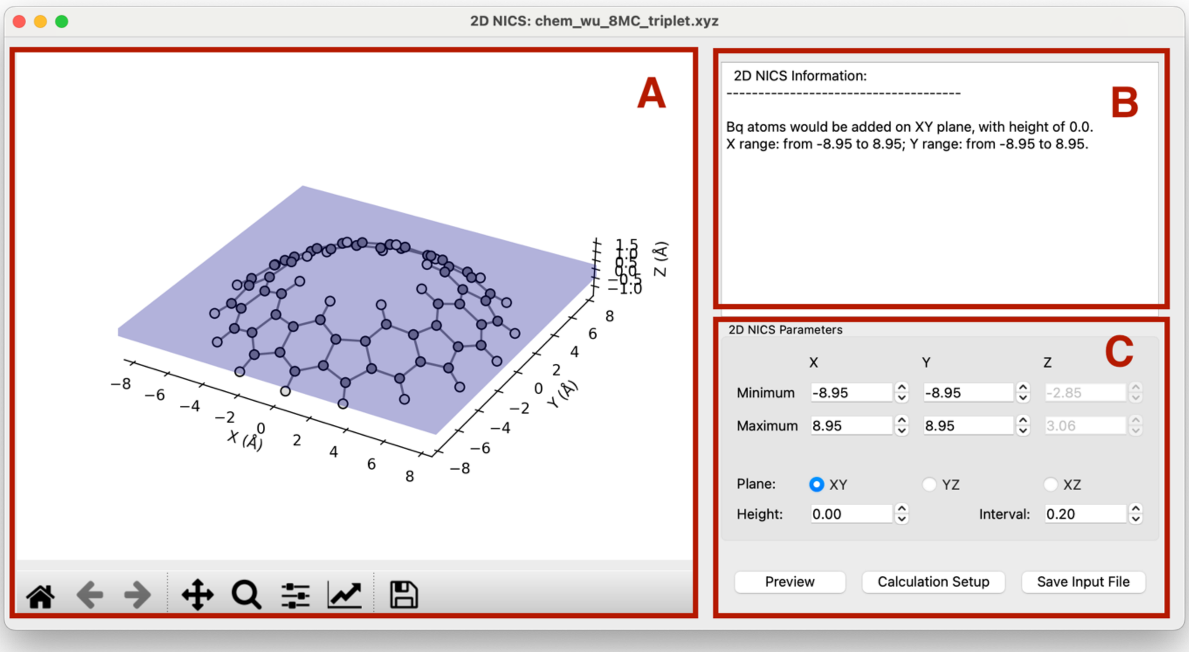




4.6 Create Input File for 2D NICS

4.6.1 Basic Information

The 2D NICS (ICSS) function can be accessed by clicking the 2D NICS button after opening a file. **Area A** provides a preview of the molecular structure and the region for NICS probes. Information about the probe region is summarized in **area B**. In **area C**, users can adjust the probe region, modify the height, and set the interval for ghost atoms.



2D NICS: chem_wu_8MC_triplet.xyz

2D NICS Information:

Bq atoms would be added on XY plane, with height of 0.0.
X range: from -8.95 to 8.95; Y range: from -8.95 to 8.95.

2D NICS Parameters

	X	Y	Z
Minimum	-8.95	-8.95	-2.85
Maximum	8.95	8.95	3.06

Plane: [✓ XY] YZ XZ

Height: 0.00 Interval: 0.20

If the 2D NICS parameters changed, the graph in **area A** will not be updated unless the user clicks the "Preview" button in **area C**.

4.6.2 Example: AWA Molecule

Follows is a step-by-step tutorial for how to use 2D NICS module in py.**Aroma 4**. The molecule in this section is taken from **8MC** in *Chem*, **2018**, *4*, 1586 by Wu *et al.*

The molecule is placed parallel to XY plane, we want to put the NICS probes on the XY plane (target I) and XZ plane (target II), *i.e.,* height is 0, to reproduce the Figure 3b in the original literature. The interval of 0.2 Å (default value in py.**Aroma 4**) would be used.

< Procedure >

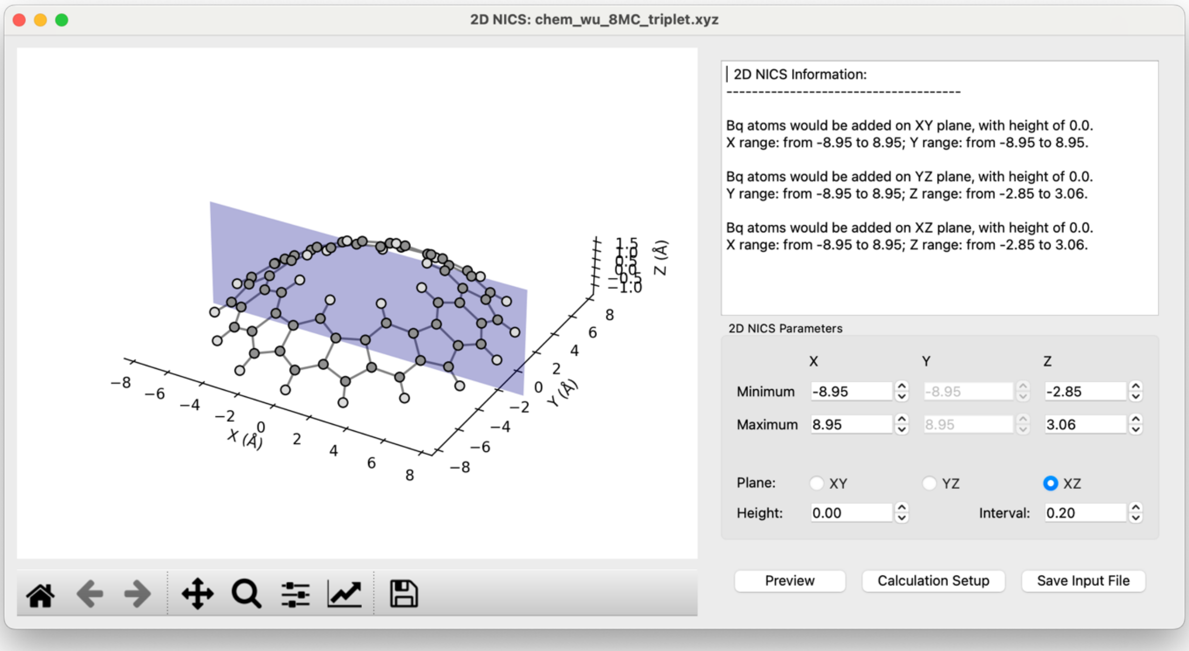
(0) Open *chem_wu_8MC_triplet.xyz* in py.**Aroma 4**.

(1) Click "Save Input File", by this time we can get the *.gjf* file for the target I.

(2) Choose "XZ" in "Plane:" option, and the "Y" parameters would be disabled, and "Z" parameters would be enabled.

(3) Click “Preview” to check the probe region.

(4) Click “Save Input File”, by this time we can get the *.gjf* file for the target II.



2D NICS Information:

Bq atoms would be added on XY plane, with height of 0.0.
X range: from -8.95 to 8.95; Y range: from -8.95 to 8.95.

Bq atoms would be added on YZ plane, with height of 0.0.
Y range: from -8.95 to 8.95; Z range: from -2.85 to 3.06.

Bq atoms would be added on XZ plane, with height of 0.0.
X range: from -8.95 to 8.95; Z range: from -2.85 to 3.06.

2D NICS Parameters

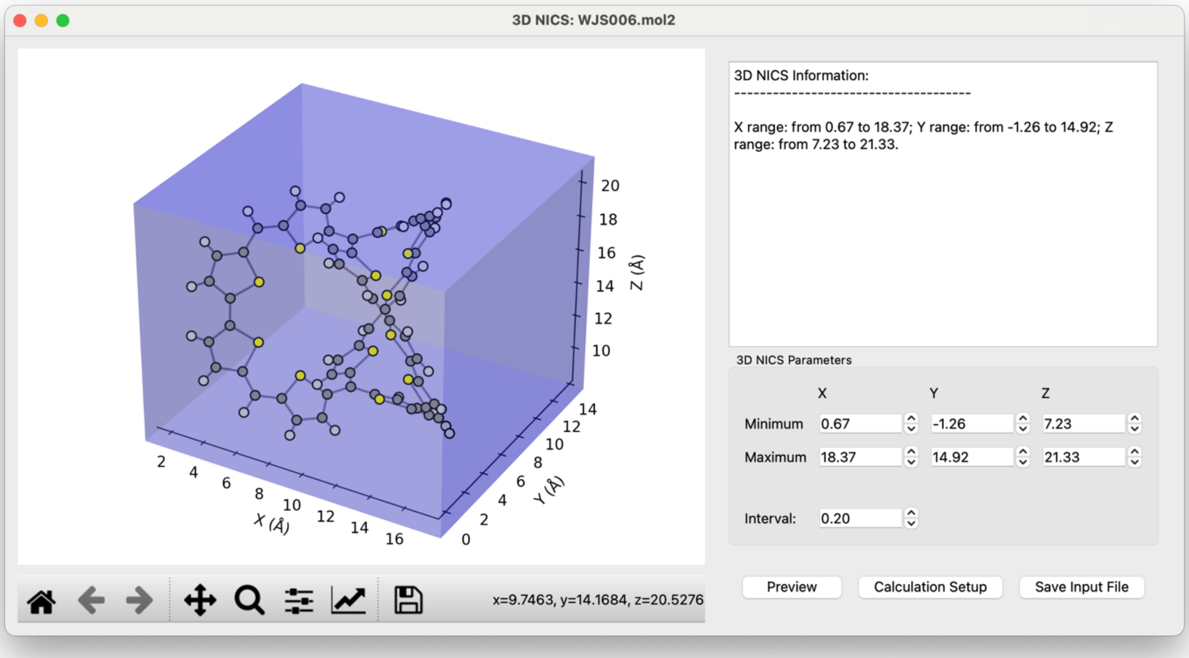
	X	Y	Z
Minimum	-8.95	-8.95	-2.85
Maximum	8.95	8.95	3.06

Plane: XY YZ [✓ XZ]

Height: 0.00 Interval: 0.20

4.7 Create Input File for 3D NICS

The 3D NICS (ICSS) function can be accessed by clicking the 3D NICS button after opening a file. The usage of the 3D NICS module is very similar to that of the 2D NICS module described in *Section 4.6*. Users just need to define the probe region by the limitation on X, Y, Z direction, and the interval between ghost atoms., then, click "Save Input File" button. Further details will be skipped. An example, ***c*-T12**$^{4+}$ from *Nat. Chem.*, **2020**, *12*, 242 is shown below.


3D NICS: WJS006.mol2

3D NICS Information:

X range: from 0.67 to 18.37; Y range: from -1.26 to 14.92; Z range: from 7.23 to 21.33.

3D NICS Parameters

	X	Y	Z
Minimum	0.67	-1.26	7.23
Maximum	18.37	14.92	21.33

Interval: 0.20

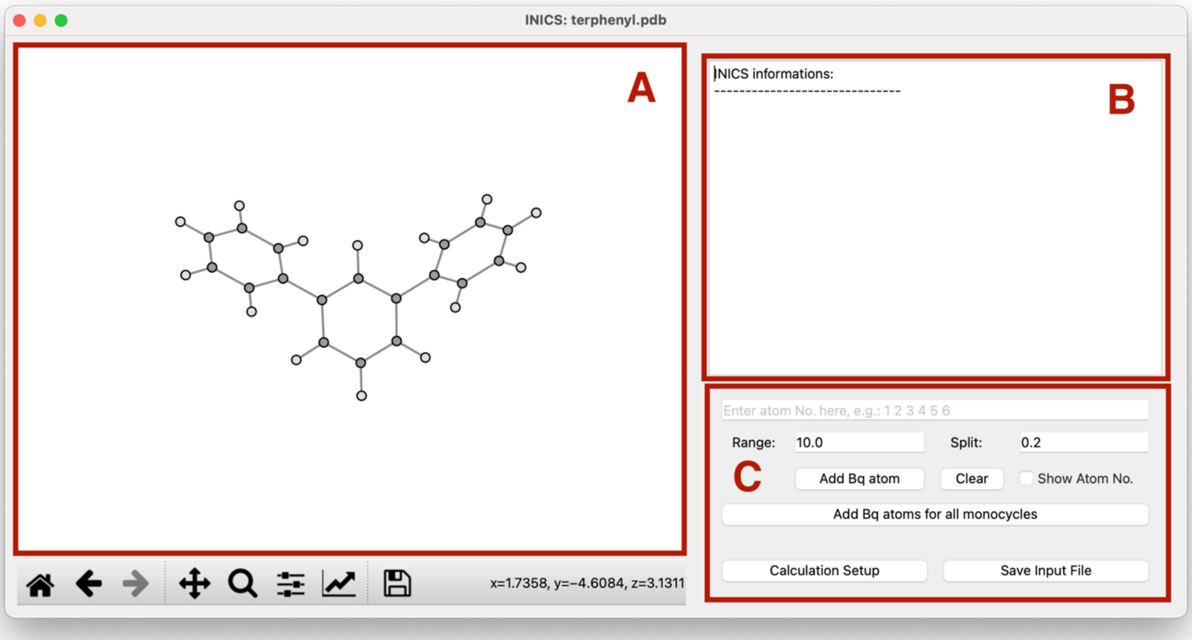
x=9.7463, y=14.1684, z=20.5276

From Developer: *Gaussian* might have a limit on the number of ghost atoms, and it is somehow dependent of the version of *Gaussian* and the computer. To make *Gaussian* run normally, I set a limit to py.**Aroma 4** on the number of ghost atoms to 7000 in a single *.gjf* file. Thus, especially for 3D NICS (ICSS), several files might be generated depends on the probe region, naming start from "_0001*.gjf*".

4.8 Create Input File for INICS

4.8.1 Basic Information

The INICS function can be accessed by clicking the INICS button after opening a file. Main window of INICS module includes three parts: The molecular geometry will be displayed in **area A**, and **area C** allows users to modify the INICS parameters and add ghost atoms. The information about INICS probes will be summarized in **area B**.



Users are required to specify the range and split value (interval) in **area C**, for adding ghost atoms. By default, the range/split are set to 10.0/0.2, which will add ghost atoms from –10 Å below the ring plane to 10 Å above the ring plane, and the neighboring ghost atoms are separated in 0.2 Å. The default setting is accurate enough for normal systems.

If there are more than one monocycle in the geometry, users can add ghost atoms for all cyclic unit at once by clicking “Add Bq atoms for all monocycles”, just like describe in previous. Of course, users can choose the ring of their interest by inputting the atom indexes in the text bar. The ghost atoms will be updated in **area A** after clicking “Add Bq atoms …”.

Note: Although py.**Aroma 4** allows users add NICS probes for a random user defined cycle, its output file could not be analyzed by py.**Aroma 4** in current version.

4.8.2 Example: Terphenyl

Follows is a step-by-step tutorial for how to use INICS module in py.**Aroma 4**. The target molecule is terphenyl, consist of three phenyl rings and two of terminal rings are identical in chemical environment. Our target is adding ghost atom from (–10, 10), split = 0.2, using the default setting.

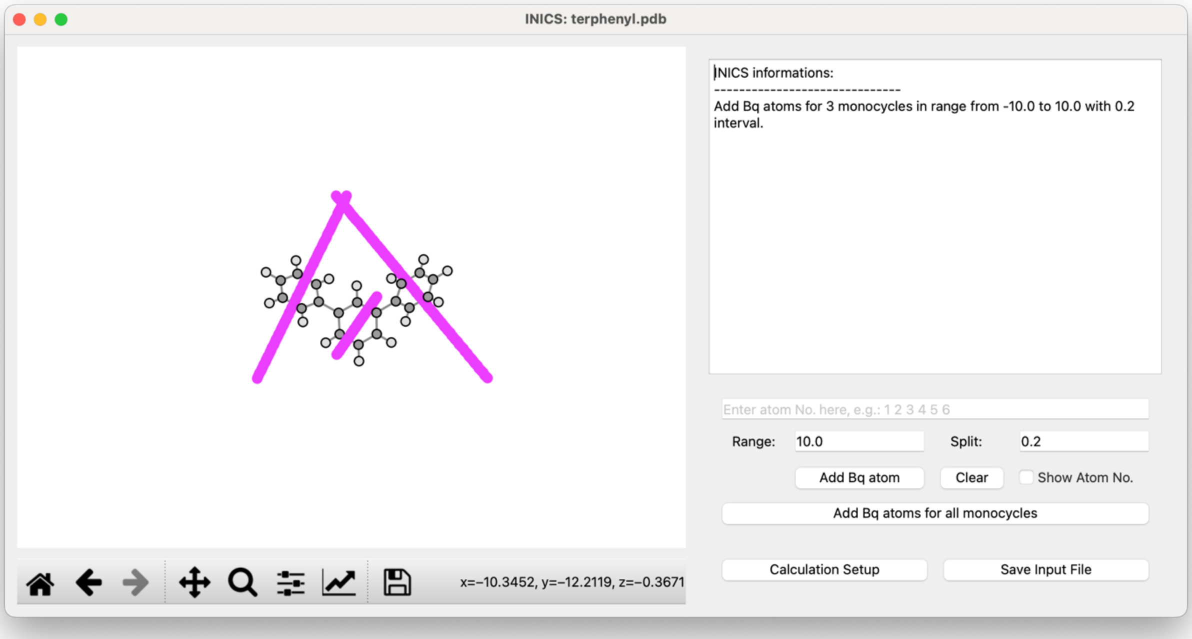
< Procedure >

(0) Open *terphenyl.pdb* in py.**Aroma 4**.

(1) Check "Show Atom No.".

(2) Click "Add Bq atoms for all monocycles".

(3) Click "Save Input File".



INICS informations:

Add Bq atoms for 3 monocycles in range from -10.0 to 10.0 with 0.2 interval.

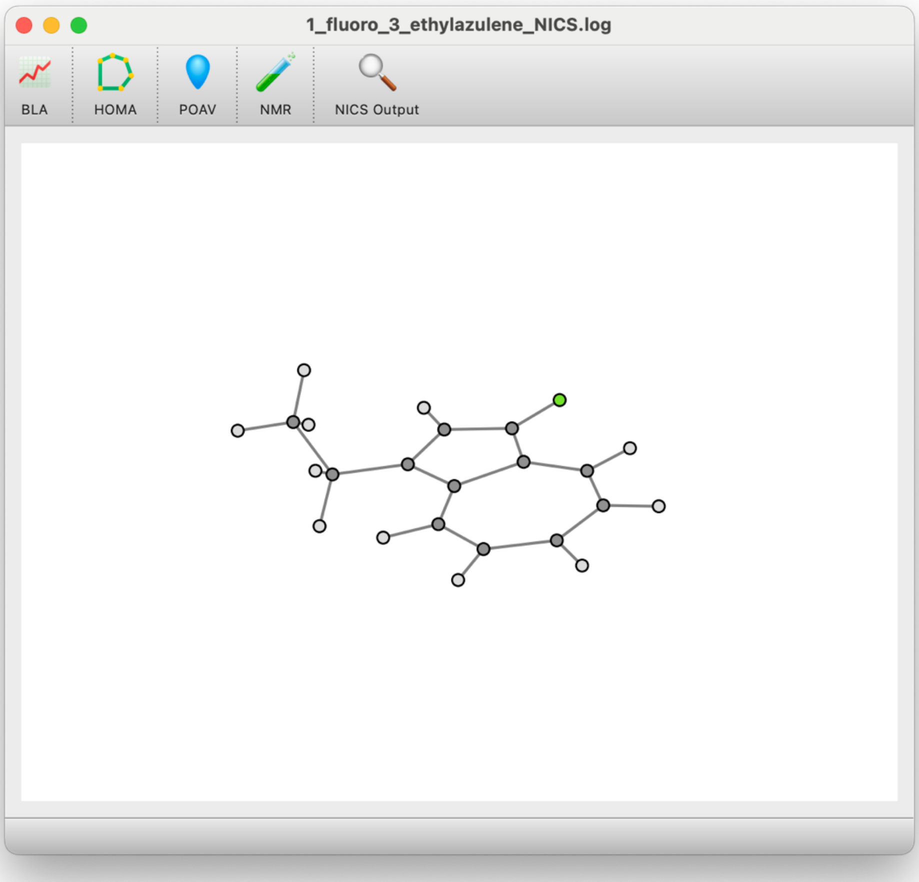
Range: 10.0 Split: 0.2

x=−10.3452, y=−12.2119, z=−0.3671

4.9 Analyze Output Files for NICS Calculation

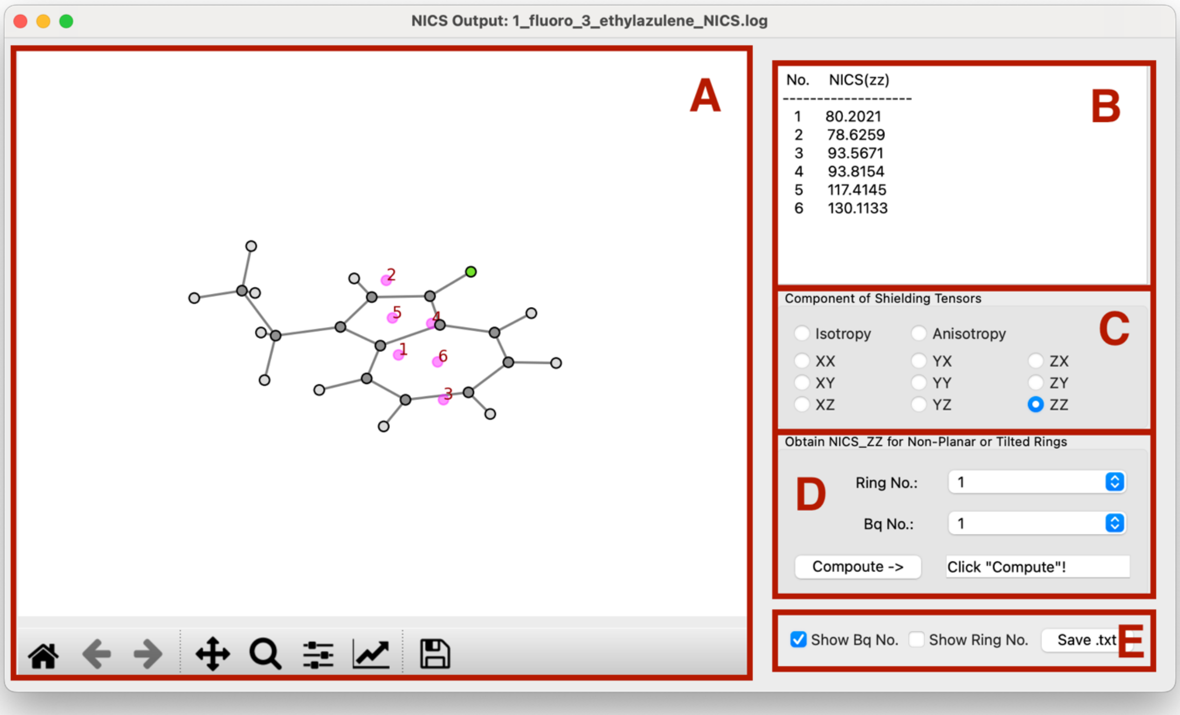
4.9.1 Basic Information

py.**Aroma 4** software is able to automatically extract shielding tensors from a *Gaussian* output file. When reading an output file with ghost atoms, the geometry without the ghost atoms will be displayed on the main window. Users can access the BLA, HOMA, and POAV modules from the current window, and analyze the NICS output by clicking the "NICS Output" button.



4.9.2 Single Point NICS

For NICS, the geometry with ghost atoms would be displayed in **area A**, and the shielding tensors are summarized in **area B**. The atom indices of ghost atoms could be hide by unchecking the "Show Bq No." in **area E**. By default, the ZZ component of shielding tensors would be displayed in **area B**. Users can check the radio button in **area C** for other component, and the data in **area B** would be updated automatically. The result in **area B** could be saved as "*.txt*" file by clicking the "Save *.txt*" button in **area E**. **Area D** provide a function for calculate pseudo-NICS$_{ZZ}$, *i.e.*, NICS$_{\perp}$ for non-planar or tilted rings.



NICS Output: 1_fluoro_3_ethylazulene_NICS.log

```
No.   NICS(zz)
------------------
1     80.2021
2     78.6259
3     93.5671
4     93.8154
5     117.4145
6     130.1133
```

Component of Shielding Tensors

Isotropy   Anisotropy
XX   YX   ZX
XY   YY   ZY
XZ   YZ   [✓ ZZ]

Obtain NICS_ZZ for Non-Planar or Tilted Rings

Ring No.: 1

Bq No.: 1

Compoute ->   Click "Compute"!

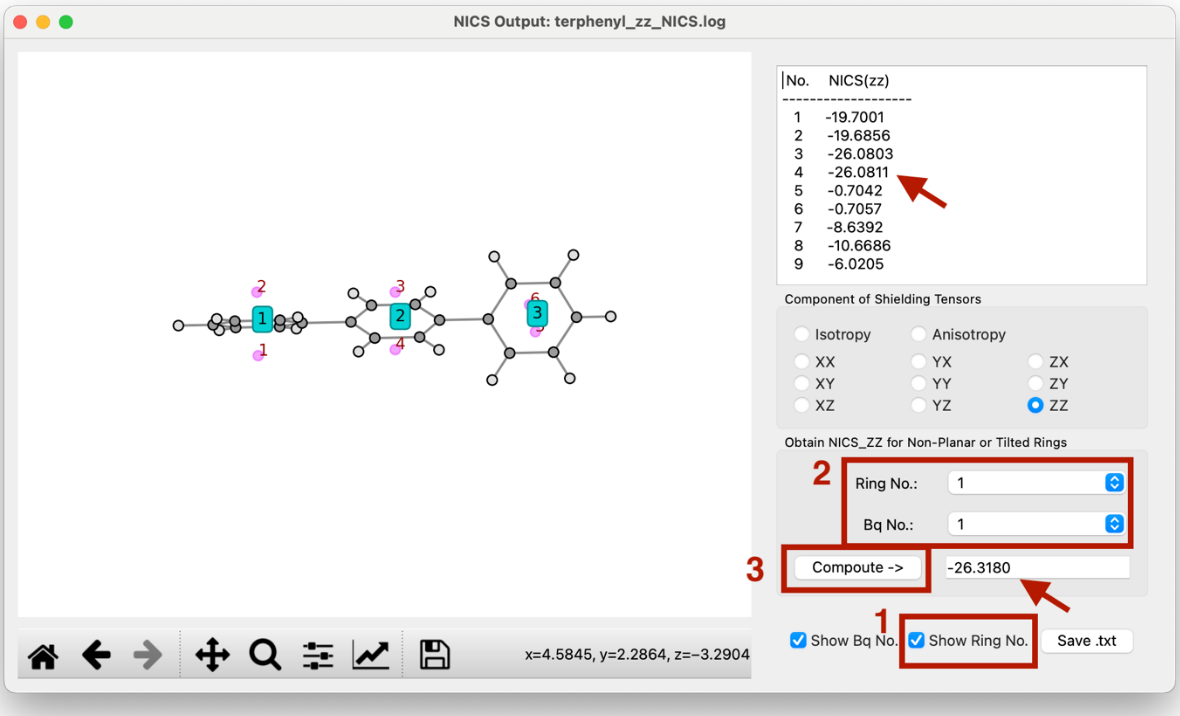
[✓ Show Bq No.]   Show Ring No.   Save .txt

For disordered ring or rings those are not parallel to Cartesian plane, py.**Aroma 4** provides a useful function for compute pseudo-NICS$_{ZZ}$ based on shielding tensors. Follows is an example for a terphenyl molecule, the middle phenyl ring (No. 2) is placed on the XY plane, whose NICS$_{ZZ}$ will be the reference. The left phenyl ring (No. 1) has a dihedral angle with XY plane of 30°, and right phenyl ring (No. 3) is perpendicular to XY plane.

The NICS(1)$_{ZZ}$ of ring No. 2 is –26.0803, indicating its high aromaticity. On the other hand, the NICS(1)$_{ZZ}$ from output file are –19.7001 and –0.7057 for ring No.1 and No. 3, respectively. Since the ring No. 1 and No. 3 are not placed on (or parallel) to XY plane, the ZZ shielding tensors are no longer reliable.

Compute the shielding tensor on the direction that perpendicular to the ring plane, *i.e.*, NICS$_{\perp}$ in py.**Aroma 4** is easy. Users just need to select the ring No. and Bq No. in "Obtain NICS_ZZ for Non-Planar or Tilted Rings" section, then click "Compute ->". The pseudo-NICS$_{ZZ}$ (NICS$_{\perp}$) value will be computed and displayed in the text bar.

For example, the NICS(1)$_{\perp}$ of ring No. 1 and No. 2 were calculated to –26.3180 and –28.6472, respectively, much close to NICS(1)$_{ZZ}$ of ring No. 2. Since ring No. 3 is nearly parallel to XZ plane, the NICS(1)$_{YY}$ = –26.6393 could be treated as NICS(1)$_{\perp}$.


NICS Output: terphenyl_zz_NICS.log
1
2
3
Component of Shielding Tensors
Isotropy
Anisotropy
XX
YX
ZX
XY
YY
ZY
XZ
YZ
[✓ ZZ]
Obtain NICS_ZZ for Non-Planar or Tilted Rings
Compoute ->
Show Bq No.
Show Ring No.
Save .txt
x=4.5845, y=2.2864, z=-3.2904

```
|No.    NICS(zz)
-------------------
  1    -19.7001
  2    -19.6856
  3    -26.0803
  4    -26.0811
  5    -0.7042
  6    -0.7057
  7    -8.6392
  8    -10.6686
  9    -6.0205
```

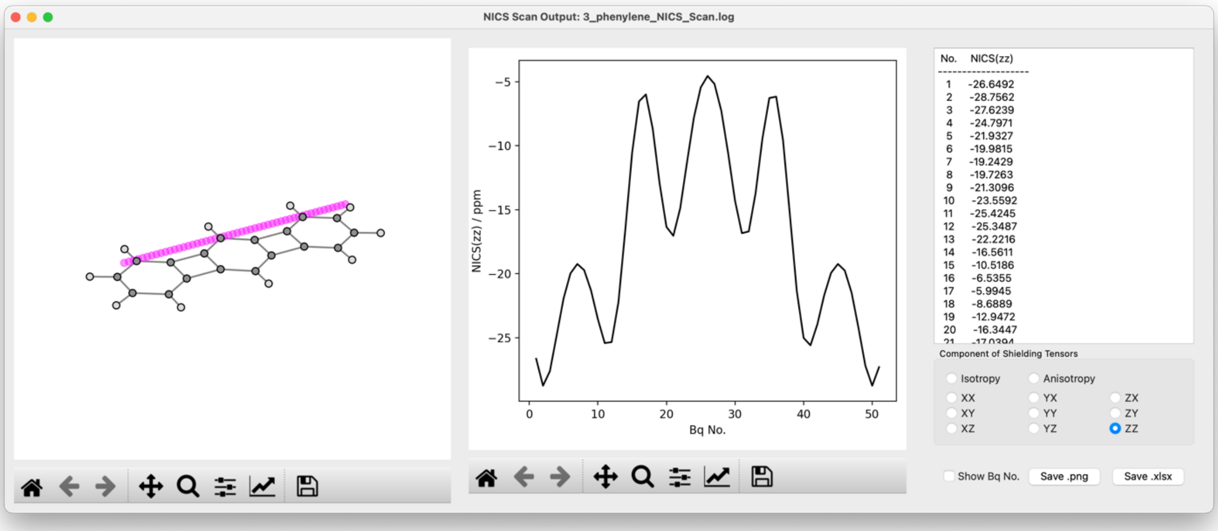
Ring No.: 1

Bq No.: 1

-26.3180

4.9.3 NICS Scan

The main function of the NICS scan output is almost the same as NICS. The atom indices of ghost atoms are hidden by default for clarity, and the NICS scan plot is displayed in the middle part. When the component of the shielding tensor is changed, the NICS scan plot is also automatically updated. Users can save the plot by clicking the "Save *.png*" button and save the data to a *.xlsx* file by clicking the "Save *.xlsx*" button.

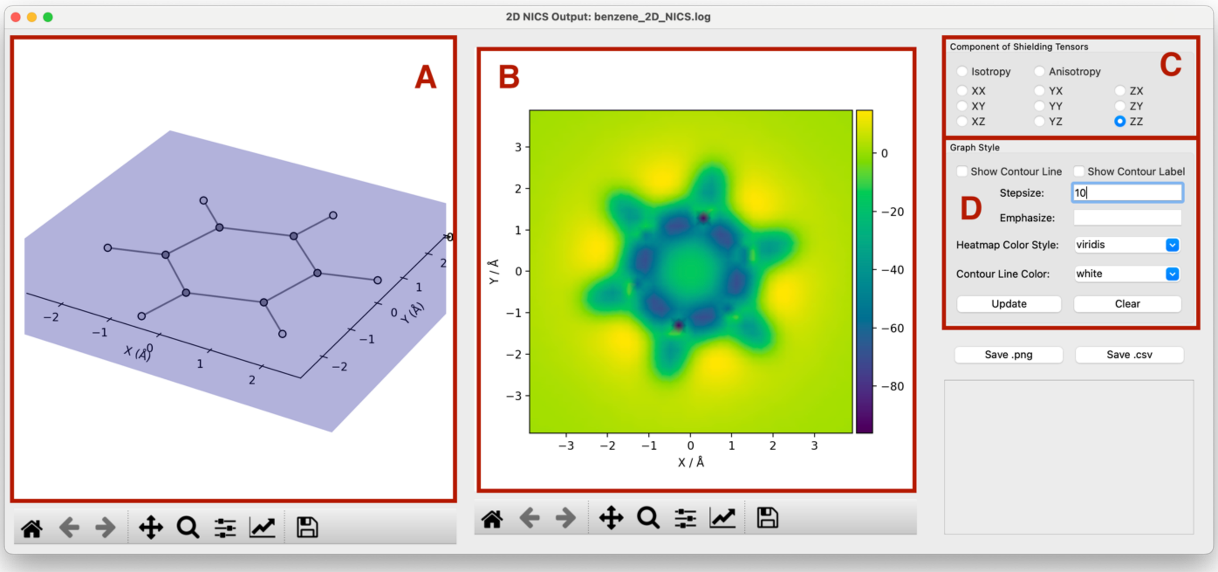


```
No.    NICS(zz)
-------------------
 1    -26.6492
 2    -28.7562
 3    -27.6239
 4    -24.7971
 5    -21.9327
 6    -19.9815
 7    -19.2429
 8    -19.7263
 9    -21.3096
10    -23.5592
11    -25.4245
12    -25.3487
13    -22.2216
14    -16.5611
15    -10.5186
16    -6.5355
17    -5.9945
18    -8.6889
19    -12.9472
20    -16.3447
21    -17.0394
```

4.9.4 2D NICS/ICSS

The main window of the 2D NICS/ICSS output is shown below. In **area A**, there is a preview of the molecule and the region of NICS probes (colored in pale purple). In **area B**, a heatmap of the 2D NICS/ICSS plot will be displayed. Users can modify the component of the shielding tensors from **area C**, and the heatmap will be automatically updated. Users can save the plot as an image by clicking the "Save *.png*" button and save the data to a *.csv* file by clicking the "Save *.csv*" button in **area D**.

In **area D**, users can customize the heatmap. To show contour lines in the heatmap, users can check the "Show Contour Line" box, and the interval of the contour lines is controlled in the "Stepsize" text bar. If users want to highlight a specified contour line, just input the value in the "Emphasize" text bar and click "Update". The label of the highlighted contour line can be added by checking the "Show Contour Label" option.


2D NICS Output: benzene_2D_NICS.log

Component of Shielding Tensors: Isotropy, Anisotropy, XX, YX, ZX, XY, YY, ZY, XZ, YZ, [✓ ZZ]

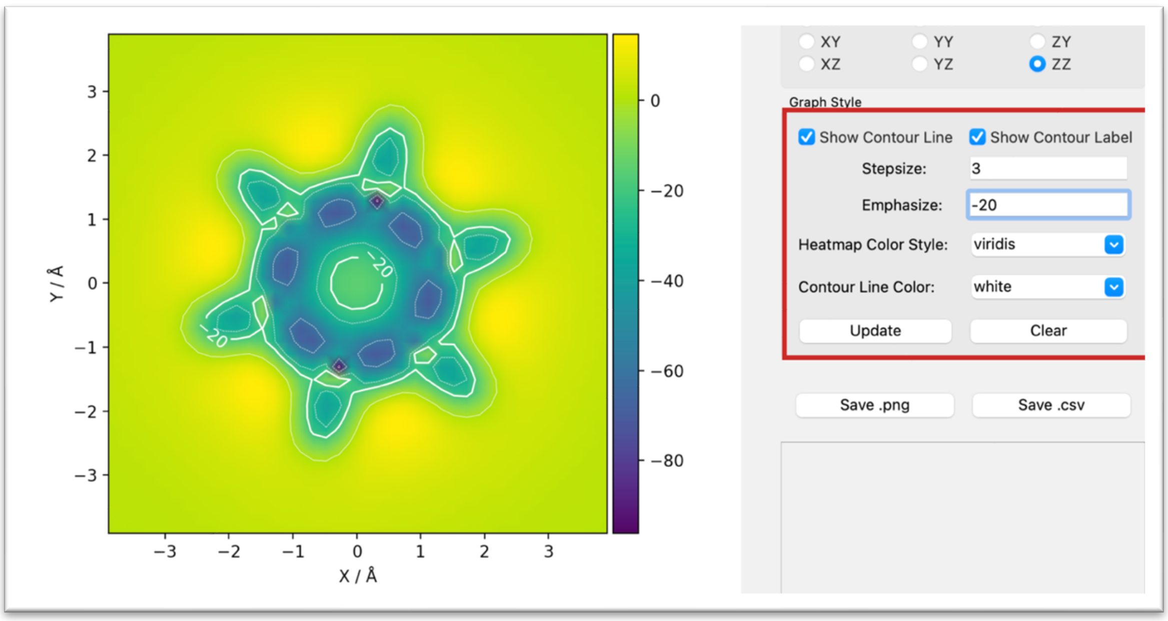
Graph Style: ☐ Show Contour Line ☐ Show Contour Label

Stepsize: 10

Emphasize:

Heatmap Color Style: viridis

Contour Line Color: white


XY, YY, ZY, XZ, YZ, [✓ ZZ]

Graph Style: ✓ Show Contour Line ✓ Show Contour Label

Stepsize: 3

Emphasize: -20

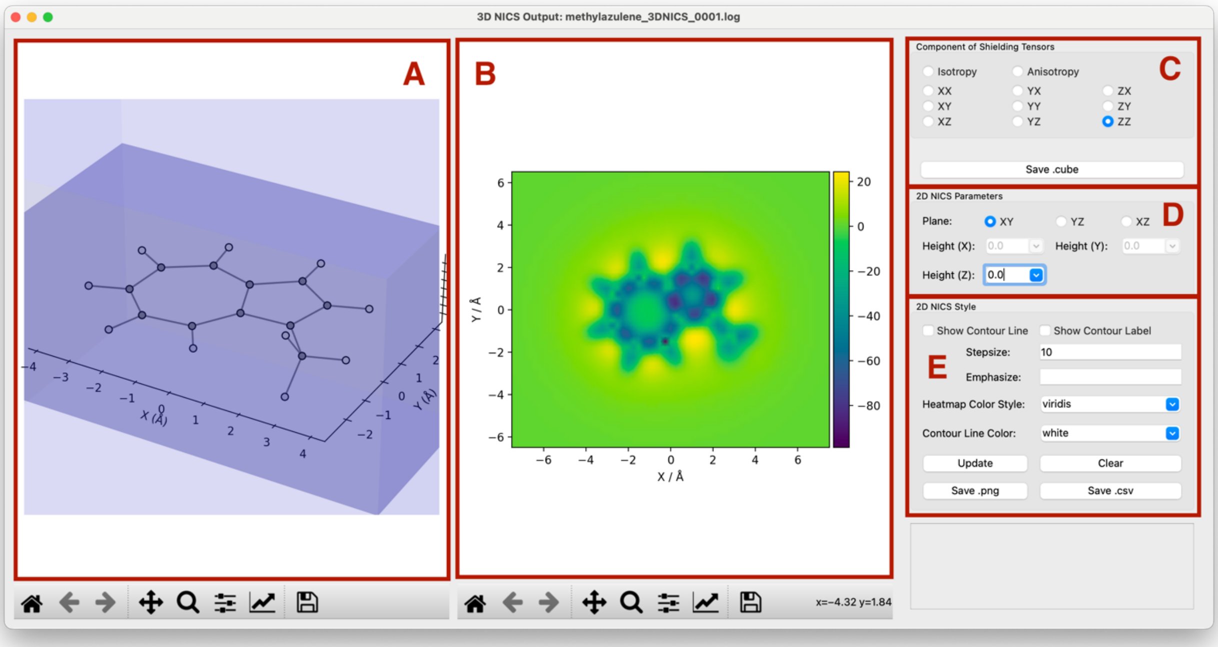
Heatmap Color Style: viridis

Contour Line Color: white

The heatmap function in py.**Aroma 4** is powered by *Matplotlib*, so the color definition follows the rules of *Matplotlib*. Users can change the color style of the heatmap and the color of the contour line from the list in **area D**. Colors not listed are also supported, as long as it follows the rule of *Matplotlib*, and users can simply input the name of the color and click "Update". For more information about color definition in *Matplotlib*, please refer to the Appendix I for colormap and Appendix II for single color specifying.

4.9.5 3D NICS/ICSS

The main window of the 3D NICS/ICSS output is shown below. In **area A**, there is a preview of the molecule and the region of NICS probes. The 2D NICS/ICSS function is also supported in the 3D NICS/ICSS module, thus, a 2D heatmap on the XY plane is displayed in **area B** by default. Users can choose the plane and height in **area D** and customize the plotting style in **area E**, similar to the options available in the 2D NICS/ICSS module. In **area C**, users can choose which component of the shielding tensors to investigate, and the data can be saved as a *.cube* file.


3D NICS Output: methylazulene_3DNICS_0001.log

Component of Shielding Tensors: Isotropy, Anisotropy, XX, XY, XZ, YX, YY, YZ, ZX, ZY, [✓ ZZ]

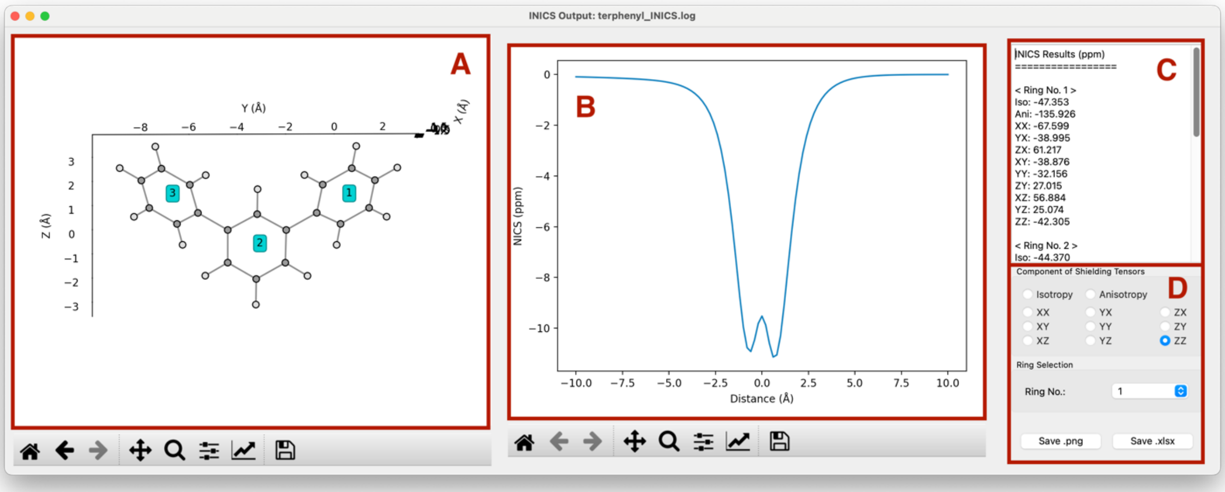
2D NICS Parameters:
Plane: [✓ XY] YZ XZ
Height (X): 0.0
Height (Y): 0.0
Height (Z): 0.0

2D NICS Style:
Show Contour Line ☐
Show Contour Label ☐
Stepsize: 10
Emphasize:
Heatmap Color Style: viridis
Contour Line Color: white

x=-4.32 y=1.84

4.9.6 INICS

The main window of the INICS output is shown below. In **area A**, there is a preview of the molecule with the ring indexes. A distance versus NICS plot will be plotted in **area B**, using NICS$_{ZZ}$ for the ring No. 1. Users can choose the tensor type and ring No. from the button group and list in **area D**. The **area C** summarized INICS value of all cycles based on all kinds of component. The plot in **area B** could be saved as *.png* file and all shielding tensors could be saved in a *.xlsx* file via the button in **area E**.

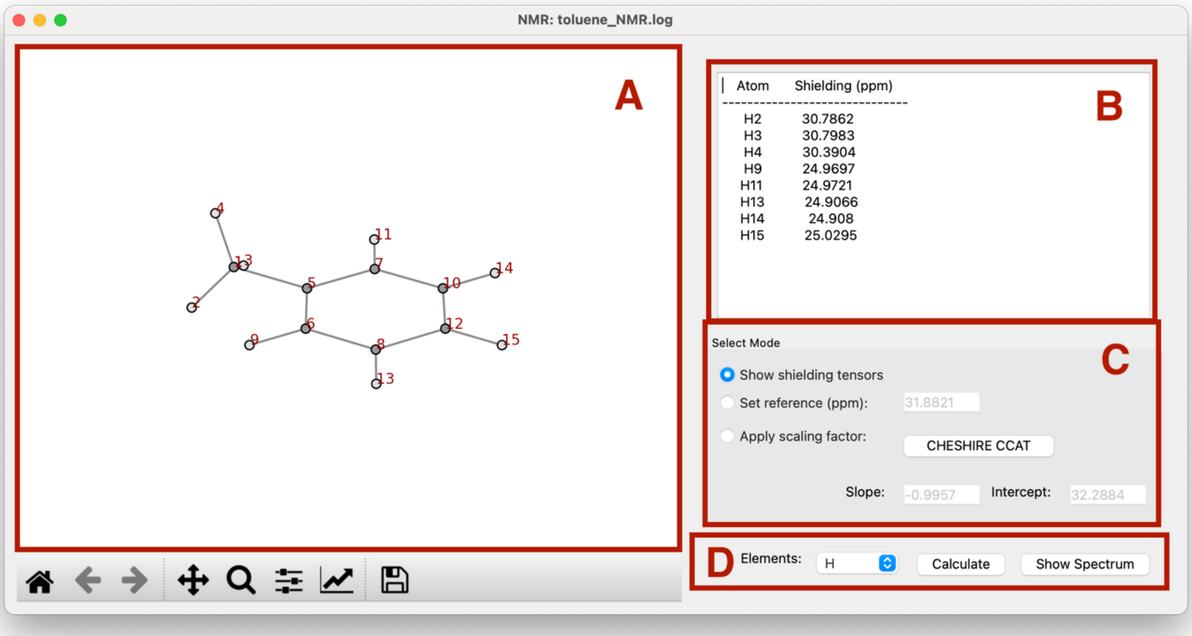


```
NICS Results (ppm)
=================

< Ring No. 1 >
Iso: -47.353
Ani: -135.926
XX: -67.599
YX: -38.995
ZX: 61.217
XY: -38.876
YY: -32.156
ZY: 27.015
XZ: 56.884
YZ: 25.074
ZZ: -42.305

< Ring No. 2 >
Iso: -44.370
```

4.10 NMR Plotting

The py.**NMR** module would be active when reading a *Gaussian* output file of NMR job. The output files for NICS analyses are also possible to active py.**NMR** module. The NMR module in py.**Aroma 4** is more powerful than py.**NMR**.

4.10.1 Main Function



```
| Atom      Shielding (ppm)
--------------------------------
   H2        30.7862
   H3        30.7983
   H4        30.3904
   H9        24.9697
   H11       24.9721
   H13        24.9066
   H14         24.908
   H15        25.0295
```

[✓ Show shielding tensors]
Set reference (ppm): 31.8821
Apply scaling factor:
Slope: -0.9957 Intercept: 32.2884
Elements: H

Main window of py.**NMR** module is shown as above. Just like other modules, the geometry (without ghost atoms) will be displayed in **area A**. The isotropic shielding tensors will be displayed in **area B**. Users can select the element of interest in the molecule form the "Element" list in **area D**.

py.**NMR** module provides three modes to process shielding tensors, users can choose from **area C**. The "Show shielding tensors" will do nothing on the isotropic shielding tensors. "Set reference" allows users to input a reference value to compute chemical shift, the default value is shielding tensor of protons in trimethylsilane (TMS) calculated at B3LYP/6-311+G(2d,p) level of theory. "Apply scaling factor" allows users to scale the chemical shift based on the empirical scaling factors. The scaling factor could be found in CHESHIRE CCAT[6] website, and users can easily access the website by click "CHESHIRE CCAT" button.

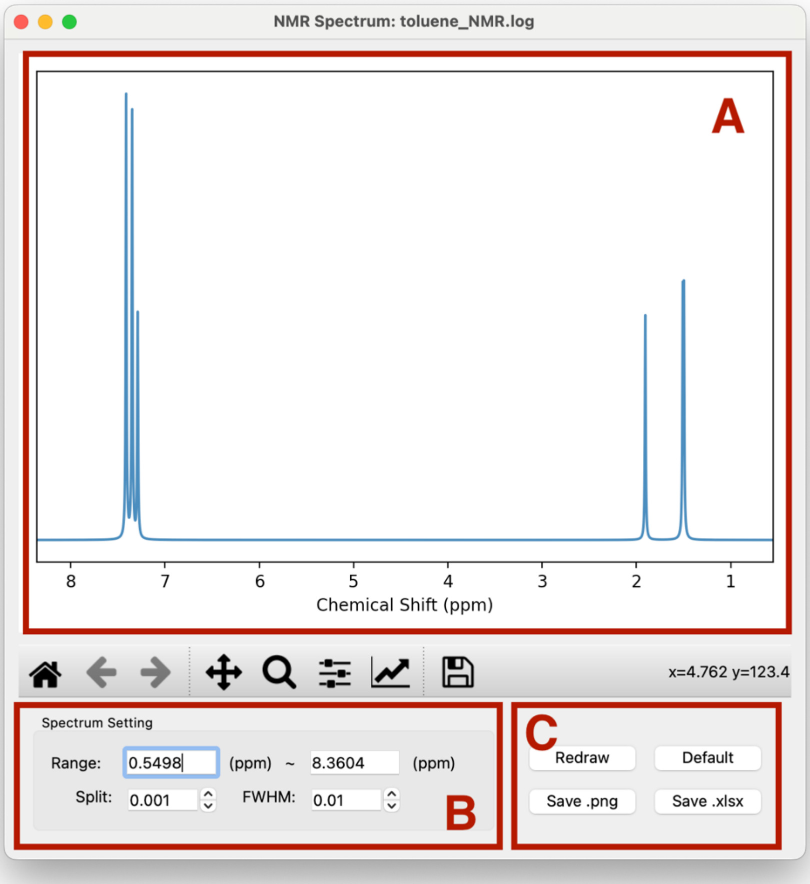
Users need to click "Calculate" in **area D** to update the shielding tensors/chemical shift if the mode or element changed, the new results will be updated in **area B**. And users can also check the NMR spectrum by clicking "Show Spectrum" button.

[6] http://cheshirenmr.info/ScalingFactors.htm

4.10.2 NMR Spectrum

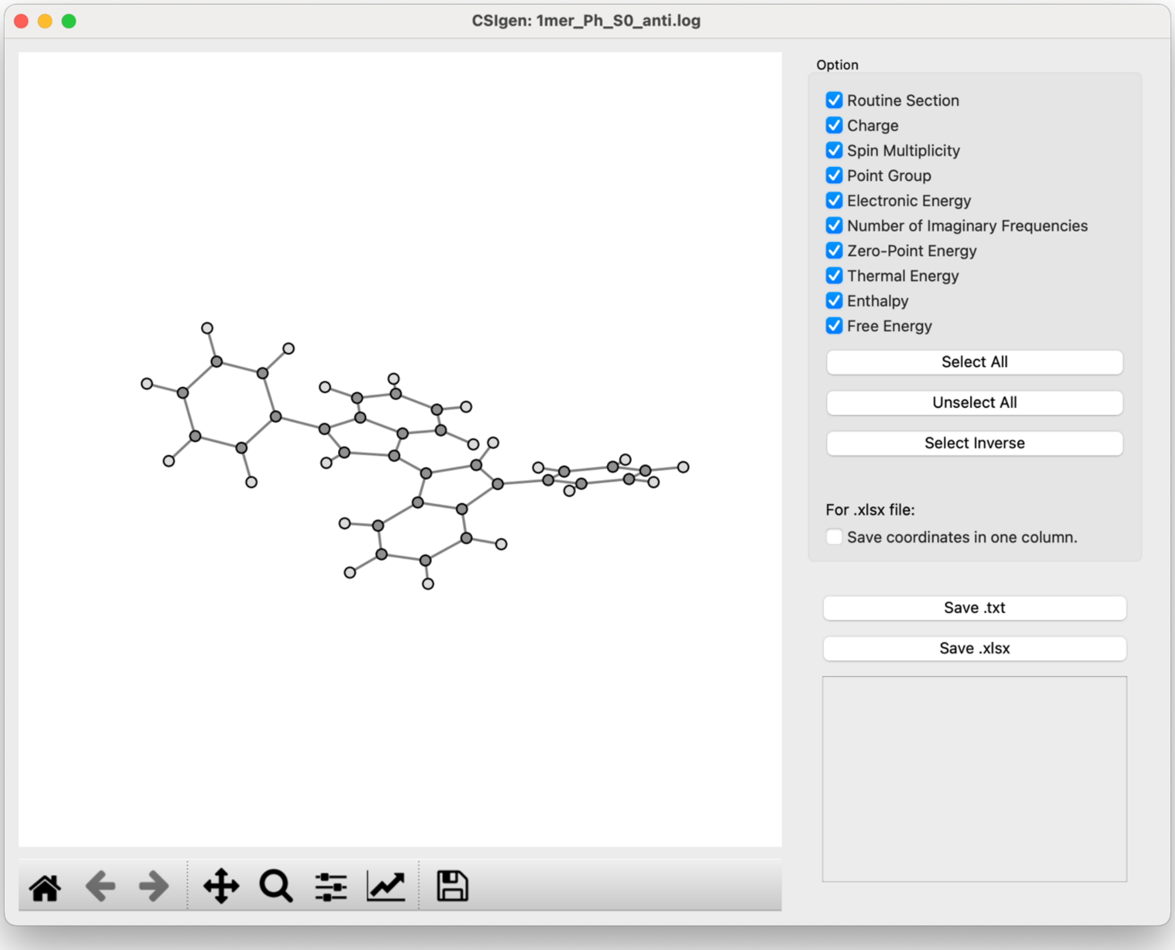
In NMR spectrum window, the NMR spectrum will be plotted in **area A**. Users can modify the plotting parameters like plot range, split and FWHM in **area B**. A smaller split value will give a smoother spectrum, and a smaller FWHM value will give sharper peaks (See Appendix IV). If parameters changed, users are required to click "Redraw" button to update the spectrum, and "Default" button allows user use default parameters.

The NMR spectrum can be saved as *.png* and *.xlsx* via the buttons in **area D**.



4.11 Generate Computational Supporting Material

The **CSIgen** module would be active when reading a *Gaussian* output file without ghost atoms to py.**Aroma 4**. The main window of **CSIgen** module is show below.



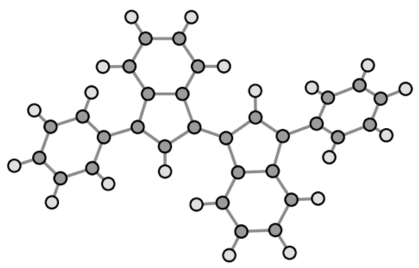
The geometry of the molecule is displayed in left area. In the check list at right part, users can choose the information needed for the supporting information. By default, all information is selected. The options "Number of Imaginary Frequencies", "Zero-Point Energy", "Thermal Energy", "Enthalpy" and "Free Energy" are only available when a `freq` job is detected.

Users can choose to save the supporting information as a *.txt* or *.xlsx* file. When saving as a *.xlsx* file, the Cartesian coordinates are saved in two columns by default, but this can be turned off by checking the "Save coordinates in one column" option.

The follows page provides an example using the **CSIgen** module in py.**Aroma 4** to generate supporting information. The geometry in the upper left was added later.

Hint: When saving the coordinates in two columns, the width is just to fit the "*Moderate*" margins setting in Microsoft Word.

1mer_Ph_S0_anti



```
#p opt freq rwb97xd/6-31g(d,p)
```

Charge = 0, Multiplicity = 1, Point group = C1

Number of imaginary frequencies = 0

Electronic Energy = -1154.87028937 Hartree

Sum of electronic and zero-point Energies = -1154.463745 Hartree

Sum of electronic and thermal Energies = -1154.441758 Hartree

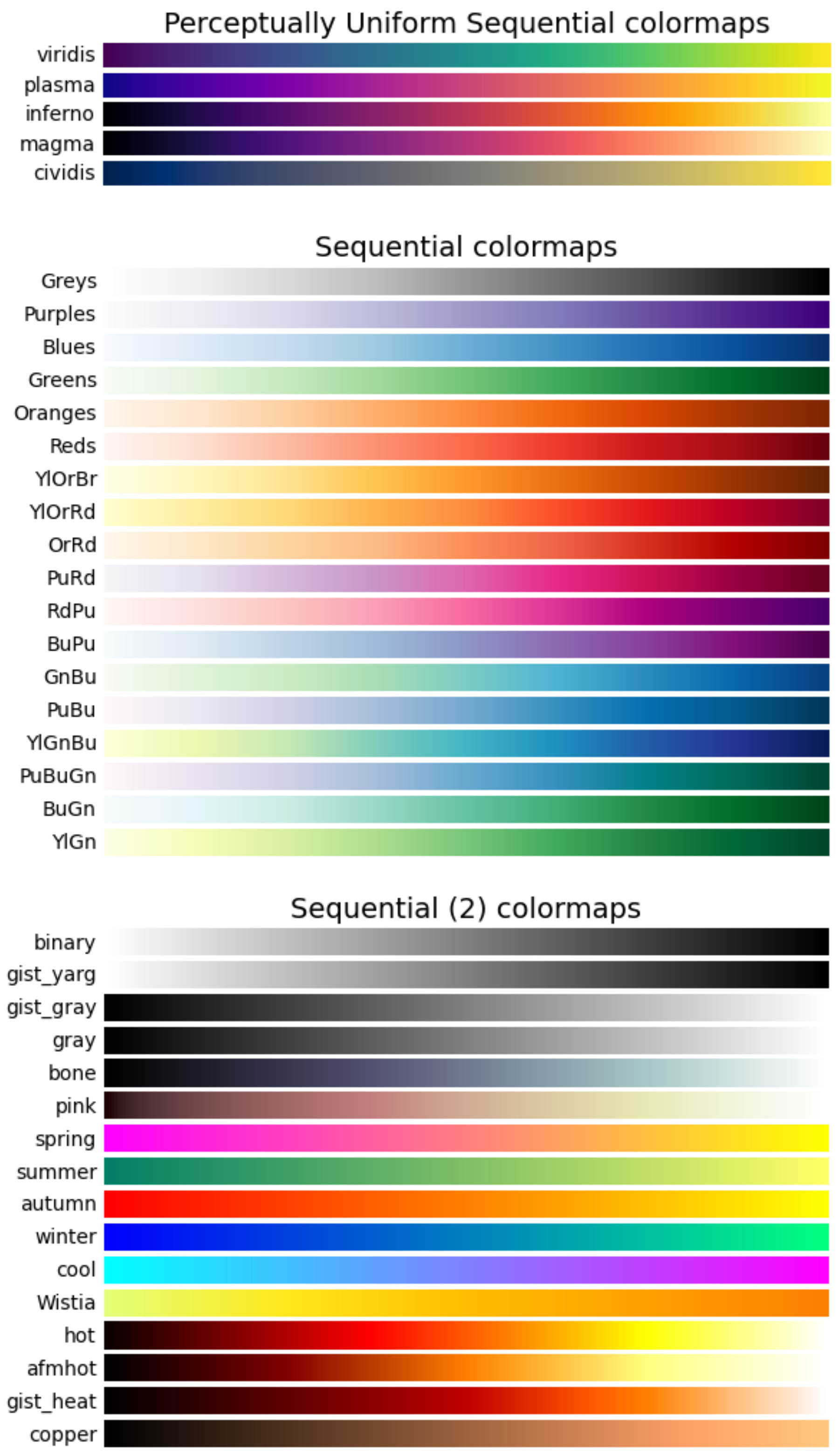
Sum of electronic and thermal Enthalpies = -1154.440814 Hartree

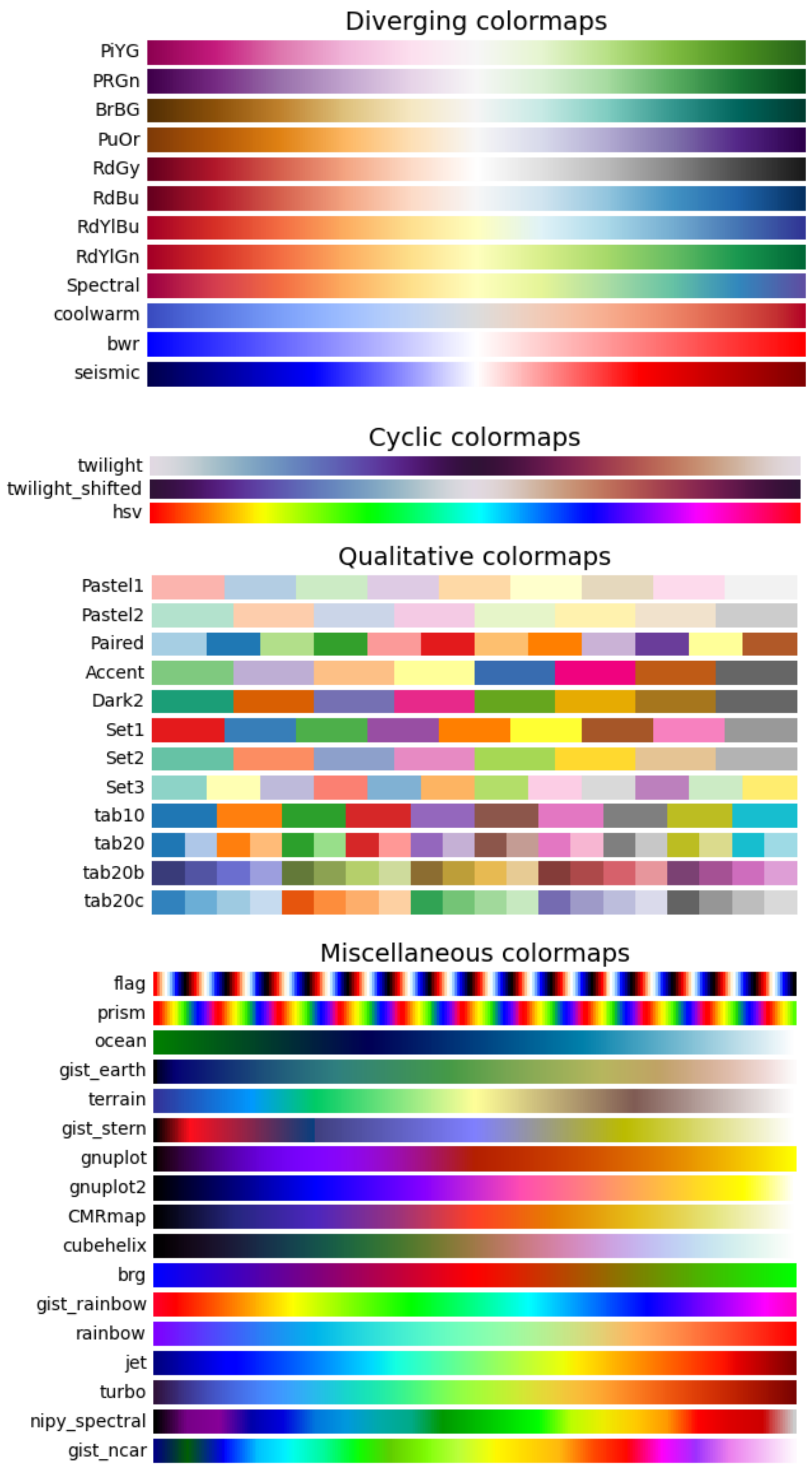
Sum of electronic and thermal Free Energies = -1154.516642 Hartree

	Cartesian Coordinates				Cartesian Coordinates		
Atoms	***X***	***Y***	***Z***	**Atoms**	***X***	***Y***	***Z***
H	4.435677	-2.606388	-0.231397	C	3.356437	-2.664594	-0.328848
C	0.569521	-2.799826	-0.699768	C	2.571309	-1.527073	-0.212788
C	2.735491	-3.884182	-0.603806	C	1.358363	-3.948847	-0.791078
C	1.171317	-1.589434	-0.381275	H	3.334927	-4.784470	-0.693673
H	0.890493	-4.899083	-1.027388	H	-0.491294	-2.867073	-0.908157
C	2.946798	-0.115707	-0.003726	C	1.825138	0.639775	-0.089644
H	1.789433	1.700971	0.109068	C	0.648524	-0.205519	-0.281946
H	-4.435780	2.606331	-0.231253	C	-3.356547	2.664605	-0.328741
C	-0.569663	2.800007	-0.699804	C	-2.571346	1.527130	-0.212737
C	-2.735690	3.884236	-0.603706	C	-1.358578	3.948982	-0.791055
C	-1.171361	1.589573	-0.381277	H	-3.335187	4.784487	-0.693533
H	-0.890777	4.899247	-1.027383	H	0.491130	2.867330	-0.908265
C	-2.946755	0.115735	-0.003721	C	-1.825064	-0.639688	-0.089728
H	-1.789301	-1.700902	0.108896	C	-0.648491	0.205678	-0.281947
C	-4.302591	-0.380861	0.272258	C	-6.864325	-1.376529	0.828160
C	-5.157951	0.298025	1.149401	C	-4.753603	-1.566015	-0.321422
C	-6.022475	-2.060311	-0.044873	C	-6.427119	-0.197450	1.425233
H	-4.812770	1.203896	1.637694	H	-4.104710	-2.086849	-1.019008
H	-6.358149	-2.977689	-0.518023	H	-7.074646	0.336285	2.113683
H	-7.856805	-1.760298	1.041443	C	4.302664	0.380807	0.272255
C	6.864458	1.376320	0.828162	C	5.158007	-0.298169	1.149344
C	4.753723	1.565971	-0.321369	C	6.022624	2.060190	-0.044816
C	6.427205	0.197228	1.425180	H	4.812791	-1.204049	1.637593
H	4.104843	2.086876	-1.018915	H	6.358333	2.977577	-0.517923
H	7.074719	-0.336576	2.113587	H	7.856959	1.760028	1.041449

Appendix

I Colormaps in *Matplotlib*




Diverging colormaps
PiYG
PRGn
BrBG
PuOr
RdGy
RdBu
RdYlBu
RdYlGn
Spectral
coolwarm
bwr
seismic
Cyclic colormaps
twilight
twilight_shifted
hsv
Qualitative colormaps
Pastel1
Pastel2
Paired
Accent
Dark2
Set1
Set2
Set3
tab10
tab20
tab20b
tab20c
Miscellaneous colormaps
flag
prism
ocean
gist_earth
terrain
gist_stern
gnuplot
gnuplot2
CMRmap
cubehelix
brg
gist_rainbow
rainbow
jet
turbo
nipy_spectral
gist_ncar

II Colors in *Matplotlib*

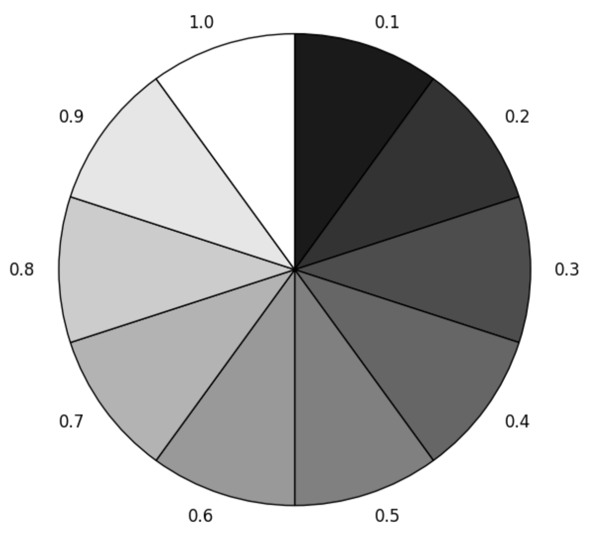
Follows contents are taken from https://matplotlib.org/stable/tutorials/colors/colors.html , *Matplotlib* recognizes the follows formats to specify a color.

Format	Example
RGB or RGBA (red, green, blue, alpha) tuple of float values in a closed interval [0, 1].	• `(0.1, 0.2, 0.5)` • `(0.1, 0.2, 0.5, 0.3)`
Case-insensitive hex RGB or RGBA string.	• `'#0f0f0f'` • `'#0f0f0f80'`
Case-insensitive RGB or RGBA string equivalent hex shorthand of duplicated characters.	• `'#abc'` as `'#aabbcc'` • `'#fb1'` as `'#ffbb11'`
String representation of float value in closed interval [0, 1] for grayscale values.	• `'0'` as black • `'1'` as white • `'0.8'` as light gray
Single character shorthand notation for some basic colors.	• `'b'` as blue • `'g'` as green • `'r'` as red • `'c'` as cyan • `'m'` as magenta • `'y'` as yellow • `'k'` as black • `'w'` as white
Case-insensitive X11/CSS4 color name with no spaces.	• `'aquamarine'` • `'mediumseagreen'`
Case-insensitive color name from xkcd color survey with 'xkcd:' prefix.	• `'xkcd:sky blue'` • `'xkcd:eggshell'`
Case-insensitive Tableau Colors from 'T10' categorical palette.	• `'tab:blue'` • `'tab:orange'` • `'tab:green'` • `'tab:red'` • `'tab:purple'` • `'tab:brown'` • `'tab:pink'` • `'tab:gray'` • `'tab:olive'` • `'tab:cyan'`
"CN" color spec where 'C' precedes a number acting as an index into the default property cycle.	• `'C0'` • `'C1'`

And here are some color names available in *matplotlib*.

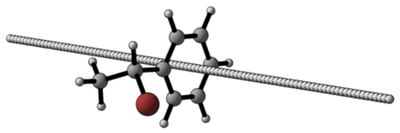
⬛	black	⬛	dimgrey	⬛	dimgray	⬛	gray
⬛	grey	⬜	darkgrey	⬜	darkgray	⬜	silver
⬜	lightgray	⬜	lightgrey	⬜	gainsboro	⬜	whitesmoke
⬜	white	⬜	snow	🟫	rosybrown	🟥	lightcoral
🟥	indianred	🟥	brown	🟥	firebrick	🟥	maroon
🟥	darkred	🟥	red	⬜	mistyrose	🟥	salmon
🟥	tomato	🟧	darksalmon	🟧	coral	🟧	orangered
🟧	lightsalmon	🟫	sienna	⬜	seashell	🟫	chocolate
🟫	saddlebrown	🟧	sandybrown	🟧	peachpuff	🟫	peru
⬜	linen	⬜	bisque	🟧	darkorange	🟫	burlywood
⬜	antiquewhite	🟫	tan	🟨	navajowhite	⬜	blanchedalmond
⬜	papayawhip	🟨	moccasin	🟧	orange	🟨	wheat
⬜	oldlace	⬜	floralwhite	🟨	darkgoldenrod	🟨	goldenrod
⬜	cornsilk	🟨	gold	⬜	lemonchiffon	🟨	khaki
🟨	palegoldenrod	🟨	darkkhaki	⬜	ivory	⬜	beige
⬜	lightyellow	⬜	lightgoldenrodyellow	🟨	olive	🟨	yellow
🟩	olivedrab	🟩	yellowgreen	🟩	darkolivegreen	🟩	greenyellow
🟩	chartreuse	🟩	lawngreen	🟩	sage	🟩	lightsage
🟩	darksage	⬜	honeydew	🟩	darkseagreen	🟩	palegreen
🟩	lightgreen	🟩	forestgreen	🟩	limegreen	🟩	darkgreen
🟩	green	🟩	lime	🟩	seagreen	🟩	mediumseagreen
🟩	springgreen	⬜	mintcream	🟩	mediumspringgreen	🟩	mediumaquamarine
🟩	aquamarine	🟦	turquoise	🟦	lightseagreen	🟦	mediumturquoise
⬜	azure	⬜	lightcyan	🟦	paleturquoise	⬛	darkslategray
⬛	darkslategrey	🟦	teal	🟦	darkcyan	🟦	aqua
🟦	cyan	🟦	darkturquoise	🟦	cadetblue	🟦	powderblue
🟦	lightblue	🟦	deepskyblue	🟦	skyblue	🟦	lightskyblue
🟦	steelblue	⬜	aliceblue	🟦	dodgerblue	⬛	lightslategrey
⬛	lightslategray	⬛	slategray	⬛	slategrey	🟦	lightsteelblue
🟦	cornflowerblue	🟦	royalblue	⬜	ghostwhite	⬜	lavender
🟦	midnightblue	🟦	navy	🟦	darkblue	🟦	mediumblue
🟦	blue	🟪	slateblue	🟪	darkslateblue	🟪	mediumslateblue
🟪	mediumpurple	🟪	blueviolet	🟪	indigo	🟪	darkorchid
🟪	darkviolet	🟪	mediumorchid	🟪	thistle	🟪	plum
🟪	violet	🟪	purple	🟪	darkmagenta	🟪	fuchsia
🟪	magenta	🟪	orchid	🟪	mediumvioletred	🟥	deeppink
🟥	hotpink	⬜	lavenderblush	🟥	palevioletred	🟥	crimson
🟥	pink	🟥	lightpink				

About grayscale values:



III Accuracy of Integral NICS

Effects of range and interval on INICS index were investigated using (1-bromoethyl)benzene. As shown in follows, the default setting (Range: –10 to 10 Å, interval: 0.2 Å) provides well results.



	Range	**Interval** / Å	**INICS** / ppm
	–10 to 10 Å	0.1	–45.182
(Default)	–10 to 10 Å	0.2	–45.182
	–10 to 10 Å	0.5	–45.180
	–8 to 8 Å	0.2	–44.809
	–5 to 5 Å	0.2	–43.286
	–2 to 2 Å	0.2	–33.214

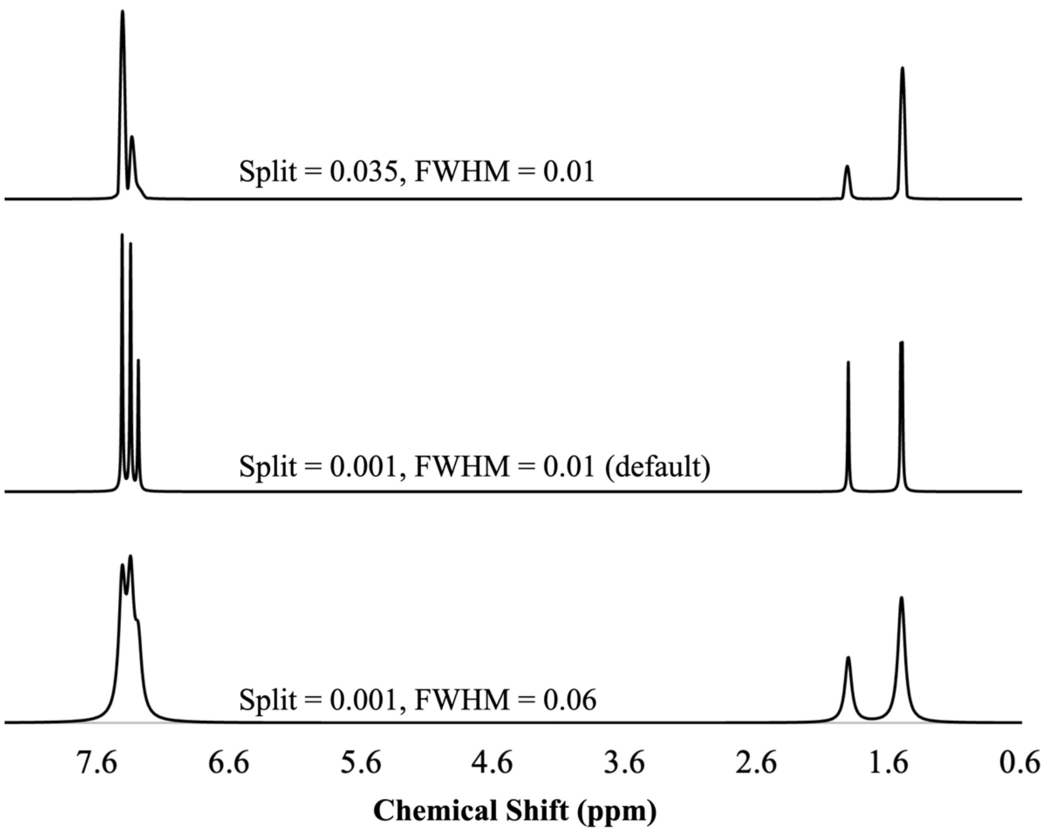
IV Lorentzian Line Broadening in NMR Spectrum

NMR spectrum follows Lorentzian function for line broadening:

$$L(x) = \frac{\mathrm{FWHM}}{2\pi} \frac{1}{(x - x_i)^2 + 0.25 \times \mathrm{FWHM}^2}$$

where, FWHM indicates "Full Width at Half Maximum" and x_i indicates the chemical shift of target peak.

Follows are examples of how "split" and "FWHM" affect the line shape in NMR spectrum.


Split = 0.035, FWHM = 0.01
Split = 0.001, FWHM = 0.01 (default)
Split = 0.001, FWHM = 0.06
Chemical Shift (ppm)