

---

# **gimic Documentation**

***Release 0.0***

**GIMIC developers**

**Jul 27, 2023**



---

## Contents

---

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Installation</b>	<b>3</b>
2.1	Parallelization . . . . .	4
2.2	Installation on Stallo supercomputer . . . . .	4
2.3	Using BLAS1 and BLAS2 routines . . . . .	4
2.4	Installation on a Mac . . . . .	4
<b>3</b>	<b>Testing</b>	<b>7</b>
3.1	Adding new tests . . . . .	7
<b>4</b>	<b>Important formulas</b>	<b>9</b>
4.1	GIMIC key equation . . . . .	9
4.2	Anisotropy of the induced current (ACID) . . . . .	9
<b>5</b>	<b>Development</b>	<b>11</b>
<b>6</b>	<b>Usage</b>	<b>13</b>
6.1	Interfaces to GIMIC . . . . .	13
<b>7</b>	<b>The GIMIC input file</b>	<b>19</b>
7.1	Keywords . . . . .	19
7.2	Grid section . . . . .	20
7.3	Advanced section . . . . .	21
7.4	Section: Essential . . . . .	21
7.5	Current-density calculations . . . . .	22
7.6	Integration of the current strength . . . . .	22
<b>8</b>	<b>Grids</b>	<b>23</b>
8.1	Basic grids . . . . .	23
8.2	Bond grids . . . . .	24
<b>9</b>	<b>Description of generated files</b>	<b>27</b>
<b>10</b>	<b>Interactive bash scripts</b>	<b>29</b>
10.1	Introduction . . . . .	29
10.2	Setup . . . . .	29
10.3	Quick summary . . . . .	30

10.4 Detailed description . . . . .	30
10.5 Some tips and advice . . . . .	33
<b>11 Vector plots</b>	<b>35</b>
<b>12 Streamlines</b>	<b>37</b>
<b>13 (Signed) modulus density plots</b>	<b>39</b>
<b>14 ACID plots</b>	<b>41</b>
<b>15 Interpretation of the results of GIMIC calculations</b>	<b>43</b>
15.1 3D integration . . . . .	43
15.2 Integration planes . . . . .	45

# CHAPTER 1

---

## Introduction

---

This is the GIMIC program for calculating magnetically induced currents in molecules. For this program produce any kind of useful information, you need to provide it with an AO density matrix and three (effective) magnetically perturbed AO density matrices in the proper format. Currently only recent versions of ACES2 (CFOUR), Turbomole, QChem, LSDalton, FERMION++, Gaussian can produce these matrices. Dalton is in the works. If you would like to add your favourite program to the list please use the source, Luke.

- For instructions how to compile and install this program refer to the INSTALL file in the top level directory.
- There is an annotated example input in the examples/ directory.
- For information on command line flags available run: `'gimic -help'`

The following features have been implemented in the program

- Current densities in 2D or 3D
- The (signed) modulus of the current
- The divergence of the current (this is useful for checking gauge invariance vs. gauge independence). Note, this module does not work at the moment.
- Vector representation of the current in 2D or 3D
- Integration of the current flow through defined cut-planes in molecules
- Open-shells and spin currents
- Parallel execution through MPI (optional). Note, this works only if you check out the “stable” branch.

Utility programs to extract the AO density and perturbed densities from ACES2, Turbomole, QChem, LSDALTON and FERMION++ calculations are included in the GIMIC source distribution, see tools directory. Turbomole 5.10 and newer also has the GIMIC interface built in.



## CHAPTER 2

---

### Installation

---

Before compiling GIMIC you need to make sure that you have installed ideally all of the packages collected in the `requirements.txt` file. You need minimum the ones listed below.

```
* cython
* numpy
* runtest == 2.3.2
```

A convenient way to install the packages listed in `requirements.txt` is to install the Anaconda2 (<https://www.anaconda.com/distribution/>) Python distribution first. Then you can simply install all the packages listed in the file `requirements.txt` by using:

```
$ conda install name-of-package
```

GIMIC requires CMake to configure and build. CMake is invoked via a front-end script called `setup`:

```
$ ./setup
$ cd build
$ make
$ make install
```

To see all available options, run:

```
$ ./setup --help
```

Branch “master”:: GIMIC requires CMake to configure and build.:

```
$ mkdir build
$ cd build
$ cmake ../
$ make
$ make install
```

Test the installation with:

```
$ cd build
$ make test
```

Note, some tests may require Valgrind and will fail if this debugger is not available. However, this is no need to worry if all other tests pass.

## 2.1 Parallelization

OpenMP parallelization is available:

```
$ ./setup --omp
```

MPI parallelization is in the works.

## 2.2 Installation on Stallo supercomputer

```
$ git clone git@github.com:qmccurrents/gimic.git
$ cd gimic
$ module load Python/2.7.12-foss-2016b
$ module load CMake/3.7.2-foss-2016b
$ virtualenv venv
$ source venv/bin/activate
$ pip install -r requirements.txt
$ ./setup
$ cd build
$ make
$ make install
```

## 2.3 Using BLAS1 and BLAS2 routines

With GNU compilers use:

```
$ ./setup --blas
```

With Intel compilers and MKL use:

```
$ ./setup --fc=ifort --cc=icc --cxx=icpc --cmake-options="-D ENABLE_MKL_FLAG=ON"
```

## 2.4 Installation on a Mac

The main problem with installing GIMIC on a Mac is setting the correct paths and force the program to find all libraries. Here we share some recommendations but can not guarantee that GIMIC will for sure work on your Mac. GIMIC has been developed on a Ubuntu/Linux operating system. Note, there are for sure more elegant ways to get GIMIC installed on a Mac. If you figure them out, please share.

- Install git and cmake.
- Install Anaconda2 then you should be able to install most of the recommended packages listed in “requirements.txt”.



- For installing the C++ (gcc) and Fortran (gfortran) compilers you can use Xcode, Brew or MacPorts. You just need to make sure that GIMIC is able to find them. A way to solve this is editing the “.bashrc” file.
- Check if you have the following paths in your “.bashrc”. If not add them.
  - export PATH=/Users/your\_username:/Users/your\_username/anaconda/bin:\$PATH
  - PATH="/Applications/CMake.app/Contents/bin":\$PATH
  - export PATH=/Users/your\_username/gimic/build/bin:\$PATH
  - export PATH="/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin"



The testing procedure of the code is simplified using the **runtest** library written by Radovan Bast.

Make sure that you have it installed for the appropriate Python or virtual environment:

```
$ pip install git+https://github.com/bast/runtest.git@master
```

When compilation has completed, call `make test` from the `build` directory.

### 3.1 Adding new tests

Adding new tests can be done in the `test` directory. The provided tests can serve as example. The definition of what are tested variables is done in the `test` executable. Remember to add the name of your new test in the `test/CMakeLists.txt` file. In case of doubts, read the [documentation](#) of `runtest`.



### 4.1 GIMIC key equation

The GIMIC working equation for the evaluation of the current density susceptibility tensor

$$\begin{aligned} \frac{\partial \chi^*_{\mu}(\mathbf{r})}{\partial B_{\tau}} &= \sum_{\nu} \left\{ \frac{\partial h}{\partial m^*K_{\gamma}} \chi^*_{\mu}(\mathbf{r}) \frac{\partial m^*K_{\gamma}}{\partial B_{\tau}} + \frac{\partial h}{\partial m^*K_{\gamma}} \chi^*_{\nu}(\mathbf{r}) \frac{\partial m^*K_{\gamma}}{\partial B_{\tau}} \right\} \\ &+ \sum_{\nu} \left\{ \frac{\partial h}{\partial m^*K_{\gamma}} \chi^*_{\mu}(\mathbf{r}) \frac{\partial m^*K_{\gamma}}{\partial B_{\tau}} - \epsilon_{\gamma\tau\delta} \left[ \sum_{\nu} \frac{\partial h}{\partial m^*K_{\gamma}} \chi^*_{\nu}(\mathbf{r}) \frac{\partial m^*K_{\gamma}}{\partial B_{\delta}} \right] \right\} \end{aligned}$$

See also:

1. J. Jusélius, D. Sundholm, J. Gauss, J. Chem. Phys., 121, 3952 (2004)
2. S. Taubert, D. Sundholm, J. Jusélius, J. Chem. Phys., 134, 054132 (2011)

### 4.2 Anisotropy of the induced current (ACID)

The formula for the ACID method

$$\begin{aligned} \Delta J^2(\mathbf{r}) &= \frac{1}{3} \left[ \left( J_x(\mathbf{r}) - J_y(\mathbf{r}) \right)^2 + \left( J_y(\mathbf{r}) - J_z(\mathbf{r}) \right)^2 + \left( J_z(\mathbf{r}) - J_x(\mathbf{r}) \right)^2 \right] \\ &+ \frac{1}{2} \left[ \left( J_x(\mathbf{r}) + J_y(\mathbf{r}) \right)^2 + \left( J_y(\mathbf{r}) + J_z(\mathbf{r}) \right)^2 + \left( J_z(\mathbf{r}) + J_x(\mathbf{r}) \right)^2 \right] \end{aligned}$$

See also:

1. R. Herges and D. Geuenich, J. Phys. Chem. A, 105, 3214 (2001)
2. H. Fliegl, J. Jusélius and D. Sundholm, J. Phys. Chem. A, 120, 5658, (2016)



## CHAPTER 5

---

### Development

---

The GIMIC code is based on Fortran and Python. Python is employed to parse the input files and call the Fortran executable (`gimic.bin` in the installation directory).

Remember to add tests using `runtest` for any new introduced features, as described in the Testing section of the documentation.

Adding new keywords to the GIMIC input file is done in the `src/gimic.in` file. The variable type is defined in the beginning of the file. The keyword can be made compulsory or optional.





Three files are needed in order to run GIMIC:

1. The XDENS file containing the effective one-particle density, and the magnetically perturbed densities in AO basis;
2. The MOL file with information on molecular geometry and basis sets;
3. The GIMIC input file. The default name is `gimic.inp`, however, any other file name is acceptable.

The XDENS and MOL files are obtained externally using other quantum chemistry programs which support an interface to gimic. Details are presented below.

Before doing the actual calculation with GIMIC, it might be a good idea to check if the grid is correctly defined. The `dryrun` flag produces the `grid.xyz` file which can be examined in any molecular viewer.

```
$ gimic --dryrun
```

The calculation with GIMIC is started by:

```
$ gimic [gimic.inp] > gimic.out
```

The square brackets mean that it is not necessary to give the name of the input file if it is called `gimic.inp` since it is recognised by default.

## 6.1 Interfaces to GIMIC

The following sections explain how to obtain the XDENS and the MOL files.

### 6.1.1 Running TURBOMOLE

As of TURBOMOLE 5.10, the GIMIC interface is part of the official distribution. To produce the necessary files to run GIMIC, you first need to optimize the wavefunction/density of the molecule, before running the `mpshift` program which produces the perturbed density matrices. Before you run `mpshift`, you need to edit the `control` file and

add the `$gimic` keyword. When the calculation has finished, TURBOMOLE writes two files called CAODENS (AO density data) and XCAODENS (perturbed density data). In case either of them is missing, the `mpshift` calculation has not converged. It is necessary to add the `$csmxiter N` keyword to the control file, where `N` is larger than the default value of 30. Afterwards, run the `turbo2gimic.py` script (distributed with GIMIC) in the same directory to produce the MOL and XDENS files.

```
$ turbo2gimic.py > MOL
```

### 6.1.2 Running LSDalton

The scripts/input needed can be found in `tools/lsdalton2gimic`.

Written by C. Kumar, University of Oslo, [chandan.kumar@kjemi.uio.no](mailto:chandan.kumar@kjemi.uio.no).

.GIMIC needs to be added in LSDALTON.INP.

Note that open-shell calculations are not supported.

Run in the same directory:

```
python lsdalton2gimic.py
```

### 6.1.3 Running GAUSSIAN

The nuclear shielding calculation on Gaussian needs to be performed by including the keyword `IOP(10/33=2)` in order to print the perturbed density matrices in the output file. Explicitly specifying `NMR=GIAO` is not necessary since using GIAO's is the default in G09. The keyword `Int=NoBasisTransform` is needed in order to prevent Gaussian from transforming the generalized contraction basis sets. It ensures that the results will match with the ones obtained in Turbomole.

The general structure of the input file looks like this:

```
%chk=file.chk
#P <method>/<basis> nmr pop=regular Int=NoBasisTransform IOP(10/33=2)
<empty line>
title
<empty line>
<charge> <multiplicity>
<coordinates>
<empty line>
```

In the above, the lines with the `< >` symbols are supposed to be modified. These bracket symbols are not part of the actual input file.

If the basis set needs to be specified explicitly, the input file is structured as follows:

```
%chk=file.chk
#P <method> nmr pop=regular Int=NoBasisTransform IOP(10/33=2)
<empty line>
title
<empty line>
<charge> <multiplicity>
<coordinates>
<empty line>
<basis set specification>
```

(continues on next page)

(continued from previous page)

```
<****>
<empty line>
```

If ECPs are needed, then the method specification line should like as in the example below.

```
#P TPSS/GenEcp nmr pop=regular Int=NoBasisTransform IOp(10/33=2)
```

It seems there is a small difference between the keywords `nosymmetry` and `Symmetry=None`. The latter should only be used if `nosymmetry` generates an error.

The scripts/input needed can be found in `/tools/g092gimic`.

This script has been provided by Vincent Liegeois from the University of Namur, [vincent.liegeois@unamur.be](mailto:vincent.liegeois@unamur.be).

The tool consists of two parts:

- 1) `Gaussian2gimic.py` which is the main script
- 2) `BasisSet.py` which is a module file containing the functions to read the basis set and to do the transformations from Spherical to Cartesian. This file just needs to be put in the same directory as `Gaussian2gimic.py`.

`Gaussian2gimic.py` uses `optionparser` to sets its different options. Therefore, `Gaussian2gimic.py -h` will give you the full description.

There are two options: `-i` and `-t`, however the latter is optional. The command which runs the script on a formatted checkpoint file from GAUSSIAN is the following:

```
Gaussian2gimic.py -i file.fchk
```

It will produce the XDENS and MOL files

The extra option `-t` accept the argument of the XDENS file, which matches the output of the interface to TURBOMOLE obtained with `turbo2gimic.py`.

```
Gaussian2gimic.py -i file.fchk -t XDENS_turbo
```

In addition to creating XDENS and MOL files, the script prints to the terminal a comparison with the values obtained from GAUSSIAN and from TURBOMOLE. This is intended as a way to make sure that both numbers are similar.

For example, for a calculation on paranitroaniline with HF/6-311G(2df,2pd), the maximum error on the density matrix is 5.8E-5 while the maximum errors on the perturbed density matrices are 2.1e-2, 2.6e-2, 5.0e-2.

For the same molecule but with HF/cc-pVTZ, the maximum errors are: 4.8e-5, 2.5e-2, 5.4e-2, 7.8e-2.

The MOL file produced by `Gaussian2gimic` is slightly different from the one obtained by `turbo2gimic`.

Indeed, `turbo2gimic.py` gives the basis set exactly as obtained on <https://bse.pnl.gov/bse/portal> with “optimized general contractions” checked. However, the coefficients are not normalized with “optimized general contractions” checked. They are normalized when “optimized general contractions” is unchecked.

For example, the first atomic orbital in the cc-pVTZ basis set for carbon with “optimized general contractions” checked consist of a contraction of 8 GTOs, while there are 10 GTOs in the contraction if “optimized general contractions” is unchecked. The coefficients given in the website and in `turbo2gimic.py` are optimized for the contraction of the 10 GTOs rather than 8. Gaussian and therefore `Gaussian2gimic.py` give the coefficients that are normalized for the contraction of the 8 GTOs.

Example input for benzene:

```
%Chk=benzeneg09.chk
%mem=2000mb

#p B3LYP/Def2TZVP SCF=Tight NMR=GIAO Int=NoBasisTransform IOp(10/33=2)

Benzene Gaussian NMR example

0 1
C 1.2049777911 0.6956942520 0.0000000000
C 1.2049777911 -0.6956942520 0.0000000000
C 0.0000000000 -1.3913885041 0.0000000000
C -1.2049777911 -0.6956942520 0.0000000000
C -1.2049777911 0.6956942520 0.0000000000
C 0.0000000000 1.3913885041 0.0000000000
H 2.1430161769 1.2372709666 0.0000000000
H 2.1430161769 -1.2372709666 0.0000000000
H 0.0000000000 -2.4745419332 0.0000000000
H -2.1430161769 -1.2372709666 0.0000000000
H -2.1430161769 1.2372709666 0.0000000000
H 0.0000000000 2.4745419332 0.0000000000
```

Running Gaussian creates a file `benzeneg09.chk`. You need to convert this `*.chk` file to a formatted `*.fchk` file.

```
$ formchk file.chk file.fchk
```

Then you can proceed as described above and generate the MOL and XDENS files with:

```
$ Gaussian2gimic.py --input=benzeneg09.fchk
```

Note that for open-shell cases you need to add “`gfprint pop=regular iop(10/33=2)`” and use the Gaussian `*.log` file instead of the `*.fchk` file.

Example input for the triplet dication of benzene:

```
%LindaWorkers=cib26-2
%NProcShared=20
%Chk=benzeneg09.chk
%mem=2000mb

#p POP=FULL GFPrint nosymmetry B3LYP/DEF2TZVP SCF=Tight NMR IOp(10/33=2)

Benzene Gaussian NMR example triplet dication

2 3
C 1.2049777911 0.6956942520 0.0000000000
C 1.2049777911 -0.6956942520 0.0000000000
C 0.0000000000 -1.3913885041 0.0000000000
C -1.2049777911 -0.6956942520 0.0000000000
C -1.2049777911 0.6956942520 0.0000000000
C 0.0000000000 1.3913885041 0.0000000000
H 2.1430161769 1.2372709666 0.0000000000
H 2.1430161769 -1.2372709666 0.0000000000
H 0.0000000000 -2.4745419332 0.0000000000
H -2.1430161769 -1.2372709666 0.0000000000
H -2.1430161769 1.2372709666 0.0000000000
H 0.0000000000 2.4745419332 0.0000000000
```

When the calculation completes, run in the terminal:

```
mv file.out > file.log
Gaussian2gimic.py --input=file.log
```

For the present example, a current strength susceptibility of 8.4 nA/T was calculated.

### 6.1.4 Running QChem and FERMION++

The scripts/input needed can be found in `/tools/qchem`.

Written by J. Kussmann, University of Munich, [jkupe@cup.uni-muenchen.de](mailto:jkupe@cup.uni-muenchen.de).

Convert Output of Q-Chem or FermiONS++ for GIMIC (TURBOMOLE format)

For a list of options, type:

```
qc2tm -h
```

which prints

```
USAGE: qc2tm -t <qchem or fermions> -qcout <output-file> -scr
      <scratch-directory> -s2c (opt.) -openshell (opt.)
```

### 6.1.5 Running CFOUR

Do a normal NMR calculation and then run the `xcpdens` program distributed with GIMIC to make the XDENS file. Then run the `MOL2mol.sh` script to produce the MOL file.

### 6.1.6 Running ACES2

Using ACES2, the special driver script `xgimic2.sh` must be used to run the NMR shielding calculation. Modify the script to suit your needs (and set the paths correctly). If the NMR calculation is done with symmetry, the MOL file must be converted to C1 symmetry using the script `MOL2mol.sh`, prior to running GIMIC.

Example ZMAT:

```
CO2
O    2.14516685791074    0.00000000000000    0.00000000000000
C    0.000000000000622    0.00000000000000    0.00000000000000
O   -2.14516685791393    0.00000000000000    0.00000000000000

*ACES2 (CALC=CCSD, BASIS=tzp, UNITS=BOHR
COORD=CARTESIAN
MEMORY=250000000
REFERENCE=RHF
SYMMETRY=ON
PROPERTY=NMR
MULTIPLICITY=1
CHARGE=0
SCF_MAXCYC=200, CC_MAXCYC=150, CC_EXPORDER=40
CC_CONV=10, SCF_CONV=10, LINEQ_CONV=10, CONV=10
LINEQ_EXPAN=30)
```

Run ACES2 via `xgimic2.sh` to produce the XDENS file:

```
$ xgimic2.sh --cc >aces2.out &
```

Convert the symmetry-adapted MOL file to C1 symmetry:

```
$ MOL2mol.sh
```

The new MOL file is now called mol.

---

## The GIMIC input file

---

The GIMIC input file is parsed by the `getkw` python parser, which defines a grammar system based on sections and keywords in a recursive manner. The input is in principle line oriented, but lines may be continued using a pipe symbol `|` at the end of a line. Furthermore, blanks and tabs are insignificant, with the exception of strings. Lines may be commented until end-of-line with a hash sign (`#`).

Sections are delimited by curly brackets `section{...}`, and may have a keyword argument enclosed in parentheses `section(argument){...}`.

Keywords come in two different types; simple keywords consisting of integers, reals or strings (enclosed in double quote marks `"string"`), and array keywords. Array keywords are enclosed in square brackets and the elements – integers, reals or strings – are delimited by a comma, e.g. `keyword = [0,1,2]`.

### 7.1 Keywords

The top level section defines a few global parameters:

**calc=cdens,integral** This keyword determines what is to be calculated, and in what order. The possible options are: 'cdens' – calculate current densities, 'integral' – integrate the current flow through a cut-plane. Each of these options have their own respective sections to specify options and grids.

**title** Useless keyword, but since every program with a bit of self respect has a title, GIMIC also has one...

**basis=MOL** Name of the MOL file (eg. MOL or mol or whatever). A relative path can also be given.

**density=XDENS** Name of the density file (eg. XDENS). A relative path can also be given.

**debug=1** Set debug level. The higher the number, the more useless output one gets.

**openshell=false** Open-shell calculation

**magnet=[Bx,By,Bz]** Define the direction of the external magnetic field by its vector components

**magnet\_axis=z** Specify the magnetic field along a defined axis. Valid options are: i,j,k or x,y,z or X. "i,j,k" are the directions of the basis vectors defining the integration plane. "x,y,z" are the absolute fixed laboratory axis. Note that `magnet\_axis=X` is used to specify the magnetic field along the direction which is orthogonal to the molecular plane, but parallel to the integration plane.

## 7.2 Grid section

### 7.2.1 Grid types

The currently implemented grids are

**even** The grid points are spaced uniformly

**base** Uniformly distributed grid points in 2D or 3D. Used in current-density calculations

**bond** Defining an integration plane passing through a particular chemical bond. Used for the integration of the current strength.

Custom grids can also be employed.

### 7.2.2 Definition

A grid can be defined in the following manner:

```
Grid(type) { ...  
}
```

### 7.2.3 Bond grid

The keywords specific to the **bond** type grid employed in the integration of the strength of the current density are:

**bond=[a,b]** Define the integration plane to cross the bond between atoms *a* and *b*. The indices are chosen according to the MOL file (`coord` file from Turbomole calculations). Alternatively, two Cartesian coordinates can be employed via the keywords `coord1` and `coord2`.

**coord1 = [x1,y1,z1]** and

**coord2 = [x2,y2,z2]** Specify a pair of Cartesian coordinates instead of atomic indices. The integration plane will cross the line between the two points. Alternatively, use atomic indices using the keyword `bond=[a,b]`.

**fixpoint=c** Specify an atomic index to use as the third point defining the integration plane. Alternatively, specify an arbitrary point using the keyword `fixcoord`.

**fixcoord=[x, y, z]** Specify an arbitrary point in 3D as the third point defining the integration plane. Alternatively, specify an atomic index using the keyword `fixpoint`.

**distance=r** Define the distance between the two atoms or two Cartesian coordinates between which the integration plane will cross.

**height=[-a, b]** Specify the distance *a* between the bottom vertex of the integration plane and the bond using the number *-a*. Specify the distance *b* between the top vertex of the integration plane and the bond using the number *b*.

**width=[-a,b]** Specify the lengths *a* and *b* on both sides of the chemical bond.

**type=gauss** Use Gauss distribution of grid points for the Gauss quadrature

**gauss\_order=9** The order of the Gauss quadrature



### 7.2.4 Base grid

The keywords for the **base** type grid employed in current-density calculations are:

**origin**=[x, y, z] This keyword specified the Cartesian coordinate of the bottom left corner of the plane or cube used during the current-density calculation.

**ivec**=[x, y, z] The direction of the vertical basis vector for the plane or cube.

**jvec**=[x, y, z] The direction of the horizontal basis vector for the plane or cube. The third vector specifying the cube is calculated as  $k = i \times j$ , therefore it is not given explicitly.

**length**=[a, b, c] The length of each side of the plane or cube.

### 7.2.5 Universal keywords

The following keywords are valid for both base and bond grids.

**spacing**=[a, b, c] The distance between the grid points in the three directions (the basis vectors i, j, and k). One should specify either `spacing` or `grid_points`.

**grid\_points**=[a, b, c] The specific number of grid points in each direction. One should specify either `spacing` or `grid_points`.

**rotation**=[a, b, c] The angles of rotation in space.

**rotation\_origin**=[x, y, z] The point in space around which to rotate. If not specified, the rotation is done at the middle of the bond.

## 7.3 Advanced section

The keywords are given in the section:

```
Advanced { ...
}
```

The available keywords are:

**lip\_order**=5 Polynomial order of the Lagrange Interpolation Polynomials. If a calculation has been preformed on a even spaced grid, generate a grid suitable for Gaussian integration by doing Lagrange interpolation

**spherical**=off Use spherical cartesians (i.e. 5d/7f/10g...). This is usually handled automagically. Experts only.

**diamag**=on Turn on/off diamagnetic contributions (experts only)

**paramag**=on Turn on/off paramagnetic contributions (experts only)

**GIAO**=on Turn on/off gauge including atomic orbitals (experts only)

**screening**=on Use screening to speed up calculations

**screen\_thrs**=1.d-8 Screening threshold

## 7.4 Section: Essential

The section contains some more specific keywords.

### 7.4.1 ACID calculations

ACID calculations can be defined by setting the calculation type to `calc=cdens` and the grid to type `base`:  
`Grid(base) {...}`

**acid=on** Turn on/off ACID calculation. It can only be done in current-density calculations with the `calc=cdens` keyword and the respective grid.

### 7.4.2 Modulus of the current density

Calculate the  $\text{mod}(\mathbf{J})$  integral, this is useful to verify that the actual integration grid is sensible in “tricky” molecules.

**jmod=off** Unless necessary, it should be turned off to save computational time.

## 7.5 Current-density calculations

Current density calculations are specified using the keyword `calc = cdens`.

ACID calculations can be performed in the current-density cal

**acid=on** Turn on/off ACID calculation

The produced files can be visualised in ParaView:

- `acid.vti`: contains information about ACID function as VTK file compatible with ParaView
- `jvec.vti`: contains information about the current density vector function as VTK file compatible with ParaView
- `jmod.vti`: contains the signed modulus of the current density vector function and can be visualized via two isosurfaces in ParaView

Calculate the  $\text{mod}(\mathbf{J})$  integral, this is useful to verify that the actual integration grid is sensible in “tricky” molecules.

## 7.6 Integration of the current strength

The calculation type has to be set to `calc=integral` and the grid to type `bond`. `Grid(bond) {...}`

There are three possible grid topologies in gimic: 'std (or base)' which is 3-dimensional (although the extend in one direction may be zero), 'bond' which is 2-dimensional, and 'file' which is an arbitrary set of points read from a file. 'std' grids are especially useful for visualisations of an entire current vector field. 'bond' grid are used to integrate the current that flows through the grid.

For an 'std' or 'bond' grid one can specify the spacing (type) of points in each dimension. The choices are 'even' for an equidistant grid, 'gauss' for a Gauss grid and 'lobatto' for a Gauss-Lobatto grid. The recommended choices are 'even' for 'std' and 'gauss' for 'bond'. When a quadrature grid is specified the order of the quadrature must also be specified with the 'gauss\_order' keyword. The number of grid points in each direction is specified either explicitly using either of the array keywords 'grid\_points' or 'spacing'. If the chosen grid is not a simple even spaced grid, the actual number of grid points will be adjusted upwards to fit the requirements of the chosen quadrature.

The shape of the grid can also be modified by the 'radius' key, which specifies a cutoff radius. This can be useful for integration. Sometimes it's practical to be able to specify a grid relative to a well know starting point. The 'rotation' keyword specifies Euler angles for rotation according to the x->y->z convention. Note that the magnetic field is not rotated, unless it is specified with 'magnet\_axis=i,j or k'.

GIMIC automatically outputs a number of .xyz files containing dummy points to show how the grids defined actually are laid out in space.

## 8.1 Basic grids

The 'std' grid is defined by giving an 'origin' and two orthogonal basis vectors 'ivec' and 'jvec' which define a plane. The third axis is determined from  $\vec{k} = \vec{i} \times \vec{j}$ . The array 'lengths' specifies the grid dimensions in each direction.

### 8.1.1 grid(std)

```
type=even
origin=-8.0, -8.0, 0.0
    Origin of grid
```

(continues on next page)

(continued from previous page)

```

ivec=1.0, 0.0, 0.0
    Basis vector i

jvec=0.0, 1.0, 0.0
    Basis vector j ( k = i x j )

lengths=16.0, 16.0, 0.0
    Lenthths of (i,j,k)

spacing=0.5, 0.5, 0.5
    Spacing of points on grid (i,j,k)

grid\_points=50, 50, 0
    Number of gridpoints on grid (i,j,k)

rotation=0.0, 0.0, 0.0
    Rotation of (i,j,k) -> (i',j',k') in degrees. Given as Euler angles
    in the x->y->z convention.

```

## 8.2 Bond grids

The 'bond' type grids define a plane through a bond, or any other defined vector. The plane is orthogonal to the vector defining the bond. The bond can be specified either by giving two atom indices, 'bond=[1,2]', or by specifying a pair of coordinates, 'coord1' and 'coord2'. The position of the grid between two atoms is determined by the 'distance' key, which specifies the distance from atom 1 towards atom 2. For analysing dia- and paramagnetic contributions, the positive direction of the bond is taken to be from atom 1 towards atom 2. Since one vector is not enough to uniquely defining the coordinate system (rotations around the bond are arbitrary), a fixpoint must be specified using either the 'fixpoint' atom index or the 'fixcoord' keyword. This triple of coordinates is also used to fix the direction of the magnetic field when the 'magnet\_axis=T' is used.

The shape and size of the bond grid is specified by the keywords 'width' and 'height'. Each of the four components is relative to a point on the line connecting the two reference atoms/coordinates. Typically (but not necessarily) the first component of both ranges is negative and the second component is positive.

```

width=[-1.5, 5.0]
height=[-5.0, 5.0]

```

### 8.2.1 grid(bond):

**type=gausslobatto** Use uneven distribution of grid points for quadrature

**bond=1,2** Atom indices for bond specification

**fixpoint=5** Atom index to use for fixing the magnetic field and grid orientation

**coord1=0.0, 0.0, 3.14** Coordinate of atom 1

**coord2=0.0, 0.0, -3.14** Coordinate of atom 2

**fixcoord=0.0, 0.0, 0.0** Fixation coordinate

**distance=1.5** Place grid 'distance' between atoms 1 and atom 2

**gauss\_order=9** Order for Gauss quadrature

**spacing=0.5, 0.5, 0.5** Spacing of points on grid (i,j,k) (approximate)

**grid\_points=50, 50, 0** Number of grid points on grid (i,j,k) (approximate)

**height=-4.0, 4.0** Grid size relative to grid center

**width=-1.0, 6.0** Grid size relative to grid center

**radius=3.0** Create a round grid by cutting off at radius

**rotation=0.0, 0.0, 0.0** Rotation of (i,j,k)  $\rightarrow$  (i',j',k') in degrees. Given as Euler angles in the x $\rightarrow$ y $\rightarrow$ z convention.



---

## Description of generated files

---

Note “file.txt” means text file format and “file.vti” stands for VTK file format, which is compatible with ParaView. The text files start with a given grid point (x,y,z) followed by either information about the modulus of the current density  $f(x,y,z)$  or the components of the current density vector  $J_x, J_y, J_z$ .

- MOL: contains molecular coordinates and basis functions, needed as input file for a gimic calculation
- XDENS: contains information about the AO density and first order perturbed AO density, needed as input for a gimic calculation
- `gimic.inp`: gimic input file, here either the cubic grid or a grid across a bond is specified, needed as input information for a gimic calculation
- `gimic.out`: gimic output file, contains information about the integrated current strength susceptibility
- `grid.xyz`: contains molecular coordinates in Ångström and cube points (8) or integration plane corner points (4)
- `mol.xyz`: contains molecular coordinates in Ångström

closed-shell calculation:

- `acid.vti`: contains information about ACID function
- `jvec.txt`: contains information about a grid point x,y,z and the current density vector in that point  $j_x, j_y, j_z$  (x,y,z,  $f_{\text{current}}(x,y,z)$ , the point is given in bohr (it used to be in Ångström)
- `jvec.vti`: contains information about the current density vector function
- `jmod.txt`: contains a grid point and the modulus of the current density vector function, x,y,z,  $f_{\text{mod}}(x,y,z)$
- `jmod.vti`: contains the signed modulus of the current density vector function

only for open-shell calculations:

- `jmodalpha.txt`: contains the signed modulus of the alpha current density vector function
- `jmodalpha.vti`: contains the signed modulus of the alpha current density vector function
- `jmodbeta.txt`: contains the signed modulus of the beta current density vector function
- `jmodbeta.vti`: contains the signed modulus of the beta current density vector function

- `jmodspindens.txt`: contains the signed modulus of the difference (alpha - beta) current density vector function
- `jmodspindens.vti`: contains the signed modulus of the difference (alpha - beta) current density vector function
- `jvecalpha.txt`: contains information about the alpha current density vector function
- `jvecalpha.vti`: contains information about the alpha current density vector function
- `jvecbeta.txt`: contains information about the beta current density vector function
- `jvecbeta.vti`: contains information about the beta current density vector function
- `jvecspindens.txt`: contains information about the difference (alpha - beta) current density vector function
- `jvecspindens.vti`: contains information about the difference (alpha - beta) current density vector function



---

## Interactive bash scripts

---

### 10.1 Introduction

GIMIC calculations can be started interactively using bash scripts. They are useful for adjusting the integration plane of choice, for starting current profile analysis, and submission of parallel tasks on a cluster with the SLURM scheduling system (i.e. one where you submit jobs using `sbatch`). There is a local version for the current profile analysis, which can run on a laptop, for example. Note that at the moment they are working with the `coord` file from TURBOMOLE. It will not be hard to make the input universal by using an XYZ file instead, so if you need it, let us know. The current density analyses employing GIMIC are most easily done if the magnetic field, pointing along the Z axis, is perpendicular to the molecular plane of a planar molecule. The orientation of the magnetic field can, of course, be modified accordingly.

**NOTE:** The script assumes that the `MOL` and `XDENS` files exist. They can be prepared using `turbo2gimic.py` or `Gaussian2gimic.py` once the nuclear shielding calculation has finished. Also the coordinates need to be prepared in XYZ format, too. This can be done with the Turbomole script for conversion of the `coord` file to XYZ format.

```
t2x coord > coord.xyz
```

### 10.2 Setup

In the `jobscripts` directory there is a script called `setup.sh`. It takes care of preparing the scripts for the specific machine.

**NOTE:** When preparing the scripts for a cluster, first it is necessary to modify the `jobscript-header` file before running `setup.sh`. You likely need to adjust the name of the partition, number of tasks per node according to the specific cluster, as well as load the necessary modules in order to start GIMIC. Do not add time limit, job task name, or number of cores as parameters because the GIMIC script will ask for that before submitting the calculation.

Remember, that the `setup` script hardcodes the paths to the directories, so if you want to use the scripts on another computer, you need to execute `./setup` there. Setup needs to be done only once - after you `git pull`, so as to install the newly implemented features and bug fixes. The prepared scripts can then be called from the calculation

directory. should be called. The workflow can be made easier by add the export `$GIMIC_HOME=/your/own/path`, which is printed after setup. There are some bash aliases and functions in the file `useful-aliases`, which you may want to add to your `./bashrc`.

The scripts are still under development but the ones that are recommended for use at the moment are `current-profile-cluster.sh`, `current-profile-local.sh`, and `gimic-run.sh`. The others are still in development phase, so if you are curious, you can try but functionality might be unclear or limited.

VMD is a highly-recommended tools for visualisation of molecules, in particular, the integration planes.

## 10.3 Quick summary

1. Choose an integration plane
2. If not obvious where to start integrating from, you can test its beginning and end points using the `gimic-run.sh` script and visualise the `grid.xyz` file. Adjust the `gimic.inp` file manually if necessary.
3. Run the current profile for the integration plane (`current-profile-local.sh` if running on your laptop, or `current-profile-cluster.sh` if submitting as batch job on a cluster).
4. Inspect the profile plot, check the sign of the currents, reverse the signs if necessary.
5. Make a better-quality plot using the `plot-current-profile.sh` script.

## 10.4 Detailed descriptipon

In the `jobscripts` directory you can find several bash scripts. They should be called as:

### EXAMPLE:

```
$ $GIMIC_HOME/jobscripts/current-profile-cluster.sh

# OR if using the suggested aliases:

$ gcurrent
```

The input for the scripts is based on selecting two atoms between which the integration plane lies. By default the integration plane is perpendicular to the bond between them. When you start the script, it asks for indices of the selected atoms according to the `coord` file. The atomic indices should be given in counter-clockwise manner. We have a sign problem, so for now we have to live with the fact that sometimes the diatropic current and the paratropic current can have the wrong sign but the magnitude is obtained correctly. Meaning that *it is not necessary to restart the calculation if the signs are inverted*. It can be post-processed. In planar molecules the choice of the two atoms is straight-forward but in non-planar ones, you need to choose the two atoms to be with very similar Z coordinates when placing the magnetic field in the Z direction. It is a known issue, which will hopefully be resolved in the future.

**TIP:** Check the indices of the atoms from the `xyz` file, for example with `xmakemol`.

**NOTE:** Atom indices start from 0 in VMD, whereas in GIMIC and `xmakemol`, they start from 1.

A directory with a name include the atomic indices of the bond is created, for example `current_profile_39.119`. This means that several current profiles can be analysed at the same time, each written to a directory of its own. If a directory with the same calculation already exists, it will ask for confirmation to overwrite it.

Next, you need to define the dimensions of the integration plane. It crosses the bond at the midpoint between the two attoms by default, however if needed, one can shift it to be closer to the one or the other atom. Usually we are interested in starting the integration from the centre of a ring to infinity, or maybe from the centre of the ring to another. Two parameters take care of that: the **start** value and the **out** value. These parameters can be defined either by manually

entering a distance in atomic units, or using the atomic indices as defined in the `coord` file. By default, the script offers integration to infinity, which for practical purposes means 10 bohr. When entering these atomic indices, please type them on the same row and then press enter, for example,

```
1 2 3 4 5 [ENTER]
```

**TIP:** The integration can be defined to start or end at the centre of any number of atoms, including single atoms.

The script will calculate the distance between the bond and the geometric centre of the selected atoms. It is recommended to add extra 0.5 bohr because the origin of the vortex lies in the geometrical centre of the ring only for symmetric molecules. Adding the additional distance from the bond allows the origin of the vortex to be distinguished easily on the current profile. Note that the **start** value can be negative, meaning that integration occurs on the other side of the midpoint of the bond. The height of the integration plane is usually defined to be 10 bohr above and below the molecular plane (the keywords for that are **up** and **down**).

The current profile analysis is based on making thin slices of the defined integration plane, and executing a separate GIMIC calculation for each slice. The default width of the slice is 0.02 bohr but it can be modified if necessary for some reason. The number of grid points for the Gaussian quadrature used for the numerical integration is defined using the **spacing** keywords for X, Y and Z directions. The default values have been tested and are recommended.

The magnetic field direction needs to be specified either as manually entered Cartesian coordinates, where the default is the Z direction (0; 0; -1), or using the program maximise-projection by Lukas Wirz. The program will be added to the GIMIC repository when it is finalized. More manipulation of the integration plane is done using the **fixed coordinate** and the **rotation** of the plane. The fixed coordinate is the third coordinate, which defines the integration plane. Its exact usage can be found in the source code at `src/fgimic/grid.f90`. The script calculates an estimate, however, it is not always satisfactory. Alternatively, three Cartesian coordinates for the fixed point can be entered. This parameter is among the most difficult concepts in GIMIC, so one needs to get a feeling for it.

After these parameters are specified, a summary will be printed again, and their values will also be written to the file `current_profile_#.#/calculation.dat`. A dry run is performed to check if there are enough grid points for the Gaussian quadrature. If there are at least 27x9x1, then the result will be reliable. If that part succeeds, the input files for each of the slices of the integration plane are created. After that the script asks if a visualisation of the integration plane should be done. Selecting this option calls calculations of the grid at the first and last slices and writes a `grid.check.xyz` file. The current profile script can be put to background using CTRL+Z and the grid file opened, for example, in VMD. A useful representation of the plane can be done using the procedure below, which has to be placed in the `~/vmdrc` file. It actually draws the integration planes for all the opened molecule in VMD.

```
proc intplanes {} {
    set loadedMolecules [molinfo list]

    foreach molid $loadedMolecules {

        mol showrep $molid 0 off
        mol modselect 0 $molid "all not element X Be"
        set xel [atomselect $molid "element X"]
        set coords [$xel get index]

        for {set i 0} {$i < 4} {incr i} {
            lassign $coords i1 i2 i3 i4
        }

        set c1 [atomselect $molid "index $i1"]
        set c2 [atomselect $molid "index $i2"]
        set c3 [atomselect $molid "index $i3"]
        set c4 [atomselect $molid "index $i4"]
    }
}
```

(continues on next page)

(continued from previous page)

```

lassign [$c1 get {x y z}] pos1
lassign [$c2 get {x y z}] pos2
lassign [$c3 get {x y z}] pos3
lassign [$c4 get {x y z}] pos4

draw color red
set LINEWIDTH 6
draw line $pos1 $pos2 width $LINEWIDTH
draw line $pos3 $pos4 width $LINEWIDTH
draw line $pos1 $pos3 width $LINEWIDTH
draw line $pos2 $pos4 width $LINEWIDTH

set posHalf1 [ vecscale 0.5 [ vecadd $pos1 $pos2 ] ]
set posHalf2 [ vecscale 0.5 [ vecadd $pos3 $pos4 ] ]

set LINEWIDTH 3
draw line $posHalf1 $posHalf2 width $LINEWIDTH
}

mol representation CPK 0.600000 0.300000 50.000000 50.000000
mol color Element
mol material Opaque
mol addrep $molid
mol modselect 1 $molid "all not element X Be"
}

```

The magnetic field vector can be drawn using the following VMD procedure:

```

proc arrow {start ending scaling} {
  set end [ vecscale $scaling $ending]
  set middle [vecadd $start [vecscale 0.8 [vecsub $end $start]]]
  draw cylinder $start $middle radius [ expr 0.08*$scaling]
  draw cone $middle $end radius [ expr 0.15*$scaling ]
}

```

After closing VMD, the current profile script should be brought back using `fg` and pressing enter again. On a cluster it will ask how many of the slices should be calculated in parallel, and what is the batch job limit. With that done, the `sbatch` command will be executed and one needs to wait for it to finish. When the job finishes, in the current profile directory there will be the `current_profile.dat` file. It lists four columns: the first one is the distance along the integration plane, and the next are net current, diatropic contribution and paratropic contribution respectively. These data are plotted as EPS files in the current profile directory.

**NOTE:** One should make sure that the net current far from the molecule is diatropic (positive by convention). If not, the sign should be reversed using `is wrong`. The following alias can be used:

```

alias revcurrent="mv current_profile.dat current_profile.dat.1 && awk '{printf \"%%.
↪ 6f\\t%.6f\\t%.6f\\t%.6f\\n\\", \\$1, -\\$2, -\\$4, -\\$3}' current_profile.dat.1 > current_
↪ profile.dat"

```

The current strength of different peaks on the current profile can be obtained using the function below. It takes two numbers as arguments - distance along the integration plane. It sums the values of the current in the slices between these distances and returns the net, diatropic and paratropic currents. The distances can be obtained from the output in the file `profile-points.out`. In the first column of the three sections there is distance along the integration plane, at which the diatropic, paratropic or net current vanish. One can estimate from the current profile plot which points are interesting and then take their actual values from the file. The data about the points can also be obtained by calling the script `crit_pts.sh` from the directory of the current profile.

```
function anprofile() { awk -v lower= -v upper= '{ if (( >= lower) && ( <= upper)) {
↪total+=; dia+=; para+=; } } END { printf(nTotal current: %fnDiatropic:
↪%fnParatropic: %fnn, total, dia, para); } ' current_profile.dat ; };

# used as:

$ anprofile 0.22 1.5

Net current: 0.066627
Diatropic: 0.134426
Paratropic: -0.067799
```

Finally, the current profile plot can then be done anew with `plot-current-profile.sh`. It is still under development and it might not be very user-friendly at the moment, so feel free to ask questions about it. Please let us know if you have any further questions, bugs and ambiguous parts.

## 10.5 Some tips and advice

**TIP:** A short way to preview the integration plane without doing the calculations are dry runs:

```
function dryrun() { gimic --dryrun "$@" > /dev/null ; xmakemol -f grid.xyz; };
```

Choosing integration planes can be tricky. One way to get a better feeling for the current densities in a molecule is to start with the 3D calculation of the current density and exploring it in Paraview. You can use the new `3D-run.sh` script. It is rather basic at the moment, and the input file needs to be inspected and the grid checked before starting the calculation. Its aim is to create a decently large box around the molecule. For molecules larger than 100 atoms it is reasonable to use spacing of 1 bohr, otherwise the calculation takes too long. In small systems 0.5 bohr is a good choice; less than that might be an overkill, unless one needs close-up views of currents. The calculation can only run in serial. Once the calculation starts, GIMIC gives a good estimate of how long it will take. If it is unreasonable, size of the box or the spacing should be adjusted. The 3D calculation will give the `jvec.vti` file. It also prints the `mol.xyz` file. Paraview cannot handle XYZ files at the moment, so they need to be converted to CML format first. The bash function below can be used. It employs openbabel. In case openbabel is missing, the `mol.xyz` file can be saved as `mol.cml` in Avogadro and the first line of the function commented out.

```
function molecule() {
babel -ixyz mol.xyz -ocml mol.cml
awk '{ {FS="\n"; {OFS="\n";
    if ($1 ~ "<atom id") {
        if ($5 ~ "spinMultiplicity")
            { print $1, $2, $3, $4, $5, $6, $7, $8/0.526, $9, $10/0.526, $11, $12/0.
↪526, $13 }
        else { print $1, $2, $3, $4, $5, $6/0.526, $7, $8/0.526, $9, $10/0.526, $11,
↪}
    }
    else print $0; }' mol.cml > mol-bohr.cml

# It takes an XYZ file as an argument:

$ molecule mol.xyz
```

The provided Paraview state file `3D-LIC.pvsm`, it will ask about the location of the `jvec.vti` and `mol-bohr.cml` files. After they are loaded, it should present the line integral convolution (LIC) representation. In the *Slice* filter changing the z component of the origin permits exploring the current densities vertically. The number of arrows illustrating the current direction is adjusted from the *Glyph* filter. In the *Masking* group the selected Glyph Mode is

*Every Nth Point.* Change the stride according to your preference. The length of the arrows is adjusted from the Scaling group, the *Scaling Factor* value.

The 3D visual inspection help identifying which current vortices are interesting and where the integration plane would cross as few other vortices as possible.

---

## Vector plots

---

The information needed for vector plots is in the file “jvec.{vti,vtu}” and can be visualised using ParaView (<https://www.paraview.org/>).

For the visualisation of the current vector field we use two different file formats. ‘vti’/ImageData for grids that consist of orthogonal basis vectors in two or three dimensions and evenly spaced grid points, and ‘vtu’/UnstructuredGrid for arbitrary sets of points. vti/vtu files consist of nodes (grid points) and cells. In vti files the node positions are defined by a range and spacing in each dimension and the (cuboid) cells are defined implicitly. In vtu files, the grid point coordinates are given explicitly and each cell is defined by a cell type (10 for tetrahedral) and the indices of the defining nodes. For each node we write down a vector (the induced current) while the cells are not assigned a value for typical visualisations (ie, the value can be set to 0.0). However, cells need to be defined—paraview interpolates the values given at grid points within each cell.

In case of vti files, all required information is given by the grid and no further input is required.

In case of vtu files we require an external program to create a tetrahedral mesh. One choice for such a program is TetGen (<http://wias-berlin.de/software/index.jsp?id=TetGen&lang=1>). The workflow is:

1. obtain a grid of your choice. The file format should be: no header entries, each line contains xyz of a grid point. That is, the number of lines equals the number of grid points. A grid can for example be obtained from the numgrid library (but those grids will typically be more expensive than necessary for visualisation purposes), or by writing a short script to generate a custom grid for a class of molecules. A universally useful grid can be generated with the provided script ‘unstructured\_grid\_gen.py’.
2. run ‘grd2node.sh <grid file>’ to obtain ‘grid.node’ which can be read by tetgen.
3. run ‘tetgen grid.node’ to obtain ‘grid.1.ele’. This file contains one quadruple of indices for each tetrahedral cell.
4. run your gimic job to obtain ‘jvec.vtu’. The generated ‘grid.1.ele’ is read automatically.

(What we cannot do yet: it might for some systems be useful to visualise properties not in a volume but on a surface, as for example the surface of a fullerene. This can also be done using vtu-files, but one needs to write a new output routine. For that it will be required to find the cell type of a triangle, and find the ‘tetgen’ command for 2D Delaunay triangulations. ETA: one day.)

Check also the section on the interpretation of GIMIC results and the Youtube examples.





## CHAPTER 12

---

### Streamlines

---

Follow the instructions under `/examples/benzene/ParaView-README`

File needed: “jvec.vti” Note, you need to generate your own molecule file, eg. in cml format, which is readable in ParaView. You can use for example Avogadro (<https://avogadro.cc/>) and read in mol.xyz and export it to mol.cml. Then you need to convert the mol.cml file to bohr.

Note that animations are also possible using ParaView.



## CHAPTER 13

---

(Signed) modulus density plots

---

This is possible with ParaView using the file “jmod.vti”.



## CHAPTER 14

---

### ACID plots

---

This is possible with ParaView using the file “acid.vti”.



---

## Interpretation of the results of GIMIC calculations

---

For the thorough interpretation of the induced current densities in a molecule it is recommended to do both visual and quantitative investigation. It is recommended to get a feeling of the molecule by running 3D calculations first. After that, one can choose suitable locations for the integration planes that will quantify the observed vortices.

It is important to keep that when placing an integration plane, we can only determine whether the current flow to the left or to the right at each grid point. This is caused by the fact that tropicity is a non-local property - i.e. one needs to follow streamlines obtained with path integration in order to determine whether the current flow is clockwise (diatropic) or counterclockwise (paratropic) with respect to the magnetic field vector. When integrating through a whole benzene molecule, there are two symmetric pairs of positive and negative contributions exactly because the current flow on one side of the vortex origin is going to the left with respect to the integration plane and in then as it returns, it looks as if it is flowing to the right. One needs topology analysis, which is not part of our code. We tackle the issue by applying some intuition. We know that inside every molecular ring there should be paratropic current, which we call negative simply out of convention. Then we know that around every molecule (aromatic, non-aromatic, or antiaromatic) there is a diatropic current, which we call positive.

We have a Youtube channel showing some of the basics: [https://www.youtube.com/channel/UC\\_5dN4qhYT0tr-itnsjoE0w/videos](https://www.youtube.com/channel/UC_5dN4qhYT0tr-itnsjoE0w/videos).

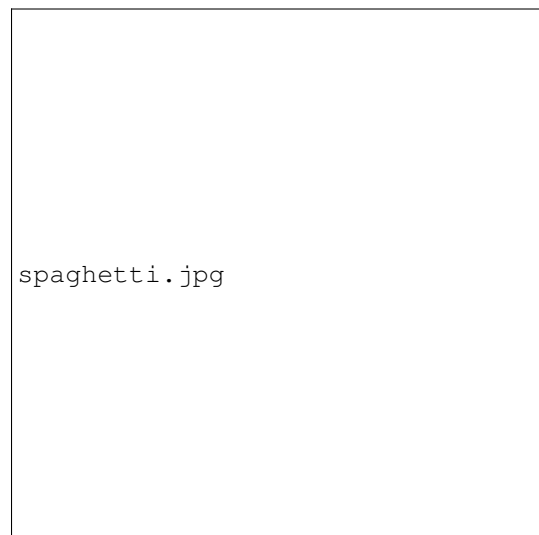
### 15.1 3D integration

A current density calculation can be performed using the `cdens` set of keywords in the input file. It is possible to use the interactive script called `3D-run.sh` to define a box around the molecule and specify the distance between the grid points. We recommend to use Paraview to visualise the results (<https://www.paraview.org/>). Our recommended setup for Paraview can be seen in this video (<https://www.youtube.com/watch?v=NngG1g3Bb7Q&t=5s>).

The `jvec.vti` files is what concerns us the most. Also, it is necessary to convert the molecular coordinates to CML format and bohr units. This is easily done using the bash function `molecule` given in the description of the section on the interactive scripts.

It is possible to do streamline plots using the StreamTracer tool. Slices on a plane can be done using the Surface LIC (line integral convolution) plug-in to show the direction of the current flow.

### 15.1.1 Streamline (spaghetti) plots



In this video ([https://www.youtube.com/watch?v=Pjy\\_0jucmGQ&t=28s](https://www.youtube.com/watch?v=Pjy_0jucmGQ&t=28s)) we present in short how to create the streamline “spaghetti” plots using Paraview. A pre-requisite is to have performed a current density calculations and that the `jvec.vti` and `mol-bohr.cml` files exist. The latter is not directly written by GIMIC but in the documentation we have suggested a bash function which creates it.

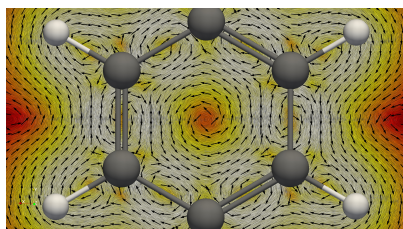
After the two files are loaded on the blank board we see the molecule and a rectangular object which shows the boundaries of the defined integration grid for the calculation. Then we apply the Stream Tracer tool. What it does is to define some points in space and calculates their trajectory as if they were particles moving in the vector field. It simply does path integration along the direction defined by the vector field at each point. By default it uses a line (High Resolution Line Source) as a seed type. It has to be changed to Point Source from the drop-down menu. The next step is to change the radius of the sphere. Usually a small sphere works better to distinguish different current loops. The length of the field line can be defined to match the size of the molecule. An important parameter is the Integration Direction. Forward or backward directions can be used to visually determine the tropicity of the current flow. One needs to check the magnetic field direction defined in the calculation and then look at the molecule along that axis. For example when the field points to the negative Z direction, then one needs to rotate the molecule, so that the view is towards the negative Z axis. Setting forward integration direction produces the correct tropicity direction according to the convention. It is accepted that diatropic current flow clockwise, while paratropic currents - counterclockwise. In any molecule on the outside, far from the nuclei, there is diatropic flow with strength determined by the magnetic response. Also, in the vicinity of the atomic nuclei there also are diatropic currents flowing clockwise. Inside molecular rings there is a paratropic counterclockwise current pathway, irrespective of whether the ring is aromatic or antiaromatic.

It appears that our job is already finished here, however we have not considered yet the current strength. Many of the curves turn out to be very weak and negligible. The lines can be made thicker by applying the Tube filter and a suitable colour scheme can be set with a suitable range of values. We typically use the Blackbody radiation scheme with range  $[1e-6; 0.1]$ , however based on the particular investigated molecule the range may need to be adjusted. With that done one can freely explore the streamlines in the molecule. It is sometimes useful to make a very small sphere but with many points. When the result looks good the view can be saved as PNG, for example. It is recommended to export high-resolution images with high quality compression in case of JPG format.

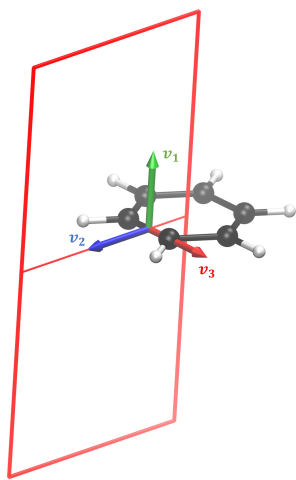
### 15.1.2 Line integral convolution on a surface

(coming)





## 15.2 Integration planes



### 15.2.1 Current profile integration

In this video we present how to interactively calculate a current profile across a bond in GIMIC. We start off in a directory where the nuclear shielding calculation for the benzene molecule has been executed with Turbomole.

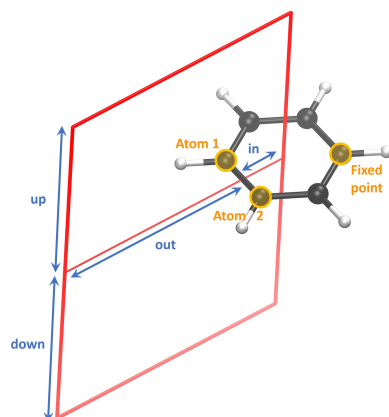
The molecular coordinates can be previewed by converting the TURBOMOLE coord file to XYZ format using the t2x program. In this example we show the coordinates in the basic XMAKEMOL program. It is lightweight and works over SSH smoothly. XMAKEMOL loads a molecule showing hydrogen bonds as dashed lines. They can be turned off by pressing the H key on the keyboard. The atomic indices can be shown by pressing the N key. The latter is needed for the definition of the integration plane. Zooming in closer can be done by pressing CTRL + P. The first slider changes the zoom level. The “Toggle depth” option switches between perspective and orthographic projection of the atoms. XMAKEMOL can be downloaded from <http://www.nongnu.org/xmakemol>. Please note that we are not affiliated or contributing to the development of XMAKEMOL in any way. Of course, one can use any other molecular visualisation program, such as Molden or VMD.

When we have inspected the molecular geometry and selected the bond through which to place an integration plane, it is time to prepare the gimic input files. The files CAODENS and XCAODENS contain the unperturbed and the magnetically perturbed density matrices. If they are missing, then the nuclear shielding calculation did not complete. They are converted to the file XDENS using the Python script `turbo2gimic.py`. It is handy to define it as a bash alias. Please pipe the output to the file called MOL.

With that done, the current profile script can be called. In this example we show the version for a cluster, however, it can be used in the same manner on a local machine. It is good to remember that GIMIC works in atomic units. We have decided to integrate the current in the benzene molecule. This can be done by placing an integration plane across any of the C = C bonds. The bond is defined using the indices of the atoms according to the coord file. XMAKEMOL makes the choice easy by displaying the indices. NOTE to VMD users! The first atom in the XYZ file has the index

one. VMD lists the indices starting from 0, so the given index needs to be increased by one. In this example we chose to integrate across the bond between atoms 4 and 5. According to our convention the atoms are entered in clockwise order, i.e. 4 – 5 instead of 5 – 4 in this case. The script asks for a possible suffix for the directory name. This can be generally omitted, however sometimes it might be necessary to run calculations across the same bond but along different integration planes. In practice this means that if the suffix “modified” is entered, then the created calculation directory is `current_profile_4.5_modified`.

Next, we need to define at which point the bond and the integration plane cross. Unless there is a particular reason, we set the plane to cross the midpoint of the bond. Possible reasons to shift it would be, for example, additional current vortices, which might interfere with the estimation of the currents.



Then the integration plane has to be positioned in space. Integration always occurs from right to left, so keep that in mind when defining the in and out distances. The in value can be defined either by specifying atomic indices or by choosing a certain value away from the bond. In the presented case we chose to integrate from the centre of the ring to infinity. Clearly, in this case it is easier to define the starting point through the indices of three atoms. This is done by entering the indices in arbitrary order but on the same line. The geometrical centroid of the hexagon is printed. For symmetric molecules, such as benzene, this is a good starting point. Otherwise, it is recommended to increase the distance a bit since the current vortex is not symmetric, either. In this example we increase the value by about 0.15 bohr. We want the end point to be infinitely far away from the molecule, where the current vanished. Practice has shown that usually at 8 bohr away from the bond, the current reaches 0. Similarly, we integrate infinitely far above and below the molecule by setting the up and down values to 8 bohr.

The next steps is to specify the thickness of the slices. We recommend to use 0.02 bohr to produce a nice high-resolution plot. Each slice is executed as a separate calculation. Grid point spacing refers to the grid for each slice. The pre-defined values have proven to yield good results.

The magnetic field direction can be specified in several different ways. The strongest currents are induced when the field is perpendicular to the molecular ring. Traditionally, molecules are placed to lie in the xy plane, thus the magnetic field vector  $B = (0; 0; -1)$ . In some cases it is necessary to rotate the field with respect to the integration plane. For example, bond currents can be calculated by setting  $B = (-1; 0; 0)$  in the chosen example, so that the field is perpendicular to the bond but lies in the plane of the benzene molecule. However, this definition of  $B \parallel X$ ,  $Y$  or  $Z$  can be overridden. If the molecule was not placed in the xy plane, and particularly in non-planar molecules, it is best to define the atoms with respect to which the field will be perpendicular. The tools `maximize_projection` and `plane` in the GIMIC repository need to be available for the automatic calculation of the field vector.

Afterwards, it is necessary to define the “fixed point”. It is the third point which specifies the integration plane. The script attempts to calculate it, however, it sometimes fails. In those cases, one needs to manually try and choose an atom which is located clockwise with respect to the bond atoms (4 and 5 in this example). Having done that, one can further rotate the integration plane if needed. It can be done by entering an angle in degrees. By default, the origin of rotation is the bond midpoint, however, the origin can be modified if needed.

The script prints a summary of the defined integration plane and performs a dry run for the first slice of the integration plane.

(to be continued, and the video has to be finished)