isoenum Documentation

Release 0.4.1

Andrey Smelter, Hunter N.B. Moseley

Aug 14, 2019

Contents:

CHAPTER 1

isoenum

Isotopic (`iso`) enumerator (`enum`) - enumerates isotopically resolved InChI (International Chemical Identifier) for metabolites.

The `isoenum` Python package provides command-line interface that allows you to enumerate the possible isotopically-resolved `InChI` from one of the Chemical Table file (`CTfile`) formats (i.e. `molfile`, `SDfile`) used to describe chemical molecules and reactions as well as from `InChI` itself.

See Tutorial documentation for usage examples of `isoenum` Python package as well as `isoenum` docker container.

1.1 Links

- isoenum @ GitHub
- isoenum @ PyPI

- isoenum @ DockerHub
- isoenum @ ReadTheDocs

1.2 Installation

The `isoenum` package runs under Python 2.7 and Python 3.4+. Use pip to install.

1.2.1 Install on Linux, Mac OS X

```
python3 -m pip install isoenum
```

1.2.2 Install on Windows

```
py -3 -m pip install isoenum
```

1.2.3 Dependencies

The `isoenum` Python package requires a **non-pip-installable** dependency: the Open Babel chemistry library version 2.3.90 or later, which relies on `InChI` library version 1.0.4 or later to perform `InChI` conversions.

Refer to the official documentation to install Open Babel on your system:

- Official Installation Instructions: http://openbabel.org/wiki/Category:Installation

1.3 Docker

In addition to PyPI package, `Dockerfile` and the automatically built DockerHub container, which contains the `isoenum` Python package and all its dependencies, are also provided.

To use the `isoenum` Python package, you will need to setup docker for your system and `pull` or `build` the docker container.

1.3.1 Install docker

Install `docker`:

- Follow the instructions to install docker on your system: https://docs.docker.com/engine/installation
  - Ubuntu
  - Debian
  - CentOS
  - Fedora
  - Mac
  - Windows

1.3.2 Setup container

Setup the `isoenum` container:

- `pull` the built image from the DockerHub:

```
# docker pull moseleybioinformaticslab/isoenum
# docker tag moseleybioinformaticslab/isoenum:latest isoenum:latest  # retag␣
↪docker image
# docker rmi moseleybioinformaticslab/isoenum  # remove after you have␣
↪retagged it
```

- or `build` an image using `Dockerfile` at the root of this repo by running `docker build` from the directory containing `Dockerfile`:

```
# docker build -t isoenum .
```

1.4 Development version installation

1.4.1 Install development version on Linux, Mac OS X

```
python3 -m pip install git+git://github.com/MoseleyBioinformaticsLab/isoenum.git
```

1.4.2 Install development version on Windows

```
py -3 -m pip install git+git://github.com/MoseleyBioinformaticsLab/isoenum.git
```

1.5 License

This package is distributed under the BSD *license*.

CHAPTER 2

Documentation index:

2.1 The isoenum Tutorial

Note: Read *The isoenum Docker Tutorial* to see examples provided below but with the use of a docker container with the `isoenum` Python package and all its dependencies instead of using the `isoenum` Python package directly.

2.1.1 Command-line interface

The `isoenum` package provides an easy-to-use command-line interface that allows the specification of isotopes for the creation of isotopically-resolved `InChI`.

To display all available options, run:

```
$ python3 -m isoenum --help
```

Output:

```
Isotopic enumerator (isoenum) command-line interface

Usage:
    isoenum -h | --help
    isoenum --version
    isoenum name (<path-to-ctfile-file-or-inchi-file-or-inchi-string>)
                 [--specific=<isotope:element:position>...]
                 [--all=<isotope:element>...]
                 [--enumerate=<isotope:element:min:max>...]
                 [--complete | --partial]
                 [--ignore-iso]
                 [--format=<format>]
                 [--output=<path>]
                 [--verbose]
```

(continues on next page)

(continued from previous page)

```
    isoenum ionize (<path-to-ctfile-file-or-inchi-file-or-inchi-string>)
                (--state=<element:position:charge>)
                [--format=<format>]
                [--output=<path>]
    isoenum nmr (<path-to-ctfile-file-or-inchi-file-or-inchi-string>)
                [--type=<experiment-type>]
                [--jcoupling=<name>...]
                [--decoupled=<element>...]
                [--format=<format>]
                [--output=<path>]
                [--subset]
                [--verbose]

Options:
    -h, --help                                      Show this screen.
    --verbose                                       Print more information.
    -v, --version                                   Show version.
    -a, --all=<isotope:element>                     Specify element and isotope, e.g. -a␣
↪13:C or --all=13:C
    -s, --specific=<isotope:element:position>       Specify element, isotope and specific␣
↪position,
                                                    e.g. -s 13:C:1 or --specific=13:C:1.
    -e, --enumerate=<isotope:element:min:max>       Enumerate all isotopically-resolved␣
↪CTfile or InChI,
                                                    e.g. -e 13:C:2:4 or --
↪enumerate=13:C:2:4
    -c, --complete                                  Use complete labeling schema, i.e.␣
↪every atom must specify
                                                    "ISO" property, partial labeling␣
↪schema will be used otherwise
                                                    for specified labeling information␣
↪only.
    -p, --partial                                   Use partial labeling schema, i.e.␣
↪generate labeling schema
                                                    from the provided labeling information.
    -i, --ignore-iso                                Ignore existing "ISO" specification in␣
↪the CTfile or InChI.
    -f, --format=<format>                           Format of output: inchi, mol, sdf, csv,
↪ json [default: inchi].
    -o, --output=<path>                             Path to output file.
    -t, --type=<experiment-type>                    Type of NMR experiment [default: 1D1H].
    -j, --jcoupling=<type>                          Allowed J couplings.
    -d, --decoupled=<element>                       Turn off J coupling for a given␣
↪element.
    -z, --state=<element:position:charge>           Create ionized form of InChI from␣
↪neutral molecule,
                                                    e.g. N:6:+1, O:8:-1.
    --subset                                        Create atom subsets for each resonance.
```

2.1.2 Usage examples

Input files

We will use several input files to generate isotopically-resolved `InChI`.

- `Molfile` example:

```
pentane-2_2-diol
OpenBabel02101812223D

 19 18  0  0  0  0  0  0  0  0999 V2000
    0.8986   -0.0477    0.0323 C   0  0  0  0  0  0  0  0  0  0  0  0
    5.0960    0.7629    2.7277 C   0  0  0  0  0  0  0  0  0  0  0  0
    2.4213   -0.0579   -0.0025 C   0  0  0  0  0  0  0  0  0  0  0  0
    3.0115    0.3840    1.3416 C   0  0  0  0  0  0  0  0  0  0  0  0
    4.5473    0.3524    1.3660 C   0  0  0  0  0  0  0  0  0  0  0  0
    4.9506   -0.9689    1.0442 O   0  0  0  0  0  0  0  0  0  0  0  0
    5.0892    1.2477    0.4081 O   0  0  0  0  0  0  0  0  0  0  0  0
    0.4966   -0.3656   -0.9348 H   0  0  0  0  0  0  0  0  0  0  0  0
    0.5151    0.9566    0.2461 H   0  0  0  0  0  0  0  0  0  0  0  0
    0.5209   -0.7306    0.7986 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.7913    0.0636    3.5124 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.7734    1.7738    2.9979 H   0  0  0  0  0  0  0  0  0  0  0  0
    6.1923    0.7728    2.7063 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.7620    0.6098   -0.8011 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.7573   -1.0707   -0.2471 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.6291   -0.2841    2.1226 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.6589    1.3989    1.5754 H   0  0  0  0  0  0  0  0  0  0  0  0
    5.5197   -1.2976    1.7554 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.6303    2.0937    0.5065 H   0  0  0  0  0  0  0  0  0  0  0  0
  1  3  1  0  0  0  0
  1  8  1  0  0  0  0
  1  9  1  0  0  0  0
  1 10  1  0  0  0  0
  2  5  1  0  0  0  0
  2 11  1  0  0  0  0
  2 12  1  0  0  0  0
  2 13  1  0  0  0  0
  3  4  1  0  0  0  0
  3 14  1  0  0  0  0
  3 15  1  0  0  0  0
  4  5  1  0  0  0  0
  4 16  1  0  0  0  0
  4 17  1  0  0  0  0
  5  6  1  0  0  0  0
  5  7  1  0  0  0  0
  6 18  1  0  0  0  0
  7 19  1  0  0  0  0
M  ISO  1   1  12
M  END
```

- Text file containing `InChI` string:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3
```

- `SDfile` (i.e. `Molfile` plus data) example:

```
pentane-2_2-diol
OpenBabel02101812223D

 19 18  0  0  0  0  0  0  0  0999 V2000
    0.8986   -0.0477    0.0323 C   0  0  0  0  0  0  0  0  0  0  0  0
    5.0960    0.7629    2.7277 C   0  0  0  0  0  0  0  0  0  0  0  0
    2.4213   -0.0579   -0.0025 C   0  0  0  0  0  0  0  0  0  0  0  0
```

(continues on next page)

(continued from previous page)

```
    3.0115    0.3840    1.3416 C   0  0  0  0  0  0  0  0  0  0  0  0
    4.5473    0.3524    1.3660 C   0  0  0  0  0  0  0  0  0  0  0  0
    4.9506   -0.9689    1.0442 O   0  0  0  0  0  0  0  0  0  0  0  0
    5.0892    1.2477    0.4081 O   0  0  0  0  0  0  0  0  0  0  0  0
    0.4966   -0.3656   -0.9348 H   0  0  0  0  0  0  0  0  0  0  0  0
    0.5151    0.9566    0.2461 H   0  0  0  0  0  0  0  0  0  0  0  0
    0.5209   -0.7306    0.7986 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.7913    0.0636    3.5124 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.7734    1.7738    2.9979 H   0  0  0  0  0  0  0  0  0  0  0  0
    6.1923    0.7728    2.7063 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.7620    0.6098   -0.8011 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.7573   -1.0707   -0.2471 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.6291   -0.2841    2.1226 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.6589    1.3989    1.5754 H   0  0  0  0  0  0  0  0  0  0  0  0
    5.5197   -1.2976    1.7554 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.6303    2.0937    0.5065 H   0  0  0  0  0  0  0  0  0  0  0  0
  1  3  1  0  0  0  0
  1  8  1  0  0  0  0
  1  9  1  0  0  0  0
  1 10  1  0  0  0  0
  2  5  1  0  0  0  0
  2 11  1  0  0  0  0
  2 12  1  0  0  0  0
  2 13  1  0  0  0  0
  3  4  1  0  0  0  0
  3 14  1  0  0  0  0
  3 15  1  0  0  0  0
  4  5  1  0  0  0  0
  4 16  1  0  0  0  0
  4 17  1  0  0  0  0
  5  6  1  0  0  0  0
  5  7  1  0  0  0  0
  6 18  1  0  0  0  0
  7 19  1  0  0  0  0
M  ISO  1   1  12
M  END
> <InChI>
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0

$$$$
```

Input file/string specification

As shown above, the `isoenum` command-line interface asks the user to provide one required parameter `<path-to-ctfile-file-or-inchi-file-or-inchi-string>`, which is the file or string with information required to create isotopically-resolved `InChI`:

- Path to `CTfile` (i.e. `Molfile` or `SDfile`).

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol
```

- Path to the file containing an `InChI`.

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.inchi
```

- `InChI` string.

```
$ python3 -m isoenum name 'InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3'
```

or

```
$ python3 -m isoenum name '1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3'
```

The isoenum name command

The `name` command of the `isoenum` command-line interface provides facilities to add isotopic layer information to a molecule in order to create isotopically-resolved `InChI`.

Isotopic layer specification: specific atoms option

The `-s` or `--specific` option allows the user to specify the isotopic information for an atom at a specific position within a molecule (e.g. carbon at position "2" will have absolute mass "13").

- To designate the isotope of a specific atom within a given `Molfile`, use the `-s` or `--specific` option. For example, specify the second carbon atom as carbon "13":

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -s 13:C:2
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --specific=13:C:2
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
```

- To designate the isotope for several atoms, repeat the `-s` or `--specific` option:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -s 13:C:1 -s 13:C:2
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --specific=13:C:1 --
↪specific=13:C:2
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
```

Note: Since the original file already contained the `ISO` specification for the first carbon atom, it did not change the designation of that atom (i.e. `i1+0` was retained).

- To ignore existing `ISO` specifications, provide the `-i` or `--ignore-iso` option:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -s 13:C:1 -s 13:C:2␣
↪-i
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --specific=C-13-1 --
↪specific=C-13-2 --ignore-iso
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1
```

Isotopic layer specification: all atoms of a specific type option

The `-a` or `--all` option allows the user to specify the isotopic information for all atoms of a specific type (e.g. all carbons within a molecule will have absolute mass "13" etc.)

- To add isotope designations to all atoms of a specific element, use the `-a` or `--all` option:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -a 13:C
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --all=13:C
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1
```

- To add isotope designations to different types of atoms, repeat the `-a` or `--all` option for each desired element:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -a 13:C -a 18:O
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --all=13:C --
↪all=18:O
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1,6+2,7+2
```

- To ignore existing `ISO` specifications, combine with the `-i` or `--ignore-iso` option:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -a 13:C -a 18:O -i
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --all=13:C --
↪all=18:O --ignore-iso
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,3+1,4+1,5+1,6+2,7+2
```

- Also the `-a` or `--all` option can be combined with the `-s` or `--specific` option which has higher priority:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -a 13:C -s 12:C:3 -i
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --all=13:C --
↪specific=12:C:3 --ignore-iso
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,3+0,4+1,5+1
```

Isotopic layer specification: enumerate atoms of specific type option

The `-e` or `--enumerate` option allows the user to create a set of `InChI` for a molecule with a different number of isotopes (e.g. create all `InChI` where the number of carbon atoms with absolute mass "13" ranges from 0 to 5).

- To enumerate atoms of a specific element type, use the `-e` or `--enumerate` option:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -e 13:C
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --enumerate=13:C
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0
```

- A minimum and maximum number can be set to limit `InChI` generation to desired minimum and maximum number of atoms of the specified element. For example, generate all possible `InChI` where the number of carbon "13" atoms is in the range from 3 to 4:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -e 13:C:3:4
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --enumerate=13:C:3:4
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+1
```

- To ignore existing `ISO` specifications, combine it with the `-i` or `ignore-iso` option:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -e 13:C:3:4 -i
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --
↪enumerate=13:C:3:4 --ignore-iso
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,3+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i2+1,3+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i2+1,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i2+1,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i2+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i3+1,4+1,5+1
```

- To enumerate multiple atom types just repeat the `-e` or `--enumerate` option for the desired element:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -e 13:C:3:4 -e␣
↪18:O:1:2
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --
↪enumerate=13:C:3:4 --enumerate=18:O:1:2
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1,6+2/t5-/m0/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1,6+2/t5-/m1/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,6+2/t5-/m0/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,6+2/t5-/m1/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+1,6+2/t5-/m0/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+1,6+2/t5-/m1/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+1,6+2/t5-/m0/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+1,6+2/t5-/m1/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+1,6+2/t5-/m0/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+1,6+2/t5-/m1/s1
```

- The `-e` (`--enumerate`) option can be combined with the `-a` (`--all`) and `-s` (`--specific`) options except the `-e` (`--enumerate`) option cannot specify the same element as the `-a` (`--all`) option.

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -e 13:C:2:4 -a 18:O␣
↪-s 12:C:3
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --
↪enumerate=13:C:2:4 --all=18:O --specific=12:C:3
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0,4+1,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0,4+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+0,4+1,5+1,6+2,7+2
```

- It is also possible to combine the `-e` or `--enumerate` option for the same element but different isotopes (also note that we are not specifying minimum number in this example, it will be set to 0 by default). For example, we want to generate `InChI` with up to 2 carbon "12" and up to 2 carbon "13":

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -e 13:C:2 -e 12:C:2
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --enumerate=13:C:2 -
↪-enumerate=12:C:2
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+0,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+0,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,5+1
```

(continues on next page)

(continued from previous page)

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+0
```

The isoenum ionize command

The `ionize` command of `isoenum` command-line interface provides facilities to add charge information to a given molecule.

For example, the following `InChI` represents amino acid L-Valine:

```
InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1
```

with the following chemical structure:

- To create `InChI` that represents the zwitterion form:

```
$ python3 -m isoenum ionize 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1' -z N:6:+1 -z O:8:-1 -f inchi
```

or

```
$ python3 -m isoenum ionize 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1' --state=N:6:+1 --state=O:8:-1 --format=inchi
```

Output:

```
InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/f/h6H
```

The generated `InChI` corresponds to the following structure:

Note: Notice that the `InChI` is non-standard. This is because zwitterionic forms of molecules cannot be represented in standard `InChI`, but can be represented with the standard fixed hydrogen layer (`FixedH`) extension.

For a second example, the following `InChI` represents neutral Adenosine monophosphate (AMP):

```
InChI=1S/C10H14N5O7P/c11-8-5-9(13-2-12-8)15(3-14-5)10-7(17)6(16)4(22-10)1-21-23(18,
↪19)20/h2-4,6-7,10,16-17H,1H2,(H2,11,12,13)(H2,18,19,20)/t4-,6-,7-,10-/m1/s1
```

with the following chemical structure:

- To create the biochemically-relevant ionized `InChI`:

```
$ python3 -m isoenum ionize 'InChI=1S/C10H14N5O7P/c11-8-5-9(13-2-12-8)15(3-14-5)10-
↪7(17)6(16)4(22-10)1-21-23(18,19)20/h2-4,6-7,10,16-17H,1H2,(H2,11,12,13)(H2,18,19,
↪20)/t4-,6-,7-,10-/m1/s1' -z O:18:-1 -z O:19:-1 -f inchi
```

or

```
$ python3 -m isoenum ionize 'InChI=1S/C10H14N5O7P/c11-8-5-9(13-2-12-8)15(3-14-5)10-
↪7(17)6(16)4(22-10)1-21-23(18,19)20/h2-4,6-7,10,16-17H,1H2,(H2,11,12,13)(H2,18,19,
↪20)/t4-,6-,7-,10-/m1/s1' --state=O:18:-1 --state=O:19:-1 --format=inchi
```

Output:

```
InChI=1/C10H14N5O7P/c11-8-5-9(13-2-12-8)15(3-14-5)10-7(17)6(16)4(22-10)1-21-23(18,
↪19)20/h2-4,6-7,10,16-17H,1H2,(H2,11,12,13)(H2,18,19,20)/p-2/t4-,6-,7-,10-/m1/s1/
↪fC10H12N5O7P/h11H2/q-2
```

The generated `InChI` corresponds to the following structure:

The isoenum nmr command

The `nmr` command of the `isoenum` command-line interface provides facilities to create isotopically-resolved `InChI` based on theoretical NMR coupling patterns (e.g. J1CH, J3HH, etc.).

For example, the following `InChI` represents amino acid L-Valine:

```
InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1
```

with the following chemical structure:

- To create the theoretically possible set of `InChI` for "1D1H" NMR experiment:

```
$ python3 -m isoenum nmr 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1' --type=1D1H --format=csv
```

or

```
$ python3 -m isoenum nmr 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1' -t 1D1H -f csv
```

Output:

```
[1H9,1H10,1H11]HResonance   InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1/i1H3/t3-,4-
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:13C1]J1CH        InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3/t3-,4-
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:1H15]J3HH        InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4-
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:13C1]J1CH + [1H9,1H10,1H11:1H15]J3HH
↪InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3,3H/t3-,
↪4-
[1H12,1H13,1H14]HResonance  InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1/i1H3/t3-,4+/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:13C2]J1CH      InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3/t3-,4+/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:1H15]J3HH      InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4+/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:13C2]J1CH + [1H12,1H13,1H14:1H15]J3HH
↪InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3,3H/t3-,
↪4+/m1
[1H15]HResonance    InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i3H
```

(continues on next page)

(continued from previous page)

```
[1H15]HResonance + [1H15:13C3]J1CH  InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-
↪2H3,(H,7,8)/t4-/m0/s1/i3+1H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,
↪6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4-
[1H15]HResonance + [1H15:1H12,1H13,1H14]J3HH        InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4+/m1
[1H15]HResonance + [1H15:1H16]J3HH  InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-
↪2H3,(H,7,8)/t4-/m0/s1/i3H,4H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,1H14]J3HH     InChI=1S/
↪C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,2H3,3H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H16]J3HH       InChI=1S/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H,4H/t3-,4-
[1H15]HResonance + [1H15:1H12,1H13,1H14]J3HH + [1H15:1H16]J3HH      InChI=1S/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H,4H/t3-,4+/m1
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,1H14]J3HH +␣
↪[1H15:1H16]J3HH   InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i1H3,2H3,3H,4H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH       InChI=1S/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H/t3-,4-
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H12,1H13,1H14]J3HH      InChI=1S/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H/t3-,4+/m1
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H16]J3HH        InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i3+1H,4H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,
↪1H14]J3HH   InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/
↪i1H3,2H3,3+1H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H16]J3HH␣
↪InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H,4H/
↪t3-,4-
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H12,1H13,1H14]J3HH + [1H15:1H16]J3HH␣
↪InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H,4H/
↪t3-,4+/m1
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,
↪1H14]J3HH + [1H15:1H16]J3HH InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1/i1H3,2H3,3+1H,4H
[1H16]HResonance    InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i4H
[1H16]HResonance + [1H16:13C4]J1CH  InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-
↪2H3,(H,7,8)/t4-/m0/s1/i4+1H
[1H16]HResonance + [1H16:1H15]J3HH  InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-
↪2H3,(H,7,8)/t4-/m0/s1/i3H,4H
[1H16]HResonance + [1H16:13C4]J1CH + [1H16:1H15]J3HH        InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i3H,4+1H
```

Note: Notice that the `csv` output includes a J-coupling description of the transient peak and the representative `InChI`.

Workflow to generate InChI for data deposition from NMR experiments

1. Obtain standard `InChI` with correct bonded structure and stereochemistry.
2. Convert to fully representative `InChI` with proper ionization states if necessary (the `ionize` command).
3. Enumerate partial isotopomer `InChI` (the `nmr` command).

4. Select appropriate partial isotopomer `InChI`.

For example, the following `InChI` represents amino acid L-Valine:

```
InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1
```

with the following chemical structure:

We want to generate partial isotopomer `InChI` for the zwitterionic form of l-valine. To change the ionization state within molecule, we need to use the `ionize` command on a standard `InChI`:

```
$ python3 -m isoenum ionize 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1' -z N:6:+1 -z O:8:-1 -f inchi
```

or

```
$ python3 -m isoenum ionize 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1' --state=N:6:+1 --state=O:8:-1 --format=inchi
```

Output:

```
InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/f/h6H
```

The generated `InChI` corresponds to the following structure:

Next, we want to enumerate all possible isotopomers for "1D1H" NMR experiment:

```
$ python3 -m isoenum nmr 'InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1/f/h6H' --type=1D1H --format=csv
```

or

```
$ python3 -m isoenum nmr 'InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1/f/h6H' -t 1D1H -f csv
```

Output:

```
[1H9,1H10,1H11]HResonance   InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1/i1H3/t3-,4-/f/h6H/i/tM
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:13C1]J1CH        InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3/t3-,4-/f/h6H/i/tM
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:1H15]J3HH        InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4-/f/h6H/i/tM
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:13C1]J1CH + [1H9,1H10,1H11:1H15]J3HH
↪InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3,3H/t3-,4-
↪/f/h6H/i/tM
[1H12,1H13,1H14]HResonance  InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1/i1H3/t3-,4+/m1/f/h6H/i/tM/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:13C2]J1CH      InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3/t3-,4+/m1/f/h6H/i/tM/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:1H15]J3HH      InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4+/m1/f/h6H/i/tM/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:13C2]J1CH + [1H12,1H13,1H14:1H15]J3HH
↪InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3,3H/t3-,
↪4+/m1/f/h6H/i/tM/m1
[1H15]HResonance    InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i3H/f/h6H
```

(continues on next page)

(continued from previous page)

```
[1H15]HResonance + [1H15:13C3]J1CH  InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,
↪(H,7,8)/t4-/m0/s1/i3+1H/f/h6H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,
↪6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4-/f/h6H/i/tM
[1H15]HResonance + [1H15:1H12,1H13,1H14]J3HH        InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/
↪h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4+/m1/f/h6H/i/tM/m1
[1H15]HResonance + [1H15:1H16]J3HH  InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,
↪(H,7,8)/t4-/m0/s1/i3H,4H/f/h6H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,1H14]J3HH     InChI=1/
↪C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,2H3,3H/f/h6H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H16]J3HH       InChI=1/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H,4H/t3-,4-/f/h6H/i/tM
[1H15]HResonance + [1H15:1H12,1H13,1H14]J3HH + [1H15:1H16]J3HH      InChI=1/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H,4H/t3-,4+/m1/f/h6H/i/tM/
↪m1
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,1H14]J3HH +␣
↪[1H15:1H16]J3HH   InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i1H3,2H3,3H,4H/f/h6H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH       InChI=1/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H/t3-,4-/f/h6H/i/tM
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H12,1H13,1H14]J3HH      InChI=1/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H/t3-,4+/m1/f/h6H/i/tM/m1
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H16]J3HH        InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i3+1H,4H/f/h6H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,
↪1H14]J3HH   InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/
↪i1H3,2H3,3+1H/f/h6H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H16]J3HH␣
↪InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H,4H/t3-
↪,4-/f/h6H/i/tM
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H12,1H13,1H14]J3HH + [1H15:1H16]J3HH␣
↪InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H,4H/t3-
↪,4+/m1/f/h6H/i/tM/m1
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,
↪1H14]J3HH + [1H15:1H16]J3HH InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1/i1H3,2H3,3+1H,4H/f/h6H
[1H16]HResonance    InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i4H/f/h6H
[1H16]HResonance + [1H16:13C4]J1CH  InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,
↪(H,7,8)/t4-/m0/s1/i4+1H/f/h6H
[1H16]HResonance + [1H16:1H15]J3HH  InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,
↪(H,7,8)/t4-/m0/s1/i3H,4H/f/h6H
[1H16]HResonance + [1H16:13C4]J1CH + [1H16:1H15]J3HH        InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i3H,4+1H/f/h6H
```

Finally, **select** appropriate `InChI` from the generated possibilities and **associate** them with the appropriate transient peak (chemical shift).

Output format

- There are several output formats available:
  - `inchi`: produces `InChI` string.
  - `sdf`: produces `SDfile` with one or more `Molfile` and `InChI` associated with it.
  - `mol`: the same as `sdf`.

  - `json`: produces JSON representation of `SDfile`.
  - `csv`: produces tab-separated csv file with the information from `SDfile` data block.

- To specify the `inchi` output format (which is set to default and does not require format specification), use the `-f` or `--format` option followed by `inchi`:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -s 13:C:2 -f inchi
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --specific=13:C:2 --
↪format=inchi
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
```

- To specify the `mol` or `sdf` output format, use the `-f` or `--format` option followed by `mol` or `sdf`:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -s 13:C:2 -f sdf
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --specific=13:C:2 --
↪format=sdf
```

Output:

```
pentane-2_2-diol
OpenBabel04241818183D

 19 18  0  0  0  0  0  0  0  0999 V2000
    0.8564    0.0224   -0.0199   C 0  0  0  0  0  0  0  0  0  0  0  0
    5.0590   -2.7653   -0.2642   C 0  0  0  0  0  0  0  0  0  0  0  0
    2.3767    0.0633   -0.0253   C 0  0  0  0  0  0  0  0  0  0  0  0
    2.9725   -1.3472   -0.1203   C 0  0  0  0  0  0  0  0  0  0  0  0
    4.5036   -1.3472   -0.1439   C 0  0  0  0  0  0  0  0  0  0  0  0
    4.9424   -0.5621   -1.2388   O 0  0  0  0  0  0  0  0  0  0  0  0
    5.0329   -0.7920    1.0484   O 0  0  0  0  0  0  0  0  0  0  0  0
    0.4514    1.0368    0.0457   H 0  0  0  0  0  0  0  0  0  0  0  0
    0.4813   -0.5495    0.8345   H 0  0  0  0  0  0  0  0  0  0  0  0
    0.4733   -0.4367   -0.9365   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.7458   -3.2426   -1.1982   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.7417   -3.3903    0.5788   H 0  0  0  0  0  0  0  0  0  0  0  0
    6.1556   -2.7490   -0.2585   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.7259    0.5602    0.8869   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.7092    0.6719   -0.8743   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.5969   -1.8221   -1.0358   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.6148   -1.9367    0.7329   H 0  0  0  0  0  0  0  0  0  0  0  0
    5.0489   -1.1442   -2.0068   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.6612   -1.2841    1.7969   H 0  0  0  0  0  0  0  0  0  0  0  0
  1  3  1  0  0  0  0
  1  8  1  0  0  0  0
  1  9  1  0  0  0  0
  1 10  1  0  0  0  0
  2  5  1  0  0  0  0
  2 11  1  0  0  0  0
```

(continues on next page)

(continued from previous page)

```
  2 12  1  0  0  0  0
  2 13  1  0  0  0  0
  3  4  1  0  0  0  0
  3 14  1  0  0  0  0
  3 15  1  0  0  0  0
  4  5  1  0  0  0  0
  4 16  1  0  0  0  0
  4 17  1  0  0  0  0
  5  6  1  0  0  0  0
  5  7  1  0  0  0  0
  6 18  1  0  0  0  0
  7 19  1  0  0  0  0
M  ISO  2   1  12   2  13
M  END
> <InChI>
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1

$$$$
```

- To specify the `json` output format, use the `-f` or `--format` option followed by `json`:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -s 13:C:2 -f json
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --specific=13:C:2 --
↪format=json
```

Output:

```
{
    "1": {
        "molfile": {
            "HeaderBlock": {
                "molecule_name": "",
                "software": "OpenBabel02101812223D",
                "comment": ""
            },
            "Ctab": {
                "CtabCountsLine": {
                    "number_of_atoms": "19",
                    "number_of_bonds": "18",
                    "number_of_atom_lists": "0",
                    "not_used1": "0",
                    "chiral_flag": "0",
                    "number_of_stext_entries": "0",
                    "not_used2": "0",
                    "not_used3": "0",
                    "not_used4": "0",
                    "not_used5": "0",
                    "number_of_properties": "999",
                    "version": "V2000"
                },
                "CtabAtomBlock": [
                    {
                        "x": "0.8986",
                        "y": "-0.0477",
```

(continues on next page)

(continued from previous page)


```
                        "z": "0.0323",
                        "atom_symbol": "C",
                        "mass_difference": "0",
                        "charge": "0",
                        "atom_stereo_parity": "0",
                        "hydrogen_count": "0",
                        "stereo_care_box": "0",
                        "valence": "0",
                        "h0designator": "0",
                        "not_used1": "0",
                        "not_used2": "0",
                        "atom_atom_mapping_number": "0",
                        "inversion_retention_flag": "0",
                        "exact_change_flag": "0"
                    },
                    ...
                ],
                "CtabBondBlock": [
                    {
                        "first_atom_number": "1",
                        "second_atom_number": "3",
                        "bond_type": "1",
                        "bond_stereo": "0",
                        "not_used1": "0",
                        "bond_topology": "0",
                        "reacting_center_status": "0"
                    },
                    ...
                ],
                "CtabPropertiesBlock": {
                    "ISO": [
                        {
                            "atom_number": "1",
                            "absolute_mass": "12"
                        },
                        {
                            "atom_number": "2",
                            "absolute_mass": "13"
                        }
                    ]
                }
            }
        },
        "data": {
            "InChI": [
                "InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1"
            ]
        }
    }
}
```


- To specify the `csv` output format, use the `-f` or `--format` option followed by `csv`:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -s 13:C:2 -f csv
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --specific=13:C:2 --
↪format=csv
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
```

Output file

- To save the generated output into a file, use the `-o` or `--output` option followed by the filename. For example, save the generated output in `inchi` format:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -a 13:C -f inchi -o␣
↪outfile.inchi
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --all=13:C --
↪format=inchi --output=outfile.inchi
```

The generated file will contain the following output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1
```

- To save the generated output in `mol` or `sdf` format:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -a 13:C -f sdf -o␣
↪outfile.sdf
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --all=13:C --
↪format=sdf --output=outfile.sdf
```

The generated file will contain the following output:

```
pentane-2_2-diol
OpenBabel04251811053D

 19 18  0  0  0  0  0  0  0  0999 V2000
    0.9237   -0.0881    0.1091   C 0  0  0  0  0  0  0  0  0  0  0  0
    5.1259   -2.4797    1.5667   C 0  0  0  0  0  0  0  0  0  0  0  0
    2.4438   -0.0580    0.0798   C 0  0  0  0  0  0  0  0  0  0  0  0
    3.0394   -1.2473    0.8454   C 0  0  0  0  0  0  0  0  0  0  0  0
    4.5756   -1.2658    0.8182   C 0  0  0  0  0  0  0  0  0  0  0  0
    4.9993   -1.2893   -0.5316   O 0  0  0  0  0  0  0  0  0  0  0  0
    5.1095   -0.1114    1.4395   O 0  0  0  0  0  0  0  0  0  0  0  0
    0.5176    0.7650   -0.4432   H 0  0  0  0  0  0  0  0  0  0  0  0
    0.5500   -0.0378    1.1365   H 0  0  0  0  0  0  0  0  0  0  0  0
    0.5406   -1.0041   -0.3524   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.8066   -3.4184    1.1046   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.8189   -2.4761    2.6168   H 0  0  0  0  0  0  0  0  0  0  0  0
    6.2250   -2.4670    1.5528   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.7928    0.8838    0.5163   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.7749   -0.0753   -0.9642   H 0  0  0  0  0  0  0  0  0  0  0  0
```

(continues on next page)

(continued from previous page)

```
    2.6598   -2.1729    0.3950   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.6864   -1.2108    1.8833   H 0  0  0  0  0  0  0  0  0  0  0  0
    5.1891   -2.2082   -0.7786   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.7262   -0.0485    2.3265   H 0  0  0  0  0  0  0  0  0  0  0  0
  1  3  1  0  0  0  0
  1  8  1  0  0  0  0
  1  9  1  0  0  0  0
  1 10  1  0  0  0  0
  2  5  1  0  0  0  0
  2 11  1  0  0  0  0
  2 12  1  0  0  0  0
  2 13  1  0  0  0  0
  3  4  1  0  0  0  0
  3 14  1  0  0  0  0
  3 15  1  0  0  0  0
  4  5  1  0  0  0  0
  4 16  1  0  0  0  0
  4 17  1  0  0  0  0
  5  6  1  0  0  0  0
  5  7  1  0  0  0  0
  6 18  1  0  0  0  0
  7 19  1  0  0  0  0
M  ISO  5   1  12   2  13   3  13   4  13   5  13
M  END
> <InChI>
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1

$$$$
```

- To save the generated output in `json` format:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -a 13:C -f json -o␣
↪outfile.json
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --all=13:C --
↪format=json --output=outfile.json
```

The generated file will contain the following output:

```
{
    "1": {
        "molfile": {
            "HeaderBlock": {
                "molecule_name": "",
                "software": "OpenBabel02101812223D",
                "comment": ""
            },
            "Ctab": {
                "CtabCountsLine": {
                    "number_of_atoms": "19",
                    "number_of_bonds": "18",
                    "number_of_atom_lists": "0",
                    "not_used1": "0",
                    "chiral_flag": "0",
                    "number_of_stext_entries": "0",
```

(continues on next page)

(continued from previous page)


```
                    "not_used2": "0",
                    "not_used3": "0",
                    "not_used4": "0",
                    "not_used5": "0",
                    "number_of_properties": "999",
                    "version": "V2000"
                },
                "CtabAtomBlock": [
                    {
                        "x": "0.8986",
                        "y": "-0.0477",
                        "z": "0.0323",
                        "atom_symbol": "C",
                        "mass_difference": "0",
                        "charge": "0",
                        "atom_stereo_parity": "0",
                        "hydrogen_count": "0",
                        "stereo_care_box": "0",
                        "valence": "0",
                        "h0designator": "0",
                        "not_used1": "0",
                        "not_used2": "0",
                        "atom_atom_mapping_number": "0",
                        "inversion_retention_flag": "0",
                        "exact_change_flag": "0"
                    },
                    ...
                ],
                "CtabBondBlock": [
                    {
                        "first_atom_number": "1",
                        "second_atom_number": "3",
                        "bond_type": "1",
                        "bond_stereo": "0",
                        "not_used1": "0",
                        "bond_topology": "0",
                        "reacting_center_status": "0"
                    },
                    ...
                ],
                "CtabPropertiesBlock": {
                    "ISO": [
                        {
                            "atom_number": "1",
                            "absolute_mass": "12"
                        },
                        {
                            "atom_number": "2",
                            "absolute_mass": "13"
                        }
                    ]
                }
            }
        },
        "data": {
            "InChI": [
                "InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,
↪5+1"
```


(continues on next page)

(continued from previous page)

```
            ]
        }
    }
}
```

- To save the generated output in `csv` format:

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol -a 13:C -f csv -o␣
↪outfile.csv
```

or

```
$ python3 -m isoenum name tests/example_data/pentane-2_2-diol.mol --all=13:C --
↪format=csv --output=outfile.csv
```

The generated file will contain the following output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1
```

2.2 The isoenum Docker Tutorial

2.2.1 The isoenum docker container

This section contains the examples provided above but with the use of a docker container with `isoenum` Python package and all its dependencies instead of using `isoenum` Python package directly.

After you `docker pull` or `docker build` the `isoenum` container, you can verify that it is available.

```
# docker images
```

You should see output similar to the following:

```
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
isoenum             latest              0e4c431aa519        1 day ago           862MB
```

2.2.2 Command-line interface

The `isoenum` package provides an easy-to-use command-line interface that allows the specification of isotopes for the creation of isotopically-resolved `InChI`.

- To access the `isoenum` command-line interface from docker container:

```
# docker run isoenum --help
```

Output:

```
Isotopic enumerator (isoenum) command-line interface

Usage:
    isoenum -h | --help
    isoenum --version
    isoenum name (<path-to-ctfile-file-or-inchi-file-or-inchi-string>)
```

(continues on next page)

(continued from previous page)

```
                [--specific=<isotope:element:position>...]
                [--all=<isotope:element>...]
                [--enumerate=<isotope:element:min:max>...]
                [--complete | --partial]
                [--ignore-iso]
                [--format=<format>]
                [--output=<path>]
                [--verbose]
    isoenum ionize (<path-to-ctfile-file-or-inchi-file-or-inchi-string>)
                (--state=<element:position:charge>)
                [--format=<format>]
                [--output=<path>]
    isoenum nmr (<path-to-ctfile-file-or-inchi-file-or-inchi-string>)
                [--type=<experiment-type>]
                [--jcoupling=<name>...]
                [--decoupled=<element>...]
                [--format=<format>]
                [--output=<path>]
                [--subset]
                [--verbose]

Options:
    -h, --help                                  Show this screen.
    --verbose                                   Print more information.
    -v, --version                               Show version.
    -a, --all=<isotope:element>                 Specify element and isotope, e.g. -a␣
↪13:C or --all=13:C
    -s, --specific=<isotope:element:position>   Specify element, isotope and specific␣
↪position,
                                                e.g. -s 13:C:1 or --specific=13:C:1.
    -e, --enumerate=<isotope:element:min:max>   Enumerate all isotopically-resolved␣
↪CTfile or InChI,
                                                e.g. -e 13:C:2:4 or --
↪enumerate=13:C:2:4
    -c, --complete                              Use complete labeling schema, i.e.␣
↪every atom must specify
                                                "ISO" property, partial labeling␣
↪schema will be used otherwise
                                                for specified labeling information␣
↪only.
    -p, --partial                               Use partial labeling schema, i.e.␣
↪generate labeling schema
                                                from the provided labeling information.
    -i, --ignore-iso                            Ignore existing "ISO" specification in␣
↪the CTfile or InChI.
    -f, --format=<format>                       Format of output: inchi, mol, sdf, csv,
↪ json [default: inchi].
    -o, --output=<path>                         Path to output file.
    -t, --type=<experiment-type>                Type of NMR experiment [default: 1D1H].
    -j, --jcoupling=<type>                      Allowed J couplings.
    -d, --decoupled=<element>                   Turn off J coupling for a given␣
↪element.
    -z, --state=<element:position:charge>       Create ionized form of InChI from␣
↪neutral molecule,
                                                e.g. N:6:+1, O:8:-1.
    --subset                                    Create atom subsets for each resonance.
```

2.2.3 Docker usage examples

Input files

We will use the same input files as above to generate isotopically-resolved `InChI`. Repeated here for convenience.

- `Molfile` example:

```
pentane-2_2-diol
OpenBabel02101812223D

 19 18  0  0  0  0  0  0  0  0999 V2000
    0.8986   -0.0477    0.0323 C   0  0  0  0  0  0  0  0  0  0  0  0
    5.0960    0.7629    2.7277 C   0  0  0  0  0  0  0  0  0  0  0  0
    2.4213   -0.0579   -0.0025 C   0  0  0  0  0  0  0  0  0  0  0  0
    3.0115    0.3840    1.3416 C   0  0  0  0  0  0  0  0  0  0  0  0
    4.5473    0.3524    1.3660 C   0  0  0  0  0  0  0  0  0  0  0  0
    4.9506   -0.9689    1.0442 O   0  0  0  0  0  0  0  0  0  0  0  0
    5.0892    1.2477    0.4081 O   0  0  0  0  0  0  0  0  0  0  0  0
    0.4966   -0.3656   -0.9348 H   0  0  0  0  0  0  0  0  0  0  0  0
    0.5151    0.9566    0.2461 H   0  0  0  0  0  0  0  0  0  0  0  0
    0.5209   -0.7306    0.7986 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.7913    0.0636    3.5124 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.7734    1.7738    2.9979 H   0  0  0  0  0  0  0  0  0  0  0  0
    6.1923    0.7728    2.7063 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.7620    0.6098   -0.8011 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.7573   -1.0707   -0.2471 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.6291   -0.2841    2.1226 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.6589    1.3989    1.5754 H   0  0  0  0  0  0  0  0  0  0  0  0
    5.5197   -1.2976    1.7554 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.6303    2.0937    0.5065 H   0  0  0  0  0  0  0  0  0  0  0  0
  1  3  1  0  0  0  0
  1  8  1  0  0  0  0
  1  9  1  0  0  0  0
  1 10  1  0  0  0  0
  2  5  1  0  0  0  0
  2 11  1  0  0  0  0
  2 12  1  0  0  0  0
  2 13  1  0  0  0  0
  3  4  1  0  0  0  0
  3 14  1  0  0  0  0
  3 15  1  0  0  0  0
  4  5  1  0  0  0  0
  4 16  1  0  0  0  0
  4 17  1  0  0  0  0
  5  6  1  0  0  0  0
  5  7  1  0  0  0  0
  6 18  1  0  0  0  0
  7 19  1  0  0  0  0
M  ISO  1   1  12
M  END
```

- Text file containing `InChI` string:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3
```

- `SDfile` (i.e. `Molfile` plus data) example:

```
pentane-2_2-diol
OpenBabel02101812223D

 19 18  0  0  0  0  0  0  0  0999 V2000
    0.8986   -0.0477    0.0323 C   0  0  0  0  0  0  0  0  0  0  0  0
    5.0960    0.7629    2.7277 C   0  0  0  0  0  0  0  0  0  0  0  0
    2.4213   -0.0579   -0.0025 C   0  0  0  0  0  0  0  0  0  0  0  0
    3.0115    0.3840    1.3416 C   0  0  0  0  0  0  0  0  0  0  0  0
    4.5473    0.3524    1.3660 C   0  0  0  0  0  0  0  0  0  0  0  0
    4.9506   -0.9689    1.0442 O   0  0  0  0  0  0  0  0  0  0  0  0
    5.0892    1.2477    0.4081 O   0  0  0  0  0  0  0  0  0  0  0  0
    0.4966   -0.3656   -0.9348 H   0  0  0  0  0  0  0  0  0  0  0  0
    0.5151    0.9566    0.2461 H   0  0  0  0  0  0  0  0  0  0  0  0
    0.5209   -0.7306    0.7986 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.7913    0.0636    3.5124 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.7734    1.7738    2.9979 H   0  0  0  0  0  0  0  0  0  0  0  0
    6.1923    0.7728    2.7063 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.7620    0.6098   -0.8011 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.7573   -1.0707   -0.2471 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.6291   -0.2841    2.1226 H   0  0  0  0  0  0  0  0  0  0  0  0
    2.6589    1.3989    1.5754 H   0  0  0  0  0  0  0  0  0  0  0  0
    5.5197   -1.2976    1.7554 H   0  0  0  0  0  0  0  0  0  0  0  0
    4.6303    2.0937    0.5065 H   0  0  0  0  0  0  0  0  0  0  0  0
  1  3  1  0  0  0  0
  1  8  1  0  0  0  0
  1  9  1  0  0  0  0
  1 10  1  0  0  0  0
  2  5  1  0  0  0  0
  2 11  1  0  0  0  0
  2 12  1  0  0  0  0
  2 13  1  0  0  0  0
  3  4  1  0  0  0  0
  3 14  1  0  0  0  0
  3 15  1  0  0  0  0
  4  5  1  0  0  0  0
  4 16  1  0  0  0  0
  4 17  1  0  0  0  0
  5  6  1  0  0  0  0
  5  7  1  0  0  0  0
  6 18  1  0  0  0  0
  7 19  1  0  0  0  0
M  ISO  1   1  12
M  END
> <InChI>
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0

$$$$
```

Input file/string specification

As shown above, the `isoenum` command-line interface asks the user to provide one required parameter `<path-to-ctfile-file-or-inchi-file-or-inchi-string>`, which is the file or string with information required to create isotopically-resolved `InChI`.

In order to provide the input file path to the `isoenum` docker container, you will need to mount it as a volume for the docker container so the container can see it.

Warning: You need to provide the absolute path to the input file, otherwise the docker container will not be able to see it.

For example, `-v /absolute/path/to/input.txt:/input.txt`, where path on the left side of `:` is the absolute path on the host machine and the path on the right side of `:` is the path within the docker container.

To illustrate, let's invoke the `isoenum` docker container and provide input files:

- Path to `CTfile` (i.e. `Molfile` or `SDfile`).

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol
```

- Path to the file containing an `InChI`.

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol
```

- `InChI` string.

```
# docker run isoenum name 'InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3'
```

or

```
# docker run isoenum name '1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3'
```

The isoenum name command

The `name` command of the `isoenum` command-line interface provides facilities to add isotopic layer information to molecule in order to create isotopically-resolved `InChI`.

Isotopic layer specification: specific atoms option

The `-s` or `--specific` option allows the user to specify the isotopic information for an atom at a specific position within a molecule (e.g. carbon at position "2" will have absolute mass "13").

- To designate the isotope of a specific atom within a given `Molfile`, use the `-s` or `--specific` option. For example, specify the second carbon atom as carbon "13":

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -s 13:C:2
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --specific=13:C:2
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
```

- To designate the isotope for several atoms, repeat the `-s` or `--specific` option:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -s 13:C:1 -s 13:C:2
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --specific=13:C:1 --specific=13:C:2
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
```

Note: Since the original file already contained the `ISO` specification for the first carbon atom, it did not change the designation of that atom (i.e. `i1+0` was retained).

- To ignore existing `ISO` specifications, provide the `-i` or `--ignore-iso` option:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -s 13:C:1 -s 13:C:2 -i
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --specific=13:C:1 --specific=13:C:2 --ignore-iso
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1
```

Isotopic layer specification: all atoms of a specific type option

The `-a` or `--all` option allows the user to specify the isotopic information for all atoms of a specific type (e.g. all carbons within a molecule will have absolute mass "13" etc.)

- To add isotope designations to all atoms of a specific element, use the `-a` or `--all` option:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -a 13:C
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --all=13:C
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1
```

- To add isotope designations to different types of atoms, repeat the `-a` or `--all` option for each desired element:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -a 13:C -a 18:O
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --all=13:C --all=18:O
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1,6+2,7+2
```

- To ignore existing `ISO` specifications, combine with the `-i` or `--ignore-iso` option:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -a 13:C -a 18:O -i
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --all=13:C --all=18:O --ignore-iso
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,3+1,4+1,5+1,6+2,7+2
```

- Also the `-a` or `--all` option can be combined with the `-s` or `--specific` option which has higher priority:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -a 13:C -s 12:C:3 -i
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --all=13:C --specific=12:C:3 --ignore-iso
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,3+0,4+1,5+1
```

Isotopic layer specification: enumerate atoms of specific type option

The `-e` or `--enumerate` option allows the user to create a set of `InChI` for a molecule with a different number of isotopes (e.g. create all `InChI` where the number of carbon atoms with absolute mass "13" ranges from 0 to 5).

- To enumerate atoms of a specific element type, use the `-e` or `--enumerate` option:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -e 13:C
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --enumerate=13:C
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+1
```

(continues on next page)

(continued from previous page)

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0
```

- A minimum and maximum number can be set to limit `InChI` generation to desired minimum and maximum number of atoms of the specified element. For example, generate all possible `InChI` where the number of carbon "13" atoms is in the range from 3 to 4:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -e 13:C:3:4
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --enumerate=13:C:3:4
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+1
```

- To ignore existing `ISO` specifications, combine it with the `-i` or `ignore-iso` option:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -e 13:C:3:4 -i
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --enumerate=13:C:3:4 --ignore-iso
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,2+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,3+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+1,4+1,5+1
```

(continues on next page)

(continued from previous page)

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i2+1,3+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i2+1,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i2+1,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i2+1,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i3+1,4+1,5+1
```

- To enumerate multiple atom types, repeat the `-e` or `--enumerate` option for the desired element:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -e 13:C:3:4 -e 18:O:1:2
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --enumerate=13:C:3:4 --enumerate=18:O:1:2
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1,6+2/t5-/m0/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1,6+2/t5-/m1/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,6+2/t5-/m0/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,6+2/t5-/m1/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+1,6+2/t5-/m0/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+1,6+2/t5-/m1/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+1,6+2/t5-/m0/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+1,6+2/t5-/m1/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+1,6+2/t5-/m0/s1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+1,6+2/t5-/m1/s1
```

- The `-e` (`--enumerate`) option can be combined with the `-a` (`--all`) and `-s` (`--specific`) options except the `-e` (`--enumerate`) option cannot specify the same element as the `-a` (`--all`) option.

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -e 13:C:2:4 -a 18:O -s 12:C:3
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --enumerate=13:C:2:4 --all=18:O --specific=12:C:3
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0,4+1,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0,4+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0,5+1,6+2,7+2
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+0,4+1,5+1,6+2,7+2
```

- It is also possible to combine the `-e` or `--enumerate` option for the same element but different isotopes (also note that we are not specifying minimum number in this example, it will be set to 0 by default). For example, we want to generate `InChI` with up to 2 carbon "12" and up to 2 carbon "13":

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -e 13:C:2 -e 12:C:2
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --enumerate=13:C:2 --enumerate=12:C:2
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+0,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+0,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,3+1,4+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,4+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,3+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+0,3+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,4+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0,5+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+0,4+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,5+0
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+0
```

The isoenum ionize command

The `ionize` command of `isoenum` command-line interface provides facilities to add charge information to a given molecule.

For example, the following `InChI` represents amino acid L-Valine:

```
InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1
```

with the following chemical structure:

- To create `InChI` that represents the zwitterion form:

```
# docker run isoenum ionize 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1' -z N:6:+1 -z O:8:-1 -f inchi
```

or

```
# docker run isoenum ionize 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1' --state=N:6:+1 --state=O:8:-1 --format=inchi
```

Output:

```
InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/f/h6H
```

The generated `InChI` corresponds to the following structure:

For a second example, the following `InChI` represents neutral Adenosine monophosphate (AMP):

```
InChI=1S/C10H14N5O7P/c11-8-5-9(13-2-12-8)15(3-14-5)10-7(17)6(16)4(22-10)1-21-23(18,
↪19)20/h2-4,6-7,10,16-17H,1H2,(H2,11,12,13)(H2,18,19,20)/t4-,6-,7-,10-/m1/s1
```

with the following chemical structure:

- To create the biochemically-relevant ionized `InChI`:

```
# docker run isoenum ionize 'InChI=1S/C10H14N5O7P/c11-8-5-9(13-2-12-8)15(3-14-5)10-
↪7(17)6(16)4(22-10)1-21-23(18,19)20/h2-4,6-7,10,16-17H,1H2,(H2,11,12,13)(H2,18,19,
↪20)/t4-,6-,7-,10-/m1/s1' -z O:18:-1 -z O:19:-1 -f inchi
```

or

```
# docker run isoenum ionize 'InChI=1S/C10H14N5O7P/c11-8-5-9(13-2-12-8)15(3-14-5)10-
↪7(17)6(16)4(22-10)1-21-23(18,19)20/h2-4,6-7,10,16-17H,1H2,(H2,11,12,13)(H2,18,19,
↪20)/t4-,6-,7-,10-/m1/s1' --state=O:18:-1 --state=O:19:-1 --format=inchi
```

Output:

```
InChI=1/C10H14N5O7P/c11-8-5-9(13-2-12-8)15(3-14-5)10-7(17)6(16)4(22-10)1-21-23(18,
↪19)20/h2-4,6-7,10,16-17H,1H2,(H2,11,12,13)(H2,18,19,20)/p-2/t4-,6-,7-,10-/m1/s1/
↪fC10H12N5O7P/h11H2/q-2
```

The generated `InChI` corresponds to the following structure:

The isoenum nmr command

The `nmr` command of the `isoenum` command-line interface provides facilities to create isotopically-resolved `InChI` based on theoretical NMR coupling patterns (e.g. J1CH, J3HH, etc.).

For example, the following `InChI` represents amino acid L-Valine:

```
InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1
```

with the following chemical structure:

- To create the theoretically possible set of `InChI` for "1D1H" NMR experiment:

```
# docker run isoenum nmr 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1' --type=1D1H --format=csv
```

or

```
# docker run isoenum nmr 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1' -t 1D1H -f csv
```

Output:

```
[1H9,1H10,1H11]HResonance   InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1/i1H3/t3-,4-
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:13C1]J1CH        InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3/t3-,4-
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:1H15]J3HH        InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4-
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:13C1]J1CH + [1H9,1H10,1H11:1H15]J3HH
↪InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3,3H/t3-,
↪4-
[1H12,1H13,1H14]HResonance  InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1/i1H3/t3-,4+/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:13C2]J1CH      InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3/t3-,4+/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:1H15]J3HH      InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4+/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:13C2]J1CH + [1H12,1H13,1H14:1H15]J3HH
↪InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3,3H/t3-,
↪4+/m1
[1H15]HResonance    InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i3H
[1H15]HResonance + [1H15:13C3]J1CH  InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-
↪2H3,(H,7,8)/t4-/m0/s1/i3+1H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,
↪6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4-
[1H15]HResonance + [1H15:1H12,1H13,1H14]J3HH        InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4+/m1
[1H15]HResonance + [1H15:1H16]J3HH  InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-
↪2H3,(H,7,8)/t4-/m0/s1/i3H,4H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,1H14]J3HH     InChI=1S/
↪C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,2H3,3H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H16]J3HH       InChI=1S/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H,4H/t3-,4-
[1H15]HResonance + [1H15:1H12,1H13,1H14]J3HH + [1H15:1H16]J3HH      InChI=1S/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H,4H/t3-,4+/m1
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,1H14]J3HH +
↪[1H15:1H16]J3HH   InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i1H3,2H3,3H,4H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH       InChI=1S/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H/t3-,4-
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H12,1H13,1H14]J3HH      InChI=1S/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H/t3-,4+/m1
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H16]J3HH        InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i3+1H,4H
```

(continues on next page)

(continued from previous page)

```
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,
↪1H14]J3HH   InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/
↪i1H3,2H3,3+1H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H16]J3HH
↪InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H,4H/
↪t3-,4-
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H12,1H13,1H14]J3HH + [1H15:1H16]J3HH
↪InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H,4H/
↪t3-,4+/m1
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,
↪1H14]J3HH + [1H15:1H16]J3HH InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1/i1H3,2H3,3+1H,4H
[1H16]HResonance    InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i4H
[1H16]HResonance + [1H16:13C4]J1CH  InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-
↪2H3,(H,7,8)/t4-/m0/s1/i4+1H
[1H16]HResonance + [1H16:1H15]J3HH  InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-
↪2H3,(H,7,8)/t4-/m0/s1/i3H,4H
[1H16]HResonance + [1H16:13C4]J1CH + [1H16:1H15]J3HH        InChI=1S/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i3H,4+1H
```

Note: Notice that the `csv` output includes a J-coupling description of the transient peak and the representative `InChI`.

Workflow to generate InChI for data deposition from NMR experiments

1. Obtain standard `InChI` with correct bonded structure and stereochemistry.
2. Convert to fully representative `InChI` with proper ionization states if necessary (the `ionize` command).
3. Enumerate partial isotopomer `InChI` (the `nmr` command).
4. Select appropriate partial isotopomer `InChI`.

For example, the following `InChI` represents amino acid L-Valine:

```
InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1
```

with the following chemical structure:

We want to generate partial isotopomer `InChI` for the zwitterionic form of l-valine. To change the ionization state within molecule, we need to use the `ionize` command on a standard `InChI`:

```
# docker run isoenum ionize 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1' -z N:6:+1 -z O:8:-1 -f inchi
```

or

```
# docker run isoenum ionize 'InChI=1S/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1' --state=N:6:+1 --state=O:8:-1 --format=inchi
```

Output:

```
InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/f/h6H
```

The generated `InChI` corresponds to the following structure:

Next, we want to enumerate all possible isotopomers for "1D1H" NMR experiment:

```
# docker run isoenum nmr 'InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1/f/h6H' --type=1D1H --format=csv
```

or

```
# docker run isoenum nmr 'InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1/f/h6H' -t 1D1H -f csv
```

Output:

```
[1H9,1H10,1H11]HResonance   InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1/i1H3/t3-,4-/f/h6H/i/tM
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:13C1]J1CH        InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3/t3-,4-/f/h6H/i/tM
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:1H15]J3HH        InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4-/f/h6H/i/tM
[1H9,1H10,1H11]HResonance + [1H9,1H10,1H11:13C1]J1CH + [1H9,1H10,1H11:1H15]J3HH
↪InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3,3H/t3-,4-
↪/f/h6H/i/tM
[1H12,1H13,1H14]HResonance  InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/
↪t4-/m0/s1/i1H3/t3-,4+/m1/f/h6H/i/tM/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:13C2]J1CH      InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3/t3-,4+/m1/f/h6H/i/tM/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:1H15]J3HH      InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4+/m1/f/h6H/i/tM/m1
[1H12,1H13,1H14]HResonance + [1H12,1H13,1H14:13C2]J1CH + [1H12,1H13,1H14:1H15]J3HH
↪InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1+1H3,3H/t3-,
↪4+/m1/f/h6H/i/tM/m1
[1H15]HResonance    InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i3H/f/h6H
[1H15]HResonance + [1H15:13C3]J1CH  InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,
↪(H,7,8)/t4-/m0/s1/i3+1H/f/h6H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,
↪6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4-/f/h6H/i/tM
[1H15]HResonance + [1H15:1H12,1H13,1H14]J3HH        InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/
↪h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H/t3-,4+/m1/f/h6H/i/tM/m1
[1H15]HResonance + [1H15:1H16]J3HH  InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,
↪(H,7,8)/t4-/m0/s1/i3H,4H/f/h6H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,1H14]J3HH     InChI=1/
↪C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,2H3,3H/f/h6H
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H16]J3HH       InChI=1/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H,4H/t3-,4-/f/h6H/i/tM
[1H15]HResonance + [1H15:1H12,1H13,1H14]J3HH + [1H15:1H16]J3HH      InChI=1/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3H,4H/t3-,4+/m1/f/h6H/i/tM/
↪m1
[1H15]HResonance + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,1H14]J3HH +
↪[1H15:1H16]J3HH   InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i1H3,2H3,3H,4H/f/h6H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH       InChI=1/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H/t3-,4-/f/h6H/i/tM
```

(continues on next page)

(continued from previous page)

```
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H12,1H13,1H14]J3HH      InChI=1/C5H11NO2/
↪c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H/t3-,4+/m1/f/h6H/i/tM/m1
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H16]J3HH        InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i3+1H,4H/f/h6H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,
↪1H14]J3HH   InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/
↪i1H3,2H3,3+1H/f/h6H
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H16]J3HH
↪InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H,4H/t3-
↪,4-/f/h6H/i/tM
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H12,1H13,1H14]J3HH + [1H15:1H16]J3HH
↪InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i1H3,3+1H,4H/t3-
↪,4+/m1/f/h6H/i/tM/m1
[1H15]HResonance + [1H15:13C3]J1CH + [1H15:1H9,1H10,1H11]J3HH + [1H15:1H12,1H13,
↪1H14]J3HH + [1H15:1H16]J3HH InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,
↪8)/t4-/m0/s1/i1H3,2H3,3+1H,4H/f/h6H
[1H16]HResonance    InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/
↪s1/i4H/f/h6H
[1H16]HResonance + [1H16:13C4]J1CH  InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,
↪(H,7,8)/t4-/m0/s1/i4+1H/f/h6H
[1H16]HResonance + [1H16:1H15]J3HH  InChI=1/C5H11NO2/c1-3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,
↪(H,7,8)/t4-/m0/s1/i3H,4H/f/h6H
[1H16]HResonance + [1H16:13C4]J1CH + [1H16:1H15]J3HH        InChI=1/C5H11NO2/c1-
↪3(2)4(6)5(7)8/h3-4H,6H2,1-2H3,(H,7,8)/t4-/m0/s1/i3H,4+1H/f/h6H
```

Finally, **select** appropriate `InChI` from the generated possibilities and **associate** them with the appropriate transient peak (chemical shift).

Output format

- There are several output formats available:
  - `inchi`: produces `InChI` string.
  - `sdf`: produces `SDfile` with one or more `Molfile` and `InChI` associated with it.
  - `mol`: the same as `sdf`.
  - `json`: produces JSON representation of `SDfile`.
  - `csv`: produces tab-separated csv file with the information from `SDfile` data block.
- To specify the `inchi` output format (which is set to default and does not require format specification), use the `-f` or `--format` option followed by `inchi`:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -s 13:C:2 -f inchi
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --specific=13:C:2 --format=inchi
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
```

- To specify the `mol` or `sdf` output format, use the `-f` or `--format` option followed by `mol` or `sdf`:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -s 13:C:2 -f sdf
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --specific=13:C:2 --format=sdf
```

Output:

```
pentane-2_2-diol
OpenBabel04241818183D

 19 18  0  0  0  0  0  0  0  0999 V2000
    0.8564    0.0224   -0.0199   C 0  0  0  0  0  0  0  0  0  0  0  0
    5.0590   -2.7653   -0.2642   C 0  0  0  0  0  0  0  0  0  0  0  0
    2.3767    0.0633   -0.0253   C 0  0  0  0  0  0  0  0  0  0  0  0
    2.9725   -1.3472   -0.1203   C 0  0  0  0  0  0  0  0  0  0  0  0
    4.5036   -1.3472   -0.1439   C 0  0  0  0  0  0  0  0  0  0  0  0
    4.9424   -0.5621   -1.2388   O 0  0  0  0  0  0  0  0  0  0  0  0
    5.0329   -0.7920    1.0484   O 0  0  0  0  0  0  0  0  0  0  0  0
    0.4514    1.0368    0.0457   H 0  0  0  0  0  0  0  0  0  0  0  0
    0.4813   -0.5495    0.8345   H 0  0  0  0  0  0  0  0  0  0  0  0
    0.4733   -0.4367   -0.9365   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.7458   -3.2426   -1.1982   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.7417   -3.3903    0.5788   H 0  0  0  0  0  0  0  0  0  0  0  0
    6.1556   -2.7490   -0.2585   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.7259    0.5602    0.8869   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.7092    0.6719   -0.8743   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.5969   -1.8221   -1.0358   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.6148   -1.9367    0.7329   H 0  0  0  0  0  0  0  0  0  0  0  0
    5.0489   -1.1442   -2.0068   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.6612   -1.2841    1.7969   H 0  0  0  0  0  0  0  0  0  0  0  0
  1  3  1  0  0  0  0
  1  8  1  0  0  0  0
  1  9  1  0  0  0  0
  1 10  1  0  0  0  0
  2  5  1  0  0  0  0
  2 11  1  0  0  0  0
  2 12  1  0  0  0  0
  2 13  1  0  0  0  0
  3  4  1  0  0  0  0
  3 14  1  0  0  0  0
  3 15  1  0  0  0  0
  4  5  1  0  0  0  0
  4 16  1  0  0  0  0
  4 17  1  0  0  0  0
  5  6  1  0  0  0  0
  5  7  1  0  0  0  0
  6 18  1  0  0  0  0
  7 19  1  0  0  0  0
M  ISO  2   1  12   2  13
M  END
> <InChI>
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1

$$$$
```

- To specify the `json` output format, use the `-f` or `--format` option followed by `json`:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -s 13:C:2 -f json
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --specific=13:C:2 --format=json
```

Output:


```
{
    "1": {
        "molfile": {
            "HeaderBlock": {
                "molecule_name": "",
                "software": "OpenBabel02101812223D",
                "comment": ""
            },
            "Ctab": {
                "CtabCountsLine": {
                    "number_of_atoms": "19",
                    "number_of_bonds": "18",
                    "number_of_atom_lists": "0",
                    "not_used1": "0",
                    "chiral_flag": "0",
                    "number_of_stext_entries": "0",
                    "not_used2": "0",
                    "not_used3": "0",
                    "not_used4": "0",
                    "not_used5": "0",
                    "number_of_properties": "999",
                    "version": "V2000"
                },
                "CtabAtomBlock": [
                    {
                        "x": "0.8986",
                        "y": "-0.0477",
                        "z": "0.0323",
                        "atom_symbol": "C",
                        "mass_difference": "0",
                        "charge": "0",
                        "atom_stereo_parity": "0",
                        "hydrogen_count": "0",
                        "stereo_care_box": "0",
                        "valence": "0",
                        "h0designator": "0",
                        "not_used1": "0",
                        "not_used2": "0",
                        "atom_atom_mapping_number": "0",
                        "inversion_retention_flag": "0",
                        "exact_change_flag": "0"
                    },
                    ...
                ],
                "CtabBondBlock": [
                    {
```


(continues on next page)

(continued from previous page)


```
                        "first_atom_number": "1",
                        "second_atom_number": "3",
                        "bond_type": "1",
                        "bond_stereo": "0",
                        "not_used1": "0",
                        "bond_topology": "0",
                        "reacting_center_status": "0"
                    },
                    ...
                ],
                "CtabPropertiesBlock": {
                    "ISO": [
                        {
                            "atom_number": "1",
                            "absolute_mass": "12"
                        },
                        {
                            "atom_number": "2",
                            "absolute_mass": "13"
                        }
                    ]
                }
            }
        },
        "data": {
            "InChI": [
                "InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1"
            ]
        }
    }
}
```


- To specify the `csv` output format, use the `-f` or `--format` option followed by `csv`:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol -s 13:C:2 -f csv
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol isoenum␣
↪name /pentane-2_2-diol.mol --specific=13:C:2 --format=csv
```

Output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1
```

Output file

In the case of using the `isoenum` docker container, both the input file and the output file must be mounted as volumes for the docker container to see them.

Warning: You need to provide the absolute path to the input and output files, otherwise the docker container will not be able to see them.

For example, `-v /absolute/path/to/input.txt:/input.txt`, where path on the left side of `:` is the absolute path on the host machine and the path on the right side of `:` is the path within the docker container.

In the same way, you will need to create an empty text file and mount it as a volume, so the docker container can write to it, `-v /absolute/path/to/output.txt:/output.txt`, where the path on the left side of `:` is the absolute path on the host machine and the path on the right side of `:` is the path within the docker container.

- To save the generated output into a file, use the `-o` or `--output` option followed by filename. For example, save the generated output in `inchi` format:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol \
             -v /absolute/path/to/outfile.inchi:/outfile.inchi \
             isoenum name /pentane-2_2-diol.mol -a 13:C -f inchi -o /outfile.inchi
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol \
             -v /absolute/path/to/outfile.inchi:/outfile.inchi \
             isoenum name /pentane-2_2-diol.mol --all=13:C --format=inchi --output=/
↪outfile.inchi
```

The generated file will contain the following output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1
```

- To save the generated output in `mol` or `sdf` format:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol \
             -v /absolute/path/to/outfile.sdf:/outfile.sdf \
             isoenum name /pentane-2_2-diol.mol -a 13:C -f sdf -o /outfile.sdf
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol \
             -v /absolute/path/to/outfile.sdf:/outfile.sdf \
             isoenum name /pentane-2_2-diol.mol --all=13:C --format=sdf --output=/
↪outfile.sdf
```

The generated file will contain the following output:

```
pentane-2_2-diol
OpenBabel04251811053D

 19 18  0  0  0  0  0  0  0  0999 V2000
    0.9237   -0.0881    0.1091   C 0  0  0  0  0  0  0  0  0  0  0  0
    5.1259   -2.4797    1.5667   C 0  0  0  0  0  0  0  0  0  0  0  0
    2.4438   -0.0580    0.0798   C 0  0  0  0  0  0  0  0  0  0  0  0
    3.0394   -1.2473    0.8454   C 0  0  0  0  0  0  0  0  0  0  0  0
    4.5756   -1.2658    0.8182   C 0  0  0  0  0  0  0  0  0  0  0  0
    4.9993   -1.2893   -0.5316   O 0  0  0  0  0  0  0  0  0  0  0  0
    5.1095   -0.1114    1.4395   O 0  0  0  0  0  0  0  0  0  0  0  0
    0.5176    0.7650   -0.4432   H 0  0  0  0  0  0  0  0  0  0  0  0
    0.5500   -0.0378    1.1365   H 0  0  0  0  0  0  0  0  0  0  0  0
    0.5406   -1.0041   -0.3524   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.8066   -3.4184    1.1046   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.8189   -2.4761    2.6168   H 0  0  0  0  0  0  0  0  0  0  0  0
    6.2250   -2.4670    1.5528   H 0  0  0  0  0  0  0  0  0  0  0  0
```

(continues on next page)

(continued from previous page)

```
    2.7928    0.8838    0.5163   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.7749   -0.0753   -0.9642   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.6598   -2.1729    0.3950   H 0  0  0  0  0  0  0  0  0  0  0  0
    2.6864   -1.2108    1.8833   H 0  0  0  0  0  0  0  0  0  0  0  0
    5.1891   -2.2082   -0.7786   H 0  0  0  0  0  0  0  0  0  0  0  0
    4.7262   -0.0485    2.3265   H 0  0  0  0  0  0  0  0  0  0  0  0
  1  3  1  0  0  0  0
  1  8  1  0  0  0  0
  1  9  1  0  0  0  0
  1 10  1  0  0  0  0
  2  5  1  0  0  0  0
  2 11  1  0  0  0  0
  2 12  1  0  0  0  0
  2 13  1  0  0  0  0
  3  4  1  0  0  0  0
  3 14  1  0  0  0  0
  3 15  1  0  0  0  0
  4  5  1  0  0  0  0
  4 16  1  0  0  0  0
  4 17  1  0  0  0  0
  5  6  1  0  0  0  0
  5  7  1  0  0  0  0
  6 18  1  0  0  0  0
  7 19  1  0  0  0  0
M  ISO  5   1  12   2  13   3  13   4  13   5  13
M  END
> <InChI>
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1

$$$$
```

- To save the generated output in `json` format:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol \
             -v /absolute/path/to/outfile.sdf:/outfile.json \
             isoenum name /pentane-2_2-diol.mol -a 13:C -f json -o outfile.json
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol \
             -v /absolute/path/to/outfile.sdf:/outfile.json \
             isoenum name /pentane-2_2-diol.mol --all=13:C --format=json --
↪output=outfile.json
```

The generated file will contain the following output:

```
{
    "1": {
        "molfile": {
            "HeaderBlock": {
                "molecule_name": "",
                "software": "OpenBabel02101812223D",
                "comment": ""
            },
            "Ctab": {
                "CtabCountsLine": {
                    "number_of_atoms": "19",
```

(continues on next page)

(continued from previous page)


```
            "number_of_bonds": "18",
            "number_of_atom_lists": "0",
            "not_used1": "0",
            "chiral_flag": "0",
            "number_of_stext_entries": "0",
            "not_used2": "0",
            "not_used3": "0",
            "not_used4": "0",
            "not_used5": "0",
            "number_of_properties": "999",
            "version": "V2000"
        },
        "CtabAtomBlock": [
            {
                "x": "0.8986",
                "y": "-0.0477",
                "z": "0.0323",
                "atom_symbol": "C",
                "mass_difference": "0",
                "charge": "0",
                "atom_stereo_parity": "0",
                "hydrogen_count": "0",
                "stereo_care_box": "0",
                "valence": "0",
                "h0designator": "0",
                "not_used1": "0",
                "not_used2": "0",
                "atom_atom_mapping_number": "0",
                "inversion_retention_flag": "0",
                "exact_change_flag": "0"
            },
            ...
        ],
        "CtabBondBlock": [
            {
                "first_atom_number": "1",
                "second_atom_number": "3",
                "bond_type": "1",
                "bond_stereo": "0",
                "not_used1": "0",
                "bond_topology": "0",
                "reacting_center_status": "0"
            },
            ...
        ],
        "CtabPropertiesBlock": {
            "ISO": [
                {
                    "atom_number": "1",
                    "absolute_mass": "12"
                },
                {
                    "atom_number": "2",
                    "absolute_mass": "13"
                }
            ]
        }
```


(continues on next page)

(continued from previous page)

```
            }
        },
        "data": {
            "InChI": [
                "InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,
↪5+1"
            ]
        }
    }
}
```

- To save the generated output in `csv` format:

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol \
             -v /absolute/path/to/outfile.sdf:/outfile.csv \
             isoenum name /pentane-2_2-diol.mol -a 13:C -f csv -o outfile.csv
```

or

```
# docker run -v /absolute/path/to/pentane-2_2-diol.mol:/pentane-2_2-diol.mol \
             -v /absolute/path/to/outfile.sdf:/outfile.csv \
             isoenum name /pentane-2_2-diol.mol --all=13:C --format=csv --
↪output=outfile.csv
```

The generated file will contain the following output:

```
InChI=1S/C5H12O2/c1-3-4-5(2,6)7/h6-7H,3-4H2,1-2H3/i1+0,2+1,3+1,4+1,5+1
```

2.3 The isoenum API Reference

This package provides routines to generate isotopically-resolved `InChI` (International Chemical Identifier) from non-isotopically labeled `CTfile` formatted files (e.g. `Molfiles`, `SDfiles`) or non-isotopically labeled InChI identifiers.

This package includes the following modules:

`cli` This module provides the command-line interface for the `isoenum` package.

`api` This module provides routines to augment `CTfile` objects with additional information and used by `isoenum` package CLI.

`labeling` This module provides functions for generating a labeling schema.

`nmr` This module provides descriptions of coupling combinations that could be observed within NMR experiments.

`openbabel` This module provides functions to call the Open Babel software to convert between `InChI` and `CTfile` formatted files.

`conf` This module provides the processing of configuration files necessary for isotopic enumerator.

`fileio` This module provides functions for generating `CTfile` objects and convert `CTfile` objects into `InChI`.

`utils` This module provides reusable utility functions.

`exceptions` This module provides `isoenum` package custom exceptions.

2.3.1 isoenum.cli

Isotopic enumerator (isoenum) command-line interface

Usage: isoenum -h | –help

isoenum –version

isoenum name (<path-to-ctfile-file-or-inchi-file-or-inchi-string>) [–specific=<isotope:element:position>...] [–all=<isotope:element>...] [–enumerate=<isotope:element:min:max>...] [–complete | –partial] [–ignore-iso] [–format=<format>] [–output=<path>] [–verbose]

isoenum ionize (<path-to-ctfile-file-or-inchi-file-or-inchi-string>) (–state=<element:position:charge>...) [–format=<format>] [–output=<path>]

isoenum nmr (<path-to-ctfile-file-or-inchi-file-or-inchi-string>) [–type=<experiment-type>] [–jcoupling=<name>...] [–decoupled=<element>...] [–format=<format>] [–output=<path>] [–subset] [–verbose]

isoenum vis (<path-to-ctfile-file-or-inchi-file-or-inchi-string>) (–format=<format>) (–output=<path>)

Options:

-h, --help Show this screen.

--verbose Print more information.

-v, --version Show version.

-a, --all=<isotope:element> Specify element and isotope, e.g. -a 13:C or –all=13:C

-s, --specific=<isotope:element:position> Specify element, isotope and specific position, e.g. -s 13:C:1 or –specific=13:C:1.

-e, --enumerate=<isotope:element:min:max> Enumerate all isotopically-resolved CTfile or InChI, e.g. -e 13:C:2:4 or –enumerate=13:C:2:4

-c, --complete Use complete labeling schema, i.e. every atom must specify "ISO" property, partial labeling schema will be used otherwise for specified labeling information only.

-p, --partial Use partial labeling schema, i.e. generate labeling schema from the provided labeling information.

-i, --ignore-iso Ignore existing "ISO" specification in the CTfile or InChI.

-f, --format=<format> Format of output: inchi, mol, sdf, csv, json [default: inchi].

-o, --output=<path> Path to output file.

-t, --type=<experiment-type> Type of NMR experiment [default: 1D1H].

-j, --jcoupling=<type> Allowed J couplings.

-d, --decoupled=<element> Turn off J coupling for a given element.

-z, --state=<element:position:charge> Create ionized form of InChI from neutral molecule, e.g. N:6:+1, O:8:-1.

--subset Create atom subsets for each resonance.

`isoenum.cli.`**`cli`**(*cmdargs*)

Process command-line arguments.

Parameters **`cmdargs`** (*dict*) – Command-line arguments.

Returns None.

Return type None

isoenum.cli.**save_output**(*outputstr*, *path*, *file_format*)
Save output results into file or print to stdout.

Parameters

- **outputstr** (*str*) – Output string.
- **path** (*str*) – Where to save results.
- **file_format** (*str*) – File format to create file extension.

Returns None.

Return type None

isoenum.cli.**create_output**(*sdfile*, *path=None*, *file_format='inchi'*)
Create output containing conversion results.

Parameters

- **sdfile** (SDfile.) – SDfile instance.
- **path** (*str*) – Path to where file will be saved.
- **file_format** (*str*) – File format: 'inchi', 'mol', 'sdf', 'json', or 'csv'.

Returns None.

Return type None

2.3.2 isoenum.labeling

This module contains a function to generate a labeling schema based on provided cli parameters.

isoenum.labeling.**create_labeling_schema**(*complete_labeling_schema*, *ignore_existing_isotopes*, *all_iso*, *specific_iso*, *existing_iso*, *enumerate_iso*, *isotopes_conf*, *ctfile*)
Create labeling schema.

Parameters

- **complete_labeling_schema** (*bool*) – Specifies if default isotopes to be added to isotopic layer.
- **ignore_existing_isotopes** (*bool*) – Specifies if will ignore existing isotopic layer.
- **all_iso** (*dict*) – Atom specific isotopes *–all* option.
- **specific_iso** (*dict*) – Atom number specific isotopes from *–specific* option.
- **existing_iso** (*dict*) – Atom number specific isotopes from Molfile.
- **enumerate_iso** (*list*) – List of isotopes from *–enumerate* option.
- **isotopes_conf** (*dict*) – Default isotopes.
- **ctfile** (Molfile) – Instance of Molfile.

Returns Labeling schema.

Return type list

2.3.3 isoenum.nmr

This module provides descriptions of coupling combinations that could be observed within NMR experiments.

class isoenum.nmr.**ResonanceAtom**(*atom*, *isotope*)
Resonance atom - part of the coupling path.

class isoenum.nmr.**Coupling**(*coupling_path=None*, *nmr_active_atoms=None*, *subset_atoms=None*)
Coupling provides information on the connectivity of molecules.

coupling_type
Coupling type.

Returns Coupling type.

Return type str

subset
Generate coupling path subsets.

Returns Coupling path subsets.

Return type list

is_resonance_compatible(*resonance*)
Test if coupling is compatible with resonance.

Parameters **resonance** (*Coupling*) – Subclass of *Coupling*.

Returns True if compatible, False otherwise.

Return type True or *False*

classmethod couplings(*atom*)
Generate list of possible couplings for a given atom type.

Parameters **atom** (ctfile.Atom) – Atom type.

Returns List of couplings.

Return type list

name
Specific coupling name that indicates interacting atoms.

Returns Specific coupling name.

Return type str

hydrogen_coupling_path
Coupling path that consists of hydrogen atoms.

Returns Coupling path.

Return type list

hydrogen_coupling_path_repr
Hydrogen coupling path representation.

Returns List of hydrogen groups representing hydrogen coupling path.

Return type list

class isoenum.nmr.**J1CH**(*coupling_path=None*, *nmr_active_atoms=None*, *subset_atoms=None*)
J1CH coupling type: C-H.

couplings(*carbon_atom*)
Generate list of possible J1CH couplings.

Parameters **carbon_atom** – Carbon atom.

Returns List of couplings.

Return type list

class isoenum.nmr.**J2HH**(*coupling_path=None*, *nmr_active_atoms=None*, *subset_atoms=None*)
J2HH coupling type: H-C-H.

couplings(*carbon_atom*)
Generate list of possible J2HH couplings.

Parameters **carbon_atom** – Carbon atom.

Returns List of couplings.

Return type list

class isoenum.nmr.**J3HH**(*coupling_path=None*, *nmr_active_atoms=None*, *subset_atoms=None*)
J3HH coupling type: H-C-C-H.

couplings(*carbon_atom*)
Generate list of possible J3HH couplings.

Parameters **carbon_atom** – Carbon atom.

Returns List of couplings.

Return type list

is_resonance_compatible(*resonance*)
Test if J3HH coupling is compatible with resonance.

Parameters **resonance** (*Coupling*) – Subclass of *Coupling*.

Returns True if compatible, False otherwise.

Return type True or *False*

class isoenum.nmr.**HResonance**(*coupling_path=None*, *nmr_active_atoms=None*, *subset_atoms=None*)
Hydrogen resonance.

couplings(*carbon_atom*)
Generate resonance.

Parameters **carbon_atom** – Carbon atom.

Returns List of couplings.

Return type list

class isoenum.nmr.**NMRExperiment**(*name*, *couplings*, *decoupled*, *default_coupling_definitions*)
NMR experiment.

generate_coupling_combinations(*molfile*, *subset=False*)
Generate possible J couplings for a Molfile.

class isoenum.nmr.**NMR1D1H**(*name*, *couplings=None*, *decoupled=None*, *default_coupling_definitions=(HResonance(coupling_path=None)*, *J1CH(coupling_path=None)*, *J2HH(coupling_path=None)*, *J3HH(coupling_path=None))*)
1D 1H NMR experiment.

generate_coupling_combinations(*molfile*, *subset=False*)
Generate 1D 1H NMR experiment J couplings for a `Molfile`.

Parameters

- **molfile** (`ctfile.Molfile`) – `Molfile` object.
- **subset** (`True` or `False`) – Generate couplings subsets?

Returns List of possible couplings for a given `Molfile`.

Return type `list`

class isoenum.nmr.**NMR1DCHSQC**(*name*, *couplings=None*, *decoupled=None*, *default_coupling_definitions=(HResonance(coupling_path=None)*, *J1CH(coupling_path=None)*, *J2HH(coupling_path=None)*, *J3HH(coupling_path=None))*)
1D 13C HSQC NMR experiment.

isoenum.nmr.**create_nmr_experiment**(*name*, *couplings=None*, *decoupled=None*)
Create NMR experiment upon provided experiment type.

Parameters

- **name** (*str*) – NMR experiment type.
- **couplings** (*list*) – List of coupling types.
- **decoupled** (*list*) – List of elements.

Returns NMR experiment.

Return type *NMRExperiment*.

2.3.4 isoenum.conf

This module provides processing of configuration files necessary for isotopic enumerator.

2.3.5 isoenum.fileio

This module provides routines to generate `CTfile` objects and convert `CTfile` objects into `InChI` and vice versa.

isoenum.fileio.**create_ctfile**(*path_or_id*, *xyx_coordinates='–gen2D'*, *explicit_hydrogens='-h'*)
Guess what type of path is provided, i.e. is it existing file in `Molfile` format, existing file in `SDfile` file format, existing file containing `InChI` string, or `InChI` string and tries to create `CTfile` object.

Parameters

- **path_or_id** (*str*) – Path to `Molfile`, `SDfile`, `InChI`, or `InChI` string.
- **xyx_coordinates** (*str*) – Option that generates x, y, z coordinates (e.g., "–gen2D" or "–gen3D").
- **explicit_hydrogens** (*str*) – Option that makes hydrogens atoms explicit when generating `CTfile` object.

Returns Subclass of `CTfile` object.

Return type `Molfile` or `SDfile`

isoenum.fileio.**create_ctfile_from_ctfile_str**(*ctfile_str*)
Create `CTfile` object from `CTfile` string.

Parameters **ctfile_str** (*str*) – CTfile string.

Returns Subclass of CTfile object.

Return type CTfile

isoenum.fileio.**create_ctfile_from_identifier_file**(*path*, *output_format='mol'*, ***options*)

Create CTfile instance from InChI or SMILES identifier file.

Parameters

- **path** (*str*) – Path to file containing InChI or SMILES identifier.
- **output_format** (*str*) – Output file format used by Open Babel to generate CTfile.
- **options** – Additional options to be passed to Open Babel.

Returns Subclass of CTfile object.

Return type CTfile

isoenum.fileio.**create_ctfile_from_identifier_str**(*identifier_str*, *output_format='mol'*, ***options*)

Create CTfile instance from InChI or SMILES identifier string.

Parameters

- **identifier_str** (*str*) – InChI or SMILES identifier string.
- **output_format** (*str*) – Output file format used by Open Babel to generate CTfile.
- **options** – Additional options to be passed to Open Babel.

Returns Subclass of CTfile object.

Return type CTfile

isoenum.fileio.**guess_identifier_format**(*identifier_str*)

Guess identifier format.

Parameters **identifier_str** (*str*) – Chemical identifier string.

Returns 'inchi' or 'smiles' string.

Return type str

isoenum.fileio.**create_inchi_from_ctfile_obj**(*ctf*, ***options*)

Create InChI from CTfile instance.

Parameters **ctf** (CTfile) – Instance of CTfile.

Returns InChI string.

Return type str

isoenum.fileio.**normalize_ctfile_obj**(*ctf*, *xyx_coordinates='–gen2D'*, *explicit_hydrogens='-h'*)

Normalize CTfile object.

Parameters **ctf** (CTfile) – CTfile object.

Returns Normalized CTfile object.

Return type CTfile

isoenum.fileio.**create_empty_sdfile_obj**()

Create empty SDfile object.

Returns SDfile object.

Return type SDfile

isoenum.fileio.**create_empty_molfile_obj**()
Create empty Molfile object.

Returns Molfile object.

Return type Molfile

isoenum.fileio.**create_svg_str**(*inchi_str*, ***options*)
Create SVG string from InChI identifier string.

Parameters

- **inchi_str** (*str*) – InChI identifier.
- **options** – Additional options to be passed to Open Babel.

Returns SVG XML code.

Return type str

isoenum.fileio.**circular_consistency_test**(*inchi_str*)
Perform conversion from InChI string to Molfile and back to InChI string.

Parameters **inchi_str** (*str*) – InChI string.

Returns None

Return type None

2.3.6 isoenum.utils

This module provides reusable utility functions.

isoenum.utils.**is_url**(*path*)
Test if path represents a valid URL.

Parameters **path** (*str*) – Path to file.

Returns True if path is valid url string, False otherwise.

Return type True or False

isoenum.utils.**all_combinations**(*items*)
Generate combinations of variable size.

Parameters **items** – Sequence of items.

Returns List of all combinations.

Return type list

2.3.7 isoenum.openbabel

This module provides *convert()* to convert between different file formats used in molecular modeling and computational chemistry (e.g. InChI, SMILES, Molfile, etc.).

isoenum.openbabel.**convert**(*input_file_path*, *output_file_path*, *input_format*, *output_format*, ***options*)
Convert between formats using Open Babel.

Parameters

- **input_file_path** (*str*) – Path to input file.
- **output_file_path** (*str*) – Path to output file.
- **input_format** (*str*) – Input file format.
- **output_format** (*str*) – Output file format.
- **options** – Additional key-value options to pass to Open Babel.

Returns None.

Return type None

2.3.8 isoenum.exceptions

Custom exceptions.

exception isoenum.exceptions.**EmptyCTFileError**
Invalid CTFile object error.

2.4 Release History

2.4.1 0.4.1 (2019-08-12)

Improvements

- Added custom exceptions module.

2.4.2 0.4.0 (2019-07-30)

Improvements

- Added documentation on how to install development version from GitHub.
- Simplified set of functions that interfaces with Open Babel.
- Added default output formats configuration file.
- Generalized functions that create `CTfile` objects to accept both `InChI` and `SMILES`.
- Added API functions module.
- Added functionality to annotate *Molfiles* that have the same *InChI* and coupling type but different labeling schema as magnetically equivalent.

Bugfixes

- Fixed bug that used arbitrary output file format extension to save the results.
- Fixed bug not being able to pass two-word options to Open Babel.

2.4.3 0.3.1 (2018-08-27)

Improvements

- Updated "The isoenum Tutorial" documentation.
- Restructured tutorial into two sections: "The isoenum Tutorial" and "The isoenum Docker Tutorial".

- Added logic to use "FixedH" option to create `InChI` for charged molecules.

Bugfixes

- Fixed bug that gave an error if existing "ISO" property was present.
- Fixed bug generating incorrect coupling for "C" atoms that did not have any "H" attached to it (e.g. "C=O").
- Fixed bug that did not create multiple subsets for "J3HH" couplings.

2.4.4 0.3.0 (2018-08-22)

Improvements

- Added `ionize` command and functionality in order to create charged versions of neutral molecules, e.g. zwitterion forms of amino acids (https://en.wikipedia.org/wiki/Zwitterion)
- Removed Open Babel system call stdout messages.

2.4.5 0.2.1 (2018-08-20)

Improvements

- Updated CLI isotope specification: `--specific=<isotope:element:position>`, `--all=<isotope:element>`, `--enumerate=<isotope:element:min:max>`.
- Updated all documentation examples according to new CLI.
- Removed trailing "n" from `InChI` strings.

2.4.6 0.2.0 (2018-08-15)

Improvements

- Added new command-line interface command `nmr` and module in order to generate `InChI` associated with observed coupling combination for a given NMR experiment type.
- Added "1D 1H" and "1D 13C HSQC" NMR experiments.
- Added new output formats "json" and "csv".
- Updated API documentation.

2.4.7 0.1.5 (2018-04-26)

Bugfixes

- Fixed incorrect rendering on PyPI project page.

2.4.8 0.1.4 (2018-04-25)

Improvements

- Added ReadTheDocs documentation: http://isoenum.readthedocs.io

2.4.9 0.1.3 (2018-04-20)

Improvements

- Added "Dockerfile" to create docker container with `isoenum` software and all required dependencies ready-to-use.
- Changes to cli options: replaced "-f" and "–full" with "-c" and "–complete" to specify labeling schema. Used "-f" for "–format" option.

2.4.10 0.1.2 (2018-04-19)

Improvements

- Added ability to save conversion results into file via "–output" cli option and in different formats ("inchi", "mol", and "sdf") via "–format" cli option.

2.4.11 0.1.1 (2018-04-17)

Improvements

- Added ability to process `SDfiles` in addition to `Molfiles` and `InChI`.
- Added basic unit tests for command-line interface.

Bugfixes

- Fixed bug of not including package data configuration files into source distribution.

2.4.12 0.1.0 (2018-04-16)

- Initial public release.

2.5 License

(continues on next page)

(continued from previous page)

```
NO EXPRESS OR IMPLIED LICENSES TO ANY PARTY'S PATENT RIGHTS ARE GRANTED BY
THIS LICENSE. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND
CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR
CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR
BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER
IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Python Module Index

i

Index

R

S