lirtmats_quick_start

December 4, 2025

1 Quick Start

This python script describes how to use `LiRTMaTS` python package. The input data and retention time reference files used here are in https://github.com/wanchanglin/lirtmats/tree/master/examples/data.

1.1 Setup

The users need to load python package `LAMP` before using `LiRTMaTS`. It's functions used here are for loading data set and summarising the matching results. For details, see https://github.com/wanchanglin/lamp.

[1]:
```
import sqlite3
import pandas as pd
from lamp import anno
import lirtmats.lirtmats as rtm
```

1.2 Data Loading

`LiRTMaTS` supports text files separated by comma (`,`) or tab (`\t`). The Microsoft's XLSX is also supported, using argument `sheet_name` to indicate which sheet is used for input data. The default is 0 for the first sheet.

Here we use a small example data set with `tsv` format. This data set includes peak list and intensity data matrix. `LiRTMaTS` requires peak list's name, m/z value and retention time. User needs to indicate the locations of feature name, m/z value, retention time and starting points of data matrix from data. Here they are 1, 2, 3 and 4, respectively.

[2]:
```
cols = [1, 2, 3, 4]
data_fn = "./data/df_pos_3.tsv"                # use tsv file
df = anno.read_peak(data_fn, cols, sep='\t')
df
```

[2]:
```
        name          mz          rt          D121          A122  \
0   M102T899  102.034153  898.850160  1.404584e+07  3.689953e+06
1   M102T849  102.034154  849.085350  1.473961e+07           NaN
2    M105T45  105.042677   45.353942  5.520865e+05  1.813279e+05
3    M105T54  105.054961   54.350049  6.669635e+05  4.833251e+06
4  M105T48_1  105.074216   47.538626  6.310113e+05           NaN
```

```
…          …           …           …             …             …
1995    M299T296  299.233645  295.569540  8.125150e+04  1.020165e+05
1996   M300T43_1  299.919504   42.832066  5.042924e+04           NaN
1997     M300T62  300.119720   62.428854           NaN  3.914945e+05
1998  M300T285_2  300.124255  285.061758           NaN  4.602130e+05
1999    M300T288  300.181271  287.944377  7.880306e+05  1.738638e+06

              A125          A126          A127          A128          B131  \
0     3.598363e+06  1.138875e+07  4.887524e+06  2.104782e+06  7.288258e+06
1     5.934387e+06           NaN  4.607624e+06  5.969186e+06  3.367949e+06
2     2.734923e+05  2.342655e+05  6.241395e+04  1.068277e+05  1.192451e+05
3     2.137479e+06  1.552473e+06  1.753294e+06  2.301363e+06           NaN
4     5.199302e+05  4.302566e+05  5.650141e+05  3.635406e+05  1.096530e+06
…              …             …             …             …             …
1995  2.209362e+05  3.557402e+05  6.039153e+05           NaN  2.330915e+05
1996           NaN  2.222376e+05  3.763288e+05  2.094474e+05  1.163715e+05
1997  5.182468e+05  7.492101e+05  1.546338e+06  5.741346e+05  9.712791e+05
1998  4.559729e+05  9.718658e+05  3.864969e+05  3.877729e+05  1.315307e+06
1999  1.113482e+06  4.063701e+06  3.788191e+06  1.201084e+06  2.988076e+06

      …          E214          E215          E216          H234  \
0     …  3.125203e+06  3.608369e+06           NaN  4.763811e+06
1     …  1.276006e+07  1.490770e+07  2.880142e+06  4.263577e+06
2     …  3.092946e+04  1.788324e+05  1.810794e+05  3.225256e+05
3     …  1.186390e+06  3.001167e+06  2.558921e+06           NaN
4     …  7.882748e+05           NaN  9.822090e+05  4.974403e+05
…   …             …             …             …             …
1995  …  3.872671e+05  1.632064e+05  7.224218e+04  3.678394e+04
1996  …  4.035525e+05  2.032260e+05  2.700920e+05           NaN
1997  …  8.554399e+05  7.431820e+05  8.878200e+05           NaN
1998  …  2.418197e+06  2.917536e+06  9.108396e+05  4.583314e+05
1999  …  2.907005e+06  3.365814e+06  2.761628e+06  1.865813e+06

              H235          H236          H237          H238          H239  \
0     2.281365e+06           NaN  3.404450e+06  3.720441e+06  4.539032e+05
1              NaN           NaN           NaN  4.437697e+06  6.777076e+06
2              NaN  3.734778e+05  1.935349e+05           NaN  1.094705e+05
3              NaN  1.695460e+06           NaN  1.834140e+06  1.029692e+06
4     3.604541e+05  1.340656e+06           NaN           NaN  6.020203e+05
…              …             …             …             …             …
1995  9.526812e+04  5.785549e+04  6.183749e+05           NaN  2.915690e+04
1996  2.675647e+05  2.695188e+05  2.750383e+05  2.882957e+05  6.720465e+04
1997  3.625514e+05  4.987110e+05  1.393237e+06  5.217566e+05           NaN
1998  4.022556e+05  2.673259e+05           NaN           NaN  8.926295e+04
1999  1.956308e+06           NaN  2.918514e+06           NaN           NaN

              H240
```

```
0                NaN
1       6.341930e+06
2       1.946732e+05
3       4.382618e+05
4       3.597655e+05
…                …
1995             NaN
1996    3.352428e+05
1997    1.257126e+05
1998    2.126753e+04
1999             NaN

[2000 rows x 40 columns]
```

Data frame `df` now includes only `name`, `mz`, `rt` and intensity data matrix.

1.3 Retention Time Matching

To perform retention time matching, users use either default retention time library or their own reference file. The reference file must have one column: `rt_lib` which is used for retention time matching with a range or torrance in seconds. Also the column `ion_mode` should be required for indication of positive or negative mode matching. If `ion_mode` is not included in the reference file, all rows will be used for matching.

[3]:
```python
ion_mode = "pos"
# ref_path = ""  # if empty, use default reference file for matching
ref_path = "./data/rt_lib_202509.tsv"
ref = rtm.read_rt(ref_path, ion_mode=ion_mode)
ref
```

[3]:
```
                identifier                         metabolite_name  rt_lib  \
0     ACMG_aqC18_POS_0001                    MS5029_Isovaleraldehyde    24.6
1     ACMG_aqC18_POS_0002        L057_Dihydroxyfumaric acid hydrate    27.0
2     ACMG_aqC18_POS_0003                          L061_Benzoic acid    27.0
3     ACMG_aqC18_POS_0004                               L052_Spermine    28.2
4     ACMG_aqC18_POS_0005                             L021_Spermidine    30.0
…                       …                                        …      …
2827  ACMG_aqC18_POS_1412                       LIM3312_Cholesterol   659.4
2828  ACMG_aqC18_POS_1413              L013_5alpha-Cholestan-3-one   672.6
2829  ACMG_aqC18_POS_1414    LIM3310_5alpha-Cholest-7-en-3beta-ol   675.0
2830  ACMG_aqC18_POS_1415                  L0302_5alpha-Cholestanol   681.6
2831  ACMG_aqC18_POS_1416               L045_10Z-Nonadecenoic acid   723.6

                         inchikey  ion_mod
0     QPUYECUOLPXSFR-UHFFFAOYSA-N  positive
1     SEKGMJVHSBBHRD-WZHZPDAFSA-M  positive
2     DMBUODUULYCPAK-UHFFFAOYSA-N  positive
3     XDSPGKDYYRNYJI-IUPFWZBJSA-N  positive
```

```
4     HELXLJCILKEWJH-NCGAPWICSA-N  positive
…                               …         …
2827  ASOSVCXGWPDUGN-UHFFFAOYSA-N  negative
2828  XQCZBXHVTFVIFE-UHFFFAOYSA-N  negative
2829  WLFXSECCHULRRO-UHFFFAOYSA-N  negative
2830  YCIMNLLNPGFGHC-UHFFFAOYSA-N  negative
2831  QIGBRXMKCJKVMJ-UHFFFAOYSA-N  negative

[2832 rows x 5 columns]
```

`rt_tol` is a threshold for the retention time matching window. The unit is seconds and the default value is 5.

[4]:
```
rt_tol = 5
res = rtm.comp_match_rt(df, ref, rt_tol)
res
```

[4]:
```
              id          rt          identifier  \
0        M105T45   45.353942  ACMG_aqC18_POS_0280
1        M105T45   45.353942  ACMG_aqC18_POS_0281
2        M105T45   45.353942  ACMG_aqC18_POS_0282
3        M105T45   45.353942  ACMG_aqC18_POS_0283
4        M105T45   45.353942  ACMG_aqC18_POS_0284
…              …           …                   …
150065  M300T288  287.944377  ACMG_aqC18_POS_0942
150066  M300T288  287.944377  ACMG_aqC18_POS_0943
150067  M300T288  287.944377  ACMG_aqC18_POS_0944
150068  M300T288  287.944377  ACMG_aqC18_POS_0945
150069  M300T288  287.944377  ACMG_aqC18_POS_0945

                          metabolite_name rt_lib                     inchikey  \
0       L0309_Asymmetric dimethylarginine   40.5  ZDLDXNCMJBOYJV-YFKPBYRVSA-N
1       MS5037_Ribonic acid gamma-lactone   40.5  DAUAQNGYDSHRET-UHFFFAOYSA-N
2               L018_L-Dihydroorotic acid   40.5  KCDXJAYRVLXPFO-UHFFFAOYSA-N
3                        L030_Stachydrine   40.5  ITECRQOOEQWFPE-UHFFFAOYSA-N
4                    L072_Aminoadipic acid   40.5  JYPHNHPXFNEZBR-UHFFFAOYSA-N
…                                       …    …                            …
150065              MS5008_Ethyl crotonate  291.0  OZWKMVRBQXNZKK-UHFFFAOYSA-N
150066          MS5032_2-Phenyl-1-propanol  291.0  DKYWVDODHFEZIM-UHFFFAOYSA-N
150067         L015_Methyl indole-3-acetate  291.0  RTIXKCRFFJGDFG-UHFFFAOYSA-N
150068              L003_Cinnamic aldehyde  291.6  FNYLWPVRPXGIIP-UHFFFAOYSA-N
150069              L003_Cinnamic aldehyde  291.6  FNYLWPVRPXGIIP-UHFFFAOYSA-N

         ion_mod  rt_range
0       positive         5
1       positive         5
2       positive         5
3       positive         5
```

```
4       positive        5
…          …         …
150065  negative        5
150066  negative        5
150067  negative        5
150068  positive        5
150069  negative        5

[150070 rows x 8 columns]
```

1.4 Summarize Results

The function `comp_summ` in package `LAMP` summarises the retention time matching.

```
[5]: sr, mr = anno.comp_summ(df, res)
```

This function combines peak table with retention time matching results and returns two results in different formats. `sr` is single row results for each peak id in peak table `df`:

```
[6]: sr
```

```
[6]:            name           mz          rt  rt_range  \
0        M100T54   100.075925   53.810924       5.0
1      M1015T254  1014.985384  253.626177       5.0
2       M101T228   101.060060  228.125403       5.0
3       M102T849   102.034154  849.085350       NaN
4       M102T899   102.034153  898.850160       NaN
…          …           …           …         …
1995    M865T700   865.244172  700.365420       NaN
1996    M919T647   918.701782  646.988220       5.0
1997  M925T237_1   924.898294  236.964462       5.0
1998    M933T267   933.410460  266.976471       5.0
1999    M934T242   933.932365  242.395371       5.0

                                            identifier  \
0     ACMG_aqC18_POS_0389::ACMG_aqC18_POS_0389::ACMG…
1     ACMG_aqC18_POS_0782::ACMG_aqC18_POS_0783::ACMG…
2     ACMG_aqC18_POS_0654::ACMG_aqC18_POS_0654::ACMG…
3                                                  NaN
4                                                  NaN
…                                                …
1995                                               NaN
1996  ACMG_aqC18_POS_1407::ACMG_aqC18_POS_1407::ACMG…
1997  ACMG_aqC18_POS_0690::ACMG_aqC18_POS_0691::ACMG…
1998  ACMG_aqC18_POS_0839::ACMG_aqC18_POS_0840::ACMG…
1999  ACMG_aqC18_POS_0720::ACMG_aqC18_POS_0721::ACMG…

                                       metabolite_name  \
0       L0488_Maleic acid::L0488_Maleic acid::L0321_L-…
1       L0481_3-Hydroxydecanedioic acid::LIM3308_Suber…
2       MS5018_Dimethyl maleate::MS5018_Dimethyl malea…
3                                                   NaN
4                                                   NaN
…                                                 …
1995                                                NaN
1996    L005_Vitamin K1::L005_Vitamin K1::LIM3314_Phyl…
1997    L0306_Syringic acid::L0315_ortho-Hydroxyphenyl…
1998    MS5012_Diethyl malonate::MS5019_Trimethylaceti…
1999    LIM3312_Aspartame::MS5023_Ethyl levulinate::MS…

                                                 rt_lib  \
0       48.9::48.9::49.2::49.2::50.4::50.4::50.4::50.4…
1       249.00000000000003::249.00000000000003::249.00…
2       223.2::223.2::223.8::223.8::223.8::223.8::223…
3                                                   NaN
4                                                   NaN
…                                                 …
1995                                                NaN
1996             642.6::642.6::643.2::643.2::647.1::647.1
1997    232.2::232.2::232.2::232.2::232.2::232.2::232…
1998    262.2::262.2::262.2::262.2::262.2::262.2::262…
1999    237.6::237.6::237.6::237.6::237.6::237.6::237…

                                               inchikey  \
0       JJVNINGBHGBWJH-UHFFFAOYSA-N::JJVNINGBHGBWJH-UH…
1       FVWJYYTZTCVBKE-ROUWMTJPSA-N::TVZGACDUOSZQKY-UH…
2       KIWQWJKWBHZMDT-UHFFFAOYSA-N::KIWQWJKWBHZMDT-UH…
3                                                   NaN
4                                                   NaN
…                                                 …
1995                                                NaN
1996    ZFDIRQKJPRINOQ-HYXAFXHYSA-N::ZFDIRQKJPRINOQ-HY…
1997    AFBPFSWMIHJQDM-UHFFFAOYSA-N::OISVCGZHLKNMSJ-UH…
1998    KEVYVLWNCKMXJX-UHFFFAOYSA-N::WTTJVINHCBCLGX-ZD…
1999    MBDOYVRWFFCFHM-SNAWJCMRSA-N::XPFVYQJUAUNWIW-UH…

                                                ion_mod
0       positive::negative::positive::negative::positi…
1       positive::positive::negative::negative::positi…
2       positive::negative::positive::positive::positi…
3                                                   NaN
4                                                   NaN
…                                                 …
1995                                                NaN
1996    positive::negative::positive::negative::positi…
```

```
1997   positive::positive::positive::positive::positi…
1998   positive::positive::positive::negative::negati…
1999   positive::positive::positive::positive::negati…

[2000 rows x 9 columns]
```

mr is multiple rows format if the match more than once from the reference file:

[7]:
```
mr
```

[7]:
```
             name          mz          rt           identifier  \
0         M100T54  100.075925   53.810924  ACMG_aqC18_POS_0389
1         M100T54  100.075925   53.810924  ACMG_aqC18_POS_0389
2         M100T54  100.075925   53.810924  ACMG_aqC18_POS_0390
3         M100T54  100.075925   53.810924  ACMG_aqC18_POS_0390
4         M100T54  100.075925   53.810924  ACMG_aqC18_POS_0391
…             …           …           …                    …
150218   M934T242  933.932365  242.395371  ACMG_aqC18_POS_0775
150219   M934T242  933.932365  242.395371  ACMG_aqC18_POS_0772
150220   M934T242  933.932365  242.395371  ACMG_aqC18_POS_0773
150221   M934T242  933.932365  242.395371  ACMG_aqC18_POS_0774
150222   M934T242  933.932365  242.395371  ACMG_aqC18_POS_0775

                           metabolite_name rt_lib  \
0                        L0488_Maleic acid   48.9
1                        L0488_Maleic acid   48.9
2                         L0321_L-Theanine   49.2
3                         L0321_L-Theanine   49.2
4                      L0310_Dihydrothymine   50.4
…                                      …    …
150218          L035_2-Methoxybenzoic acid  247.2
150219                MS5015_Phenylglyoxal  247.2
150220  MS5021_Ethyl 2-methylacetoacetate   247.2
150221            L012_Homoveratrumic acid  247.2
150222          L035_2-Methoxybenzoic acid  247.2

                           inchikey   ion_mod  rt_range
0       JJVNINGBHGBWJH-UHFFFAOYSA-N  positive       5.0
1       JJVNINGBHGBWJH-UHFFFAOYSA-N  negative       5.0
2       SULYEHHGGXARJS-UHFFFAOYSA-N  positive       5.0
3       SULYEHHGGXARJS-UHFFFAOYSA-N  negative       5.0
4       YPTJKHVBDCRKNF-UHFFFAOYSA-N  positive       5.0
…                               …       …       …
150218  RFKITWRHKUYMRJ-UHFFFAOYSA-N  positive       5.0
150219  QWIZNVHXZXRPDR-WSCXOGSTSA-N  negative       5.0
150220  BHTRKEVKTKCXOH-LBSADWJPSA-N  negative       5.0
150221  SEBFKMXJBCUCAI-UHFFFAOYSA-N  negative       5.0
150222  RFKITWRHKUYMRJ-UHFFFAOYSA-N  negative       5.0
```

```
[150223 rows x 9 columns]
```

All of results can be saved into a `sqlite3` database and use DB Browser for SQLite to view. Or save these results in other formats, such as TSV, CSV or XLSX, separately.

[8]:
```
f_save = False          # here we do NOT save results
db_out = "test.db"
sr_out = "test_s.tsv"
mr_out = "test_m.tsv"
xlsx_out = "test.xlsx"
```

[9]:
```
if f_save:
    # save all results into a sqlite3 database
    conn = sqlite3.connect(db_out)
    df[["name", "mz", "rt"]].to_sql("peaklist",
                                    conn,
                                    if_exists="replace",
                                    index=False)
    mr.to_sql("anno_mr", conn, if_exists="replace", index=False)
    sr.to_sql("anno_sr", conn, if_exists="replace", index=False)

    conn.commit()
    conn.close()

    # save results into text files
    sr.to_csv(sr_out, sep="\t", index=False)
    mr.to_csv(mr_out, sep="\t", index=False)

    # save results into Excel format
    with pd.ExcelWriter(xlsx_out, mode="w", engine="openpyxl") as writer:
        sr.to_excel(writer, sheet_name="single-row", index=False)
        mr.to_excel(writer, sheet_name="multiple-row", index=False)
```

It should be noted that saving of Excel file takes much longer time than text files.

1.5 End User Usages

`LiRTMaTS` provides two computation options: command line interface(CLI) and graphical user interface (GUI).

To use GUI, you need to open a terminal and type in:

```
$ lirtmats gui
```

To use CLI, open a terminal and type in command with required arguments, something like:

```
lirtmats cli \
  --input-data "./data/df_pos_3.tsv" \
  --input-sep "tab" \
  --col-idx "1, 2, 3, 4" \
  --rt-path "" \
  --rt-sep "tab" \
  --rt-tol "5.0" \
  --ion-mode "pos" \
  --save-db \
  --summ-type "xlsx" \
```

Execution of this command line will produce `df_pos_3_rtm.db` and `df_pos_3_rtm.xlsx` in the directory `./data/`. If the `summ-type` is `tsv` or `csv`, files `df_pos_3_rtm_s.tsv` or `df_pos_3_rtm_s.csv` and `df_pos_3_rtm_m.tsv` or `df_pos_3_rtm_m.csv` will be saved into `./data`.

For the best practice, you can create a bash script `.sh` (Linux and MacOS) or Windows script `.bat` to contain these CLI arguments. Change parameters in these files each time when processing new data set.

For example, there are `lirtmats_cli.sh` and `lirtmats_cli.bat` in https://github.com/wanchanglin/lirtmats/tree/master/examples.

- For Linux and MacOS terminal:

  ```
  $ chmod +x lirtmats_cli.sh
  $ ./lirtmats_cli.sh
  ```

- For Windows terminal:

  ```
  $ lirtmats_cli.bat
  ```

Note that if users use `xlsx` files for input data and reference file when using GUI or CLI, all data must be in the first sheet. If you use `LiRTMaTS` functions in your python scripts, there are no such requirements.