Mass spectrometry analysis of lipid binding properties of human lipid transfer proteins

Dénes Türei

March 22, 2016

1 Data input

1.1 MS1 intensities

In subdirectories named `<LTP name>_[neg|pos][ update]?` under `/gavin/projects/human_LTPS` and in `2015_06_Popeye` under the former, we collected all files named `features/*LABELFREE*/[feature|<LTP name>].csv`, except those of the controls (having *'ctrl'* instead of the LTP name in the directory names).

```
ltp.get_filenames()
```

From the column headers, sample IDs *(a09, a10, a11, a12, b01)* and variable types *(m/z, RT mean, Normalized Area)* were determined. Quality values were read then from the 2nd col, m/z values from the 4th, retention time ranges from the 5th and charges from the 6th column. The order of the samples is indifferent, the order of the variables should be always the same, starting from the 7th column. Now we read in only the *normalized areas,* because *retention times* should be in the range specified in 5th column, and the m/z's should be the same across all the fractions. Note: some typos occurring in the headers of certain files have been also corrected.

```
ltp.read_data()
ltp.read_file_np()
```

1.2 Time of measurements

Theoretic masses of lipids in the standard we read from *Metabolites.xlsx*. From the same sheet, measured masses of the most abundant ions are read, so we calculated drift values by day.

```
ltp.standards_theoretic_masses()
```

TODO: Read the time of each measurement from a file provided by Marco. Read the features from all measurements of the standard.

1.3 LTP containing samples

The file `control_sample.csv` for each LTP and for each fraction contains 0/1 values telling whether a fraction contains the LTP or not based on SDS pages. This file has been manually compiled by Antonella. Missing values are non-measured fractions (those will have *None* values later in Python, so can be distinguished from zeros).

```
ltp.read_samples()
```

1.4 Fractions boundaries

The elution volume boundaries for each fraction are defined in a small *csv* file called `fractions.csv`. E.g. fraction *a6* is from 0.60 to 0.75 ml, and so on.

```
ltp.protein_profiles()
```

1.5 UV absorbance values

In `SEC_profiles` directory, all *xls* files visited and based on the fraction boundaries, all measured absorbance values are looked up for each fraction. The minimum absorbance at each LTP subtracted from every value, because these

are often negative, what is nonsense. Then the mean of absorbance values are calculated for each fraction. Hereafter the series of these values for fractions *a9–b1* are referred as *protein profiles,* and saved to the file `proteins_by_fraction.csv` for further use.

```
ltp.protein_profiles()
ltp.write_pptable()
```

1.6 Known binders

One table from the review containing the literature curated binding properties of the LTPs has been saved to `binding_properties.csv`, with adding one column containing the lipid class abbreviations used everywhere later across this analysis. After the classification and identification of features, we calculated the enrichment of the known binders among the features classified as binder (Fisher's exact test).

```
ltp.read_binding_properties()
```

1.7 Exact masses from SwissLipids

The whole SwissLipids database was downloaded from here. We read in SwissLipids IDs, names and classification levels with monoisotopic exact masses (15^{th} column).

```
ltp.get_swisslipids_exact()
```

1.8 Exact masses from LipidMaps

The whole LipidMaps database was downloaded from here. From the extracted file `LMSDFDownload28Jun15FinalAll.sdf`, LipidMaps IDs, names, synonyms

and monoisotopic exact masses have been read in.

```
ltp.get_lipidmaps()
ltp.lipidmaps_exact()
```

1.9 Lipid names and most abundant adducts

These data are being read from `lipid_names.csv`. This table is from Antonella's thesis, with addition of 2 columns helping the classification of database records.

```
ltp.read_lipid_names()
```

1.10 MS2 fragments

The MS2 fragment masses have been annotated by Marco and Antonella based on the MS2 runs of the lipid standard (headgroup fragments), and calculated values of fatty acid fragments (files `lipid_fragments_positive_mode.txt` and `lipid_fragments_negative_mode.txt`). An extra column has been added to the lines of the headgroup fragments, listing the lipid classes where the fragment appears.

```
ltp.read_metabolite_lines()
```

1.11 Atomic masses, weights and isotopic abundances

The atomic mass of different isotoped of every elements was fetched from this table, while the weights of every element from this one. The relative abundances of isotopes have been read from here. Proton and electron masses are hardcoded as constants with 12 digits accuracy. These data are needed for all

calculations with adduct masses.

```
mass.getMasses()
mass.getMassMonoIso()
mass.getMassFirstIso()
mass.getWeightStd()
mass.getFreqIso()
```

1.12 MS2 intensities

Mgf files containing MS2 data are looked up in the *Results* subdirectory for each LTP and for each fraction. For each MS1 m/z value, MS2 records with matching m/z *(pepmass)* with accuracy of ± 0.02 and retention time are looked up, MS2 m/z values and intensities are read in.

```
ltp.ms2_filenames()
ltp.ms2_map()
ltp.ms2_main()
```

2 Recalibration against the instrument drift

TODO: Based on measurement timings, lipid standard masses, and measured masses from standards, calculate the drift of the instrument, and correct all m/z's by this. After apply a tolerance of ± 0.01 instead of ± 0.02.

3 Basic filters

All of these filters should have positive result for one feature to be considered valid.

3.1 Quality

Quality express for how long the feature can be followed by the mass spectrometer. Its value must be larger than or equal to 0.2.

```
ltp.quality_filter()
```

3.2 Charge

Now we consider only the features with charge $z = 1$.

```
ltp.charge_filter()
```

TODO: Consider $2-$ and $3-$ charges in negative mode, to have better chance to find PIP2 and PIP3.

3.3 Area

Area under the peak curve is proportional to the sum of the intensities of the feature across all detections. Its value should be greater than or equal to 10.000.

```
ltp.area_filter()
```

3.4 Peak size

The maximum intensity of protein containing fractions should be at least $2\times$ higher than the highest value of any other samples (non protein containing fractions, controls).

```
ltp.peaksize_filter()
```

TODO: Apply higher peak size ratio up to 5.0, depending on the area.

4 Similarity of protein profiles and feature intensity profiles

4.1 Removing non LTP containing mean absorbance values

All absorbance values from samples assumed to not contain the LTP have been replaced with zero.

```
ltp.zero_controls()
```

4.2 Normalizing profiles

All protein profiles have been divided by its maximum, so scaled to the range 0-1. All MS intensity profiles have been scaled to the range 0-1 (minimum subtracted, divided by maximum).

```
ltp.norm_profile()
ltp.norm_profiles()
```

4.3 All intensities vs. protein profile

The distance or similarity between intensity profiles and the protein profile have been computed using the following metrics: Spearman correlation, Pearson rank correlation, Robust Rank Correlation, Goodman-Kruskal's gamma, Kendall's tau and Euclidean distance. We divided the Euclidean distance by the number of values. At the current version these are not used except the last

one.

```
ltp.profiles_corrs()
ltp.roco()
ltp.euclidean_dist()
ltp.euclidean_dist_norm()
```

4.4 Calculating Euclidean distances

We built an all-to-all distance matrix of all the intensity profiles plus the protein profile, just like the protein profile was one of the intensity profiles.

```
ltp.euclide ltp.distance_matrix()
```

4.5 Clustering features

Using the distance matrix, we clustered the features applying Ward's linkage method. To select the cluster of those features most similar to the protein profile, we need to set a threshold. We tested thresholds based on the desired number of clusters and the percentage of the maximum distance. Currently the latter is used.

```
ltp.features_clustering()
```

TODO: In some cases expected known binders do not cluster together with the protein profile. In addition, the uncertainity of the UV absorbance profiles, and the low number of data points we use for calculating the distance, makes this step the most problematic point of the analysis. To resolve this issue, alternatively we should look at those features clustering together with known binders. Known binders are already known from the literature, this should be complemented with the results of the HPTLC experiments.

5 Comparing positive and negative modes

We look for matches between exact masses of positive and negative features. Here we consider the combinations of all possible adducts.

```
ltp.negative_positive2()
```

6 Lipid database lookup

For each MS1 m/z the exact masses have been calculated assuming all possible adducts. Those have been matched against the exact masses of all *Species* level entities in the databases. We tried to identify the class for each resulted hit, and dropped those where the assumed adduct was different than the most abundant adduct for the given lipid class, or the lipid class found to not ionize in the given mode. We kept all matching records where we could not identify the lipid class, or we did not know about the ionization and adduct formation characteristics of the lipid class.

```
ltp.lipid_lookup_exact()
```

7 Headgroup identification

7.1 From MS1 database records

We identified lipid classes (headgroups) for all records found matching one m/z by looking up a set of keywords in their names and synonyms. These keywords were manually collected by Denes and read from the file described at 1.9.

```
ltp.ms1_headgroups()
```

7.2 From MS2 data

We identified the headgroups as the intersection of the sets of possible headgroups for all the detected headgroup fragments. If the intersection is empty,

then we kept the union of all possible headgroups. If the intersection was of size one, then the identification is unambiguous.

With identification of the fatty acids in MS2, we limited the possible headgroups to those with these fatty acids might give the detected total mass. To achieve this, we compared the sum of the carbon count and unsaturated bound count of the fatty acids detected in MS2 with the same numbers extracted from the database records.

```
ltp.ms2_headgroups()
ltp.headgroups_by_fattya()
ltp.identity_combined()
ltp.combined_table()
```

TODO: Distinguish PG and BMP based on intensities of different fragments.

7.3 MS1 and MS2 combined

We took the intersection of the headgroups identified from MS1 (database records) and MS2. We need MS1 and MS2 evidences supporting each other and at least one of them unambiguous to accept the identity of a feature.

```
ltp.feature_identity_table()
```

8 Fatty acid identification

8.1 From MS1 records

For all database records, in the names and synonyms we found the number of carbon atoms and double bounds in the format *x:y*, and we summed these if necessary.

```
ltp.fattyacid_from_lipid_name()
```

8.2 From MS2 data

Among MS2 fragments we looked up the known fatty acid fragment masses and selected the two with the highest intensities.

```
ltp.ms2_fattya()
```