Adduct Clustering Model for MS Data

February 12, 2015

1 Introduction

To annotate precursor masses of peak features in an input file against some database of molecules, we have to do some crude form of discretisation anyway (when matching the mass values). In this report, we introduce the discretisation process early on – allowing us to separate peak features into bins. We then perform the clustering of peak features based on the possible adduct transformations that 'connects' a peak to these bins.

2 Adduct Transformations

A given precursor mass will be oberved with the addition of an adduct. For example, rather than measuring M, we might measure $M + H$, or $M + ACN$, or $M + ACN + H$ etc. Because we measure the mass per unit charge, working out the mass of the particlar ionization product (molecule of interest plus adduct) is not as easy as you might think. Put everything from adduct-notes.pdf here ..

3 Discretisation

We can bin the data along the mass dimension. We index peak features by $n = 1, ..., N$ and the bins (i.e. possible precursor mass clusters) by $k = 1, ..., K$. Given any peak feature n with observed mass m_n, we want to compute the precursor mass q_k, associated to a cluster k, by applying the

$M + H$ adduct inverse-transformation:

$$q_k = \frac{m_n|c| + ce - \sum_i h_i G_i}{n} \tag{1}$$

where c is the charge, e is the mass of the electron, h_i and G_i are the adduct parts. Specifically for the M+H adduct, $n = 1.0$ and $ce - \sum_i h_i G_i = 1.00727645199076$. Further details can be found in Adduct_notes.pdf.

Given q_k, we then create a mass bin centered at q_k

$$q_k \pm b_k \tag{2}$$

where $b_k = q_k \cdot w \cdot (1e - 6)$ is the bin width The value of w is specified by the user. This process is repeated for all N observed peak features, resulting in K bins, where $N = K$.

Each bin now corresponds to a 'valid' precursor mass that an observed peak can be assigned to – based on the crucial assumption that an acceptable precursor mass clustering should have the $M + H$ adduct peak inside. Additionally, a peak is connected to a bin only if it's within some RT tolerance value and only if the M+H adduct is the one with the largest intensity. More details on the implementation and data structure can be found in Section 6.

Note that in this binning scheme, there are usually overlapping bins. It's a good idea to explore other binning approach, e.g. non-overlapping bin at fixed interval, etc, and see if this affects the results significantly (don't think so?).

In dealing with multiple files, we can do the binning process iteratively. Process the first file, create all the bins. Process the second file and assigning peaks to existing bins (from the first file) if already exists and creating new bins if necessary. Repeat until all files are exhausted.

4 Discrete Model

4.1 Description

Denote the peak feature by $d_n = (x_n, y_n)$ where x_n is the mass value and y_n the RT value. We use the variable $z_n = k$ to denote the assignment of peak feature n to bin k.

Given the data, we want to infer the assignment of the z_n variables to the precursor mass clusters. Assume a fixed number of clusters based on the known number of 'valid' precursor masses K, each $z_1, ..., z_n$ is therefore a categorical variable independently drawn from a categorical distribution with parameter $\boldsymbol{\theta}$. In turn, the parameter vector $\boldsymbol{\theta}$ of length K is drawn from a Dirichlet distribution with parameter α. So, the likelihood of a peak n to be assigned into a cluster k depends on **(1)** whether there's a possible transformation from the observed mass x_n to any precursor mass q_k given the list of adducts, and **(2)** based on the RT values. The model is therefore

$$\begin{aligned} \boldsymbol{\theta} &\sim Dir(\alpha) && (3) \\ z_n = k &\sim Cat(\boldsymbol{\theta}) && (4) \\ d_n &\sim L(d_n|z_n = k, ...) && (5) \end{aligned}$$

The likelihood $L(d_n|z_n = k, ...)$ can be factorised into its mass and RT terms

$$L(d_n|z_n = k, ...) = p(x_n|z_n = k, ...) \cdot p(y_n|z_n = k, ...) \tag{6}$$

For the mass term $p(x_n|z_n = k, ...)$, let $I_k(x_n)$ to be the indicator function defined as

$$I_k(x_n) \begin{cases} 1 & \text{there is a known adduct transformation from } x_n \text{ to bin k} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

In short, $I_k(x_n) = 1$ if there is an adduct transformation that lets us reach cluster k from x_n (i.e. $q_k - b_k \leq x_n \leq q_k + b_k$ for precursor mass q_k and its interval b_k). We call this adduct transformation as a 'valid' transformation. For each x_n, define k^* to be the list of all such valid transformations originating from peak n. Then the mass likelihood of peak n in cluster k is

$$p(x_n|z_n = k, ...) = \frac{1}{|k^*|} I_k(x_n) \tag{8}$$

For the RT term $p(y_n|z_n = k, ...)$, y_n is normally distributed with mean μ_k and some precision (inverse variance) δ. The cluster mean μ_k is in turn drawn from another normal distribution with mean μ_0 and precision τ_0

$$\begin{aligned} p(y_n|z_n = k, \mu_k, \delta, ...) &= \mathcal{N}(y_n|\mu_k, \delta^{-1}) && (9) \\ p(\mu_k|\mu_0, \tau_0) &= \mathcal{N}(\mu_k|\mu_0, \tau_0^{-1}) && (10) \end{aligned}$$

For Gibbs sampling, we need the conditional distribution

$$\begin{aligned} p(z_n = k|\boldsymbol{\theta}, ...) &\propto (\alpha_k + z_k) \cdot L(d_n|z_n = k) && (11) \\ &= (\alpha_k + z_k) \cdot p(x_n|z_n = k, ...) \cdot p(y_n|z_n = k, ...) && (12) \end{aligned}$$

where $p(x_n|z_n = k, ...)$ is as defined in eq. (8). For the RT term, we marginalise over all values of μ_k and get:

$$p(y_n|z_n = k, ...) \quad = \quad \mathcal{N}(y_n|\beta_k, \lambda_k^{-1}) \tag{13}$$

where $\lambda_k = ((\tau_0 + \sigma c_k)^{-1} + \delta^{-1})^{-1}$ and $\beta_k = \frac{1}{\lambda_k}[(\mu_0\tau_0) + (\delta \sum_n y_{n\in k})]$. Here, $y_{n\in k}$ denotes the RT values of any peak n currently assigned to cluster k, and c_k the count of such peaks.

4.2 Results

Results can be found in the Python notebook at `http://nbviewer.ipython.org/github/sdrogers/metabolomics_tools/blob/master/discretisation/notebooks/Test_discrete_clustering.ipynb`

5 Continuous Model

Put continuous model description here

5.1 Description

5.2 Results

6 Data Structure

Implementations of the object models and data structures described in this report can be found inside `metabolomics_tools/discretisation/models.py`.

Throughout the whole report, N refers to the number of peaks in an input file and K refers to the total number of clusters. Peaks (i.e. features) are indexed by $n = 1, ..., N$ and clusters are indexed by $k = 1, ..., K$. Remember that by design, generally $N == K$ in some of our models described in this report.

A feature is a tuple of (m/z, RT, intensity) values, stored in a `Feature` object. Additionally, we also have a `DatabaseEntry` object, corresponding to a record from the molecule database. User-defined adduct transformations are stored in the `Transformation` object.

The `PeakData` object is a container of all those stuff above, plus more. Inside `PeakData`, we have the lists of features (`PeakData.features`), database entries (`PeakData.database`) and adduct transformations (`PeakData.transformations`). For computational convenience, the `PeakData` object also contains Numpy matrices of the m/z, RT and intensity values of peak features. Each attribute is stored in an N x 1 matrix. They are `PeakData.mass`, `PeakData.rt` and `PeakData.intensity`.

For the purpose of modelling, features are then discretised into bins – based on some user-defined mass and RT tolerances. The mapping between features to all the valid bins they can go into are stored in `PeakData.possible`, an N x K matrix where entries in the matrix are indices of the possible adduct transformations connecting a peak to any bin. This is subjected to the constraints on mass tolerance, RT tolerance and the additional constraint that the M+H adduct has to be the one with the largest intensity (i.e. there can be no other adduct transformation from a peak to any bin where the peak intensity is larger than the one for M+H). Additionally, there is another N x K matrix, `PeakData.transformed`, that stores the actual transformed mass values. The class `FileLoader` can be used to load some input file and prepare `PeakData` object and everything inside. The output from any of the clustering model we've developed is an N x K matrix of peak-to-cluster assignments. In the case of Gibbs sampling, this will usually contains the accumulated of counts of peak n in cluster k throughout the collected posterior samples.

7 Derivations

Put whatever derivations here for future reference.

7.1 Prior for finite Dirichlet-categorical mixture model

Some note on the derivation for the conditional prior in the standard Dirichlet-categorical finite mixture model. This is more or less the same for Dirichlet-multinomial mixture model, except that the multinomial version has an additional constant in front.

The joint distribution of the data is

$$p(z_1, ..., z_n, \boldsymbol{\theta}, \alpha) = \prod_n p(z_n = k|\boldsymbol{\theta})p(\boldsymbol{\theta}|\alpha) \tag{14}$$

$$p(z_n = k|\boldsymbol{\theta}) = Cat(\boldsymbol{\theta}) = \prod_k (\boldsymbol{\theta}_k)^{z_k} \tag{15}$$

$$p(\boldsymbol{\theta}|\alpha) = Dir(\alpha) = \frac{1}{B(\alpha)} \prod_k (\boldsymbol{\theta}_k)^{\alpha_k - 1} \tag{16}$$

For Gibbs sampling, we need $p(z_n|\boldsymbol{\theta}, ...)$. This is

$$\begin{aligned}
p(z_n = k|\boldsymbol{\theta}, ...) &\propto p(z_n = k, \boldsymbol{\theta}, ...) && (17)\\
&= p(z_n = k|\boldsymbol{\theta}, ...) \cdot p(\boldsymbol{\theta}|\alpha) && (18)\\
&= \int [p(z_n = k|\boldsymbol{\theta}, ...) \cdot p(\boldsymbol{\theta}|\alpha)] \, d\theta && (19)\\
&= \int \left[\prod_k (\boldsymbol{\theta}_k)^{z_k} \cdot \frac{1}{B(\alpha)} \prod_k (\boldsymbol{\theta}_k)^{\alpha_k - 1}\right] d\theta && (20)\\
&= \frac{1}{B(\alpha)} \int \left[\prod_k (\boldsymbol{\theta}_k)^{z_k + \alpha_k - 1}\right] d\theta && (21)\\
&= \frac{B(\alpha + z)}{B(\alpha)} && (22)
\end{aligned}$$

Eq (21) is obtained as such: we know

$$B(\alpha) = \int \left[\prod_k (\boldsymbol{\theta}_k)^{z_k + \alpha_k - 1}\right] d\theta \tag{23}$$

as it's the normalising constant in the multinomial pdf, so

$$B(\alpha + z) = \int \left[\prod_k (\boldsymbol{\theta}_k)^{z_k + \alpha_k - 1}\right] d\theta \tag{24}$$

Continuing from eq (22), the beta function is defined as $B(\alpha) = \frac{\prod_k \Gamma(\alpha_k)}{\Gamma(\sum_k \alpha_k)}$.

Expand everything, and we get

$$\begin{aligned}
p(z_n = k|\boldsymbol{\theta}, ...) &\propto \left(\frac{\prod_k \Gamma(\alpha_k + z_k)}{\Gamma(\sum_k \alpha_k + z_k)}\right)\left(\frac{\Gamma(\sum_k \alpha_k)}{\prod_k \Gamma(\alpha_k)}\right) && (25)\\
&= \left(\frac{\Gamma(\sum_k \alpha_k)}{\Gamma(\sum_k \alpha_k + z_k)}\right)\left(\frac{\prod_k \Gamma(\alpha_k + z_k)}{\prod_k \Gamma(\alpha_k)}\right) && (26)\\
&\propto \prod_k \frac{\Gamma(\alpha_k + z_k)}{\Gamma(\alpha_k)} && (27)\\
&\propto \prod_k \Gamma(\alpha_k + z_k) && (28)\\
&\propto ...? && (29)\\
&\propto \alpha_k + z_k && (30)
\end{aligned}$$

Something wrong with the notations used above ... what is z_k here? Should be the number of peaks assigned to cluster k.

References