Retrosynthetic Analysis Via Weighted Graphs

Marco A. Vivero Avila, University of California Berkeley

May 7, 2014

Introduction

Retrosynthetic analysis provides an organizational framework for planning the synthesis of a desired target molecule X given some set $\mathrm{Synth}(X)$ of chemical precursors, or synthons. In retrosynthetic analysis, one "works backwards" by starting with the target molecule X and decomposing it into synthons[1]. This retrosynthetic decomposition is denoted by

$$X \Rightarrow \sum_{i\in[n]} Y_i,\ Y_i \in \mathrm{Synth}(X) \text{ and } n \in \mathbb{N}.$$

The latter expression is taken to be a formal sum. Moreover, this expression assumes that there is a valid chemical reaction $\sum Y_i \Rightarrow X$.

Now, graphs provide a natural way of visually representing molecules as exemplified below:

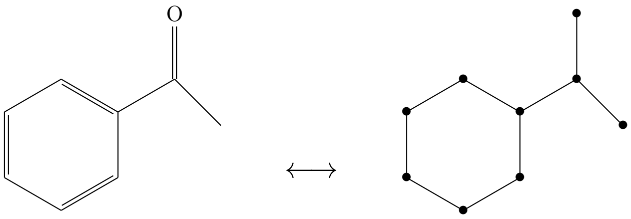

Figure 1

Therefore we can assign to any target molecule X a corresponding simple graph $G(X)$. Our objective is to take advantage of this graphical representation of molecules by modeling this retrosynthetic decomposition as a partition of $G(X)$. To do this, we will represent X by a pair $(G(X), \omega_X)$ where $\omega_X : E_{G(X)} \to \mathbb{R}_{\geq 0}$ assigns a non-negative weight to each edge of $G(X)$. These weights are to be interpreted as some measure of bond strength between the corresponding atoms/ functional groups.

This paper will be divided into three parts: (i) a review of data clustering methods; (ii) a review of Energy Partition Analysis which will be used to assess a weight function; (iii) an application of these algorithms on target molecules with known retrosynthetic decompositions.

1. Graph Partitions

We begin with a review of some basic graph theory terminology that will be used in the subsequent graph partition algorithms. We subsequently detour into the realm of statistical clustering, first briefly mentioning the k-means clustering algorithm, which will lead into our main graph partitioning tool: spectral clustering. We will subsequently mention another tool for graph partitioning known as semidefinite spectral clustering.

Graph Preliminaries

The following definitions are adapted from [3]. Let $G(V, E)$ be a simple graph with vertex set $V = [n] := \{1, ..., n-1, n\}$ for some $n \in \mathbb{N}$. We assume that if $ij \in E$, then there is a corresponding weight $w_{ij} \in \mathbb{R}_{\geq 0}$, for all $i, j \in V$. The adjacency matrix of G is the matrix A with ij^{th} entry

$$\delta_{ij} = \begin{cases} 1, & \text{if } ij \in E; \\ 0, & \text{otherwise.} \end{cases}$$

The weighted adjacency matrix of G is the matrix W with $(W)_{ij} = \delta_{ij} w_{ij}$ where $(W)_{ij} = 0$ if $ij \notin E$. The degree of vertex i is defined as

$$d_i := \sum_{j \in [n]} w_{ij}.$$

The degree matrix D of G is the matrix with $(D)_{ij} = d_i$ if $i = j$, and 0 otherwise.

A subset $S \subset V$ is connected if for any $i, j \in V$, there is a path $(i, k_1, k_2, ..., k_m, j)$ such that $k_h \in S$ for all $h \in [m]$. S is called a connected component if it is connected and $\{ij \in E : i \in S, j \in V \backslash S\} = \emptyset$.

A partition of G is a finite collection of nonempty subsets $G_1 := (V_1, E_1), ..., G_k := (V_k, E_k) \subset G$ satisfying $V_1 \cup \cdots \cup V_k = V$ and $V_i \cap V_j = \emptyset$ for $i \neq j$ with each having corresponding edge set $E_i := \{ij \in E : i, j \in V_i\}$.

The graph Laplacian of G is the matrix $L := D - W$.

Proposition 1. L is symmetric and positive semidefinite.

Proof: Clearly, $D = D^t$, and since G is a simple graph, $(W)_{ij} = (W)_{ji}$, and thus $L^t = (D - W) = D^t - W^t = D - W = L$. We have that for any $\mathbf{x} := (x_1, ..., x_n)^t \in \mathbb{R}^n$:

$$\begin{aligned} \mathbf{x}^t L \mathbf{x} = \mathbf{x}^t D \mathbf{x} - \mathbf{x}^t W \mathbf{x} &= \sum_{i \in [n]} d_i x_i^2 - \sum_{i \in [n]} \sum_{j \in [n]} w_{ij} x_i x_j \\ &= \frac{1}{2} \left(\sum_{i \in [n]} \sum_{j \in [n]} w_{ij} x_i^2 - 2 w_{ij} x_i x_j + w_{ij} x_j^2 \right) \\ &= \frac{1}{2} \sum_{i \in [n]} \sum_{j \in [n]} w_{ij} (x_i - x_j)^2. \end{aligned}$$

It follows that L is symmetric and positive-semidefinite. ■

Given a graph Laplacian L with eigenvalues $\lambda_1, ..., \lambda_n$, we assume, after reordering, that $\lambda_1 \leq ... \leq \lambda_n$. Hence, when we say that $\mathbf{v}_1, ..., \mathbf{v}_k$ are the first k eigenvectors of L, these are the eigenvectors corresponding to the k smallest eigenvalues of L.

Now, if G has k connected components $G_1, ..., G_k$, then after reordering of the vertices according to the components each vertex belongs to we clearly have that L will be a block matrix consisting of k blocks where block k is the Laplacian corresponding to the k^{th} connected component. Moreover, We have the following proposition:

Proposition 2. Let $G = ([n], E)$ be a connected graph. Then the eigenvalue 0 of L occurs with multiplicity 1 with corresponding eigenvector $\mathbb{1} := (1, ..., 1)^t \in \mathbb{R}^n$.

Proof: Now, suppose that $\mathbf{x}$ is an eigenvector with eigenvalue 0. It follows that

$$\mathbf{x}^t L \mathbf{x} = \frac{1}{2} \sum_{i \in [n]} \sum_{j \in [n]} w_{ij} (x_i - x_j)^2 = 0.$$

Since G is connected, $w_{ij} > 0$ for some $ij \in E$. Since the weights are non-negative, it follows that the latter sum is 0 if and only if $x_i = x_j$ for all $i, j \in [n]$. It follows that $\mathbf{x} \in \text{span}\{\mathbb{1}\}$, which completes the proof. ■

It follows from the previous discussion and Prop. 2 that if G has k connected components, then 0 occurs as an eigenvalue with multiplicity k where the corresponding eigenspace is spanned by

$$(\mathbb{1}_{G_i})_j := \begin{cases} 1, & \text{if } j \in A_i; \\ 0, & \text{otherwise,} \end{cases}$$

where $(\mathbb{1}_{G_i})_j$ is the j^{th} component of $\mathbb{1}_{G_i} \in \mathbb{R}^n$.

This concept will be important in our discussion of spectral clustering. First, we interlude with a brief discussion of k-means clustering.

k-Means Clustering

Suppose you have k finite subsets $G_1, ..., G_k \subset \mathbb{R}^n$. The centroid $\mathbf{c}_i$ of G_i is defined as

$$\mathbf{c}_i := \frac{1}{|G_i|} \sum_{\mathbf{x} \in G_i} x.$$

Suppose you are given m observations $\mathbf{x}_1, ..., \mathbf{x}_m \in \mathbb{R}^n$. We wish to group these points together based on their relative Euclidean distance (i.e. group "close" points together in Euclidean space). Formally, we wish to find clusters $G_1, ..., G_k$ with centroids $\mathbf{c}_1, ..., \mathbf{c}_k$ such that the following quantity is minimized[2]

$$\sum_{i \in [k]} \sum_{\mathbf{x} \in G_i} ||\mathbf{x} - \mathbf{c}_i||^2.$$

The following k-means algorithm is an adaptation from the one given in p. 510 of [2]. The set of "optimal" clusters is determined as follows:

```
Input: $\mathbf{x}_1, ..., \mathbf{x}_m$, $M :=$ max number of iterations, $\epsilon :=$ tolerated error.
  1. Initialize centroids $\mathbf{c}_1, ..., \mathbf{c}_k \in \mathbb{R}^n$ with corresponding clusters $G_1, ..., G_k$.
  2. Repeat:
       a. For $i \in [m]$:
              Set $N_i := \min_j ||\mathbf{x}_i - \mathbf{c}_j||$, and assign $\mathbf{x}_i \to G_{N_i}$.
       b. For $i \in [k]$:
              Let $\mathbf{c}_i'$ to be the mean of the elements in $G_i$, and
              assign $\mathbf{c}_i' \to \mathbf{c}_i$.
       c. Stop if:
              i. Number of iterations $> M$, or
              ii. $||\mathbf{c}_i - \mathbf{c}_i'|| < \epsilon$ for every $i \in [k]$.
Output: $G_1, ..., G_k$.
```

Spectral Clustering

In the following, we begin with a weighted graph $G = (V := [n], E)$ in which the weights w_{ij} are be interpreted as a measure of similarity between vertices i, j. We wish to find an optimal partition $V_1, ..., V_k \subset V$ based on the similarities of the connected vertices. Letting L be the graph Laplacian of G, the following gives a spectral clustering algorithm, as presented in pg. 5 of [3]:

```
Input: $n \times n$ Laplacian matrix $L$, number of clusters $k$ to be constructed.
  1. Compute the first $k$ eigenvectors $\mathtt{v}_1, ..., \mathtt{v}_k$ of $L$.
  2. Let $V$ be the $n \times k$ matrix with $\mathtt{v}_j$ as the $j^{\text{th}}$ column vector.
  3. Let $y_i$ be the $i^{\text{th}}$ row vector of $V$.
  4. Cluster the $y_i$ via the $k$-means algorithm into clusters $C_1, ..., C_k$.
  5. For $i \in [k]$, let $A_i := \{j : y_j \in C_i\}$.
Output: $A_1, ..., A_k$.
```

Intuitively, spectral clustering uses the Laplacian L to embed the given points in $\mathbb{R}^k$ according to connected components of $G_1, ..., G_k$. If G has k connected components, then Prop. 2 implies that the k smallest eigenvalues of the given L will serve as approximations to the 0 eigenvalue of the desired k-connected components[2].This embedding facilitates clustering via the k-means algorithm.

As seen in [4], spectral clustering can be seen as a relaxation in the solution of the multiway-graph equipartition problem of a weighted graph G. In the latter, a partition of $G_1, ..., G_k \subset G$ is desired, such that the following is minimized

$$\text{MECut}(G_1, ..., G_k) := \frac{1}{2} \sum_{j \in [k]} \mathbb{1}_{G_j}^T (D - W) \mathbb{1}_{G_j}$$

where $\mathbb{1}_{G_j}$ is defined in the preliminaries, and

$$D = \begin{pmatrix} D_{G_1} & 0 & \cdots & 0 \\ 0 & D_{G_2} & \cdots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & D_{G_k} \end{pmatrix} \text{ and } W = \begin{pmatrix} W_{G_1,G_1} & W_{G_1,G_2} & \cdots & W_{G_1,G_n} \\ W_{G_2,G_1} & W_{G_2,G_2} & \cdots & W_{G_2,G_n} \\ \vdots & \vdots & & \vdots \\ W_{G_n,G_1} & W_{G_2,G_2} & \cdots & W_{G_n,G_n} \end{pmatrix},$$

where $W_{G_i,G_j} := [W_{ij}]_{i \in V_i, j \in V_j}$, and D_{G_i} is the degree matrix corresponding to the vertices in V_i. Again, following the results in [4], letting X be the matrix with j^{th} column vectors $\mathbb{1}_{G_j} \in \mathbb{R}^n$, the latter problem can be restated in the following manner

$$\operatorname*{argmin}_X \text{trace}(X^t L X) \qquad (1)$$

where the "minimum is achieved when X is taken to be any orthogonal basis for the subspace spanned by the eigenvalues corresponding to the k smallest eigenvalues of L." The spectral relaxation is precisely when L is taken to be the graph Laplacian, and thus it is clear that the given spectral clustering algorithm gives a minimal solution with respect to this relaxation.

Semidefinite Spectral Clustering

Another relaxation for the latter optimization problem is explored in [4], and thoroughly derived in section 4 of [5]. (1) is written as the following quadratic program with corresponding objective function

$$\begin{aligned} &\operatorname*{argmin}_X \ \text{trace}(X^t L X) \\ \text{s.t.} \quad &(X)_{ij}^2 = (X)_{ij} \\ &X e_k = e_n \\ &X^t e_n = m e_k := \frac{n}{k} e_k \end{aligned}$$

where $e_n := [1, ..., 1]^T \in \mathbb{R}^n$. The latter is reduced to the following semidefinite program in ([4], [7]):

$$\begin{aligned} &\underset{S}{\text{argmin}} \;\; \text{trace}(V^t L_e V Y) \\ \text{s.t.} \quad &\text{diag}(VSV^t) = [1, (1/k)e_k]^t \in \mathbb{R}^{nk+1}, \\ &S \text{ is positive semi-definite.} \end{aligned}$$

Here we have that:

$$L_e := \left(\begin{array}{c|c} 0 & \mathbf{0}^T \in \mathbb{R}^{nk} \\ \hline \mathbf{0} \in R^{nk} & I_k \otimes L \end{array}\right), V := \left(\begin{array}{c|c} 1 & \mathbf{0}^T \in \mathbb{R}^{(n-1)(k-1)} \\ \hline \frac{1}{k} e_k \otimes e_n \in \mathbb{R}^{nk} & V_k \otimes V_n \end{array}\right), \; \text{and} V_k := \left(\begin{array}{c} I_{k-1} \\ \hline e_{k-1}^t \end{array}\right).$$

where I_k is the multiplicative identity in $M_{k \times k}(\mathbb{R})$. Interior point methods can be used to solve the latter semidefinite program. The *optimal feasible matrix* is then defined as $Y^* = VS^*V^t$ where S^* is the solution found in the latter SDP. The matrix Z^* is taken to be the sum of the k diagonal $n \times n$ block matrices of $Y^*_{2:nk+1,2:nk+1} \in M_{nk \times nk}(\mathbb{R})$. Let $\lambda_1, ..., \lambda_n \in \mathbb{R}$ be the eigenvalues of Z^* listed in descending order with corresponding eigenvectors $\mathbf{v}_1, ..., \mathbf{v}_n$. Letting $X^* \in M_{n \times k}(\mathbb{R})$ is taken to be the matrix with j^{th} column vector $\mathbf{v}_j$. We can now perform k-means clustering on the rows of X^*, and cluster the n graph vertices as was done in the last step of the spectral clustering algorithm presented in the previous section.

The latter discussion summarizes the semidefinite spectral clustering algorithm given in [4].

2. Assessing Chemical Weights

Recall that given a molecule X, we have a corresponding graph $G(X)$ where each vertex corresponds to some atom or functional group, and each edge corresponds to bonds between these molecular components. The latter section gives methods for decomposing a graph into connected components using weights. This section considers possible choices for our weight function $\omega x : E_{G(X)} \to \mathbb{R}_{\geq 0}$. Our tool in determining weights will be Energy Partition Analysis.

Energy Partition Analysis

Energy partition analysis looks at the interaction energy between two molecular fragments. This can be thought of as the bonding energy between any two fragments. This interaction energy ΔE can be decomposed as

$$\Delta E = \Delta E_{def} + \Delta E_{int}.$$

The first term is the energy required to lift each fragment from its individual equilibrium state to its state in the whole compound. The second term is the energy difference between the fragment in its state within the molecule and the rest of the molecule.

$$\Delta E_{\text{int}} = \Delta E_{\text{electrostatic}} + \Delta E_{\text{Pauli}} + \Delta E_{orb}.$$

The first term can be thought of as the energy due to classical considerations of electric charges amongst the molecular fragments. The second term is due to steric interactions which result from the fact that certain electrons cannot occupy the same regions of space. The final component is the energy gained when each fragment orbital is allowed to relax to an optimal equilibrium state within the molecule. The quantity $-\Delta E$ is otherwise known as the bond dissociation energy between two given fragments ([6], [7]).

The above decompositions of this term give further possible choices for edge weights. However, in practice, given the empirical data available, the natural choice for edge weights in our graphical representation of molecules is the bond dissociation energy ΔE between the respective vertices. The following section applies the latter algorithms where the weights are given as the respective bond dissociation energies.

3. Application of Algorithms

In this section, we apply the spectral clustering algorithm on the graphical representations of explicit molecules using the software `R`. Technical difficulties experienced in the implementation of the semidefinite spectral clustering algorithm in `R` have not been overcome. However, there exists an `R` framework for the SDP solver `CSPD`[8] which has showed promise for implementation.

The following provide the results of the graph partitions given by the standard spectral clustering algorithm for particular examples. The respective weights are taken to be approximate bond dissociation energies for each bond present in the molecule. These energies are taken from [9]. We begin with the molecule presented in the introduction:

Example 1

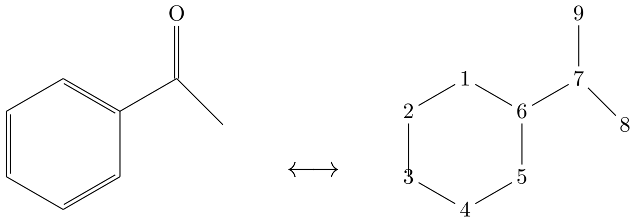

Figure 2

In the latter molecule, the expected decomposition is given by breaking the bond connecting atoms 6 and 7[1]. That is, we expect the partition $\{1,2,3,4,5,6\}, \{7,8,9\}$. We make the following simplifying assumptions given the bond dissociation energies provided in [9]: (1) the double bonds in the benzene ring are weighted as CH2=CH2, (ii) all other bonds between Carbon atoms are weighted as C-C bonds, and (iii) the C=O bond is weighted as is. The weights are given in the table below:

Edge	Associated Weight
$(1,2)$	0.607
$(2,3)$	0.682
$(3,4)$	0.607
$(4,5)$	0.682
$(5,6)$	0.607
$(6,7)$	0.607
$(7,8)$	0.607
$(1,6)$	0.682
$(7,9)$	0.749

The corresponding Laplacian is

$$L = \begin{pmatrix} 1.289 & -0.607 & 0 & 0 & 0 & -0.682 & 0 & 0 & 0 \\ -0.607 & 1.289 & -0.682 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -0.682 & 1.289 & -0.607 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.607 & 1.289 & -0.682 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.682 & 1.289 & -0.607 & 0 & 0 & 0 \\ -0.682 & 0 & 0 & 0 & -0.607 & 1.896 & -0.607 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.607 & 1.963 & -0.607 & -0.749 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.607 & 0.607 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -0.749 & 0 & 0.749 \end{pmatrix}$$

The latter matrix was input into R which resulted in the following clustering vector

```
Vertex 1 Vertex 2 Vertex 3 Vertex 4 Vertex 5
       2        2        2        2        2
Vertex 6 Vertex 7 Vertex 8 Vertex 9
       2        1        1        1
```

This clustering vector corresponds to the partition $\{1,2,3,4,5,6\},\{7,8,9\}$, which was the expected result.

Example 2

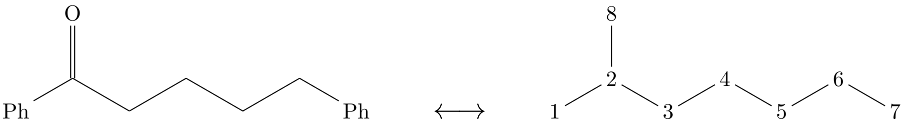

Figure 3

The expected decomposition in this example is given by breaking the bond 34[1]. This corresponds to the partition $\{1,2,3,8\},\{4,5,6,7\} \subset \{1,...,8\}$. We make the following simplifying assumptions (given the bond dissociation energies provided): (1) the bonds between the benzene ring (Ph) and the carbons are weighted as C6H5-CH3 bonds, (ii) the carbons attached to vertex 2 are weighted as C-C bonds, (iii) the remaining bonds between Carbon atoms are weighted as CH3-CH3 bonds, and (iv) the C=O bond is weighted as is. The weights are given in the following table:

Edge	Associated Weight
$(1,2)$	0.389
$(2,3)$	0.607
$(3,4)$	0.368
$(4,5)$	0.368
$(5,6)$	0.368
$(6,7)$	0.389
$(2,8)$	0.749

The corresponding Laplacian is given by

$$L = \begin{pmatrix} 0.389 & -0.389 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.389 & 1.745 & -0.607 & 0 & 0 & 0 & 0 & -0.749 \\ 0 & -0.607 & 0.975 & -0.368 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.368 & 0.736 & -0.368 & 0 & 0 & 0 \\ 0 & 0 & 0 & -0.368 & 0.736 & -0.368 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.368 & 0.757 & -0.389 & 0 \\ 0 & 0 & 0 & 0 & 0 & -0.389 & 0.389 & 0 \\ 0 & -0.749 & 0 & 0 & 0 & 0 & 0 & 0.749 \end{pmatrix}$$

The latter matrix was input into `R` which resulted in the clustering vector:

```
Vertex 1 Vertex 2 Vertex 3 Vertex 4
       2        2        2        2
Vertex 5 Vertex 6 Vertex 7 Vertex 8
       1        1        1        2
```

The latter corresponds to the partition $\{1,2,3,4,8\},\{5,6,7\}$ of V. Note that this differs from the expected result in that atom 4 was inappropriately clustered given the expected partition.

Example 3

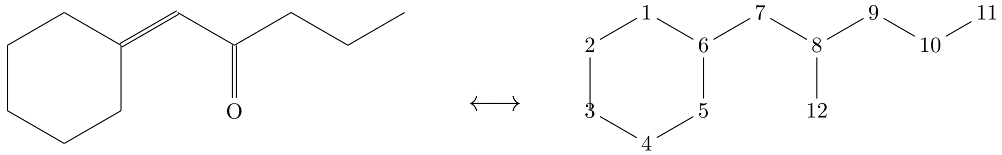

Figure 4

The assigned weights are given in the following table:

Edge	Associated Weight
$(1,2)$	0.368
$(2,3)$	0.368
$(3,4)$	0.368
$(4,5)$	0.368
$(5,6)$	0.607
$(6,7)$	0.682
$(7,8)$	0.607
$(8,9)$	0.607
$(9,10)$	0.368
$(10,11)$	0.368
$(1,6)$	0.607
$(8,12)$	0.749

We made the following simplifying assumptions: (1) bonds appearing with weight 0.368 are weighted as CH3-CH3 bonds; (2) those with weight 0.607 are weighted as C-C bonds, (3) those with weight 0.682 are weighted as CH2=CH2 bonds; and (4) those with weight 0.749, as C=O bonds[9]. The expected decomposition is given by breaking the bond between atoms 6 and 7[10], that is, it is given by the partition $\{1,2,...,6\},\{7,8,...,12\}$ of the molecular graph. Spectral clustering gives the following clustering vector

```
Vertex 1 Vertex 2 Vertex 3 Vertex 4 Vertex 5 Vertex 6
       2        2        2        2        2        2
Vertex 7  Vertex 8  Vertex 9 Vertex 10 Vertex 11 Vertex 12
       2         1         1         1         1         1
```

which corresponds to the partition $\{1, 2, ..., 7\}, \{8, 9, ..., 12\}$. In this example, the vertex 7 was inappropriately clustered given the expected partition.

We do note that in this and the previous example, the vertex placed in the "incorrect" (based on the expected partition) cluster is one of the vertices at which the bond is broken.

Example 4

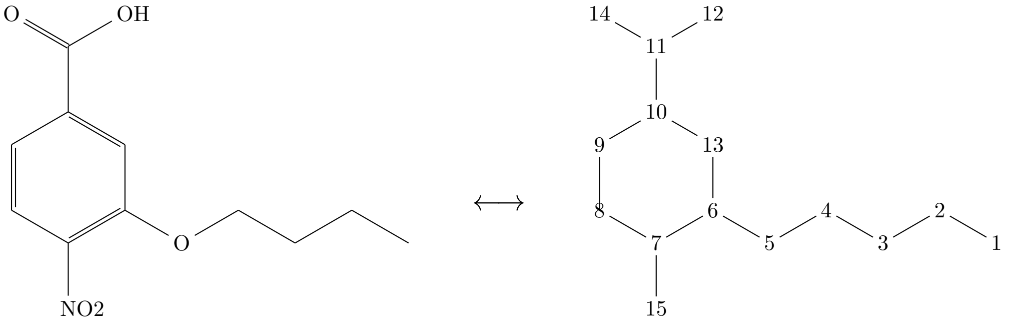

Figure 5

The assigned weights are given in the following table:

Edge	Associated Weight
$(1, 2)$	0.368
$(2, 3)$	0.368
$(3, 4)$	0.368
$(4, 5)$	1.0765
$(5, 6)$	1.0765
$(6, 7)$	0.682
$(6, 13)$	0.607
$(7, 8)$	0.607
$(7, 15)$	0.247
$(8, 9)$	0.682
$(9, 10)$	0.607
$(10, 11)$	0.607
$(10, 13)$	0.682
$(11, 12)$	1.0765
$(11, 14)$	0.749

We made the following simplifying assumptions: (1) bonds appearing with weight 0.368 are weighted as CH3-CH3 bonds; (2) those with weight 0.607 are weighted as C-C bonds, (3) those with weight 0.682 are weighted as CH2=CH2 bonds; (4) those with weight 0.749, as C=O bonds; (5) those with weight 1.0765, as C-O bonds; and (6) those with weight 0.247, as C=O bonds[9].

Here we will perform two rounds of spectral clustering: a partition the molecular graph into k clusters for $k = 2$ and $k = 3$. The expected partition in the former case is given by breaking the bond $(4, 5)$, and the latter by breaking the bonds $(4, 5)$ and $(7, 15)$[10]. These correspond to the respective partitions $\{1, 2, 3, 4\}, \{1, 2, ..., 15\} \backslash \{1, 2, 3, 4\}$ and $\{1, 2, 3, 4\}, \{5, 6, ..., 14\}, \{15\}$ of the presented molecular graph[10].

For $k = 2$, we obtain the following clustering vector:

```
Vertex 1 Vertex 2 Vertex 3 Vertex 4 Vertex 5 Vertex 6 Vertex 7 Vertex 8
       2        2        2        1        1        1        1        1
Vertex 8  Vertex 9 Vertex 10 Vertex 11 Vertex 12 Vertex 13 Vertex 14 Vertex 15
       1         1         1         1         1         1         1         1
```

For $k = 3$, we obtain the following clustering vector:

```
Vertex 1 Vertex 2 Vertex 3 Vertex 4 Vertex 5 Vertex 6 Vertex 7 Vertex 8
       3        3        3        1        1        1        1        1
Vertex 8  Vertex 9 Vertex 10 Vertex 11 Vertex 12 Vertex 13 Vertex 14 Vertex 15
       1         1         2         2         2         1         2         1
```

For $k = 2$, the results are the same as in Examples 2 and 3, in which one of the vertices at which a bond is being broken is "misclustered" (particularly vertex 4). For $k = 3$, we obtain the partition $\{1, 2, 3\}, \{3, 4, ..., 9, 13, 15\}, \{10, 11, 12, 14\}$. This unfortunately did not come close to our expected partition for this molecule.

4. Summary

We saw in the previous section that the spectral clustering algorithm served as a useful tool for giving a bi-partition of a molecular graph. It performed satisfactorily in all four examples. The interesting remark to note is that in each of the latter three examples, the bi-partition "mis-clustered" only one vertex which was one of the vertices at which the bond was expected to break. This observation leads to us accept this method as satisfactory for these molecular bi-partitions. Unfortunately, in the last example when we classified into 3 clusters, the result deviated from our expected molecular decomposition significantly.

Questions in Need of Answers

One of the ideas that was not implemented in this molecular decomposition process is that of introducing catalyst reactions into this molecular decomposition framework. Many of the reactions seen in [1] and [10] involve the use of catalysts. However, often these reactions may change the fundamental structure of the original molecular graph. What is the best way to codify these reactions into our decomposition process?

Also, as stated at the beginning of Section 4, the semidefinite spectral clustering proposed in [4] has not been fully implemented in `R`. It would be interesting to see how the results from the two methods differ in the examples above, if at all. This will be implemented in the near future, again the `R` package `Rcsdp` provides an SDP-solving framework that will aid in its ultimate implementation.

5. References

1. A. Cammidge, *Retrosynthesis Tutorial.*
2. T. Hastie, R. Tibshirani, J. Friedman, *Elements of Statistical Learning 2ed.*
3. U. von Luxburg, *A Tutorial on Spectral Clustering.*
4. J. Kim, S. Choi, *Semidefinite Spectral Clustering*
5. Q. Zhao, *Semidefinite Programming for Assignment and Partitioning Problems*
6. A. Kovacs, C. Esterhuysen, G. Frenking, *The Nature of the Chemical Bond Revisited: An Energy-Partitioning Analysis of Nonpolar Bonds*
7. P. Hunt, *Molecular Orbitals and Population Analysis*
8. B. Borchers, *CSDP 2.3 User's Guide*
9. *Properties of Atoms, Radicals, and Bonds:*
   `http://web.chem.ucsb.edu/ zakariangroup/11---bonddissociationenergy.pdf`
10. G. Rowlan, *123.312 Advanced Organic Chemistry: Retrosynthesis, Tutorial*