Independent Study – Learning Music Structure by Laplacian Formula

Peter Yun-shao Sung*

Abstract. There are many approaches to analyzing music structure by features extracted from dimension of time series. With contruction of similarity matrix, repeated pattern can be captured which is the building block for large-scale structure. This is the work based on the Laplacian Matrix, which is essential start point of spectral clustering. We introduce variables that are trainable to reduce the cost of Laplacian Matrix from true lable, and run this method on wide variable of music recordings. Finally, we demonstrate using these trained variable for performing proper music segmentation.

1. Laplacian formula. Normalized Laplacian matrix is the essential start point for identify music segmentation, and the correct boundary detection is done in my baseline approach (Ref2). For proper boundary detection, we woule like to train the initial laplacian matrix (L) close to true laplacian (L^*) from true inerval annotation from SALAMI dataset. Therefore, to train and update the model, this section is for deriving the L and $\frac{\partial L}{\partial w_{i,j}}$
Given the definition of normalized laplacian matrix:

$$L := I - D^{\frac{1}{2}} W D^{\frac{1}{2}} \tag{1.1}$$

D is degree matrix defined as the diagnal matrix with degrees $d_1, d_2, \ldots, d_n$, which d_i is accumulation of similarity coefficient $w_{i,j}$ betoween time point i and j which defined as followed:

$$d_i = \sum_{j \neq i}^{n} w_{ij} \tag{1.2}$$

After multiplication and the result of equation 1.1 can be rewrite as:

$$L := I - D^{\frac{1}{2}} W D^{\frac{1}{2}} = \begin{pmatrix} 1 & \frac{-w_{12}}{\sqrt{d_1 d_2}} & \cdots & \frac{-w_{1n}}{\sqrt{d_1 d_n}} \\ \frac{-w_{21}}{\sqrt{d_2 d_1}} & 1 & \cdots & \frac{-w_{2n}}{\sqrt{d_2 d_n}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{-w_{n1}}{\sqrt{d_n d_1}} & \frac{-w_{n2}}{\sqrt{d_n d_2}} & \cdots & 1 \end{pmatrix} = \begin{cases} \frac{-w_{i,j}}{\sqrt{d_i d_j}} & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases} \tag{1.3}$$

To take the derivative of L w.r.t $w_{i,j}$. Results is as follow and detail derivation is in appendix 2.1:

$$\frac{\partial L}{\partial w_{i,j}} = \begin{cases} 0 & \text{, if } i = j \\ \frac{-1}{\sqrt{d_i d_j}} + \frac{w_{i,j}(d_i + d_j)}{2(d_i d_i)^{\frac{3}{2}}} & \text{, for position } (i,j),(j,i) \\ \frac{w_{l,k}}{2\sqrt{d_k} d_l^{\frac{3}{2}}} & \text{, for all position } (l,k),(k,l), \text{ where } k \neq i \& j \text{ and } l = i \| j \\ 0 & \text{for any other position} \end{cases} \tag{1.4}$$

The key idea of this derivation is that, when taking derivative of $w_{i,j}$ all elements on the i-th and j-th row and column will be altered because of the degree changes in equation 1.2.

*yss265@nyu.edu

2. Models. We like to build the model minimize the loss J between L and L^*, which defined as followed:

$$J := \frac{1}{2}||L^* - L||_2^2 = \frac{1}{2}\sum_i \sum_j (L^*_{i,j} - L_{i,j})^2 \tag{2.1}$$

With the loss function defined, we would like to design our model with trainable variables θ that minimizing the loss fuction during the update:

$$\theta^{t+1} = \theta^t - \alpha \frac{\partial J}{\partial \theta} \tag{2.2}$$

2.1. Model 1– train on sigma. Here we design our model with trainable varialbe $\sigma_{i,j}$, which is gaussian width to define the similarity coefficient $w_{i,j}$ between features x_i and x_j:

$$w_{i,j} = exp(-(\frac{||x_i - x_j||_2}{\sigma_{i,j}})^2) \tag{2.3}$$

As we like to minimize the loss during each of the $\sigma_{i,j}$ update (eqation 2.2), the derivitive need to expand by chain rule:

$$\frac{\partial J}{\partial \sigma_{i,j}} = sum[(L - L^*) \odot \frac{\partial L}{\partial w_{i,j}}] \cdot \frac{\partial w_{i,j}}{\partial \sigma_{i,j}} \tag{2.4}$$

Where $\odot$ and sum are element-wise multiplication and summation respectively. The idea of this derivation is when there is slight changes on $\sigma_{i,j}$, it directly affcts $w_{i,j}$ and also all elements on i-th and j-th row and column by the degree in equation 1.2. Figure 2.1 is the test result and the relative error between numerical and analytical method are all below 1e-5.

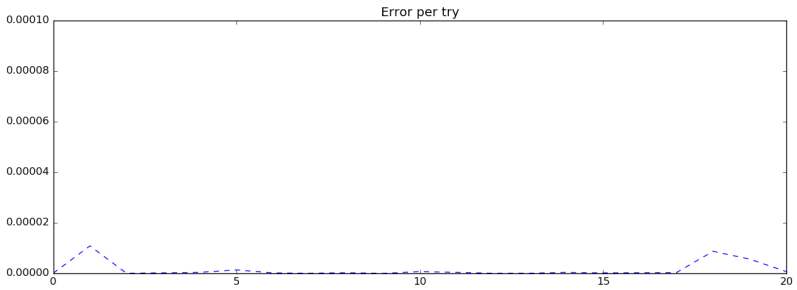

Figure 2.1. *Deriative of loss w.r.t* $\sigma_{i,j}$. *Relative error* $\frac{|f_a - f_n|}{max(|f_a|,|f_n|)}$ *per each try*

Combining equation 2.4 and 2.2, we performed the update for each of $\sigma_{i,j}$. We are training the variables for minimizing loss on laplacian, but since laplacian is normalized version of recurrent matrix that values are hard to visualize, therefore we all showing the result on recurrence matrix. Figure 2.2 is the recurrent matrix after sigma being trained. With boundary detection (Ref2) performed, the detected boundaries are identical. The video of recurrence matrix updates over each steps can is also available in Ref3.

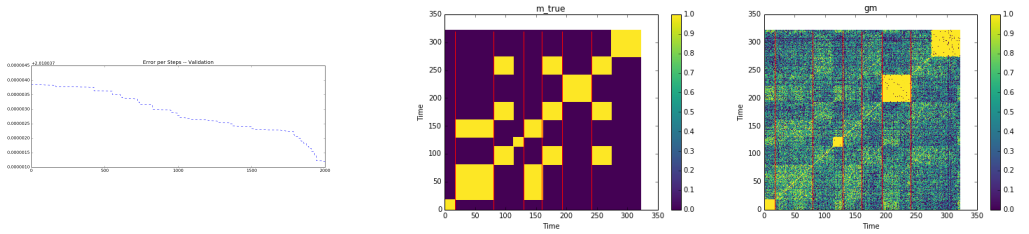

Figure 2.2. *Left: loss per step. Recurrence matrix of Middle: true sigment annotation, and Right: feature with trained sigma*

2.2. Model 2– train on Q. Training on Model 1 is not scalable. First is due to the traing is on $\sigma_{i,j}$, which is the similarity between time point i and j and cannot apply to other song. Second is due to time interval will be different for each songs. Therefore, traing on time scale is not realistic. Here we propos the other method that is trainning on cqt bin and define as follow:

$$\begin{aligned} W_{i,j} &= S_{i,j}^T \cdot Q \cdot S_{i,j} = \Sigma_k (S_{i,j}^k)^2 \cdot q_k \\ \text{which, } S_{i,j} &= cqt[i] - cqt[j] \end{aligned} \tag{2.5}$$

$Q = [q_1, q_2, ..., q_n]$ is diagnoal matrix with the shape of number of cqt bins. Therefore, we can rewrite our objective function to be:

$$\arg\min_Q J(L) = \arg\min_Q \frac{1}{2}||L^* - L||_2^2 \tag{2.6}$$

And the update rule for each of q_i will be:

$$q_k = q_k - \alpha \frac{\partial J}{\partial q_k} \tag{2.7}$$

which:

$$\frac{\partial J}{\partial q_k} = \sum_{i,j} (L_{i,j} - L_{i,j}^*) \frac{\partial L_{i,j}}{\partial q_k} \tag{2.8}$$

changing each q_k is like changing the weight of k-th cqt frequency bin, and therefore this will affect all elements of recurrent matrix, and $w_{i,j}$, d_i, and d_j from equation 1.3 are all the function of q_k:

$$\begin{aligned} \frac{\partial L_{i,j}}{\partial q_k} &= \frac{-1}{\sqrt{d_i d_j}} \frac{\partial w_{i,j}}{\partial q_k} + \frac{w_{i,j}}{2(d_i d_j)^{\frac{3}{2}}} \left(\frac{\partial d_i}{\partial q_k} d_j + d_i \frac{\partial d_j}{\partial q_k}\right) \\ \frac{\partial w_{i,j}}{q_k} &= (s_{i,j}^k)^2 \\ \frac{\partial d_i}{\partial q_k} &= \sum_l (S_{l,i}^k)^2 \end{aligned} \tag{2.9}$$

REFERENCES

[1] A Tutorial on Spectral Clustering
[2] "https://github.com/jfriend08/IS/blob/master/reports/midway/report.pdf"
[3] "https://www.youtube.com/watch?v=XmyKnymXB2Y"

3. Appendix.

3.1. Differential of Laplacian Matrix. If we would like to take derivative of Laplacian w.r.t variable $w_{i,j}$ in the symmetric matrix W. Basically, except for $L_{i,j}$ the components of $L_{i,k}$, $L_{k,i}$, $L_{k,j}$, $L_{j,k}$ will need to consider.
For position $L_{i,j}$:

$$\begin{aligned} L_{i,j} &= L_{j,i} = \frac{-w_{i,j}}{\sqrt{d_i d_j}} \\ \frac{\partial L_{i,j}}{\partial w_{i,j}} &= \frac{\partial L_{j,i}}{\partial w_{i,j}} = \frac{-1}{\sqrt{d_i d_j}} + \frac{w_{i,j}}{2(d_i d_i)^{\frac{3}{2}}}(\frac{\partial d_i}{\partial w_{i,j}} d_j + d_i \frac{\partial d_j}{\partial w_{i,j}}) \\ &= \frac{-1}{\sqrt{d_i d_j}} + \frac{w_{i,j}(d_i + d_j)}{2(d_i d_i)^{\frac{3}{2}}} \end{aligned} \tag{3.1}$$

For position $L_{l,k}$, $L_{k,l}$, where $k \neq i\&j$ and $l = i \| j$:

$$\begin{aligned} L_{k,l} &= L_{l,k} = \frac{-w_{k,l}}{\sqrt{d_k d_l}} \\ \frac{\partial L_{k,l}}{\partial w_{i,j}} &= \frac{\partial L_{k,l}}{\partial w_{i,j}} = \frac{w_{k,l}}{2\sqrt{d_k} d_l^{\frac{3}{2}}} \end{aligned} \tag{3.2}$$