Tree-based up-down Algorithm

Let D = (V,E) be the complete, undirected, weighted graph, such that each vertex corresponds to a column of the matrix D, and the weight of an edge equals the coherence between the two columns it connects.

Given a cut $X(E_x)$ in this graph splitting D into subgraphs $A(V_a,E_a)$ and $B(V_b,E_b)$, we define the quantities

$$\mu_a = \max \{ w(e) \mid e \in E_a \}$$

$$\mu_b = \max \{ w(e) \mid e \in E_b \}$$

$$\mu_x = \max \{ w(e) \mid e \in E_x \}$$

$$\mu_d = \max \{ w(e) \mid e \in E \} = \max \{\mu_a, \mu_b, \mu_m\}$$

Note: μ_d is defined for the graph D itself, but μ_a and μ_b require the cut to also be defined.

Also, for any graph G, let μ^*_G be the 'effective coherence'. That is, the minimum of the coherence achieved using any possible splitting (or sequence of splits) for this graph. This also includes the case where no splitting is done (in which case, $\mu^*_G = \mu_d$).

Algorithm

Define a subroutine MaxSplit as:

<u>MaxSplit(G(V,E)):</u>

```
• Q = Sorted list of edges of G in decreasing order of weight
• Initialize cut X = empty set
• While Q is not empty
    o $e_i$ = Pop the next edge from Q
    o Let X' = X U $e_i$
    o If X' is can be extended to a valid cut in the graph:
        ▪ X = X'
• EndWhile
  (Note: The while loop does not end when we are not able to add an edge to X. We only stop after traversing ALL of the edges in G)
• $A(V_a,E_a)$, $B(V_b,E_b)$ = Subgraphs of G formed by cut X
• Return A, B, X
```

Next, we need a subroutine to make a tree corresponding to a given graph

<u>MakeSubTree(G(V,E)):</u>

```
• T = empty Tree (each node of T is structure as {childA, childB, corresponding $\mu^*$ value})
• Initialize T with root node as {V, max( w(e) | e ∈ E )}
  {list of all vertices in G, maximum weight edge in G}
• $A(V_a,E_a)$, $B(V_b,E_b)$, $X(V_x,E_x)$ = MaxSplit(G)
• If size($V_a$) == 1:
    o Ta = {null, null, 0}
• Else:
    o Ta = MakeSubTree(A)    #recursive call

• If size($V_b$) == 1:
    o Tb = {null, null,0}
• Else:
    o Tb = MakeSubTree(B)    #recursive call

• c = max(w(e) | e ∈ $E_x$)           # max crossing edge
• $\mu'$ = max($\mu_a$*, $\mu_b$*)    #These are obtained from the recursive call
• $\gamma = \frac{\mu' + \sqrt{\mu'^2 + c^2}}{2}$
• $\mu$* = min($c$, $\gamma$)
• T = *Ta,*Tb, $\mu$*                 #  *Ta is a pointer to Ta
```

Main (G(V,E)):

```
    T = MakeSubTree(G)
    Output T
```

The μ* value at each node is automatically calculated in the process. The value at the root node of the output is the μ* value for the graph G.