Advance Algorithm in Bioinformatics – CSC 8540 – Fall 2022

October 11 2022

Pardis Sadatian Moghaddam

Panther ID: 002722641

Is the following matrix additive? If not, give a reason. If yes, give the additive tree.

	S1=a	S2=b	S3=c	S4=d
S1=a	0	3	8	7
S2=b		0	7	6
S3=c			0	5
S4=d				0

When M has at least 4 taxa, M many not be additive.

M is additive if and only if the 4-point condition is satisfied and symmetric, that is, for any four taxa in S, we can label them as a,b,c,d such that Mac + Mbd = Mad + Mbc >= Mab + Mcd

Let's choose S1, S2, S3 and S4 as the four taxa.

Mac + Mbd = 8 + 6 = 14

Mab + Mcd = 3 + 5 = 8

Mad + Mbc = 7 + 7 = 14

Mac + Mbd = Mad + Mbc >= Mab + Mcd => 14=14 >= 8

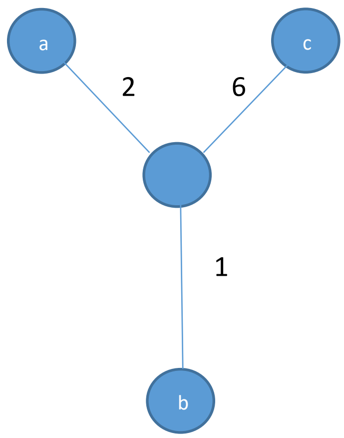
This matrix is additive. So, here is the additive tree:

In here we have AC = 8

We try to connect b, So we have:

(Dab + Dcb – Dac)/2 = (3 + 7 – 8)/2 = 1

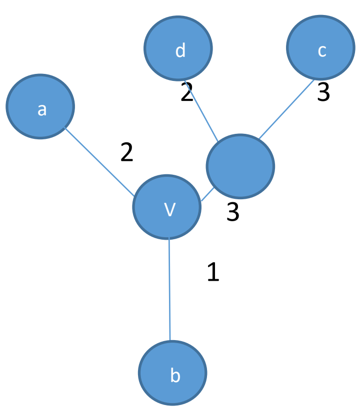
So we have this:



Next step we need to add d:

We connect d between a and c

(Dad + Dcd – Dac)/2 = (7 + 5 – 8)/2 = 2



2. Construct an additive tree for the following distance matrix.

	A	B	C	D	E
A	0	10	9	16	8
B		0	15	22	8
C			0	13	13
D				0	20
E					0

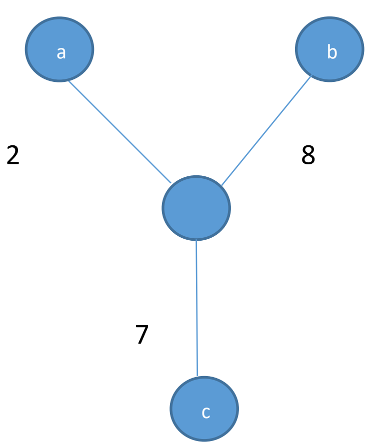
dic =(Mij + Mik – Mjk)/2

djc = (Mij + Mjk - Mik)/2

dkc = (Mik + Mjk – Mij)/2

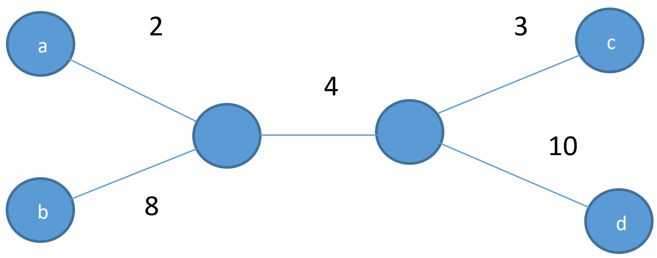
We try to connect c, So we have:

(Dac + Dbc – Dab)/2 = (9 + 15 – 10)/2 = 7



The next step is trying to connect b, We try to connect it between b and c:

(Dbd + Dcd – Dbc)/2 = (22 + 13 – 15)/2 = 10

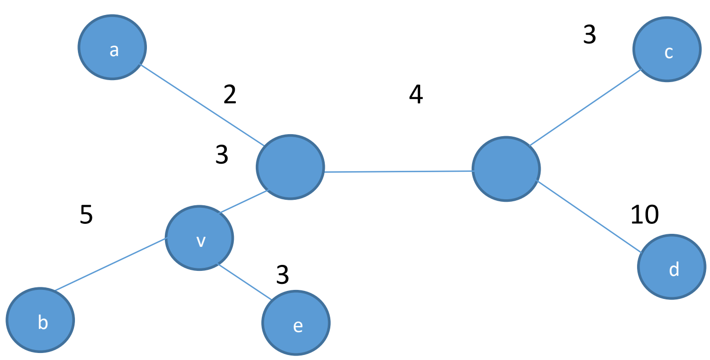


The last step is trying to connect E, we try to connect it:

Between B and C:

Dve = (Dbe + Dce – Dbc)/2 = (8 + 13 – 15)/2 = 3

Dbv = Dbe – Dve = 8 – 3 = 5



20. For the following matrix M, can you construct the nearly additive tree using neighbor-joining?

m	S1	S2	S3	S4	S5
S1	0	7	11	13	15
S2	7	0	12	14	18
S3	11	12	0	8	10
S4	13	14	8	0	5
S5	15	18	10	5	0

Round 1

Step 1

Compute r'i for each terminal node:

r'S1= (7 + 11 + 13 + 15)/3 = 15.5

r'S2= (7 + 12 + 14 + 18)/3 = 17

r'S3= (11 + 12 + 8 + 10)/3 = 13.6

r'S4= (13 + 14 + 8 + 5)/3 = 13.3

r'S5= (15 + 18 + 10 + 5)/3 = 16

m	S1	S2	S3	S4	S5	r'i
S1	0	7	11	13	15	15.5
S2		0	12	14	18	17
S3			0	8	10	13.6
S4				0	5	13.3
S5					0	16

Step 2

d'i,j = di,j - r'i - r'j

ds1s2 =-25.3

ds1s3 = -17.9

ds1s4 = -15.6

ds1s5 = -16.3

ds2s3 = -18.6

ds2s4 = -16.3

ds2s5 = -15

ds3s4 = -18.9

ds3s5 = -19.6

ds4s5 = -24

Here we have:

m	S1	S2	S3	S4	S5
S1	0	-25.3	-17.9	-15.6	-16.3
S2		0	-18.6	-16.3	-15
S3			0	-18.9	-19.6
S4				0	-24.3
S5					0

S1 and S2 are neighbors:

Step3

Calculate branch lengths:

Vi = 0.5 (di,j) + 0.5(r'i - r'j) = 2.65

Vj = 0.5 (di,j) + 0.5(r'j - r'j) = 4.35

m	S1S2	S3	S4	S5
S1S2	0	8	10	13
S3	8	0	8	10
S4	10	8	0	5
S5	13	10	5	0

Ds1s2 , s3 = (ds1s3 + ds2s3 – ds1s2)/2 = (11 + 12 - 7)/2 = 8

Ds1s2 , s4 = (ds1s4 + ds2s4 – ds1s2)/2 = (13 + 14 - 7)/2 = 10

Ds1s2 , s5 = (ds1s5 + ds2s5 – ds1s2)/2 = (15 + 18 - 7)/2 = 13

Round 2

Step 1

Compute r’i for each terminal node:

r’S1S2= (8 + 10 + 13)/2 = 15.5

r’S3= (8 + 8 + 10)/2 = 13

r’S4= (10 + 8 + 5)/2 = 11.5

r’S5= (13 + 10 + 5)/2 = 14

m	S1S2	S3	S4	S5	r’i
S1S2	0	8	10	13	15.5
S3		0	8	10	13
S4			0	5	11.5
S5				0	14

Step 2

d’i,j = di,j - r’i - r’j

ds1s2,S3 = -20.5

ds1s2,S4 = -17

ds1s2,S5 = -16.5

ds3s4= -16.5

ds3s5= -17

ds4s5= -20.5

m	S1S2	S3	S4	S5
S1S2	0	-20.5	-17	-16.5
S3		0	-16.5	-17
S4			0	-20.5
S5				0

S4 and S5 are neighbors:

Step3

Calculate branch lengths:

Vi = 0.5 (di,j) + 0.5(r'i - r'j) = 1.25

Vj = 0.5 (di,j) + 0.5(r'i - r'j) = 3.75

m	S1S2	S4S5	S3
S1S2	0	9	8
S4S5		0	7.5
S3			0

Ds1s2 , s4s5 = (ds4, s1s2 + ds5,s1s2 – ds4s5)/2 = (10 + 13 – 5)/2 = 9

Ds1s2, s3 = (ds1,s3 + ds2,s3 – ds1s2)/2 = (11 +12 – 7)/2 = 8

Ds4s5 , s3 = (ds4s3 + ds5ds3 – ds4s5)/2 = (8 + 10 -5)/2 = 6.5

Round 3

Step 1

Compute r'i for each terminal node:

r'S1S2 = 17

r'S3S4 = 16.5

r'S5 = 20

m	S1S2	S4S5	S3	r'i
S1S2	0	9	8	17
S4S5		0	6.5	15.5
S3			0	14.5

Step 2

d’i,j = di,j - r’i - r’j

ds1s2,s4s4 = 9 – 17 – 15.5 = -23.5

ds1s2,s5 = 8 – 17 – 14.5 = -23.5

ds3s4,s5 = 6.5 – 12 – 20 = -23.5

m	S1S2	S4S5	S3
S1S2	0	-23.5	-23.5
S4S5		0	-23.5
S3			0

S1S2 and S3S4 are neighbors:

Step3

Calculate branch lengths:

Vi = 0.5 (di,j) + 0.5(r’i - r’j) = 5.25

Vj = 0.5 (di,j) + 0.5(r’j - r’i) = 3.75

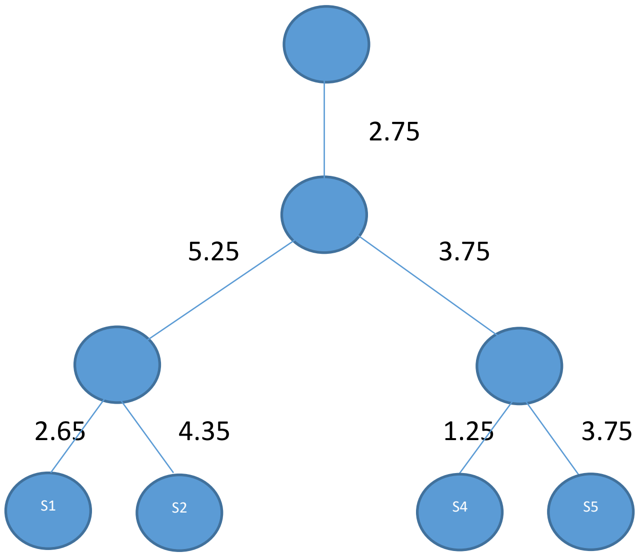
Round 4

m	S1S2S4S5	S3
S1S2S4S5	0	5.5
S3	5.5	0

DS1S2S3S4 , s3 = (ds1s2, s3 + ds4s5,s3 – ds1s2,s4s5)/2 = (8 + 6.5 - 9)/2 = 2.75

Vi=Vj = 2.75

Here is the nearly additive tree



4. Given an additive tree T for n species, describe an efficient algorithm to compute the additive distance matrix for T. What is the time complexity?

We can try the UPGMA algorithm. (Unweighted Pair group Method with Arithmetic mean)

When considering clusters S and T, the UPGMA is equal to the average of the distances taken over all pairs of individual elements $s \in S$ and $t \in T$.

$D(S,T) = (\sum_{s \in S, t \in T} d(s,t)) / |S| \cdot |T|$

We also have updated distance S U T ao we have new cluster A. We have this average distance:

$d(SUT), A = |S| \cdot dS,A + |T|.dT,A / |S| + |T|$

We can consider the height of the tree to every vertex. We can calculating the maximum likelihood of finding the neighbor node in a tree.

The time and space complexity of UPGMA is O(n^2), since there are n-1 iterations, with O(n) work in each one.

The second way to compute the additive tree is DFS algorithm. We can go through the tree. V is the vertexes and E is egdes. **The time complexity for this is O(|V| + |E|)**.