REPORTS@SCM

AN ELECTRONIC JOURNAL OF THE
SOCIETAT CATALANA DE MATEMÀTIQUES

Some improvements to the Erik+2 method for unbalanced partitions

Óscar Rivero Salgado
Universitat Politècnica de Catalunya
rversal@hotmail.com

*__Pol Torrent i Soler__
Universitat Politècnica de Catalunya
ptorrent@me.com

∗Corresponding author

Resum *(CAT)*

En aquest article proposem millores al mètode Erik+2, que s'empra per obtenir la distribució de les espècies a les fulles d'un arbre filogenètic, suggerint solucions als problemes provocats per la falta de dades experimentals quan es tracta amb un nombre elevat d'espècies. Presentem una nova tècnica per calcular les puntuacions assignades a cada distribució que es basa en aplicar successivament el mètode Erik+2 tenint en compte les files i columnes més plenes de la matriu de dades observades i compensant les puntuacions obtingudes per files i per columnes segons les dimensions de la matriu. Segons aquestes dimensions també proposem normalitzacions de les puntuacions obtingudes.

Abstract *(ENG)*

We aim to improve the Erik+2 method for obtaining the right distributions at the leaves of a phylogenetic tree, by addressing the problems that are due to the lack of enough experimental data when dealing with a high number of species. We introduce a new procedure based on successive applications of the Erik+2 method to take into account the most filled rows and columns of the observed data matrix and on balancing the scores obtained from both rows and columns. We also propose normalizations to compare the scores based on the dimensions of the data matrix.

Keywords: *Phylogenetics, Flattening matrix, Erik+2 Method.*
MSC (2010): *92D15, 60J20.*
Received: *September 17th, 2015.*
Accepted: *October 29th, 2015.*

Acknowledgement

The authors would like to thank Marta Casanellas for her vital guide and support while working on this project.

1. Introduction

Phylogenetics is a classical branch of science whose main aim is to determine evolutionary relationships between species. We typically have DNA sequences from genes of the different species we are studying and the classical approach would be to perform some kind of statistical analysis to determine the tree that fits the best to our data. However, in recent years, the use of tools from algebraic geometry have let to obtain a great progress in this field: we could talk about a new branch, phylogenetic algebraic geometry, that would study algebraic varieties representing statistical models of evolution, mixing that way mathematics, statistics, biology and computation. We take as our starting point the approach of Nicholas Eriksson and others, that uses the computation of the singular value decomposition of a matrix to study the distance to a particular algebraic variety. In recent years, Marta Casanellas and Jesús Fernández-Sánchez developed an improved version, Erik+2, that leaded to better results in the case of four species. Now, we try to extend their idea to the case of more species (here we work with the case of 12), having the necessity of doing some ponderations during the process concerning the size of the submatrices to obtain a result that, and even though our result is not optimal, provides some good approaches.

2. Background

The evolution of species is usually modeled in a phylogenetic tree $\mathcal{T}$. The leaves of the tree represent current species and the root the common ancestor. The aim of phylogenetics is to determine the phylogenetic tree of a set of species from the DNA sequences of current species. Due to its structure, we can deal with DNA sequences as if they were a sequence of nucleotides (A, C, G, T). For this reason, we need a statistical model for the substitutions of nucleotides to face our problem. We will work under the following assumptions:

(i) the trees are binary (which means that two branches come out of the root, if it exists, and that they are divided into another two branches in each node);

(ii) the processes in each branch do only depend on the common father node;

(iii) mutations of the DNA chain occur randomly;

(iv) each position of the DNA sequence evolves independently and under the same mutation probabilities; this means it is enough to model one position of the chain.

Following these assumptions we can think the nucleotide mutation process as a Markov process by assigning to each edge e a transition matrix

$$S_e = \begin{pmatrix} \mathrm{P}(\mathtt{A}\,|\,\mathtt{A},e) & \mathrm{P}(\mathtt{C}\,|\,\mathtt{A},e) & \mathrm{P}(\mathtt{G}\,|\,\mathtt{A},e) & \mathrm{P}(\mathtt{T}\,|\,\mathtt{A},e) \\ \mathrm{P}(\mathtt{A}\,|\,\mathtt{C},e) & \mathrm{P}(\mathtt{C}\,|\,\mathtt{C},e) & \mathrm{P}(\mathtt{G}\,|\,\mathtt{C},e) & \mathrm{P}(\mathtt{T}\,|\,\mathtt{C},e) \\ \mathrm{P}(\mathtt{A}\,|\,\mathtt{G},e) & \mathrm{P}(\mathtt{C}\,|\,\mathtt{G},e) & \mathrm{P}(\mathtt{G}\,|\,\mathtt{G},e) & \mathrm{P}(\mathtt{T}\,|\,\mathtt{G},e) \\ \mathrm{P}(\mathtt{A}\,|\,\mathtt{T},e) & \mathrm{P}(\mathtt{C}\,|\,\mathtt{T},e) & \mathrm{P}(\mathtt{G}\,|\,\mathtt{T},e) & \mathrm{P}(\mathtt{T}\,|\,\mathtt{T},e) \end{pmatrix},$$

where $\mathrm{P}(\mathtt{I}\,|\,\mathtt{J},e)$ is the probability of the nucleotide in the father node J becoming I after the edge e. These entries are unknown and along with the distribution in the root $\pi = (\pi_{\mathtt{A}}, \pi_{\mathtt{C}}, \pi_{\mathtt{G}}, \pi_{\mathtt{T}})$ are the parameters of

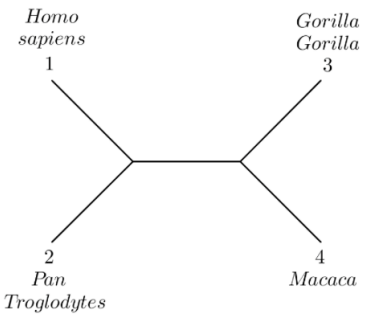

Figure 1: A example of an unrooted 4-leaf phylogenetic tree.

our model. By imposing conditions on the matrix S_e, one obtains different models. We deal with the most general Markov model; see [4].

We define now the random variables X_i as the state of the leaf i for $i \in \{1, \dots, n\}$ so that X_i takes values in $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}\} = \mathcal{K}$, where n is the number of leaves of the tree. Now let $p_{x_1 x_2 \cdots x_n} = \mathrm{P}(X_1 = x_1, \dots, X_n = x_n)$ be the joint distribution at the leaves of the tree. Those probabilities can be calculated using only the entries of the transition matrices.

We are now ready to state the main definition and the main theorem we will need to understand Erik+2 method.

Definition 2.1. Let $A|B$ be a partition of the leaves (that is, if $L(\mathcal{T})$ is the set of leaves of the rooted tree $\mathcal{T}$ then $L(\mathcal{T}) = A \cup B$ and $A \cap B = \varnothing$), where we also assume that A and B are ordered sets. Then we define the *flattening matrix* $\mathrm{flat}_{A|B}$ of a joint distribution vector p associated to the partition $A|B$ as the $4^{|A|} \times 4^{|B|}$ matrix

$$\mathrm{flat}_{A|B}(p) = \begin{pmatrix} p_{\mathtt{AA}\cdots\mathtt{AA}} & p_{\mathtt{AA}\cdots\mathtt{AC}} & p_{\mathtt{AA}\cdots\mathtt{AG}} & \cdots & p_{\mathtt{AA}\cdots\mathtt{TT}} \\ p_{\mathtt{AC}\cdots\mathtt{AA}} & p_{\mathtt{AC}\cdots\mathtt{AC}} & p_{\mathtt{AC}\cdots\mathtt{AG}} & \cdots & p_{\mathtt{AC}\cdots\mathtt{TT}} \\ p_{\mathtt{AG}\cdots\mathtt{AA}} & p_{\mathtt{AG}\cdots\mathtt{AC}} & p_{\mathtt{AG}\cdots\mathtt{AG}} & \cdots & p_{\mathtt{AG}\cdots\mathtt{TT}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_{\mathtt{TT}\cdots\mathtt{AA}} & p_{\mathtt{TT}\cdots\mathtt{AC}} & p_{\mathtt{TT}\cdots\mathtt{AG}} & \cdots & p_{\mathtt{TT}\cdots\mathtt{TT}} \end{pmatrix}.$$

That is, each column of the flattening matrix corresponds to a state of the leaves in B and each row to a state of the leaves in A. We will call such a partition an *edge split* if we can remove an edge such that all the leaves in A are in the same connected component and all the leaves in B are in the other one, and we will refer as the *size of the partition* to the pair $(|A|, |B|)$ (though we will usually write it as $|A| \times |B|$).

For instance, in the previous example 12|34 is an edge split partition, while 13|24 is not. Now we are ready to state the following result.

Theorem 2.2 ([1, 2])**.** *Let $A|B$ be a partition of the set of leaves of the tree $\mathcal{T}$ and let p be the joint distribution at the leaves of $\mathcal{T}$ for certain parameters. If that partition is an edge split, then* $\mathrm{rank}\ \mathrm{flat}_{A|B}(p) \leq 4$, *whereas if it is not an edge split partition and the parameters are general enough and $|A|, |B| > 1$, then* $\mathrm{rank}\ \mathrm{flat}_{A|B}(p) > 4$. □

For the case with $n = 4$ species at the leaves if the parameters are "general enough", one can show that the rank of the flattening matrix for partitions which are not an edge split is maximum (i.e., 16) but, since we will be dealing with cases with $n = 12$, we cannot assume this as true (cf., [1]).

2.1 The Erik+2 method

We start off with a set of ordered nucleotide sequences (one for each leaf in our tree, as they are the observed DNA chains of current species) which we will assume that have no gaps and have all the same length. We think of this set of nucleotide sequences as an *alignment*, that is, nucleotides at the same position of the different sequences are supposed to have evolved from the same nucleotide of the common ancestor.

From this experimental data we can calculate the relative frequencies $\tilde{p}_{x_1x_2\cdots x_n}$, which we will use as estimators for the true probabilities $p_{x_1x_2\cdots x_n}$ (in fact it can be shown that those are the maximum likelihood estimators for the true probabilities, see [3]). Given a partition of the leaves $A|B$, we can build the estimated flattening matrix $\widetilde{\text{flat}}_{A|B}$ just like we did above, but this time using the relative frequencies instead of the true probabilities. We aim to determine the right topology of the tree (i.e., to determine which species is at each leaf) by studying which partitions of the leaves are an edge split according to the experimental data and which are not.

By the theorem we stated above, if that matrix was exactly the flattening matrix we should be able to distinguish between edge splits and the other ones because edge splits would have exactly rank 4 or less and the other ones would not. This could be done easily by checking whether all 5×5 minors vanish or not, but since we only have the estimated matrices we have to develop a method to decide which one is "closer" to rank 4 matrices and to do so we will take the distance induced by the Frobenius norm.

Lemma 2.3. *If M is an $m \times n$ matrix and $\{\sigma_i\}$ are its singular values (ordered from big to small), the Frobenius distance of M to $\mathcal{V}$ (the set of rank 4 or lower matrices) in the Frobenius norm is*

$$d(M, \mathcal{V}) = \sum_{i=5}^{\min\{m,n\}} \sigma_i^2.$$

□

The ErikSVD method (see [1]) uses this fact to give a score to each flattening matrix. Indeed, it works as follows: given an alignment and a partition $A|B$, it computes the estimated flattening matrix and then it obtains the singular value decomposition of the matrix and computes the distance $d(\widetilde{\text{flat}}_{A|B}, \mathcal{V})$ which is the score assigned to the partition. Hence the partition which is estimated to be an edge split is the one having the lowest score.

The Erik+2 method (see [3]) slightly modifies the previous procedure by taking into account that the rank of the flattening matrix could be affected by the presence of long-branch attraction situations. The solution given by the Erik+2 method is to normalize first rows and then columns so each one sums up to 1. Scores obtained after normalizing by both rows and columns are taken into account to compute the final score.

One has to take into account that if we are dealing with a case with $n = 4$ then the flattening matrices for 2×2 partitions will have dimension 16×16. But in our case we used the algorithm to treat cases with 12 species, which leads to flattening matrices with dimensions $4^2 \times 4^{10}$ for 2×10 (actually the dimensions of the matrix we were dealing with computationally were about 16×60000 since we were only taking into account nonempty rows and columns) and $4^5 \times 4^7$ for 5×7 partitions. This explains why alignments with size 100000 work fine with 4 species but often are not enough to fill bigger flattening matrices so as to give a closer approach to the theoretical situation.

Since the number of singular values depends on the dimensions of the matrix and these dimensions depend on the cardinal of the subsets that form the partition, another interesting point is to ensure that

we can compare scores obtained from partitions whose subsets have different cardinals (and hence their flattening matrices have different dimensions).

3. Our proposed modifications

In this section we describe some of the most successful modifications out of the ones we tried. We start off with the observation that for the 2×10 sized partitions the flattening matrices have lots of columns which contain a single element due to the lack of data and that this fact can easily alter the rank of the matrix. Since the theoretical model stated that we should be dealing with matrices of rank approximately 4, we conjectured that there should be an important amount of data in a few rows and columns.

First of all we looked at how data should be distributed if the alignment was completely random (in this paper we will always assume that a random alignment is an alignment such that the distribution of its columns is uniform) to compare it to the actual flattening matrices. The following lemmas will allow us to make those estimations.

Lemma 3.1. *In a randomly generated alignment of length n, the expected number of nonempty columns of a flattening matrix of c columns is*

$$a_n = -c\left(\frac{c-1}{c}\right)^n + c.$$

Proof. We can easily build a recurrence by noticing that, when we have an alignment of length i, then a_{i+1} is simply the probability of the new datum being on an already occupied column times the current number of occupied columns plus the probability of it being on an empty column times the current number of occupied columns plus one. Noticing that the number of currently occupied columns is a_i (so a_{i+1} can only take the values a_i and $a_i + 1$ each one with its probability) we can write

$$a_{i+1} = \mathrm{P}(\text{new datum is in occupied column}) \cdot a_i + \mathrm{P}(\text{new datum not in occupied column}) \cdot (a_i + 1) =$$

$$= \frac{c - a_i}{c}(a_i + 1) + \frac{a_i}{c}a_i.$$

Simplifying, one obtains $ca_{i+1} - (c-1)a_i = c$. Then, we just need to resolve the recurrence. Putting it in an homogeneous form we obtain $ca_{i+2} - (2c-1)a_{i+1} + (c-1)a_i = 0$ so, the characteristic polynomial has roots 1 and $(c-1)/c$ and we get a solution of the form

$$a_n = \alpha\left(\frac{c-1}{c}\right)^n + \beta.$$

By setting initial conditions one obtains the result. □

Lemma 3.2. *In a randomly generated alignment of length n, the expected number of columns with a single matrix of a flattening matrix of c columns is*

$$b_n = a_n\left(\frac{a_n - 1}{a_n}\right)^{n-a_n},$$

where a_n is defined as in the previous lemma.

Proof. We have that a_n columns are occupied so we can focus in the case where each one of them has a single entry and that we have $n - a_n$ data left to distribute. Since having a single entry is now equivalent to not getting any of those remaining data, we can apply the previous lemma with $n = n - a_n$ and $c = a_n$, so the number of occupied columns is now

$$X = -a_n \left(\frac{a_n - 1}{a_n} \right)^{n-a_n} + a_n,$$

and the number of not occupied (and hence with a single entry) columns $a_n - X$ is the one stated above. □

Assuming alignments of size 10^5 as the ones we had, we obtained, for instance, that for the 2×10 partition there would be on average 95380 nonempty columns, where 90869 of them have only one entry. The actual matrices have about 60000 nonempty columns, 40000 of them having a single entry, hence dispersion is lower than in the random model but not much lower. For 5×7 partitions, we observed that random matrices have entries in almost all columns (we computed an average of 16347 nonempty columns out of $4^7 = 16384$ possible, and we expected that just 98 columns had one entry). In this case we observed that, on average, we had 9000 nonempty columns so dispersion was also lower than in the random case. This data is obtained from the following lemmas and completed in table 1.

Partition	Number of columns	Expected nonempty	Expected single entry
2 vs 10	$4^{10} = 1048576$	95380	90869
3 vs 9	$4^9 = 262144$	83137	67874
4 vs 8	$4^8 = 65536$	51287	19837
5 vs 7	$4^7 = 16384$	16347	98

Table 1: Expected number of nonempty columns and columns with a single entry assuming alignments of length $n = 10^5$.

We can also use recurrences to estimate the number of entries in the most populated rows. To normalize, we will need to look at the number of entries at the most populated half according to our proposed method that will be explained below (since the most populated half has a greater weight in the final score). Taking into account that half, we will look at how many entries we have in the most populated sub-half, and so on (this works since the number of rows is always a power of 2). We will treat the problem of determining the number of entries in the most populated half as the problem of looking for the expected cardinality of the most populated half (tails or heads) when we toss n times a coin (this is equivalent to our problem since the data distribution is uniform). Let c_n be that number. Clearly $c_1 = 1$ and $c_n = c_{n-1} + 1/2$ if n is even (since the new coin will result in the result which is currently most frequent with probability $1/2$), and if n is odd we see that c_n is $1/2$ plus the previous number of coins in the most populated half, as before, but we have to take into account the existence of draws by adding an extra term that takes care of this probability, resulting in

$$c_n = c_{n-1} + \frac{1}{2} + \frac{1}{2} \frac{\binom{n-1}{(n-1)/2}}{2^{n-1}},$$

for odd n. By Stirling's approximation we get

$$c_n \approx c_{n-1} + \frac{1}{2} + \frac{1}{\sqrt{2\pi(n-1)}}$$

so, for n big enough, by adding up both results we get

$$c_n \approx \frac{n+1}{2} + \frac{1}{2\sqrt{\pi}} \left(\sum_{i=1}^{(n-1)/2} \frac{1}{\sqrt{i}} \right).$$

We also looked with detail to some cases and found out the following patterns for flattening matrices coming from an edge split. They usually (respect to flattening matrices not coming from a partition which is an edge split) have a lower amount of nonempty rows and columns, have less rows and columns with only 1 entry, and have more entries in the most populated rows.

This led us to think that it would be convenient to reorder rows and columns according to their number of entries, in order to have the most populated (and hence most significant) rows and columns in the first place. Then we consider the sub-matrices obtained by taking the m rows and the first k columns, where m is the number of rows of the matrix and k is a parameter of the method (we used $k = 1000$). We apply the Erik+2 method to those sub-matrices and then we extend the sub-matrix with k more columns, compute the score again and so on, and finally we add up all the scores. In order to compare the scores between partitions of different size, it is convenient to divide the score by the number of total SVDs done. However, when dealing with partitions of the same size, this does not help since it would decrease the score for wrong matrices which usually have more nonempty columns.

We also considered to do an analogous procedure for both rows and columns, i.e., considering sub-matrices of size $k_1 \times k_2$, and then increase both k_1 and k_2 but, since we are usually dealing with matrices which have $m \ll n$, we did not see a significant improvement of the results. Due to this fact we also need to multiply by m the score obtained by normalizing the columns, and by n the score obtained by normalizing the rows, in order to have the same order of magnitude.

Since we are adding up scores of matrices with different dimensions, the next step is to give estimates for the value of those scores so we can normalize. If we have an $m \times n$ matrix and we normalize the rows so as the elements of each row sum up to 1, we get

$$\sqrt{\frac{\sum\sum a_{ij}^2}{mn}} \geq \frac{\sum\sum a_{ij}}{mn} = \frac{m}{mn} = \frac{1}{n}$$

since each row adds up to one (we assume that in each row there is at least one entry since the method does only take into account nonempty rows). We obtain

$$\sqrt{\sum\sum a_{ij}^2} \geq \sqrt{\frac{m}{n}} \Longrightarrow n\sqrt{\sum\sum a_{ij}^2} \geq \sqrt{mn}$$

and, by symmetry, we obtain the same result when we normalize columns and multiply by m. To get an upper bound notice that, since $(\sum b_i)^2 = 1$ (where the b_i are elements of a row or a column which has been normalized), we obtain $\sum b_i^2 \leq 1$. By proceeding this way, we get $\sqrt{\sum\sum a_{ij}^2} \leq \sqrt{m}$ and, multiplying by n and arguing analogously for rows and columns, we finally get that

$$n \cdot \text{rownorm} + m \cdot \text{colnorm} \in [2\sqrt{mn}, (\sqrt{m} + \sqrt{n})\sqrt{mn}].$$

The experimental results tell us that neither of those bounds is sharp enough.

We try another approximation: assuming $\hat{e}_i = e/m$ where e is the total number of entries of the matrix and $\hat{e}_i$ is an estimator for the number of entries in a row then, for the ML estimator properties (we can view that estimator as an estimator for a binomial distribution), $\hat{1/e_i} = m/e$. This way we see that there is approximately one datum for each one of the e_i entries and we have

$$\sqrt{\sum\sum a_{ij}^2} = \sqrt{\sum \frac{1}{e_i^2} e_i} = \sqrt{\sum \frac{1}{e_i}} \sim \sqrt{\sum \frac{m}{e}} = \sqrt{\frac{m^2}{e}} = \frac{m}{\sqrt{e}}.$$

Hence, after multiplying by n, we get a value of $mn/\sqrt{e}$ (the value obtained for the other normalization is the same, by symmetry). These are in good agreement with this value so it results a nice normalization. This is the expected value for a random matrix which has only ones at $e \ll mn$, but not a bound (as the matrix gets further away from the random model, the value gets also more different from this one).

This normalization is interesting because it makes the sub-scores of the sub-matrices have similar values as we increase the number of rows of the sub-matrices instead of having an increasing sequence as we prove in the following lemma.

Lemma 3.3. *Consider the sequence of values (x_n) corresponding to the Frobenius norm of the matrix obtained by taking into account the first n columns, and then normalizing by rows and columns. For n sufficiently big, (x_n) becomes increasing.*

Proof. If we normalize by columns, the result follows trivially since the other columns remain unchanged and we add a new positive term to the computation of the norm. If we normalize by rows, it suffices to show that, if the matrix has s data in the row and the new column adds d data, then

$$\sqrt{\frac{\sum a_i^2}{s^2}} \leq \sqrt{\frac{\sum a_i^2 + d}{(s+d)^2}}$$

which is equivalent to $\sum a_i^2 \leq ds^2/(2sd + d^2)$. By using the inequality between the arithmetic mean and the quadratic mean, we get

$$\sum a_i^2 \leq \frac{\sum a_i}{n} = \frac{s}{n}$$

so we need

$$\frac{s}{n} \leq \frac{ds^2}{2ds + s^2} \iff s \geq \frac{d}{n-2},$$

which is true for n big enough (and in general for our matrices n will almost always be big enough). □

After this discussion, since dispersion is high, we assume that our data will be closer to the random model and hence the score we assign to a $m \times n$ sub-matrix is the following (the overall score is obtained after adding up all the scores given to sub-matrices):

$$\text{score}(M) = \frac{n \cdot \text{rowscore} + m \cdot \text{colscore}}{mn/\sqrt{e}}. \tag{1}$$

After computing the overall score, we can divide by either the number of SVDs done (so as to compare our score to scores coming from partitions with different size) or by the expected number of SVDs for that size of the partition, in order to keep a penalty to flattening matrices which require a higher number of SVDs, because they have a higher number of columns.

4. Performance tests for several methods

To test the performance of our method and to compare it to the original Erik+2 method, we considered a set of 100 data files corresponding to trees with 12 leaves with the same topology but with random branch lengths. For every data set, we obtained the scores for 9 partitions, 3 of size 2×10, 3 of size 3×9 and 3 of size 5×7, where one partition of each size was an edge split and the rest were not.

The following tables 2 and 3 contain the information of the performance (success[1] and scores assigned to both edge splits and other partitions) for the methods corresponding to the following scores: sc_1 is the number of nonempty rows of the flattening matrix, sc_2 is the number of nonempty columns of the flattening matrices, sc_3 is the original Erik+2 score, sc_4 the Erik+2 score using the $mn/\sqrt{e}$ normalization, sc_5 the score given by the variant of our method[2] without dividing by the number of SVDs computed, sc_6 the score given by our method taking the arithmetic mean of the scores obtained for each sub-matrix, and sc_7 the score of our method taking a pondered mean of the sub-scores.

Partition	sc_1	sc_2	sc_3	sc_4	sc_5	sc_6	sc_7
2 vs 10	100	64	33	40	70	65	67
3 vs 9	100	50	39	31	42	35	36
5 vs 7	97	76	58	21	47	24	26

Table 2: Percentage of success of the different methods (where each method is represented by its score).

Partition		sc_1	sc_2	sc_3	sc_4	sc_5	sc_6	sc_7
2 vs 10	(ES)	16	57418	3209	181	9972	176	177
2 vs 10	(NES)	16	59347	3206	181	10502	179	183
3 vs 9	(ES)	62	38453	13926	296	10929	291	291
3 vs 9	(NES)	64	39422	14396	293	11134	288	288
5 vs 7	(ES)	890	8745	75489	589	4530	620	677
5 vs 7	(NES)	954	9816	87601	560	4814	584	638

Table 3: Average of the score given to edge splits (rows labelled with (ES)) and to partitions which are not edge splits (labelled with (NES)) by each method.

5. Conclusions

We can see that our method (score 5) works significantly better than the original Erik+2 method for 2 vs 10 partitions, since it recognizes the edge split of the three partitions 70 out of 100 times, and the original method worked fine only 33% of the time. This could be explained by the fact that the Erik+2 method

[1] We consider a test successful if the score the method assigned to the edge split is lower or equal than the score it assigned to two partitions which were not an edge split of that size. Notice that with our data set we could make 100 test for each size.

[2] We will refer as "our method" to the method that implements the modifications proposed above: reorder rows and columns, consider sub-matrices formed by the first ik columns in the i-th iteration, compute the score for each sub-matrix using (1) and add up all the scores, resulting in score 5. Scores 6 and 7 slightly modify this method by taking the mean of the sub-scores.

computes a single SVD where only 16 singular values are obtained (notice that the Erik+2 method is more accurate as the partition is more balanced), and that the dispersion present in flattening matrices coming from unbalanced partitions fits nicely with the assumptions we made to obtain the $mn/\sqrt{e}$ normalization.

For the 3 vs 9 case, our method turns out to be slightly better but not significantly; neither the original method nor ours provided a satisfactory result, so we think new ideas should be introduced to deal with this problem. In the 5 vs 7 case the most effective score turns out to be the original Erik+2 method, but we should notice that the percentage of success in taking the score as simply the number of columns is really high and the averaged difference of columns between edge splits and the other partitions is percentage-wise the most significant. The score sc_1 is not reliable for unbalanced partitions as the number of rows of the flattening matrices of unbalanced partitions is small and almost never there is an empty row (we can see in table 3 that, for 2 vs 10 and 3 vs 9 partitions, that averages for both edge splits and the rest of partitions are really close to 16 and 64, the number of rows of the flattening matrices). Nevertheless, when we consider balanced partitions (e.g., 5 vs 7) and hence the number of rows of the flattening matrices is higher, we can take it into account since for those cases the difference of scores between edge splits and the other partitions is noticeable and it has a huge percentage of success.

We can also see that, while we have reduced the relative difference between scores when averaging (although by doing this we are decreasing the percentage of success), those scores are not yet comparable. A noticeable fact is that for the method that works better (without averaging) scores obtained for the first two sizes are really close, but for the 5 vs 7 it reduces to less than one half (this is due to the fact that we make much less SVDs, as one can see looking at the averaged score), while for the original Erik+2 the score shows a steady increasing trend when the partition gets more balanced. We should also note that we worked estimations for the norm of the matrix and not for the distance to rank 4 flattening itself (they differ in the square of the first 4 singular values) and this could affect the success of our method in some cases.

References

[1] N. Eriksson, "Tree construction using singular value decomposition", in *Algebraic Statistics for computational biology*, 347–358, Cambridge University Press, New York, 2005.

[2] M. Casanellas and J. Fernández-Sánchez, "Relevant phylogenetic invariants of equivariant models", *J. de mathématiques Pures et Appliquées* **96** (2010), 207–229.

[3] J. Fernández-Sánchez, M. Casanellas, "Invariant versus quartet inference when evolution is heterogeneous across sites and lineages", *Systematic Biology* (2015), to appear.

[4] E. Allman, J.A. Rhodes, "Phyogenetic ideals and varieties for the general Markov model", *Adv. in Appl. Math.* **40** (2008), 127–148.