Blueprint of the "Alignment Neighborhood Explorer" (ANEX) (tentatively named)

Written by: Kiyoshi Ezawa

(in expecting the collaboration with Prof. Tetsushi Yada)

(Finished on August 4th, 2016; added CC4 statement on August 14th, 2016)

Outline

This computer program, the **"alignment neighborhood explorer" (ANEX)**, will take an input multiple sequence alignment (MSA), which is typically reconstructed by an aligner, and will explore the MSA's neighborhoods in the MSA space.

The program will explore the neighborhoods via a combination of "elementary moves," namely, "shifts" and "splits" of individual gaps, "merges," "purges" and "ex-nihilo"s of pairs of gaps, (and potential reverse moves of "(incomplete-)collapse of independent insertions (iCII)", "(incomplete-)creation of spurious independent insertions (iCSII)," **… POSTPONED!!)** etc.

Then, the program will assign predicted relative probabilities to the MSAs in the explored neighborhoods. The probabilities will be calculated based on a genuine stochastic evolutionary model of an entire sequence via substitutions and insertions/deletions (indels) with realistic indel length distributions.

As a "pre-processing" step, the program will also identify MSA portions with "potentially complex errors," where restoring the true MSA is very likely formidable. This will be done based on some properties of MSA regions, like the indel density and the possible involvement of long indels.

[This program was motivated by some preliminary analyses (Ezawa 2016a).]

Table of contents

1. Overall workflow

The flowchart of the program is shown in Figure 1. After being fed with an input MSA and a tree [(o)], it will first scan the MSA for the **gap configuration** of the entire MSA and, optionally, also for the portions likely containing **"purge"-type errors** [(i)]. During the course of this scan, it will also identify the portions with "**likely complex errors**" [(ii)]. Then, it will explore the neighborhoods of the MSA via **a sort of window analysis**, while calculating the **increments/decrements in the (log-) probability of the alignment** compared to that of the input MSA [(iii)]. Finally, the results will be **display**ed [(iv)]. In the following sections, I will elaborate on how these steps will be performed.

2. Scanning the input MSA

Given an input MSA, the program will first scan it horizontally, from left to right, and (conceptually) chop the MSA into an alternating series of **gapped and gapless segments** (on the left of step (i) in Figure 1). This operation is based purely on the **gap configuration** of the MSA. More precisely, a gapped segment is defined as a run of contiguous gapped columns delimited by gapless columns (or a gapless column and a MSA end), and a gapless segment is defined as a run of contiguous gapless columns (Ezawa, Graur and Landan, Part III). Optionally, the program will also search for portions where **"purge"-type errors** are likely (on the right of step (i) in Figure 1). (For "purge"-type errors, see, *e.g.*, Landan and Graur 2009; Ezawa 2016a.) This will be performed via a **sliding-window analysis** (Figure 2). In each window (of a specified width), the residue configuration will be examined. And, for each branch of the tree, the "observed" number of substitutions will be compared to the expected (or "predicted") number of substitutions within the window. The former will be estimated from the residue pattern within the window (using a likelihood analysis), and the latter will be estimated from the branch lengths (and optionally from the "observed" numbers of substitutions along other branches, see Appendix A1 for more details). If the "observation" significantly exceeds the "prediction" (according to a specified condition), the window will be marked as **"purge-like error suspected."** And overlapping windows will be merged to form a portion where a "purge"-like error is likely. **[For this purpose, it is necessary to examine the "purge"-like errors identified in the simulation analyses in more detail. See sections SR-1 and SM-1 in Supplementary Materiala, Part 2 for some preliminary analyses.]**

3. Identifying the MSA portions with "likely complex errors"

Quite frequently, MSAs reconstructed by an aligner (even if it is state-of-the-art) contain **"complex" errors**, each of which cannot easily be classified into a definite type or expressed

as a combination of definite types (Ezawa 2016a). It will also be very difficult to rectify such "complex" errors and to recover the true MSA from the reconstructed one in such a MSA portion. Therefore, it would be wise to exclude such portions from further analyses, and it would also be honest to tell the users which portions of the input MSA likely contain complex errors. The program will identify such portions by examining the **gap patterns** and the **predicted indel events** in each gapped segment. **[For this purpose, it is necessary to characterize the "complex" errors identified in the simulation analyses in more detail. See sections SR-3 and SM-3 of Supplementary Materials, Part 2, for some preliminary analyses.]**

4. Exploring the neighborhoods of an input MSA

After scanning the input MSA, the program will explore its neighborhoods via **a sort of window analysis**. Each **window** will be defined with a gapped segment, or (optionally) a region where a "purge"-type error is suspected, accompanied by its flanking regions of a user-specified width. Each flanking region may contain a gap or gaps. In that case, the region will be extended to fully include such (a) neighboring gapped segment(s).

In each window, the program will examine the MSAs obtained from the reconstructed MSA via an "**elementary move**" (Landan and Graur 2009; Ezawa 2016a), or a combination of a few elementary moves. (Users will be able to set the upper bound to the number of elementary moves to be examined.) Basically, the elementary moves are defined as the move of a "**gap-block**," which is a block of gaps delimited by two inter-column positions and a branch in the phylogenetic tree (Figure 3A). The program will attempt the following elementary moves (illustrated in Figure 4).

(i) A "**shift**" of a gap-block. This move just re-positions a gap-block without affecting others nearby (if at all) (Figure 4A). This can be done as long as the gap-block is **"isolated"** from others. A gap-block is called "isolated" either if it does not overlap others in horizontal positions, or if it is separated from each horizontally overlapping one by at least three branches in the phylogenetic tree (see panels B, C and D of Figure 3).

(ii) A **"purge"** of two *equal-sized* gap-blocks that affect the complementary sets of sequences (, which are separated by a single branch) (Figure 4B). (This move is the reverse of the "ex-nihilo" of a pair of (spurious) blocks.) This move is defined utterly uniquely, and therefore should be processed very quickly.

(iii-a) A "**merge**" of two gap-blocks that affect the same set of sequences and that are separated by a number of columns (Figure 4C). (This move is the reverse of a (erroneous) "split" of a gap-block.) This move is defined uniquely, *modulo* a **shift** of the resulting single

gap-block. Thus, it should be processed as quickly as shifts.

(iii-b) A "**merge**" of two *non-equal-sized* gap-blocks that affect the complementary sets of sequences (Figure 4D). This move, too, is defined uniquely, *modulo* a **shift** of the resulting single gap-block. Thus, it also should be processed as quickly as shifts.

(iv-a) A "**split**" of a single gap-block into two gap-blocks both of which affect the same set of sequences (the reverse of Figure 4C). There are {the block-length minus one} ways of splitting the gap-block, resulting in pairs of gap-blocks of different length-combinations. And each gap-block in each resulting pair may **shift** by some sites. Thus, it may take some time to process this move. If possible, it would be better to devise a measure to process this case quickly.

(iv-b) A "**split**" of a single gap-block into two gap-blocks, each of which affects each of the complementary sets of sequences (the reverse of Figure 4D). Although there are (theoretically) infinite ways of such moves, which result in different combinations of block-lengths, we will restrict the moves to consider by setting an upper bound on the shorter block-length. As in the previous case, each gap-block in each resulting pair may **shift** by some sites, which may consume quite an amount of time. If possible, it would be better to devise a measure to process this case quickly.

(v) An "**ex-nihilo**" of a pair of gap-blocks, each of which affects each of the complementary sets of sequences (the reverse of Figure 4B). There are many ways of performing this move, with different sets of original columns affected, and with different sizes of created gap-blocks. If possible, it would be better to devise a measure to process this case quickly.

(vi-a) A "**vertical-merge**" of a pair of *equal-sized* gap-blocks that affect two sequence sets separated by two branches (Figure 4E). (This is the reverse of a (erroneous) vertical-split of a gap-block.) This move is defined uniquely, modulo a **shift** of the resulting gap-block. Thus, it should be processed as quickly as a shift.

(vi-b) A "**vertical-merge**" of a pair of *equal-sized* sequence-blocks that involve two sequence sets separated by two branches (Figure 4F). (This is the reverse of a (erroneous) vertical-split of a sequence-block.) A "**sequence-block**" is defined as a complement of a gap-block (within the relevant MSA segment) (Figure 3E). This move, too, is defined uniquely, modulo a **shift** of the resulting gap-block (complementary to the resulting sequence block). Thus, it should be processed as quickly as a shift.

(vii-a) A "**vertical-split**" of a single gap-block into two (complementary) gap-blocks that affect two sequence sets separated by two branches (the reverse of Figure 4E). As long as the original gap-block affects a single clade (or the complement of a single clade), we can define a unique vertical-split (at the top node of the clade). (Other "vertical splits" actually increase

the number of indels by two or more each, as if they were CIIs (Figure S1A). … Such true MSAs seem very unlikely, though the probability is not zero... the actual frequency of errors rectified by these moves needs be examined in the future…) In such a vertical-split, each gap-block in the resulting pair may **shift** by some sites. Thus, the time-consumption should be on the same order as that for double-shifts.

(vii-b) A "**vertical-split**" of a single sequence-block into two (complementary) sequence-blocks that involve two sequence sets separated by two branches (the reverse of Figure 4F). In this case, too, as long as the original sequence-block involves a single clade (or the complement of a single clade), we can define a unique vertical-split (at the top node of the clade). (Other "vertical-split"s are actually "ex-nihilo"s aligned with a gap (Figure S1B). … The true MSAs with such patterns seem very unlikely, because the number of indels increases by two or more each…the actual frequency of errors rectified by these moves needs be examined in the future…) In such a vertical split, each resulting block may **shift**. Thus, the time-consumption should be on the same order as that for double-shifts.

(viii) "Shift + Shift" (Figure 4G) **… what's this??? … needs be examined further.**

[NOTE ADDED on Aug 4 (Thu), 2016: our preliminary analyses (SM-2 and SR-2 of Supplementary Materials, Part 2) indicated that most of the "Shift + Shift" errors are just simple one or two shift(s), which the current prototype script failed to annotate correctly because of the erroneous block partitioning. Thus, these cases should be handled similarly to the usual shift or two shifts.]

5. Calculating (logarithmic) probabilities of explored MSAs

Under most of the probabilistic models currently used for sequence evolutionary analyses, as well as under the (genuine stochastic) evolutionary models we are going to use, the occurrence probability of a MSA can be expressed as a product of its substitution component and indel component. (The proofs under the evolutionary models are given, *e.g.*, in Ezawa, Graur and Landan, Part IV and Ezawa 2016a.) Thus, we will discuss the calculations of these probability components separately.

5-1. Calculation of indel components of MSA probabilities

If the indel evolutionary model satisfies the "sufficient and nearly necessary set of conditions," the probability is factorable into a product of the overall contribution (determined by the root sequence state(s)) and multiplication factors contributed by gapped segments

(Ezawa, unpublished A; draft: Ezawa, Graur and Landan. Part I). Therefore, we can focus on the change in the multiplication factors from the gapped segments affected by the move in question (and possibly also the change in the overall factor). Especially, let's consider that we use an indel model whose indel rates are *space-homogeneous* (*i.e.*, homogeneous along the sequence) at least within the MSA portion in question. Then, **(1)** as long as the affected gap-block is *isolated* from all others (like in panels B and C of Figure 3), the multiplication factor *will* remain virtually *unchanged* under its shift. (See Appendix A2 for a proof.) Besides, **(2)** under a parameter setting where the first-approximation (by parsimonious indel histories alone) is quite good, the detailed (fine-grained) positional relationships between the *non-isolated* gap-blocks do *not* substantially *influence* the multiplication factor, as long as their *topologies* (and the sizes of the parsimonious indels) remain unchanged. (See Appendix A3 for a demonstration using some typical examples.) **These two good properties will help substantially reduce the computational burden when calculating the indel component.**

Let us now consider what we should do when trying different types of moves. (Recall that the moves are illustrated in Figure 4.)

(i) When trying a "**shift**," we don't have to do anything (as long as the gap-block is isolated). When the gap-block starts to interfere with others, we need to deal with the move accordingly.

(ii) When trying a "**purge**" of two gap-blocks, subtract the multiplication factors from the gapped segments that the gap-blocks belonged to, and add the contributions from newly created gapless columns. (We may also want to examine "intermediate states," where only portions of the blocks were purged (Figure S2A).)

(iii-a) When trying a "**merge**" of two gap-blocks affecting the same sequence set, subtract the multiplication factor(s) from the gapped segment(s) accommodating the gap-blocks to be merged, and add the multiplication factor from the new gapped segment accommodating the resulting gap-block. (We may also want to examine "intermediate states," where the bridging gapless segment moves… This corresponds to a "merge + split" (Figure S2B).)

(iii-b) When trying a "**merge**" of two gap-blocks affecting two complementary sequence sets, follow the rule similar to that in (iii-a), and also add contributions from newly created gapless columns. (We may also want to examine "intermediate states," where the two gap-blocks reduce by the same size… This corresponds to a "merge + split," or an "incomplete merge" (Figure S2C).)

(iv-a) When trying a "**split**" into two gap-blocks affecting the same sequence set, we try a number of combinations. For each combination, subtract the factor from the block to be split, and add the factors from the blocks newly created.

(iv-b) When trying a "**split**" into two gap-blocks affecting the complementary sequence sets,

too, we try a number of combinations. For each combination, follow the rule similar to that in (iv-a), and subtract contributions from "decomposed" gapless columns.

(v) When trying an "**ex-nihilo**," we try a number of combinations. For the group of combinations creating two gap-blocks of a particular size, add the factors from the gap-blocks newly created, and subtract contributions from "decomposed" gapless columns.

(vi-a) When trying a "**vertical-merge**" of a pair of (equal-sized) gap-blocks, subtract the factors from the gap-blocks to be merged, and add the factor from the newly created gap-block, and also add contributions from the newly created gapless columns.

(vi-b) When trying a "**vertical-merge**" of a pair of (equal-sized) sequence-blocks, subtract the factors from the segments containing the sequence-blocks to be merged, and add the factor from the segment containing newly created sequence-block.

(vii-a) When trying a "**vertical-split**" of a single gap-block into two gap-blocks, subtract the factor from the block to be split, and add the factors from the resulting two blocks, and also subtract the contributions from the gapless columns that were "split."

(vii-b) When trying a "**vertical-split**" of a single sequence-block into two sequence-blocks, subtract the factor from the segment containing the sequence-block to be split, and add the factors from the two segments containing the resulting sequence-blocks.

NOTE: Basically, when trying each of the moves (ii)-(v), it would suffice to change the factors contributed from the branch phylogenetically separating the gap-block(s). When trying each of (vi-a), (vi-b), (vii-a) and (vii-b), it would basically suffice to change the factors contributed from the three branches involved in the move. But there may be some exceptional cases. So, due caution should be exercised.

5-2. Calculation of substitution components of MSA probabilities

Basically, the calculations will be performed via Felsenstein's pruning (*aka* peeling) algorithm (Felsenstein 1981, 2004). (But the pulley principle will not be used if we need to keep the root position.) Either when scanning the entire MSA, or when beginning to explore the MSA neighborhood in each window, it would be convenient to store 2 sets of probabilities, $\{P_L(\omega; b, c)\}_{\omega\in\Omega}$ and $\{P_U(\omega; b, c)\}_{\omega\in\Omega}$, assigned to each combination of a branch (denoted as b) and a MSA column (denoted as c) (Figure 5A). Here, Ω denotes the "alphabet" of residues (bases or amino acids) used for the sequences. $P_L(\omega; b, c)$ is the conditional probability that, given residue $\omega\,(\in\Omega)$ at the lower-end of branch b, we observe the residues of all its descendant extant sequences present in column c. Probabilities of this type can be calculated via the standard pruning algorithm in a bottom-up manner (*i.e.*,

from leaves to the root) (described incidentally in Appendix A4). $P_U(\omega; b, c)$ is the joint probability that, in column c, we observe $\omega\ (\in \Omega)$ at the upper-end of branch b and the residues of all extant sequences on the upper-end-side of b, which is the complement of the set of descendant sequences of b. Probabilities of this type can be calculated via a top-down recursion (Appendix A4). It should be noted that these two sets of probabilities satisfy the following equation (for each branch b and each column c):

$$P(c) = \sum_{\omega\in\Omega} \sum_{\omega'\in\Omega} P_U(\omega; b, c)\ P(\omega \mapsto \omega'; b)\ P_L(\omega'; b, c)\ . \quad \text{--- Eq.(1)}$$

Here $P(\omega \mapsto \omega'; b)$ is the conditional probability that, given $\omega\ (\in \Omega)$ at the upper-end of b, we have $\omega'\ (\in \Omega)$ at its lower-end. And $P(c)$ is the probability that we observe the residues of all extant sequences present in column c. If there is enough memory space, it may be more convenient to store also the sets of "extended" probabilities, $\{\tilde{P}_L(\omega; b, c)\}_{\omega\in\Omega}$ and $\{\tilde{P}_U(\omega; b, c)\}_{\omega\in\Omega}$. They are defined as follows:

$$\tilde{P}_L(\omega; b, c) \equiv \sum_{\omega'\in\Omega} P(\omega \mapsto \omega'; b)\ P_L(\omega'; b, c)\ , \quad \text{--- Eq.(2)}$$

$$\tilde{P}_U(\omega; b, c) \equiv \sum_{\omega'\in\Omega} P_U(\omega'; b, c)\ P(\omega' \mapsto \omega; b)\ . \quad \text{--- Eq.(3)}$$

(For each of these probabilities, the sub-tree under consideration extends by branch b. See Figure 5, panels B and C.) These probabilities, P_L, P_U, $\tilde{P}_L$ and $\tilde{P}_U$, can be quickly calculated via a recursion algorithm on a round-trip traversal of the tree (Appendix A4). Using the "extended" probabilities, Eq.(1) can be simplified as:

$$P(c) = \sum_{\omega\in\Omega} \tilde{P}_U(\omega; b, c)\ P_L(\omega; b, c) = \sum_{\omega\in\Omega} P_U(\omega; b, c)\ \tilde{P}_L(\omega; b, c)\ . \quad \text{--- Eqs. (1'a,1'b)}$$

Moreover, they also satisfy a "three-way equation" at each bifurcated internal node n (other than the root):

$$P(c) = \sum_{\omega\in\Omega} \tilde{P}_U(\omega; b_P(n), c)\ \tilde{P}_L(\omega; b_{C1}(n), c)\ \tilde{P}_L(\omega; b_{C2}(n), c)\ . \quad \text{--- Eq.(4)}$$

Here $b_P(n)$ is the "parent" branch that is immediately above n, and $b_{C1}(n)$ and $b_{C2}(n)$ are the "child" branches that are immediately below n. (It is easy to generalize the equation to a multi-furcated node.) Its counterpart at the root (n^R) is simply the final equation of the recursion for the pruning algorithm:

$$P(c) = \sum_{\omega\in\Omega} \left[P(\omega; n^R) \prod_{b\in Ch(n^R)} \tilde{P}_L(\omega; b, c) \right]\ . \quad \text{--- Eq.(5)}$$

Here $P(\omega; n^R)$ is the probability (or the frequency) of residue $\omega\ (\in \Omega)$ at the root (n^R), and $Ch(n^R)$ denotes the set of child branches of n^R. **The equations 1'a, 1'b, 4 and 5 will play important roles when calculating the changes in the MSA probability due to an elementary move (and to the shifts accompanying each move).**

Let us now consider what we should do when trying different types of moves. (Recall that the moves are illustrated in Figure 4.)

(i) When a gap-block "**shift**"s by a single site, the change in the logarithmic probability of the residue pattern (Res) of the MSA within a window ($\alpha(W)$), $\Delta \log\left(P_{\mathrm{Res}}\left[\alpha(W) \right] \right)$, is calculated easily. For this purpose, let c' be the "semi-column" flanking the gap-block (and only involving sequences affected by the gap-block). Suppose that this c' moves when the gap-block shifts. Let c''_{bf} and c''_{af} be the "semi-columns" (involving complementary sequences) that were aligned with c' before and after the shift, respectively (Figure 6). And let (c', c''_{bf}) be the column formed by aligning the semi-columns c' and c''_{bf}. Similarly, let $(-, c''_{bf})$ be the column formed by aligning the semi-column c''_{bf} with a gap-only semi-column. Using these notations, the **change in the logarithmic probability** is calculated as:

$$\Delta \log\left(P_{\mathrm{Res}}\left[\alpha(W) \right] \right) \qquad \text{--- Eq.(6)}$$
$$= \log\left(P\left((c', c''_{af}) \right) \right) + \log\left(P\left((-, c''_{bf}) \right) \right) - \log\left(P\left((c', c''_{bf}) \right) \right) - \log\left(P\left((-, c''_{af}) \right) \right).$$

Each term on the right hand side can be easily calculated via Eq.(1'a) (or Eq.(1'b)). The shift of a gap-block by a number of sites results from a succession of its shifts by a single site each. Therefore, changes in the logarithmic probability via various shifts of a gap-block, by up to a specified number of sites, can be calculated quite quickly, nearly with the same time complexity as the calculation of the logarithmic probability of a MSA in a single window. Even if an isolated gap-block horizontally overlaps another *non-intefering* gap-block, the effects of their simultaneous shifts may still be calculated fairly quickly. (See Appendix A5 for a theoretical contemplation.)

(ii) When "**purge**"ing a pair of gap-blocks, the easiest way to calculate $\Delta \log\left(P_{\mathrm{Res}}\left[\alpha(W) \right] \right)$ would be to directly calculate $P(c)$ for all affected columns, before and after the purge, using Eq.(1'a) (or Eq.(1'b)).

(iii-a) As far as the substitution probabilities are concerned, "**merge**"ing a pair of gap-blocks affecting the same sequence set is equivalent to "shift"ing one of the blocks by a particular number of sites. Thus, basically, $\Delta \log\left(P_{\mathrm{Res}}\left[\alpha(W) \right] \right)$ can be calculated as in (i).

(iii-b) When "**merge**"ing a pair of gap-blocks affecting the complementary sequence sets, $\Delta \log\left(P_{\mathrm{Res}}\left[\alpha(W) \right] \right)$ could be calculated directly, by calculating $P(c)$ for all affected columns, before and after the merge, as in (ii).

(iv-a) When "**split**"ing a gap-block into two gap-blocks affecting the same sequence set, calculate as in (iii-a), noting that the MSAs 'before' and 'after' the move were swapped.

(iv-b) When "**split**"ing a gap-block into two gap-blocks affecting the complementary sequence sets, calculate as in (iii-b), noting that the MSAs 'before' and 'after' the move were swapped.

(v) When examining an "**ex-nihilo**," calculate as in (ii), noting that MSAs 'before' and 'after'

the move were swapped.

(vi-a) When examining a "**vertical-merge**" of a pair of gap-blocks, the easiest way to calculate $\Delta \log\left(P_{\mathrm{Res}}\left[\alpha(W) \right] \right)$ would be to directly calculate $P(c)$ for all affected columns, before and after the move, using the three-way equation (Eq.(4)). NOTE that the contributions from a *pure* gap-block should be $\tilde{P}_L\left(\omega; b, c \right) = 1$ if it is a descendant of the "pivotal" node (n_{PV}), which separates the clades of the two gap-blocks, and $\tilde{P}_U\left(\omega; b, c \right) = P\left(\omega; n_{PV} \right)$ if it is on the ancestral side of n_{PV} . (Incidentally, we assume that $P\left(\omega; n_{PV} \right)$ is given by:

$$P\left(\omega; n_{PV} \right) = \sum_{\omega' \in \Omega} P\left(\omega'; n^R \right) P\left(\omega' \mapsto \omega; pth(n^R, n_{PV}) \right) , \quad \text{--- Eq.(7)}$$

where $pth(n^R, n_{PV})$ is the shortest path connecting n^R and n_{PV} .)

(vi-b) When examining a "**vertical-merge**" of a pair of sequence-blocks, calculate similarly to (vi-a). The same note applies when dealing with a gap-block aligned with either sequence-block.

(vii-a) When examining a "**vertical-split**" of a gap-block, calculate in the same way as in (vi-a), noting that MSAs 'before' and 'after' the move were swapped.

(vii-b) When examining a "**vertical-split**" of a sequence-block, calculate in the same way as in (vi-b), noting that MSAs 'before' and 'after' the move were swapped.

(viii) When examining a "shift + shift" …??? (… Examine the concrete cases that were observed in the simulation analyses.) (NOTE ADDED on Aug 4 (Thu), 2016: Our preliminary analyses (sections SM-2 and SR-2 in Supplementary Materials, Part 2) indicated that this case should be dealt with in a similar way as a normal shift or a superposition of two normal shifts are.)

NOTE: For all cases (ii) through (vii-b) (except (i)), changes in the logarithmic probability, $\log\left(P_{\mathrm{Res}}\left[\alpha(W) \right] \right)$, via additional "shift"s can be calculated in the same way as in (i), as long as the shifted gap-block does not overlap any other horizontally.

6. How to output/display the results

Hmm… this is the problem…

One possible way would be to show a line graph for the **"shift"s** of each gap-block, where the abscissa is the position of the gap-block (more precisely, the relative position from the original one) and the ordinate is the (relative) MSA probability.

The question is what we should do for other moves, such as "merge"s, "purge"s, "split"s,

etc… (One possibility would be to give the increment in the probability for each of the candidate moves, and accompany a line graph (or a contour map, etc.) showing the increments by shifts from each such candidate.)

7. Possible applications of the results/outputs

The results obtained will be applied to a wide variety of analyses on molecular evolution, such as the predictions of functions, positive selection, 2ndary and tertiary structures, etc. [(We could also perform the analyses on *de novo* gene creation. .. because we will be able to take almost full account of uncertainties accompanying the inferences or estimations, the results will be somewhat reliable and meaningful.) … Considering the results of my (ongoing) preliminary analyses on fast-evolving mammals, however, it might be better *not* to use the yeast data for this purpose… they may be too divergent…needs to be examined further…]
The output of the program is the approximate probability distribution of possible MSAs. Therefore, we can estimate the **expected value (weighted average), variance, confidence intervals, distribution itself, etc.**, for each property/event (or combination of properties/events) we want to analyze. Of course, I expect that the expected value thus obtained will be more accurate than the "point estimation" of the property obtained from a single optimum MSA (output by a commonly used aligner). **(… Needs be confirmed by extensive analyses, of course.) In my view, however, its most important feature is that the program will indicate possible fluctuations on the results, and thus will honestly tell how accurately we can learn about the subject of our interest from the input data at hand, which will greatly help us avoid overconfidence on our predictions.** The program will also indicate regions that likely contain "complex" errors. This function itself may also be useful, because such regions are expected to be the hotbeds of erroneous predictions, especially on some peculiar evolutionary behaviors of sequences, such as positive selection, etc.

8. Conceivable problems

As with any brand-new method, there could be many problems that this program will suffer from. Here we discuss three major problems, namely, those of (i) how to provide the program with accurate model parameters (including the tree), (ii) autocorrelation, and (iii) overfitting.

(i) Model parameters (including phylogenetic tree)

The methods explained in this "blueprint" require a fixed set of model parameters, including the phylogenetic tree. Usually, however, we do not know the correct parameters before conducting an analysis, and thus we have to estimate the parameters from the input data at

hand. Theoretically, an ideal way would be to estimate the *joint* probability distribution of the MSAs and the model parameters including the phylogenetic trees (*e.g.*, Holmes and Bruno 2001; Lunter et al. 2005; Suchard and Redelings 2006; Novak et al. 2008). At present, however, this is very time-consuming and impractical *even if we use an unrealistically simplistic probabilistic model of indels (, which are not even evolutionary models, rigorously speaking)*. (Especially, the ML inference of the phylogenetic tree(s) is one of the most time-consuming problems in the study of molecular evolution.) Therefore, at least for the moment, it would be inevitable to first obtain a point estimation of the set of model parameters (including the tree), maybe using the input MSA, and to obtain the approximate MSA distribution under such a point-estimation. Or, we may develop a more sophisticated method that alternatingly (and iteratively) estimates (1) the set of model parameters (or their approximate distributions, maybe via bootstrapping) and (2) the approximate MSA distribution, where the estimation of one uses (the previous estimate of) the other as input. This may improve the accuracy of the parameter estimation. Similar method was actually developed for the reconstruction of a single-optimum MSA, and was indeed shown to improve the accuracy (Gotoh 1996).

(ii) Autocorrelations

It would be virtually impractical, in terms of both computational time and memory space, to perform an analysis by using the (approximate) probability distribution of MSAs of *entire* sequences, because the number of alternative MSAs would be super-astronomical. Therefore, it would be inevitable to perform a **window analysis**, using the approximate probability distribution of MSAs obtained for each window as defined in Section 4, and to integrate the results of all windows to obtain the "summary" result for the MSAs of *entire* sequences. Here comes the potential problem of **autocorrelations**, because each window is fairly likely to overlap the neighboring ones. Solving this problem would be very important for obtaining accurate results of the analyses.

(iii) Overfitting

At least in principle, our theoretical formulation of sequence evolution via indels (Ezawa, Graur and Landan, Parts I and II) can incorporate regional variation of indel rates to some extent. When actually incorporating this feature, however, we will need to be careful about the problem of **overfitting**, because, in general, sequences in a MSA experienced much fewer indels than substitutions during the evolution that created the MSA. Thus, it is very likely that simply employing a single optimum set of indel parameters for each MSA region will lead to

misinterpreting the stochastic fluctuation of an indel process as an indel rate variation. A possible way to mitigate this problem would be to give **an approximate distribution of the sets of indel parameters**, giving a discrete number of typical sets (or classes) and their relative probabilities in the *entire* MSA. The relative probabilities would be based on some well-known (continuous) distribution, such as a gamma-distribution and a beta-distribution. Such a method was developed for substitution models (Yang 2006, 2007), and it is commonly used for molecular evolution analyses these days. And I expect that the method, possibly with some modifications, will also be applicable to the problem of sequence evolution via indels.

Appendix

A1. Sliding-window analysis to identify MSA portions where "purge"-like errors are likely

A "**purge**" is a phenomenon where an aligner tries to optimize the MSA score by erroneously annihilating a pair of complementary gap-blocks that are usually quite close to each other (Figure 4B). In consequence, a "purge" tends to create a small-scale "**shear,**" where mutually aligned sequence blocks exhibit a nearly random residue match/mismatch pattern against each other. Taking advantage of this tendency, we may be able to identify MSA portions that likely harbor "purge"-like errors. Here we will describe a possible method.

Along a particular branch (b), we calculate the "**observed" number** and the **expected (or "predicted") number** of **substitutions** in each MSA column (c). Here, the "number of substitutions" does *not literally* mean the number that the site experienced along the branch. Rather, it means the number of *differences* in the residue between both ends of the branch. Thus, the "number of substitutions" is 0 if both ends have the same residue, and 1 otherwise.

The "**observed" number** is the number of substitutions estimated from the observed residue configuration in c. For this purpose, we use the key probabilities $\{P_L(\omega; b, c)\}_{\omega\in\Omega}$ and $\{P_U(\omega; b, c)\}_{\omega\in\Omega}$ introduced in subsection 5-2. (See Appendix A4 for how to calculate them.) More precisely, from these probabilities, we calculate the "**semi-posterior probabilities**" at both ends of b (and in column c):

$$P_{(L)}[\omega \mid n_L(b), c] \equiv P(\omega; n_L(b))\, P_L(\omega; b, c) \Big/ \Big[\sum_{\omega'\in\Omega} P(\omega'; n_L(b))\, P_L(\omega'; b, c)\Big],$$

--- Eq.(A1-1)

$$P_{(U)}[\omega \mid n_U(b), c] \equiv P_U(\omega; b, c) \Big/ \Big[\sum_{\omega'\in\Omega} P_U(\omega'; b, c)\Big]. \text{ --- Eq.(A1-2)}$$

Here, $n_U(b)$ and $n_L(b)$ denote the upper- and lower-ends, respectively. $P(\omega; n_L(b))$ denotes the probability (or relative frequency) of $\omega\,(\in\Omega)$ at node $n_L(b)$, which can be obtained by evolving $\{P(\omega; n^R)\}_{\omega\in\Omega}$ according to $P(\omega'\mapsto\omega''; b')$ along the branches (b') from the root to $n_L(b)$. The "**observed" number of substitutions** along b in a single column (c) may then be estimated as:

$$D_{Obs}(b, c) = 1 - \sum_{\omega\in\Omega}\{P_{(L)}[\omega \mid n_L(b), c]\, P_{(U)}[\omega \mid n_U(b), c]\}. \text{ --- Eq.(A1-3)}$$

[NOTE: It will be better to use the joint probability:

$P_{(U,L)}[\omega, \omega' \mid b, c] \equiv P_U(\omega; b, c)\; P(\omega\mapsto\omega'; b)\; P_L(\omega'; b, c) \;/\; P(c)$,

and define: $D_{Obs2}(b, c) = 1 - \sum_{\omega\in\Omega} P_{(U,L)}[\omega, \omega'=\omega \mid b, c]$ --- Eq.(A1-3').]

Then, the total "observed" number in a window (W) is simply given by a summation:

$$D_{Obs}(b, W) = \sum_{c \in W} D_{Obs}(b, c) \ . \quad \text{--- Eq.(A1-4)}$$

A simplest way to estimate the **expected number of substitutions** along b in a column (C) would be to use $\{P(\omega \mapsto \omega'; b)\}_{\omega,\omega' \in \Omega}$ and $\{P_{(U)}[\omega \mid n_U(b), c]\}_{\omega \in \Omega}$:

$$D_{Exp(1-a)}(b, c) = 1 - \sum_{\omega \in \Omega} \{P_{(U)}[\omega \mid n_U(b), c] P(\omega \mapsto \omega; b)\} \ . \quad \text{--- Eq.(A1-5a)}$$

Another simplest way would be to use $\{P(\omega; n_L(b))\}_{\omega \in \Omega}$, $\{P(\omega; n_U(b))\}_{\omega \in \Omega}$, $\{P(\omega \mapsto \omega'; b)\}_{\omega,\omega' \in \Omega}$ and $\{P_L(\omega; b, c)\}_{\omega \in \Omega}$ as follows:

$$D_{Exp(1-b)}(b, c) = 1 - \sum_{\omega \in \Omega} \{P_{(L)}[\omega \mapsto \omega \mid b, c]\} \ . \quad \text{--- Eq.(A1-5b)}$$

Here,

$$P_{(L)}[\omega \mapsto \omega' \mid b, c] \equiv P(\omega'; n_L(b)) P_L(\omega'; b, c) \ P^T(\omega' \mapsto \omega; b) \Big/ \left[\sum_{\omega'' \in \Omega} P(\omega''; n_L(b)) P_L(\omega''; b, c)\right]$$

, --- Eq.(A1-5c)

where we used the "time-reversed transition" probabilities defined as:

$$P^T(\omega' \mapsto \omega; b) \equiv P[(\omega, n_U(b)) \mid (\omega', n_L(b))] = P(\omega; n_U(b)) \ P(\omega \mapsto \omega'; b) / P(\omega'; n_L(b)) \ .$$

--- Eq.(A1-5d)

An unbiased way to estimate the expected values would be to use both of the above expected numbers. Here, we will use an arithmetic mean:

$$D_{Exp(1)}(b, c) = [D_{Exp(1-a)}(b, c) + D_{Exp(1-b)}(b, c)] / 2 \ , \quad \text{--- Eq.(A1-5e)}$$

and finally obtain:

$$D_{Exp(1)}(b, W) = \sum_{c \in W} D_{Exp(1)}(b, c) \ . \quad \text{--- Eq.(A1-6)}$$

[NOTE: In general, Eq.(A1-5e) should be almost equal to the more naively obtained value:

$D_{Exp(1')}(b, c) \equiv 1 - \sum_{\omega \in \Omega} \{P(\omega; n_U(b)) \ P(\omega \mapsto \omega; b)\}$ --- Eq.(A1-5e').]

A possible drawback of $D_{Exp(1)}(b, W)$ is that it may be too simple to take account of biological factors such as selection and mutation hot-spots. To incorporate such factors, a useful way would be to take advantage of the "observations" along different branches (and in the same window). There would be a variety of ways to implement this idea. Here, we will propose to **scale** Eq.(A1-6) by the ratio of the total "observed" number across the branches (except b) to the total "expected" number:

$$D_{Exp(2)}(b, W) \equiv R_D(b, W) \, D_{Exp(1)}(b, W) \ , \quad \text{--- Eq.(A1-7a)}$$

with

$$R_D(b, W) \equiv \left[\sum_{b' \in E_T, \, b' \neq b} D_{Obs}(b', W)\right] \Big/ \left[\sum_{b' \in E_T, \, b' \neq b} D_{Exp(1)}(b', W)\right] \ . \quad \text{--- Eq.(A1-7b)}$$

Here E_T denotes the set of all branches (edges) in the tree (T). Regardless of whether we choose $D_{Exp(1)}(b, W)$ or $D_{Exp(2)}(b, W)$ for the "expected" number of substitutions, we will use it as a reference (denoted as $D_{Exp}(b, W)$) to which the "observed" number, $D_{Obs}(b, W)$,

will be compared. Probably a simplest way to do this would be to calculate the P-value of such an "observation" under the binomial distribution with the "number of trials," $|W|$, and the per-trial probability, $p_D \equiv D_{Exp}(b, W)/|W|$:

$$P\left[D_{obs} \quad D_{Obs}(b, W)\right] = \sum_{x=D_{Obs}(b, W)}^{|W|} \frac{|W|!}{x!(|W|-x)!}(p_D)^x(1-p_D)^{|W|-x} \quad . \quad \text{--- Eq.(A1-8)}$$

We can set a threshold value (P_{Thrsh}) and regard the window as a region potentially undergoing a "purge" (along the branch) when Eq.(A1-8) is less than P_{Thrsh}.

A2. Proof that the multiplication factor(s) of the indel probability is independent of the relative positions between *isolated* gap-blocks

In (Ezawa, unpublished A; draft: Ezawa, Graur and Landan (Part I)), we proved that the alignment probabilities are factorable into the overall factor and the contributions from gapped segments (*i.e.*, local MSAs), provided that the evolutionary model satisfies certain conditions. In this blueprint, our calculation is based on an evolutionary model satisfying such conditions. Thus it is obvious that the indel probability is independent of the relative positions *as long as* the isolated gap-blocks in question are mutually separated by at least a gapless column (as in Figure 3B). We can therefore focus on the cases where the *isolated* gap-blocks overlap horizontally (as in Figure 3C and Figure S3A). It should be recalled here that, by definition, horizontally overlapped yet isolated gap-blocks (or, more precisely, the clades they affect) must be separated from each other by at least three branches. This means that there are at least two intervening nodes, each of which has at least one descendant sequence with residues aligned with the gap-blocks (*e.g.*, Figure S3A). Then, thanks to the **phylogenetic correctness condition** that the ancestral sequence states must satisfy (Chindelevitch et al. 2006; Diallo et al. 2007), it follows that the two gap-blocks must always be separated *phylogenetically* by at least two nodes whose gap-aligned positions are stuffed with residues (*e.g.*, Figure S3B). This in turn means that the indel history that resulted in one gap-block must always be confined in a subtree separate from that for the other gap-block, and they must never interact with each other to form a larger indel history (Figure S3C). (In fact, these facts originally motivated the aforementioned definition of the *isolated* gap-blocks.) (More precisely, there are some non-parsimonious indel histories that "connect" the isolated gap-blocks (Figure S4). However, such indel histories always require at least two more indels that are exquisitely coordinated in terms of both the positions and the indel sizes. Such histories are quite similar to the non-parsimonious indel histories that can yield type (i) local PWAs (Ezawa, unpublished B; draft: Ezawa, Graur and Landan, Part II), and thus in general their

contributions are expected to be negligible as well.) The multiplication factor contributed from each gapped segment (*i.e.*, local MSA) is a summation of terms over the sets of ancestral gap-configurations (Eq.(SA-1.2) in Supplementary appendix SA-1). And each term is a contribution from indel histories with a fixed set of ancestral gap-configurations, which could be expressed as a product of factors over the branches (Eq.(SA-1.3') in Supplementary appendix SA-1). Now, because, in dominantly contributing ancestral state sets, each of the (at least two) nodes separating the isolated gap-blocks has a fixed gap-configuration (with all sites occupied by residues), the sets of ancestral gap-configurations could be approximately expressed as a direct product of at least two partial state sets, one containing the clade for one gap-block and another containing the clade for the other gap-block (Eq.(SA-1.7) in Supplementary appendix SA-1; Figure S3C). **Therefore, in this case, the multiplication factor can be *further* factorized into the product of contributions: one from the portion of the tree with "fixed" ancestral states, and the others from the isolated gap-blocks** (Eqs.(SA-1.9,10) in Supplementary appendix SA-1; Figure S3D)**.** The independent horizontal re-locations of these gap-blocks do not change these factors (as long as they interfere with no neighboring gap-block). Thus, any change in the relative (horizontal) position between the *isolated* gap-blocks has *virtually* **no impact** on the segment-wise multiplication factor for the indel component of the MSA probability. See Supplementary appendix SA-1 for the mathematical details.

A3. Demonstration that the indel multiplication factor depends *only weakly* on the detailed (fine-grained) positional relationships between *non-isolated* (*i.e.*, interfering) gap-blocks

I think it too premature to prove it generally, because the related concepts are not adequately developed yet. Instead, I will consider a few (several) example cases here and demonstrate what the section title says.

We already proved in section A2 that the indel multiplication factor is independent of the relative positional relationships between *isolated* gap-blocks. So, in this section, we will focus only on the relative relationships between *non-isolated* (*i.e.*, *interfering*) gap-blocks. For this purpose, it is sufficient to consider the gap states of (ancestral) sequences connected with a 3-OTU tree (as in Figure 7). We consider three typical patterns: (A) two nested runs of gaps (Figure 7A); (B) two overlapping yet non-nested runs of gaps (Figure 7B); and (C) two non-homologous sequence-blocks sitting contiguously (Figure 7C). As will be explained below, the patterns A and B are topologically different; pattern A is topologically equivalent to the pattern in Figure 7D, but pattern B is not. And pattern C is topologically different from the

pattern in Figure 7E, where the sequence-blocks are separated via a gapless segment.

Pattern A can be created by two parsimonious indel histories (panels A and B of Figure 8), as well as by numerous non-parsimonious ones including some next-to-parsimonious ones (*e.g.*, panels C and D of Figure 8). In contrast, pattern B can be created by only one parsimonious indel history (panel A of Figure 9), as well as by numerous non-parsimonious ones including some next-to-parsimonious ones (*e.g.*, panels B-D of Figure 9). The indel histories that can create the pattern in panel D are nearly the same as those in Figure 8. The histories show the same trajectories of the **sequence lengths** and also the same series of **insertions and deletions with particular sizes**; they differ only in the detailed *horizontal* positions of indels. As can be seen from (Ezawa, unpublished A; drafts: Ezawa, Graur and Landan, Part I), *when the indel rates are space-homogeneous*, the portion of the multiplication factor contributed by each indel history is determined solely by the trajectory of the **sequence lengths** and the series (actually the set) of **insertions and deletions with particular sizes**, but it is independent of detailed *horizontal* positions of indels. (See, *e.g.*, Eq.(SM-2.7) and Eq. (R8-1.4) in Ezawa, unpublished A.) Therefore, *as long as the parsimonious indel histories contribute predominantly to the multiplication factor*, the multiplication factor from pattern A is nearly equal to that from the pattern in Figure 7D, but the contributions from patterns A and B will differ considerably from each other.

Patten C can be created by three *types of* parsimonious histories (panels A, B and C of Figure 10), as well as by some non-parsimonious histories (*e.g.*, panel D of Figure 10). In contrast, the pattern in Figure 7E can be created by two parsimonious histories (panels A and B of Figure 11), as well as by some non-parsimonious histories (*e.g.*, panel C of Figure 11). Therefore, pattern C should be substantially more likely to occur than the pattern in Figure 7E (provided that the number of intervening gapless columns is fixed). On the other hand, the occurrence probability should remain unchanged even if a different (nonzero) number of gapless columns separate the two gap-blocks in Figure 7E (provided that the entire region encompassing the two gap-blocks has a fixed length).

These examples demonstrate that, *at least as long as the contributions by parsimonious indel histories predominate*, **the multiplication factor is determined mostly by the coarse-grained topological relationships between gap-blocks**, but depends only weakly on the fine-grained positional relationships between them.

A4. Algorithm to quickly calculate key probabilities on residue configuration in each column

In subsection 5-2, we defined 4 sets of important probabilities, $\left\{ P_L(\omega; b, c) \right\}_{\omega \in \Omega}$,

$\{P_U(\omega; b, c)\}_{\omega\in\Omega}$, $\{\tilde{P}_L(\omega; b, c)\}_{\omega\in\Omega}$ and $\{\tilde{P}_U(\omega; b, c)\}_{\omega\in\Omega}$, associated with each combination of a MSA column (c) and a branch (b) (Figure 5). As we saw in the subsection, these probabilities provide essential building blocks when we calculate changes in the residue component of the MSA probability. Here, we give an algorithm that quickly calculates these 4 sets of probabilities along all branches and in a single column. The algorithm is organized in two stages: the first stage proceeds "bottom-up" along the tree, *i.e.*, from the leaves to the root node; and the second proceeds "top-down," *i.e.*, from the root to the leaves.

The first, "bottom-up" stage is actually a slightly modified version of Felsenstein's pruning algorithm (Felsenstein 1981, 2004), and efficiently calculates $\{P_L(\omega; b, c)\}_{\omega\in\Omega}$ and $\{\tilde{P}_L(\omega; b, c)\}_{\omega\in\Omega}$. It starts with an "initialization" step, where each external branch (denoted as b^x here) is associated with the following probabilities:

$$P_L(\omega; b^x, c) = \delta(\omega, \omega(c, n^x)) \quad , \text{ --- Eq.(A4-1a)}$$

$$\tilde{P}_L(\omega; b^x, c) = P(\omega \mapsto \omega(c, n^x); b^x) \quad . \text{ --- Eq.(A4-1b)}$$

Here $\omega(c, n^x)$ denotes the residue that the sequence at the external node n^x has in column c. (The external node n^x is the lower-end of b^x.) The symbol $\delta(\omega, \omega')$ denotes Kronecker's delta: $\delta(\omega, \omega') = 1$ if $\omega = \omega'$, $= 0$ otherwise. [NOTE: When the sequence has no site in the column c, $P_L(\omega; b^x, c) = \tilde{P}_L(\omega; b^x, c) = 1$ are assigned.] And, as already explained in subsection 5-2, $P(\omega \mapsto \omega'; b)$ is the probability that the lower-end of branch b has residue ω' conditioned on that its upper-end has ω. After the initialization, as the algorithm climbs up the tree, each internal branch (denoted as b) is associated with the probabilities via the following recursion relations:

$$P_L(\omega; b, c) = \prod_{b'\in Ch(b)} \tilde{P}_L(\omega; b', c) \quad , \text{ --- Eq.(A4-2a)}$$

$$\tilde{P}_L(\omega; b, c) = \sum_{\omega'\in\Omega} P(\omega \mapsto \omega'; b)\, P_L(\omega'; b, c) \quad . \text{ --- Eq.(A4-2b)}$$

Here $Ch(b)$ denotes the set of "child" branches of branch b. Eq.(A4-2a) follows easily from what these probabilities mean (see Figure 5, panels A and B), and Eq.(A4-2b) is nothing other than the definition of $\tilde{P}_L(\omega; b, c)$ (*i.e.*, Eq.(2)).

The second, "top-down" stage calculates $\{P_U(\omega; b, c)\}_{\omega\in\Omega}$ and $\{\tilde{P}_U(\omega; b, c)\}_{\omega\in\Omega}$ efficiently. In its "initialization" step, it associates each child branch of the root ($b \in Ch(n^R)$) with the probabilities:

$$P_U(\omega; b, c) = P(\omega; n^R) \prod_{b'\in Ch(n^R),\, b'\neq b} \tilde{P}_L(\omega; b', c) \quad , \text{ --- Eq.(A4-3a)}$$

$$\tilde{P}_U(\omega; b, c) = \sum_{\omega'\in\Omega} P_U(\omega'; b, c)\, P(\omega' \mapsto \omega; b) \quad . \text{ --- Eq.(A4-3b)}$$

Here $P(\omega; n^R)$ is the probability (or the relative frequency) of residue ω at the root node (

n^R). Eq.(A4-3a) easily follows from what $P_U(\omega; b, c)$ and $\tilde{P}_L(\omega; b', c)$ mean (Figure 5, panels A and B), and Eq.(A4-3b) is nothing other than the definition of $\tilde{P}_U(\omega; b, c)$ (*i.e.*, Eq.(3)). After the initialization, as the algorithm goes down the tree, each internal branch (again, denoted as b) is associated with the probabilities via the following recursion relations:

$$P_U(\omega; b, c) = \tilde{P}_U(\omega; p(b), c) \prod_{b' \in Ch(p(b)),\, b' \neq b} \tilde{P}_L(\omega; b', c) \ , \quad \text{--- Eq.(A4-4a)}$$

$$\tilde{P}_U(\omega; b, c) = \sum_{\omega' \in \Omega} P_U(\omega'; b, c)\, P(\omega' \mapsto \omega; b) \ . \quad \text{--- Eq.(A4-4b)}$$

Here, $p(b)$ denotes the "parent" branch of b, and Eq.(A4-4b) is, again, exactly Eq.(3).

The entire algorithm (on a single column) has the time complexity of $O\left(|E_T| \times |\Omega| \times \left(|\Omega| + \langle |n| \rangle_T\right)\right)$, where $|E_T|$ is the number of branches (edges) in the tree, $|\Omega|$ is the number of letters in the alphabet (Ω), and $\langle |n| \rangle_T$ is the average connectivity over the internal nodes in the tree.

A5. Contemplating on fast computation of the effects of simultaneous shifts of *non-interfering* (i.e., *isolated*) gap-blocks on the residue component of MSA probability

When the *isolated* gap-blocks are separated by at least two gapless columns (as in Figure 3B) or, even if they *horizontally* overlap each other (as in Figure 3C), when their shifts only affect different columns independently (as in Figure 12A), it is almost trivial to show that the change in the log substitution probability ($\Delta \log\left[P_{\mathrm{Res}}\left[\breve{\alpha}(W)\right]\right]$) by such two simultaneous shifts is exactly the summation of two changes, each of which resulted from a single individual shift (as given by Eq.(6)). (Actually, the gap-blocks need not necessarily be separated by three or more branches in situations like in Figure 12A.) Thus, **we can focus on cases where a column (or a pair of columns) is affected simultaneously by the shifts of two *isolated* gap-blocks (as in Figure 12, panels B and C)**. By carefully examining the cases like in panels B and C of Figure 12, we find that the effects of the shifts of two isolated gap-blocks can be regarded as nearly independent of each other **if the following approximate equation holds**:

$$\log\left(P\left((c', c''', c'')\right)\right) + \log\left(P\left((-, c'', -)\right)\right) \overset{?}{\approx} \log\left(P\left((c', c''', -)\right)\right) + \log\left(P\left((-, c''', c'')\right)\right) \ .$$

--- Eq.(A5-1)

(The proof is given in Supplementary appendix SA-2.) Here, c' and c'' are the portions of MSA columns flanking the two gap-blocks in question, and c''' is the remaining portion of the column. (c', c''', c'') denotes the MSA column created by aligning these portions. $(c', c''', -)$ denotes the column made from (c', c''', c'') by replacing the residues in c''

with gaps. And other similar symbols can be interpreted accordingly. Unfortunately, our further examination revealed that **the approximate equation, Eq.(A5-1), should fail to hold in general,** especially unless at least one of the involved branches is so long that the transition probabilities will saturate. (See Supplementary appendix SA-3 and Figure 13 for details.)

Nevertheless, we could still calculate the effects of double-shifts pretty fast, though with some possible twists**.** This is because the shifts of each gap-block can provide at most three possible residue configurations to (the relevant portion of) each affected column (Figure 14). Thus, when calculating the effects of the shifts of two isolated gap-blocks that possibly overlap horizontally, it would be sufficient to pre-calculate and store the probabilities of $3\times3=9$ possible residue configurations for each column (Figure 15), and to deploy them properly. This way, we may be able to calculate the effects of the shifts of up to 3 (or may be 4) isolated yet possibly overlapping gap-blocks fairly quickly. (In the former case, we need to calculate the probabilities of $3\times3\times3=27$ configurations per column. A possible problem would be on whether or not there is a quick way to deploy the proper column-wise residue configurations to all possible cases.)