Theory and Applications of Categories, Vol. 33, No. 40, 2018, pp. 1269–1317.

CATEGORY THEORY FOR GENETICS I: MUTATIONS AND SEQUENCE ALIGNMENTS

RÉMY TUYÉRAS

Abstract. The present article is the first of a series whose goal is to define a logical formalism in which it is possible to reason about genetics. In this paper, we introduce the main concepts of our language whose domain of discourse consists of a class of limit-sketches and their associated models. While our program will aim to show that different phenomena of genetics can be modeled by changing the category in which the models take their values, in this paper, we study models in the category of sets to capture mutation mechanisms such as insertions, deletions, substitutions, duplications and inversions. We show how the proposed formalism can be used for constructing multiple sequence alignments with an emphasis on mutation mechanisms.

1. Introduction

1.1. Short presentation. The goal of the present article is to define a type of algebraic structures in which it is possible to *do genetics*. The main operation provided by these structures is a formal way of 'gluing' different pieces of information together. To motivate the various notions introduced in this paper, we focus on a particular example, namely the construction of multiple sequence alignments, which are often used in phylogenetics. The proposed formalism can be applied to many other situations that would more generally look at sequential polymer comparisons and/or interactions.

1.2. Motivations. Our objective is to construct a bridge between two completely disconnected domains of science, specifically genetics and category theory, through a series of papers. While genetics is well-known for its complexity, category theory is recognized for its clarity and expressive power [27, 2, 12]. The goal of the present program would be to reach a level of abstraction that would allow one to tackle questions whose formulation are too complicated to be addressed with the current tools.

The language of the present paper is rather mathematical, but the results and definitions that it contains always try to capture the biological reality. Throughout the paper,

This research was supported by the AFOSR grant, *Categorical approach to agent interaction*, FA9550-14-1-0031 and the AFOSR grant, *Pixel matrices and other compositional analyses of interconnected systems*, FA9550-17-1-0058.

Received by the editors 2017-12-01 and, in final form, 2018-12-06.

Transmitted by Kathryn Hess. Published on 2018-12-13.

2010 Mathematics Subject Classification: 18C99, 92B99.

Key words and phrases: Multiple sequence alignment, mutation mechanism, right Kan extension, category, limit sketch, pedigrad.

some terms might be used in a biological sense while others might be used in a mathematical one – this will usually be specified. For instance, the sentence "a structure in which it is possible to do genetics" means that we want to define a formal language rather than a model of some particular living body. The need for such an abstraction, in biology, has, for example, been recognized in [10, 24].

Attempts at linking genetics (or in fact molecular biology) to categorical thinking are not new. A first example is [22], in which a category-like formalism is used to discuss the algebraic properties of "DNA wallpapers". Another work is [3], in which Carbone & Gromov model DNA, RNA and proteins by using topological and geometrical objects such as surfaces and moduli spaces. The program proposed herein tries to understand the mechanisms of genetics in themselves by forgetting the spacial aspect and focusing on the biological operations occurring in the body. Such an algebraic approach to biology has already been discussed, from the point of view of neuroscience, in several unpublished works by Ehresmann (for example, see [6]) using the concepts of *limit* and *cone*. The present paper takes a step further, in the context of genetics, by providing a precise 'limit theory' (in fact, a *limit sketch*) that can be used to formalize precise concepts of genetics. In this respect, our structures will define formal environments in which one wants to express a problem and say things about its solution.

In addition to offering a formalism, the proposed program aims to tackle technical and/or conceptual problems of various sub-fields of genetics. While the next article [28] will focus on questions related to genotypes, phenotypes, haplotypes, homologous recombination, and genetic linkage, the present article focuses on questions related to the construction of multiple sequence alignments [19, 15, 5] from a mechanistic point of view (see section 2.1 for an introduction).

Specifically, the present work is an attempt to give a categorical answer to the program proposed in [14, 13] regarding the construction of multiple sequence alignments by trying to "[recognize] mechanisms rather than assuming that all the variation occurs at random [at every position in the DNA strand]" [14, page 156, right col., l. 5]. More precisely, our goal is to show that the language of category theory can be used to put more emphasis on evolutionary mechanisms so that "[mutation] events [can] be identified as the alignment proceeds rather than being identified after the alignment is completed"[14, page 156, right col., l. 9].

Finally, the reader can find a Python library that aims to implement the content of the whole program at the web address:

`https://github.com/remytuyeras/pedigrad-library`.

The library will be updated as the program evolves towards more tools.

1.3. Road map and results. The goal of the present paper is to define a class of theories, called *chromologies*, whose models, called *pedigrads*, will be shown (through the program) to recover various aspects of genetics by changing the associated categories of values.

We begin by defining chromologies in sections 2.2 through section 2.36, while the pedigrads for these theories will be defined in sections 2.38 & 2.40. Intuitively, chromologies

allow us to do all sorts of basic DNA manipulations such as sequence alignments, CRISPR [17] and homologous recombination, whereas the pedigrads allow us to give a context to these operations (which can be handled differently depending on the environment in which they are processed).

In section 3, we define a class of functors (Definition 3.13) that model the environments of DNA sequences (Example 3.15). We show that these functors can be endowed with two types of pedigrad structures in the category of sets (Theorems 3.47 & 3.49): the first type detects the exact consistency of the data while the second type detects the consistency of the data up to uncertainty (see Example 3.37). These functors are then used to formalize the concept of sequence alignment in terms of a functor (Definition 3.23). We will see that taking the right Kan extension of a sequence alignment functor can be viewed as constructing multiple sequence alignments (Remark 3.34). In Example 3.38, we will see that the right Kan extension of a sequence alignment functor contains both local and global pieces of information that inform us of the presence of uncertainties in the integrated data. At the end of the section, we will see in this uncertainty a justification for the concepts of chromology and pedigrad, which will give us ways to locate, specify or isolate the existing uncertainties (see Example 3.50).

In section 4, we will introduce the concept of a slice of a sequence alignment functor (Definition 4.9), which will allow us to resolve the mentioned uncertainties and select multiple sequence alignments that are consistent with the overall data. First, in section 4.6, we will discuss the selection of consistent sequence alignments through the use of chromologies. Then, in Remark 4.14, we will suggest an algorithm for constructing multiple sequence alignments via the use of slices. Finally, in Section 4.15, we will show that the presence of uncertainties in the right Kan extension of a sequence alignment functor can be due to mutation mechanisms. We will show that the resolution of these uncertainties, permitted by slices, gives us a way to recognize mutation mechanisms (Examples 4.16 & 4.18).

1.4. Acknowledgments. I would like to thank the referee for their very useful comments and remarks, which led to a significant improvement of an earlier version of this paper. I would also like to thank Brendan Fong, David Spivak and Eric Neumann for useful discussions. Finally, I would like to thank Andres Saez and Anjanet Loon for their careful reading of this manuscript.

2. Chromologies and Pedigrads

The goal of this section is to introduce a set of theories whose logical models try to capture the logic of genetics. To justify why our theories look the way they do, we need to recall a few facts regarding the construction of theories in general. First, recall that, classically, models for theories are defined as sets equipped with some operations. For instance, a *ring* is a set R equipped with two operations $\cdot : R \times R \to R$ and $+ : R \times R \to R$ making certain diagrams commute.

More categorically, rings are also product-preserving functors from a certain product sketch[1] `Ring` (the theory) to the category **Set** of sets and functions [7]. This functorial point of view was introduced by Lawvere [9] in 1963 via the concept of what is now called a *Lawvere theory* – the theory `Ring` being an example. The advantage of functors over sets equipped with functions is that functors allow us to clearly distinguish between what is intrinsically true in a model (via the theory) and what can occasionally be true in the model (via the images of the functor). Then, the formalism accompanying the language of functors allows us to more carefully think about the mechanisms governing the models.

Since Lawvere theories were meant to capture the logic of algebraic structures equipped with multivariate functions, their objects were taken to be the set of natural numbers in order to specify the arities of the functions. Along those lines, since the goal of the present section is to define a theory that captures the logic of genetics and whose operations take DNA segments as inputs, the objects of our theory will look like DNA segments. Note that, while, in rings, one *adds* and *multiplies* terms together, in genetics, one *cuts*, *aligns* and *recombines* DNA strands together. Therefore, our theory will be based on these operations.

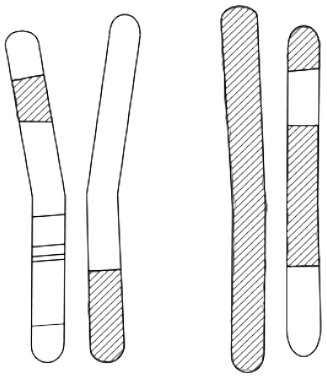
More specifically, recall that an integer object in a Lawvere theory can be represented as a finite sequence of atoms; e.g the object 6 would be represented by six atoms as follows.

$$6 = (\bullet\bullet\bullet\bullet\bullet\bullet) \tag{1}$$

These atoms can make it easier to see how the models defined on the Lawvere theory send the integer objects to the product objects in the category of values; *e.g.* for a given functor R, the image $R(6)$ would be sent to a product of the following form where $R(\bullet) = R(1)$.

$$R(\bullet) \times R(\bullet) \times R(\bullet) \times R(\bullet) \times R(\bullet) \times R(\bullet)$$

In the case of DNA, the idea is to copy the previous picture, but by adding enough information to be able to model genetic mechanisms. If one looks at the type of pictures drawn by biologists to explain homologous recombination, alignment methods or even genetic linkage, one can often see pictures of chromosomal patches subdivided in terms of selected and masked regions, as shown below.



[1] A small category equipped with a subset of its wide spans (see Definition 2.34).

These colored regions are obviously reminiscent of the term *chromo-some*[2] itself. The regional separations are also reminiscent of some sort of topology – or metric. If one tries to merge these topological and colored components with the type of atomic representation given in (1), we are likely to end up with the following type of pictures.

$$(\bullet\bullet\bullet)(\circ\circ)(\bullet\bullet\bullet\bullet)(\bullet\bullet\bullet\bullet\bullet)(\circ\circ\circ)(\circ) \tag{2}$$

In picture (2), the black nodes could indicate the regions of the chromosome that one wants to use while the white nodes could indicate the parts of the chromosome that one wants to ignore (mask). Note that a black-and-white paper does not give more than two colors to color the previous type of objects. Therefore, we will not hesitate to use labels to represent new colors. Formally, our sets of colors will be encoded by pre-ordered sets, whose semantics will allow us, among other operations, to *select* and *cut*.

2.1. Main example. To help the exposition of the present paper, most of our examples will focus on a single problem, which will give a story to our demonstration. Each example will illustrate how the mathematical definitions given in this paper can be used to clarify, explain or solve precise aspects of our problem.

Without any further introduction, our problem will look at the alleles of four different individuals for the same gene. Our goal will be to show how one can use chromologies and their pedigrads to help us relate these four individuals. In general, the first step to establishing the genealogy of a set of individuals is to align their genetic data according to an evolution model (*e.g.* using substitution, insertion and deletion mutations) in order to determine how one individual evolved from another one (see [19, 15, 5]).

In our case, we will consider the set of individuals given in the following table with respect to the corresponding genetic data shown on the right.

Individuals	Alleles
`Anne`	`ACCGACTG`
`Bob`	`ACATCTG`
`Craig`	`ACCGTCA`
`Doug`	`ACTACTG`

Recall that, in bioinformatics, the only analytic method that can compare and align a set of DNA sequences is the *dynamic programming algorithm* (see [25] or [15, Chapter 3 and page 71]). Other methods obviously exist, but these require heuristics that generally only produce an approximation of the best alignment [5, 8, 4, 11]. A common aspect between these methods is that they all compare DNA sequences two by two. Many of them even rely on the pairwise dynamic programming algorithm, which can compare any given pair of sequences together. For instance, to compare the genetic data of two individuals, say `Anne` and `Bob`, via the dynamic programming algorithm [15, page 71], we would first draw a table as given below, on the left-hand side, where second topmost row and second leftmost column are initialized with canonical scores (called *gap penalties*).

[2]meaning *color-body*

	ε	A	C	C	G	A	C	T	G
ε	0	1	2	3	4	5	6	7	8
A	1								
C	2								
A	3								
T	4								
C	5								
T	6								
G	7								

$\Rightarrow$

	ε	A	C	C	G	A	C	T	G
ε	0	1	2	3	4	5	6	7	8
A	1	0	1	2	3	4	5	6	7
C	2	1	0	1	2	3	4	5	6
A	3	2	1	1	2	2	3	4	5
T	4	3	2	2	2	3	3	3	4
C	5	4	3	2	3	3	3	4	4
T	6	5	4	3	3	4	4	3	4
G	7	6	5	4	3	4	5	4	3

Then, we would produce the table given on the right-hand side by following two types of scoring rules, one taking care of matches and the other one taking care of mismatches. Usually, these rules tell us how to fill out a box in the case where we have a two-by-two matrix whose bottom-right corner is empty and whose remaining boxes are already filled with scores, as shown below.

p	q
r	

In our case, the scoring table was filled with the following rules:

1) if the nucleotides labeling the column and the row of the empty box are equal, then the empty box should be filled with the score p;

2) if the nucleotides labeling the column and the row of the empty box are different, then the empty box should be filled with the score $\mathsf{min}(p, q, r) + 1$;

The best alignments for `Anne` and `Bob` are then obtained by tracing back the previous rules from the bottom-right corner of the table to its top-left corner. All the paths of moves (from right to left) that would make the earlier rules hold describe the nucleotide comparisons that give rise to the best alignments. In the present case, there are more than one paths. One of them is shown in the table given below, on the left (starting from the bottom-right corner). The associated alignment, given on the right, is read from the left-top corner to right-bottom one, where every symbol ε represents a stationary move from the point of view of the sequences.

	ε	A	C	C	G	A	C	T	G
ε	**0**	1	2	3	4	5	6	7	8
A	1	**0**	**1**	2	3	4	5	6	7
C	2	1	0	**1**	**2**	3	4	5	6
A	3	2	1	1	**2**	**2**	3	4	5
T	4	3	2	2	**2**	**3**	3	3	4
C	5	4	3	2	3	3	**3**	4	4
T	6	5	4	3	3	4	4	**3**	4
G	7	6	5	4	3	4	5	4	**3**

$\Rightarrow$

```
ACCGAεCTG  Anne
AεCεATCTG  Bob
```

Of course, constructing a sequence alignment for a set of four individuals by only using pairwise comparisons is likely to miss certain optimal alignments. In fact, computing an optimal alignment for our set of individuals would require a generalization of the previous algorithm in a hypercube. The problem with an algorithm based on such a structure is that it can rapidly become computationally expensive. As a result, biologists prefer to use heuristics such as the so-called *progressive method* [15, 5, 8, 4, 11].

The need for these heuristics show that passing from a two-dimensional point of view to a higher dimensional one, by trying to "glue" the previous tables together (in order to reconstruct the hypercube) contains a lot of subtleties.

In this article, we will show how category theory can help formalize, clarify and reason about this passage by defining formal "gluing methods" of two-dimensional tables as used above. These "gluings algorithms" will take the form of limit-preserving functors, our so-called *pedigrads*, and the associated gluing instructions will be specified by collections of limit-cones, our so-called *chromologies*.

2.2. Pre-ordered sets. Throughout the paper, the most basic notions of ordered sets are expected to be known by the reader (*e.g.* partially ordered sets; totally (or linearly) ordered sets; pre-ordered sets; see [12, Page 11]). However, because pre-orders will play an important role, it was felt appropriate to recall their definition and give some examples of interest in a separate section. We also recall the definition of order-preserving functions and define a category of pre-ordered sets.

2.3. Definition. [Pre-ordered sets] *A* pre-ordered set *consists of a set* Ω *and a binary relation* $\leq$ *on* Ω *satisfying the following logical implications.*

1) (reflexivity) for every $x \in \Omega$*, the relation* $x \leq x$ *holds;*

2) (transitivity) for every $x, y, z \in \Omega$*, if* $x \leq y$ *and* $y \leq z$ *hold, then so does* $x \leq z$*.*

2.4. Example. The set $\{0, 1\}$ is a pre-ordered set when equipped with the relations $0 \leq 1$; $0 \leq 0$ and $1 \leq 1$. The resulting pre-ordered set is usually known as the *Boolean* pre-ordered set.

2.5. Remark. [Representation] Pre-ordered sets may happen to be sets of labels (or even sets of structures) instead of being sets of integers. In the case of the Boolean pre-ordered set given in Example 2.4, the labels `false` and `true` will sometimes be used instead of the integers 0 and 1, mainly for the sake of clarity when integers are used for another purpose.

2.6. Example. The set $\{0, 1\}$ could also be equipped with the discrete pre-order made of the reflexive relations $0 \leq 0$ and $1 \leq 1$ only.

2.7. Example. For every positive integer n, the n-fold Cartesian product $\{0, 1\}^{\times n}$ of the pre-ordered set given in Example 2.4 is equipped with a pre-order relation $\leq$ that relates two tuples in $\{0, 1\}^{\times n}$, say $(x_1, \ldots, x_n) \leq (y_1, \ldots, y_n)$, if, and only if, the relation $x_i \leq y_i$ holds for every index i between 1 and n.

2.8. Example. The interval $[0,1]$ is a pre-ordered set for the usual pre-order "being less than or equal to" defined on the set $\mathbb{R}$ of real numbers.

2.9. Remark. [Pre-order categories] Recall that a pre-ordered set is also a category in which there exists at most one arrow between every pair of objects. In the sequel, a pre-ordered set will sometimes be called a *pre-order category* to emphasize its categorical nature.

2.10. Definition. [Order-preserving functions] *Let $(\Omega_1, \leq_1)$ and $(\Omega_2, \leq_2)$ be two pre-ordered sets. We shall speak of an* order-preserving function *from $(\Omega_1, \leq_1)$ to $(\Omega_2, \leq_2)$ to refer to a function $f : \Omega_1 \to \Omega_2$ for which every relation $x \leq_1 y$ in Ω_1 gives rise to a relation $f(x) \leq_2 f(y)$ in Ω_2.*

2.11. Convention. [Notation] *We shall denote by* **pOrd** *the category whose objects are pre-ordered sets and whose morphisms are order-preserving functions.*

2.12. Example. [Projection] For every positive integer n, the n-fold Cartesian product $\{0,1\}^{\times n}$ of Example 2.7 is equipped with a canonical collection of n functions $\pi_i : \{0,1\}^{\times n} \to \{0,1\}$, for each $i \in \{1,\ldots,n\}$, where a function π_i sends a tuple $(x_1,\ldots,x_n)$ in $\{0,1\}^{\times n}$ to its i-th component x_i in $\{0,1\}$. These functions obviously preserve the order relations of $\{0,1\}^{\times n}$ in $\{0,1\}$ and thus define morphisms in **pOrd**.

2.13. Finite sets of integers. For every positive integer n, we will denote by $[n]$ the finite set of integers $\{1,2,\ldots,n\}$. We will also let $[0]$ denote the empty set. In the sequel, for every non-negative integer n, the set $[n]$ will implicitly be equipped with the order associated with the set of integers (note that the order associated with $[0]$ is the empty order).

2.14. Segments. Let $(\Omega, \preceq)$ denote a pre-ordered set. A *segment* over Ω consists of a pair of non-negative integers (n_1, n_0), an order-preserving surjection[3] $t : [n_1] \to [n_0]$ and a function $c : [n_0] \to \Omega$.

2.15. Remark. [Representation] Segments have all the necessary data to encode the type of pictures given in (2). For a segment (t, c) as defined above, the finite set $[n_1]$ represents the range of elements composing the segment

$$n_1 = \bullet\bullet\cdots\bullet$$

while the fibers $t^{-1}(1),\ldots,t^{-1}(n_0)$ of the surjection $t : [n_1] \to [n_0]$ gather these elements into patches (see the brackets below).

$$t = (\bullet\bullet\bullet)(\bullet\bullet\bullet\bullet)(\bullet\bullet\cdots\bullet)(\bullet\bullet)$$

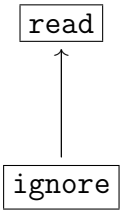
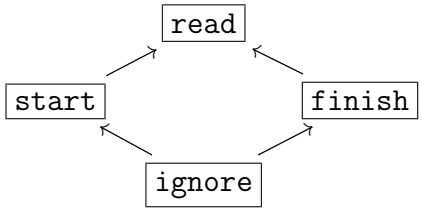
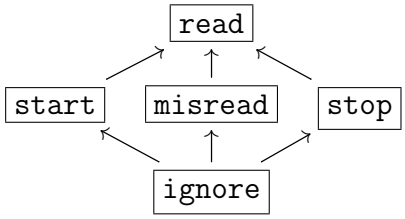
Finally, the different colors associated with the patches of the segment are specified by the map $c : [n_0] \to \Omega$. For instance, if we take Ω to be the Boolean pre-ordered set

[3] *i.e.* an order-preserving function that is a surjection.

$\{\texttt{false} \leq \texttt{true}\}$ of Example 2.4 (see Remark 2.5) and we choose to associate the white color with the `false` value and the black color with the `true` value, then a set of relations of the form $c(1) = \texttt{false}$, $c(2) = \texttt{true}$, ..., $c(n_0 - 1) = \texttt{true}$, and $c(n_0) = \texttt{true}$ will be represented by coloring all the elements of $[n_1]$ living in the fibers $t^{-1}(1)$, $t^{-1}(2)$, ..., $t^{-1}(n_0 - 1)$, and $t^{-1}(n_0)$ in white and then in black up to the last one, as shown below.

$$(t, c) = \ (\circ\circ\circ)(\bullet\bullet\bullet\bullet)(\bullet\bullet\cdots\bullet)(\bullet\bullet)$$

Note that if Ω contains more elements, then we need to use more colors (which can also be represented by numbers). These colors could also mean all sorts of things, including actions such as `ignore`, `read`, `start reading`, `stop reading`, `misread` (or `mutate`). The pre-order on the colors would then specify semantic priorities between the different tasks or functions associated with the colors (see the table of pictures below). All these features will be illustrated throughout the examples and remarks of section 3.

2 colors	4 colors	5 colors
$\{0, 1\}$	$\{0, 1, 2, 3\}$	$\{0, 1, 2, 3, 4\}$
		

2.16. Remark. [Notations] Note that the specification of the data n_1 and n_0 is redundant with the data of the function t and c. Later on, a segment will often be denoted as a pair (t, c) and, every so often, as an arrow $(t, c) : [n_1] \multimap [n_0]$.

2.17. Convention. [Domains, topologies & types] *For every segment $(t, c) : [n_1] \multimap [n_0]$, the data $[n_1]$ will be called the* domain *of (t, c), the data t will be called the* topology *of (t, c) and the data (n_1, n_0) will be called the* type *of (t, c). The type of a segment will always be specified as an arrow of the form $[n_1] \multimap [n_0]$.*

2.18. Definition. [Homologous segments] *Two segments (t, c) and (t', c') over Ω will be said to be* homologous *if their topologies t and t' are equal.*

2.19. Definition. [Quasi-homologous segments] *Two segments $(t, c) : [n_1] \multimap [n_0]$ and $(t', c') : [n'_1] \multimap [n'_0]$ over Ω will be said to be* quasi-homologous *if their domains $[n_1]$ and $[n'_1]$ are equal.*

2.20. Morphisms of segments. Let $(\Omega, \preceq)$ be a pre-ordered set and $(t, c) : [n_1] \multimap [n_0]$ and $(t', c') : [n'_1] \multimap [n'_0]$ be two segments over Ω. A morphism of segments from (t, c) to (t', c') consists of

1) an order-preserving injection $f_1 : [n_1] \to [n'_1]$;

2) an order-preserving function $f_0 : [n_0] \to [n'_0]$;

such that the inequality $c' \circ f_0(i) \preceq c(i)$ holds for every $i \in [n_0]$ and the following diagram commutes.

$$\begin{array}{ccc} [n_1] & \xrightarrow{t} & [n_0] \\ {\scriptstyle f_1}\downarrow & & \downarrow{\scriptstyle f_0} \\ [n'_1] & \xrightarrow{t'} & [n'_0] \end{array}$$

It is easy to check that the class of morphisms of segments over Ω is stable under component-wise compositions and admits identities on every segment. We will denote by $\mathbf{Seg}(\Omega)$ the resulting category whose objects are segments over Ω and whose arrows are morphisms between these.

From now on, we will regard the notations f_1 and f_0 given above as a conventional notation for morphisms in $\mathbf{Seg}(\Omega)$. Below, we give several examples of typical morphisms in $\mathbf{Seg}(\Omega)$ where Ω is taken to be the Boolean pre-ordered set of Example 2.4.

2.21. Example. [Locality] If both components f_1 and f_0 are identities, then the inequality $c' \circ f_0 \preceq c$ 'decreases' the colors of the segment as illustrated below, on the left.

$$\begin{array}{c} (\bullet\bullet\bullet)(\bullet\bullet)(\bullet\bullet\bullet\bullet)(\bullet\bullet\bullet\bullet\bullet)(\circ\circ\circ)(\bullet) \\ \downarrow \\ (\circ\circ\circ)(\circ\circ)(\bullet\bullet\bullet\bullet)(\bullet\bullet\bullet\bullet\bullet)(\circ\circ\circ)(\circ) \end{array} \qquad \begin{array}{c} (\ldots)(\circ\circ\circ)(\ldots) \\ \not\downarrow \\ (\ldots)(\bullet\bullet\bullet)(\ldots) \end{array}$$

Interpretation: This type of morphisms tells us that one is able to select/cut local patches from a segment. This is, for instance, the type of morphisms that one may want to use to model CRISPR, namely separating a patch from a segment. Note that, because reading a segment (in black) has a higher semantic priority than ignoring it (in white), turning white regions into black ones, as shown above, on the right, is forbidden. The order relation on the colors can therefore be a way of encoding forgetful operations (*e.g.* irreversible or energy-releasing events).

2.22. Example. [Relativity] If only the component f_1 is an identity morphism, then the component f_0 can merge the regions defining the topology.

$$\begin{array}{c} (\bullet\bullet\bullet)(\circ\circ)(\bullet\bullet\bullet\bullet)(\bullet\bullet\bullet\bullet\bullet)(\circ\circ\circ)(\circ) \\ \downarrow \\ (\circ\circ\circ\circ\circ)(\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet)(\circ\circ\circ\circ) \end{array}$$

Interpretation: This type of morphisms implies that the way one parses the patches of a segment influences the way one parses the whole segment (*e.g.* from codons to genes). However, because there is no arrow that increases the number of brackets from its domain to its codomain, the way one parses a segment might not necessarily reflect the way the patches are parsed (*e.g.* from gene to codons).

2.23. Example. [Flexibility] If the component f_1 is not an identity morphism, then the range of the segment increases. Below, we suppose that the identity $c' \circ f_0 = c$ holds.

$$(\bullet\bullet\bullet)(\circ\circ)(\bullet\bullet\bullet\bullet)(\bullet\bullet\bullet\bullet\bullet)(\circ\circ\circ)(\circ)$$
$$\downarrow$$
$$(\bullet\bullet\bullet\bullet)(\bullet)(\circ\circ\circ\circ)(\bullet\bullet\bullet\bullet)(\bullet\bullet\bullet\bullet\bullet)(\circ\circ\circ)(\circ)$$

Interpretation: This type of morphisms allows one to insert particular nucleotides or spaces in the parsing of a segment. For instance, spaces become necessary if one wants to align segments that are not necessarily (quasi-)homologous (this was shown in section 2.1). A morphism inserting a space would then correspond to a choice of 'sequence alignment' in bioinformatics (see Example 3.16, Example 3.38 and section 4.15).

2.24. Remark. [Initial object] For every pre-ordered set $(\Omega, \preceq)$, the segment (over Ω) of type $[0] \multimap [0]$ that is given by the obvious order-preserving surjection $! : \emptyset \to \emptyset$ and the canonical function $! : \emptyset \to \Omega$ is an initial object in $\mathbf{Seg}(\Omega)$. Note that such an object is formal and does not really possess any biological interpretation other than giving a way to express the idea of 'absence'.

2.25. Relating categories of segments. So far, our examples have only considered categories of segments over the Boolean pre-ordered set $\{0 \leq 1\}$. In practice, Boolean segments are convenient and easy to think about. Thus, it can be useful to have ways to go from a category of segments whose pre-ordered set is not $\{0 \leq 1\}$ to the category of segments whose pre-ordered is $\{0 \leq 1\}$. The goal of Proposition 2.26 is to show that this type of transfer is possible.

2.26. Proposition. [Functor] *Let $f : (\Omega_1, \preceq_1) \to (\Omega_2, \preceq_2)$ be a morphism in* **pOrd**. *The mapping rule $(t, c) \mapsto (t, f \circ c)$ extends to a faithful functor $\mathbf{Seg}(f) : \mathbf{Seg}(\Omega_1) \to \mathbf{Seg}(\Omega_2)$ sending a morphism (f_1, f_0) to the same pair (f_1, f_0).*

Proof. Let $(f_1, f_0) : (t, c) \to (t', c')$ be an arrow in $\mathbf{Seg}(\Omega_1)$ and let $[n_1]$ denote the domain of (t, c). Because f is an order-preserving function, the relation $c' \circ f_0(i) \preceq_1 c(i)$, satisfied for every element $i \in [n_1]$, gives rise to a relation $f \circ c' \circ f_0(i) \preceq_2 f \circ c(i)$, for every element $i \in [n_1]$. Since the domain of the segment $(t, f \circ c)$ is also $[n_1]$, the previous relation shows that the pair (f_1, f_0) defines a representative for an arrow of the form $(t, f \circ c) \to (t', f \circ c')$ in $\mathbf{Seg}(\Omega_2)$. The faithfulness property as well as the composition and identity axioms follow easily. ∎

In fact, Proposition 2.26 hides a functor structure on the category of pre-ordered sets, but this structure will not be needed in this paper.

2.27. Example. [Preparation example] The present example is the first of a series that address the goals presented in section 2.1. Let $(\Omega, \preceq)$ denote the Boolean pre-ordered set $\{0 \leq 1\}$. We follow the notation of Definition 2.7 and denote the pre-ordered set $\{0, 1\}^{\times 4}$ as $\Omega^{\times 4}$. Since $\Omega^{\times 4}$ is the 4-fold Cartesian product of Ω, it is equipped with four

order-preserving functions $\pi_i : \Omega^{\times 4} \to \Omega$ (see Example 2.12), which we purposely index with $i \in \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$. Later, each of these indices will be used to represent one of the four individuals of section 2.1. Now, by Proposition 2.26, a morphism $\pi_i : \Omega^{\times 4} \to \Omega$ induces a functor as follows.

$$\mathbf{Seg}(\pi_i) : \mathbf{Seg}(\Omega^{\times 4}) \to \mathbf{Seg}(\Omega)$$

If we represent an element $(\mathtt{x_a}, \mathtt{x_b}, \mathtt{x_c}, \mathtt{x_d})$ in $\Omega^{\times 4}$ as a word $[\mathtt{x_a x_b x_c x_d}]$, then the functor $\mathbf{Seg}(\pi_i)$ satisfies the following mapping rules for the various values of i shown in the rightmost column.

$\mathbf{Seg}(\Omega^{\times 4})$	$\longrightarrow$	$\mathbf{Seg}(\Omega)$	i
([1010][1010])([0110][0110][0110])([1111][1111])	$\mapsto$	(●●)(○○○)(●●)	$i = \mathtt{a}$
([1010][1010])([0110][0110][0110])([1111][1111])	$\mapsto$	(○○)(●●●)(●●)	$i = \mathtt{b}$
([1010][1010])([0110][0110][0110])([1111][1111])	$\mapsto$	(●●)(●●●)(●●)	$i = \mathtt{c}$
([1010][1010])([0110][0110][0110])([1111][1111])	$\mapsto$	(○○)(○○○)(●●)	$i = \mathtt{d}$

In the sequel, the category of segments $\mathbf{Seg}(\Omega^{\times 4})$ will be used as a logic to reason about our main example presented in section 2.1.

2.28. Pre-orders on homologous segments. Let $(\Omega, \preceq)$ be a pre-ordered set and let $t : [n_1] \to [n_0]$ be an order-preserving surjection. The subcategory of $\mathbf{Seg}(\Omega)$ whose objects are the homologous segments of topology t and whose arrows are the morphisms of segments for which the components f_0 and f_1 are identities will be denoted by $\mathbf{Seg}(\Omega : t)$ and referred to as the *category of homologous segments (over Ω) of topology t*.

2.29. Proposition. [Pre-order category] *For every order-preserving surjection $t : [n_1] \to [n_0]$, the category $\mathbf{Seg}(\Omega : t)$ is a pre-order category.*

Proof. According to section 2.20 and the definition of $\mathbf{Seg}(\Omega : t)$, giving an arrow $(t, c) \to (t, c')$ in $\mathbf{Seg}(\Omega : t)$ amounts to giving a pre-order relation $c'(i) \preceq c(i)$ in $(\Omega, \preceq)$ for every $i \in [n_0]$. It is straightforward to see that this defines a reflexive and transitive binary relation. ∎

2.30. Pre-orders on quasi-homologous segments. Let $(\Omega, \preceq)$ be a pre-ordered set and let n_1 be a non-negative integer. The subcategory of $\mathbf{Seg}(\Omega)$ whose objects are the quasi-homologous segments of domain $[n_1]$ and whose arrows are the morphisms of segments for which the component f_1 is an identity will be denoted by $\mathbf{Seg}(\Omega \,|\, n_1)$ and called the *category of quasi-homologous segments (over Ω) of domain n_1*.

2.31. Proposition. [Pre-order category] *For every non-negative integer n_1, the category of quasi-homologous segments $\mathbf{Seg}(\Omega \,|\, n_1)$ is a pre-order category.*

Proof. Let $(\mathrm{id}, f_0) : (t, c) \to (t', c')$ and $(\mathrm{id}, g_0) : (t, c) \to (t', c')$ be two morphisms in $\mathbf{Seg}(\Omega \,|\, n_1)$. We want to show that these morphisms are equal. According section 2.20 and the definition of $\mathbf{Seg}(\Omega \,|\, n_1)$, the two identities $f_0 \circ t = t'$ and $g_0 \circ t = t'$ hold, which implies that the identity $g_0 \circ t = f_0 \circ t$ holds. Because t is an epimorphism, the identity $g_0 = f_0$ must hold. ∎

2.32. Remark. [Zero domain] The category $\mathbf{Seg}(\Omega \,|\, 0)$ of quasi-homologous segments with empty domain [0] is a terminal category whose only object is the initial object of $\mathbf{Seg}(\Omega)$.

2.33. Cones. Recall that a *cone* in a category $\mathcal{C}$ consists of an object X in $\mathcal{C}$, a small category A, a functor $F : A \to \mathcal{C}$ and a natural transformation $\Delta_A(X) \Rightarrow F$ where $\Delta_A(X)$ denotes the constant functor $A \to \mathbf{1} \to \mathcal{C}$ mapping every object in A to the object X in $\mathcal{C}$.

2.34. Definition. [Wide spans] *In the sequel, we shall speak of a* wide span *to refer to a cone $\Delta_A(X) \Rightarrow F$ defined over a finite discrete small category A whose objects are ordered with respect to a total order (this will allow us to have canonical choices of limit constructions).*

2.35. Example. [Wide spans] Giving a wide span in a category $\mathcal{C}$ amounts to giving a finite collection of arrows $\mathbf{S} := \{f_i : X \to F_i\}_{i\in[n]}$ in $\mathcal{C}$. When the category $\mathcal{C}$ has products, the implicit order of the set $[n] = \{1, \ldots, n\}$ can be used to give a specific representative to the product of the collection $\{F_i\}_{i\in[n]}$ in $\mathcal{C}$.

2.36. Chromologies. A *chromology* is a pre-ordered set $(\Omega, \preceq)$ that is equipped, for every non-negative integer n, with a set $D[n]$ of cones in the category $\mathbf{Seg}(\Omega \,|\, n)$. Such a chromology will later be denoted as a pair (Ω, D).

2.37. Remark. [Future examples] In section 3.39, we will see several examples of chromologies, which will be used throughout this article.

2.38. Logical systems. By a *logical system*, we mean a category $\mathcal{C}$ that is equipped with a subclass of its cones $\mathcal{W}$ (see section 2.33).

2.39. Remark. [Size matters] The only difference between a logical system and a limit sketch is the sizes of their collections of objects: that of the latter is a set while that of the former is a class. This does make a difference in the type of properties that the two definitions satisfy. Because of their sizes, logical systems will only be used as codomains of functors. On the other hand, a chromology, which is a limit sketch, will often turn out to be the domain of a functor.

2.40. Pedigrads. Pedigrads are algebraic structures that model the logical rules of chromologies. Their name refers to the concept of 'pedigree' used in genetics to draw the genealogy of a set of taxa. Let (Ω, D) be a chromology and $(\mathcal{C}, \mathcal{W})$ be a logical system. A *pedigrad in* $(\mathcal{C}, \mathcal{W})$ *for* (Ω, D) is a functor $\mathbf{Seg}(\Omega) \to \mathcal{C}$ sending, for every non-negative integer n, the cones in $D[n]$ to cones in $\mathcal{W}$.

2.41. Convention. [$\mathcal{W}$-pedigrads] *Because we will often consider the same category $\mathcal{C}$ for different classes of cones $\mathcal{W}$, we will often refer to a pedigrad in $(\mathcal{C}, \mathcal{W})$ as a $\mathcal{W}$-pedigrad.*

3. Examples of pedigrads in sets

The goal of the present section is to formalize the concept of sequence alignment in terms of a functor on a subcategory of segments and to show that the right Kan extension of this functor on the whole category of segments computes what can be seen as multiple sequence alignments. The consistency of the integrated data is then studied by using the concepts of pedigrad and chromology.

3.1. Truncation functors. In this section, we define a truncation operation (Definition 3.2) and show that this operation is a functor on a category of quasi-homologous segments (Proposition 3.7). Extending this functoriality property to the whole category of segments is not straightforward and requires a few more steps (see Proposition 3.8). In section 3.11, we use the resulting functor to construct pedigrads.

3.2. Definition. [Truncation] *For every segment* $(t, c) : [n_1] \multimap [n_0]$ *over* Ω *and element* $b \in \Omega$*, we will denote by* $\mathsf{Tr}_b(t, c)$ *the subset* $\{i \in [n_1] \mid b \preceq c \circ t(i)\}$ *of* $[n_1]$*. This is the set of all elements in* $[n_1]$ *whose images via* $c \circ t$ *are greater than or equal to* b *in* Ω*.*

3.3. Example. [Truncation] Let $(\Omega, \preceq)$ be the Boolean pre-ordered set $\{0 \leq 1\}$. If we consider the segment (t, c) of $\mathbf{Seg}(\Omega)$ given below on the left, then the operation Tr_b for which b is taken to be equal to 1 will only select the integers in the domain of (t, c) that are associated with black nodes. On the other hand, the operation Tr_b for which b is taken to be equal to 0 will select all the integers in the domain of (t, c).

$$(t, c) = \ (\bullet\bullet\bullet)(\circ\circ)(\bullet\bullet\bullet\bullet)(\circ\circ\circ\circ\circ)(\bullet\bullet\bullet)(\circ) \qquad \begin{array}{l} \mathsf{Tr}_1(t, c) = \{1, 2, 3, 6, 7, 8, 9, 15, 16, 17\} \\ \mathsf{Tr}_0(t, c) = [18] \end{array}$$

Similarly, if we let $(\Omega, \preceq)$ denote the pre-ordered set $\{0 \leq 1 \leq 2\}$ (used in Remark 3.26), then we obtain the following truncations for the segment (t, c) given below, on the left.

$$(t, c) = \ (\mathtt{111})(\mathtt{00})(\mathtt{2222})(\mathtt{00000})(\mathtt{111})(\mathtt{0}) \qquad \begin{array}{l} \mathsf{Tr}_2(t, c) = \{6, 7, 8, 9\} \\ \mathsf{Tr}_1(t, c) = \{1, 2, 3, 6, 7, 8, 9, 15, 16, 17\} \\ \mathsf{Tr}_0(t, c) = [18] \end{array}$$

Here, the reader may have noticed that any relation of the form $b \preceq b'$ will lead to an inclusion of the form $\mathsf{Tr}_{b'}(t, c) \subseteq \mathsf{Tr}_b(t, c)$. This property, even though interesting, is not used in this paper.

3.4. Definition. [Sub-objects] *For every non-negative integer* n*, we will speak of a* sub-object *of* $[n]$ *to refer to a subset of* $[n]$*. A* morphism of sub-objects of $[n]$ *is an inclusion of sets between the two sub-objects.*

3.5. Example. [Truncation operations and sub-objects] Let $(\Omega, \preceq)$ be the Boolean pre-ordered set $\{0 \leq 1\}$. If we consider the morphism of segments that is given in Example 2.22, which we recall below, on the left-hand side, we can see that the truncation operation Tr_1 gives, on the right, two sub-objects of the domain [18] that can be related via a morphism of sub-objects.

$$\begin{array}{rl} (t,c) = & (\bullet\bullet\bullet)(\circ\circ)(\bullet\bullet\bullet\bullet)(\bullet\bullet\bullet\bullet\bullet)(\circ\circ\circ)(\circ) \\ & \downarrow \\ (t',c') = & (\circ\circ\circ\circ\circ)(\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet\bullet)(\circ\circ\circ\circ) \end{array} \qquad \begin{array}{l} \mathsf{Tr}_1(t,c) = \{1,2,3,6,7,8,9,10,11,12,13,14\} \\ \qquad\qquad\qquad\qquad \cup\mathrel{|} \\ \mathsf{Tr}_1(t',c') = \{6,7,8,9,10,11,12,13,14\} \end{array}$$

The fact that a morphism of segments of the form $(t,c) \to (t',c')$ gives rise to an inclusion $\mathsf{Tr}_1(t',c') \subseteq \mathsf{Tr}_1(t,c)$ is explained by Lemma 3.6.

3.6. Lemma. *Let $(f_1, f_0) : (t,c) \to (t',c')$ be a morphism in $\mathbf{Seg}(\Omega)$. If the relation $f_1(i) \in \mathsf{Tr}_b(t',c')$ holds, then so does the relation $i \in \mathsf{Tr}_b(t,c)$.*

Proof. Recall that, by definition of a morphism in $\mathbf{Seg}(\Omega)$, the inequality $c' \circ f_0 \preceq c$ holds. Now, if the relation $f_1(i) \in \mathsf{Tr}_b(t',c')$ holds, then so do the following pre-order relations.

$$b \preceq c' \circ t' \circ f_1(i) = c' \circ f_0 \circ t(i) \preceq c \circ t(i)$$

By transitivity, we obtain the inequality $b \preceq c \circ t(i)$, which implies that i must be in $\mathsf{Tr}_b(t,c)$. ∎

The following proposition only shows one side of the functorial properties satisfied by the truncation operation. Proposition 3.8 will give a different functor construction, which is related to that given in Proposition 3.7 via the statement of Proposition 3.10. While Proposition 3.8 will later be used to construct pedigrads, Proposition 3.7 will be used to deduce properties related to them.

3.7. Proposition. *For every element $b \in \Omega$ and non-negative integer n_1, the mapping $(t,c) \mapsto \mathsf{Tr}_b(t,c)$ extends to a functor $\mathsf{Tr}_b : \mathbf{Seg}(\Omega \,|\, n_1) \to \mathbf{Set}^{\mathrm{op}}$, which factorizes through the opposite category of sub-objects of $[n_1]$.*

Proof. By definition, for every segment (t,c) in $\mathbf{Seg}(\Omega \,|\, n_1)$, the set $\mathsf{Tr}_b(t,c)$ is a subset of $[n_1]$. For every morphism $(\mathrm{id}, f_0) : (t,c) \to (t,c')$ in $\mathbf{Seg}(\Omega \,|\, n_1)$, Lemma 3.6 shows that there is an inclusion $\mathsf{Tr}_b(t,c') \subseteq \mathsf{Tr}_b(t,c)$. Since the opposite category of sub-objects of $[n_1]$ is a pre-order category, the functor structure is obvious and the statement follows. ∎

The extension of the functorial property given in Proposition 3.7 to the whole category of segments requires to change the codomain category $\mathbf{Set}$ to the category $\mathbf{Set}_*$ of pointed sets and point-preserving maps (see Example 3.9). In this respect, recall that there is an adjunction

$$\mathbf{Set} \underset{\mathbb{U}}{\overset{\mathbb{F}}{\underset{\longleftarrow}{\overset{\longrightarrow}{\bot}}}} \mathbf{Set}_*$$

whose right adjoint $\mathbb{U} : \mathbf{Set}_* \to \mathbf{Set}$ forgets the pointed structure (*i.e.* $\mathbb{U} : (X, p) \mapsto X$) and whose left adjoint $\mathbb{F} : \mathbf{Set} \to \mathbf{Set}_*$ maps a set X to the obvious pointed set $(X+\{*\}, *)$ and maps a function $f : X \to Y$ to the coproduct map $f + \{*\} : X + \{*\} \to Y + \{*\}$.

3.8. Proposition. *For every element $b \in \Omega$, the mapping $(t, c) \mapsto \mathbb{F}\mathsf{Tr}_b(t, c)$ extends to a functor $\mathsf{Tr}_b^* : \mathbf{Seg}(\Omega) \to \mathbf{Set}_*^{\mathrm{op}}$ mapping every function $(f_1, f_0) : (t, c) \to (t', c')$ in $\mathbf{Seg}(\Omega)$ to the following map of pointed sets.*

$$\begin{array}{rcccll} \mathsf{Tr}_b^*(f_1, f_0) & : & \mathbb{F}\mathsf{Tr}_b(t', c') & \to & \mathbb{F}\mathsf{Tr}_b(t, c) & \\ & & j & \mapsto & i & \text{if } \exists i \in \mathsf{Tr}_b(t, c) \,:\, j = f_1(i); \\ & & j & \mapsto & * & \text{otherwise.} \end{array}$$

Proof. The well-definedness of the point-preserving map $\mathsf{Tr}_b^*(f_1, f_0)$ follows from Lemma 3.6. The mapping Tr_b^* obviously satisfies the identity axiom associated with the concept of a functor. The composition axiom is shown as follows. Take two morphisms $(f_1, f_0) : (t, c) \to (t', c')$ and $(f_1', f_0') : (t', c') \to (t'', c'')$ in $\mathbf{Seg}(\Omega)$. The image of the composition $(f_1', f_0') \circ (f_1, f_0)$ via the operation Tr_b^* takes the following form.

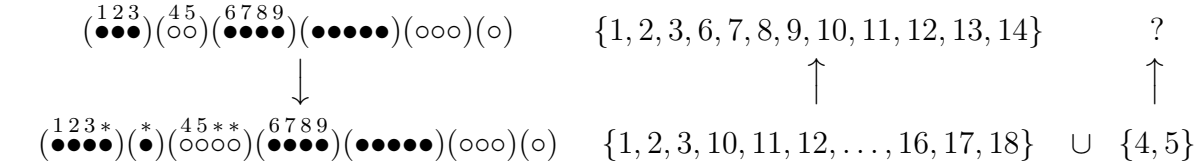
$$\begin{array}{rcll} j & \mapsto & (f_1' \circ f_1)^{-1}(j) & \text{if } (f_1' \circ f_1)^{-1}(j) \neq \emptyset \\ j & \mapsto & * & \text{otherwise.} \end{array}$$

Since the identity $(f_1' \circ f_1)^{-1}(j) = (f_1)^{-1}((f_1')^{-1}(j))$ holds whenever the inequality $(f_1')^{-1}(j) \neq \emptyset$ is satisfied, we deduce that the following equation holds.

$$\mathsf{Tr}_b^*((f_1', f_0') \circ (f_1, f_0)) = \mathsf{Tr}_b^*(f_1, f_0) \circ \mathsf{Tr}_b^*(f_1', f_0')$$

This last equation shows that Tr_b^* is a functor going to the opposite category of $\mathbf{Set}_*$. ∎

3.9. Example. [Truncation operations and pointed sets] Let $(\Omega, \preceq)$ be the Boolean pre-ordered set $\{0 \leq 1\}$. Consider the morphism of segments of Example 2.23, given below, on the left. Its mapping has further been detailed by using adequate labeling to show how the first segment is mapped to the second one. On the right, we can see its image via the truncation operation Tr_1 where we see that the indices 4 and 5, in the truncated codomain, do not have corresponding indices in the truncated domain.



A way to associate the indices 4 and 5 with an element in the truncated domain is to formally add one, thus explaining the pointed structure used in Proposition 3.8.

3.10. Proposition. *For every element $b \in \Omega$ and non-negative integer n_1, the following diagram commutes.*

$$\begin{array}{ccc} \mathbf{Seg}(\Omega \,|\, n_1) & \xrightarrow{\subseteq} & \mathbf{Seg}(\Omega) \\ {\scriptstyle \mathsf{Tr}_b}\downarrow & & \downarrow{\scriptstyle \mathsf{Tr}_b^*} \\ \mathbf{Set}^{\mathrm{op}} & \xrightarrow[\mathbb{F}^{\mathrm{op}}]{} & \mathbf{Set}_*^{\mathrm{op}} \end{array}$$

Proof. By definition, if we restrict the functor $\mathsf{Tr}_b^* : \mathbf{Seg}(\Omega) \to \mathbf{Set}_*^{\mathrm{op}}$ to the subcategory $\mathbf{Seg}(\Omega \,|\, n_1) \hookrightarrow \mathbf{Seg}(\Omega)$, then every morphism $(t, c) \to (t, c')$ in the pre-order category $\mathbf{Seg}(\Omega \,|\, n_1)$ is sent to the following map in $\mathbf{Set}_*$ (see Proposition 3.8).

$$\begin{array}{rcccll} \mathsf{Tr}_b^*(f_1, f_0) & : & \mathbb{F}\mathsf{Tr}_b(t, c') & \to & \mathbb{F}\mathsf{Tr}_b(t, c) & \\ & & j & \mapsto & j & j \in \mathsf{Tr}_b(t, c') \\ & & * & \mapsto & * & \text{otherwise.} \end{array}$$

This means that the restriction of Tr_b^* on $\mathbf{Seg}(\Omega \,|\, n_1)$ can be retrieved from the application of the functor $\mathbb{F}$ on the images of Tr_b. ∎

3.11. Examples of pedigrads in sets. In this section, we construct a collection of functors $\mathbf{Seg}(\Omega) \to \mathbf{Set}$ for any pointed set (E, ε) and parameter b in Ω (see Definition 3.13). Later on, we will define various classes of cones $\mathcal{W}$ in $\mathbf{Set}$ for which these functors are $\mathcal{W}$-pedigrad (see Theorem 3.47 and Theorem 3.49).

3.12. Convention. [Notation] *In the sequel, the hom-set of a category $\mathcal{C}$ from an object X to an object Y will be denoted as $\mathcal{C}(X, Y)$. For instance, the set of functions from a set X to a set Y will be denoted by $\mathbf{Set}(X, Y)$. Also, recall that, for any category $\mathcal{C}$, the hom-sets give rise to a functor $\mathcal{C}(\_, \_) : \mathcal{C}^{\mathrm{op}} \times \mathcal{C} \to \mathbf{Set}$ called the* hom-functor *[12, page 27].*

3.13. Definition. [Environment functors] *For every element $b \in \Omega$, we will denote by E_b^ε the functor $\mathbf{Seg}(\Omega) \to \mathbf{Set}$ defined as the composition of the following pair of functors.*

$$\mathbf{Seg}(\Omega) \xrightarrow{\mathsf{Tr}_b^*} \mathbf{Set}_*^{\mathrm{op}} \xrightarrow{\mathbf{Set}_*(\_, (E, \varepsilon))} \mathbf{Set}$$

3.14. Remark. For every object (t, c) in $\mathbf{Seg}(\Omega)$, an element in $E_b^\varepsilon(t, c)$ can be seen as a function of the form $\mathsf{Tr}_b(t, c) \to E$ according to the following series of bijections.

$$\begin{array}{rll} E_b^\varepsilon(t, c) & = \mathbf{Set}_*(\mathsf{Tr}_b^*(t, c), (E, \varepsilon)) & \\ & = \mathbf{Set}_*(\mathbb{F}\mathsf{Tr}_b(t, c), (E, \varepsilon)) & (\text{Definition of } \mathsf{Tr}_b^*) \\ & \cong \mathbf{Set}(\mathsf{Tr}_b(t, c), \mathbb{U}(E, \varepsilon)) & (\mathbb{F} \dashv \mathbb{U}) \\ & = \mathbf{Set}(\mathsf{Tr}_b(t, c), E) & (\text{Definition of } \mathbb{U}) \end{array}$$

Because the set $\mathsf{Tr}_b(t, c)$ is equipped with the usual order of natural numbers, we will represent an element in $E_b^\varepsilon(t, c)$ as a word of elements in E (see Example 3.15).

3.15. Example. [Objects] Suppose that $(\Omega, \preceq)$ denotes the Boolean pre-ordered set $\{0 \leq 1\}$ and let (E, ε) be the pointed set $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}, \varepsilon\}$. If we consider the segment

$$(t, c) = \ (\bullet\bullet\bullet)(\circ\circ)(\bullet\bullet\bullet\bullet)(\bullet\bullet\bullet\bullet\bullet)(\circ\circ\circ)(\circ)$$

then the set $E_1^{\varepsilon}(t, c)$ (where $b = 1$) will contain the following words (which have been parenthesized for clarity), among many others.

$$(\mathtt{AG}\varepsilon)(\mathtt{TCAA})(\mathtt{TAGG}\varepsilon);$$
$$(\mathtt{GT}\varepsilon)(\varepsilon\varepsilon\varepsilon\mathtt{C})(\mathtt{AGTAC});$$
$$(\mathtt{TAA})(\mathtt{GATC})(\mathtt{AGTTT});$$
$$\text{etc.}$$

3.16. Example. [Morphisms] Suppose that Ω denotes the Boolean pre-ordered set $\{0 \leq 1\}$ and let (E, ε) be the pointed set $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}, \varepsilon\}$. If we consider the morphism of segments given below, in which we use adequate labeling to show how the first segment is included in the second one,

$$(\overset{1\,2\,3}{\bullet\bullet\bullet})(\overset{4\,5}{\circ\circ})(\overset{6\,7\,8\,9}{\bullet\bullet\bullet\bullet})(\overset{10\,11}{\bullet\ \bullet}) \ \to \ (\overset{1\,2\,3\,*\,*}{\bullet\bullet\bullet\bullet\bullet})(\overset{4\,5\,*}{\circ\circ\circ})(\overset{6\,7\,8\,9}{\bullet\bullet\bullet\bullet})(\overset{*}{\bullet})(\overset{10\,11}{\circ\ \circ})$$

then the image of the previous arrow via E_1^{ε} is a function whose mappings rules look as follows.

$$\begin{array}{rcl}(\mathtt{AG}\varepsilon)(\mathtt{TCAA})(\mathtt{GC}) & \mapsto & (\mathtt{AG}\varepsilon\varepsilon\varepsilon)(\mathtt{TCAA})(\varepsilon);\\ (\mathtt{GT}\varepsilon)(\varepsilon\varepsilon\varepsilon\mathtt{C})(\mathtt{TA}) & \mapsto & (\mathtt{GT}\varepsilon\varepsilon\varepsilon)(\varepsilon\varepsilon\varepsilon\mathtt{C})(\varepsilon);\\ (\mathtt{TAA})(\mathtt{GATC})(\mathtt{AA}) & \mapsto & (\mathtt{TAA}\varepsilon\varepsilon)(\mathtt{GATC})(\varepsilon);\end{array}$$
$$\text{etc.}$$

Note that if one restricts oneself to morphisms in $\mathbf{Seg}(\Omega)$ that only insert new nodes and do not turn any black node into white ones, then the mappings associated with the images of such morphisms can be seen as gap insertion operations. As seen in section 2.1, these operations are used in sequence alignment algorithms to compare sequences of different lengths together.

$$(\overset{1\,2\,3}{\bullet\bullet\bullet})(\overset{4\,5}{\circ\circ})(\overset{6\,7\,8\,9}{\bullet\bullet\bullet\bullet})(\overset{10\,11}{\bullet\ \bullet}) \ \to \ (\overset{1\,2\,3}{\bullet\bullet\bullet})(\overset{4\,5\,*}{\circ\circ\circ})(\overset{6\,7\,*\,*\,8\,9}{\bullet\bullet\bullet\bullet\bullet\bullet})(\overset{*\,10\,11}{\bullet\ \bullet\ \bullet})$$

$$(\mathtt{GAC})(\mathtt{ATTC})(\mathtt{CT}) \quad \mapsto \quad (\mathtt{GAC})(\mathtt{AT}\varepsilon\varepsilon\mathtt{TC})(\varepsilon\mathtt{CT});$$
$$\text{etc.}$$

3.17. Proposition. *For every domain* $[n_1]$*, the restriction of the functor* $E_b^{\varepsilon} : \mathbf{Seg}(\Omega) \to \mathbf{Set}$ *on* $\mathbf{Seg}(\Omega \,|\, n_1)$ *is isomorphic to the functor* $\mathbf{Set}(\mathsf{Tr}_b(\_), E) : \mathbf{Seg}(\Omega \,|\, n_1) \to \mathbf{Set}$*. In other words, the following diagram commutes up to an isomorphism of functors.*

$$\begin{array}{ccc}\mathbf{Seg}(\Omega \,|\, n_1) & \xrightarrow{\subseteq} & \mathbf{Seg}(\Omega)\\ {\scriptstyle \mathsf{Tr}_b}\downarrow & & \downarrow{\scriptstyle E_b^{\varepsilon}}\\ \mathbf{Set}^{\mathrm{op}} & \xrightarrow[\mathbf{Set}(\_,E)]{} & \mathbf{Set}\end{array}$$

PROOF. Note that the following series of isomorphisms hold on $\mathbf{Seg}(\Omega \,|\, n_1)$.

$$\begin{aligned} E_b^{\varepsilon}(\_) &= \mathbf{Set}_*\Big(\mathsf{Tr}_b^*(\_), (E,\varepsilon)\Big) \\ &= \mathbf{Set}_*\Big(\mathbb{F}\mathsf{Tr}_b(\_), (E,\varepsilon)\Big) && \text{(Proposition 3.10)} \\ &\cong \mathbf{Set}\Big(\mathsf{Tr}_b(\_), \mathbb{U}(E,\varepsilon)\Big) && (\mathbb{F} \dashv \mathbb{U}) \\ &= \mathbf{Set}\Big(\mathsf{Tr}_b(\_), E\Big) && \text{(Definition of } \mathbb{U}) \end{aligned}$$

Since these isomorphisms are natural on $\mathbf{Seg}(\Omega \,|\, n_1)$, the statement follows. ∎

3.18. SEQUENCE ALIGNMENTS. In this section, we use the functors defined in section 3.11 to formalize the concept of sequence alignment. The examples given in this section mainly focus on addressing our main example presented in section 2.1.

3.19. DEFINITION. [Alignment specification] *We shall speak of an* alignment specification *to refer to a wide span (Definition 2.34) in the category* **pOrd** *of pre-ordered sets.*

3.20. EXAMPLE. [Alignment specification] Let $(\Omega, \preceq)$ be the Boolean pre-ordered set $\{0 \leq 1\}$. If we use the notations of Example 2.27, then the following collection of morphisms in **pOrd** defines an alignment specification.

$$\{\pi_i : \Omega^{\times 4} \to \Omega\}_{i \in \{\mathtt{a},\mathtt{b},\mathtt{c},\mathtt{d}\}} \tag{3}$$

Here, the discrete category $\{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$ is implicitly ordered with respect to the alphabetic order and hence makes (3) a wide span as defined in Definition 2.34.

In general, alignment specifications do not necessarily need to be universal cones and the codomains of the arrows do not need be equal either. For instance, the following pair of wide spans define two valid alignment specifications.

$$\left\{ \begin{array}{lccc} \pi_{\mathtt{a}}: & \Omega^{\times 4} & \to & \Omega, \\ \pi_{\mathtt{c}}: & \Omega^{\times 4} & \to & \Omega, \\ \pi_{\mathtt{d}}: & \Omega^{\times 4} & \to & \Omega \end{array} \right\} \qquad \left\{ \begin{array}{lccc} \pi_{\mathtt{a}} \times \pi_{\mathtt{b}}: & \Omega^{\times 4} & \to & \Omega^{\times 2}, \\ \pi_{\mathtt{c}}: & \Omega^{\times 4} & \to & \Omega, \\ \pi_{\mathtt{d}}: & \Omega^{\times 4} & \to & \Omega \end{array} \right\}$$

In the next article [28], we will use alignment specifications made of identity morphisms.

3.21. DEFINITION. [Aligned pedigrads] *Let* $\mathbf{A} = \{f_i : (\Omega, \preceq) \to (\Omega_i, \preceq_i)\}_{i \in A}$ *be an alignment specification and* b *be an element in* Ω. *We denote by* $\mathbf{A}E_b^{\varepsilon}$ *the functor* $\mathbf{Seg}(\Omega) \to \mathbf{Set}$ *resulting from the composition of the three functors given in (4), where*

- *the rightmost functor is the obvious Cartesian functor of* **Set***;*
- *the middle functor is the Cartesian product of the functors* $E^{\varepsilon}_{f_i(b)} : \mathbf{Seg}(\Omega_i) \to \mathbf{Set}$*;*
- *and the leftmost functor is the product adjoint of the cone induced by the image of* $\mathbf{A}$ *via* **Seg** *(section 2.25).*

$$\mathbf{Seg}(\Omega) \xrightarrow{(\mathbf{Seg}(f_i))_{i \in A}} \prod_{i \in A} \mathbf{Seg}(\Omega_i) \xrightarrow{\prod_{i \in A} E^{\varepsilon}_{f_i(b)}} \prod_{i \in A} \mathbf{Set} \xrightarrow{\quad\times\quad} \mathbf{Set} \tag{4}$$

Such a functor will be called the alignment *of* E_b^{ε} *on* $\mathbf{A}$.

3.22. Example. [Aligned pedigrads] Let $(\Omega, \preceq)$ denote the Boolean pre-ordered $\{0 \leq 1\}$ and let $\mathbf{A}$ denote the alignment specification given in (3). As usual, we shall let (E, ε) denote the pointed set $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}, \varepsilon\}$. For any given segment (t, c) in $\mathbf{Seg}(\Omega^{\times 4})$ and element $b \in \Omega^{\times 4}$, the set $\mathbf{A}E_b^\varepsilon(t, c)$ is equal to the following Cartesian product of sets.

$$E^\varepsilon_{\pi_{\mathtt{a}}(b)}(t, \pi_{\mathtt{a}} \circ c) \times E^\varepsilon_{\pi_{\mathtt{b}}(b)}(t, \pi_{\mathtt{b}} \circ c) \times E^\varepsilon_{\pi_{\mathtt{c}}(b)}(t, \pi_{\mathtt{c}} \circ c) \times E^\varepsilon_{\pi_{\mathtt{d}}(b)}(t, \pi_{\mathtt{d}} \circ c)$$

The following table illustrates what the elements of the previous set look like for different segments (t, c) in $\mathbf{Seg}(\Omega^{\times 4})$ and a fixed value b in $\Omega^{\times 4}$. As usual, parentheses are added for clarity and the underscore symbols[4] are only used to represent spaces "to be filled out".

b	(t, c)	$\mathbf{A}E_b^\varepsilon(t, c)$
(1,1,1,1)	([1111][1111])([1111][1111][1111])	(AG)(CGT) (GT)(AAT) (CA)(AAC) (AT)(TCG); (CG)(GTε); (CC)(εAC); (Cε)(ATG); (AC)(TTG); (CT)(εCA); etc. (AT)(GGG) (εε)(TGC) (AC)(TTG)
	([1010][1010])([0110][0110][0110])	(AG) _ _ _ (GT) _ _ _ (CA) _ _ _ _ _ (TCG); _ _ (GTε); _ _ (εAC); etc. (Cε)(ATG) (AC)(TTG) (CT)(εCA)
	([0011][0011])([0011][0011][0011])	(Cε)(ATG); (AC)(TTG); (CT)(εCA); (AT)(GGG); (εε)(TGC); (AC)(TTG); etc.
	([1100][1100])([1100][1100][1100])	(AG)(CGT); (GT)(AAT); (CA)(AAC); (AT)(TCG); (CG)(GTε); (CC)(εAC); etc.

We can see that $\mathbf{A}E_b^\varepsilon(t, c)$ contains what we would like to understand as sequence alignments. It would therefore be natural to try to model our example, given in section 2.1, relatively to the functor $\mathbf{A}E_b^\varepsilon$. In what follows, we define the concept of sequence alignment relative to alignments of functors E_b^ε for some pointed set (E, ε), pre-ordered set Ω and element $b \in \Omega$.

The following definition formalizes the concept of sequence alignment in terms of a subcategory, a functor and a natural transformation. In Example 3.25, we will see how one can use this concept to reason about the problem given in section 2.1.

3.23. Definition. [Sequence alignment functors] *Let* $\mathbf{A}$ *be an alignment specification as given in Definition 3.21 and* b *be an element in* Ω*. We define a* sequence alignment functor over $\mathbf{A}E_b^\varepsilon$ *as a triple* (ι, T, σ) *where* ι *is an inclusion functor* $\iota : B \to \mathbf{Seg}(\Omega)$*,* T *is a functor* $B \to \mathbf{Set}$ *and* σ *is a natural monomorphism* $T \Rightarrow \mathbf{A}E_b^\varepsilon \circ \iota$*.*

3.24. Convention. [Notations] *In the sequel, it will be convenient to have short notations for the segments of the subcategory* $B \hookrightarrow \mathbf{Seg}(\Omega)$ *associated with a sequence alignment functor. In the context of the present article, most of our segments will have trivial*

[4]These underscore symbols should not be confused with the dash symbols that is sometimes used in bioinformatics to denote a substitution. Note that, in our case, the symbol ε already plays the role of the dash symbols.

topologies – given by the canonical surjection $!_{[n]} : [n] \to [1]$ *– and will hence be of the following form for some color* $b \in \Omega$*.*

$$(bbb...bb)$$

In this respect, we shall denote any segment whose topology t *is of the form* $!_{[n]} : [n] \to [1]$ *and whose function* $c : [1] \to \Omega$ *picks out an element* $b \in \Omega$ *as a pair* $(!_{[n]}, b)$*.*

3.25. Example. [Sequence alignment functors] The present example is a continuation of Example 3.22 and aims to illustrate the use of Definition 3.23 in the context of our problem introduced in section 2.1. We shall therefore use the same notations as those used in Example 3.22.

As mentioned at the end of Example 3.22, we want to model the example of section 2.1 relative to the functor $\mathbf{A}E_b^\varepsilon$ defined in the example. The idea is to pick out, via the concept of sequence alignment functors introduced in Definition 3.23, the pairwise sequence alignments living in the images of $\mathbf{A}E_b^\varepsilon$ outputted by the dynamic programming algorithm presented in section 2.1. In this respect, let us compute the sequence alignments of every pair of individuals given in section 2.1 by following the method described therein. Doing so, we obtain the following table of pairwise sequence alignments.

Pairs	Sequence alignments
Anne Bob	ACCGACTG ACCGACTG ACCGAεCTG ACCGACTG AεCATCTG ACAεTCTG AεCεATCTG ACATεCTG
Anne Craig	ACCGACTG ACCGACTG ACCGTCεA ACCGTCAε
Anne Doug	ACCGACTG ACCGACTG AεCTACTG ACTεACTG
Bob Craig	AεCATCTG ACAεTCTG AεCATCTG ACAεTCTG ACCGTCεA ACCGTCεA ACCGTCAε ACCGTCAε
Bob Doug	ACATCTG ACATεCTG ACεATCTG ACTACTG ACεTACTG ACTAεCTG
Craig Doug	ACCGTCA ACCGTCεA ACCGTCεA ACCGTεCεA ACCGTCεA ACεCGTCA ACTACTG AεCTACTG ACTεACTG AεCεTACTG ACTAεCTG ACTACTεG
	ACCεGTCA ACCGεTCA ACεεCGTCA ACCGTCAε ACCGTCAε ACCGTεCAε ACTACTεG ACTACTεG ACTACεTεG AεCTACTG ACTεACTG AεCεTACTG
	ACCGTCAε ACεCGTCA ACCεGTCA ACCGεTCA ACεεCGTCA ACTAεCTG ACTACTGε ACTACTGε ACTACTGε ACTACεTGε

If we now take b to be equal to the element $(1, 1, 1, 1)$ in $\Omega^{\times 4}$, the previous table can reasonably be seen as a 'part' of the functor $\mathbf{A}E_b^\varepsilon$ by interpreting each pairwise sequence alignment given above as an element in one of the images of $\mathbf{A}E_b^\varepsilon$ (see the table given in Example 3.22). In this example, we shall implement this 'part' by considering the sequence alignment functor (B, ι, T, σ) whose subcategory $B \hookrightarrow \mathbf{Seg}(\Omega)$ is the union

of the full subcategories of $\mathbf{Seg}(\Omega^{\times 4} \mid 7)$, $\mathbf{Seg}(\Omega^{\times 4} \mid 8)$ and $\mathbf{Seg}(\Omega^{\times 4} \mid 9)$ that contain the following segments:

$$\begin{array}{lllll} (!_{[7]}, [0101]) & (!_{[7]}, [0011]) & (!_{[7]}, [0001]) & (!_{[8]}, [1100]) & (!_{[8]}, [1010]) \\ (!_{[8]}, [1001]) & (!_{[8]}, [0110]) & (!_{[8]}, [0101]) & (!_{[8]}, [0011]) & (!_{[8]}, [0001]) \\ (!_{[8]}, [1000]) & (!_{[8]}, [0100]) & (!_{[8]}, [0010]) & (!_{[9]}, [0011]) & (!_{[9]}, [1100]) \end{array}$$

and whose monomorphism $T \Rightarrow \mathbf{A}E_b^{\varepsilon} \circ \iota$ picks out the pairwise sequence alignments shown in the previous table. Before making this last statement more precise, let us explain how the colors and the domains of the segments of B (shown above) will be used to organize the pairwise sequence alignments contained in the images of T. First, as already suggested in Example 2.27, we want to use a tuple $(\mathtt{x_a}, \mathtt{x_b}, \mathtt{x_c}, \mathtt{x_d})$ in $\Omega^{\times 4}$ to specify whether an individual is included in a sequence alignment or not by setting the variable indexed by the initial of the individual to 1 or 0, respectively. For instance, setting $\mathtt{x_a}$ to 1 would mean that `Anne` is part of the alignment computation. Second, we want to make the cardinality of the domains of the segments match the length of the pairwise sequence alignments with which they are associated. Thus, we could decide to encode the row of the previous table comparing `Anne` and `Bob` by taking the following images for T.

$$T(!_{[8]}, [1100]) = \left\{ \begin{matrix} \mathtt{ACCGACTG} \\ \mathtt{A}\varepsilon\mathtt{CATCTG} \end{matrix}, \begin{matrix} \mathtt{ACCGACTG} \\ \mathtt{ACA}\varepsilon\mathtt{TCTG} \end{matrix}, \begin{matrix} \mathtt{ACCGACTG} \\ \mathtt{ACAT}\varepsilon\mathtt{CTG} \end{matrix} \right\}$$

$$T(!_{[9]}, [1100]) = \left\{ \begin{matrix} \mathtt{ACCGA}\varepsilon\mathtt{CTG} \\ \mathtt{A}\varepsilon\mathtt{C}\varepsilon\mathtt{ATCTG} \end{matrix} \right\}$$

Similarly, the other images of T on segments containing exactly two symbols `1` could be taken as follows (the specification of the second last image, of cardinality 12, is left to the reader).

T
$T(!_{[8]}, [1010]) = \left\{ \begin{matrix} \mathtt{ACCGACTG} \\ \mathtt{ACCGTC}\varepsilon\mathtt{A} \end{matrix}; \begin{matrix} \mathtt{ACCGACTG} \\ \mathtt{ACCGTCA}\varepsilon \end{matrix} \right\}$
$T(!_{[8]}, [1001]) = \left\{ \begin{matrix} \mathtt{ACCGACTG} \\ \mathtt{A}\varepsilon\mathtt{CTACTG} \end{matrix}; \begin{matrix} \mathtt{ACCGACTG} \\ \mathtt{ACT}\varepsilon\mathtt{ACTG} \end{matrix} \right\}$
$T(!_{[8]}, [0110]) = \left\{ \begin{matrix} \mathtt{A}\varepsilon\mathtt{CATCTG} \\ \mathtt{ACCGTC}\varepsilon\mathtt{A} \end{matrix}; \begin{matrix} \mathtt{ACA}\varepsilon\mathtt{TCTG} \\ \mathtt{ACCGTC}\varepsilon\mathtt{A} \end{matrix}; \begin{matrix} \mathtt{A}\varepsilon\mathtt{CATCTG} \\ \mathtt{ACCGTCA}\varepsilon \end{matrix}; \begin{matrix} \mathtt{ACA}\varepsilon\mathtt{TCTG} \\ \mathtt{ACCGTCA}\varepsilon \end{matrix} \right\}$
$T(!_{[7]}, [0101]) = \left\{ \begin{matrix} \mathtt{ACATCTG} \\ \mathtt{ACTACTG} \end{matrix} \right\}$
$T(!_{[8]}, [0101]) = \left\{ \begin{matrix} \mathtt{ACAT}\varepsilon\mathtt{CTG} \\ \mathtt{AC}\varepsilon\mathtt{TACTG} \end{matrix}; \begin{matrix} \mathtt{AC}\varepsilon\mathtt{ATCTG} \\ \mathtt{ACTA}\varepsilon\mathtt{CTG} \end{matrix} \right\}$
$T(!_{[7]}, [0011]) = \left\{ \begin{matrix} \mathtt{ACCGTCA} \\ \mathtt{ACTACTG} \end{matrix} \right\}$
$T(!_{[8]}, [0011]) = \left\{ \begin{matrix} \mathtt{ACCGTC}\varepsilon\mathtt{A} \\ \mathtt{A}\varepsilon\mathtt{CTACTG} \end{matrix}; \ldots \right\}$
$T(!_{[9]}, [0011]) = \left\{ \begin{matrix} \mathtt{ACCGT}\varepsilon\mathtt{C}\varepsilon\mathtt{A} \\ \mathtt{A}\varepsilon\mathtt{C}\varepsilon\mathtt{TACTG} \end{matrix}; \begin{matrix} \mathtt{AC}\varepsilon\varepsilon\mathtt{CGTCA} \\ \mathtt{ACTAC}\varepsilon\mathtt{T}\varepsilon\mathtt{G} \end{matrix}; \begin{matrix} \mathtt{ACCGT}\varepsilon\mathtt{CA}\varepsilon \\ \mathtt{A}\varepsilon\mathtt{C}\varepsilon\mathtt{TACTG} \end{matrix}; \begin{matrix} \mathtt{AC}\varepsilon\varepsilon\mathtt{CGTCA} \\ \mathtt{ACTAC}\varepsilon\mathtt{TG}\varepsilon \end{matrix} \right\}$

In addition to these sets, we also want to include sets that enable us to compare the previous sequence alignments. For instance, the segments indexing the images $T(!_{[8]}, [1100])$ and $T(!_{[8]}, [0110])$ both have their coordinates $\mathtt{x_b}$ set to 1, so they should both be able to go to the image $T(!_{[8]}, [0100])$. An easy choice for such an image is to pick

$$T(!_{[8]}, [0100]) := \mathbf{A}E_b^{\varepsilon}(!_{[8]}, [0100])$$

so that we have a diagram of functions as shown in (5) by sending the DNA sequences of Bob – which constitute the bottom rows of the sequence alignments of $T(!_{[8]}, [1100])$ and the top rows of the sequence alignments of $T(!_{[8]}, [0110])$ – to the corresponding DNA sequences in $\mathbf{A}E_b^{\varepsilon}(!_{[8]}, [0010])$.

$$\begin{array}{ccc} T(!_{[8]}, [1100]) & & T(!_{[8]}, [0110]) \\ & \searrow \quad \swarrow & \\ & T(!_{[8]}, [0100]) & \end{array} \tag{5}$$

We can proceed similarly for the other images of T. However, we want to be cautious in doing so as every relation of the form (5) will correspond to a test of compatibility between the sequence alignments contained in the images of T. Indeed, trying to link too many images of T together may later lead to a set of empty 'associations'. For instance, linking the set $T(!_{[8]}, [0011])$ and $T(!_{[9]}, [0011])$ through the set $\mathbf{A}E_b^{\varepsilon}(!_{[9]}, [0001])$ (as shown below, in (6)) will label certain alignments of $T(!_{[8]}, [0011])$ as 'inconsistent' because they cannot be related to those of $T(!_{[9]}, [0011])$. Ultimately, this is the type of conclusion that we would like to reach, but, in the present situation, the considered sets of alignments belong to the same pair of individuals and hence should not be able to contradict each other (see Remark 3.26 for further discussion).

$$\begin{array}{ccccccc} T(!_{[8]}, [0011]) & \to & \mathbf{A}E_b^{\varepsilon}(!_{[8]}, [0001]) & \to & \mathbf{A}E_b^{\varepsilon}(!_{[9]}, [0001]) & \leftarrow & T(!_{[9]}, [0011]) \\ \mathtt{ACCGTC}\varepsilon\mathtt{A} & & & & & & \\ \mathtt{ACT}\varepsilon\mathtt{ACTG} & \mapsto & \mathtt{ACT}\varepsilon\mathtt{ACTG} & \mapsto & \mathtt{A}\varepsilon\mathtt{CT}\varepsilon\mathtt{ACTG} & \leftarrow\!\shortmid & ??? \end{array} \tag{6}$$

In other words, for this example, we want to take $T(!_{[n]}, c)$ to be the set $\mathbf{A}E_b^{\varepsilon}(!_{[n]}, c)$ for any integer n and element c in $\Omega^{\times 4}$ satisfying the following relation.

$$(n, c) \in \{(8, [1000]), (8, [0100]), (8, [0010]), (8, [0001]), (7, [0001])\}$$

We then link the images of T at segments whose colors contain exactly two symbols 1 to the images of T at segments whose colors contain a single 1 by forgetting either the top row or the bottom row of each pairwise sequence alignment contained in the domain, in the same fashion as in diagram (5). The obtained maps are obviously indexed by the arrows of B that makes sense with the forgetful operation they define. Doing so defines an inclusion functor $\iota : B \hookrightarrow \mathbf{Seg}(\Omega^{\times 4})$, a functor $T : B \to \mathbf{Set}$ and a natural transformation $T \Rightarrow \mathbf{A}E_b^{\varepsilon} \circ \iota$ that model the table given at the beginning of this example.

3.26. Remark. [Resolving inconsistencies] The present remark discusses the choices made in Example 3.25 for the construction of the functor $T : B \to \mathbf{Set}$. We shall keep the same notations as those introduced therein. First, recall that the category B (the domain of T) was defined so that the images $T(!_{[8]}, [0011])$ and $T(!_{[9]}, [0011])$ could not connect through another image of T.

$$\begin{array}{ccc} T(!_{[8]}, [0011]) & & T(!_{[9]}, [0011]) \\ & \searrow \quad \swarrow & \\ & \mathbf{A}E_b^{\varepsilon}(!_{[9]}, [0001]) & \end{array} \tag{7}$$

The reason for this was that not every element in $T(!_{[8]}, [0011])$ could find a corresponding element in $T(!_{[9]}, [0011])$ through the set $\mathbf{A}E_b^{\varepsilon}(!_{[9]}, [0001])$ so that diagram (7) would eventually lead to label those elements as inconsistent. In our case, this type of scenario should be avoided because we are mainly interested in unravelling inconsistencies between different tables produced by the dynamic programming algorithm of section 2.1 (while diagram (7) compares alignments coming from the same table). However, it could certainly be interesting to be able to know whether the sequence alignments of `Craig` and `Doug` of length 8 are consistent with those of length 9. A way to do this without creating a conflict with our main goal would be to add a new color to Ω, say by taking the pre-ordered set $\Omega = \{0 \leq 1 \leq 2\}$ and keeping $b = (1, 1, 1, 1)$. Then, we could use this new color to study the compatibility of the alignments coming from the same table. For instance, we could give T an image at the segment $(!_{[8]}, [0022])$ that would contain the same alignments as those contained in $T(!_{[9]}, [0011])$, but the resulting image $T(!_{[8]}, [0022])$ would be linked to the image $T(!_{[8]}, [0011])$ through the image

$$T(!_{[9]}, [0001]) := \mathbf{A}E_b^{\varepsilon}(!_{[9]}, [0001])$$

as shown in the diagram given below (the left-hand side arrow makes sense with the functor structure because the inequality $[0001] \preceq [0022]$ holds in $\Omega^{\times 4}$).

$$\begin{array}{ccc} T(!_{[8]}, [0022]) & & T(!_{[9]}, [0011]) \\ & \searrow \quad \swarrow & \\ & T(!_{[9]}, [0001]) & \end{array}$$

Meanwhile, the image $T(!_{[8]}, [0011])$ would be reserved to studying the compatibility with the other pairs of individuals and would be isolated from $T(!_{[9]}, [0011])$ because the category $\mathbf{Seg}(\Omega)$ does not allow morphisms of the type $(!_{[8]}, [0011]) \to (!_{[8]}, [0022])$ to exist.

3.27. From right Kan extensions to multiple sequence alignments. In this section, we show that the right Kan extension (see Definition 3.32) of a sequence alignment functor (Definition 3.23) contains what one would like to understand as the outputs of the table gluing algorithm described at the end of section 2.1.

We start the section with an example showing how sequence alignment functors can be used, along with limits, to reason about the relatedness of a group of individuals. In particular, we illustrate it in the context of our main example (section 2.1).

3.28. Example. [Reasoning with sequence alignment functors] The present example continues the discussion of Example 3.25 and shows how the sequence alignment functor (ι, T, σ) defined therein can be used to reason about our main example given in section 2.1. More specifically, we will show how the sequence alignment functor (ι, T, σ) can be used to deduce phylogenetic relationships between the individuals of section 2.1 by looking at whether certain arrows induced by the structure of (ι, T, σ) are proper surjections, bijections and simply functions.

We start this example by looking at a surjection. First, an easy calculation shows that the pullback of diagram (5) can be mapped surjectively onto the image $T(!_{[8]}, [1010])$ by forgetting the DNA sequences associated with Bob.

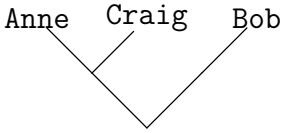
$$T(!_{[8]}, [1100]) \times_{x_b} T(!_{[8]}, [0110]) \to T(!_{[8]}, [1010]) \tag{8}$$

More specifically, the associated surjection, of the form shown in (8), maps the pairs of pairwise sequence alignments shown in (9) (the domain of (8)) to the pairwise sequence alignments shown in (10) (the codomain of (8)) by forgetting the bottom and top sequences of the first and second components of the pairs contained in the domain.

$$\left\{ \begin{pmatrix} \texttt{ACCGACTG} \\ \texttt{A}\varepsilon\texttt{CATCTG} \\ \texttt{A}\varepsilon\texttt{CATCTG} \\ \texttt{ACCGTC}\varepsilon\texttt{A} \end{pmatrix} \begin{pmatrix} \texttt{ACCGACTG} \\ \texttt{ACA}\varepsilon\texttt{TCTG} \\ \texttt{ACA}\varepsilon\texttt{TCTG} \\ \texttt{ACCGTC}\varepsilon\texttt{A} \end{pmatrix} \begin{pmatrix} \texttt{ACCGACTG} \\ \texttt{A}\varepsilon\texttt{CATCTG} \\ \texttt{A}\varepsilon\texttt{CATCTG} \\ \texttt{ACCGTCA}\varepsilon \end{pmatrix} \begin{pmatrix} \texttt{ACCGACTG} \\ \texttt{ACA}\varepsilon\texttt{TCTG} \\ \texttt{ACA}\varepsilon\texttt{TCTG} \\ \texttt{ACCGTCA}\varepsilon \end{pmatrix} \right\} \tag{9}$$

$$\left\{ \begin{pmatrix} \texttt{ACCGACTG} \\ \texttt{ACCGTC}\varepsilon\texttt{A} \end{pmatrix} \begin{pmatrix} \texttt{ACCGACTG} \\ \texttt{ACCGTCA}\varepsilon \end{pmatrix} \right\} \tag{10}$$

As seen in (9), the elements of the domain can be interpreted as multiple sequence alignments of Anne, Bob and Craig, where the sequence of Bob is repeated twice. The fact that these multiple sequence alignments can be sent to the pairwise sequences of Anne and Craig tells us that Bob does not inform us of new ways of relating Anne and Craig together. Furthermore, the fact that function (9) is also a surjection tells us that the sequence of Bob is also unable to provide any potential correction to the sequence alignments of Anne and Craig (by showing us some sequence alignment of Anne and Caig that would not be supported by the sequence alignments of (9)). In the present case, it is as if Bob is unnecessary for understanding possible nuances in the evolution of Anne and Craig. A very probable reason could be that Anne and Craig are much closer to each other genetically than they are to Bob – this would be represented by the following evolutionary tree.



From the point of view of the problem exposed in section 2.1, surjection (8) tells us that gluing the comparison table of Anne and Bob with the comparison table of Bob and Craig

along the edge of Bob does not contradict the table of Anne and Craig. In other words, there is no obstruction for passing from the two dimensional comparisons of Anne with Bob and Bob with Craig to the three dimensional table comparing Anne, Bob and Craig together.

Let us now look at an example for which the function is not surjective. First, recall that the sequence alignment (ι, T, σ) was defined such that cospan (11) exists (see Example 3.25).

$$\begin{array}{ccc} T(!_{[8]}, [1010]) & & T(!_{[8]}, [1001]) \\ & \searrow \quad \swarrow & \\ & T(!_{[8]}, [1000]) & \end{array} \tag{11}$$

An easy calculation then shows that the pullback of cospan (11) contains multiple sequence alignments of Anne, Craig and Doug that can be mapped to the sequence alignments of Craig and Doug contained in $T(!_{[8]}, [0011])$ by forgetting the DNA sequences associated with Anne.

$$T(!_{[8]}, [1010]) \times_{\mathtt{a}} T(!_{[8]}, [1001]) \to T(!_{[8]}, [0011]) \tag{12}$$

However, comparing the cardinalities of the domain and codomain of the resulting function, shown in (12), informs us that this mapping cannot be surjective. This suggests that, while Anne does not give new ways of relating Craig and Doug together, the relatedness of Craig and Doug may still be nuanced by the consideration of Anne. In other words, the evolution of Craig and Doug cannot completely be explained without the sequence of Anne. Such a relationship could be represented by one of the following evolutionary trees, in which the removal of Anne may prevent us from understanding how different Craig and Doug are.

From the point of view of the problem exposed in section 2.1, function (12) tells us that gluing the comparison table of Anne and Craig with the comparison table of Anne and Doug along the edge of Anne does not confirm all the pairwise sequence alignments computed for Craig and Doug. This suggests that the relatedness of Craig and Doug may be too old to be completely described by the table computed for Craig and Doug and would be better analyzed through the gluing of other tables.

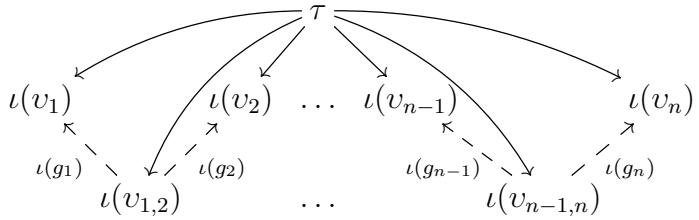
Our goal is now to formalize the discussion of Example 3.28 through the notion of right Kan extension (given in Definition 3.32). This will allow us to motivate the introduction of chromologies in Example 3.38. We start by defining the category on which the right Kan extension is *computed*.

3.29. Definition. [Extending category] *Let $(\Omega, \preceq)$ be a pre-ordered set. For every object τ in $\mathbf{Seg}(\Omega)$ and functor $\iota : B \to \mathbf{Seg}(\Omega)$, we will denote by $(\tau \downarrow \iota)$ the category whose objects are pairs (υ, f) where υ is an object in B and f is a morphism $\tau \to \iota(\upsilon)$ in $\mathbf{Seg}(\Omega)$*

and whose arrows $(\upsilon, f) \to (\upsilon', f')$ *are given by morphisms* $g : \upsilon \to \upsilon'$ *in* B *that make the following square commute in* $\mathbf{Seg}(\Omega)$.

$$\begin{array}{ccc} \tau & = & \tau \\ {\scriptstyle f}\downarrow & & \downarrow{\scriptstyle f'} \\ \iota(\upsilon) & \xrightarrow[\iota(g)]{} & \iota(\upsilon') \end{array}$$

3.30. Remark. [Extending category as a cone] Let $(\Omega, \preceq)$ be a pre-ordered set. For every object τ in $\mathbf{Seg}(\Omega)$ and functor $\iota : B \to \mathbf{Seg}(\Omega)$, the category $(\tau \downarrow \iota)$ defined in Definition 3.29 can be pictured as a cone of the form $\Delta_{(\tau\downarrow\iota)}(\tau) \Rightarrow \iota_\tau \circ \iota$ (section 2.33). The arrows of the transformation associated with the cone are given by the objects of the category $(\tau \downarrow \iota)$ (see non-dashed arrows shown below) while its diagram is formed by the arrows of $(\tau \downarrow \iota)$ (see dashed arrows shown below).



3.31. Convention. [Extending diagram] *Let* $(\Omega, \preceq)$ *be a pre-ordered set. For every object* τ *in* $\mathbf{Seg}(\Omega)$ *and functor* $\iota : B \to \mathbf{Seg}(\Omega)$*, we will denote by* ι_τ *the obvious functor* $(\tau \downarrow \iota) \to B$ *that maps an object* (υ, f) *in* $(\tau \downarrow \iota)$ *to the object* υ *in* B *and maps an arrow* $g : (\upsilon, f) \to (\upsilon', f')$ *in* $(\tau \downarrow \iota)$ *to the arrow* $g : \upsilon \to \upsilon'$ *in* B.

The following definition introduces the concept of right Kan extension by using its well-known expression in terms of limits (see [12, Chap. X, Th. 1]).

3.32. Definition. [Right Kan extensions] *Let* $(\Omega, \preceq)$ *be a pre-ordered set, and* $\iota : B \to \mathbf{Seg}(\Omega)$ *and* $T : B \to \mathbf{Set}$ *be two functors. We define the* right Kan extension of T along ι *as the canonical functor* $\mathsf{Ran}_\iota T : \mathbf{Seg}(\Omega) \to \mathbf{Set}$ *defined by the following limit construction at every object* τ *in* $\mathbf{Seg}(\Omega)$.

$$\mathsf{Ran}_\iota T(\tau) = \mathsf{lim}_{(\tau\downarrow\iota)} T \circ \iota_\tau \tag{13}$$

The images of this functor on the arrows of $\mathbf{Seg}(\Omega)$ *will be described, later, in Remark 3.35.*

3.33. Example. [Right Kan extensions] Let (ι, T, σ) denote the sequence alignment functor over $\mathbf{A}E_b^\varepsilon$ defined Example 3.28. The goal of the present example is to show what the images of the right Kan extension of $T : B \to \mathbf{Set}$ along the inclusion $\iota : B \to \mathbf{Seg}(\Omega^{\times 4})$ look like. We will compute two images, namely $\mathsf{Ran}_\iota T(!_{[8]}, [1100])$ and $\mathsf{Ran}_\iota T(!_{[8]}, [1110])$.

Let us start with the image of the segment $(!_{[8]}, [1100])$. According to Definition 3.32, we need to compute the set of objects and the set of arrows defining the category $((!_{[8]}, [1100]) \downarrow$

ι). We follow Definition 3.29, for which we use the category B described in Example 3.25, and deduce that the objects of $((!_{[8]}, [1100]) \downarrow \iota)$ are of the following form.

$$\begin{array}{ll} (!_{[8]}, [1100]) \to (!_{[8]}, [0100]) & (!_{[8]}, [1100]) \to (!_{[8]}, [1000]) \\ (!_{[8]}, [1100]) \to (!_{[8]}, [1100]) & (!_{[8]}, [1100]) \to (!_{[9]}, [1100]) \end{array} \tag{14}$$

Then, Definition 3.29 tells us that the arrows of the category $((!_{[8]}, [1100]) \downarrow \iota)$ are given by the arrows of B that can form commutative triangles with the arrows of (14).

Let us describe the arrows of $((!_{[8]}, [1100]) \downarrow \iota)$ more explicitly. First, by definition of B, any object of $((!_{[8]}, [1100]) \downarrow \iota)$ of the type shown in the bottom-right corner of (14) is isolated from any object of a different type in (14). Since there are exactly 9 representatives of the type

$$(!_{[8]}, [1100]) \to (!_{[9]}, [1100])$$

in $\mathbf{Seg}(\Omega^{\times 4} \,|\, 9)$, the category $((!_{[8]}, [1100]) \downarrow \iota)$ has exactly 9 isolated objects of this type. However, because the image of the segment $(!_{[9]}, [1100])$ via T is terminal (see Example 3.25), these isolated objects will not matter in the computation of the limit $\mathsf{Ran}_\iota T(!_{[8]}, [1100])$ (see formula (13)). What matters is the diagram encoded by the other objects, which we now describe. First, because $\mathbf{Seg}(\Omega^{\times 4} \,|\, 8)$ is a pre-order category (Proposition 2.31), the arrows of $((!_{[8]}, [1100]) \downarrow \iota)$ that relate the remaining objects are unique. Also, observe that the object $(!_{[8]}, [1100])$ can be related to the objects $(!_{[8]}, [0100])$ and $(!_{[8]}, [1000])$ through a cospan in B. Using formula (13), we deduce that isomorphism (15) holds.

$$\mathsf{Ran}_\iota T(!_{[8]}, [1100]) \cong T(!_{[8]}, [1100]) \tag{15}$$

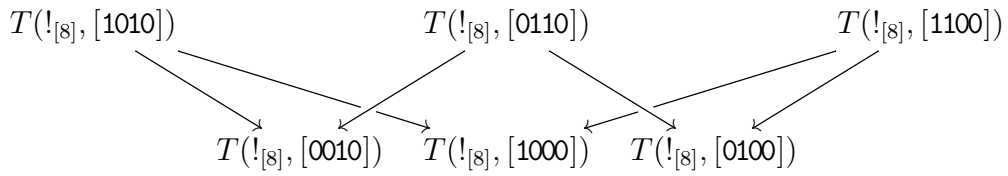
In other words, the image of the right Kan extension $\mathsf{Ran}_\iota T$ at the segment $(!_{[8]}, [1100])$ contains all the pairwise sequence alignments of `Anne` and `Bob` coming from the dynamic programming algorithm.

Let us now compute the image of the segment $(!_{[8]}, [1110])$. To do so, we need to describe the set of objects and the set of arrows of the category $((!_{[8]}, [1110]) \downarrow \iota)$. In the present case, the set of objects is made of arrows of the following type.

$$(!_{[8]}, [1110]) \to (!_{[8]}, [1000]) \qquad (!_{[8]}, [1110]) \to (!_{[8]}, [0100]) \qquad (!_{[8]}, [1110]) \to (!_{[8]}, [0010])$$

$$\begin{array}{ll} (!_{[8]}, [1110]) \to (!_{[8]}, [1010]) & (!_{[8]}, [1110]) \to (!_{[8]}, [0110]) \\ (!_{[8]}, [1110]) \to (!_{[8]}, [1100]) & (!_{[8]}, [1110]) \to (!_{[9]}, [1100]) \end{array}$$

After eliminating the objects whose images via T are terminal and only considering the morphisms that relate the remaining objects in $((!_{[8]}, [1110]) \downarrow \iota)$, we can deduce from formula (13) that the image of $\mathsf{Ran}_\iota T$ at the segment $(!_{[8]}, [1110])$ is isomorphic to the limit of the following diagram.



(16)

As shown in Example 3.28, the elements of $\mathsf{Ran}_\iota T(!_{[8]}, [1110])$ can be seen as triples of pairwise sequence alignments contained in $T(!_{[8]}, [1010])$, $T(!_{[8]}, [0110])$ and $T(!_{[8]}, [1100])$ that encode multiple sequence alignments between the sequences of `Anne`, `Bob` and `Craig`.

3.34. Remark. [Right Kan extension and gluing of tables] It should be clear that the images of the right Kan extension computed in Example 3.33 capture the higher dimensional gluings mentioned in section 2.1. The idea is that one formally specifies how the pairwise comparison tables are glued together via the category defined in Definition 3.29 and one uses the limit formula given in Definition 3.32 to collect the compatible sequence alignments for these gluings. As a result, the right Kan extension can be seen as a model for a multiple sequence alignment algorithm. From now on, our goal will be to describe how this model can be used to reason about the mechanisms linking the individuals considered in the sequence alignment functor.

3.35. Remark. [Functoriality] Let $(\Omega, \preceq)$ be a pre-ordered set, and $\iota : B \to \mathbf{Seg}(\Omega)$ and $T : B \to \mathbf{Set}$ be two functors. Let us explain why the map $\tau \mapsto \mathsf{Ran}_\iota T(\tau)$ constructed in Definition 3.32 induces a functor from $\mathbf{Seg}(\Omega)$ to $\mathbf{Set}$. First, notice that the functor $\iota_\tau : (\tau \downarrow \iota) \to B$ defined in Convention 3.31 is natural in τ on the opposite category $\mathbf{Seg}(\Omega)^{\mathrm{op}}$, which means that every morphism $h : \tau \to \tau'$ in $\mathbf{Seg}(\Omega)$ induces a functor

$$h^* : (\tau' \downarrow \iota) \to (\tau \downarrow \iota)$$

for which the identity $\iota_{\tau'} = \iota_\tau \circ h^*$ holds. This last equation means that the functor h^* sends an object (υ, f) in $(\tau' \downarrow \iota)$ to the object $(\upsilon, f \circ h)$ in $(\tau \downarrow \iota)$. The image of the morphism $h : \tau \to \tau'$ via $\mathsf{Ran}_\iota T$ is then the canonical morphism induced by pre-composing the diagram of the limit shown below, on the left, with h^*.

$$\mathsf{lim}_{(\tau\downarrow\iota)} T \circ \iota_\tau \xrightarrow{\mathsf{Ran}_\iota T(h)} \mathsf{lim}_{(\tau\downarrow\iota)} T \circ \iota_\tau \circ h^*$$

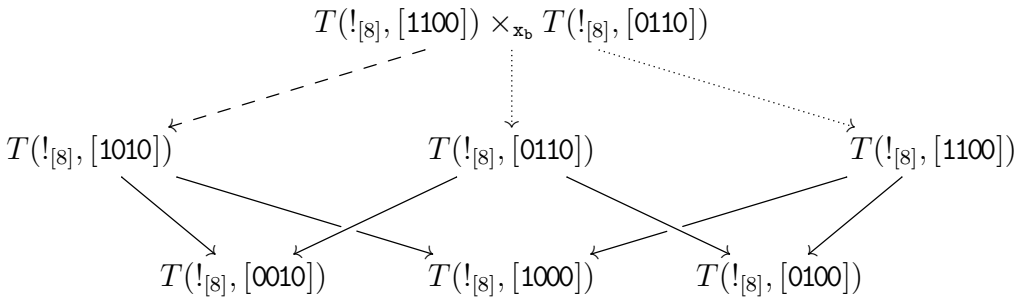
The functor structure of the mapping $\tau \mapsto \mathsf{Ran}_\iota T(\tau)$ then follows from the universality of limits.

3.36. Example. [Reasoning with right Kan extensions] The present example is a continuation of the discussion of Example 3.28 from the point of view of Definition 3.32 and Example 3.33. We shall keep the same notations as those used in Example 3.28 and let (ι, T, σ) denote the sequence alignment functor over $\mathbf{A}E_b^\varepsilon$ used thereof. Our goal is to reformulate the statement given in Example 3.28 regarding the surjection

$$T(!_{[8]}, [1100]) \times_{\mathtt{x_b}} T(!_{[8]}, [0110]) \to T(!_{[8]}, [1010])$$

in terms of the right Kan extension of T along ι. First, it follows from the description given therein that this function, represented by the dashed arrow below, induces a cone

over diagram (16) – the cone is shown below, using the dashed and dotted arrows.



The universal property of limits then gives us a factorization of the dashed surjection through one of the canonical projections associated with the limit of diagram (16). More specifically, this projection is of the form shown in (17).

$$\mathsf{Ran}_\iota T(!_{[8]}, [1110]) \to T(!_{[8]}, [1010]) \tag{17}$$

The usual properties of surjections (or, in fact, those of epimorphisms) imply that projection (17) is also a surjection. Using a similar reasoning to the one used to deduce isomorphism (15), we can show that the codomain of projection (17) is isomorphic to $\mathsf{Ran}_\iota T(!_{[8]}, [1010])$. This means that (17) is of the following form.

$$\mathsf{Ran}_\iota T(!_{[8]}, [1110]) \longrightarrow \mathsf{Ran}_\iota T(!_{[8]}, [1010]) \tag{18}$$

In fact, we can even show that the previous surjection is the image of the obvious morphism of segments

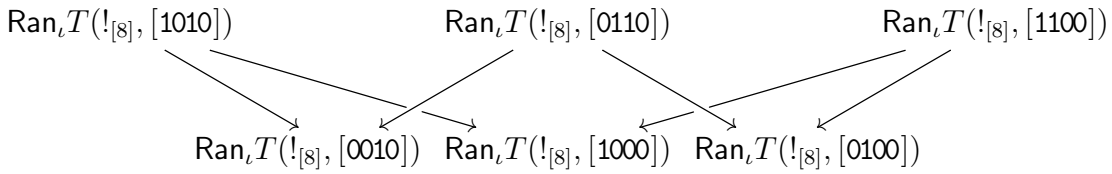
$$(!_{[8]}, [1110]) \to (!_{[8]}, [1010])$$

via the functor described in Remark 3.35. In the spirit of Remark 3.34, in which the images of the right Kan extension were interpreted as models for higher dimensional gluings of comparison tables between DNA sequences, surjection (18) tells us that the higher dimensional gluing of the comparison tables of `Anne`, `Bob` and `Craig` is completely captured by the comparison table of `Anne` and `Craig`, with some uncertainty as to what exactly links `Anne` and `Craig` through `Bob`.

3.37. Example. [Reasoning about data consistency] As already suggested through the discussions of Example 3.25, Remark 3.26 and Example 3.28, limits can give ways to assess the *consistency* of the data. For instance, the right Kan extension of a sequence alignment functor is given by a limit (Definition 3.32) that computes *compatible* sets of sequence alignments (these compatible sets of alignments were interpreted as multiple sequence alignments in Remark 3.34). In this example, we compute limits of images of the right Kan extension to study how *consistent* these compatible sets of sequence alignments are. As before, we assess this consistency by studying the properties of certain canonical arrows induced by these limits.

We shall continue the discussion started in Example 3.36 and look at the limit of diagram (16). First, observe that we can copy the reasoning that was used in Example

3.33 to deduce isomorphism (15) to show that diagram (16) is in fact a diagram of images of $\mathsf{Ran}_\iota T$ as follows.



Remark 3.35 even tells us that the previous diagram is the image of the underlying diagram of segments via the right Kan extension $\mathsf{Ran}_\iota T$. More specifically, if we let $F : A \to \mathbf{Seg}(\Omega)$ denote the diagram of segments indexing the previous diagram, then the computation of the image $\mathsf{Ran}_\iota T(!_{[8]}, [1110])$ described in Example 3.33 shows that the following canonical arrow is an isomorphism in **Set**.

$$\mathsf{Ran}_\iota T(!_{[8]}, [1110]) \to \mathsf{lim}_A \mathsf{Ran}_\iota T \circ F$$

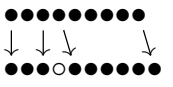
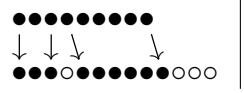
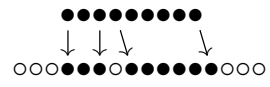
From the point of view of Remark 3.34, this means that there is no uncertainty as to how `Anne`, `Bob` and `Craig` relate to each other from the point of view of their pairwise sequence alignments – the previous isomorphism thus informs us that the table gluing procedure is perfectly consistent. In Example 3.38, we will look at an instance of a canonical arrow that fails to be an isomorphism and is only a surjection. We will see that this type of arrow informs us that the integrated data is associated with some uncertainties.

The idea behind the right Kan extension of a sequence alignment functor is to collect *all* the local and global information that is accessible from the point of view of a particular segment (for more intuition, see Remark 3.30). Since the domain category of a sequence alignment functor can be designed to control the integration of this information (see Remark 3.26), the right Kan extension of a sequence alignment functor gives us a controlled procedure to construct multiple sequence alignments (Remark 3.34) through the parsing of local and global pieces of information within the data set.

In bioinformatics, similar heuristics have been developed for the construction of sequence alignments, one of the most popular being the algorithm BLAST [1]. This algorithm constructs a sequence alignment by looking at the local patches of a set of DNA strands and aligning them according to a given scoring system. While the right Kan extension of a sequence alignment functor proceeds in a similar fashion, its scoring system is more categorical than numerical – this is discussed in Example 3.38.

3.38. Example. [Global alignments *versus* local alignments] In bioinformatics, the dynamic programming algorithm presented in section 2.1 is usually used with two main classes of scoring systems. The first class, known as the *Needleman-Wunsch algorithm* [16], aims to find global sequence alignments by initializing the comparison table with incremented gap penalties. This was the type of scoring system that we used in section 2.1, in which gap penalties were incremented by 1. The second class is known as the *Smith-Waterman algorithm* [26] and is used to find local sequence alignments by initializing the comparison table with null gap penalties. Hybrid scoring systems that only set

gap penalties to 0 for either the rows or the columns can be used to detect semi-global sequence alignments.

Global	Semi-global	Local
		

In this example, our goal is to show that the right Kan extension $\mathsf{Ran}_\iota T$ associated with the sequence alignment functor constructed in Example 3.25 captures both local and global aspects of the sequence alignment algorithms mentioned earlier. We will see that the local information is detected by the type of morphism presented in Example 2.23. The subsequent discussion will show that local pieces of information often come with more uncertainty than global ones. The first part of our discussion will consist in computing the images of the four homologous segments $(!_{[8]}, [1000])$, $(!_{[8]}, [1010])$, $(!_{[8]}, [1001])$ and $(!_{[8]}, [1011])$ via $\mathsf{Ran}_\iota T$. Because the computation of these images are all very similar and all follow formula (13), we will only detail the calculation of the image $\mathsf{Ran}_\iota T(!_{[8]}, [1011])$ and directly give the images of the other segments.

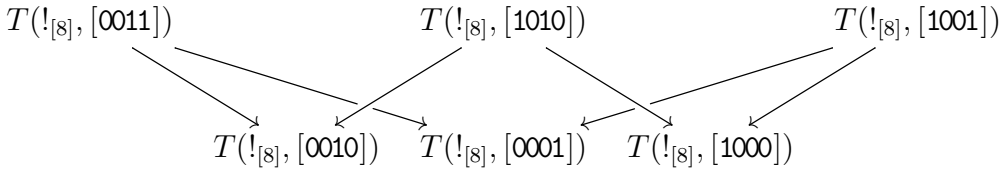
To compute the image of the segment $(!_{[8]}, [1011])$ via $\mathsf{Ran}_\iota T$, we need to look at the collection of objects of the category $((!_{[7]}, [1011]) \downarrow \iota)$ which consists of all the arrows of the following type in $\mathbf{Seg}(\Omega^{\times 4})$.

$$(!_{[8]}, [1011]) \to (!_{[8]}, [1001]) \qquad (!_{[8]}, [1011]) \to (!_{[8]}, [1010]) \qquad (!_{[8]}, [1011]) \to (!_{[8]}, [0011])$$
$$(!_{[8]}, [1011]) \to (!_{[8]}, [1000]) \qquad (!_{[8]}, [1011]) \to (!_{[8]}, [0010]) \qquad (!_{[8]}, [1011]) \to (!_{[8]}, [0001])$$
$$(!_{[8]}, [1011]) \to (!_{[9]}, [0011])$$

While the objects encoded by arrows in $\mathbf{Seg}(\Omega^{\times 4} \,|\, 8)$ are unique (see Proposition 2.31), the objects of the type $(!_{[8]}, [1011]) \to (!_{[9]}, [0011])$ possess exactly 9 representatives. After examining the relations existing between these objects in $((!_{[8]}, [1011]) \downarrow \iota)$, formula (13) implies that the image of $\mathsf{Ran}_\iota T$ at the segment $(!_{[8]}, [1011])$ is isomorphic to the set

$$L_8([1011]) \times T(!_{[9]}, [0011])^{\times 9}$$

where we denote by $L_8([0111])$ the limit of the diagram induced by the arrows of the category $((!_{[7]}, [1011]) \downarrow \iota)$ between the segments of the domain $[8]$ (see the diagram below).

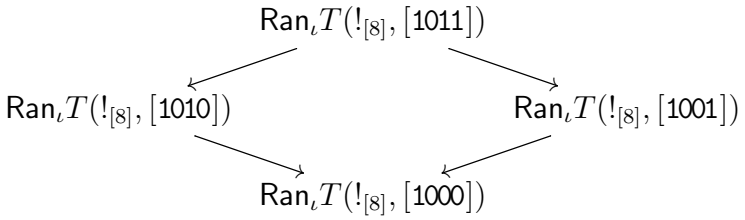


A similar analysis for the images of $\mathsf{Ran}_\iota T$ at the segments $(!_{[8]}, [1000])$, $(!_{[8]}, [1010])$, and $(!_{[8]}, [1001])$ gives the following collection of isomorphisms.

$$\mathsf{Ran}_\iota T(!_{[8]}, [1000]) \cong T(!_{[8]}, [1000])$$

$$\mathsf{Ran}_\iota T(!_{[8]}, [1010]) \cong T(!_{[8]}, [1010])$$
$$\mathsf{Ran}_\iota T(!_{[8]}, [1001]) \cong T(!_{[8]}, [1001])$$

Let us now use the images of the four segments $(!_{[8]}, [1000])$, $(!_{[8]}, [1010])$, $(!_{[8]}, [1001])$ and $(!_{[8]}, [1011])$ to see how the images of the right Kan extension integrate the local pieces of information available from the sequence alignment functor (ι, T, σ). First, the functoriality of $\mathsf{Ran}_\iota T$ gives us a commutative diagram as follows for the obvious choices of morphisms in $\mathbf{Seg}(\Omega \mid 8)$.



Even though the conclusion of Example 3.37 could suggest that this diagram is a pullback, our computation shows that the image $\mathsf{Ran}_\iota T(!_{[8]}, [1011])$ is not isomorphic to the pullback of the lower part of the diagram and is only related to it via a projection of the following form.

$$L_8([1011]) \times T(!_{[9]}, [0011])^{\times 9} \longrightarrow T(!_{[8]}, [1010]) \times_{\mathtt{x_a}} T(!_{[8]}, [1001]) \tag{19}$$

Here, we can view the set $T(!_{[9]}, [0011])^{\times 9}$ as (formally) containing the local sections of length 8 taken from the sequence alignments of length 9 associated with `Doug` and `Craig`. Arrow (19) then tries to relate these local sections to the local sections of length 8 taken from the sequence alignment of `Anne` and `Craig` and that of `Anne` and `Doug`. However, arrow (19) fails to map the elements of the product

$$T(!_{[9]}, [0011])^{\times 9}$$

to elements in its codomain and is forced to forget these elements in the same way as a proper Cartesian projection map would do. This failure is not surprising since we designed the domain of T so that the alignments of $T(!_{[9]}, [0011])$ can never be connected to those of $T(!_{[8]}, [0011])$ through $T(!_{[9]}, [0001])$ and $T(!_{[9]}, [0010])$. The reason for this was to prevent the limit construction of the right Kan extension from forgetting the alignments of $T(!_{[8]}, [0011])$ that were inconsistent with the alignments of $T(!_{[9]}, [0011])$ (see Example 3.25). In fact, the reader can check that if we had done so, then we would also have prevented the resulting version of (19) from being a bijection. This suggests that whatever method we try to use, the data contained in T tends to prevent arrow (19) from being a bijection.

▷ *Interpretation:* This last fact actually hides important information about the relatedness of our four individuals. Indeed, the difference between the conclusion of the present example and that of Example 3.37, in which we were able to show that a certain canonical arrow was a bijection, informs us that the genetic data of `Anne`, `Bob` and `Craig`

are overall rather similar while the genetic data of `Anne`, `Craig` and `Doug` are much more different. This can already be seen in the sizes of the images of the functor T at the segments $(!_{[8]}, [\texttt{0011}])$ and $(!_{[9]}, [\texttt{0011}])$, which are much larger than the images of T at the other segments (see the table of Example 3.25). In fact, the sizes of these images are related to the uncertainty of finding the right alignment for the sequences of `Craig` and `Doug`, and the simple fact that T even has an image at the segment $(!_{[9]}, [\texttt{0011}])$, while the genetic data of `Craig` and `Doug` is only of length 7, tells us that the dynamic programming algorithm is struggling to find an obvious match between the sequences of `Craig` and `Doug`.

▷ *Conclusion:* We see that the obstruction – or rather the uncertainty – resulting from aligning a set of distant DNA sequences is detected by the ability of certain canonical arrows to be isomorphisms or epimorphisms. It is precisely for these reasons that the concepts of chromology and pedigrad become relevant to the study of our main example.

3.39. Exactly distributive and injective chromologies. The goal of the present section is to define two canonical classes of chromologies. As usual, we let $(\Omega, \preceq)$ be a pre-ordered set, b be an element in Ω, A be a small category, τ be an object in $\mathbf{Seg}(\Omega \,|\, n)$ and $\rho : \Delta_A(\tau) \Rightarrow \theta$ be a cone in $\mathbf{Seg}(\Omega \,|\, n)$ for some non-negative integer n. First, note that the application of the truncation functor $\mathsf{Tr}_b : \mathbf{Seg}(\Omega \,|\, n) \to \mathbf{Set}^{\mathrm{op}}$ on the cone ρ gives rise to a cocone in $\mathbf{Set}$ as follows.

$$\mathsf{Tr}_b(\rho) : \mathsf{Tr}_b\theta \Rightarrow \Delta_A \circ \mathsf{Tr}_b(\tau)$$

The colimit adjoint of this natural transformation in $\mathbf{Set}$ gives us a function as follows.

$$\mathsf{colim}_A\mathsf{Tr}_b(\rho) : \mathsf{colim}_A\mathsf{Tr}_b\theta \longrightarrow \mathsf{Tr}_b(\tau) \tag{20}$$

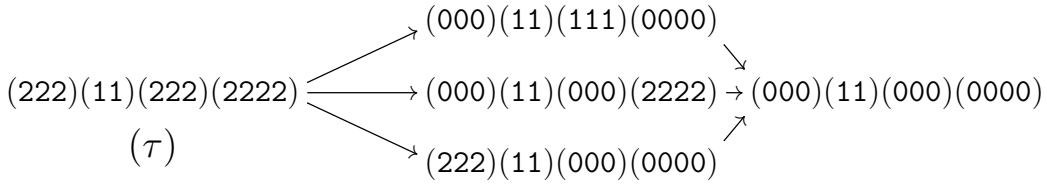
3.40. Definition. [Exactly distributive cones] *A cone of the form $\rho : \Delta_A(\tau) \Rightarrow \theta$ in $\mathbf{Seg}(\Omega \,|\, n)$ will be said to be* exactly b-distributive *if the arrow of (20) is an isomorphism in* $\mathbf{Set}$.

3.41. Definition. [Injective cones] *A cone of the form $\rho : \Delta_A(\tau) \Rightarrow \theta$ in $\mathbf{Seg}(\Omega \,|\, n)$ will be said to be b-*injective *if the arrow of (20) is a monomorphism in* $\mathbf{Set}$.

3.42. Example. [Exactly distributive cones] Let Ω denote the pre-ordered set $\{0 \leq 1 \leq 2\}$. In this example, we give various instance of exactly distributive cones and injective cones in $\mathbf{Seg}(\Omega)$. Before showing these instances, let us mention that a cone $\Delta(\tau) \Rightarrow \theta$ should be seen as a structure specifying an integration operation from the diagram θ to the object τ – this may be useful to understand what these cones are meant to specify.

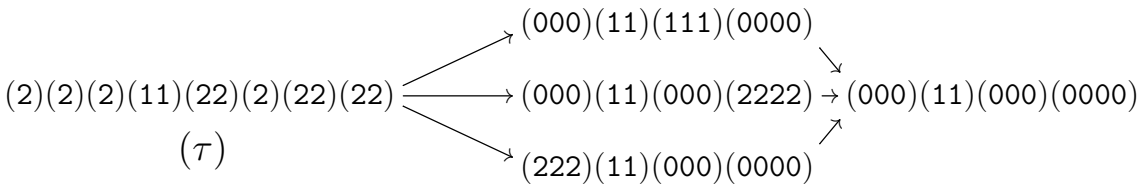
First, we can give the following diagram, living in one of the pre-order categories $\mathbf{Seg}(\Omega : t)$ for the obvious topology t of domain [12], as an example of an exactly 1-

distributive cone, but also as an example of a 2-injective cone.



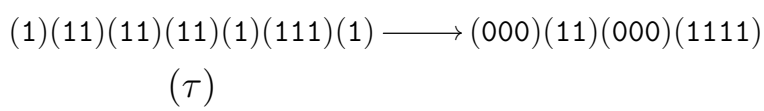
Here, the idea is that an exactly 1-distributive cone cannot have nodes of color above 1 at a given position in the segment if these nodes are not both mapped to a common node of color above 1. Intuitively, one could imagine to use this type of cone to integrate different alignment methods on the objects of the diagram θ into a unique one on the object τ.

We now give the following diagram as an example of a exactly 1-distributive cone and a 2-injective cone in the category of quasi-homologous segments $\mathbf{Seg}(\Omega \,|\, 12)$.



The difference between the very first cone and the one given above is that the latter specifies an integration operation whose action also applies to a more refined topology (on τ). For instance, aligning a set of DNA strands with respect to the codon topology will necessarily align the DNA strands with respect to the nucleotide topology.

Finally, the following arrow in $\mathbf{Seg}(\Omega \,|\, 12)$ is an example of an exactly 0-distributive cone as well as an example of a 1-injective cone.



3.43. Definition. [Exactly distributive chromologies] *Let b be an element in Ω. A chromology (Ω, D) will be said to be* exactly b-distributive *if all the cones in D are exactly b-distributive.*

3.44. Definition. [Injective chromologies] *Let b be an element in Ω. A chromology (Ω, D) will be said to be* b-injective *if all the cones in D are b-injective.*

3.45. Logical systems for pedigrads in sets. In this section, we show that the functors defined in Definition 3.13 are pedigrads in two different logical systems of **Set** for two different types of chromologies.

3.46. Definition. [Logical systems of bijections] *We will denote by $\mathcal{W}^{bij}$ the class of cones $\Delta_A(X) \Rightarrow F$ in* **Set** *whose limit adjoints $X \to \mathsf{lim}_A F$ are bijections.*

In section 4, we will show that one can use the following theorem to study the information contained in a sequence alignment functor (Definition 3.23).

3.47. Theorem. *For every element b in Ω and exactly b-distributive chromology (Ω, D), the functor $E^{\varepsilon}_{b} : \mathbf{Seg}(\Omega) \to \mathbf{Set}$ is a $\mathcal{W}^{bij}$-pedigrad for (Ω, D).*

Proof. Let $\rho : \Delta_A(\tau) \Rightarrow \theta$ be a cone in $D[n_1]$ for some given non-negative integer n_1. Because (Ω, D) is an exactly b-distributive chromology, it follows from Definition 3.40 and Definition 3.43 that the canonical arrow

$$\mathsf{colim}_A \mathsf{Tr}_b\theta \to \mathsf{Tr}_b(\tau)$$

is an isomorphism in $\mathbf{Set}$. As a result, the image of this arrow via the functor $\mathbf{Set}(\_, E) : \mathbf{Set}^{\mathrm{op}} \to \mathbf{Set}$ is a bijection. By Proposition 3.17 and the usual definition of colimits in $\mathbf{Set}$, the resulting bijection is (naturally) isomorphic to the following canonical arrow.

$$E^{\varepsilon}_{b}(\tau) \to \mathsf{lim}_A E^{\varepsilon}_{b} \circ \theta$$

This precisely shows that $E^{\varepsilon}_{b} : \mathbf{Seg}(\Omega) \to \mathbf{Set}$ is a $\mathcal{W}^{\mathrm{bij}}$-pedigrad for (Ω, D). ∎

3.48. Definition. [Logical systems of surjections] *We will denote by $\mathcal{W}^{surj}$ the class of cones $\Delta_A(X) \Rightarrow F$ in $\mathbf{Set}$ whose limit adjoints $X \to \mathsf{lim}_A F$ are surjections.*

3.49. Theorem. *For every element b in Ω and b-injective chromology (Ω, D), the functor $E^{\varepsilon}_{b} : \mathbf{Seg}(\Omega) \to \mathbf{Set}$ is a $\mathcal{W}^{surj}$-pedigrad for (Ω, D).*

Proof. Before showing the statement, recall that for every monomorphism $m : A \to B$ in $\mathbf{Set}$, the function $\mathbf{Set}(j, E) : \mathbf{Set}(B, E) \to \mathbf{Set}(A, E)$ is a surjection. Indeed, because E has a pointed structure, every function $f : A \to E$ can be extended to a function $f' : B \to E$ by mapping every $x \in B \backslash A$ to the point ε of E. We can check that the identity $f = f' \circ j$ holds, which amounts to saying that the image of j via the functor $\mathbf{Set}(\_, E) : \mathbf{Set}^{\mathrm{op}} \to \mathbf{Set}$ is a surjection.

We now prove the statement. Let $\rho : \Delta_A(\tau) \Rightarrow \theta$ be a cone in $D[n_1]$ for some given non-negative integer n_1. Because (Ω, D) is a b-injective chromology, it follows from Definition 3.40 and Definition 3.43 that the canonical arrow

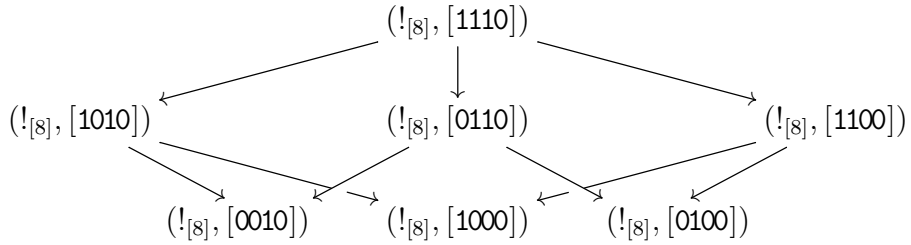
$$\mathsf{colim}_A \mathsf{Tr}_b\theta \to \mathsf{Tr}_b(\tau)$$

is a monomorphism in $\mathbf{Set}$. As a result, the image of this arrow via the functor $\mathbf{Set}(\_, E) : \mathbf{Set}^{\mathrm{op}} \to \mathbf{Set}$ is a surjection. By Proposition 3.17 and the usual definition of colimits in $\mathbf{Set}$, the resulting surjection is (naturally) isomorphic to the following canonical arrow.

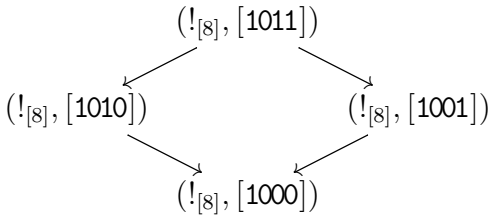
$$E^{\varepsilon}_{b}(\tau) \to \mathsf{lim}_A E^{\varepsilon}_{b} \circ \theta$$

This precisely shows that $E^{\varepsilon}_{b} : \mathbf{Seg}(\Omega) \to \mathbf{Set}$ is a $\mathcal{W}^{\mathrm{surj}}$-pedigrad for (Ω, D). ∎

3.50. Example. [Controlling uncertainties via chromologies] The goal of this example is to show that the presence of uncertainties discussed at the end of Example 3.38 can be controlled through the design of chromologies. We will keep the same notations as those used therein. First, recall that the point of Example 3.37 was to show that the following exactly $(1,1,1,1)$-distributive cone in $\mathbf{Seg}(\Omega^{\times 4})$ was suitable to define an exactly $(1,1,1,1)$-distributive chromology that would make the right Kan extension $\mathsf{Ran}_\iota T : \mathbf{Seg}(\Omega^{\times 4}) \to \mathbf{Set}$ a $\mathcal{W}^{\mathrm{bij}}$-pedigrad.



On the other hand, the point of Example 3.38 was to show that not every exactly $(1,1,1,1)$-distributive cone is suitable to make a functor a $\mathcal{W}^{\mathrm{bij}}$-pedigrad. In particular, it was shown that the following cone could not make the right Kan extension $\mathsf{Ran}_\iota T$ a $\mathcal{W}^{\mathrm{bij}}$-pedigrad, but only a $\mathcal{W}^{\mathrm{surj}}$-pedigrad



(21)

The reason for this obstruction was that the genetic data of `Craig` was much different from that of `Doug`. As a result, the image of the functor $T : B \to \mathbf{Set}$ at the segment $(!_{[9]}, [0011])$ was non-trivial and thus prevented the canonical limit arrow associated with the previous cone from being a bijection. At least, knowing that it is a surjection tells us that there is no inconsistencies between the table of `Anne` and `Craig` and that of `Anne` and `Doug`

Note that the obstruction associated with (21) to make $\mathsf{Ran}_\iota T$ a $\mathcal{W}^{\mathrm{bij}}$-pedigrad could be reduced if we could prevent the objects of (21) from going to segments of B that are associated with this so-called uncertainty. This would prevent the limit construction of $\mathsf{Ran}_\iota T$ from considering too many images of T in its computation.

A way to do so could be to change the topology of the segments on which the functor $T : B \to \mathbf{Set}$ is defined. Of course, being able to do so would mean that we either know more about our problem or that we make an assumption about our four individuals. For example, we could consider the situation in which one decides to parenthesize all the adjacent matches that appear in the pairwise sequence alignments of T together. Here is an illustration. First, recall that the image of $T(!_{[9]}, [0011])$ was taken to be as follows (see Example 3.25).

$$T(!_{[9]},[0011]) = \left\{ \begin{matrix} \texttt{ACCGT}\varepsilon\texttt{C}\varepsilon\texttt{A} \\ \texttt{A}\varepsilon\texttt{C}\varepsilon\texttt{TACTG} \end{matrix} ; \begin{matrix} \texttt{AC}\varepsilon\varepsilon\texttt{CGTCA} \\ \texttt{ACTAC}\varepsilon\texttt{T}\varepsilon\texttt{G} \end{matrix} ; \begin{matrix} \texttt{ACCGT}\varepsilon\texttt{CA}\varepsilon \\ \texttt{A}\varepsilon\texttt{C}\varepsilon\texttt{TACTG} \end{matrix} ; \begin{matrix} \texttt{AC}\varepsilon\varepsilon\texttt{CGTCA} \\ \texttt{ACTAC}\varepsilon\texttt{TG}\varepsilon \end{matrix} \right\}$$

If we now parenthesize the matching and mismatching patches of maximal lengths in the sequence alignments of $T(!_{[9]},[0011])$, we obtain the parenthesized sequence alignments shown in the following table.

$T(!_{[9]},[0011])$	
$T(t_1,[0011]) =$	$\left\{ \begin{matrix} (\texttt{A})(\texttt{C})(\texttt{C})(\texttt{G})(\texttt{T})(\varepsilon)(\texttt{C})(\varepsilon\texttt{A}) \\ (\texttt{A})(\varepsilon)(\texttt{C})(\varepsilon)(\texttt{T})(\texttt{A})(\texttt{C})(\texttt{TG}) \end{matrix} , \begin{matrix} (\texttt{A})(\texttt{C})(\texttt{C})(\texttt{G})(\texttt{T})(\varepsilon)(\texttt{C})(\texttt{A}\varepsilon) \\ (\texttt{A})(\varepsilon)(\texttt{C})(\varepsilon)(\texttt{T})(\texttt{A})(\texttt{C})(\texttt{TG}) \end{matrix} \right\}$
$T(t_2,[0011]) =$	$\left\{ \begin{matrix} (\texttt{AC})(\varepsilon\varepsilon)(\texttt{C})(\texttt{G})(\texttt{T})(\texttt{CA}) \\ (\texttt{AC})(\texttt{TA})(\texttt{C})(\varepsilon)(\texttt{T})(\varepsilon\texttt{G}) \end{matrix} , \begin{matrix} (\texttt{AC})(\varepsilon\varepsilon)(\texttt{C})(\texttt{G})(\texttt{T})(\texttt{CA}) \\ (\texttt{AC})(\texttt{TA})(\texttt{C})(\varepsilon)(\texttt{T})(\texttt{G}\varepsilon) \end{matrix} \right\}$

Because this bracketing suggests the use of two new topologies, we want to stay consistent with the definition of T and associate the previous alignments with the two segments of non-terminal topologies shown on the left-hand side of the previous table. Since a segment of trivial topology $!_{[8]} : [8] \to [1]$ cannot be mapped to segments of non-terminal topologies, the domain of the canonical arrow

$$\mathsf{Ran}_\iota T(!_{[8]},[1011]) \to \mathsf{Ran}_\iota T \circ F, \tag{22}$$

where the diagram F is the lower cospan of diagram (21), will not contain the sets $T(t_1,[0011])$ and $T(t_2,[0011])$ (and the term $T(!_{[9]},[0011])^{\times 9}$ no longer appears in the domain). This means that arrow (22) is more likely to be a bijection of sets, hence making diagram (21) more likely to be suitable for the definition of a chromology that makes $\mathsf{Ran}_\iota T$ a $\mathcal{W}^{\mathrm{bij}}$-pedigrad.

We thus conclude that the process of looking for chromologies (or, in fact, their cones), given a set of possible functors $T : B \to \mathbf{Set}$, can be seen as a way of isolating uncertainties and hence producing a refined analysis of the genetic data of `Anne`, `Bob`, `Craig` and `Doug`.

4. Solving our problem and identifying mechanisms

In this section, we formalize what should be seen as the categorical answer of the problem exposed in section 2.1, namely a method to assess the validity of the multiple sequence alignments computed by a right Kan extension of a sequence alignment functor. More specifically, we show that chromologies give a way to select multiple sequence alignments with respect to various types of mechanisms. As usual, we will let (E,ε) denote a pointed set and $(\Omega,\preceq)$ be a pre-ordered set.

4.1. Link between right Kan extensions and pedigrads.

In this section, we recall the definition of the unit associated with a right Kan extension and relate this unit to the concepts of pedigrad and chromology.

4.2. Convention. [Notations] *For every small category C, we will denote by $[C, \mathbf{Set}]$ the category whose objects are functors from C to* **Set** *and whose arrows are natural transformations.*

The following proposition states that a right Kan extension in **Set** along a certain functor ι (Definition 3.32) is a right adjoint for the pre-composition functor induced by the functor ι – see [12, Chapter X] for more detail.

4.3. Theorem. [12] *Let $(\Omega, \preceq)$ be a pre-ordered set and $\iota : B \to \mathbf{Seg}(\Omega)$ be a functor. The right Kan extension operation Ran_ι induces a functor $[B, \mathbf{Set}] \to [\mathbf{Seg}(\Omega), \mathbf{Set}]$ that maps any natural transformation $\sigma : A \Rightarrow B$ in* **Set** *over B to the canonical natural transformation (23) induced by the limit construction of Definition 3.32.*

$$\mathsf{Ran}_\iota \sigma : \mathsf{lim}_{(\tau \downarrow \iota)} A \circ \iota_\tau \Rightarrow \mathsf{lim}_{(\tau \downarrow \iota)} B \circ \iota_\tau \tag{23}$$

The resulting functor $\mathsf{Ran}_\iota : [B, \mathbf{Set}] \to [\mathbf{Seg}(\Omega), \mathbf{Set}]$ is a right adjoint for the pre-composition functor induced by ι.

4.4. Remark. [Units of right Kan extensions] The present remark reminds the reader about the form of the unit associated with the adjunction described in Theorem 4.3. Let $(\Omega, \preceq)$ be a pre-ordered set and $\iota : B \to \mathbf{Seg}(\Omega)$ be a functor. As a right adjoint of the pre-composition functor

$$\_ \circ \iota : [\mathbf{Seg}(\Omega), \mathbf{Set}] \to [B, \mathbf{Set}],$$

the functor Ran_ι is associated with a natural transformation, called the *unit*, of the form shown in (24) for every functor $P : \mathbf{Seg}(\Omega) \to \mathbf{Set}$.

$$\eta : P \Rightarrow \mathsf{Ran}_\iota (P \circ \iota) \tag{24}$$

The components of (24) correspond to the canonical arrows associated with the limit construction of Definition 3.32. This means that the evaluation of the previous natural transformation at an object τ in $\mathbf{Seg}(\Omega)$, as shown below in (25), is the limit adjoint arrow for the cone described in Remark 3.30.

$$P(\tau) \to \mathsf{lim}_{(\tau \downarrow \iota)} P \circ \iota_\tau \circ \iota \tag{25}$$

4.5. Remark. [Right Kan extensions and pedigrads] The present remark extends Remark 4.4 and shows how right Kan extensions and pedigrads are related. First, note that the arrow shown in (25) looks a lot like the type of arrow used in Definition 3.46 and Definition 3.48 to define $\mathcal{W}^{\mathrm{bij}}$- and $\mathcal{W}^{\mathrm{surj}}$-pedigrads. To make this more precise, let us denote by $\rho_\iota[\tau]$ the cone defined in Remark 3.30, that is to say the obvious natural transformation

$$\Delta_{(\tau \downarrow \iota)}(\tau) \Rightarrow \iota_\tau \circ \iota$$

induced by the objects of the category $(\tau \downarrow \iota)$ over the diagram formed by its arrows. Let us also suppose that the cone $\rho_i[\tau]$ is part of a chromology (Ω, D). In this case, we can notice two facts:

- if P is a $\mathcal{W}^{\text{bij}}$-pedigrad for (Ω, D), then arrow (25) is a bijection.

- if P is a $\mathcal{W}^{\text{surj}}$-pedigrad for (Ω, D), then arrow (25) is a surjection.

Later on, we will use these two facts with some functor P that is the composition of functors of the form described in Definition 3.13 and Example 2.27 (for instance, see Convention 4.8). In particular, if the pedigrad P can be written as a composite $Q \circ R$ where Q is a more canonical pedigrad, then we will prefer to apply the previous two points to Q, namely by looking at whether the image of the cone $\rho_\iota[\tau]$ via R is in the chromology of Q.

4.6. Slices of a sequence alignment. In this section, we define the concept of slice for a sequence alignment functor. This concept will later be used to reason about the possible mutation mechanisms contained in a sequence alignment functor. Throughout this section, we shall let $\mathbf{A}$ be an alignment specification of the form $\{f_i : (\Omega, \preceq) \to (\Omega_i, \preceq_i)\}_{i \in A}$.

4.7. Remark. [Projection maps] Let (E, ε) be a pointed set. For every element b in Ω, the product structure of $\mathbf{A}E_b^\epsilon$ (see Definition 3.21) gives us the following natural projection for every $i \in A$.

$$\kappa_i : \mathbf{A}E_b^\varepsilon \Rightarrow E_{f_i(b)}^\varepsilon \circ \mathbf{Seg}(f_i)$$

This arrow, living in $[\mathbf{Seg}(\Omega), \mathbf{Set}]$, will repeatedly be used throughout this section.

4.8. Convention. [Notation] *For the sake of convenience, for every element $i \in A$ and element $b \in \Omega$, we will let $f_i^* E_b^\varepsilon$ denote the composite functor $E_{f_i(b)}^\varepsilon \circ \mathbf{Seg}(f_i) : \mathbf{Seg}(\Omega) \to \mathbf{Set}$.*

In Example 3.38, we saw that, even for exactly distributive chromologies (Definition 3.43), the right Kan extension of a sequence alignment functor is not necessarily a $\mathcal{W}^{\text{bij}}$-pedigrad. On the other hand, the functor of Definition 3.13 was shown to be a $\mathcal{W}^{\text{bij}}$-pedigrad for any such cone (Theorem 3.47). The idea of Definition 4.9, given below, is to compare a non-pedigrad object to a pedigrad object in order to detect the pieces of pedigradic information that would live in the non-pedigrad object.

4.9. Definition. [Slices] *Let b be an element in Ω and (ι, T, σ) be a sequence alignment functor over $\mathbf{A}E_b^\varepsilon$. For every element $i \in A$, we will speak of the i-*slice *of (ι, T, σ) to refer to the pullback arrow $\eta_i^* : [T/\mathbf{A}E_b^\varepsilon]_i \Rightarrow \mathsf{Ran}_\iota T$ of the unit of the right Kan extension at the functor $f_i^* E_b^\varepsilon$ along the natural transformation $\mathsf{Ran}_\iota(\kappa_i \circ \sigma)$ (as shown below).*

$$\begin{array}{ccc} [T/\mathbf{A}E_b^\varepsilon]_i & \Longrightarrow & f_i^* E_b^\varepsilon \\ {\scriptstyle \eta_i^*}\Downarrow & \lrcorner & \Downarrow{\scriptstyle \eta} \\ \mathsf{Ran}_\iota T & \underset{\mathsf{Ran}_\iota \sigma}{\Longrightarrow} \mathsf{Ran}_\iota(\mathbf{A}E_b^\varepsilon \circ \iota) & \underset{\mathsf{Ran}_\iota \kappa_i}{\Longrightarrow} \mathsf{Ran}_\iota(f_i^* E_b^\varepsilon \circ \iota) \end{array} \tag{26}$$

4.10. Remark. [General *versus* individual slices] Throughout the present section, the reader may wonder why we only consider the pullback of η along a natural transformation of the form $\mathsf{Ran}_\iota(\kappa_i \circ \sigma)$ while the pullback of η along the natural transformation $\mathsf{Ran}_\iota(\sigma)$ is left out. The reason is that the former integrates the data with respect to a unique 'individual' $i \in A$ while the latter consider all 'individuals' in A. As a result, the latter will contain very few elements, if any. The algorithm proposed in Remark 4.14 tries to maximize both the size of the integrated data and the number of individuals considered.

4.11. Remark. [Data integration along cones] The idea behind the slice of a sequence alignment functor is to select the multiple sequence alignments of $\mathsf{Ran}_\iota T$ for which the type of uncertainty described in Example 3.38 and Example 3.50 can be resolved from the point of view of a particular individual. While Example 4.12 will illustrate how this type of uncertainty can be eliminated, the present remark shows how the cones of a chromology inform us of the ways via which we can safely integrate the data through slices.

Let b be an element in Ω, (ι, T, σ) be a sequence alignment functor over $\mathbf{A}E_b^\varepsilon$ and τ be an object in $\mathbf{Seg}(\Omega)$. For every element $i \in A$, evaluating the i-slice of (ι, T, σ) at τ gives us a pullback square in **Set** as follows.

$$\begin{array}{ccc} [T/\mathbf{A}E_b^\varepsilon]_i(\tau) & \xrightarrow{\sigma_\tau^*} & f_i^* E_b^\varepsilon(\tau) \\ \downarrow & & \downarrow{\scriptstyle \eta_\tau} \\ \mathsf{lim}_{(\tau\downarrow\iota)} T(\tau) & \xrightarrow[\mathsf{Ran}_\iota(\kappa_i\circ\sigma)_\tau]{} & \mathsf{lim}_{(\tau\downarrow\iota)} f_i^* E_b^\varepsilon \circ \iota_\tau \circ \iota \end{array} \tag{27}$$

Now, recall that, by universal property of pullbacks, the pullback of an isomorphism (resp. a surjection) is an isomorphism (resp. a surjection). By Remark 4.5, this means that if

- the image of the cone $\rho_\iota[\tau]$ via $\mathbf{Seg}(f_i)$ is in the chromology of $E^\varepsilon_{f_i(b)}$, and
- the functor $E^\varepsilon_{f_i(b)}$ is a $\mathcal{W}^{\mathrm{bij}}$-pedigrad (resp. $\mathcal{W}^{\mathrm{surj}}$-pedigrad),

then the leftmost vertical arrow of diagram (27) is an isomorphism (resp. a surjection). In this sense, we would like to say that all the gluings computed by $\mathsf{Ran}_\iota T$ make sense (resp. make sense up to some uncertainty) from the point of view of f_i, for they can be lifted to the pullback $[T/\mathbf{A}E_b^\varepsilon]_i(\tau)$.

Thus, the role of the functor $[T/\mathbf{A}E_b^\varepsilon]_i$ is to take care of selecting all those multiple sequence alignments generated by T that make sense with the component $f_i : (\Omega, \preceq) \to (\Omega, \preceq_i)$, where the idea of "making sense" is strongly related to the pedigrad structure of the functor

$$E^\varepsilon_{f_i(b)} : \mathbf{Seg}(\Omega_i) \to \mathbf{Set}.$$

In Example 4.12, we show that something more subtle happens when the cone $\mathbf{Seg}(f_i)\rho_\iota[\tau]$ is not in the chromology of $E^\varepsilon_{f_i(b)}$.

4.12. Example. [Resolving uncertainties] Let (ι, T, σ) be the sequence alignment constructed in Example 3.25. The present example discusses the meaning of the elements contained in the image $\mathsf{Ran}_\iota T(!_{[8]}, [\mathtt{1011}])$, which we computed in Example 3.38, from the point of view of slices. We shall consider the same notations as those used in Example 3.38, but, for convenience, we will denote the segment $(!_{[8]}, [\mathtt{1011}])$ of $\mathbf{Seg}(\Omega^{\times 4})$ as τ. In this case, the evaluation of diagram (26) at τ is of the following form for every $i \in \{\mathtt{a}, \mathtt{b}, \mathtt{c}, \mathtt{d}\}$.

$$\begin{array}{ccccc} [T/\mathbf{A}E_b^\varepsilon]_i(\tau) & & \xrightarrow{\quad\sigma_\tau^*\quad} & & \pi_i^* E_b^\varepsilon(\tau) \\ \downarrow & & & & \downarrow{\scriptstyle \eta_\tau} \\ L_8([\mathtt{1011}]) \times T(!_{[9]}, [\mathtt{0011}])^{\times 9} & \xrightarrow[\mathsf{Ran}_\iota(\sigma)_\tau]{} & \mathsf{Ran}_\iota(\mathbf{A}E_b^\varepsilon \circ \iota)(\tau) & \xrightarrow[\mathsf{Ran}_\iota(\kappa_i)_\tau]{} & \mathsf{Ran}_\iota(\pi_i^* E_b^\varepsilon \circ \iota)(\tau) \end{array} \tag{28}$$

We are going to illustrate a case in which the 9-fold Cartesian product

$$T(!_{[9]}, [\mathtt{0011}])^{\times 9} \tag{29}$$

appearing in the left-bottom corner of (28) (*i.e* the object $\mathsf{Ran}_\iota T(\tau)$) prevents pullback (28) from lifting any element in $\mathsf{Ran}_\iota T(\tau)$ to the i-slice $[T/\mathbf{A}E_b^\varepsilon]_i(\tau)$. The reason for this obstruction is that object (29) contains tuples that fail to match the type of tuples associated with the images of the function η_τ (given on the right of diagram (28)).

Before discussing the aforementioned obstruction, let us describe the form of the mappings associated with the arrows of diagram (28) in more detail (see diagram (30), below, for future reference). First, recall that every copy $T(!_{[9]}, [\mathtt{0011}])$ of the 9-fold Cartesian product given in (29) is associated with one of the 9 morphisms of the form $(!_{[8]}, [\mathtt{1011}]) \to (!_{[9]}, [\mathtt{0011}])$ in $\mathbf{Seg}(\{0,1\}^{\times 4})$ (see examples below).

$$(\overset{1}{\bullet}\overset{2}{\bullet}\overset{3}{\bullet}\overset{4}{\bullet}\overset{5}{\bullet}\overset{6}{\bullet}\overset{7}{\bullet}) \longrightarrow (\overset{*}{\bullet}\overset{1}{\bullet}\overset{2}{\bullet}\overset{3}{\bullet}\overset{4}{\bullet}\overset{5}{\bullet}\overset{6}{\bullet}\overset{7}{\bullet}), \qquad (\overset{1}{\bullet}\overset{2}{\bullet}\overset{3}{\bullet}\overset{4}{\bullet}\overset{5}{\bullet}\overset{6}{\bullet}\overset{7}{\bullet}) \longrightarrow (\overset{1}{\bullet}\overset{*}{\bullet}\overset{2}{\bullet}\overset{3}{\bullet}\overset{4}{\bullet}\overset{5}{\bullet}\overset{6}{\bullet}\overset{7}{\bullet}), \qquad \text{etc.}$$

Let us denote these morphisms as $g_k : (!_{[8]}, [\mathtt{1011}]) \to (!_{[9]}, [\mathtt{0011}])$ for every $k \in [9]$. The leftmost horizontal arrow given at the bottom of (28) then sends a pair (x, y), where x is an element of $L_8([\mathtt{1011}])$ and $y = (y_1, \ldots, y_9)$ is an element of the 9-fold Cartesian product of $T(!_{[9]}, [\mathtt{0011}])$, to the same tuple (x, y) in $\mathsf{Ran}_\iota(\mathbf{A}E_b^\varepsilon \circ \iota)(\tau)$ provided that one sees x and $y_1, \ldots, y_9$ as elements taken from the images of $\mathbf{A}E_b^\varepsilon$ (see Definition 3.23). Then, the tuple (x, y) in $\mathsf{Ran}_\iota(\mathbf{A}E_b^\varepsilon \circ \iota)(\tau)$ is sent to a tuple of the form $(x_i, (y_{1,i}, \ldots, y_{9,i}))$ made of the i-th projections of x and $y_1, \ldots, y_9$ with respect to the 4-fold product structure underlying the definition of $\mathbf{A}E_b^\varepsilon$. It follows that the tuple (x, y) living in the left-bottom corner of (28) is lifted to the left-top corner of (28) if there exists an element z in $\pi_i^* E_b^\varepsilon(\tau)$ whose image through η_τ is equal to the image $\mathsf{Ran}_\iota(\kappa_i)_\tau(x, y)$.

$$\begin{array}{ccccc} ((x,y),z) & \vdash\!\!- - - - - - - - - - - - \to & & & z \\ \top \downarrow \mid & & & & \downarrow{\scriptstyle \eta_\tau} \\ (x,y) & \xmapsto{\mathsf{Ran}_\iota(\sigma)_\tau} & (x,y) & \xmapsto{\mathsf{Ran}_\iota(\kappa_i)_\tau} & (x_i, (y_{1,i}, \ldots, y_{9,i})) \end{array} \tag{30}$$

Let us now focus on a particular example. Let the index i be equal to the element $\mathtt{c}$ representing $\mathtt{Craig}$'s viewpoint. If we take z to be the element $\mathtt{ACCGTC}\varepsilon\mathtt{A}$ in $\pi_{\mathtt{c}}^* E_b^{\varepsilon}(!_{[8]}, [\mathtt{1011}])$, then its image through η_τ is of the following form.

$$\left(\mathtt{ACCGTC}\varepsilon\mathtt{A} \, , \left(\varepsilon\mathtt{ACCGTC}\varepsilon\mathtt{A} \, , \;\; \mathtt{A}\varepsilon\mathtt{CCGTC}\varepsilon\mathtt{A} \, , \ldots , \;\; \mathtt{ACCGTC}\varepsilon\mathtt{A}\varepsilon \right) \right) \tag{31}$$

On the other hand, the following element

$$\left(\begin{array}{l} \mathtt{ACCGACTG} \\ \mathtt{ACCGTC}\varepsilon\mathtt{A} \\ \mathtt{A}\varepsilon\mathtt{CTACTG} \end{array} , \left(\begin{array}{l} \mathtt{ACCGT}\varepsilon\mathtt{C}\varepsilon\mathtt{A} \\ \mathtt{A}\varepsilon\mathtt{C}\varepsilon\mathtt{TACTG} \end{array} , \begin{array}{l} \mathtt{ACCGT}\varepsilon\mathtt{C}\varepsilon\mathtt{A} \\ \mathtt{A}\varepsilon\mathtt{C}\varepsilon\mathtt{TACTG} \end{array} , \ldots , \begin{array}{l} \mathtt{ACCGT}\varepsilon\mathtt{C}\varepsilon\mathtt{A} \\ \mathtt{A}\varepsilon\mathtt{C}\varepsilon\mathtt{TACTG} \end{array} \right) \right) , \tag{32}$$

living in the image $\mathsf{Ran}_\iota T(\tau) = L_8([\mathtt{1011}]) \times T(!_{[9]}, [\mathtt{0011}])^{\times 9}$, is sent through $\mathsf{Ran}_\iota(\kappa_i \circ \sigma)_\tau$ to the following tuple in $\mathsf{Ran}_\iota(\pi_i^* E_b^{\varepsilon} \circ \iota)(\tau)$.

$$\left(\mathtt{ACCGTC}\varepsilon\mathtt{A} \, , \left(\mathtt{ACCGT}\varepsilon\mathtt{C}\varepsilon\mathtt{A} \, , \;\; \mathtt{ACCGT}\varepsilon\mathtt{C}\varepsilon\mathtt{A} \, , \ldots , \;\; \mathtt{ACCGT}\varepsilon\mathtt{C}\varepsilon\mathtt{A} \right) \right) \tag{33}$$

As can be seen, the element $z = \mathtt{ACCGTC}\varepsilon\mathtt{A}$ cannot be a valid integration of the element given in (32) because the image of (32) through $\mathsf{Ran}_\iota(\kappa_i \circ \sigma)_\tau$ – shown in (33) – is not equal to the image of z through η_τ – shown in (31). The reader can easily see that the main reason for this is that the rightmost tuple of (32) (which stands for $y = (y_1, \ldots, y_9)$) consists of equal components (*i.e.* $y_1 = \ldots = y_9$) while the existence of a lift would imply that the resulting collection of components $y_{1,i}, \ldots, y_{9,i}$ can equal the distinct components of the image of z through η_τ. Obviously, to do so, we would need to make the components of the rightmost tuple of (32) vary through various pairwise sequence alignments. Unfortunately, an analysis of the elements of $T(!_{[9]}, [\mathtt{0011}])$ quickly reveals that the set $T(!_{[9]}, [\mathtt{0011}])$ does not contain enough elements to make the elements of $L_8([\mathtt{1011}]) \times T(!_{[9]}, [\mathtt{0011}])^{\times 9}$ match the images of the morphism η_τ. This shows that there is not enough evidence that the alignment

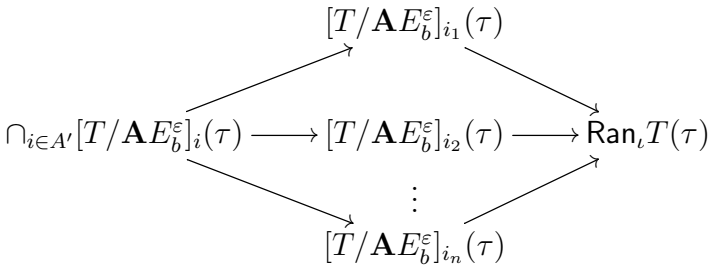
$$\begin{array}{l} \mathtt{ACCGACTG} \\ \mathtt{ACCGTC}\varepsilon\mathtt{A} \\ \mathtt{A}\varepsilon\mathtt{CTACTG} \end{array} \tag{34}$$

is a good alignment from the point of view of $\mathtt{Craig}$ (*i.e.* $i = \mathtt{c}$). Here the main obstruction is that the exponent of $T(!_{[9]}, [\mathtt{0011}])^{\times 9}$ is too big for the cardinality of $T(!_{[9]}, [\mathtt{0011}])$ (which is due to the uncertainty related to aligning distant DNA sequences). On the other hand, adding more colors (see Example 3.26) and using more complex topologies (see Example 3.50) can reduce the exponent of $T(!_{[9]}, [\mathtt{0011}])^{\times 9}$, which would have the consequence of making alignment (34) more likely to be liftable from the point of view of the added knowledge.

4.13. Remark. [Multiple sequence alignments and mechanisms] Tuple (32), displayed in Example 4.12, shows us that slices produce two types of data integration. The first type of integration looks at the construction of multiple sequence alignments, such as the one shown on the left of tuple (32), while the second type of integration concerns the

other part of tuple (32) that consists of the collection of pairwise sequence alignments. Example 4.12 does not tell us much about this type of integration, except for creating some uncertainty. In section 4.15, we will see that this uncertainty is actually associated with the presence and detection of mutation mechanisms. In the case of Example 4.12, no mechanism could be recognized from the data, which translates into an absence of lift.

4.14. Remark. [Finding a multiple sequence alignment] Example 4.12 and Remark 4.13 implicitly motivate an algebraic method to select multiple sequence alignments along with mechanisms. Specifically, the method would look at the pullbacks of the slices of a certain sequence alignment functor, say (ι, T, σ) and, for a given segment τ, would try to find the maximal subset $A' \subseteq A$ for which the wide pullback of the i-slices, for every $i \in A'$, is maximal at the segment τ (see below).



Ideally, the segment τ should only be made of a maximal color, but segments of intermediate colors could also be used for heuristics, if necessary.

4.15. Slices and mechanisms. In section 4.6, we showed that the cones associated with the slices of a sequence alignment functor (ι, T, σ) could be used to query multiple sequence alignments that make sense from the point of view of a particular individual (see Example 4.12). Then, in Remark 4.14, we suggested that these queries could be used to find a set of multiple sequence alignments in the right Kan extension of T along ι that maximize the number of individuals agreeing with the alignments.

The goal of this section is to show that the querying process inherent to slices (Definition 4.9) can be used to query mechanisms, too. Here, we will show that mechanisms such as duplication events, which are mutations responsible for triggering certain cancers [18], and inversion events, which are rearrangements of certain sections of a segment in reverse order, can be detected through particular types of cones (see Remark 4.11).

Note that previous works have already investigated the recognition of duplication and inversion mechanisms in multiple sequence alignments [21, 20, 23, 29]. While these papers develop methods for predetermined patterns, slices are more flexible in that they inform us of existing patterns without requiring us to know what these patterns should look like. More specifically, these patterns are encoded in the shape of the cones described in Remark 4.11 and are thus systematically given through the computation of right Kan extension. The recognition of mechanisms could then be done by comparing the shape of the cones for which the slices are non empty with the shape of cones that are known to characterize specific mechanisms.

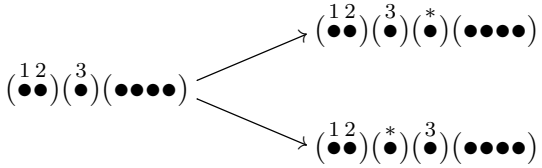
The present section does not introduce any new concept and only aims to show examples. The reader will be assumed to remember the reasoning of Example 4.12, which we intend to mimic in this section – the goal is again to match certain tuples through the arrows of diagram (26).

4.16. Example. [Duplication mechanisms] For our first example, we shall let $\mathbf{A}$ denote the alignment specification defined by the following collection of projections (already used in Example 3.20)

$$\{\pi_i : \{0,1\}^{\times 4} \to \{0,1\}\}_{i \in \{\mathtt{a},\mathtt{b},\mathtt{c},\mathtt{d}\}}$$

and let b be the element $(1,1,1,1)$ of $\{0,1\}^{\times 4}$. Because the sequence alignment functor of Example 3.25 was built from data that are not suited for a good illustration of duplication mechanisms, we will let (ι, T, σ) be an undetermined sequence alignment functor over $\mathbf{A}E_b^{\varepsilon}$, where E is the set $\{\mathtt{A}, \mathtt{C}, \mathtt{G}, \mathtt{T}, \varepsilon\}$. Our goal is to illustrate the types of situations in which duplication mechanisms can be queried within the right Kan extension.

For every object τ in $\mathbf{Seg}(\{0,1\}^{\times 4})$, we will denote by $\rho_\iota[\tau]$ the cone in $\mathbf{Seg}(\{0,1\}^{\times 4})$ encoded by the arrows and objects of the category $(\tau \downarrow \iota)$ (see Remark 3.30 for a detailed description). As in Example 4.12, we want to study the functor T from the point of view of a certain individual; we will consider `Craig`, who is, as usual, associated with the index `c`. Now, suppose that the image of the cone $\rho_\iota[\tau]$ via the functor $\mathbf{Seg}(\pi_{\mathtt{c}}) : \mathbf{Seg}(\{0,1\}^{\times 4}) \to \mathbf{Seg}(\{0,1\})$ is of the form shown in (35).



(35)

For such a cone, the pullback of Definition 4.9 lifts any element of $\mathsf{Ran}_\iota T(\tau)$ to `Craig`'s slice if this element can be sent, through the function $\mathsf{Ran}_\iota(\kappa_{\mathtt{c}} \circ \sigma)_\tau : \mathsf{Ran}_\iota T(\tau) \to \mathsf{Ran}_\iota(\pi_{\mathtt{c}}^* E_b^{\varepsilon} \circ \iota)(\tau)$, to an image of η_τ in $\mathsf{Ran}_\iota(\pi_{\mathtt{c}}^* E_b^{\varepsilon} \circ \iota)(\tau)$ – for instance, a pair of the following form.

$$\eta_\tau(\mathtt{x_1x_2Zx_3x_4x_5x_6}) = (\mathtt{x_1x_2\varepsilon Zx_3x_4x_5x_6},\ \mathtt{x_1x_2Z\varepsilon x_3x_4x_5x_6}) \tag{36}$$

If our sequence alignment functor (ι, T, σ) is constructed from the outputs of a dynamic programming algorithm, as in Example 3.25, then the elements of $\mathsf{Ran}_\iota T(\tau)$ that match pair (36) through the function $\mathsf{Ran}_\iota(\kappa_{\mathtt{c}} \circ \sigma)_\tau$ will most likely be tuples of sequence alignments in which each of the components of tuple (36) appears. An example of such an element in $\mathsf{Ran}_\iota T(\tau)$ is given by the following pair of sequence alignments, which tries to align the sequence $\mathtt{x_1x_2ZZx_3x_4x_5x_6}$ with the sequence $\mathtt{x_1x_2Zx_3x_4x_5x_6}$.

$$\begin{pmatrix} \mathtt{x_1x_2\varepsilon Zx_3x_4x_5x_6} & \mathtt{x_1x_2Z\varepsilon x_3x_4x_5x_6} \\ \mathtt{x_1x_2ZZx_3x_4x_5x_6}, & \mathtt{x_1x_2ZZx_3x_4x_5x_6} \end{pmatrix}$$

In this case, the tuple given below lives in the image of `Craig`'s slice at the segment τ (*i.e.* $[T/\mathbf{A}E_b^{\varepsilon}]_{\mathtt{c}}(\tau)$) and plays the role of the desired lift.

$$\begin{pmatrix} \mathtt{x_1x_2\varepsilon Zx_3x_4x_5x_6} & \mathtt{x_1x_2Z\varepsilon x_3x_4x_5x_6} & \\ \mathtt{x_1x_2ZZx_3x_4x_5x_6}, & \mathtt{x_1x_2ZZx_3x_4x_5x_6}, & \mathtt{x_1x_2Zx_3x_4x_5x_6} \end{pmatrix}$$

Intuitively, the previous element tells us that `Craig` is separated from a certain other individual (whose genetic data is $\mathtt{x_1x_2ZZx_3x_4x_5x_6}$) by a duplication mechanism. On the other hand, if the sequence of the other individual were of a different form, say $\mathtt{x_1x_2ZYx_3x_4x_5x_6}$, the scoring system of the dynamic programming algorithm would only be expected to provide the sequence alignment given below, on the left (at least for an adequate scoring system).

$$\begin{array}{ll} \mathtt{x_1x_2Z\varepsilon x_3x_4x_5x_6} & \cancel{\mathtt{x_1x_2\varepsilon Zx_3x_4x_5x_6}} \\ \mathtt{x_1x_2ZYx_3x_4x_5x_6} & \cancel{\mathtt{x_1x_2ZYx_3x_4x_5x_6}} \end{array}$$

In the end, this would prevent the existence of an element in $\mathsf{Ran}_\iota T(\tau)$ matching tuple (36) and would hence prevent the existence of a lift to `Craig`'s slice.

To conclude, designing the domain B of the functor T (either using colors or brackets) such that cone (35) is the image of a cone of the form $\rho_\iota[\tau]$ via $\mathbf{Seg}(\pi_{\mathtt{c}})$ will force the pullback of Definition 4.9 to select alignments that may be explained by duplication mechanisms – at least from `Craig`'s viewpoint.

4.17. Remark. [Types of cones detecting mechanisms] Note that, while multiple sequence alignments would be lifted along cones that belong to chromologies (as in Remark 4.11), mechanisms (such as duplications) would usually be lifted along cones that are not part of chromologies, since they would usually be made of morphisms of the type described in Example 2.23 to create mutation events (see diagram (35) and the definition of section 2.36).

4.18. Example. [Inversion mechanisms] Let us now give an example of a cone that lifts inversion mechanisms. This time, we will need to take Ω to be the pre-ordered set $\{0 \leq 1 \leq 2\}$. We will implicitly use the color 2 of Ω to restrict the number of arrows that $\rho_\iota[\tau]$ may possess (see diagram (37)). In addition, we will to take $\mathbf{A}$ to be the alignment specification consisting of the following projections.

$$\{\pi_i : \{0,1,2\}^{\times 4} \to \{0,1,2\}\}_{i \in \{\mathtt{a},\mathtt{b},\mathtt{c},\mathtt{d}\}}$$

The element b will be taken to be equal to the element $(1,1,1,1)$ of $\{0,1,2\}^{\times 4}$ and (ι, T, σ) will denote an undetermined sequence alignment over $\mathbf{A}E_b^\varepsilon$, where E is the set $\{\mathtt{A},\mathtt{C},\mathtt{G},\mathtt{T},\varepsilon\}$. For the present example, we will suppose that the image of the cone $\rho_\iota[\tau]$ via the functor $\mathbf{Seg}(\pi_{\mathtt{c}}) : \mathbf{Seg}(\{0,1,2\}^{\times 4}) \to \mathbf{Seg}(\{0,1,2\})$ is of the form given in (37).

$$\begin{array}{ccc} & & (\overset{1}{\mathtt{1}}\overset{2}{\mathtt{1}})(\overset{*}{\mathtt{2}})(\overset{*}{\mathtt{2}})(\overset{3}{\mathtt{1}})(\overset{4}{\mathtt{1}})(\overset{5}{\mathtt{1}})(\mathtt{1111}) \\ (\overset{1}{\mathtt{1}}\overset{2}{\mathtt{1}})(\overset{3}{\mathtt{1}})(\overset{4}{\mathtt{1}})(\overset{5}{\mathtt{1}})(\mathtt{1111}) & \longrightarrow & (\overset{1}{\mathtt{1}}\overset{2}{\mathtt{1}})(\overset{3}{\mathtt{1}})(\overset{*}{\mathtt{2}})(\overset{4}{\mathtt{1}})(\overset{*}{\mathtt{2}})(\overset{5}{\mathtt{1}})(\mathtt{1111}) \\ & & (\overset{1}{\mathtt{1}}\overset{2}{\mathtt{1}})(\overset{3}{\mathtt{1}})(\overset{4}{\mathtt{1}})(\overset{5}{\mathtt{1}})(\overset{*}{\mathtt{2}})(\overset{*}{\mathtt{2}})(\mathtt{1111}) \end{array} \tag{37}$$

We want to show that, for such a cone, `Craig`'s slice, evaluated at the segment τ, can detect inversion mechanisms. First, by the shape of cone (37), the pullback of Definition

4.9 lifts any element of $\mathsf{Ran}_\iota T(\tau)$ to `Craig`'s slice if this element can be sent, through the function $\mathsf{Ran}_\iota(\kappa_{\mathtt{c}} \circ \sigma)_\tau : \mathsf{Ran}_\iota T(\tau) \to \mathsf{Ran}_\iota(\pi_{\mathtt{c}}^* E_b^\varepsilon \circ \iota)(\tau)$, to an image of η_τ in $\mathsf{Ran}_\iota(\pi_{\mathtt{c}}^* E_b^\varepsilon \circ \iota)(\tau)$, and hence a triple of the following form.

$$(\mathtt{x_1x_2}\varepsilon\varepsilon\mathtt{ABCx_3x_4x_5x_6},\ \mathtt{x_1x_2A}\varepsilon\mathtt{B}\varepsilon\mathtt{Cx_3x_4x_5x_6},\ \mathtt{x_1x_2ABC}\varepsilon\varepsilon\mathtt{x_3x_4x_5x_6}) \tag{38}$$

If our sequence alignment functor (ι, T, σ) is constructed from the outputs of a dynamic programming algorithm, as in Example 3.25, then the elements of $\mathsf{Ran}_\iota T(\tau)$ that match pair (38) through the function $\mathsf{Ran}_\iota(\kappa_{\mathtt{c}} \circ \sigma)_\tau$ will most likely be tuples of sequence alignments in which each of the components of (38) appears. An example of such an element in $\mathsf{Ran}_\iota T(\tau)$ is given by the following triple of sequence alignments, which tries to align the sequence $\mathtt{x_1x_2ABCx_3x_4x_5x_6}$ with the sequence $\mathtt{x_1x_2CBAx_3x_4x_5x_6}$.

$$\left(\begin{array}{ccc} \mathtt{x_1x_2}\varepsilon\varepsilon\mathtt{ABCx_3x_4x_5x_6} & \mathtt{x_1x_2A}\varepsilon\mathtt{B}\varepsilon\mathtt{Cx_3x_4x_5x_6} & \mathtt{x_1x_2ABC}\varepsilon\varepsilon\mathtt{x_3x_4x_5x_6} \\ \mathtt{x_1x_2CBA}\varepsilon\varepsilon\mathtt{x_3x_4x_5x_6} & \mathtt{x_1x_2}\varepsilon\mathtt{CBA}\varepsilon\mathtt{x_3x_4x_5x_6} & \mathtt{x_1x_2}\varepsilon\varepsilon\mathtt{CBAx_3x_4x_5x_6} \end{array}\right) \tag{39}$$

In this case, the tuple displayed below lives in the image of `Craig`'s slice at the segment τ and plays the role of the desired lift.

$$\left(\begin{array}{ccc} \mathtt{x_1x_2}\varepsilon\varepsilon\mathtt{ABCx_3x_4x_5x_6} & \mathtt{x_1x_2A}\varepsilon\mathtt{B}\varepsilon\mathtt{Cx_3x_4x_5x_6} & \mathtt{x_1x_2ABC}\varepsilon\varepsilon\mathtt{x_3x_4x_5x_6} \\ \mathtt{x_1x_2CBA}\varepsilon\varepsilon\mathtt{x_3x_4x_5x_6} & \mathtt{x_1x_2}\varepsilon\mathtt{CBA}\varepsilon\mathtt{x_3x_4x_5x_6} & \mathtt{x_1x_2}\varepsilon\varepsilon\mathtt{CBAx_3x_4x_5x_6} \end{array}, \mathtt{x_1x_2ABCx_3x_4x_5x_6}\right)$$

As can be seen, this type of tuple tries to align the sequence $\mathtt{x_1x_2ABCx_3x_4x_5x_6}$ with the sequence $\mathtt{x_1x_2CBAx_3x_4x_5x_6}$, which are clearly related by an inversion of the patch `ABC`. As in Example 4.16, trying a different sequence, say $\mathtt{x_1x_2EFGx_3x_4x_5x_6}$, is unlikely to create a triple as in (39) and hence a lift to `Craig`'s slice. In other words, the earlier tuple specifically tells us that `Craig` is separated from a certain other individual (whose genetic data is $\mathtt{x_1x_2CBAx_3x_4x_5x_6}$) by an inversion mechanism.

To conclude, designing the domain B of the functor T such that cone (37) is the image of a cone of the form $\rho_\iota[\tau]$ via $\mathbf{Seg}(\pi_{\mathtt{c}})$ will force the pullback of Definition 4.9 to select alignments that may be explained by inversion mechanisms – at least from `Craig`'s viewpoint.

5. Conclusion

We formalized the concept of sequence alignment in terms of a subcategory B of segments (Definitions 2.14 & 2.20) and a functor $T : B \to \mathbf{Set}$ whose images contain usual sequence alignments (Definition 3.23). We showed that we could design comparison rules between the sequence alignments contained in T through the structure of the category B (Examples 3.25 & 3.26). We then showed that we could integrate multiple sequence alignments from the data contained in T by using the right Kan extension of T to the whole category of segments (section 3.27). In addition, we showed how the right Kan extension of T could be used to study the consistency of the integrated data through the use of limits and the existence of certain functions (see Example 3.28). While inconsistent

data could be associated with non-surjective functions, consistent data could be associated with either surjections or isomorphisms. Regarding these last two types of arrows, we showed that isomorphisms informed us of a perfect consistency (Example 3.37) while surjections indicated some uncertainty (Examples 3.36, & 3.38), which we later related to the presence of mutation mechanisms (Remark 4.13). We then introduced the concept of slice (Definition 4.9) as a way to resolve this uncertainty and, at the same time, discover mutation mechanisms (Remark 4.14 and section 4.15).

References

[1] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, D. J. Lipman, (5 Oct 1990) *Basic local alignment search tool*, Journal of Molecular Biology, Volume 215, Issue 3, pp 403–410.

[2] R. Brown, T. Porter, (2006), *Category Theory: an abstract setting for analogy and comparison*, In: What is Category Theory? Advanced Studies in Mathematics and Logic, Polimetrica Publisher, Italy, pp 257–274.

[3] A. Carbone, M. Gromov, (2001), *Mathematical slices of molecular biology*, Journal Article, La Gazette des Mathématiciens, Société Mathématique de France, special edition, pp 11–80.

[4] B. Chowdhury, G. Garai, (2017), *A review on multiple sequence alignment from the perspective of genetic algorithm*, Genomics, Volume 109, Issues 5–6, pp 419–431.

[5] J. Daugelaite, A. O' Driscoll, and R. D. Sleator, (2013), *An Overview of Multiple Sequence Alignments and Cloud Computing in Bioinformatics*, ISRN Biomathematics, Volume 2013, Article ID 615630, 14 pp.

[6] A. Ehresmann, (2013), *Le rôle des limites projectives dans le développement des mémoires procédurale et sémantique*, Séminaire Mamuphi, ENS; see http://ehres.pagesperso-orange.fr/.

[7] C. Ehresmann, (1968), *Esquisses et types de structures algébriques*. Bull. Instit. Polit. Iaşi **XIV**.

[8] D. F. Feng, R. F. Doolittle, (1987), *Progressive sequence alignment as a prerequisitetto correct phylogenetic trees*. J Mol Evol., Volume 25, Issue 4, pp 351–360.

[9] W. Lawvere, (1963), *Functorial Semantics of Algebraic Theories*. Ph.D. Thesis, Columbia University, New York, NY, USA.

[10] Y. Lazebnik, (Sep 2002), *Can a biologist fix a radio?–Or, what I learned while studying apoptosis.*, Cancer Cell., Volume 2, Issue 3, pp 179–82.

[11] Ari Löytynoja, Nick Goldman, (Jul 2005), *An algorithm for progressive multiple alignment of sequences with insertions*, Proceedings of the National Academy of Sciences, Volume 102, Issue 30, pp 10557–10562.

[12] S. Mac Lane, (1998), *Categories for the working mathematician*, Graduate texts in mathematics, Springer, New York.

[13] D. A. Morrison, (Feb 2009), *Why would phylogeneticists ignore computerized sequence alignment?*, Syst Biol., Volume 58, Issue 1, pp 150–158.

[14] D. A. Morrison, (Oct 2009), *A framework for phylogenetic sequence alignment*, Plant Systematics and Evolution, Volume 282, Issue 3–4, pp 127–149.

[15] D.W. Mount, (2004), *Bioinformatics: Sequence and Genome Analysis*, Cold Spring Harbor, NY: Cold Spring Harbor Laboratory Press, 692 pp.

[16] P. B. Needleman, C. D. Wunsch, (1970), *A general method applicable to the search for similarities in the amino acid sequence of two proteins*, Journal of Molecular Biology, Volume 48, Issue 3, pp 443–453

[17] E. Pennisi, (2013), *The CRISPR Craze*, Science, Volume 341, Issue 6148, pp 833–836.

[18] A. B. Reams, R. R. John, (Feb 2015), *Mechanisms of Gene Duplication and Amplification*, Cold Spring Harbor Perspectives in Biology, Volume 7, Issue 2: a016592.

[19] M. Rosenberg (Ed.), (2009), *Sequence Alignment: Methods, Models, Concepts, and Strategies*. University of California Press.

[20] M. Sammeth, J. Stoye, (2006), *Comparing Tandem Repeats with Duplications and Excisions of Variable Degree.*, IEEE/ACM Transactions on Computational Biology and Bioinformatics 3.

[21] M. Sammeth, T. Weniger, D. Harmsen, J. Stoye, (2005), *Alignment of Tandem Repeats with Excision, Duplication, Substitution and Indels (EDSI)*, Algorithms in Bioinformatics, Volume 3692, pp 276–290.

[22] J. Sawamura , S. Morishita , J. Ishigooka, (7 May 2014), *A symmetry model for genetic coding via a wallpaper group composed of the traditional four bases and an imaginary base E: towards category theory-like systematization of molecular/genetic biology*, Theoretical Biology and Medical Modelling, Volume 11, Issue 18.

[23] M. Schöniger, M. S. Waterman, (1992), *A local algorithm for DNA sequence alignment with inversions*, Bulletin of Mathematical Biology, Volume 54, Issue 4, pp 521-536.

[24] M. R. Servedio, Y. Brandvain, et al., (9 Dec 2014), *Not Just a Theory–The Utility of Mathematical Models in Evolutionary Biology*, PLOS Biology, Volume 12, Issue 12: e1002017

[25] V. Simossis, J. Kleinjung , J. Heringa , (Nov 2003), *An overview of multiple sequence alignment*, Current Protocols in Bioinformatics, Volume 3, Issue 1, pp 3.7.1-3.7.26.

[26] T. F. Smith, M. S. Waterman, (1981), *Identification of Common Molecular Subsequences*, Journal of Molecular Biology, Volume 147, pp 195–197.

[27] D. I. Spivak, (2014), *Category theory for the sciences*. MIT Press, Cambridge, MA, viii+486 pp.

[28] Rémy Tuyéras, (2018), *Category theory for genetics II: genotype, phenotype and haplotype*, https://arxiv.org/abs/1805.07004

[29] A. F. Vellozo , C. E. R. Alves, A. P. do Lago, (2006), *Alignment with Non-overlapping Inversions in $O(n^3)$-Time.*, In: Bücher P., Moret B.M.E. (eds) Algorithms in Bioinformatics, WABI 2006, Lecture Notes in Computer Science, Volume 4175, Springer, Berlin, Heidelberg.

M.I.T., Department of Mathematics, 77 Massachusetts avenue, Cambridge, MA 02139; currently in the M.I.T. Computer Science and Artificial Intelligence Laboratory (CSAIL).
Email: rtuyeras@mit.edu

This article may be accessed at http://www.tac.mta.ca/tac/

THEORY AND APPLICATIONS OF CATEGORIES will disseminate articles that significantly advance the study of categorical algebra or methods, or that make significant new contributions to mathematical science using categorical methods. The scope of the journal includes: all areas of pure category theory, including higher dimensional categories; applications of category theory to algebra, geometry and topology and other areas of mathematics; applications of category theory to computer science, physics and other mathematical sciences; contributions to scientific knowledge that make use of categorical methods.
Articles appearing in the journal have been carefully and critically refereed under the responsibility of members of the Editorial Board. Only papers judged to be both significant and excellent are accepted for publication.

SUBSCRIPTION INFORMATION Individual subscribers receive abstracts of articles by e-mail as they are published. To subscribe, send e-mail to `tac@mta.ca` including a full name and postal address. Full text of the journal is freely available at `http://www.tac.mta.ca/tac/`.

INFORMATION FOR AUTHORS LaTeX2e is required. Articles may be submitted in PDF by email directly to a Transmitting Editor following the author instructions at `http://www.tac.mta.ca/tac/authinfo.html`.