**Conception d’un programme d’alignement d’embedding par programmation dynamique.**

Jean Delhomme, Tatiana Galochkina, Jean-Christophe Gelly

Université Paris Cité

# Introduction

Les algorithmes d’alignement de séquences génomiques et protéiques sont des outils indispensables de la biologie moderne. Historiquement, ces algorithmes construisent une analogie de séquences base par base ou acide aminé par acide aminé. Cette analogie représente alors une proximité évolutive ou une similarité de structure et de fonction (1). L’émergence du *deep learning* et des *language models* affine cette approche en encodant une séquence d’acides aminés en une représentation vectorielle (*embedding*) capturant les propriétés structurales et fonctionnelles de la protéine (**Figure 1**) (2).

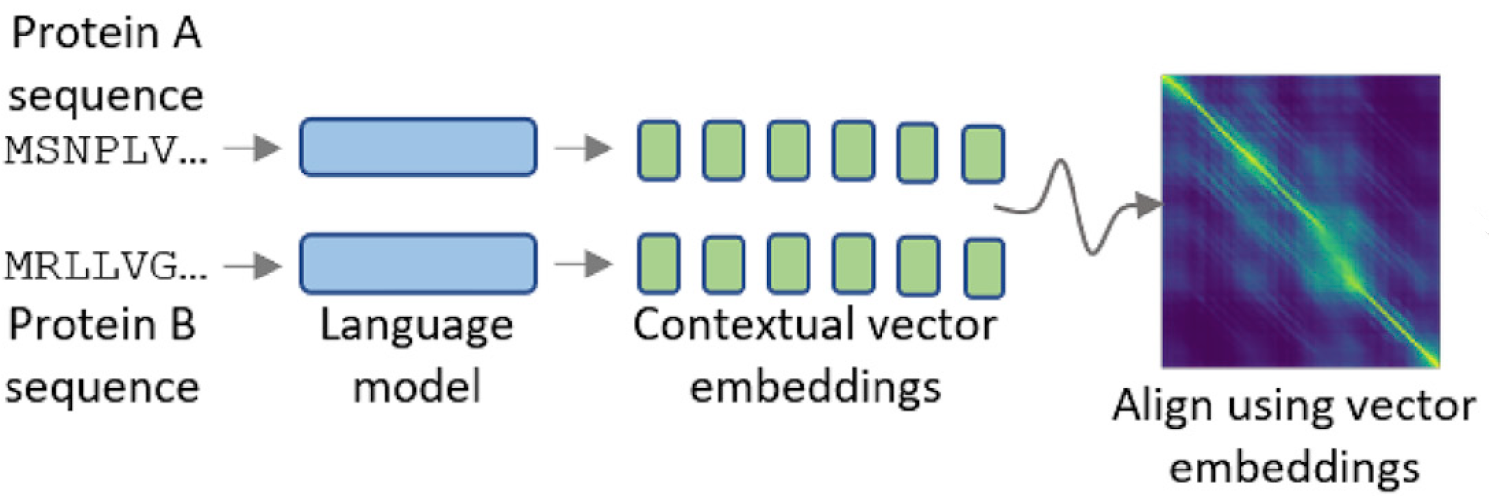


Figure 1 : Utilisation des embeddings pour un alignement de séquences. (2) Les embeddings sont générés à l’aide d’un *language model* appliqué à chaque séquence protéique. Le produit scalaire entre les vecteurs de chaque résidu de la protéine A est ensuite calculé contre les vecteurs de chaque résidu de la protéine B. La matrice obtenue est ensuite utilisée pour l’alignement et sert de matrice de similarité.

L’objectif de ce travail est de concevoir un programme dynamique permettant d’aligner des séquences représentées par un *embedding*. Le programme doit reprendre les algorithmes de Needleman et Wunsch (3), Smith et Waterman (4) et semi-global. Il sera capable de générer la matrice de similarité en calculant les produits scalaires des vecteurs d’*embeddings* des deux protéines. Il génèrera des pénalités fixes et affines de *gap*, pourra traiter des vecteurs d’*embeddings* de taille arbitraire et donnera en sortie l’alignement représenté sous forme de séquence.

# Matériel et méthodes

Le programme est exécutable depuis un terminal linux. Il utilise python 3.10.4 et numpy 1.23.1.

Le programme a recourt à des fichiers .fasta et .t5emb. Les fichiers d’embeddings ont été générés par la méthode T5 ProtTrans (5).

Nous avons utilisé l’algorithme de Needleman et Wunsch (3) pour notre alignement global et l’algorithme de Smith et Waterman pour l’alignement local (4). L’alignement semi-global a été réalisé à partir d’un algorithme de Needleman et Wunsch modifié. La **Figure 2** compare les différentes approches. Les trois algorithmes se basent sur de la programmation dynamique et ont recours à une matrice de similarité. C’est la matrice calculée à partir des produits scalaires des embeddings qui est utilisée ici.

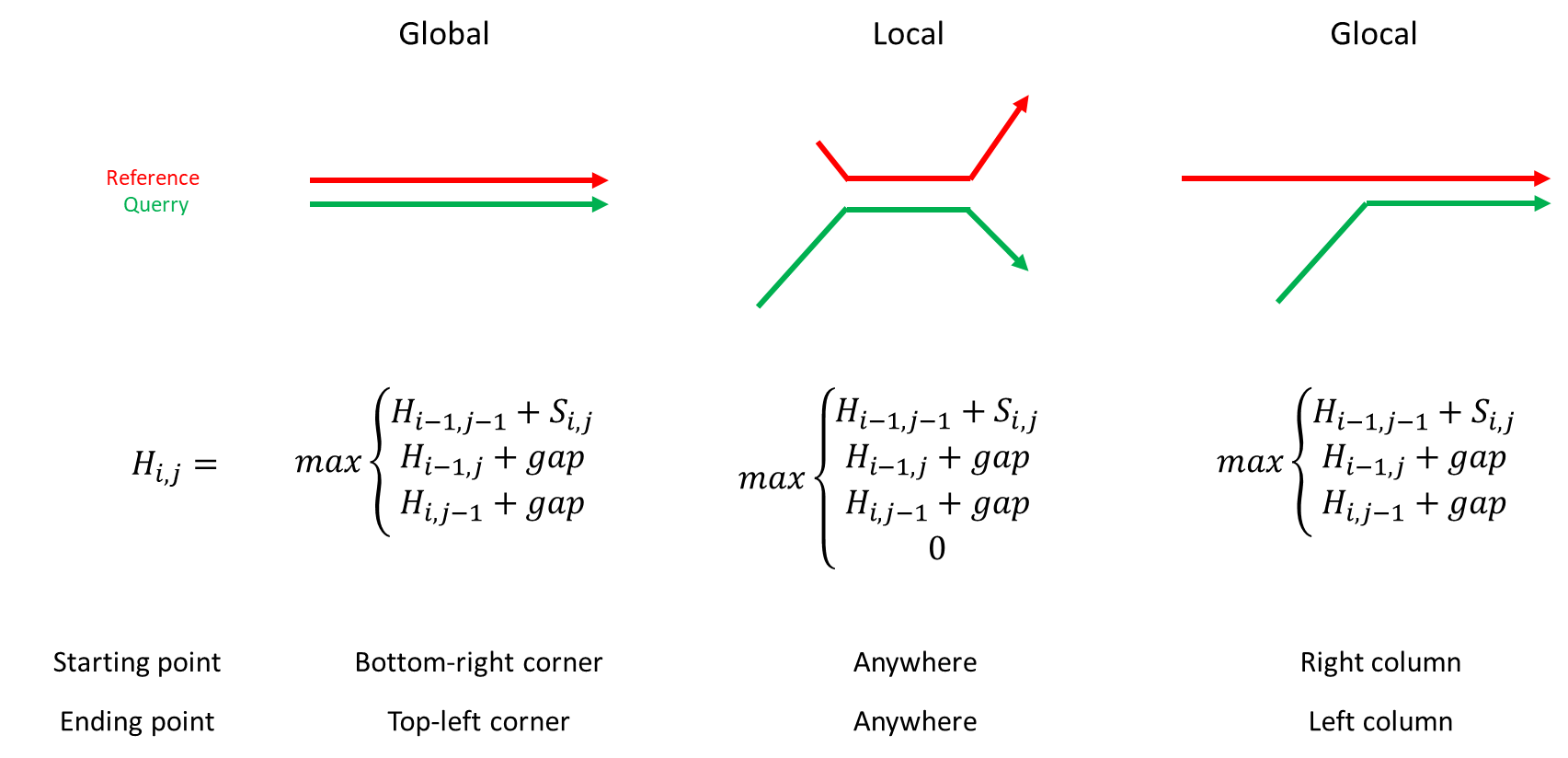


Figure 2 : Différences entre les algorithmes d’alignement global, local et semi-global. L’alignement global se fait sur l’ensemble des deux séquences. L’alignement local ne se fait que sur une région à forte homologie. L’alignement semi-global combine les deux approches, une séquence est considérée dans son intégralité tandis que l’autre non. H est la matrice de score et S la matrice de similarité. i et j correspondent aux indexes des lignes et des colonnes de ces matrices. Starting et ending point correspondent à la position des cellules utilisées pour débuter et finir le tracé de la séquence optimale.

Afin d’estimer la capacité qu’à l’algorithme de retrouver une ressemblance structurale entre deux protéines, le fichier [www.dsimb.inserm.fr/~gelly/data/TMSCORES\_HOMSTRAD.txt](http://www.dsimb.inserm.fr/~gelly/data/TMSCORES_HOMSTRAD.txt) est utilisé comme référence. Ce fichier présente ne colonne 1 l’identifiant de la première protéine, en colonne 2, l’identifiant de la seconde protéine. Les colonnes 7 et 8 correspondent aux TMscores mesurés. Un TMscore compris entre 0.5 et 1 indique la présence d’une ressemblance structurale entre les protéines.

# Résultats

**Cas d’alignement optimal**

L’algorithme est d’abord testé dans un cas optimal d’alignement. Ce test permet de s’assurer de la cohérence des résultats obtenus. Nous effectuons ce test sur la protéine 6PF2K\_1bif et sur les trois méthodes d’alignement. Les résultats obtenus sont identiques entre les trois méthodes. La **Figure 3** présente l’alignement obtenu. L’algorithme affiche également le score d’alignement de la séquence.

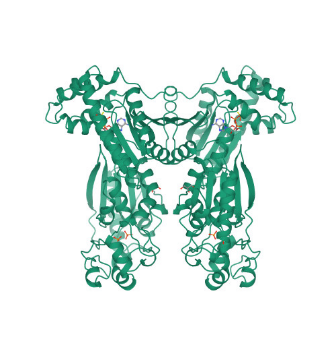


Figure 3 : Cas d’alignement optimal. La protéine 6PF2K\_1bif est alignée sur elle-même en suivant les algorithmes d’alignement global, local et semi-global. Les résultats des trois alignements sont identiques.

Le résultat de ce test est satisfaisant. Les deux protéines sont parfaitement alignées ce qui correspond à un TMscore de 1. Notons également que le score d’alignement obtenu de 6595.005 correspond au score obtenu dans le fichier de référence TMSCORES\_HOMSTRAD.txt ce qui nous conforte dans le bon fonctionnement de l’algorithme.

**Cas d’alignement mauvais**

Le second alignement est lancé sur deux protéines dont le TMscore est faible. Au contraire du cas d’alignement optimal, ce test permet d’observer le comportement de l’algorithme dans le cas de protéines peu homologues. L’alignement est effectué sur les protéines 6PF2K\_1bif(6) et 7kD\_DNA\_binding\_1azpa(7). Leurs différences structurales sont montrées **Figure 4.A.** Les résultats obtenus sont identiques entre les trois méthodes. La **Figure 4.B** présente l’alignement obtenu. L’algorithme affiche également le score d’alignement de la séquence.



**A.**

**B.**

Figure 4 : Cas d’alignement mauvais. (A) Structure des deux protéines. En haut 1bif, en bas 1azpa. (B)Les protéines 6PF2K\_1bif et 7kD\_DNA\_binding\_1azpa sont alignés en suivant les algorithmes d’alignement global, local et semi-global. Les résultats des trois alignements sont identiques.

Là aussi, le résultat du test est satisfaisant. Les deux protéines sont alignées avec un nombre de gap très important et le score d’alignement est bien plus faible que précédemment. Il est tout à fait envisageable d’avoir un TMscore inférieur à 0.5 ici, comme c’est le cas dans le fichier TMSCORES\_HOMSTRAD.txt qui propose un TMsocre de 0.16150. Notons également que le score d’alignement obtenu de 229.09 correspond au score obtenu dans le fichier de référence TMSCORES\_HOMSTRAD.txt ce qui nous conforte dans le bon fonctionnement de l’algorithme.

**Variations sur les méthodes d’alignements**

Jusque-là, les différentes méthodes ont proposé des alignements rigoureusement identiques. Effectuer des alignements sur d’autres protéines pourrait faire intervenir des différences. L’alignement des protéines 6PF2K\_1bif et adk\_2ak3a(8) en propose quelques-unes. La taille du rapport ne permet pas d’afficher l’ensemble des séquences. La **Figure 5** reprend les éléments différentiels des alignements.

Les différences observées interviennent surtout dans le fait que les programmes d’alignement local et semi-global autorisent de commencer l’alignement sur des cellules différentes. Ainsi, ces deux méthodes proposent ici deux alignements. Ces alignements ne diffèrent que sur le dernier résidu. De plus, les alignements sont identiques entre les deux méthodes. Ce test nous indique cependant que le programme semble bien fonctionner dans la mesure où les *outputs* varient en fonction des méthodes.

Aussi, les scores d’alignement sont identiques selon les méthodes (2326.07) et correspondent au score du fichier TMSCORES\_HOMSTRAD.txt. Ce score d’alignement, situé entre les deux précédent correspond effectivement à un TMscore médiant, à 0.618.



Figure 5 : Variation sur les méthodes d’alignements. Les protéines 6PF2K\_1bif et 7kD\_DNA\_binding\_1azpa sont alignés en suivant les algorithmes d’alignement global, local et semi-global. Les résultats des trois alignements sont identiques.

# Discussion

Le recours à des *embeddings* pour l’alignement de séquences s’inscrit au cœur des avancées effectuées en *deep learning* et en *language model* et promet de grandes avancées en qualité d’alignement de séquences et en prédiction de structure. Les résultats obtenus au cours de ce travail sont encourageant concernant l’utilisation des embeddings. En effet, les scores d’alignements et les TMscore évoluent dans le même sens et sont cohérent vis-à-vis des similarités de structure observées entre les différentes protéines étudiées.

Cependant, bien qu’il ait été possible de s’assurer du bon fonctionnement de l’algorithme, les résultats obtenus entre les différentes méthodes d’alignement demeurent très similaires. La présence d’alignements alternatifs pour les algorithmes global et semi-global rassurent sur la bonne implémentation de ces méthodes mais plus de tests sont nécessaires. Utiliser le programme sur d’autres protéines jusqu’à obtenir des résultats différents permettrait de s’assurer du bon fonctionnement de l’ensemble des méthodes implémentées. La prise en compte des pénalités de *gap* affines pourrait également entrainer une plus grande variabilité entre les méthodes et assurerait donc le bon fonctionnement de l’algorithme. L’implémentation de ces pénalités de *gap* affines permettraient d’ailleurs un alignement bien plus robuste et manque à cet algorithme.

# Bibliographie

1. S. Altschul et al. (2017) Handbook of Discrete and Combinatorial Mathematics. 2nd edition, Chapter 20.1 Sequence
2. T. Bepler et al. (2021) Learning the protein language: Evolution, structure, and function.
3. Needleman, Saul B. & Wunsch, Christian D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins.
4. Smith, Temple F. & Waterman, Michael S. (1981) Identification of Common Molecular Subsequences.
5. [A Elnaggar](https://scholar.google.fr/citations?user=fadCsRsAAAAJ&hl=fr&oi=sra) et al. (2020) ProtTrans: Towards Cracking the Language of Life's Code Through Self-Supervised Deep Learning and High Performance Computing
6. Page pdb de la 6-phosphofructo-2-kinase/fructose-2,6-bisphosphatase bifunctional enzyme complexed with atp-g-s and phosphate : <https://www.rcsb.org/structure/1bif>
7. Page pdb de la hyperthermophile chromosomal protein sac7d bound with kinked dna duplex : <https://www.rcsb.org/structure/1azp>
8. Page pdb de 2AK3 : <https://www.rcsb.org/structure/2ak3>

ANNEXES

(4 pages max)

Structure des programmes réalisés

Exemple d’utilisation

Difficultés rencontrées

**NOTICE INFORMATIVE**

Le but des projets est de réaliser des programmes en utilisant des méthodes classiques de bio-informatique.

Nous souhaitons également un petit rapport sur ce projet 4 pages max figures incluses pour le rapport scientifique + 4 pages maximum pour les annexes relatives à la programmation proprement dite).

Vous pourrez détailler en annexe les difficultés que vous avez eu. Pour les projets les plus « scientifiques », le rapport comportera une petite introduction sur le but du travail, le cadre théorique, le matériel et méthodes, les résultats et un petite discussion (similaire à un article scientifique). En annexe vous présenterez la structure des programmes réalisés. Vous présenterez également un exemple d’utilisation de chaque programme (ex : prog1.py fichier1 ).

Ce rapport vise à détailler votre démarche. Un schéma ou plusieurs schémas sont vivement conseillés afin de clarifier la mise en place de votre projet et des programmes.

Pour vous donnez une idée des éléments de notation de la programmation, voici une liste, non exhaustive, des différents critères :

- Toutes les questions du projet ont été traités

- Les programmes fonctionnent et produisent le résultat attendu

- Facilité d’utilisation des programmes

- Les programmes fonctionnent avec des arguments (ex nom de fichier, paramètres...) (modulé suggéré argparse)

- Il y a une documentation associé au programme, fonction, classes ainsi que des commentaires dans le code (docstring, PEP8)

- Une fonction "help" a été implémentée pour chacun des programmes

- Concision du code

- Clarté du code

- Efficacité du code (rapidité d'exécution)

- Création et utilisation de modules

- Programmation orienté objet

Le rapport ainsi que les programmes devront "compiler" sous environnement Linux standard et devront contenir des fichiers exemples ainsi que des scénarios d'utilisations facile à mettre en place.

Une présentation individuelle présentant votre démarche et vos résultats aura lieu.

Vous préparez l’ordre de passage. La présentation sera de 7 minutes + 7 minutes de questions.

**Sujet :** **Conception d’un programme d’alignement d’*embedding* par programmation dynamique.**

OBJECTIF : réaliser un programme permettant d’aligner des séquences représentées par un embedding.

A notre disposition, on a trois ressources :

* Les données des embeddings
* Les données des séquences FASTA
* La ressemblance entre toutes les protéines (pour avoir une idée des scores que l’on devrait obtenir)

Le programme doit aligner des séquences représentées par des embeddings en reprenant les algorithmes de :

* Needleman et Wunsch
* Smith et Waterman
* Semi-global (Glocal ou Gloloc) (global sur une séquence et global sur l’autre)

Le programme :

* Calcul un score de match par le dot product (ou la correlation) entre les vecteurs d’embedding d’une position de la protéine 1 et une position de la protéine 2.
* Génère des pénalités fixes et affine des brêches (gap)
* Donne en sortie l’alignement représenté sous forme de séquence
* Permet de gérer des embedings de taille arbitraire (pas uniquement 1024)