Rapport EI : Jeux évolutionnaires

Modélisation de la trajectoire 3D des brins d'ADN

Arthur Hakimi, Florian Song, Farès Gati, Lucas Bello-Kern

CentraleSupélec

Janvier 2026

Sommaire

Approches d'optimisation

Pourquoi utiliser des algorithmes génétiques ?

- **Recherche exhaustive (Brute force) :** Optimale mais impossible pour les grands espaces (complexité temporelle).
- **Algorithmes déterministes :** Heuristiques sur solutions partielles, mais restent coûteux.
- **Approche stochastique (Algorithmes Génétiques) :**
  - Inspiré de la théorie de l'évolution.
  - Mécanisme : Sélection $\rightarrow$ Reproduction $\rightarrow$ Mutation.
  - **Avantage :** Complexité dépend du nombre d'individus et de générations, pas de la taille des données.

Le Problème : Modélisation ADN 3D

Modèle existant (1993) :

- Associe chaque dinucléotide à une matrice de transformation (Rotation/Translation).
- Permet de tracer la trajectoire du brin dans l'espace.

La limite :

- Le modèle ne gère pas le **repliement**.
- Par exemple les chromosomes bactériens et plasmides sont circulaires.

Objectif :

But de l'algorithme

Adapter les matrices de rotation via un algorithme génétique pour que le dernier nucléotide rejoigne le premier (fermeture de la boucle).

Architecture Globale

L'implémentation est modulaire grâce à l'usage de classes abstraites.

1. **Initialisation :** Population aléatoire (N individus).
2. **Évaluation :** Calcul de la *Fitness*.
3. **Sélection :** Choix des reproducteurs.
4. **Reproduction (Crossover) :** Création des enfants.
5. **Mutation :** Modification aléatoire pour explorer l'espace.

Fonction de Fitness :

$$\text{Minimiser } ||P_{final} - P_{initial}||_2$$

Évaluation

Fitness 1 : fermeture en position

$$f_1 = -\|X_{\text{dernier}}\|_2$$

- Distance entre le premier nucléotide et le dernier.
- Vérifie que la fin de la trajectoire revient vers le début.

Fitness 2 : fermeture positionnelle et orientation

$$f_2 = -\frac{1}{2}\Big(\|X_{N-1}\|_2 + \|X_1 - X_N\|_2\Big)$$

- Premier terme : fermeture globale de l'avant-dernier nucléotide (aussi le 1er) sur le premier nucléotide.
- Deuxième terme : écart entre le deuxième et le dernier nucléotide (le 2e aussi), assure continuité et orientation.

Stratégies de Sélection

Comparaison de 5 méthodes pour assurer la convergence :

- **Élitisme :** Sélection d'une partie de la population correspondant aux meilleurs individus.
- **Tournoi :** Duel aléatoire, on garde le meilleur.
- **Tournoi avec espoir :** Probabilité faible de choisir le moins bon (évite les minimums locaux).
- **Roulette :** Probabilité de sélection un individu dépend de son score. Plus son score est plus grand, plus sa probabilité est élevée.
- **Rang :** Sélection basée sur le classement et non directement sur leur score. La probabilité d'être sélectionné pour un individu est son rang sur la somme des rangs.

Remarque

Le meilleur individu d'une population est toujours sélectionné.

Reproduction

Crossover (Reproduction)

- **MeanCrossover :** Les angles de la table de rotation de l'enfant correspond à la moyenne de ceux des parents
- **FitnessWeightedMeanCrossover :**Ce procédé est similaire à MeanCrossover, mais la moyenne est pondérée par l'évaluation de fitness des parents. Les valeurs des angles de la table de rotation de l'enfant sont ainsi plus proches de ceux du parent ayant la meilleure évaluation.
- **ChooseBetweenParentsCrossover :**Pour chaque valeur de l'angle de la table de rotation de l'enfant, on sélectionne soit la valeur du premier parent, soit celle du second. Cette méthode s'inspire de la reproduction humaine, où chaque gène est hérité indifféremment de la mère ou du père.

Mutation

Mutation (Gaussienne)

- **GaussianAdditiveMutation (G+) :** Ajoute une valeur gaussienne à tous les coefficients de l'individu. Cette mutation reste généralement proche des valeurs initiales.
- **GaussianAdditiveDeltaMutation (G+Δ) :** Ajoute à chaque coefficient une valeur gaussienne pondérée par Δ, l'écart-type associé à ce coefficient.
- **GaussianMultiplicativeMutation (G*) :** Multiplie chaque coefficient par exp(gaussian), permettant de prendre en compte l'échelle des valeurs. Cette approche s'est révélée moins efficace dans le cadre étudié.
- **GaussianAdditiveDeltaLog10FitnessAnnealedMutation (G+ΔF) :** Ajoute une perturbation proportionnelle à la valeur gaussienne, à l'écart-type du coefficient et à $\log_{10}(1 - \text{fitness})$, favorisant une mutation adaptative dépendante de la qualité de l'individu.

Meta-Mutations

- **ThresholdMutation (T) :** Métamutation appliquant une mutation donnée avec une probabilité p_{mut}, permettant de réduire la fréquence des modifications.
- **SimulatedAnnealingMutation (SA) :** Métamutation adaptant un paramètre `key` d'une mutation par un facteur α à chaque génération (ex. mutation gaussienne : $\sigma_t = \sigma_0 \cdot \alpha^t$), selon le principe du recuit simulé.

Tests — Sélection et Mutation

- **Sélection (`test_selection.py`)**
  - Tests généraux : taille de l'échantillon, validité des indices, sélection du meilleur individu.
  - Biais de sélection : les meilleurs individus sont favorisés (tournoi, roulette, rang).
  - Cas spécifiques :
    - Élitisme : sélection exacte des meilleurs individus.
    - Tournoi avec espoir : possibilité de sélectionner l'individu le moins performant.
- **Mutation (`test_mutation.py`)**
  - Contraintes : valeurs ≥ 2 deg non mutées
  - Conservation de la taille de la population après mutation.
  - Validation des mutations (GAM, GADM, GADLogM, GMM).
  - Recuit simulé : décroissance correcte de σ (facteur α).
  - Cas limites : probabilité nulle (aucune mutation) et totale (mutation systématique).

Tests — Crossover et RotTable

- **Crossover** (`test_crossover.py`)
  - Tests généraux : nombre d'enfants après croisement, gestion des populations vides.
  - Intégrité des individus : clés et longueurs des listes des tables de rotation conservées.
  - Tests par méthode :
    - Croisement par moyenne : valeurs de l'enfant entre celles des parents.
    - Croisement par pondération : influence accrue du parent avec la meilleure fitness.
    - Croisement par choix parmi les parents : valeurs issues exclusivement des parents.
- **RotTable** (`test_RotTable.py`)
  - Adaptation du test d'origine : chargement JSON $\rightarrow$ dictionnaire $\rightarrow$ `RotTable`.

Remarque

Couverture de code $\geq 92\%$ via des tests unitaires.

Contrainte de symétrie due à la complémentarité

Rappel

Les deux chaînes de l'ADN sont considérées comme complémentaires car chaque nucléotide ne peut être couplé qu'à un autre nucléotide précis (A avec T et C avec G).

Contrainte

Pour un dinucléotide et son complémentaire, les angles sont les mêmes pour Twist et Wedge mais opposés pour Direction.

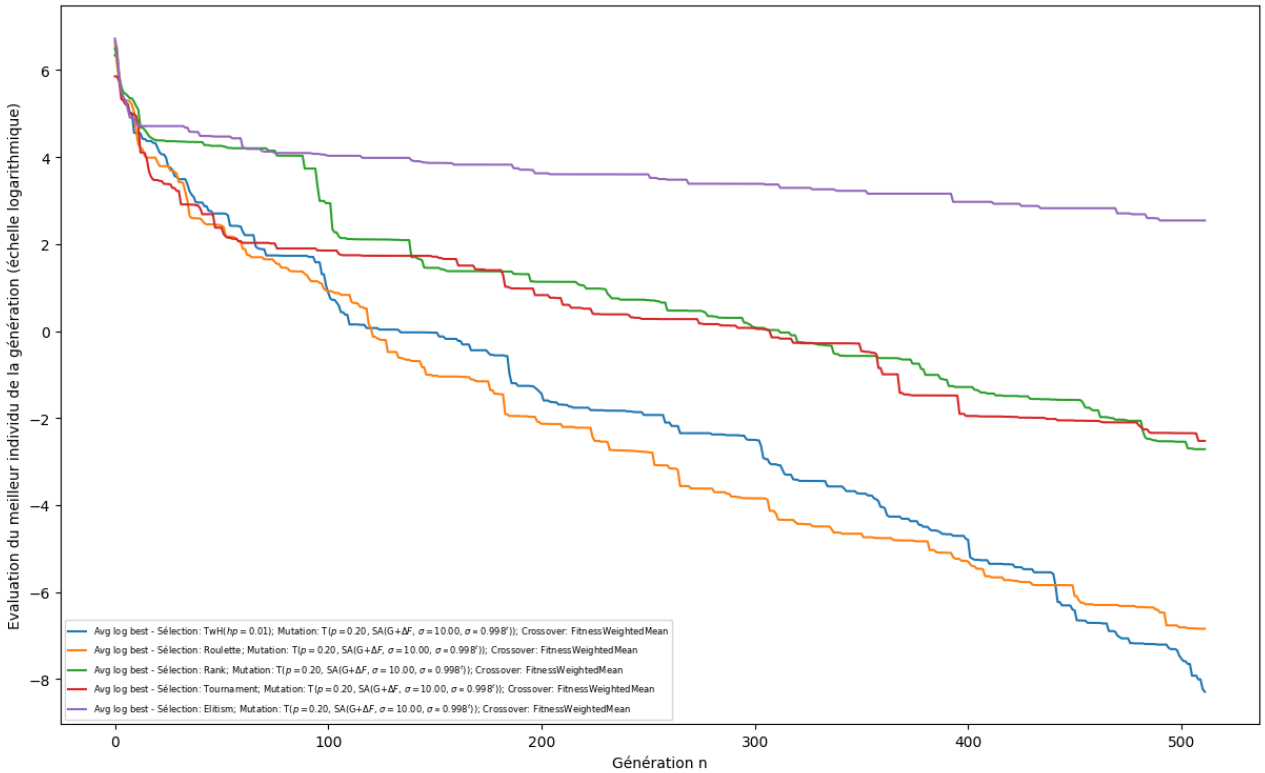
Modif de Traj3D.py et RotTable.py

Supression d'une partie dinucléotide et création d'un dictionnaire dinucleotidesComplementaryMap

Optimisation et Configuration Retenue

1. **Optimisation Algorithmique (`Traj3D.py`)**
   - Constat : 5 multiplications matricielles nécessaires par dinucléotide.
   - **Solution :** Pré-calcul du produit matriciel pour chaque dinucléotide.
   - Résultat : Réduction à **1 seule multiplication** par étape $\rightarrow$ Gain de temps critique pour les grands plasmides.
2. **Optimisation Paramètre**
   - Ajout des paramètres keepRate, duplicateRate et saltRate pour pouvoir plus facilement changer ces paramètres généraux.

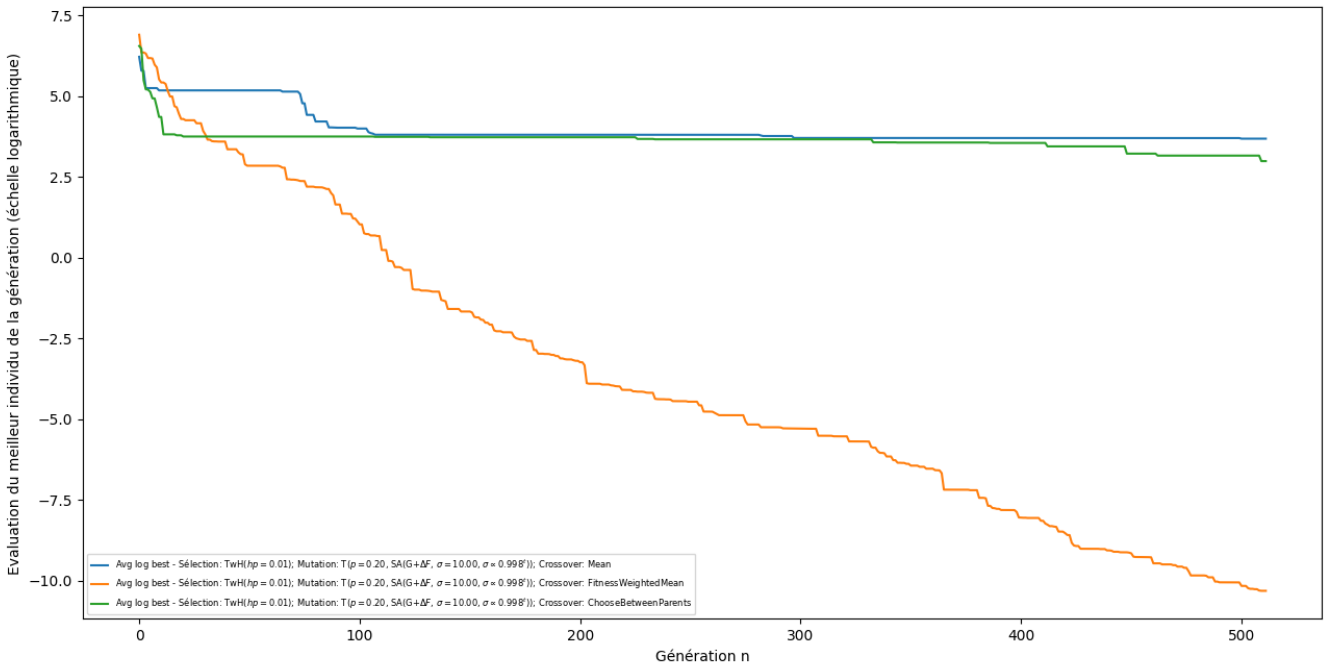
Benchmark Sélection



Gagnant : Tournoi avec espoir ($p = 0.01$)

Contrairement à l'élitisme qui sature vite, cette méthode conserve l'aléatoire nécessaire pour échapper aux minimums locaux.

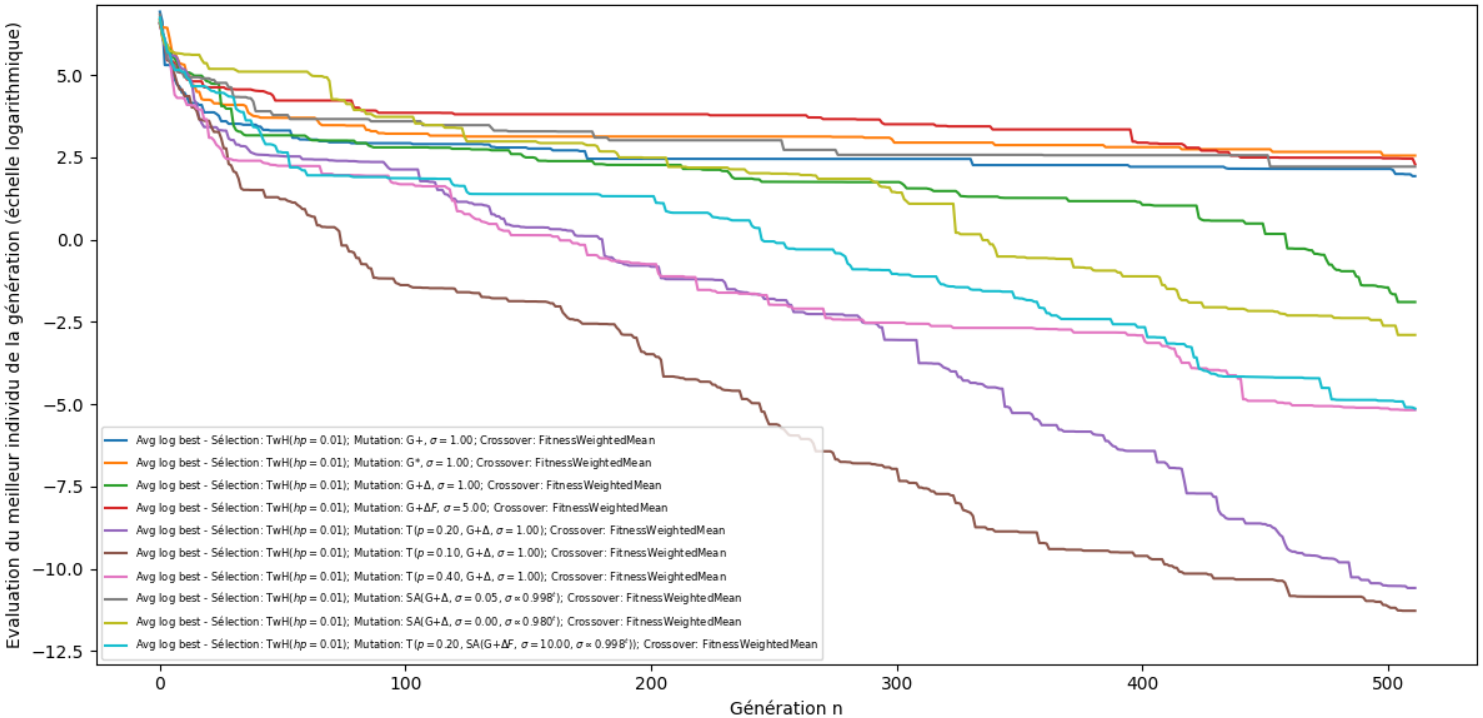
Benchmark Crossover



Gagnant : Moyenne pondérée (FitnessWeightedMean)

Le meilleur compromis : exploite les bonnes solutions tout en gardant de la diversité génétique (évite le clonage pur).

Benchmark Mutation



Configuration Optimale : Recuit Simulé

GaussianAdditiveDelta + SA : Permet une forte exploration au début (grand σ) et une convergence très précise à la fin (petit σ).

Performances Finales

Simulation finale avec la configuration optimale :

- **Paramètres :** 2048 générations, 64 individus.
- **Méthode :** Recuit simulé ($\sigma_t = \sigma_0 \cdot \alpha^t$).

Résultats de fermeture de boucle

Type	Taille (pb)	Erreur (Repliement)	Temps
Plasmide (petit)	8 000	$\approx 10^{-12}$A	10 min
Plasmide (grand)	180 000	$\approx 10^{-11}$A	2 h

L'erreur est négligeable devant la taille atomique, validant une fermeture parfaite de la boucle.

Conclusion

- **Modularité :** Une architecture flexible permettant de tester rapidement différentes stratégies évolutionnaires.
- **Robustesse :** Code validé par une couverture de tests élevée.
- **Efficacité :** Résultat très bons dans avec l'ancienne fitness (10e-12). Des résultats bons sur la nouvelle (Somme des distances = 3)

Merci de votre attention.

Avez-vous des questions ?