RÉPUBLIQUE DÉMOCRATIQUE DU CONGO

MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET UNIVERSITAIRE

UNIVERSITÉ RÉVÉREND KIM

SITE DE LINGWALA

FACULTÉ DE SCIENCES INFORMATIQUES

L3 LMD INFO/ GÉNIE LOGICIEL

BP 171. KIN XX

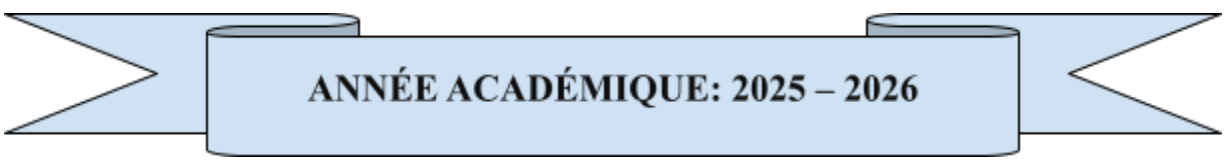
Fait par l'étudiant :

NIYA BWENGE DANIEL

Dirigé par :

Professeur : ILUNGA MBUYAMBA Elisée

Assistant : KADIMA Donatien



Le présent rapport fournit une analyse approfondie des Algorithmes Évolutionnaires en Intelligence Artificielle (IA), couvrant leur historique, leurs auteurs séminaux, leurs avantages structurels, leurs limitations opérationnelles, et leur impact stratégique dans les domaines de la robotique et de la méta-optimisation.

1. Introduction : Définition et Fondements du Calcul Évolutionnaire (CE)

Les Algorithmes Évolutionnaires (AE) constituent une classe fondamentale de recherche en Intelligence Artificielle (IA), s'inscrivant dans le champ plus vaste du Calcul Évolutionnaire (CE).[1] Ces méthodes représentent une forme de "Darwinisme Artificiel" [2], car elles s'inspirent directement des mécanismes biologiques de l'évolution naturelle et de la génétique pour résoudre des problèmes complexes d'optimisation.[3]

Le principe opératoire des AE est basé sur un processus itératif de sélection naturelle appliquée à une population de solutions candidates.[3] Contrairement aux techniques d'optimisation traditionnelles qui tentent de maintenir une seule meilleure solution, les AE travaillent avec une population (échantillonnage aléatoire).[3] Les membres les moins adaptés de cette population sont éliminées, tandis que les membres les plus performants (ceux qui présentent une meilleure aptitude, ou *fitness*) survivent, se reproduisent et continuent d'évoluer au fil des générations jusqu'à ce qu'une solution optimisée soit déterminée.[3]

1.1. Le Squelette Générique et les Opérateurs de Variation

Le cycle de vie d'un Algorithme Évolutionnaire typique inclut l'initialisation de la population, l'évaluation de la performance des individus, la sélection des individus destinés à la reproduction, l'application des opérateurs de variation (croisement et mutation), et le remplacement de l'ancienne population par la nouvelle.

Les concepts biologiques cruciaux traduits en opérations informatiques sont la reproduction, la mutation et la sélection.[3]

- **Croisement (*Crossover*) :** C'est un mécanisme d'**exploitation** qui combine les caractéristiques de deux solutions parentales jugées performantes pour créer de nouvelles solutions (*offspring*). Le croisement est essentiel, car il permet d'explorer rapidement de nouvelles régions de l'espace de recherche en recombinant des "bons blocs" génétiques trouvés chez différents parents.[6]
- **Mutation :** Cet opérateur agit comme un mécanisme d'**exploration** pur. Il introduit de petites modifications aléatoires dans le matériel génétique de la solution. La mutation est vitale pour maintenir la diversité au sein de la population et empêcher l'algorithme de converger prématurément vers des solutions sous-optimales (optima locaux).[6]

1.2. Conditions Préalables et Rôle de la Fonction d'Aptitude

Pour qu'un Algorithme Évolutionnaire puisse être appliqué efficacement, deux conditions préalables doivent être remplies [3] :

1. Les solutions candidates doivent être codées (représentées numériquement) pour le problème.
2. Une fonction d'aptitude (*fitness function*) doit être définie, capable de renvoyer une valeur mesurant la qualité de la solution (par exemple, une valeur comprise entre 1 et 100).[3]

Il est fondamental de reconnaître que l'efficacité de la convergence d'un AE est intimement liée à la conception de cette fonction d'aptitude. L'algorithme est en effet une méthode d'ordre zéro [8], ce qui signifie qu'il n'utilise aucune information sur le gradient ou la pente mathématique du paysage d'optimisation. Il est "aveugle" aux propriétés analytiques du problème et ne progresse qu'en comparant les scores de fitness des individus. Par conséquent, si la fonction d'aptitude n'est pas conçue avec précision pour refléter la qualité intrinsèque de la solution et sa capacité à discriminer entre des solutions légèrement meilleures, l'algorithme, même avec des mécanismes génétiques parfaits, évoluera vers des états sous-optimaux. Le design de la fonction de fitness est donc la contrainte de modélisation la plus critique des AE.

2. Cadre Historique et Auteurs Séminales

L'histoire des algorithmes évolutionnaires n'est pas linéaire, mais plutôt marquée par l'émergence parallèle de plusieurs écoles de pensée dans les années 1960, chacune se concentrant sur un aspect différent de l'optimisation stochastique.

2.1. Les Premières Racines et l'Émergence des Courants Fondateurs

Les premiers travaux sur l'utilisation de méthodes stochastiques pour l'optimisation remontent à 1952, [9], suivis par les premières simulations du processus d'évolution appliquées à des problèmes généraux d'optimisation par Barricelli en 1954.[9]

Cependant, les fondations théoriques et pratiques des AE reposent sur trois courants principaux qui ont vu le jour indépendamment dans les années 1960 [2] :

1. **Stratégies d'Évolution (SE) :** Ce courant a été initié en Allemagne par Ingo Rechenberg et Hans-Paul Schwefel à l'Université Technique de Berlin.[11] Dès 1965, Rechenberg a conçu le premier algorithme utilisant des stratégies d'évolution pour résoudre des problèmes d'optimisation numérique, notamment en aérodynamique.[9] Une contribution majeure de ce courant, développée par Schwefel (publiée en 1981), est le concept d'auto-adaptation, où les paramètres de la recherche, tels que les variances des distributions gaussiennes utilisées pour la mutation, sont encodés et évoluent avec la population.[13]
2. **Programmation Évolutionnaire (PE) :** Proposée en Californie en 1966 par Lawrence Fogel, Alvin Owens et Michael Walsh, cette méthode s'est initialement concentrée sur l'évolution de machines à états finis et de systèmes adaptatifs.[9]
3. **Algorithmes Génétiques (AG) :** Les Algorithmes Génétiques sont le courant le plus célèbre. Les travaux fondateurs ont été menés par **John Holland** à l'Université du Michigan dès les années 1960.[10] Holland est largement reconnu comme le père des Algorithmes Génétiques.[16] Son livre séminal de 1975, *Adaptation in Natural and Artificial Systems*, a formalisé les principes de base des AG pour l'optimisation mathématique, notamment en s'inspirant des théories de Charles Darwin.[11]

2.2. Standardisation et Unification

Les années 1980 et 1990 ont marqué une période de consolidation et de diffusion massive. **David E. Goldberg**, l'un des élèves de Holland, a joué un rôle crucial dans la popularisation des AG, surtout grâce à son ouvrage influent publié en 1989.[11] En 1992, **John Koza** a formalisé la **Programmation Génétique (PG)**, qui étend les AG à l'évolution de programmes informatiques représentés sous forme d'arbres syntaxiques.[9]

L'unification progressive de ces disciplines a conduit, après 1993, à l'adoption du terme générique de **Calcul Évolutionnaire** (*Evolutionary Computation*).[9]

La raison de cette divergence initiale puis de la convergence réside dans l'adaptation des mécanismes évolutionnaires à la nature de la variable d'optimisation. Les AG, axés sur les chaînes binaires ou discrètes (chromosomes), ont mis l'accent sur le croisement pour explorer l'espace combinatoire. Inversement, les SE conçus pour l'optimisation en variables continues

(nombres réels), ont dû développer des mutations plus sophistiquées et l'auto-adaptation pour affiner la recherche dans des paysages de fitness lisses.[18] L'efficacité d'un AE est donc profondément liée au **codage des solutions** et à la **conception des opérateurs** adaptés à l'espace de recherche concerné.

Tableau 1 : Jalons Historiques et Pères Fondateurs des Algorithmes Évolutionnaires (AE)

Décennie/an	**Fondateur(s) Clé(s)**	**Famille d'AE**	**Contribution Séminale**	**Référence Notable**
1950	Barricelli	CE Général	Premières simulations stochastiques.	[9]
1965	Rechenberg, Schwefel	Stratégies d'Évolution (SE)	Optimisation numérique, introduction des stratégies d'évolution.	[9]
1966	Fogel, Owens, Walsh	Programmation Évolutionnaire (PE)	Évolution de machines à états finis.	[15]
1975	John Holland	Algorithmes Génétiques (AG)	Formalisation théorique de l'optimisation stochastique.	[16]
1989	David E. Goldberg	Algorithmes Génétiques	Popularisation massive et applications des AG.	[17]
1992	John Koza	Programmation Génétique (PG)	Formalisation de l'évolution de programmes (arbres syntaxiques).	[11]

3. Typologie des AE : Analyse des Quatre Classes Historiques

Bien que le terme "Algorithme Évolutionnaire" (AE) englobe toutes les approches inspirées de la biologie, il est essentiel de distinguer les quatre classes historiques principales en fonction de leur représentation des solutions et de la primauté de leurs opérateurs.[11]

Tableau 2 : Caractéristiques Comparées des Principales Classes d'AE

Classe d'AE	**Représentation Typique**	**Opérateur d'Exploration Clé**	**Domaine d'Optimisation Préféré**	**Concept Technique Différenciant**
Algorithmes Génétiques (AG)	Chaînes binaires ou réelles (Chromosomes)	Croisement (Exploitation)	Optimisation combinatoire et discrète [18]	Accent sur la recombinaison structurée.
Stratégies d'Évolution (SE)	Vecteurs de nombres réels et paramètres de stratégie	Mutation et Auto-adaptation (Exploration/Affinement)	Optimisation numérique continue [18]	Évolution des hyperparamètres de mutation.
Programmation Génétique (PG)	Arbres syntaxiques représentant des programmes	Croisement et Mutation d'arbres	Apprentissage de programmes, modélisation symbolique [18]	Représentation à longueur variable (sujette au *Bloat*).
Programmation Évolutionnaire (PE)	Représentation arbitraire	Mutation (souvent l'opérateur dominant)	Systèmes adaptatifs, apprentissage de la dynamique [19]	Moins d'emphase sur le croisement que les AG.

3.1. Le Compromis Exploration/Exploitation dans les AG

Les Algorithmes Génétiques sont les plus représentatifs de la famille du CE. Leur force réside dans la gestion efficace de l'équilibre entre l'exploration et l'exploitation de l'espace de recherche. Ce dilemme classique est directement incarné par le réglage des taux de croisement et de mutation.[20]

- Le croisement, en tant qu'opérateur d'exploitation, permet de recombiner des solutions partiellement réussies pour en créer de meilleures. Son taux est typiquement élevé, souvent entre 0.6 et 0.9, afin de capitaliser sur les informations génétiques déjà acquises.[7]
- La mutation, en tant qu'opérateur d'exploration, introduit des changements aléatoires nécessaires pour sonder de nouvelles régions de l'espace de recherche et pour éviter que la population ne s'uniformise trop rapidement. Le taux de mutation est maintenu à un niveau bas, généralement entre 0.001 et 0.1, pour ne pas détruire les solutions prometteuses.[7]

Trouver l'équilibre optimal entre ces deux taux est crucial pour la performance d'un AG. Un taux de croisement trop élevé peut réduire la diversité et provoquer une convergence prématurée vers un optimum local. Inversement, un taux de mutation excessif transformerait l'algorithme en une recherche aléatoire inefficace, incapable de conserver les bonnes solutions découvertes.[20] L'ajustement de cet équilibre est fréquemment réalisé par des techniques d'analyse de sensibilité ou de réglage de paramètres.[20]

3.2. La Spécificité des Stratégies d'Évolution (SE)

Les Stratégies d'Évolution se distinguent par leur excellence dans l'optimisation numérique continue. Leur avancée technique majeure est l'**auto-adaptation**. Dans les SE, les paramètres qui régissent la recherche (tels que la taille du pas de mutation) ne sont pas fixés par l'utilisateur, mais sont eux-mêmes encodés dans le génome de l'individu et évoluent par sélection naturelle.[13] Cette capacité à ajuster dynamiquement la force et la direction de la mutation permet aux SE d'être exceptionnellement efficaces dans des espaces de recherche en haute dimension, car ils peuvent modifier leur stratégie en temps réel en fonction du paysage de fitness rencontré.[13]

4. Avantages et Puissance des Algorithmes Évolutionnaires (Robustesse et Recherche Globale)

Les Algorithmes Évolutionnaires offrent des avantages fondamentaux qui les rendent irremplaçables pour certains types de problèmes d'optimisation, en particulier ceux qui violent les hypothèses des méthodes analytiques classiques.

4.1. Recherche Globale Efficace et Optimisation Multimodale

L'un des principaux atouts des AE est leur **capacité de recherche globale**.[8] Grâce à l'utilisation d'une population de solutions explorant l'espace de manière simultanée et stochastique, les AE réduisent significativement le risque de rester bloqué dans un optimum local (solutions médiocres).[21] Cette exploration parallèle est un mécanisme puissant pour garantir la découverte de l'optimum global, même dans des paysages de fitness complexes et multimodaux.

De plus, cette approche populationnelle est naturellement adaptée à l'**optimisation multi-objective**.[11] Dans ces problèmes, où plusieurs critères souvent contradictoires doivent être optimisés simultanément, il n'existe pas une seule solution optimale, mais un ensemble de solutions de compromis appelé front de Pareto. Des variantes comme l'Algorithme Génétique Multi-Objectif (MOGA) exploitent la population pour cartographier efficacement ce front, une capacité très recherchée en conception d'ingénierie.[22]

4.2. Robustesse des Méthodes d'Ordre Zéro

La robustesse et la souplesse des AE proviennent de leur nature de méthodes d'**ordre zéro**.[8] Cela signifie qu'elles ne requièrent aucune information analytique ou différentielle sur la fonction d'aptitude. Elles opèrent uniquement sur l'évaluation de la performance (*fitness*) de chaque candidat.

Cette indépendance vis-à-vis des dérivées mathématiques confère aux AE une flexibilité inégalée :

- Elles peuvent être appliquées à des fonctions objectives non dérivables, discontinues ou bruitées.
- Elles permettent de travailler sur des espaces de recherche complexes et non standards, tels que l'optimisation topologique de formes.[8]

Cette tolérance structurelle aux fonctions de coût complexes est la principale raison de leur robustesse. L'AE n'a besoin que d'un score relatif ("la solution A est meilleure que la solution B") pour avancer. Il compense le manque d'information directionnelle précise (le gradient) par le volume massif des calculs stochastiques effectués sur la population et sur un grand nombre de générations.[23] Cette résilience est indispensable pour l'optimisation de systèmes complexes où la modélisation analytique ou la différenciation de la fonction objectif sont

impossibles, comme c'est le cas dans de nombreux problèmes de robotique ou d'ingénierie physique.[22]

5. Limitations Critiques et Défis Opérationnels (Inconvénients)

Malgré leur robustesse structurelle, les Algorithmes Évolutionnaires présentent des inconvénients pratiques majeurs qui limitent leur application universelle, notamment dans le contexte des systèmes nécessitant des performances en temps réel.

5.1. Coût Computationnel Élevé

L'inconvénient le plus significatif des AE est leur coût computationnel élevé.[10] Puisque l'approche repose sur l'évolution d'une population sur de nombreuses générations, l'étape la plus coûteuse en temps CPU est l'**évaluation de la fonction de fitness (F)**.[23]

Pour des populations de quelques dizaines d'individus et un nombre de générations de quelques centaines, cela représente déjà des dizaines de milliers de calculs de la fonction F.[23] Si l'évaluation de cette fonction est intrinsèquement lente (par exemple, nécessitant une simulation par éléments finis, l'entraînement d'un réseau de neurones, ou une simulation physique coûteuse), l'AE devient rapidement impraticable ou non compétitif par rapport aux méthodes déterministes plus rapides.

Bien que la convergence asymptotique des AE vers l'optimum global soit garantie sous certaines conditions (lorsque le temps tend vers l'infini, $t \rightarrow \infty$) [25], la lenteur de la convergence en pratique est un obstacle majeur à leur utilisation dans les scénarios exigeant une grande rapidité. La flexibilité des AE est donc directement payée par un prix élevé en calcul.

5.2. Dépendance au Paramétrage et Réglage des Hyperparamètres

Les AE sont très sensibles au réglage des hyperparamètres, tels que la taille de la population, la méthode de sélection, les taux de croisement et de mutation.[7] L'identification des valeurs optimales pour ces paramètres est essentielle pour assurer une exploration efficace et une convergence satisfaisante.

Un réglage inadéquat peut entraîner une stagnation prématurée (faible mutation/croisement) ou une instabilité excessive (forte mutation/croisement).[20] En l'absence de théorie universelle pour déterminer ces valeurs *a priori*, l'utilisateur doit souvent recourir à des expérimentations intensives et à des analyses de sensibilité pour trouver le jeu de paramètres le plus performant pour un problème donné.[20] Paradoxalement, les Algorithmes Génétiques sont eux-mêmes parfois utilisés pour optimiser les hyperparamètres d'autres algorithmes d'apprentissage automatique (méta-optimisation), soulignant que même le processus

d'optimisation évolutionnaire nécessite un réglage minutieux.[26]

5.3. Le Phénomène du *Bloat* dans la Programmation Génétique (PG)

Un défi spécifique aux familles d'AE utilisant une représentation de solution à longueur variable, comme la Programmation Génétique (PG), est le phénomène de *bloat*.[25]

Le *bloat* (ou gonflement) se caractérise par la croissance incontrôlée de la taille des individus (des arbres syntaxiques représentant les programmes) au fil des générations, sans amélioration correspondante de la fonction d'aptitude.[28] Cette croissance est souvent attribuée à l'accumulation d'**introns**, c'est-à-dire de code redondant ou inactif qui n'affecte pas l'exécution du programme.[29]

Ce phénomène est pénalisant à plusieurs égards [25] :

1. **Coût des Ressources :** Les individus gonflés consomment plus de mémoire et de temps CPU pour l'évaluation et la manipulation des opérateurs génétiques.
2. **Ralentissement de la Recherche :** La croissance excessive du code pénalise l'évolution et la capacité de la PG à se déployer à grande échelle.
3. **Interprétabilité :** Les solutions finales deviennent inutilement complexes et difficiles à comprendre par les humains.

Une des théories expliquant le *bloat* est que les introns offrent une protection contre l'effet potentiellement destructeur du croisement, conférant ainsi un léger avantage sélectif aux programmes plus longs.[29]

6. Impact Majeur sur l'Intelligence Artificielle

L'impact des Algorithmes Évolutionnaires sur l'IA est le plus profond là où les techniques basées sur le gradient (comme la rétro propagation dans le *Deep Learning*) ne sont pas applicables.

6.1. La Neuroévolution : Apprentissage des Poids et des Architectures

La Neuroévolution est un domaine crucial où les AE s'imposent comme une alternative majeure. Elle utilise les algorithmes évolutionnaires pour optimiser les composants des Réseaux de Neurones Artificiels (ANN), qu'il s'agisse de leurs poids, de leur topologie ou de leurs règles d'apprentissage.[24]

Dans le contexte du contrôle de systèmes autonomes, cette approche est essentielle. Par exemple, des chercheurs utilisent la neuroévolution pour contrôler des robots simulés, tels

que des quadrupèdes.[24] Le mécanisme implique d'utiliser une ANN comme contrôleur des mouvements, et d'appliquer ensuite des Algorithmes Génétiques pour faire évoluer et optimiser les poids de ce réseau de neurones.[24]

L'avantage distinctif de cette approche réside dans le fait que, dans des domaines comme la robotique ou l'Apprentissage par Renforcement (RL), la fonction de récompense (l'aptitude) est souvent non différentiable ou extrêmement bruitée. La Neuroévolution, grâce à la nature d'ordre zéro des AE, permet d'apprendre des comportements complexes et d'optimiser le contrôleur du robot dans cet environnement "hostile" où les méthodes classiques échoueraient.[11]

6.2. Robotique Évolutionnaire en Essaim et Systèmes Adaptatifs

Les AE sont vitaux pour la conception de systèmes autonomes et adaptatifs. En Robotique Évolutionnaire en Essaim (*Embodied Evolutionary Robotics* - EER), ils sont employés pour apprendre des comportements collectifs (d'essaims).[30] L'optimisation classique de ces systèmes est presque impossible en raison de la complexité et du nombre des interactions inter-robots. Les AE offrent une approche automatisée pour l'apprentissage de ces comportements, permettant aux essaims de robots d'évoluer leurs stratégies d'adaptation et de navigation en ligne.[16]

6.3. Métaheuristiques et Optimisation des Hyperparamètres

Dans le domaine moderne de l'apprentissage automatique, les AE sont de plus en plus utilisés comme des métaheuristiques puissantes. Bien qu'ils ne soient généralement pas assez rapides pour optimiser les millions de poids des grands modèles de *Deep Learning* (DL), ils sont particulièrement efficaces pour l'**Optimisation d'Hyperparamètres** (HPO).[26]

L'utilisation d'Algorithmes Génétiques pour trouver la configuration optimale des hyperparamètres (comme les taux d'apprentissage ou la structure des réseaux) dans le *Deep Reinforcement Learning* (DRL) est un exemple concret. Les AE permettent d'explorer l'espace discret et multimodal des configurations de paramètres, ce qui est une tâche ardue et chronophage si elle est effectuée manuellement.[26]

Le rôle stratégique des AE en IA est donc de plus en plus orienté vers l'**AutoML** et l'**auto-configuration**. Leur capacité à gérer les paysages de fitness non lisses et multimodaux leur confère une valeur unique dans l'exploration de l'espace des architectures et des hyperparamètres, laissant l'optimisation des poids internes aux méthodes basées sur le gradient.

7. Conclusion : Bilan et Perspectives d'Avenir

Les Algorithmes Évolutionnaires (AE), héritiers de courants de recherche initiés dans les années 1960 par des auteurs séminales comme John Holland, Ingo Rechenberg et Hans-Paul Schwefel, ont établi leur place en Intelligence Artificielle comme des outils d'optimisation stochastique fondamentaux.

Leur avantage principal réside dans leur **robustesse** en tant que méthodes d'ordre zéro et leur capacité à réaliser une **recherche globale** en échappant aux optima locaux.[8] Cette robustesse leur permet de s'attaquer à des problèmes complexes d'optimisation où les fonctions objectives sont non dérivables, discontinues ou définies par des simulations coûteuses.

Cependant, cette flexibilité s'accompagne de limitations opérationnelles notables, principalement le **coût computationnel élevé** lié à la nécessité de dizaines de milliers d'évaluations de la fonction de fitness.[23] De plus, le succès des AE dépend fortement de l'expertise humaine nécessaire au **réglage des hyperparamètres** critiques (taux de croisement et de mutation), et des défis structurels comme le *bloat* en Programmation Génétique.[28]

L'impact contemporain des AE en IA est de nature stratégique, se concentrant sur les systèmes adaptatifs et l'optimisation de haut niveau. Ils excellent dans la **Neuro-évolution** pour le contrôle robotique [24] et agissent comme des métaheuristiques efficaces pour l'Optimisation d'Hyperparamètres dans l'apprentissage profond.[26]

Les perspectives d'avenir pour les AE se concentrent sur la minimisation des coûts de calcul et l'amélioration de leur scalabilité. Le développement de techniques d'auto-adaptation (inspirées des Stratégies d'Évolution) et la recherche de solutions au *bloat* sont des domaines de recherche actifs visant à rendre ces algorithmes plus compétitifs face aux méthodes déterministes à grande échelle.

8. Bibliographie et Références

Les travaux sur les Algorithmes Évolutionnaires s'appuient sur plusieurs publications séminales qui ont défini le champ du Calcul Évolutionnaire :

- **Holland, J. H.** (*1975*). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor..[11]
- **Goldberg, D. E.** (*1989*). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley..[11]
- **Fogel, L. J., Owens, A. J. & Walsh, M. J.** (*1966*). *Artificial Intelligence through Simulated Evolution*. John Wiley..[15]
- **Rechenberg, I.** (Travaux des années 1960/1970). Contribution majeure à la fondation des Stratégies d'Évolution..[12]
- **Schwefel, H.-P.** (*1981*). *Numerical Optimization of Computer Models*. John Wiley. (Travaux des années 1970 sur l'auto-adaptation)..[11]
- **Koza, J. R.** (*1992*). *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press..[11]
- **Lutton, É.** (*2021*). Algorithmes génétiques, algorithmes évolutionnaires. *Techniques de l'Ingénieur*..[31]
- **Conférences Majeures :** La communauté scientifique se réunit notamment à la conférence GECCO (Genetic and Evolutionary Computation COnference), sponsorisée par l'ACM SIGEVO..[33]
- **Applications et Limites :**
    - Schoenauer, M. (*2002*). Sur les Algorithmes Évolutionnaires. *ESAIM: Proceedings*..[8]
    - Haddana, Z. (Thèse). Rôle de la neuro-évolution pour le contrôle de robots..[24]
    - Divers articles sur les défis du *bloat* et de la complexité.[28]

Sources des citations

1. consulté le novembre 27, 2025, https://www.ultralytics.com/fr/glossary/evolutionary-algorithms#:~:text=Les%20algorithmes%20%C3%A9volutionnaires%20(EA)%20repr%C3%A9sentent,naturelle%20et%20de%20la%20g%C3%A9n%C3%A9tique.
2. Algorithmes génétiques et algorithmes évolutionnaires - Techniques de l'Ingénieur, consulté le novembre 27, 2025, https://www.techniques-ingenieur.fr/base-documentaire/archives-th12/archives-technologies-logicielles-et-architecture-des-systemes-tiahb/archive-1/algorithmes-genetiques-et-algorithmes-evolutionnaires-s7218/
3. Algorithme évolutionnaire : définition et explication - Titaxium, consulté le novembre 27, 2025, https://www.titaxium.org/quest-ce-que-lalgorithme-evolutionnaire/

4. consulté le novembre 27, 2025, https://www.ionos.fr/digitalguide/sites-internet/developpement-web/genetic-algorithm/#:~:text=Un%20algorithme%20g%C3%A9n%C3%A9tique%20est%20une,des%20populations%20de%20solutions%20potentielles.
5. Algorithmes génétiques : utiliser la sélection naturelle pour bloquer le trafic des bots, consulté le novembre 27, 2025, https://datadome.co/fr/learning-center/algorithmes-genetiques-utiliser-la-section-naturelle-pour-bloquer-le-trafic-des-bots/
6. Un guide sur les algorithmes évolutionnaires - Ultralytics, consulté le novembre 27, 2025, https://www.ultralytics.com/fr/blog/what-is-an-evolutionary-algorithm-a-quick-guide
7. Chapter 5 - Crossover and Its Effects - Algorithm Afternoon, consulté le novembre 27, 2025, https://algorithmafternoon.com/books/genetic_algorithm/chapter05/
8. Représentations non structurées en optimisation topologique de formes par algorithmes évolutionnaires - ESAIM: Proceedings and Surveys, consulté le novembre 27, 2025, https://www.esaim-proc.org/articles/proc/pdf/2002/01/schoenauer.pdf
9. Algorithme évolutionniste - Historique - Techno-Science, consulté le novembre 27, 2025, https://www.techno-science.net/glossaire-definition/Algorithme-evolutionniste-page-3.html
10. Algorithmes évolutionnaires - Alliot, consulté le novembre 27, 2025, https://www.alliot.fr/genetic.shtml.fr
11. PAR Arnaud Zinfiou ALGORITHMES EVOLUTSONNA1RES POUR L'ORDONNANCEMENT INDUSTRIEL : APPLICATION À L'INDUSTRIE AUTOMOBILE DECEMBRE - UQAC Constellation, consulté le novembre 27, 2025, https://constellation.uqac.ca/id/eprint/314/1/030084689.pdf
12. Evolution Strategies - Clever Algorithms, consulté le novembre 27, 2025, https://cleveralgorithms.com/nature-inspired/evolution/evolution_strategies.html
13. (PDF) A history of evolutionary computation - ResearchGate, consulté le novembre 27, 2025, https://www.researchgate.net/publication/216300863_A_history_of_evolutionary_computation
14. Evolution strategy - Wikipedia, consulté le novembre 27, 2025, https://en.wikipedia.org/wiki/Evolution_strategy
15. Chapitre 29. Les algorithmes évolutionnaires | Cairn.info, consulté le novembre 27, 2025, https://stm.cairn.info/les-mondes-darwiniens-volume-2--9782919694402-page-907?lang=fr
16. Comprendre les algorithmes génétiques et leur utilisation - AI-FutureSchool, consulté le novembre 27, 2025, https://www.ai-futureschool.com/fr/informatique/introduction-aux-algorithmes-genetiques.php

17. Genetic Algorithms in Search, Optimization, and Machine Learning - David Edward Goldberg - Google Books, consulté le novembre 27, 2025, https://books.google.com/books/about/Genetic_Algorithms_in_Search_Optimizatio.html?id=2IIJAAAACAAJ
18. Evolutionary algorithms vs genetic algorithms | by Hey Amit | Data Scientist's Diary | Medium, consulté le novembre 27, 2025, https://medium.com/data-scientists-diary/evolutionary-algorithms-vs-genetic-algorithms-5f015eed4b45
19. Genetic algorithm - Wikipedia, consulté le novembre 27, 2025, https://en.wikipedia.org/wiki/Genetic_algorithm
20. Genetic Algorithms: Crossover Probability and Mutation Probability | Baeldung on Computer Science, consulté le novembre 27, 2025, https://www.baeldung.com/cs/genetic-algorithms-crossover-probability-and-mutation-probability
21. consulté le novembre 27, 2025, https://www.ultralytics.com/fr/blog/what-is-an-evolutionary-algorithm-a-quick-guide#:~:text=Avantages%20et%20inconv%C3%A9nients%20des%20algorithmes%20%C3%A9volutionnaires,-Voici%20quelques%2Duns&text=Capacit%C3%A9%20de%20recherche%20globale%20%3A%20Les,bloqu%C3%A9%20dans%20des%20solutions%20m%C3%A9diocres.
22. Automatic Optimization for Compliant Constant Force Mechanisms - MDPI, consulté le novembre 27, 2025, https://www.mdpi.com/2076-0825/12/2/61
23. Algorithmes évolutionnaires, consulté le novembre 27, 2025, https://www.enseignement.polytechnique.fr/profs/informatique/Eric.Goubault/poly/cours009.html
24. La neuro-évolution pour le contrôle de robot à pattes, consulté le novembre 27, 2025, http://archives.univ-biskra.dz/bitstream/123456789/21339/1/HADDANA_ZAKARIA.pdf
25. Un état des lieux de l'optimisation évolutionnaire et de ses implications en sciences pour l'ingénieur, consulté le novembre 27, 2025, https://www.emse.fr/~leriche/EvoMerge.pdf
26. Exploration-Driven Genetic Algorithms for Hyperparameter Optimisation in Deep Reinforcement Learning - MDPI, consulté le novembre 27, 2025, https://www.mdpi.com/2076-3417/15/4/2067
27. Définition de Hyperparamètres - IA, consulté le novembre 27, 2025, https://ovirank.com/definition-ia/hyperparametres
28. Code Bloat Problem in Genetic Programming - International Journal of Scientific and Research Publications, consulté le novembre 27, 2025, https://www.ijsrp.org/research-paper-0413/ijsrp-p1612.pdf
29. Introduction, consulté le novembre 27, 2025, http://www0.cs.ucl.ac.uk/staff/ucacbbl/bloat_wsc2/node1.html
30. Distributed Embodied Evolutionary Adaptation of Behaviors in Swarms of Robotic Agents, consulté le novembre 27, 2025, https://theses.fr/2017LORR0300
31. Exemples d'application Algorithmes génétiques, algorithmes évolutionnaires,