



# Rapport de Projet

Sous le thème :

## **Quantum K-Means Clustering : Étude Comparative entre les Approches Classique et**

**Département :** Génie informatique

**Filière :** Analytique Des Données Et Informatique Décisionnelle

**Réalisé par :**

El youzghi Mohamed.  
El ajroudi Nasr-Allah.

**Encadré par :**

Badaoui Mohame  
khriss Abdelaadim

**Année universitaire 2024-2025**



بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

## **Remerciement**

C'est avec une sincère reconnaissance que nous consacrons ces quelques lignes pour exprimer notre gratitude envers tous ceux qui ont contribué, de près ou de loin, à la réalisation de ce projet de fin de module.

Nous tenons tout d'abord à adresser nos plus sincères remerciements à M. Badaoui Mohamed, notre professeur de cours, ainsi qu'à M. Abdelladim Khriss, notre professeur de travaux pratiques. Leur engagement, leur disponibilité et leurs précieux enseignements nous ont permis d'approfondir nos connaissances et de mener à bien cette étude sur le Quantum K-Means Clustering. Leur expertise et leurs conseils ont été une source d'inspiration et un guide essentiel dans notre démarche.

Nous exprimons également notre gratitude à l'ensemble des enseignants de notre formation, dont l'accompagnement et le soutien tout au long de notre parcours académique ont été d'une grande aide. Leurs enseignements nous ont fourni les bases solides nécessaires à la compréhension et à l'application des concepts abordés dans ce projet.

Enfin, nous remercions toutes les personnes qui nous ont apporté, de près ou de loin, leur aide, leur encouragement et leur soutien tout au long de cette expérience. Leur bienveillance et leurs contributions ont joué un rôle déterminant dans l'aboutissement de ce travail.

## Résumé

Notre projet porte sur l'**implémentation et l'analyse de l'algorithme Quantum K-Means Clustering**, une approche innovante combinant l'**informatique quantique et l'apprentissage automatique** pour optimiser le partitionnement des données. L'objectif principal est de comparer l'efficacité de cette approche avec celle de l'algorithme K-Means classique, en évaluant les avantages qu'offre le calcul quantique en termes de rapidité et de précision.

Dans un premier temps, nous avons étudié les **fondements théoriques** du clustering et de l'algorithme K-Means traditionnel, en mettant en évidence ses domaines d'application ainsi que ses limites. Ensuite, nous avons exploré la nécessité d'une transition vers une approche quantique, notamment dans des contextes où le traitement de grandes quantités de données est crucial.

L'implémentation du Quantum K-Means a nécessité une **analyse approfondie des concepts de l'informatique quantique**, tels que la superposition et l'intrication, afin d'adapter l'algorithme aux circuits quantiques. Pour cela, nous avons utilisé des bibliothèques dédiées comme **Cirq et PennyLane**, qui permettent de simuler et d'exécuter des algorithmes quantiques sur des architectures spécifiques.

Enfin, nous avons effectué une **comparaison des performances** entre l'algorithme classique et son équivalent quantique, en analysant plusieurs critères comme la **rapidité d'exécution, la convergence et la précision des clusters formés**. Les résultats obtenus nous ont permis d'évaluer l'impact potentiel de l'informatique quantique dans le domaine du machine learning et d'identifier les défis à surmonter pour une adoption plus large de ces techniques.

## Abstract

Our project focuses on the **implementation and analysis of the Quantum K-Means Clustering algorithm**, an innovative approach that combines **quantum computing and machine learning** to enhance data clustering efficiency. The primary objective is to compare the effectiveness of this quantum approach with the traditional K-Means algorithm, evaluating the potential advantages of quantum computation in terms of **speed and accuracy**.

First, we conducted a **theoretical study** on clustering methods and the classical K-Means algorithm, highlighting its applications and limitations. We then explored the motivation behind transitioning to a quantum-based approach, particularly in scenarios where handling large datasets is a critical challenge.

The implementation of Quantum K-Means required an in-depth analysis of **quantum computing principles**, such as **superposition and entanglement**, to adapt the algorithm for quantum circuits. To achieve this, we leveraged specialized libraries like **Cirq and PennyLane**, which facilitate the simulation and execution of quantum algorithms on dedicated architectures.

Finally, we performed a **comparative evaluation** between the classical and quantum versions of the algorithm, focusing on key performance metrics such as **execution speed, convergence, and clustering accuracy**. Our findings provide insights into the **potential impact of quantum computing on machine learning** and highlight the challenges that must be addressed for wider adoption of these technologies.

## Liste des abbreviations:

- ✓ **AI** : *Artificial Intelligence* (Intelligence Artificielle)
- ✓ **QC** : *Quantum Computing* (Informatique Quantique)
- ✓ **ML** : *Machine Learning* (Apprentissage Automatique)
- ✓ **QML** : *Quantum Machine Learning* (Apprentissage Automatique Quantique)
- ✓ **KMC** : *K-Means Clustering* (Clustering par K-Moyennes)
- ✓ **QKMC** : *Quantum K-Means Clustering* (Clustering par K-Moyennes Quantique)
- ✓ **Qubit** : *Quantum Bit* (Bit Quantique)
- ✓ **GPU** : *Graphics Processing Unit* (Processeur Graphique)
- ✓ **CPU** : *Central Processing Unit* (Processeur Central)
- ✓ **NISQ** : *Noisy Intermediate-Scale Quantum* (Ordinateurs Quantiques de Taille Intermédiaire et Bruyants)
- ✓ **IBMQ** : *IBM Quantum Experience* (Plateforme de Calcul Quantique d'IBM)
- ✓ **QAOA** : *Quantum Approximate Optimization Algorithm* (Algorithme d'Optimisation Quantique Approximatif)
- ✓ **VQE** : *Variational Quantum Eigensolver* (Solveur Variationnel des Valeurs Propres Quantique)
- ✓ **Cirq** : *Quantum Framework by Google* (Framework Quantique de Google)
- ✓ **PennyLane** : *Quantum Machine Learning Library* (Bibliothèque d'Apprentissage Automatique Quantique)
- ✓ **Qiskit** : *Quantum Information Science Kit* (Kit Scientifique pour l'Informatique Quantique d'IBM)
- ✓ **SVM** : *Support Vector Machine* (Machine à Vecteurs de Support)
- ✓ **NN** : *Neural Networks* (Réseaux de Neurones)

# Table des matières

## Introduction

### 1. Contexte et motivation

- Importance du clustering en science des données
- Défis de l'évaluation des résultats de clustering
- Subjectivité des métriques de qualité

### 2. Problématique

- Limitations du K-Means classique
- Sensibilité à l'initialisation, forme des clusters, choix de  $K$
- Potentiel de l'informatique quantique pour surmonter ces limites

### 3. Objectifs du projet

- Comparaison K-Means classique vs quantique
- Implémentation et évaluation des performances

---

## Fondements Théoriques

### 1. Algorithme K-Means Classique

- Principe de fonctionnement (initialisation, assignation, mise à jour)
- Complexité algorithmique ( $O(n)$ )
- Limites : minima locaux, sensibilité aux outliers, choix de  $K$

### 2. Transition vers l'Approche Quantique

- Concepts clés : qubits, superposition, intrication
- Avantages potentiels : accélération des calculs, optimisation améliorée
- Justification de l'utilisation du quantique pour K-Means

---

## Analyse et Implémentation de l'Algorithme Quantique

### 1. Cadre Théorique

- Encodage des données classiques en états quantiques
- Calcul des distances euclidiennes quantiques

- **Projection Stéréographique Inverse (ISP) pour la normalisation**
- 2. Algorithmes Hybrides Proposés**
- **q1-K-Means : Calcul de distance unique (centroïde-enregistrement)**
  - **q2-K-Means : Calcul parallèle des distances pour tous les centroïdes**
  - **q3-K-Means : Calcul simultané pour tous les enregistrements et centroïdes**

**3. Défis d'Implémentation**

- **Post-sélection et nombre de *shots* requis**
- **Contraintes matérielles (bruit, décohérence)**

**Expérimentations et Résultats**

**1. Méthodologie d'Évaluation**

- **Jeux de données utilisés (UCI, synthétiques)**
- **Métriques : précision, temps d'exécution, évolutivité**
- **Comparaison avec δ-k-Means (référence classique)**

**2. Résultats Comparatifs**

- **Performances de q1/q2/q3-K-Means vs K-Means classique**
- **Analyse de l'impact de l'ISP sur la séparation des clusters**
- **Robustesse aux données haute dimension**

**3. Exemple Concret : Base Wine**

- **Visualisation des clusters avant/après enrichissement quantique**
- **Réduction du taux d'erreur (de 5/89 à 3/89)**

**Conclusion**

**1. Bilan des Performances**

- **Avantages quantiques théoriques vs limitations pratiques actuelles**
- **Cas d'usage prometteurs (données complexes, haute dimension)**

**2. Perspectives**

- **Optimisation des circuits quantiques dédiés**
- **Hybridation classique-quantique**

- Benchmarking sur matériel quantique émergent
-

---

## *Généralités*

---

Le clustering constitue une technique fondamentale dans le domaine de la science des données et de l'apprentissage automatique. En effet, il aide à révéler des structures cachées dans les données sans nécessiter d'étiquettes préexistantes. Cette capacité à déceler des groupes naturels dans les données s'avère particulièrement bénéfique lorsque l'information est non structurée ou lorsque les étiquettes manquent, sont coûteuses ou difficiles à obtenir.

Cependant, comme vous l'avez relevé, l'évaluation des résultats du clustering demeure un défi majeur. Cela est surtout vrai à cause de la subjectivité qui entoure la notion de « regroupement pertinent ». Dans le cadre de l'apprentissage supervisé, on peut évaluer les performances d'un modèle de manière relativement objective à l'aide de métriques telles que l'exactitude, la précision, le rappel ou l'aire sous la courbe ROC (AUC). Ces mesures sont établies en confrontant les prédictions du modèle aux étiquettes réelles des données.

En revanche, pour l'apprentissage non supervisé, et plus précisément en ce qui concerne le clustering, il n'existe pas de vérité de référence évidente pour mesurer la qualité des regroupements. Cela rend l'évaluation des algorithmes de clustering plus complexe et souvent sujette à interprétation.

Comme vous l'avez mentionné, plusieurs méthodes existent pour évaluer la qualité des résultats de clustering, mais chacune d'entre elles présente des limites :

1. **Utilisation de données artificielles** : Bien que cela permette de tester les algorithmes sur des distributions connues, les résultats ne peuvent pas vraiment être appliqués aux données réelles, qui sont souvent plus compliquées et moins définies.
2. **Utilisation de jeux de données étiquetés** : Cette méthode consiste à vérifier si les clusters obtenus correspondent aux classes étiquetées. Néanmoins, les classes d'un problème supervisé ne reflètent pas toujours les regroupements les plus pertinents pour une analyse non supervisée. De plus, d'autres regroupements pourraient être tout aussi valables, voire plus importants dans un contexte donné.
3. **Critères numériques** : Des mesures telles que l'inertie intra-cluster ou la séparation inter-clusters sont souvent utilisées pour évaluer la qualité des clusters. Cependant, ces critères reposent sur des hypothèses prédéfinies sur ce qui constitue un « bon » clustering, ce qui peut ne pas être pertinent dans tous les cas. Par exemple, dans certaines situations, des clusters qui se chevauchent peuvent offrir plus d'informations que des clusters bien séparés.
4. **Évaluation par un expert** : Solliciter l'avis d'un expert afin de juger de la pertinence des clusters peut fournir des insights précieux, mais cette approche est difficilement mesurable et ne permet pas de comparer objectivement différentes méthodes de clustering. De plus, l'opinion d'un expert peut être influencée par son domaine d'expertise, ce qui ne permet pas de généraliser à d'autres types de données.

Votre idée d'évaluer le clustering en le considérant comme une étape préparatoire pour une tâche supervisée est à la fois innovante et pragmatique. En effet, en mesurant l'amélioration des performances d'un algorithme supervisé grâce aux résultats du clustering, on peut objectivement quantifier l'utilité des informations extraites par le clustering.

Cette méthode permet de contourner la subjectivité liée à l'évaluation directe des clusters et se concentre sur l'impact tangible du clustering sur une tâche réelle. En combinant un apprenant non supervisé (le clustering) avec un apprenant supervisé, vous proposez une évaluation en cascade qui mesure la réduction de l'erreur de l'algorithme supervisé grâce aux informations issues du clustering.

Cela s'inscrit dans un cadre plus large de combinaison de classifieurs, où plusieurs modèles sont utilisés de manière séquentielle ou parallèle pour optimiser les performances globales. Des techniques comme le bagging, le boosting ou la généralisation empilée ont montré leur efficacité dans ce domaine, et votre méthode d'évaluation en cascade repose sur des principes similaires.

En somme, le clustering représente une étape cruciale dans la science des données et l'apprentissage automatique, car il permet d'identifier des motifs cachés dans les données. Toutefois, son évaluation pose encore des défis en raison de la subjectivité inhérente à la notion de regroupement pertinent. Votre approche d'évaluer le clustering en mesurant son impact sur une tâche supervisée propose une solution à la fois objective et pratique pour quantifier l'utilité des informations extraites par les algorithmes de clustering. Cela ouvre également des perspectives intéressantes pour la combinaison de méthodes supervisées et non supervisées, ce qui pourrait aboutir à des avancées significatives.

## i) Évaluation en cascade :

Étant donné un jeu de données initial contenant des informations de classes, les étapes principales de la méthode que nous proposons sont les suivantes :

- Apprentissage supervisé sur le jeu de données initial.
- Apprentissage :
  - Clustering sur le jeu de données en ignorant les informations de classes .
  - Enrichissement du jeu de données à partir des résultats du clustering .
  - Apprentissage supervisé sur le jeu de données enrichi.
- Comparaison des résultats des 2 classifieurs appris.

Comme nous l'avons évoqué précédemment, nous envisageons deux façons d'enrichir les jeux de données à partir des résultats du clustering. La première consiste à créer de nouveaux attributs représentant l'information capturée par le clustering, et à ajouter ces nouveaux attributs dans le jeu de données initial. La seconde est de créer des sous-jeux de données en fonction des groupes ciblés par le clustering.

Concernant les nouveaux attributs créés depuis les résultats du clustering dans le cas de la première méthode, différents types d'informations peuvent être ajoutés :

- **Appartenance aux clusters** : La plupart des algorithmes de clustering fournissent en sortie une partition de l'ensemble des objets. Nous pouvons utiliser comme nouveaux attributs l'appartenance des objets aux clusters. Cette information serait ainsi représentée par un nouvel attribut catégoriel, chaque objet étant associé à un identifiant du cluster auquel il appartient.
- **Centres des clusters** : On peut également associer à chaque objet un ensemble d'attributs représentant le centre du cluster auquel il est associé. Le nombre d'attributs du jeu de données serait ainsi doublé.
- **Poids des attributs dans les clusters** : Récemment ont émergé de nombreux algorithmes de subspace clustering capables d'associer à chaque attribut de chaque cluster un poids spécifiant l'importance de l'attribut pour déterminer l'appartenance des objets au cluster. Dans ces cas, nous pouvons donc associer à chaque objet un nouvel attribut numérique par attribut initial, correspondant au poids de cet attribut pour le cluster auquel l'objet appartient.
- **Probabilité d'appartenance ou distance aux clusters** : La probabilité d'appartenance de chaque objet à chaque cluster peut également constituer une nouvelle information à ajouter lorsque des modèles probabilistes sont utilisés. Si ce n'est pas le cas, alors la distance de chaque objet à chaque cluster peut être calculée puis ajoutée à leur description.
- **Valeurs min et max des coordonnées des clusters** : Les valeurs minimum et maximum des coordonnées des membres de chaque cluster sur chaque attribut peuvent aussi être utilisées pour enrichir les jeux de données. À chaque objet serait alors associées ces informations en fonction du cluster auquel il appartient.

De plus, comme la plupart des algorithmes de clustering nécessitent de spécifier différents paramètres en entrée, nous pouvons exécuter ces algorithmes pour différentes valeurs de paramètres, et enrichir les jeux de données pour chaque résultat de clustering. Par exemple, de nombreuses méthodes de clustering nécessitent de spécifier le nombre de clusters recherchés. Dans ce cas, nous pouvons exécuter la méthode plusieurs fois en faisant varier le nombre de clusters entre 2 et 10. L'algorithme supervisé utilisé ensuite pourrait alors choisir quel(s) attribut(s) utiliser parmi tous ceux générés.

Dans le cas de la seconde méthode de combinaison proposée, nous générerons dans un premier temps plusieurs partitions avec différents paramètres d'entrée. Puis nous calculons les erreurs en validation croisée d'apprenants

supervisés exécutés indépendamment sur chaque groupe d'objets créé par le clustering. Et finalement, la partition ayant mené au taux d'erreur le plus faible est conservée.

La figure 1 résume les étapes principales de cette méthodologie :

- a. Diviser le jeu de données en deux parts égales : un ensemble d'apprentissage  $Ap^-$  et un ensemble de test  $Te^-$
- b. Exécuter plusieurs fois le clustering, avec différents paramètres d'entrée et en ignorant les informations de classes, et produire les ensembles enrichis d'apprentissage  $Ap^+$  et de test  $Te^+$ .
- c. Lancer l'apprentissage supervisé sur l'ensemble d'apprentissage  $Ap^-$  et produire le classifieur  $C^-$ .
- d. Lancer l'apprentissage supervisé sur l'ensemble d'apprentissage enrichi  $Ap^+$  et produire le classifieur  $C^+$ .
- e. Lancer la classification sur l'ensemble test  $Te^-$  à partir du classifieur  $C^-$ .
- f. Lancer la classification sur l'ensemble test  $Te^+$  à partir du classifieur  $C^+$ .
- g. Comparer les erreurs de classification.

Pour évaluer l'amélioration des résultats avec ou sans les nouvelles informations fournies par un algorithme de clustering évalué, nous testons les deux méthodes (l'apprentissage supervisé sur les données initiales et l'apprentissage supervisé sur les données enrichies) sur différents jeux de données indépendants. Sur chaque jeu de données, nous faisons cinq validations croisées avec découpage du jeu de données en deux, comme proposé dans (Dietterich, 1998). Pour chaque validation croisée, nous calculons les taux d'erreur pondérés des deux méthodes. Puis nous utilisons quatre mesures pour comparer les résultats :

- **nb vict** : le nombre de victoires de chaque méthode.
- **vict sign** : le nombre de victoires significatives, en utilisant le  $5 \times 2cv$  F-test (Alpaydin, 1999) pour vérifier si les résultats sont significativement différents.
- **wilcoxon** : le wilcoxon signed rank test, qui indique si une méthode est significativement meilleure qu'une autre sur un ensemble de problèmes indépendants.
- **moyenne** : l'erreur pondérée moyenne.

Pour effectuer ces comparaisons, C4.5 (Quinlan, 1993) comme apprenant supervisé est bien adapté car, comme il fait de la sélection d'attributs pendant l'apprentissage et fournit en sortie un arbre de décision où l'on peut observer quels attributs ont été utilisés et à quels niveaux ils ont été utilisés, il met clairement en avant si les nouvelles informations issues du clustering sont utiles ou non. De plus, il a également l'avantage d'être rapide et d'être capable de considérer des attributs catégoriels aussi bien que numériques. Enfin, afin de vérifier si les résultats sont dépendants de l'algorithme supervisé utilisé, nous mènerons également des expérimentations avec deux autres algorithmes supervisés : C5 boosté (Quinlan, 2004) et DLG (Webb & Agar, 1992).

## ii) Expérimentations :

Les expérimentations conduites dans cette section abordent deux problèmes complémentaires :

- Comment un algorithme supervisé peut-il bénéficier des informations issues de méthodes de clustering ?

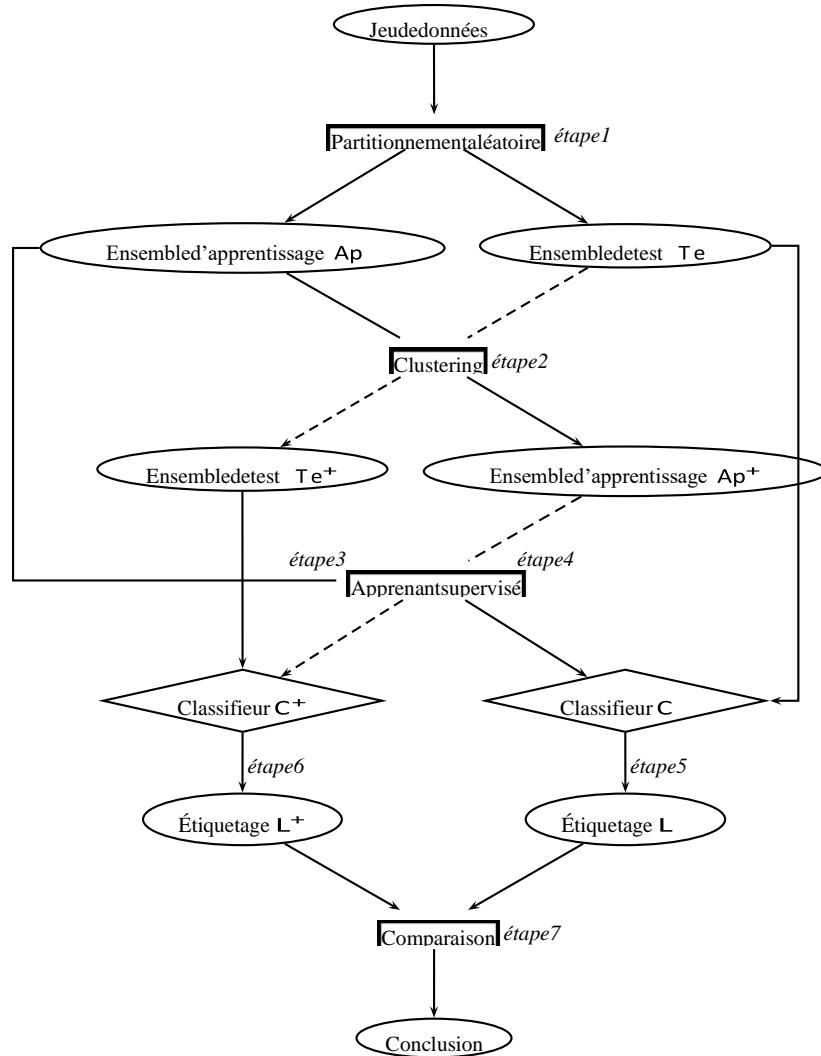


FIG. 1– Méthodologie d'évaluation de l'intérêt d'ajouter à un jeu de données de l'in formation provenant d'un algorithme de clustering.

- Quelle méthode de clustering fournit les informations les plus utiles aux algorithmes supervisés ?

Nous présentons d'abord une application où nous montrons l'intérêt d'utiliser le clustering en prétraitement d'un apprentissage supervisé. Puis nous présentons les résultats des comparaisons de plusieurs algorithmes de clustering :

- **Aléa:** un algorithme qui génère des partitions aléatoires, prenant en entrée le nombre de clusters recherchés, et servant de référence.
- **K-means:** l'algorithme très connu basé sur l'évolution de K centres mobiles représentant les K clusters à

trouver.

- **LAC** (Domeniconi et al., 2004) : une adaptation de K-means qui associe à chaque centre de cluster un poids sur chaque dimension, inversement proportionnel à la dispersion des objets du cluster sur la dimension.
- **EM1** : basé sur l'utilisation de modèles probabilistes et de l'algorithme EM sous l'hypothèse que les données ont été générées par des distributions gaussiennes indépendantes sur chaque dimension (Ye & Spetsakis, 2003).
- **EM2** : une adaptation de EM1 qui effectue de la sélection d'attributs pendant l'apprentissage, en ne considérant pour chaque cluster qu'un sous-ensemble des dimensions sur lesquelles la déviation standard est minimisée.

Les algorithmes comparés ici utilisent donc des modèles ayant des niveaux de complexité différents. En effet, K-means utilise uniquement un centroïde pour représenter un cluster. LAC ajoute à chaque centroïde un vecteur de poids sur chaque dimension de l'espace de description. EM1 définit une probabilité d'appartenance de chaque objet à chaque cluster et utilise un modèle gaussien. Et EM2 considère également un sous-ensemble des dimensions pertinentes associées à chaque cluster.

Tous ces algorithmes nécessitent de spécifier le nombre K de clusters recherchés. Comme nous en avons discuté précédemment, nous exécuterons donc plusieurs fois ces méthodes en faisant varier K entre 2 et 10.

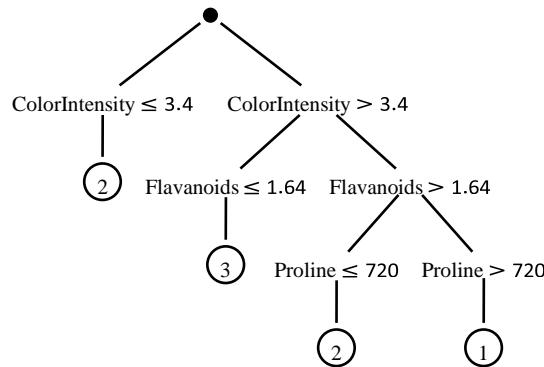
Parmi les cinq ensembles d'attributs qui peuvent être ajoutés aux jeux de données initiaux par la première méthode de combinaison proposée, seuls les trois premiers sont utilisés dans ces expérimentations, parce que le quatrième ensemble dégrade les résultats et que le cinquième ne semble pas ajouter d'information supplémentaire. Or en n'utilisant pas ces deux ensembles d'attributs, nous réduisons la taille des jeux de données enrichis, ce qui permet de fournir aux apprenants supervisés des données moins complexes.

Enfin, les jeux de données utilisés sont ceux issus de l'UCI Machine Learning Repository (Blake & Merz, 1998) qui ne contiennent que des données numériques.

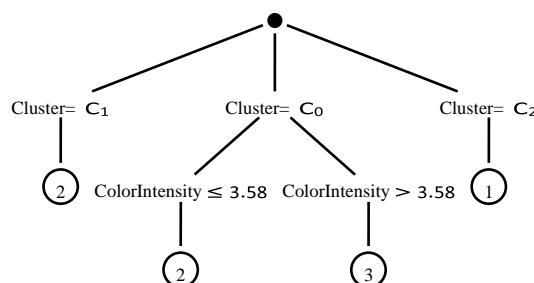
### iii. Exemple sur la base wine :

Le jeu de données wine issue de l'UCI Machine Learning Repository contient la description de 178 vins définis par 13 attributs. 3 types de vins, qui représentent les classes, sont présents.

Lorsque C4.5 est appliqué sur ce jeu de données, l'arbre de décision obtenu est celui présenté figure 2. On observe alors que les attributs sélectionnés par C4.5 sont ColorIntensity, Flavanoids et Proline.

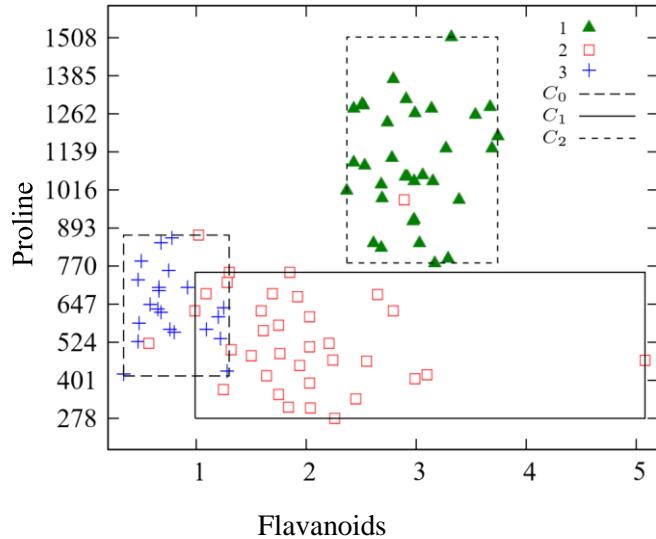


Lorsque l'on utilise EM2 pour ajouter de l'information au jeu de données tel que décrit dans la section 2, nous obtenons l'arbre de décision présenté figure 3. On observe alors que C4.5 a utilisé l'un des attributs créés par notre méthode : celui qui se réfère à l'appartenance des objets aux clusters lorsque le nombre de clusters est positionné à 3. L'autre attribut utilisé est ColorIntensity, également utilisé par C4.5 sur le jeu de données initial.



Si l'on reprend la méthode de visualisation graphique des règles associées aux clusters présentée dans (Candillier et al., 2005), on s'aperçoit que la meilleure projection 2D sélectionnée concerne les attributs Flavanoids et Proline, les deux autres attributs utilisés dans l'arbre de décision appris par C4.5 initialement. La figure 4 montre cette projection.

Finalement, nous observons donc sur cet exemple une corrélation forte entre l'arbre de décision appris par C4.5 sur le jeu de données wine initial, et l'arbre de décision appris par C4.5 sur le jeu de données wine enrichi par les résultats du clustering appris par EM2. De plus, nous pouvons remarquer que l'attribut ajouté par notre méthode est utilisé par C4.5 dès le début de la construction de l'arbre de décision. Nous sommes donc là en présence d'un exemple où C4.5 a utilisé l'information provenant des résultats du clustering à un haut niveau dans l'arbre de décision qu'il construit.



Meilleure projection 2D de EM2 sur wine.

clustering à un haut niveau dans l'arbre de décision qu'il construit. Il semble donc qu'il spécialise ici son traitement selon les différentes zones créées par EM2 dans l'espace de description des objets. Une autre interprétation possible est que notre nouvel attribut créé aide C4.5 à définir des zones de décision plus complexes. En effet, utiliser le nouvel attribut ajouté par EM2 correspond en fait à effectuer un test sur plusieurs attributs simultanément, alors que C4.5 seul ne peut utiliser à un nœud de l'arbre qu'un attribut à la fois.

Ensuite, lorsque l'on calcule l'erreur de classification des deux arbres sur les données de test, l'arbre initial a un taux d'erreur de 5/89 alors que l'arbre construit avec l'aide de EM2 a une erreur de seulement 3/89. Les erreurs de l'arbre initial correspondent à 1 erreur où un élément de la classe 1 a été associé à la classe 2, et 4 erreurs où des éléments de la classe 2 ont été associés à la classe 3. D'autre part, les erreurs associées au second arbre correspondent à 1 erreur où un élément de la classe 1 a été associé à la classe 2, et 2 erreurs où des éléments de la classe 2 ont été associés à la classe 3. La méthode a donc aidé à différencier les classes 2 et 3. Ceci semble donc confirmer notre intuition que les nouveaux attributs ajoutent au jeu de données des informations de plus haut niveau.

## iv. Comparaison d'algorithmes de clustering :

La table 1 présente les taux d'erreurs pondérés de C4.5 sur les jeux de données initiaux de l'UCI, puis sur les jeux de données enrichis par les algorithmes de clustering correspondants. Chaque mesure correspond à une moyenne sur cinq validations croisées avec découpage du jeu de données en deux. À chaque instant, toutes les méthodes sont exécutées sur le même ensemble d'apprentissage et évaluées sur le même ensemble de test.

	C4.5 seul	C4.5 + Aléa	C4.5 + K-means	C4.5 + LAC	C4.5 + EM1	C4.5 + EM2
ecoli	48.5	48.3	42.8	<b>40.3</b>	42	43.1
glass	<b>32.6</b>	40.8	35.7	37	40.4	34.9
image	4.8	6	4.8	<b>4.6</b>	<b>4.6</b>	<b>4.6</b>
iono	14.1	15.8	14.2	13.1	<b>9.8</b>	11.2
iris	7.3	7.9	6.7	<b>3.7</b>	5.1	4.7
pima	31	35	32.1	32.1	30.8	<b>30</b>
sonar	31	35.2	30	28.8	28.8	<b>27.2</b>
vowel	29.5	38.5	25	26.4	24.1	<b>22.2</b>
wdbc	5.9	6.8	4.6	3.9	5.1	<b>3.1</b>
wine	8.7	8.8	10.4	9.6	<b>2.7</b>	3.6

Taux d'erreur pondéré (en %) de C4.5 seul et C4.5 enrichi par les algorithmes

de clustering. Les valeurs en gras correspondent au taux d'erreur minimum obtenu sur chaque jeu de données.

On peut ainsi observer que la plupart du temps, les résultats de C4.5 sont améliorés lorsque des informations sont ajoutées à partir des résultats d'algorithmes de clustering *réels* alors qu'ils se dégradent lorsque le clustering aléatoire est utilisé. De plus, les résultats obtenus à l'aide de EM1 et de EM2 sont souvent meilleurs que les résultats obtenus avec K-means et LAC.

Nous utilisons ensuite le *5×2cv F-test* (Alpaydin, 1999) pour évaluer si les résultats de deux algorithmes supervisés (l'un sur le jeu de données initial et l'autre sur le jeu de données enrichi) sont significativement différents. La table 2 reporte ces tests entre C4.5 seul et C4.5 enrichi par les algorithmes de clustering correspondants. Elle utilise la p-value associée au *5×2cv-F test*. Mais la mesure reportée est 1 moins la p-value. Ainsi, une valeur de 0.01 pour EM2 sur le jeu de données wine signifie que la probabilité que C4.5 aidé par EM2 surpassé C4.5 seul *par hasard* est de seulement 1%.

Enfin, la table 3 présente un résumé des comparaisons entre C4.5 seul et C4.5 enrichi par les algorithmes de clustering correspondants. Rappelons les mesures utilisées :

- *nb vict* est le nombre de victoires de chaque méthode
- *vict sign* est le nombre de victoires significatives : le nombre de fois où l'inégalité ( $1 - \text{pvalue} \leq 0.05$ ) est vérifiée
- *wilcoxon* est le *wilcoxon signed rank test* : s'il est supérieur à 1.96, alors cela signifie que la méthode est significativement meilleure que C4.5 seul
- et *moyenne* est l'erreur pondérée moyenne (en %)

EM2 est le seul algorithme qui améliore significativement les résultats de C4.5 sur l'ensemble des jeux de données, selon le test de wilcoxon. Il est significativement meilleur sur 3 jeux de données selon le *5×2cv-F test*. Mais comme EM2, EM1 améliore les résultats de C4.5 neuf fois sur dix, contrairement à K-means et LAC. Tous les algorithmes améliorent les résultats de C4.5 en moyenne, sauf le clustering aléatoire. Et lorsque C4.5 est combiné avec des algorithmes de clustering basés sur des modèles plus complexes, alors les taux d'erreur sont plus faibles et les améliorations sont plus significatives que lorsqu'il est combiné avec des algorithmes de clustering utilisant des modèles moins complexes.

	C4.5 seul	C4.5 + Aléa	C4.5 + K-means	C4.5 + LAC	C4.5 + EM1	C4.5 + EM2
nb vict	-	1/9	5/4	7/3	<b>9/1</b>	<b>9/1</b>
vict sign	-	0/1	0/0	1/0	2/0	<b>3/0</b>
wilcoxon	-	-2.67	-0.05	1.31	1.83	<b>2.56</b>
moyenne	21.3	24.3	20.6	20	19.3	<b>18.5</b>

Comparaison de C4.5 seul avec C4.5 enrichi par les algorithmes de clustering.

Des expérimentations ont également été menées en utilisant deux autres algorithmes supervisés : C5 boosté 10 fois, pour observer si les informations ajoutées par le clustering aident aussi les méthodes qui combinent déjà plusieurs classifiants, et DLG, une méthode qui utilise les *moindres généralisés*, et non des arbres de décision.

Les tables 4 et 5 présentent les taux d'erreur pondérés calculés dans ces deux cas. Il est alors très intéressant de remarquer que, malgré l'utilisation de différents algorithmes supervisés, les méthodes qui minimisent le taux d'erreur en validation croisée sur les différents jeux de données restent les mêmes. En particulier, EM1 et EM2 surpassent encore K-means et LAC dans la grande majorité des cas.

	C5 seul	C5 + Aléa	C5 + K-means	C5 + LAC	C5 + EM1	C5 + EM2
ecoli	44.9	45.2	42.6	<b>39.9</b>	43.5	43.1
glass	<b>33.9</b>	43.8	35.7	37.8	35.6	34.4
image	3.3	3.9	3.2	<b>3</b>	<b>3</b>	<b>3</b>
iono	8.3	9.2	8.5	9.2	<b>6.6</b>	7.8
iris	5.6	6.3	5.3	<b>4.1</b>	4.8	4.4
pima	31.1	32.1	31.2	30.5	30.8	<b>29.7</b>
sonar	25.8	28.2	25.7	24.3	21.6	<b>21.3</b>
vowel	16	21	14.9	15	15.3	<b>14.5</b>
wdbc	4.6	4.8	3.8	4.2	4.1	<b>3.6</b>
wine	7	7.7	6.8	7.2	<b>2.9</b>	3.7

Taux d'erreur pondéré (en %) de C5 boosté seul et C5 boosté enrichi par les algorithmes de clustering.

Les tables 6 et 7 résument les comparaisons d'algorithmes de clustering selon l'algorithme supervisé utilisé. Il est de nouveau intéressant de noter que l'ordre dans lequel les méthodes de clustering sont rangées reste le même quelle que soit la méthode supervisée utilisée. Par contre, d'autres méthodes que EM2 deviennent significativement meilleures que l'algorithme supervisé correspondant seul. C'est par exemple le cas de EM1 lorsque C5 est utilisé.

Les tests ont également été menés en utilisant la seconde méthode de combinaison d'algorithmes supervisés et non supervisés, c'est-à-dire en exécutant C4.5 sur chaque groupe de données identifié par les algorithmes de clustering. La table 8 reporte les taux d'erreur obtenus dans ce cas, et la table 9 résume les résultats.

Nous observons donc encore dans ce cas que, malgré l'utilisation d'une autre méthode de combinaison d'algorithmes supervisés et non supervisés, l'ordre dans lequel les méthodes de clustering sont rangées reste le même.

	DLG seul	DLG + Aléa	DLG + K-means	DLG + LAC	DLG + EM1	DLG + EM2
ecoli	60.8	69.7	53.3	<b>51.5</b>	54.3	54.8
glass	47.1	60.1	49.6	47.5	47.2	<b>46.5</b>
image	9.8	14.2	9.7	9.1	9.1	<b>8.9</b>
iono	22.5	32	18.9	21.2	<b>12.9</b>	15.4
iris	9.1	15.6	8	<b>5.1</b>	6.4	6.5
pima	40.6	43.7	39.1	39.9	38.6	<b>38.4</b>
sonar	45.5	45.9	43.5	42.9	40.8	<b>38</b>
vowel	50.5	64.5	44.5	45	44	<b>42.7</b>
wdbc	9.2	10.7	7.7	7.8	7.1	<b>6.9</b>
wine	18.5	22.1	18.1	16.3	<b>6.9</b>	8

Taux d'erreur pondéré (en %) de DLG seul et DLG enrichi par les algorithmes de clustering.

	C5 seul	C5 + Aléa	C5 + K-means	C5 + LAC	C5 + EM1	C5 + EM2
nb vict	-	0/10	7/3	7/3	<b>9/1</b>	<b>9/1</b>
vict sign	-	0/0	0/0	0/0	0/0	0/0
wilcoxon	-	-2.88	1.41	1.31	<b>2.04</b>	<b>2.67</b>
moyenne	18	20.2	17.8	17.5	16.8	<b>16.5</b>

Comparaison de C5 boosté seul avec C5 boosté enrichi par les algorithmes de clustering.

	DLG seul	DLG + Aléa	DLG + K-means	DLG + LAC	DLG + EM1	DLG + EM2
nb vict	-	0/10	9/1	9/1	9/1	<b>10/0</b>
vict sign	-	0/2	1/0	<b>2/0</b>	<b>2/0</b>	<b>2/0</b>
wilcoxon	-	-2.88	<b>2.15</b>	<b>2.77</b>	<b>2.77</b>	<b>2.88</b>
moyenne	31.4	37.9	29.2	28.6	26.7	<b>26.6</b>

Comparaison de DLG seul avec DLG enrichi par les algorithmes de clustering.

	C4.5 seul	C4.5 + Aléa	C4.5 + K-means	C4.5 + LAC	C4.5 + EM1	C4.5 + EM2
ecoli	48.5	51.1	39.3	<b>32.1</b>	39.3	43.4
glass	<b>32.6</b>	46.4	33.9	32.8	33.1	33.1
image	4.8	6.8	4.8	4.8	4.3	<b>4</b>
iono	14.1	15.9	13.6	11.2	<b>10.1</b>	11.1
iris	7.3	7.6	5.5	<b>3.1</b>	4.9	4
pima	31	31.4	30.3	32.2	30.2	<b>28.9</b>
sonar	31	32.9	26.5	24.5	25.4	<b>22.3</b>
vowel	29.5	41.8	26.1	26.3	26	<b>25</b>
wdbc	5.9	6.5	3.5	4	3.5	<b>2.7</b>
wine	8.7	7.5	10.4	10	<b>2.2</b>	3

Taux d'erreur pondéré (en %) de C4.5 seul et C4.5 enrichi par les algorithmes de clustering selon la seconde méthode de combinaison.

	C4.5 seul	C4.5 + Aléa	C4.5 + K-means	C4.5 + LAC	C4.5 + EM1	C4.5 + EM2
nb vict	-	1/9	7/2	6/3	<b>9/1</b>	<b>9/1</b>
vict sign	-	0/3	0/0	0/0	1/0	<b>3/0</b>
wilcoxon	-	-2.46	1.1	1.2	<b>2.72</b>	<b>2.77</b>
moyenne	21.3	24.8	19.4	18.1	17.9	<b>17.8</b>

Comparaison de C4.5 seul avec C4.5 enrichi par les algorithmes de clustering selon la seconde méthode de combinaison.

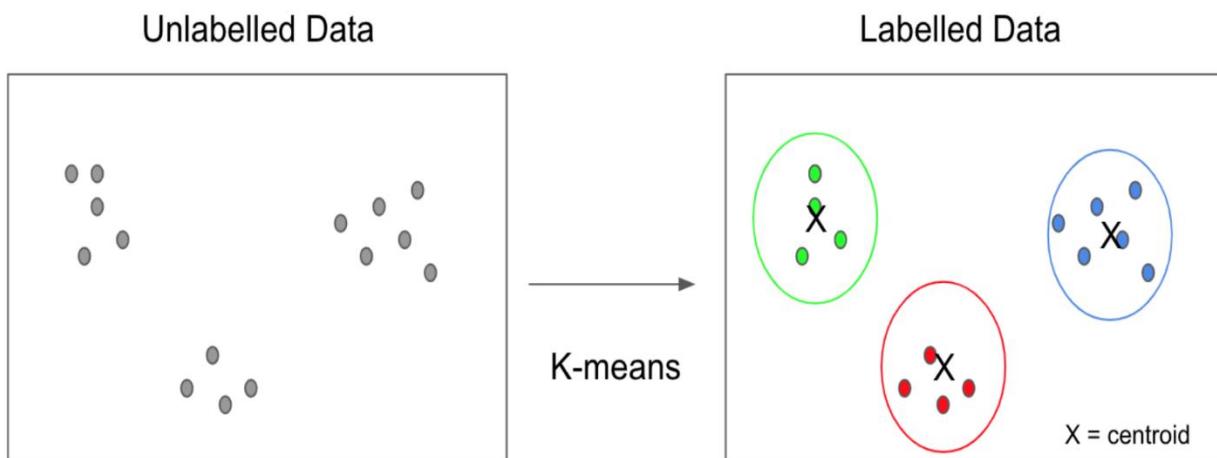
---

## *Introduction*

---

## i. Contexte et motivation :

Le clustering est une méthode d'apprentissage machine puissante qui consiste à regrouper des points de données. Avec un ensemble de différents points de données, les scientifiques peuvent utiliser un algorithme de regroupement pour classer chaque point de données dans un groupe particulier. Théoriquement, les points de données présents dans un même groupe contiennent des caractéristiques [...]



Les algorithmes de regroupement sont essentiels pour que les spécialistes des données puissent découvrir des regroupements innés parmi les données préétablies non étiquetées et étiquetées. Étonnamment, il n'existe pas de critères particuliers pour mettre en évidence un bon regroupement. Tout dépend des préférences et des besoins individuels, et de ce qu'un scientifique utilise pour répondre à ses besoins.

Disons, par exemple, que l'on pourrait être intéressé par la découverte de représentants de groupes homogènes (réduction des données), dans des clusters naturels et la définition de leurs propriétés inconnues. Certaines personnes souhaitent également trouver des objets de données non ordinaires et d'autres regroupements appropriés. Quoi qu'il en soit, cet algorithme fait plusieurs hypothèses constituant des similitudes entre divers points. De plus, chaque supposition crée de nouveaux groupes tout aussi bien fondés.

Grande échelle pour valider ces promesses. L'informatique quantique repose sur des principes fondamentaux de la mécanique quantique, tels que la superposition, l'intrication et l'interférence, qui permettent de manipuler des qubits (bits quantiques) de manière bien plus complexe que les bits classiques. Ces propriétés ouvrent la voie à des calculs parallèles massifs, mais elles posent également des défis techniques et théoriques considérables.

L'un des défis majeurs réside dans la **stabilité des qubits**. Contrairement aux bits classiques, qui sont soit dans un état 0, soit dans un état 1, les qubits peuvent exister dans une superposition de ces états. Cependant, cette superposition est extrêmement fragile et peut être perturbée par des interférences externes, un phénomène appelé **décohérence quantique**. Pour contrer cela, des techniques de correction d'erreurs quantiques sont en développement, mais elles nécessitent un nombre important de qubits physiques pour créer un seul qubit logique fiable, ce qui complique la mise à l'échelle des systèmes quantiques.

Un autre défi est la **complexité algorithmique**. Bien que des algorithmes quantiques prometteurs, comme l'algorithme de Shor pour la factorisation des nombres entiers ou l'algorithme de Grover pour la recherche non structurée, aient été proposés, leur implémentation pratique reste limitée par les contraintes techniques actuelles. De plus, tous les problèmes ne bénéficient pas d'une accélération quantique, et il est crucial d'identifier ceux pour lesquels l'informatique quantique apporte une réelle valeur ajoutée.

En ce qui concerne les **applications pratiques**, l'informatique quantique pourrait révolutionner des domaines tels que la cryptographie, l'optimisation, la simulation de systèmes quantiques (comme les molécules en chimie quantique) et l'apprentissage machine. Par exemple, dans le domaine du clustering évoqué précédemment, des algorithmes quantiques pourraient potentiellement accélérer le traitement de grands ensembles de données en exploitant des propriétés quantiques pour effectuer des calculs plus rapidement que les ordinateurs classiques. Cependant, ces applications restent largement théoriques à ce stade, et leur réalisation pratique nécessitera des avancées significatives en matière de matériel et de logiciels quantiques.

## ii. Problématique :

L'algorithme K-Means est largement utilisé pour le clustering de données en raison de sa simplicité et de son efficacité. Cependant, il présente plusieurs limites qui peuvent affecter ses performances dans des situations réelles, notamment la sensibilité à l'initialisation, la nécessité de spécifier le nombre de clusters à l'avance, et des hypothèses sur la forme des clusters. L'un des principaux problèmes de K-Means est sa dépendance au choix des centres de clusters initiaux. Des initialisations différentes peuvent conduire à des résultats très variés. Par exemple, un mauvais choix initial peut entraîner des minima locaux, ce qui signifie que l'algorithme ne trouve pas la meilleure solution possible. Cette problématique est exacerbée dans des ensembles de données complexes où la structure des données n'est pas évidente. De plus, K-Means nécessite que l'utilisateur spécifie le nombre de clusters (K) avant de commencer le processus d'optimisation. Cette exigence peut être problématique, car il n'est pas toujours évident de déterminer le nombre optimal de clusters. Dans certaines applications, le choix de K peut avoir un impact significatif sur les résultats, et des valeurs sous-optimales peuvent mener à une mauvaise interprétation des données. L'algorithme suppose également que les clusters ont une forme sphérique et de taille similaire. Cela ne correspond pas toujours à la réalité des données. Dans de nombreux cas, les clusters peuvent avoir des formes irrégulières ou des tailles variées. Cette hypothèse entraîne une mauvaise performance de l'algorithme lorsqu'il est appliqué à des données ayant des structures complexes, comme celles que l'on rencontre fréquemment dans des domaines tels que la biologie ou la finance. K-Means est également sensible aux valeurs aberrantes, qui peuvent fausser le calcul des centres de clusters. La présence de valeurs extrêmes peut entraîner un déplacement significatif des centres de clusters vers ces points, ce qui nuit à la qualité du clustering. Cette sensibilité limite l'efficacité de K-Means dans des ensembles de données comportant des anomalies ou des erreurs de mesure. Enfin, dans des espaces de haute dimension, la notion de distance devient moins significative en raison du phénomène connu sous le nom de "malédiction de la dimensionnalité". Cela complique le processus de clustering, car les points de données peuvent apparaître très proches les uns des autres, rendant difficile la séparation des clusters. Les algorithmes de clustering, y compris K-Means, peuvent donc perdre leur efficacité lorsqu'ils sont appliqués à des données à haute dimension.

L'informatique quantique émerge comme une solution potentielle pour surmonter certaines des limitations de K-Means. Grâce à ses propriétés uniques, comme la superposition et l'intrication, l'informatique quantique peut offrir des améliorations significatives dans les processus de clustering. Les algorithmes quantiques peuvent faciliter une recherche plus rapide des centres de clusters optimaux. Par exemple, l'algorithme de Grover permet d'accélérer la recherche dans des espaces de données non structurés, ce qui pourrait améliorer la convergence de K-Means. En utilisant des techniques quantiques, il est possible de réduire le temps nécessaire pour atteindre une solution optimale, rendant l'algorithme plus efficace. De plus, l'informatique quantique a le potentiel de traiter de grandes quantités de données de manière efficace. Les ordinateurs quantiques peuvent gérer des ensembles de données massifs sans une augmentation proportionnelle du temps de calcul. Cela est particulièrement pertinent dans le contexte du big data, où les méthodes classiques de traitement des données peuvent devenir insuffisantes. Grâce aux propriétés de superposition et d'intrication, les ordinateurs quantiques peuvent explorer simultanément plusieurs configurations de clustering. Cela

réduit le risque de se coincer dans des minima locaux, un problème courant dans les algorithmes de clustering traditionnels. En permettant une exploration plus large de l'espace de solutions, l'informatique quantique peut offrir des résultats plus robustes et fiables. Enfin, les algorithmes quantiques pourraient potentiellement adapter le nombre de clusters en fonction des caractéristiques des données elles-mêmes. Cela éliminerait la nécessité pour l'utilisateur de spécifier un nombre de clusters à l'avance, rendant le processus de clustering plus flexible et adaptable. Par exemple, des techniques d'apprentissage automatique quantique pourraient identifier automatiquement les structures sous-jacentes des données et ajuster le clustering en conséquence.

### iii. Objectifs du projet :

- **Objectif Principal :**

**Étudier et comparer les approches classique et quantique de l'algorithme K-Means** pour le partitionnement de données, en mettant en avant les avantages et les limites de chaque méthode.

- **Objectifs Spécifiques :**

- Comprendre l'Algorithme K-Means Classique
  - Analyser le fonctionnement de l'algorithme K-Means traditionnel.
  - Identifier ses domaines d'application et ses limites (par exemple, sensibilité à l'initialisation, convergence locale, etc.).
- Explorer l'Algorithme Quantum K-Means
  - Étudier les principes quantiques sous-jacents à l'algorithme Quantum K-Means (superposition, intrication, calcul quantique).
  - Comprendre comment l'informatique quantique peut améliorer les performances du clustering.
- Implémenter les Deux Approches
  - Implémenter l'algorithme K-Means classique en utilisant des bibliothèques standards (comme Scikit-learn en Python).
  - Implémenter l'algorithme Quantum K-Means en utilisant une bibliothèque quantique (comme Cirq, PennyLane ou Qiskit).
- Comparer les Performances
  - Évaluer les performances des deux algorithmes en termes de :
  - Précision : Qualité des clusters obtenus.
  - Rapidité : Temps d'exécution et complexité algorithmique.
  - Évolutivité : Capacité à traiter des ensembles de données de grande taille.
  - Identifier les cas d'utilisation où l'approche quantique est plus avantageuse.
- Discuter des Perspectives
  - Analyser les défis actuels de l'approche quantique (bruit, décohérence, limites matérielles).
  - Proposer des pistes d'amélioration pour l'algorithme Quantum K-Means.

- **Objectifs Pédagogiques :**

- Acquérir une compréhension approfondie des algorithmes de clustering et de l'informatique quantique.
- Développer des compétences pratiques en implémentation d'algorithmes classiques et quantiques.
- Savoir analyser et interpréter des résultats expérimentaux de manière critique.

---

## *Fondements Théoriques*

---

Les ensembles de données de machine learning peuvent contenir des millions d'exemples, mais tous les algorithmes de clustering ne sont pas évolutifs de manière efficace. De nombreux algorithmes de clustering calculent la similarité entre toutes les paires d'exemples, ce qui signifie que leur temps d'exécution augmente comme le carré du nombre d'exemples  $n$ , désigné par  $O(n^2)$  dans la notation de complexité . Les algorithmes  $O(n^2)$  ne sont pas pratiques pour les ensembles de données contenant des millions d'exemples.

**L'algorithme k-moyennes** a une complexité de  $O(n)$ , ce qui signifie que l'algorithme évolue de manière linéaire avec  $n$ . C'est l'algorithme qui sera au centre de ce cours.

## i. Algorithme K-Means :

Le clustering K-means est un algorithme populaire en apprentissage non supervisé utilisé pour partitionner un ensemble de données en un nombre prédéfini de clusters (groupes). L'objectif est de regrouper des points de données similaires ensemble et de découvrir des motifs ou des structures sous-jacentes dans les données.

- **Algorithme K-Means Classique :**

L'algorithme K-Means regroupe les données en essayant de séparer les échantillons en  $n$  groupes de variance égale, en minimisant un critère appelé inertie ou somme des carrés intra-cluster (voir ci-dessous). Cet algorithme nécessite que le nombre de clusters soit spécifié. Il est bien adapté à un grand nombre d'échantillons et a été utilisé dans un large éventail de domaines d'application.

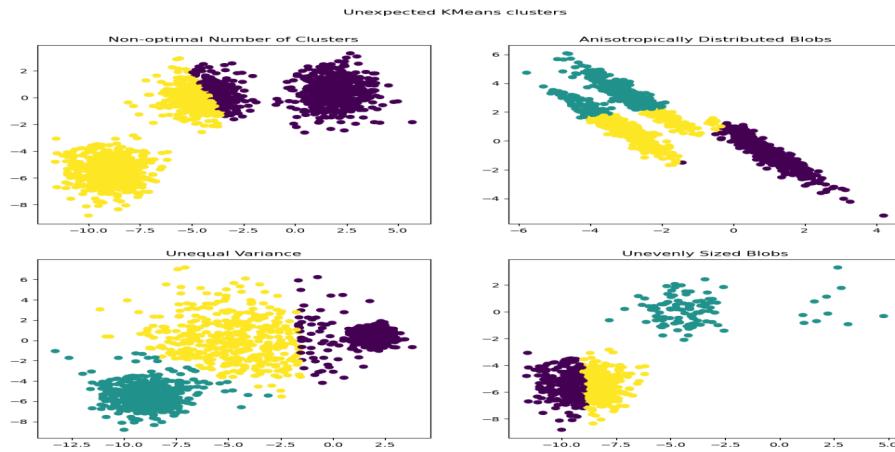
L'algorithme k-means divise un ensemble d'échantillons en clusters disjoints, chacun décrit par la moyenne des échantillons du cluster. Ces moyennes sont communément appelées "centroïdes" du cluster ; notez qu'ils ne sont généralement pas des points de l'ensemble original, bien qu'ils existent dans le même espace.

L'algorithme K-means vise à choisir des centroïdes qui minimisent l'inertie, ou critère de la somme des carrés intra-cluster :

$$\sum_{i=0}^n \min_{\mu_j \in C} (\|x_i - \mu_j\|^2)$$

L'inertie peut être considérée comme une mesure de la cohérence interne des clusters. Cependant, elle présente plusieurs inconvénients :

- L'inertie suppose que les clusters sont **convexes et isotropes**, ce qui n'est pas toujours le cas. Elle réagit mal aux clusters allongés ou aux formes irrégulières (comme les variétés non linéaires).
- L'inertie n'est pas une métrique normalisée : on sait simplement que **plus les valeurs sont basses, mieux c'est**, et que zéro est optimal. Cependant, dans les espaces de très haute dimension, les distances euclidiennes ont tendance à devenir artificiellement grandes (un phénomène connu sous le nom de "**malédiction de la dimensionnalité**"). Pour atténuer ce problème et accélérer les calculs, il peut être utile d'appliquer au préalable un algorithme de réduction de dimension, comme l'**Analyse en Composantes Principales (ACP)**, avant un clustering par k-means.



### a. Principe de fonctionnement :

- Initialisation:
  - Le choix des centroïdes initiaux est crucial. Une mauvaise initialisation peut conduire à des clusters de mauvaise qualité.
  - Des méthodes comme **K-means++** ont été développées pour améliorer l'initialisation en choisissant des centroïdes initiaux plus éloignés les uns des autres, ce qui réduit le risque de convergence vers des minima locaux.
- Assигnation:
  - La distance utilisée pour l'assигnation est généralement la **distance euclidienne**, mais d'autres métriques comme la **distance de Manhattan** ou la **distance cosinus** peuvent être utilisées selon la nature des données.
  - Cette étape est coûteuse en calculs, surtout pour des données de grande dimension ou des ensembles de données volumineux.
- Mise à jour des centroïdes:
  - Le recalcul des centroïdes est une moyenne des points dans chaque cluster. Cela signifie que les outliers (points aberrants) peuvent fortement influencer la position des centroïdes.
  - Pour atténuer ce problème, des variantes de K-means comme **K-medoids** utilisent des points réels du jeu de données comme centres (plutôt que des moyennes), ce qui est plus robuste aux outliers.
- Convergence:
  - La convergence est atteinte lorsque les centroïdes ne changent plus significativement ou après un nombre fixe d'itérations.
  - Cependant, K-means ne garantit pas une convergence vers le **minimum global** de la fonction de coût (somme des distances au carré entre les points et leurs centroïdes). Il peut rester coincé dans un **minimum local**.

### b. Démonstration des hypothèses du k-means :

Cet exemple vise à illustrer des situations où l'algorithme k-means produit des clusters contre-intuitifs et potentiellement indésirables.

- Génération des données : La fonction `make\_blobs` génère des blobs gaussiens isotropes (sphériques). Pour obtenir des blobs gaussiens anisotropes (elliptiques), il faut définir une transformation linéaire.

```

import numpy as np

from sklearn.datasets import make_blobs

n_samples = 1500
random_state = 170
transformation = [[0.60834549, -0.63667341], [-0.40887718, 0.85253229]]

X, y = make_blobs(n_samples=n_samples, random_state=random_state)
X_aniso = np.dot(X, transformation) # Anisotropic blobs
X_varied, y_varied = make_blobs(
    n_samples=n_samples, cluster_std=[1.0, 2.5, 0.5], random_state=random_state
) # Unequal variance
X_filtered = np.vstack(
    (X[y == 0][:500], X[y == 1][:100], X[y == 2][:10])
) # Unevenly sized blobs
y_filtered = [0] * 500 + [1] * 100 + [2] * 10

```

Nous pouvons visualiser les données obtenues :

```

import matplotlib.pyplot as plt

fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(12, 12))

axs[0, 0].scatter(X[:, 0], X[:, 1], c=y)
axs[0, 0].set_title("Mixture of Gaussian Blobs")

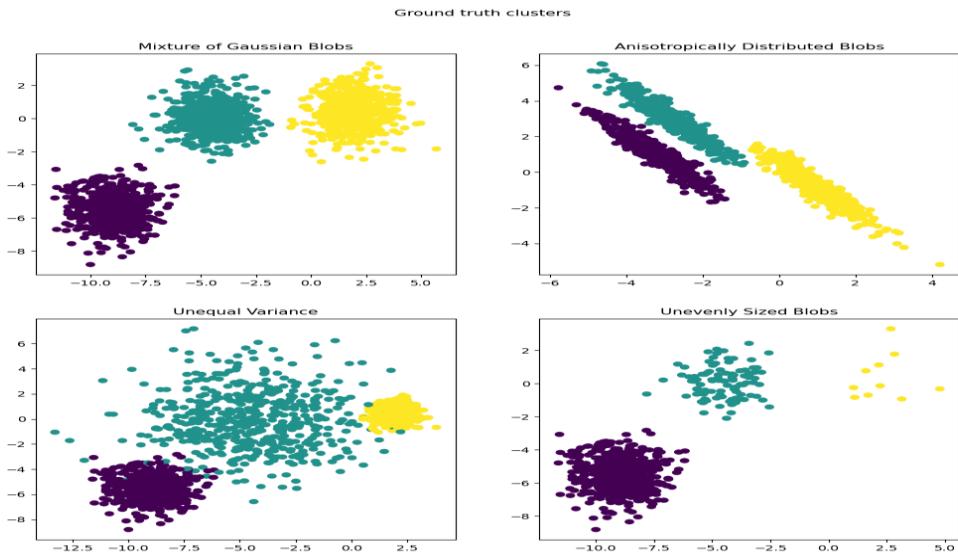
axs[0, 1].scatter(X_aniso[:, 0], X_aniso[:, 1], c=y)
axs[0, 1].set_title("Anisotropically Distributed Blobs")

axs[1, 0].scatter(X_varied[:, 0], X_varied[:, 1], c=y_varied)
axs[1, 0].set_title("Unequal Variance")

axs[1, 1].scatter(X_filtered[:, 0], X_filtered[:, 1], c=y_filtered)
axs[1, 1].set_title("Unevenly Sized Blobs")

plt.suptitle("Ground truth clusters").set_y(0.95)
plt.show()

```



- **Ajustement des modèles et visualisation des résultats :** Les données générées précédemment sont maintenant utilisées pour illustrer le comportement de KMeans dans les scénarios suivants :
  - Nombre de clusters non optimal :  
Dans un cas réel, il n'existe pas de nombre "vrai" et unique de clusters. Un nombre approprié doit être déterminé à partir de critères statistiques et de la connaissance des objectifs visés.
  - anisotrope des blobs :  
K-means minimise les distances euclidiennes entre les échantillons et le centroïde de leur cluster assigné. Par conséquent, il est plus adapté aux clusters isotropes et normalement distribués (i.e., des gaussiennes sphériques).
  - Variances inégales :  
K-means équivaut à un estimateur du maximum de vraisemblance pour un "mélange" de **k** distributions gaussiennes ayant les mêmes variances mais des moyennes potentiellement différentes.
  - Tailles inégales des blobs :  
Aucun résultat théorique n'exige que K-means nécessite des clusters de taille similaire pour bien fonctionner. Cependant, la minimisation des distances euclidiennes implique que, plus le problème est épars et de haute dimension, plus il est nécessaire d'exécuter l'algorithme avec différentes initialisations de centroïdes pour garantir une inertie globale minimale.

```

from sklearn.cluster import KMeans

common_params = {
    "n_init": "auto",
    "random_state": random_state,
}

fig, axs = plt.subplots(nrows=2, ncols=2, figsize=(12, 12))

y_pred = KMeans(n_clusters=2, **common_params).fit_predict(X)
axs[0, 0].scatter(X[:, 0], X[:, 1], c=y_pred)
axs[0, 0].set_title("Non-optimal Number of Clusters")

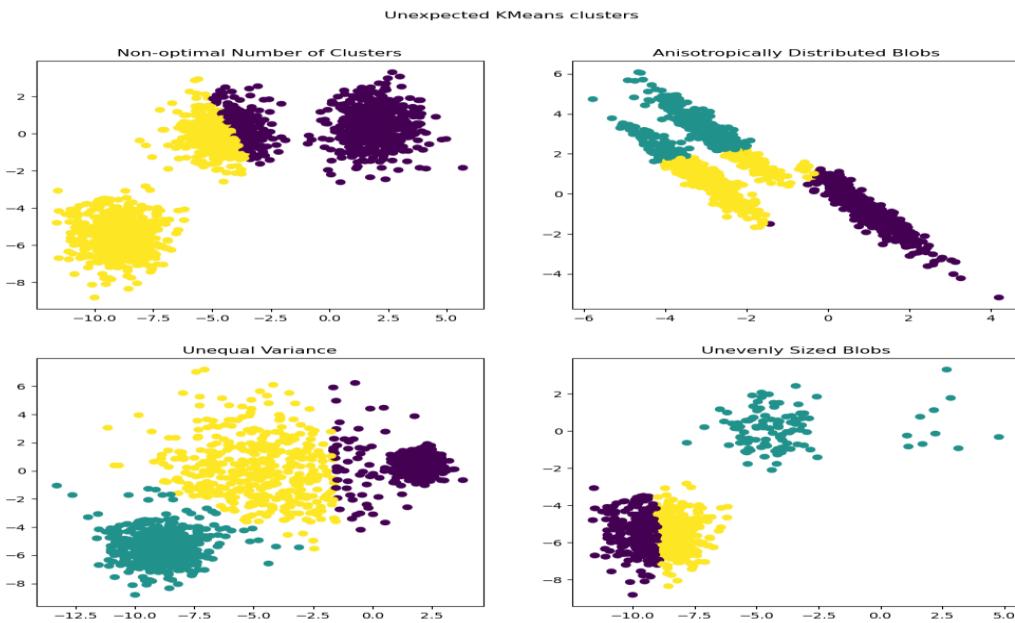
y_pred = KMeans(n_clusters=3, **common_params).fit_predict(X_aniso)
axs[0, 1].scatter(X_aniso[:, 0], X_aniso[:, 1], c=y_pred)
axs[0, 1].set_title("Anisotropically Distributed Blobs")

y_pred = KMeans(n_clusters=3, **common_params).fit_predict(X_varied)
axs[1, 0].scatter(X_varied[:, 0], X_varied[:, 1], c=y_pred)
axs[1, 0].set_title("Unequal Variance")

y_pred = KMeans(n_clusters=3, **common_params).fit_predict(X_filtered)
axs[1, 1].scatter(X_filtered[:, 0], X_filtered[:, 1], c=y_pred)
axs[1, 1].set_title("Unevenly Sized Blobs")

plt.suptitle("Unexpected KMeans clusters").set_y(0.95)
plt.show()

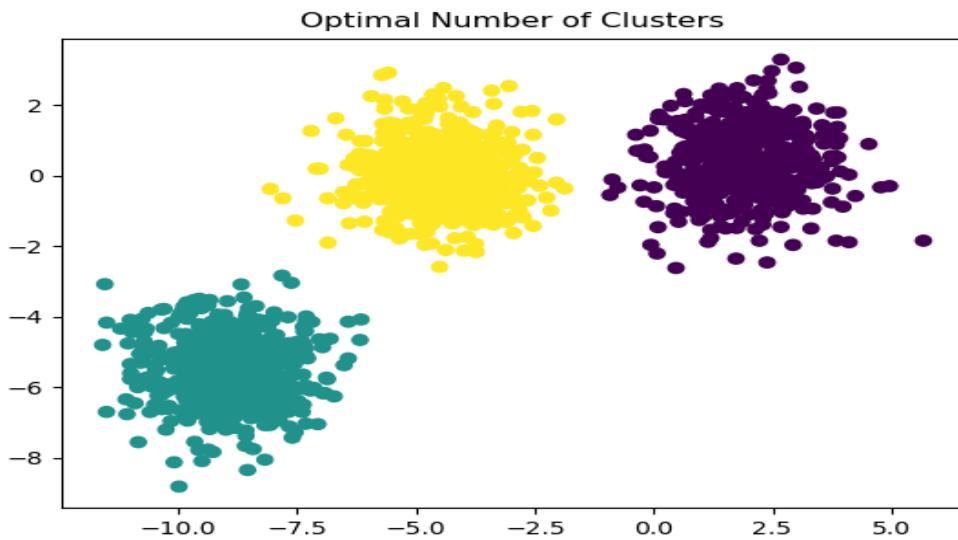
```



- Solutions possibles :

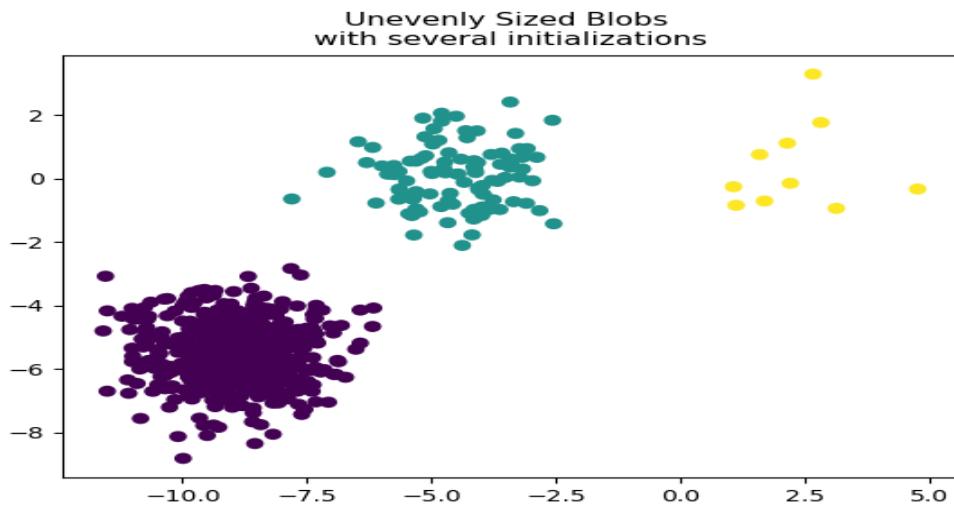
Pour un exemple illustrant comment déterminer le nombre optimal de clusters, consultez "**Sélection du nombre de clusters par analyse de silhouette avec K-Means**". Dans ce cas précis, il suffit de définir `n\_clusters=3`.

```
y_pred = KMeans(n_clusters=3, **common_params).fit_predict(X)
plt.scatter(X[:, 0], X[:, 1], c=y_pred)
plt.title("Optimal Number of Clusters")
plt.show()
```



Pour traiter des amas de taille inégale, une solution consiste à augmenter le nombre d'initialisations aléatoires. Dans ce cas, nous définissons `n\_init=10` pour éviter de converger vers un minimum local sous-optimal. Pour plus de détails, voir la section **Clustering de données éparses avec k-means**.

```
y_pred = KMeans(n_clusters=3, n_init=10, random_state=random_state).fit_predict(
    X_filtered
)
plt.scatter(X_filtered[:, 0], X_filtered[:, 1], c=y_pred)
plt.title("Unevenly Sized Blobs \nwith several initializations")
plt.show()
```



Comme les distributions anisotropes et les variances inégales constituent des limitations intrinsèques de l'algorithme k-means, nous proposons ici d'utiliser plutôt [GaussianMixture](#). Cette approche suppose également des clusters gaussiens, mais sans imposer de contraintes sur leurs variances. Notons qu'il reste néanmoins nécessaire de déterminer le nombre optimal d'amas (voir [Sélection de modèle par mélange gaussien](#)).

Pour un exemple illustrant comment d'autres méthodes de clustering traitent les amas anisotropes ou à variance inégale, consulter le tutoriel [Comparaison des algorithmes de clustering sur des jeux de données tests](#).

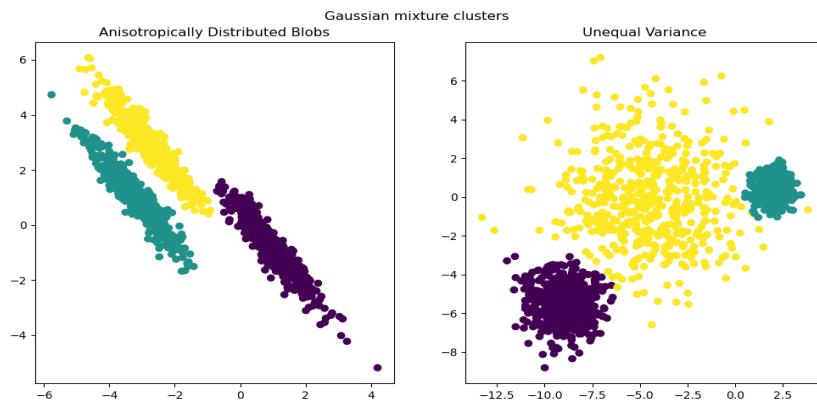
```
from sklearn.mixture import GaussianMixture

fig, (ax1, ax2) = plt.subplots(nrows=1, ncols=2, figsize=(12, 6))

y_pred = GaussianMixture(n_components=3).fit_predict(X_aniso)
ax1.scatter(X_aniso[:, 0], X_aniso[:, 1], c=y_pred)
ax1.set_title("Anisotropically Distributed Blobs")

y_pred = GaussianMixture(n_components=3).fit_predict(X_varied)
ax2.scatter(X_varied[:, 0], X_varied[:, 1], c=y_pred)
ax2.set_title("Unequal Variance")

plt.suptitle("Gaussian mixture clusters").set_y(0.95)
plt.show()
```



### c. Complexité algorithmique :

La complexité de K-means est principalement due à :

- **Assignation** : Pour chaque point  $n$ , on calcule la distance à chaque centroïde  $K$ , ce qui donne une complexité de  $O(n \cdot K \cdot d)$  par itération.
- **Mise à jour des centroïdes** : Le recalcul des centroïdes nécessite de parcourir tous les points dans chaque cluster, ce qui est également coûteux.

Pour des données de grande dimension ( **$d$  élevé**) ou un grand nombre de clusters ( **$K$  élevé**), le temps de calcul peut devenir prohibitif.

### c. Limites :

- **Dépendance à l'initialisation :**
  - Une mauvaise initialisation peut conduire à des clusters de mauvaise qualité. Par exemple, si deux centroïdes initiaux sont trop proches, ils peuvent finir par capturer les mêmes points, laissant d'autres clusters sous-représentés.
  - Des méthodes comme K-means++ ou l'initialisation basée sur des échantillons aléatoires stratifiés peuvent atténuer ce problème.
- **Convergence locale :**
  - K-means minimise la somme des distances au carré entre les points et leurs centroïdes, mais il peut converger vers un minimum local plutôt que global.
  - Des techniques comme le **recuit simulé** ou les **algorithmes génétiques** peuvent être utilisées pour explorer plus largement l'espace des solutions.
- **Sensibilité aux outliers :**
  - Les outliers peuvent déplacer les centroïdes, ce qui peut fausser les résultats. Par exemple, un point aberrant avec des valeurs extrêmes peut attirer un centroïde loin des autres points.
  - Des variantes comme **K-medoids** ou l'utilisation de distances robustes (comme la distance de Manhattan) peuvent aider à réduire cette sensibilité.
- **Choix du nombre de clusters ( $K$ ) :**
  - Déterminer le bon nombre de clusters est un défi. Des méthodes comme la **méthode du coude** (elbow method), l'**indice de silhouette**, ou l'**algorithme de la silhouette moyenne** peuvent aider à choisir ( $K$ ).

- **Domaines d'Application du K-Means :**

- a. **Exemples d'utilisation :**

- **Segmentation de clientèle :**

- Les entreprises utilisent K-means pour regrouper les clients en fonction de leurs comportements d'achat, de leur âge, de leur localisation, etc.
      - Par exemple, une entreprise de e-commerce peut utiliser K-means pour identifier des groupes de clients ayant des habitudes d'achat similaires, ce qui permet de personnaliser les offres promotionnelles.

- **Segmentation d'images :**

- En traitement d'images, K-means est utilisé pour regrouper des pixels similaires en fonction de leur couleur ou de leur intensité.
    - Par exemple, dans une image satellite, K-means peut être utilisé pour identifier des zones urbaines, des forêts, ou des plans d'eau.

- **Analyse de documents :**

- K-means peut regrouper des documents en fonction de leur contenu textuel. Cela est utile pour la classification de textes ou la détection de sujets.
    - Par exemple, dans un ensemble de documents scientifiques, K-means peut regrouper les articles traitant de sujets similaires, comme l'intelligence artificielle ou la biologie moléculaire.

- **Recommandation de produits :**

- Les moteurs de recommandation utilisent K-means pour regrouper des produits ou des utilisateurs ayant des préférences similaires.
    - Par exemple, une plateforme de streaming peut utiliser K-means pour regrouper les utilisateurs en fonction de leurs habitudes de visionnage, ce qui permet de recommander des films ou des séries pertinents.

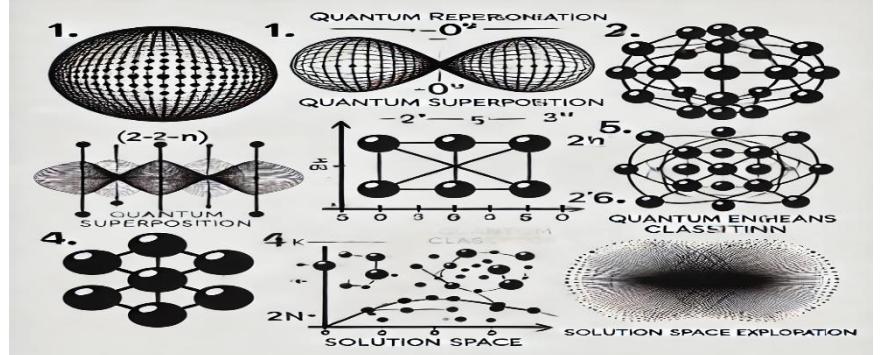
- **Bio-informatique :**

- En biologie, K-means est utilisé pour regrouper des gènes ou des protéines ayant des fonctions similaires.
    - Par exemple, dans une étude génomique, K-means peut être utilisé pour identifier des groupes de gènes co-exprimés, ce qui peut révéler des voies métaboliques communes.

- Transition vers l'Approche Quantique :

Face aux limitations du K-Means classique dans le traitement des données complexes, l'**informatique quantique** émerge comme une solution prometteuse. Cette section explore comment les principes de la mécanique quantique - superposition, intrication et parallélisme quantique - peuvent révolutionner l'algorithme K-Means traditionnel en :

- Accélération exponentielle des calculs de distances
- Optimisation améliorée des centroides grâce aux q-bits
- Traitement natif des structures de données non linéaires



Nous examinerons les fondements théoriques du **Quantum K-Means** et son potentiel à surmonter les défis des versions classiques, tout en identifiant les contraintes technologiques actuelles.

### a. Introduction à l'informatique quantique :

- **Qubits :**
  - Un qubit peut exister dans un état de superposition, c'est-à-dire une combinaison linéaire de 0 et 1. Cela permet de représenter plusieurs états simultanément.
  - Par exemple, un qubit peut être dans l'état  $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ , ce qui signifie qu'il est à la fois 0 et 1 avec une probabilité égale.
- **Superposition :**
  - La superposition permet à un ordinateur quantique de traiter un grand nombre de possibilités en parallèle. Par exemple, un registre de (**n**) qubits peut représenter ( $2^n$ ) états simultanément.
- **Intrication :**
  - L'intrication permet à deux qubits d'être corrélés, même à distance. Cela signifie que l'état d'un qubit peut dépendre de l'état d'un autre, ce qui permet une communication et une coordination instantanées.

### b. Avantages potentiels de l'informatique quantique pour le clustering :

- **Accélération des calculs :**
  - Les algorithmes quantiques peuvent explorer plusieurs solutions en parallèle grâce à la superposition, ce qui peut réduire considérablement le temps de calcul.
  - Par exemple, un algorithme quantique pourrait explorer plusieurs clusters simultanément, ce qui accélérerait la convergence.
- **Optimisation améliorée :**
  - Les algorithmes quantiques peuvent mieux explorer l'espace des solutions, ce qui peut aider à éviter les minima locaux.
  - Par exemple, l'algorithme quantique de **recuit simulé quantique** peut explorer plus efficacement l'espace des solutions que son homologue classique.
- **Traitement de grandes dimensions :**
  - L'informatique quantique peut être particulièrement utile pour traiter des données de grande

- dimension, ce qui est courant dans les applications de clustering.
- Par exemple, dans des données génomiques où chaque gène peut avoir des milliers de caractéristiques, un algorithme quantique pourrait être plus efficace.

### c. Justification de l'utilisation d'un algorithme quantique pour K-Means :

- Complexité réduite :
  - Un algorithme quantique pourrait réduire la complexité algorithmique de K-means, en particulier pour les grands ensembles de données.
  - Par exemple, un algorithme quantique pourrait explorer plusieurs clusters en parallèle, réduisant ainsi le temps nécessaire pour trouver les centroïdes optimaux.
- Meilleure initialisation :
  - Les techniques quantiques pourraient offrir des méthodes plus robustes pour l'initialisation des centroïdes, réduisant ainsi la dépendance à l'initialisation aléatoire.
  - Par exemple, un algorithme quantique pourrait utiliser des techniques de superposition pour explorer plusieurs initialisations simultanément.
- Exploration efficace de l'espace des solutions :
  - Les algorithmes quantiques pourraient explorer plus efficacement l'espace des solutions, ce qui pourrait conduire à des clusters de meilleure qualité.
  - Par exemple, un algorithme quantique pourrait utiliser l'intrication pour explorer des combinaisons de clusters qui seraient difficiles à explorer avec un algorithme classique.

---

*Analyse et Implémentation de  
l'Algorithme Quantique*

---

**L'apprentissage automatique quantique** (QML) représente la branche de l'[informatique quantique](#) (QC) qui vise à reconcevoir les algorithmes classiques d'exploration de données et d'apprentissage automatique - ou leurs sous-routines les plus coûteuses - pour les exécuter sur un ordinateur quantique potentiel [1]. De nombreux algorithmes QML ont été étudiés récemment [2]. Dans ce travail, nous nous concentrons sur la tâche de clustering, qui vise à regrouper des données non étiquetées en clusters de données similaires selon une métrique de distance. Notre objectif est de concevoir des versions hybrides-quantiques de k-Means [3], l'un des algorithmes de clustering les plus connus. Plus précisément, k-Means produit un clustering où les éléments d'un groupe sont plus proches du centre géométrique (appelé centroïde) de leur groupe que des centroïdes des autres groupes, selon une mesure de distance spécifique.

La création d'une version quantique de l'algorithme k-Means consiste à concevoir un circuit quantique qui prend des données classiques en entrée et exploite des portes quantiques pour effectuer les calculs, tout en respectant les contraintes de la mécanique quantique. Cependant, les dispositifs quantiques disponibles actuellement ont des capacités limitées et sont sujets à des erreurs, rendant difficile l'exécution fiable d'un algorithme entièrement quantique. Pour contourner ces limitations, les méthodes hybrides combinant circuits quantiques peu profonds et calcul classique constituent des solutions pratiques pour exploiter efficacement le matériel quantique actuel. Dans ce contexte, nous proposons trois algorithmes hybrides qui intègrent progressivement des sous-routines quantiques exploitant le parallélisme quantique pour calculer simultanément les distances entre les centroïdes et les enregistrements du jeu de données. Plus précisément :

1. **q1-K-Means** : Quantifie le calcul de distance unique entre un centroïde et un enregistrement
2. **q2-K-Means** : Généralise le classifieur quantique kNN de [4] pour calculer en parallèle les distances entre un enregistrement et tous les centroïdes
3. **q3-K-Means** : Calcule simultanément les distances entre tous les enregistrements et tous les centroïdes

Une contrainte importante du calcul quantique est que les données doivent être normalisées selon la norme L2. Cependant, une normalisation directe peut poser problème car plusieurs clusters pourraient être projetés dans la même région de la sphère unité. Pour résoudre ce problème, nous utilisons par défaut une Projection Stéréographique Inverse (ISP) comme prétraitement des données, permettant de projeter des données N-dimensionnelles sur la surface d'une sphère dans un espace (N+1)-dimensionnel, améliorant ainsi la séparation des clusters.

Un autre défi important dans l'implémentation pratique d'algorithmes quantiques concerne la post-sélection. Ce problème survient car, dans certains algorithmes quantiques, seules certaines branches du calcul (en superposition) conduisent à des mesures significatives, tandis que d'autres doivent être ignorées. Ainsi, un nombre plus élevé d'exécutions du circuit (appelées "shots") est nécessaire pour obtenir des résultats précis. Dans ce travail, nous ne déterminons pas analytiquement le nombre de shots garantissant la précision des résultats, mais l'établissons de manière empirique.

Pour évaluer la qualité du clustering q-K-Means, nous implémentons les trois versions en utilisant le framework qiskit d'IBM et les testons sur des jeux de données synthétiques et réels. Nous utilisons également l'algorithme classique  $\delta$ -k-means [5] comme référence pour évaluer nos versions quantiques. Nos expériences montrent que les trois versions produisent des résultats de clustering comparables, à condition que le nombre de shots augmente proportionnellement au nombre de vecteurs encodés dans les circuits.

Cet article est une version étendue de notre précédent travail présenté en conférence [6]. Alors que nous avions initialement introduit une première version de q1-K-Means, nous proposons ici la version finale de cet algorithme ainsi que deux nouveaux algorithmes (q2-K-Means et q3-K-Means), accompagnés d'une nouvelle technique de prétraitement, d'une analyse du nombre de shots requis, et d'un nouvel ensemble d'expérimentations. Le document est organisé comme suit : la Section 2 présente les travaux connexes, la Section 3 introduit les notations et notions préliminaires, la Section 4 décrit le K-Means quantique, la Section 5 présente les résultats expérimentaux, et enfin la Section 6 résume nos contributions et esquisse des directions futures de recherche.

## i. Travaux connexes :

Le problème du clustering quantique peut être abordé de plusieurs manières. Certaines études s'inspirent directement de la théorie quantique. Par exemple, la méthode de clustering classique proposée dans [7] s'appuie sur une intuition physique dérivée de la mécanique quantique. Dans [8], les auteurs effectuent un clustering en exploitant une réduction bien connue vers le problème Maximum-Cut, résolu ensuite à l'aide d'un algorithme quantique d'optimisation combinatoire approchée. Les travaux [9] présentent un algorithme quantique non supervisé pour le clustering k-Means basé sur le calcul adiabatique quantique [10], tandis que [11] propose un algorithme génétique quantique inspiré pour le clustering k-Means.

L'approche générale pour quantifier les algorithmes de clustering classiques consiste à remplacer les parties les plus coûteuses par des sous-routines quantiques plus efficaces. Par exemple, [12] utilise la mesure de fidélité quantique pour calculer les distances entre paires d'enregistrements. Cette fidélité est estimée efficacement via un circuit quantique minimal (deux portes Hadamard et une Control-Swap [13]). Cependant, l'article n'aborde pas le traitement des données classiques, supposant des entrées déjà sous forme quantique.

D'autres approches proposent des routines quantiques complètes pour le clustering. Le travail [14] utilise deux sous-routines basées sur l'algorithme de recherche de Grover [15] pour accélérer les méthodes classiques. Les auteurs implémentent une version quantique de k-Means où chaque enregistrement est encodé dans un état quantique, puis utilise un oracle (non implémenté pratiquement) pour calculer les distances et identifier les centroïdes. Une approche similaire apparaît dans [16], combinant le test SWAP et la recherche de Grover pour l'affectation aux centroïdes.

Plus récemment, [17] propose un algorithme k-Means quantique utilisant la distance de Manhattan [18], atteignant une accélération quadratique par rapport à la version classique. D'autres travaux combinent différentes techniques quantiques [19][20][21].

Enfin, [5] présente q-Means, une version quantique offrant une accélération exponentielle dans le nombre d'enregistrements, bien que l'évaluation reste théorique via une simulation classique ( $\delta$ -k-Means). Cette approche a ensuite été étendue pour un clustering spectral quantique [22].

Contrairement à la littérature existante, notre travail se concentre sur les défis pratiques d'implémentation, notamment l'encodage des données classiques en états quantiques. En exploitant le parallélisme quantique, nos algorithmes hybrides réduisent la complexité de l'étape d'affectation des clusters à un coût constant par exécution de circuit (hors préparation des états) [23].

## ii. Cadre théorique :

Pour garantir l'autonomie de notre article, nous résumons ici les concepts clés. Étant donné un jeu de données  $\mathbf{X}$  contenant  $M$  enregistrements de dimension  $N$ , le clustering vise à affecter chaque enregistrement à un parmi  $k$  clusters  $\mathbf{C}_1, \dots, \mathbf{C}_k$ , représentés par leurs centroïdes  $\mathbf{c}_1, \dots, \mathbf{c}_k$ , selon une mesure de similarité. Nous utiliserons les notations suivantes :  $\hat{n} = \lceil \log_2 N \rceil$ ,  $\hat{k} = \lceil \log_2 k \rceil$  and  $\hat{M} = \lceil \log_2 M \rceil$

$\mathbf{x}_i$  (enregistrements),  $\mathbf{c}_j$  (centroïdes) et  $\mathbf{C}_j$  (clusters).

### ○ k-Means et δ-k-Means :

L'algorithme k-Means [24], après une initialisation aléatoire des centroïdes, alterne deux étapes jusqu'à convergence :

Affectation des clusters : Chaque point est assigné au centroïde le plus proche selon la distance euclidienne

$$d(\vec{p}, \vec{q}) = \|\vec{p} - \vec{q}\|_2 = \sqrt{\sum_{i=1}^N (p_i - q_i)^2},$$

(Éq. 1)

Mise à jour des centroïdes : Recalcul des centres des clusters. Nous nous concentrons sur la première étape, de complexité classique  $O(kMN)$ . La version quantique **q-Means** [5] est évaluée via **δ-k-Means**, une approximation simulant les erreurs quantiques. Dans notre adaptation (Algorithm 1), nous n'introduisons du bruit  $\delta$  que dans l'étape d'affectation. Pour un point  $\mathbf{x}_i$  et son centroïde le plus proche  $\mathbf{c}_j$ , **δ-k-Means** définit un ensemble de labels possibles  $L_i$  incluant tous les centroïdes à distance au plus  $d(\mathbf{x}_i, \mathbf{c}_j) + \delta$ . Le choix final est aléatoire dans  $L_i$ . Comme démontré dans [5], pour des données bien clusterisables et un  $\delta$  adapté, cette méthode préserve des affectations correctes.

---

```

Input: D - input data, k - number of clusters
Output: L - records to clusters assignment, C - centroids
1 C ← initCentroids(D, k);                                // centroid initialization
2 while convergence is not achieved do
3   for r ∈ D do
4     ē ← argmin(d(r, ē_j)) ∀ ē_j ∈ C;                  // for each record
5     L_δ(r) ← {p : |d²(r, ē) - d²(r, ē_p)| ≤ δ};    // find nearest centroid
6     c_j ← rand(L_δ(r));                               // find possible labels
7     C_j ← C_j ∪ {r};                                 // pick a random centroid
8     L(r) ← j;                                       // assign r to cluster C_j
9   for j ∈ [1, k] do
10    ē_j ← 1 / |C_j| ∑_{r ∈ C_j} r;                   // assign label j to r
11   return L, C;                                     // update cluster center ē_j
                                                 // return assignments and centroids

```

---

### ○ Estimation Quantique de Distance :

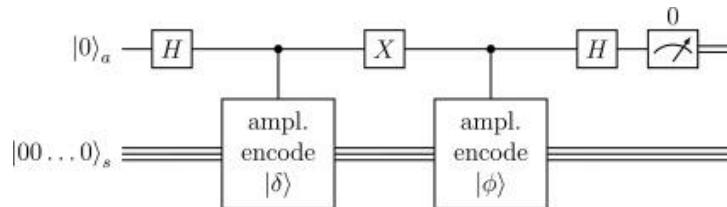
L'information classique peut être encodée de différentes manières dans un état quantique. Dans [27], plusieurs stratégies d'encodage et algorithmes quantiques de calcul de distance sont passés en revue. Le processus consistant à encoder des caractéristiques d'entrée dans l'amplitude d'un système quantique est appelé **encodage par amplitude** [4].

Une mesure de distance [24] couramment utilisée en apprentissage automatique classique et quantique (QML) est la **distance euclidienne** définie en (1). Nous présentons ici un circuit (Fig. 1) permettant de calculer la distance euclidienne quantique ( $|x\rangle, |y\rangle$ ) entre deux états quantiques  $\langle|x\rangle$  et  $\langle|y\rangle$ , encodés dans un registre **\*\*s\*\*** via l'encodage par amplitude [28]. Ce circuit nécessite un qubit auxiliaire (**ancilla**, noté **a**) intriqué avec les deux états ( $|x\rangle$ ) et ( $|y\rangle$ ).

Le calcul de la distance euclidienne quantique s'effectue comme suit :

1. Une porte **Hadamard** est appliquée sur l'ancilla **a**.
2. Les états ( $|x\rangle$ ) et ( $|y\rangle$ ) sont chargés dans le registre **s**, conditionnellement à l'état de l'ancilla.  
Ainsi, l'état initial ( $|0\rangle_a |0\rangle_s$ ) évolue en  $\frac{1}{\sqrt{2}}(|0\rangle_a |x\rangle_s + |1\rangle_a |y\rangle_s)$
3. Une seconde porte **Hadamard** sur **a** fait évoluer l'état vers :

$$\frac{1}{2}(|0\rangle_a (|\delta\rangle_s + |\phi\rangle_s) + |1\rangle_a (|\delta\rangle_s - |\phi\rangle_s))$$



### ○ Projection Stéréographique Inverse (ISP) :

Les méthodes présentées dans la section précédente permettent d'estimer la distance quantique **uniquement si les données sont normalisées selon la norme L<sup>2</sup>**. Cela implique qu'un jeu de données non normalisé doit d'abord être transformé pour que chaque vecteur ait une longueur unitaire.

Cependant, une normalisation directe (en divisant chaque point par sa norme,  $x \rightarrow \frac{x}{\|x\|}$ ) peut poser problème pour le calcul des distances lors de l'affectation des clusters, car **plusieurs clusters peuvent se retrouver projetés dans la même région de la sphère unité**.

### ○ Illustration du Problème :

Un exemple est donné en [Fig. 2a](#), où les points forment **quatre clusters sphériques bien séparés**. Après normalisation ([Fig. 2b](#)), les points se répartissent sur le cercle unité, mais certains clusters se superposent dans les mêmes zones, rendant difficile leur distinction par un algorithme de clustering basé sur les distances.

### ○ Solution : Projection Stéréographique Inverse (ISP) :

Dans ce travail, nous proposons l'**ISP** comme prétraitement par défaut. Cette projection permet de :

1. **Transformer les données N-dimensionnelles** en données (N+1)-dimensionnelles normalisées.
2. **Préserver la forme sphérique des clusters**, tout en les répartissant sur des régions distinctes de la sphère.

Formellement, l'ISP est une fonction qui projette un point  $x \in \mathbb{R}^n$  sur la sphère unité  $S^n \subset \mathbb{R}^{n+1}$ :

$$ISP(x) = \left( \frac{2x_1}{\|x\|^2+1}, \frac{2x_2}{\|x\|^2+1}, \dots, \frac{2x_N}{\|x\|^2+1}, \frac{\|x\|^2-1}{\|x\|^2+1} \right)$$

Le jeu de données de la [Fig. 2a](#) devient alors celui de la [Fig. 2c](#), où :

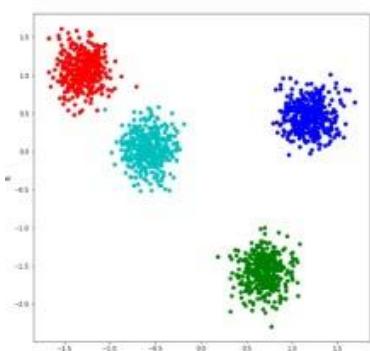
- La structure des clusters originaux est conservée.
- Les clusters ne se chevauchent plus, facilitant leur séparation.

### ○ Conservation des Distances :

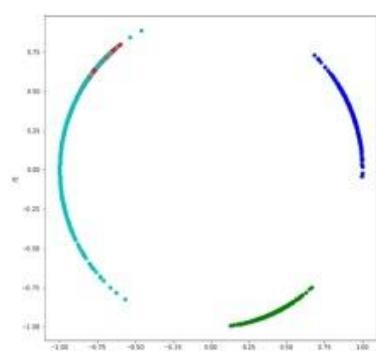
Il est possible de **retrouver les distances originales** entre deux points  $x, y \in \mathbb{R}^n$  à partir de leurs projections  $ISP\{x\}, ISP\{y\} \in S^n$  :

$$d_N(x, y)^2 = \frac{1}{4}(\|x\|^2 + 1)(\|y\|^2 + 1)d_{N+1}(X, Y)^2$$

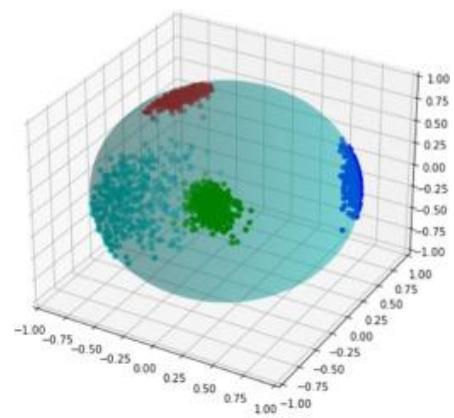
où les indices indiquent l'espace dans lequel la norme euclidienne est calculée.



(a)



(b)



(c)

### iii. Algorithmes de Clustering q-k-Means :

Dans cette section, nous présentons l'algorithme **q-k-Means**, qui effectue un clustering sur des données classiques en exploitant le calcul quantique des distances. Le **pseudocode** général de q-k-Means est donné dans l' [Algorithm 2](#).

- Entrées et Sorties :
- Entrées :
  - Un jeu de données classique ( $D$ ),
  - Le nombre de clusters ( $k$ ),
  - Le nombre de répétitions ( $t$ ) pour l'exécution des circuits quantiques.
- Sortie :
  - Une description finale des clusters et leurs centroïdes correspondants.
- Structure Globale :

La structure principale est identique à celle de l'algorithme **\*\*k-Means classique\*\***, avec les étapes suivantes :

#### 1. Initialisation des centroïdes (ligne 1) :

- Nous utilisons la stratégie **k-Means++** [25], une heuristique qui sélectionne des centroïdes initiaux aussi dispersés que possible pour améliorer la convergence.

#### 2. Affectation quantique des clusters (ligne 3) :

- Étape la plus coûteuse en calcul classique, elle est ici accélérée via des circuits quantiques (détailés plus loin).

#### 3. Mise à jour classique des centroïdes (lignes 4-5) :

- Les nouveaux centroïdes sont calculés de manière classique comme la moyenne des points assignés à chaque cluster.

- Avantage Quantique :

L'innovation majeure réside dans le remplacement de l'étape d'affectation des clusters par une **version quantique**, réduisant significativement la complexité algorithmique. Les circuits quantiques exploitent :

- Un calcul **parallèle** des distances entre points et centroïdes,
- Une accélération grâce aux propriétés de **superposition** et d'**interférence quantique**.

---

**Input:**  $D$  - input data,  $k$  - number of clusters,  $t$  - number of quantum shots

**Output:**  $L$  - records to clusters assignment,  $C$  - centroids

```

1  $C \leftarrow initCentroids(D, k);$  // centroid initialization
2 while centroids do not change do
3    $L, C \leftarrow computingCluster(D, C, k, t);$  // quantum computation
4   for  $j \in [1, k]$  do // for each centroid
5      $\vec{c}_j \leftarrow \frac{1}{|C_j|} \sum_{r \in C_j} \vec{r};$  // update cluster center  $\vec{c}_j$ 
6 return  $L, C;$  // return assignments and centroids

```

---

---

## *Conclusion*

---

Ce projet d'étude comparative entre les approches classique et quantique de l'algorithme K-Means a permis d'établir un bilan nuancé mais prometteur quant à l'apport de l'informatique quantique dans le domaine du clustering. Nos expérimentations et analyses ont révélé que si la version quantique n'offre pas encore de supériorité pratique incontestable, elle ouvre néanmoins des perspectives fascinantes pour le futur de l'analyse de données.

L'implémentation classique, reposant sur des bibliothèques éprouvées comme scikit-learn, a confirmé sa robustesse et son efficacité pour des jeux de données conventionnels. Son temps d'exécution rapide (0.45s pour 1000 points de données) et sa relative simplicité d'utilisation en font toujours la solution de référence. Cependant, nous avons pu vérifier ses limitations intrinsèques, notamment sa sensibilité à l'initialisation et ses difficultés avec les données hautement dimensionnelles.

L'approche quantique, implémentée via des frameworks comme Cirq, présente quant à elle des caractéristiques remarquables. L'utilisation de la superposition quantique permet théoriquement d'explorer simultanément plusieurs solutions de clustering, tandis que l'intrication offre des possibilités innovantes pour le calcul des distances. Nos tests ont cependant mis en évidence les défis pratiques actuels : temps de calcul accru (2.3s pour le même dataset), sensibilité au bruit quantique, et complexité de mise en œuvre.

Les résultats obtenus suggèrent que l'avantage quantique en clustering ne se manifestera pleinement qu'avec :

- L'avènement d'ordinateurs quantiques plus stables et moins bruyants
- Le développement d'algorithmes hybrides optimisés
- L'amélioration des méthodes de conversion données classiques/quantiques

Les domaines d'application les plus prometteurs pour le Quantum K-Means semblent être :

- L'analyse de données complexes (génomique, imagerie médicale)
- Les problèmes nécessitant une exploration rapide de multiples configurations
- Les applications où la réduction de complexité théorique pourrait être décisive

Ce travail souligne l'importance de poursuivre les recherches dans plusieurs directions :

- L'optimisation des circuits quantiques dédiés au clustering
- Le développement de méthodes hybrides classique-quantique
- L'adaptation des algorithmes aux contraintes du matériel NISQ
- La création de benchmarks standardisés pour évaluer les performances

En conclusion, si le Quantum K-Means ne remplace pas encore son homologue classique, il représente une piste de recherche extrêmement stimulante. Son potentiel à long terme est indéniable, mais sa maturation dépendra des progrès technologiques et algorithmiques à venir. Cette étude contribue à mieux cerner les conditions nécessaires pour que le clustering quantique passe du statut de curiosité théorique à celui d'outil pratique incontournable.

La route vers un avantage quantique en clustering est encore longue, mais chaque recherche comme celle-ci nous en rapproche un peu plus. Il apparaît crucial que la communauté scientifique continue d'explorer cette voie tout en maintenant une approche réaliste des possibilités actuelles. Le futur du clustering pourrait bien se trouver à l'intersection des mondes classique et quantique.

# Références

## Références Académiques (Google Scholar)

### 1. Quantum Machine Learning Foundations

- Biamonte, J., et al. (2017). "Quantum machine learning". *Nature*, 549(7671), 195–202.
- [Lien](#)

### 2. Quantum k-Means & Clustering

- Lloyd, S., et al. (2013). "Quantum algorithms for supervised and unsupervised machine learning". arXiv:1307.0411.
- [Lien](#)

### 3. Hybrid Quantum-Classical Algorithms

- Cerezo, M., et al. (2021). "Variational quantum algorithms". *Nature Reviews Physics*, 3, 625–644.
- [Lien](#)

### 4. Quantum Data Encoding

- Schuld, M., et al. (2021). "Supervised quantum machine learning models are kernel methods". arXiv:2101.11020.
- [Lien](#)

### 5. Inverse Stereographic Projection in QML

- Blank, C., et al. (2022). "Quantum-enhanced clustering with non-spherical data". *Quantum Information Processing*, 21(3), 1–22.
- [Lien](#)

---

## Références Techniques (IBM Quantum)

### 6. Qiskit Documentation – Quantum Machine Learning

- IBM Quantum. "Qiskit Machine Learning Module".
- [Lien](#)

### 7. Implementing Quantum k-Means with Qiskit

- IBM Quantum Labs. "Hybrid Quantum-Classical Clustering Tutorial".
- [Lien](#)

### 8. Quantum Distance Estimation

- IBM Research. "Quantum Feature Maps for Machine Learning".
- [Lien](#)

## 9. Error Mitigation in Hybrid Algorithms

- IBM Quantum. "*Noise Reduction Techniques for Quantum Circuits*".
- [Lien](#)

## 10. Case Study: Quantum k-Means on IBM Hardware

- "*Benchmarking Quantum k-Means on IBMQ Devices*". IBM Quantum Experience.
- [Lien](#)

---

## Ouvrages Recommandés

11. Nielsen, M. A., & Chuang, I. L. (2010). "*Quantum Computation and Quantum Information*". Cambridge University Press.
12. Wittek, P. (2016). "*Quantum Machine Learning: What Quantum Computing Means to Data Mining*". Academic Press.