Algorithme pour l'Investigation Numérique des Régimes Dynamiques du Modèle de Lorenz pour une Particule Active à Mémoire (F=0)

Objectif Global :
Explorer numériquement le comportement dynamique du système d'équations différentielles suivant :

```
Ẋ = Y - X
Ẏ = -(1/τ)Y + XZ
Ż = R - (1/τ)Z - XY
```

en étudiant l'influence du paramètre `R` (amplitude de l'onde) sur la vitesse moyenne asymptotique `<X>`, la sensibilité aux conditions initiales, et la multistabilité. Les résultats seront présentés sous forme de diagrammes de bifurcation, histogrammes, et visualisations d'attracteurs.

Étape 1 : Initialisation des Paramètres

- **Description :** Définition des paramètres fixes et variables du système pour l'ensemble des simulations.
- **Paramètres Fixes :**
  - **Taux de décroissance de l'onde (`τ`) :** `τ = 10`
    - *Justification :* Permet d'explorer la Région IV de la Fig. 3a de [1], zone d'intérêt pour la multistabilité et les oscillations avec dérive nette.
  - **Pas de temps (`dt`) :** `dt = 0.01`
    - *Justification :* Assure une précision suffisante dans les régimes potentiellement chaotiques. Ce choix devra être validé par des tests de convergence (voir Étape 8).
  - **Durée totale de simulation (`T_simulation`) :** `5000` unités de temps
    - *Justification :* Permet au système d'atteindre son régime asymptotique de manière fiable, conformément à [1].
  - **Durée transitoire (`T_transient`) :** `2500` unités de temps (`T_simulation / 2`)
    - *Justification :* Élimination des effets initiaux pour l'analyse des propriétés asymptotiques, suivant la méthodologie de [1].
  - **Nombre de conditions initiales par R (`N_IC`) :** `50`
    - *Justification :* Nombre suffisant pour évaluer la sensibilité aux CI et détecter la multistabilité, en accord avec la Fig. 4 de [1].
  - **Intervalle pour `X(0)` :** `[-5, 5]`
    - *Justification :* Couvre une plage représentative des vitesses initiales, basée sur [1].
- **Paramètre Variable :**

- **Amplitude de l'onde (`R`) :** Intervalle `[0.5, 3.0]` avec `N_R = 100` points.
    - *Justification :* Couvre les régions dynamiques I à IV identifiées dans la Fig. 3a de [1]. L'incrément (`≈ 0.025`) permet une bonne résolution pour observer les transitions.

Étape 2 : Génération des Conditions Initiales (CI)

- **Description :** Pour chaque valeur de `R` à étudier, générer un ensemble de `N_IC` conditions initiales distinctes.
- **Procédure :**
    1. Fixer `Y(0)_j = 0` pour toutes les conditions initiales (`j = 1` à `N_IC`).
    2. Fixer `Z(0)_j = 0` pour toutes les conditions initiales (`j = 1` à `N_IC`).
    3. Générer les `N_IC = 50` valeurs de `X(0)_j` linéairement espacées dans l'intervalle `[-5, 5]` à l'aide de la formule : `X(0)_j = -5 + (10 / (N_IC - 1)) * (j - 1)` pour `j = 1, 2, ..., 50` .
- **Note :** La position initiale `x_d(0)` n'intervient pas dans la dynamique de `X, Y, Z` pour `F=0` et n'est donc pas requise comme condition initiale pour ces équations.
- **Justification :** Cette méthode systématique et reproductible suit celle utilisée dans [1] (Fig. 4), permettant une comparaison directe et une exploration standard de la sensibilité aux CI et de la multistabilité.

Étape 3 : Intégration Numérique avec Runge-Kutta 4 (RK4)

- **Description :** Résoudre numériquement le système d'équations différentielles pour chaque couple `(R, CI_j)` .
- **Procédure pour chaque simulation :**
    1. **Initialisation :** Définir l'état initial du système `(X, Y, Z)` à `(X(0)_j, 0, 0)` et le temps `t = 0` .
    2. **Boucle temporelle :** Itérer de `t = 0` jusqu'à `t = T_simulation` par pas de `dt` . À chaque pas :
        - Calculer les dérivées `(Ẋ, Ẏ, Ż)` à l'état `(X, Y, Z)` courant en utilisant les équations du système.
        - Appliquer une étape de l'algorithme RK4 pour calculer le nouvel état `(X, Y, Z)` au temps `t + dt` .
        - **Stockage / Calcul partiel :** Si `t > T_transient` , accumuler la valeur de `X` et incrémenter un compteur pour le calcul de la moyenne (voir optimisation ci-dessous).
        - Mettre à jour le temps : `t = t + dt` .

- **Optimisation :** Pour économiser la mémoire, ne pas stocker l'intégralité de la trajectoire `X(t)`. Calculer directement la somme des valeurs de `X` et le nombre de points après `T_transient` au fur et à mesure de l'intégration.

Étape 4 : Calcul de la Moyenne Temporelle `<X>_j`

- **Description :** Calculer la vitesse moyenne asymptotique pour chaque simulation individuelle `(R, CI_j)`.
- **Procédure :**
  i. Utiliser la somme des `X` (`Somme_j`) accumulée pendant l'intégration pour `t` allant de `T_transient + dt` à `T_simulation`.
  ii. Calculer le nombre total de points temporels utilisés pour la moyenne : `N_points = (T_simulation - T_transient) / dt = 2500 / 0.01 = 250 000`.
  iii. Calculer la moyenne temporelle : `<X>_j = Somme_j / N_points`.
- **Résultat :** `<X>_j` représente la vitesse moyenne asymptotique pour la condition initiale `j` à la valeur de `R` donnée.

Étape 5 : Stockage des Résultats

- **Description :** Sauvegarder de manière organisée les résultats calculés pour chaque valeur de `R`.
- **Procédure pour chaque R :**
  i. **Collecter les résultats individuels :** Regrouper la liste des `N_IC = 50` moyennes temporelles : `[<X>_1, <X>_2, ..., <X>_50]`.
  ii. **Calculer les statistiques d'ensemble :**
    - Moyenne d'ensemble : `<X>_R = (1 / N_IC) * Σ (<X>_j)` pour `j` de 1 à `N_IC`.
    - Écart-type d'ensemble : `σ_X(R) = sqrt[ (1 / (N_IC - 1)) * Σ (<X>_j - <X>_R)^2 ]` pour `j` de 1 à `N_IC`.
  iii. **Sauvegarder :** Écrire la valeur de `R`, la liste complète des `<X>_j`, la moyenne `<X>_R`, et l'écart-type `σ_X(R)` dans un fichier de données structuré (par exemple, format CSV ou HDF5) pour analyse et visualisation ultérieures.

Étape 6 : Programme Principal (Structure Logique)

- **Description :** Séquence logique générale du code d'exécution.
- **Structure :**
    i. **Initialisation Globale :** Définir tous les paramètres (`τ` , `dt` , `T_simulation` , etc.) et générer la séquence des `N_R` valeurs de `R` à explorer. Ouvrir le fichier de sortie.
    ii. **Boucle Principale (sur `R`) :** Pour chaque valeur `R_i` dans la séquence :
        - Générer l'ensemble des `N_IC` conditions initiales `(X(0)_j, 0, 0)` .
        - Initialiser une liste vide pour stocker les `<X>_j` pour ce `R_i` .
        - **Boucle Interne (sur `CI_j`) :** Pour chaque condition initiale `j` de 1 à `N_IC` :
            - Exécuter l'intégration numérique (Étape 3).
            - Calculer la moyenne temporelle `<X>_j` (Étape 4).
            - Ajouter `<X>_j` à la liste des résultats pour `R_i` .
        - Calculer `<X>_R` et `σ_X(R)` à partir de la liste des `<X>_j` .
        - Écrire la ligne de résultats pour `R_i` (incluant `R_i` , tous les `<X>_j` , `<X>_R` , `σ_X(R)`) dans le fichier de sortie.
    iii. **Fin :** Fermer le fichier de sortie.

Étape 7 : Visualisation des Résultats

- **Description :** Utiliser les données sauvegardées pour générer des graphiques illustrant les résultats (par exemple, avec Python et Matplotlib/Seaborn).
- **Types de Visualisations :**
    i. **Diagramme de Bifurcation de la Vitesse Moyenne :**
        - *Objectif :* Montrer comment les états asymptotiques possibles (`<X>_j`) évoluent avec `R` et révéler la multistabilité.
        - *Méthode :* Tracer un nuage de points avec `R` en abscisse et toutes les `<X>_j` correspondantes en ordonnée.
    ii. **Courbes de Moyenne et d'Écart-Type d'Ensemble :**
        - *Objectif :* Fournir une vue synthétique du comportement moyen et de sa dispersion.
        - *Méthode :* Tracer `<X>_R` en fonction de `R` et `σ_X(R)` en fonction de `R` .
    iii. **Histogrammes de `<X>_j` :**
        - *Objectif :* Montrer la distribution des vitesses moyennes obtenues pour des valeurs spécifiques de `R` , illustrant la nature de la (multi)stabilité.
        - *Méthode :* Sélectionner quelques valeurs clés de `R` (représentatives des différents régimes) et tracer l'histogramme des 50 valeurs `<X>_j` associées.
    iv. **Visualisation des Attracteurs :**
        - *Objectif :* Comprendre la géométrie de l'espace des phases sous-jacente aux comportements observés.

- *Méthode :* Pour quelques couples `(R, CI_j)` intéressants, relancer la simulation en stockant la trajectoire `(X(t), Z(t))` après le transitoire, puis tracer la projection 2D de l'attracteur dans le plan `(X, Z)` .

Étape 8 : Détails Supplémentaires et Bonnes Pratiques

- **Tests de Convergence (`dt`) :**
  - *Action :* Avant de lancer la production complète, effectuer des simulations tests pour quelques valeurs de `R` (ex: une en régime stable, une en régime chaotique/multistable) avec des pas de temps plus petits (`dt/2` , `dt/10`).
  - *Critère :* Valider `dt = 0.01` si les valeurs de `<X>_j` obtenues ne varient pas significativement (ex: moins de 1%) par rapport à celles obtenues avec des pas plus fins.
- **Gestion de la Mémoire :**
  - *Action :* Implémenter le calcul de la somme des `X` et du nombre de points pour la moyenne directement pendant la boucle d'intégration (après `T_transient`), au lieu de stocker des tableaux `X(t)` très longs.
- **Parallélisation :**
  - *Opportunité :* Les `N_IC` simulations pour un `R` donné sont indépendantes. Utiliser des techniques de parallélisation (ex: OpenMP en Fortran, multiprocessing en Python si le solveur y est codé) pour distribuer le calcul de la boucle interne sur plusieurs cœurs de processeur et réduire le temps total d'exécution.