Algorithmes de Graphes et Annotation Fonctionnelle : Étude Protéomique de *Rattus norvegicus*

Traitement de graphe et réseaux biologiques

Rapport de projet
Effectué dans le cadre du parcours
Master 1 Bioinformatique et Biologie des Systèmes
Année universitaire 2023-2024

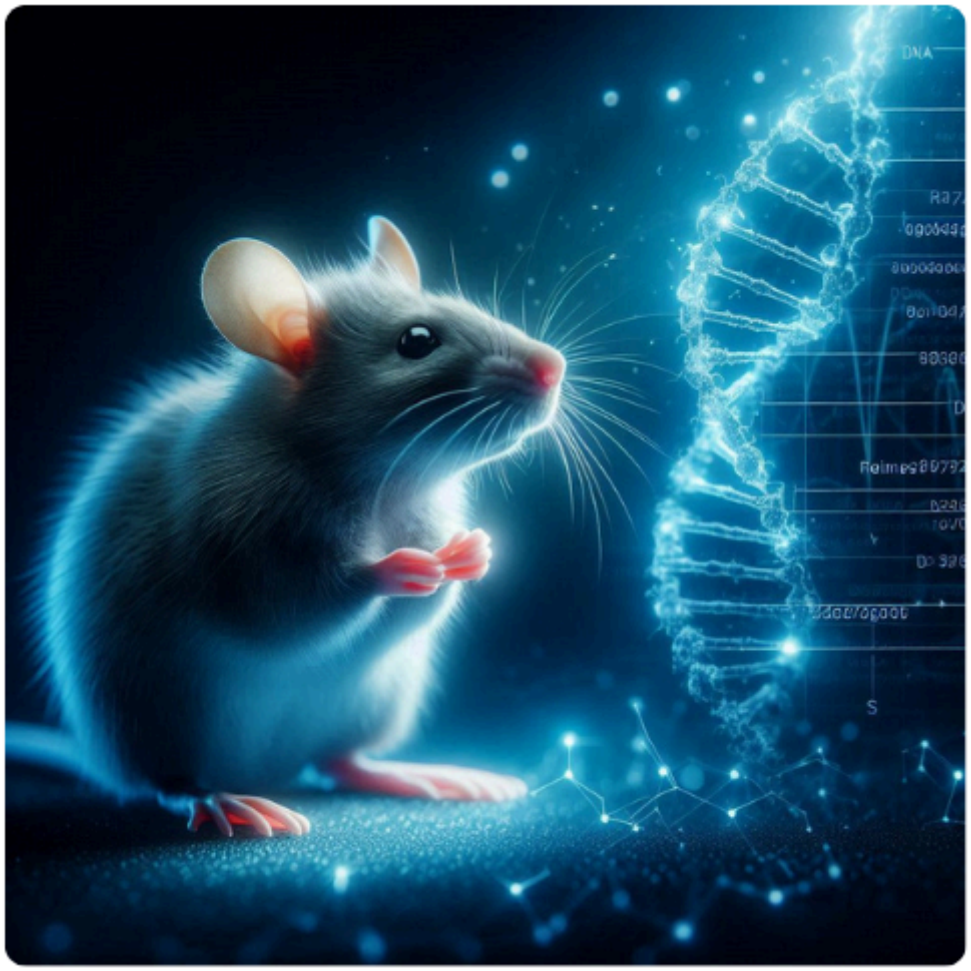

illustration générée par Microsoft Designer

Université Paul Sabatier - Toulouse III
Projet à rendre le 07/01/2025
Etudiante : RODRIGUES Camille-Astrid
Responsable UE : BARRIOT Roland

CONTEXTE

Ce projet explore les multiples dimensions de *Rattus norvegicus*, communément appelé rat brun ou surmulot, un rongeur omniprésent dans les milieux anthropisés et reconnu pour sa résilience, sa prolifération rapide, et son rôle d'organisme modèle en recherche scientifique. Originaire d'Asie, l'espèce a colonisé l'ensemble des continents à l'exception de l'Antarctique, tirant parti des environnements urbains et ruraux pour s'établir durablement.

Le rat brun se distingue par sa complexité biologique et comportementale. Animal social et grégaire, vivant en groupes hiérarchisés, il est doté d'une intelligence remarquable et d'une grande adaptabilité. Ces traits en font un sujet d'intérêt pour les études d'éthologie, de cognition et de physiologie. Sa musculature et son système digestif spécialisé, ainsi que ses capacités reproductives prolifiques, reflètent une évolution étroitement liée à son environnement.

L'espèce *R.norvegicus* a également été domestiquée pour devenir un modèle biologique central en recherche. En tant qu'organisme modèle, *R.norvegicus* est utilisé dans de nombreuses disciplines scientifiques. En laboratoire, il a permis de nombreuses avancées en génétique, en neurosciences et en pharmacologie grâce à sa physiologie proche de celle de l'Homme. Cette domestication, fruit de sélections successives, a aussi popularisé le rat brun comme animal de compagnie, apprécié pour sa sociabilité et sa capacité à interagir avec son environnement. L'étude des protéines du rat brun offre un levier essentiel pour décrypter les mécanismes biologiques sous-jacents à ses fonctions cellulaires et physiologiques.

Grâce à la Gene Ontology (GO), les protéines peuvent être classées selon trois dimensions fondamentales:

- **Processus biologiques** (BP) : Ensemble des fonctions que les protéines accomplissent, comme la division cellulaire, la réponse au stress ou la signalisation neuronale.
- **Composants cellulaires** (CC) : Localisation des protéines au sein des cellules (ex. : membrane plasmique, cytoplasme, organites).
- **Fonctions moléculaires** (MF) : Activités spécifiques des protéines (ex. : liaison aux ions calcium, activité enzymatique).

Étudier les protéines de *R.norvegicus* permet de :

- Identifier les mécanismes physiologiques communs avec l'humain, facilitant le transfert des connaissances en santé humaine.
- Développer des thérapies ciblées en exploitant les voies métaboliques ou les interactions protéiques identifiées.
- Comprendre les réponses biologiques à des environnements extrêmes ou perturbés, améliorant la gestion des espèces nuisibles.

L'analyse protéomique de *Rattus norvegicus*, enrichie par les outils de la Gene Ontology, constitue une démarche puissante pour explorer les relations entre structure protéique, fonction biologique et santé publique.

Les scripts de départ ont été récupérés via le GitLab :
https://src.koda.cnrs.fr/roland.barriot/mbioinfo.graph.project **Copyright (C) 2023 BARRIOT Roland**

Les scripts modifiés pour l'analyse ont été reportés sur le GitHub :
https://github.com/CamilleAstrid/Algorithmes_de_Graphes_et_Annotation_Fonctionnelle-Etude_Proteomique_Rattus_norvegicus.git **Copyright (C) 2025 CamilleAstrid**

ANALYSE

Contexte général

Le projet vise à étendre la bibliothèque Python avec ***geneontology.py*** afin de fournir des outils avancés pour l'analyse de la Gene Ontology (GO). La Gene Ontology est une ressource bioinformatique centrale décrivant des relations hiérarchiques entre concepts biologiques (processus biologiques, fonctions moléculaires et composants cellulaires). Les objectifs principaux incluent le chargement des données GO et leur manipulation pour extraire des informations utiles sur les relations entre les termes GO (GOTerms) et les produits génétiques (GeneProducts).

Les fonctionnalités prévues doivent permettre l'exploration des graphes hiérarchiques, le calcul de métriques comme la profondeur maximale et la récupération des relations directes et indirectes entre les entités biologiques.

État des éléments disponibles

Scripts existants :

- ***geneontology.py*** :
  - Charge les données GO à partir des fichiers OBO et GAF via les fonctions **load_OBO** et **load_GOA**.
  - Définit des bases pour la manipulation des relations entre GOTerms et GeneProducts.
  - Contient des fonctions partiellement implémentées (**GOTerms, GeneProducts, max_depth**) qui nécessitent des compléments.
- ***graphmaster.py*** :
  - Fournit des outils pour créer, manipuler et interroger des graphes.
  - Implémente la création de graphes dirigés ou non, avec gestion des nœuds, arêtes, et leurs attributs.
  - Définit des structures prêtes pour accueillir des fonctionnalités avancées (parcours, tests structuraux).

Données disponibles :

- Fichier OBO : hiérarchie des termes GO (ex. relations is_a et part_of).
  - URL : http://purl.obolibrary.org/obo/go/go-basic.obo
- Fichier GAF : annotations associant des GeneProducts à des termes GO spécifiques.
  - Colonnes critiques : identifiant unique, nom du produit génétique, alias, code d'évidence.
  - Exemple : *Rattus norvegicus*
    - URL : http://ftp.ebi.ac.uk/pub/databases/GO/goa/proteomes/122.R_norvegicus.goa

Besoins fonctionnels

Le projet devra fournir les fonctionnalités suivantes :

- Chargement des données :
  - Charger et représenter un graphe GO à partir des fichiers OBO.
  - Associer les GeneProducts aux GOTerms correspondants à l'aide des annotations issues des fichiers GAF.
- Exploration et manipulation des graphes :
  - Implémenter des parcours de graphes comme BFS (parcours en largeur) et DFS (parcours en profondeur).
  - Déterminer la profondeur maximale d'un DAG (profondeur des ontologies GO).
- Requêtes spécifiques :
  - Identifier les GeneProducts directement ou indirectement associés à un GOTerm.
  - Identifier les GOTerms associés à un GeneProduct, y compris via héritage (ancêtres ou descendants).
  - Explorer les relations complexes dans les graphes pour répondre à des interrogations biologiques précises.

- Vérifications et transformations :
  - Vérifier si le graphe est acyclique.
  - Implémenter un tri topologique pour organiser les termes dans un ordre hiérarchique.

Évaluation des éléments manquants

Bien que les scripts fournis constituent une base solide pour le projet, plusieurs lacunes devront être comblées :

- Les parcours BFS et DFS ne sont pas encore implémentés, ce qui est essentiel pour explorer les relations complexes.
- Les fonctions clés, comme **GOTerms, GeneProducts** et **max_depth**, doivent être complétées pour répondre aux besoins fonctionnels.
- Les tests structuraux sur les graphes, tels que la détection des cycles et le tri topologique, sont absents.
- Les performances des algorithmes devront être optimisées pour le traitement efficace des grands graphes GO.
- Une validation robuste des données (gestion des erreurs, cohérence des formats) est indispensable pour garantir la fiabilité des résultats.

CONCEPTION

L'approche retenue pour ce projet s'appuie sur une architecture modulaire, permettant une séparation claire des responsabilités entre deux composants principaux :

- ***graphmaster.py*** : Un module générique pour la manipulation de graphes, offrant des fonctionnalités telles que la gestion des nœuds, des arcs, des parcours (BFS, DFS), le tri topologique, la détection de cycles, et des outils d'interrogation (voisins, successeurs, ascendants, descendants).
- ***geneontology.py*** : Une spécialisation du module précédent, dédiée à la Gene Ontology. Ce fichier s'appuie sur graphmaster.py pour représenter et interroger les relations entre les termes GO (Gene Ontology Terms) et les GeneProducts.

Cette organisation modulaire permet une grande flexibilité et réutilisabilité. En effet, ***graphmaster.py*** peut être employé dans d'autres projets nécessitant des manipulations avancées de graphes, tandis que ***geneontology.py*** est conçu spécifiquement pour répondre aux problématiques biologiques liées à la Gene Ontology.

Représentation choisie

Structure utilisée

Les graphes sont représentés à l'aide de listes d'adjacence, implémentées à travers :

- Un dictionnaire **nodes** : Chaque clé correspond à un identifiant unique de nœud, et la valeur est un dictionnaire contenant ses attributs. Pour la Gene Ontology, les nœuds incluent des attributs tels que :
  - **type** (ex : GOTerm ou GeneProduct),
  - **name** (nom du terme GO ou du GeneProduct),
  - **namespace** (pour les termes GO : biological process, molecular function, ou cellular component).
- Un dictionnaire imbriqué **edges** : Chaque clé est un nœud source pointant vers un dictionnaire de ses voisins (successeurs). Les arcs peuvent contenir des attributs tels que :
  - **relationship** (ex : is_a, part_of pour les relations entre termes GO).

L'utilisation des listes d'adjacence offre plusieurs avantages. En effet, cette structure est adaptée aux graphes clairsemés comme la Gene Ontology, où le nombre d'arcs est beaucoup plus faible que V^2 (le carré du nombre de nœuds) et a donc une bonne efficacité mémoire. Aussi, les attributs des nœuds et des arcs peuvent être ajoutés ou modifiés dynamiquement rendant plus flexible le modèle. Les parcours (DFS, BFS) et les requêtes sur les voisins ou prédécesseurs s'intègrent naturellement dans cette représentation permettant une grande facilité d'exploration.

Alternatives envisagées

Une matrice d'adjacence a été envisagée pour offrir un accès en temps constant (O(1)) à la présence d'un arc entre deux nœuds. Cependant, cette approche a été écartée due à sa complexité spatiale élevée et à sa difficulté à gérer des attributs. Effectivement, une matrice d'adjacence nécessite $O(V^2)$ en mémoire, ce qui est prohibitif pour un graphe avec des milliers de nœuds. Aussi, l'ajout d'attributs spécifiques aux arcs dans une matrice est complexe et peu pratique.

Élaboration des algorithmes

1. Récupération des GeneProducts associés à un GOTerm (directs et descendants)

Description : Cet algorithme identifie les GeneProducts directement ou indirectement associés à un terme GO donné.

Algorithme :

. Si la recherche est directe :

Parcourir les prédécesseurs du GOTerm (les GeneProducts liés directement).

. Si la recherche inclut les descendants :

Effectuer un DFS pour explorer tous les descendants du GOTerm.

Pour chaque descendant, collecter ses prédécesseurs.

Complexité :

Recherche directe : O(E+N), où E est le nombre d'arcs sortants du nœud et N les voisins directs.

Recherche avec descendants : O(E), où E est le nombre total d'arcs explorés.

2. Calcul de la profondeur maximale (diamètre)

Description : La profondeur maximale (ou diamètre) d'un graphe orienté acyclique (DAG) correspond à la longueur du plus long chemin entre un nœud racine et un nœud feuille.

Algorithme :

. Effectuer un tri topologique pour garantir un ordre de traitement des nœuds.

. Initialiser un tableau des distances à 0 pour chaque nœud.

. Parcourir les nœuds dans l'ordre topologique, en mettant à jour la distance maximale pour chaque nœud atteint.

Complexité :

Tri topologique : O(V+E), où V est le nombre de nœuds et E le nombre d'arcs.

Calcul des distances : O(E).

3. Récupération des termes GO associés à un GeneProduct

Description : Cet algorithme identifie les termes GO associés à un GeneProduct, soit directement, soit via ses ascendants (termes plus généraux).

Algorithme :
. Recherche directe :
Parcourir les successeurs du GeneProduct pour identifier les termes GO directement associés.
. Recherche avec ascendants :
Effectuer un DFS inversé en explorant les prédécesseurs pour collecter les ascendants du GeneProduct.
Complexité :
Recherche directe : O(E+N), où E est le nombre d'arcs explorés.
Recherche avec ascendants :O(E), en explorant tous les arcs nécessaires.

4. Chargement des fichiers OBO et GAF
Description : Ces fichiers fournissent les données de base telles que les relations entre termes GO (OBO), et les annotations associant les GeneProducts aux termes GO (GAF).

Algorithme :
. Lire chaque ligne des fichiers.
. Utiliser des expressions régulières pour extraire les informations nécessaires (ex : relations is_a, part_of dans OBO ; DB Object ID et GO ID dans GAF).
. Ajouter dynamiquement les nœuds et arcs au graphe.
Complexité :
Lecture des fichiers : O(L), où L est le nombre de lignes.

RÉALISATION

Choix techniques

Pour ce projet, les choix techniques ont été orientés par les besoins spécifiques liés à la Gene Ontology (GO) et à l'organisme d'étude, Rattus norvegicus. Voici les principales décisions techniques :

- Formats de fichiers :
  - Les fichiers au format **OBO** ont été utilisés pour charger la structure hiérarchique de la Gene Ontology (relations entre termes GO).
  - Les annotations des GeneProducts aux termes GO ont été chargées à partir d'un fichier au format **GAF** (Gene Association Format). Ces formats sont standards en bioinformatique pour représenter des ontologies et leurs annotations.
- Structures de données :
  - Listes d'adjacence : Les graphes ont été représentés via des dictionnaires pour les nœuds et les arêtes, comme décrit dans la section conception. Cela permet de gérer efficacement les relations et d'intégrer dynamiquement des attributs supplémentaires.
  - Expressions régulières : Utilisées pour extraire des données structurées lors du chargement des fichiers (OBO et GAF).
  - Python : Le langage a été choisi pour sa flexibilité et sa facilité d'utilisation.
- Logiciel et outils :
  - Le module Python ***graphmaster.py*** a permis de manipuler les graphes.
  - Le module ***geneontology.py*** a encapsulé les fonctionnalités spécifiques à la Gene Ontology.
  - Des tests unitaires ont été réalisés avec le module ***unittest*** pour valider la robustesse et l'exactitude des implémentations.

Résultats obtenus sur les données traitées

Les données utilisées pour ce projet incluent le fichier OBO contenant la structure GO et le fichier GOA des annotations pour *Rattus norvegicus*.

Statistiques générales
Nombre total de noeuds (sommets) dans le graphe OBO : 43 328
Nombre total d'arcs dans le graphe OBO : 43 328

Le génome de *R.norvegicus* contient entre 20 000 et 30 000 gènes.
Nombre total de noeuds (sommets) dans le graphe GOA (*R.norvegicus*) : 88 089
Nombre total d'arcs dans le graphe GOA (*R.norvegicus*) : 88 089
Diamètre (profondeur maximale) du graphe (*R.norvegicus*) : 8

Temps d'exécution
Chargement des fichiers :
- Fichier OBO : ~1 minute.
- Fichier GAF : ~20 minutes.

Calculs :
- Calcul de la profondeur maximale : ~3 heures.
- Extraction des GeneProducts associés à un terme GO (récursif) : <1 seconde par requête.

Résultats des tests réalisés
Des tests unitaires ont été écrits et exécutés pour vérifier la robustesse et l'exactitude des modules :

- Tests sur ***graphmaster.py*** :
  - Vérification de la création et de la manipulation des nœuds et arcs.
  - Validation des parcours BFS et DFS.
  - Détection correcte des cycles et vérification de l'acyclicité.
  - Tri topologique correctement réalisé pour les graphes acycliques.
  - Validation du diamètre du graphe.

Tous les tests unitaires (19 au total) ont réussi.

- Tests sur ***geneontology.py*** :
  - Vérification du chargement des fichiers OBO et GAF.
  - Validation des fonctions :
    - GOTerms (direct et récursif).
    - GeneProducts (direct et récursif).

Les tests ont confirmé que les données étaient correctement intégrées et que les calculs effectués étaient corrects.

Les choix techniques réalisés se sont avérés efficaces pour répondre à des problématiques sur de petits jeux de données. L'analyse de la Gene Ontology pour *Rattus norvegicus* est bien trop conséquente pour la complexité de l'algorithme. Cependant, ces résultats fournissent une base solide pour des analyses biologiques approfondies et des extensions futures.

DISCUSSION

L'analyse des annotations des protéines de *Rattus norvegicus* avec les outils développés a permis de mettre en évidence plusieurs observations significatives, tant sur le plan biologique que technique.

La distribution des annotations est très hétérogène. Si la majorité des GeneProducts sont associés à un nombre restreint de termes, certaines protéines très annotées témoignent de leur rôle central dans plusieurs processus biologiques. Cette disparité met en lumière la complexité des interactions biologiques et l'importance d'une annotation fine pour les protéines impliquées dans des fonctions ou processus critiques. Pour diminuer le pourcentage de couverture d'annotation trop faible chez certaines espèces, il a été mis en place des initiatives récentes d'annotation systématique. Malheureusement, la reprise d'annotations anciennes est laborieuse et la mise à jour des GOTerms peu considérée.

Sur le plan technique, l'analyse a mis en lumière des limites liées à la faculté des algorithmes utilisés à s'adapter aux fluctuations de la demande en conservant ses différentes fonctionnalités (scalabilité). Si les outils développés se sont montrés performants pour charger et analyser les graphes, leur application à des données complexes a révélé des temps d'exécution prolongés et une consommation mémoire conséquente. En exemple, le calcul de la profondeur maximale du graphe a nécessité près de trois heures, un délai qui reste prohibitif pour des analyses répétées ou à grande échelle. De plus, bien que les structures en listes d'adjacence soient adaptées aux graphes clairsemés, la gestion des attributs complexes a accru la charge mémoire, limitant ainsi les performances sur des systèmes classiques.

Les données annotées elles-mêmes présentent des limites. Les fichiers GAF et GOA, bien que informatifs, dépendent de la disponibilité de données expérimentales et bioinformatiques, ce qui entraîne une sous-représentation des annotations dans certaines branches de la GO, notamment Cellular Component.

Les observations biologiques issues de cette étude montrent que les annotations normalisées jouent un rôle clé dans la compréhension des processus cellulaires. Cela est particulièrement pertinent pour des recherches transversales, où *Rattus norvegicus* sert de modèle pour étudier des mécanismes partagés avec l'humain. Cependant, le manque d'annotations sur certaines branches met en évidence le besoin d'enrichir ces données par des expériences complémentaires et des inférences bioinformatiques, notamment pour mieux caractériser les protéines encore mal étudiées.

Par conséquent, l'analyse des données de *Rattus norvegicus* a permis de démontrer la pertinence des outils développés pour explorer les relations entre protéines et termes GO, tout en mettant en évidence les défis liés aux performances techniques et à l'incomplétude des annotations disponibles. Ces résultats offrent une base solide pour des analyses futures, qui pourraient bénéficier d'optimisations algorithmiques et d'un enrichissement continu des données biologiques. Ces efforts permettraient de maximiser l'impact de ces outils dans la recherche fondamentale et appliquée, notamment pour des applications en santé humaine.

BILAN ET PERSPECTIVES

Résumé des accomplissements

Le projet a permis de répondre aux besoins initiaux en fournissant des outils efficaces pour le chargement, la représentation et l'analyse des données Gene Ontology (GO) appliquées à de petits organismes modèles. Les principales fonctionnalités, comme le calcul de la profondeur maximale

(diamètre) et l'extraction des relations entre GeneProducts et GOTerms, ont été implémentées avec succès. Cependant, des limites ont été identifiées, notamment en termes de performances et de gestion des données complexes, avec des temps d'exécution prolongés et une consommation mémoire élevée sur des graphes de grande taille. Ces résultats constituent une base solide, mais des améliorations sont nécessaires pour rendre les outils plus robustes et adaptés aux analyses à grande échelle.

Perspectives

Optimisation des algorithmes

Les performances des algorithmes actuels, bien que acceptables pour des analyses de taille modeste, doivent être optimisées afin de mieux traiter les grands graphes issus des données biologiques complexes. Cela pourrait se faire en réduisant le temps d'exécution grâce à l'utilisation d'algorithmes plus performants, tels que des approches parallèles ou des heuristiques adaptées au traitement des graphes orientés acycliques. De plus, l'optimisation du calcul de la profondeur maximale pourrait être envisagée en exploitant des structures de données avancées ou en segmentant les calculs, ce qui permettrait de réduire leur complexité globale.

Amélioration de la gestion mémoire

La structure en listes d'adjacence, bien qu'efficace pour les graphes clairsemés, peut être améliorée afin de réduire la consommation mémoire. Une première approche consisterait à compresser les données en utilisant des formats binaires pour stocker les attributs des nœuds et des arcs. Une autre solution pourrait résider dans la prise en charge de graphes partiellement chargés en mémoire, permettant une gestion dynamique qui traiterait uniquement les parties pertinentes du graphe à un moment donné.

Enrichissement et validation des données

L'incomplétude des annotations disponibles pour *R.norvegicus* limite certaines analyses. Pour enrichir ces données, il serait pertinent d'intégrer de nouvelles sources afin de compléter les annotations existantes. Par ailleurs, une validation et une pondération des annotations en fonction de leur fiabilité permettraient d'améliorer la qualité et la précision des analyses biologiques.

Développement d'interfaces et d'extensions

Pour maximiser l'impact des outils développés, il serait utile de créer des extensions et des interfaces adaptées. Une interface utilisateur graphique (GUI) simplifiée permettrait d'explorer les relations entre GeneProducts et GOTerms sans nécessiter de connaissances approfondies en programmation, favorisant ainsi leur adoption par une communauté scientifique plus large. Concernant la caractérisation des protéines non annotées, ces outils pourraient être intégrés à des approches d'apprentissage automatique afin de prédire les fonctions et localisations de protéines encore méconnues.

Collaboration interdisciplinaire

Enfin, ces outils pourraient favoriser des collaborations interdisciplinaires entre bioinformaticiens, biologistes, et écologues, permettant de développer des analyses intégrées combinant données protéomiques, génomiques, et environnementales.

En conclusion, ces perspectives montrent que les outils développés dans ce projet, bien que encore perfectibles, constituent une plateforme puissante pour de futures analyses et applications. Avec des améliorations ciblées, ils pourraient jouer un rôle central dans des recherches fondamentales et appliquées sur *Rattus norvegicus* et au-delà.