Développements méthodologiques pour l'intégration de données multi-omiques représentées dans une base de données orientée graphe

Marie-Galadriel Brière

▶ **To cite this version:**

Marie-Galadriel Brière. Développements méthodologiques pour l'intégration de données multi-omiques représentées dans une base de données orientée graphe. Informatique [cs]. 2019. dumas-02967930

HAL Id: dumas-02967930

https://dumas.ccsd.cnrs.fr/dumas-02967930

Submitted on 22 Oct 2020

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

MINISTERE DE L'AGRICULTURE, DE L'AGROALIMENTAIRE ET DE LA FORET

ECOLE NATIONALE SUPERIEURE des SCIENCES AGRONOMIQUES de BORDEAUX AQUITAINE
1, cours du Général de Gaulle - CS 40201 – 33175 GRADIGNAN cedex

M E M O I R E de fin d'études

pour l'obtention du titre

d'Ingénieur de Bordeaux Sciences Agro

DÉVELOPPEMENTS MÉTHODOLOGIQUES POUR L'INTÉGRATION DE DONNÉES MULTI-OMIQUES REPRÉSENTÉES DANS UNE BASE DE DONNÉES ORIENTÉE GRAPHE.

BRIERE, Marie-Galadriel

Spécialisation : AgroTIC

Étude réalisée à : LaBRI – Laboratoire Bordelais de Recherche en Informatique,
Domaine universitaire, 351 cours de la Libération, 33405, Talence

- 2 0 1 9 -

The French Ministry of Agriculture, Agrifood and Forestry

NATIONAL **S**CHOOL of **A**GRICULTURAL **S**CIENCES and **E**NGINEERING, **B**ORDEAUX **A**QUITAINE
1, cours du Général de Gaulle - CS 40201 – 33175 GRADIGNAN cedex, France

MASTER THESIS

Submitted in fulfillment of the requirements for the degree of

Agricultural Engineer, Bordeaux Sciences Agro

METHODOLOGICAL DEVELOPMENTS FOR THE INTEGRATION OF MULTI-OMICS DATA REPRESENTED IN A GRAPH DATABASE.

BRIERE, Marie-Galadriel

Specialisation : AgroTIC

Study completed at : LaBRI – Laboratoire Bordelais de Recherche en Informatique,
Domaine universitaire, 351 cours de la Libération, 33405, Talence

- 2 0 1 9 -

Remerciements

Je souhaite tout d'abord remercier toute l'équipe pédagogique de la spécialisation AgroTIC, et en particulier Mme. Toulon, pour leur soutient dans l'accompagnement de mon projet professionnel et ma spécialisation en bioinformatique.

Je remercie également Patricia Thébault et Raluca Uricaru, qui m'ont fait confiance pour reprendre le projet NeOmics, et pour leur encadrement très bienveillant. Je leur suis également très reconnaissante de l'opportunité qu'elles m'ont offert de réaliser une thèse, en continuité de ce projet passionnant.

Merci à Elodie Darbo, pour ses conseils et son expertise dans l'étude des données de cancérologie, et qui était là chaque semaine pour discuter avec nous des avancées du projet.

Merci enfin à toute l'équipe du LaBRI pour leur accueil chaleureux et leur gentillesse durant ces 6 mois.

Table des matières

Figures

Tableaux

Liste des abréviations

BDD : Base de Données

GDB : Graph DataBase, ou Base de Données Orientée Graphes

miARN : micro ARN

SGBD : Système de Gestion de Base de Données

TCGA : The Cancer Genome Atlas

NGS : New Generation Sequencing, Séquençage Nouvelle Génération ou Séquençage à haut-débit en français

POC : Preuve de concept ou démonstration de faisabilité, de l'anglais *Proof Of Concept*

Glossaire

Annotations : Métadonnées acquises par la recherche concernant un objet biologique (fonction d'un gène, positions dans le génome, processus biologique dans lequel est impliquée une protéine, etc…).

Cluster : Groupement d'objets retourné par un algorithme de partitionnement de données (*clustering*).

Clustering : Partitionnement de données.

Cypher : Langage de requête spécifique à Neo4j permettant d'interroger le graphe.

Expression (génique) : Ensemble des processus aboutissant à un produit génique (ARN ou protéine) par lecture d'un gène (ADN). Un gène peut être considéré comme **sous-exprimé** ou **sur-exprimé** sous certaines conditions expérimentales si son degré d'expression est inférieur ou supérieur à sa valeur "normale".

Framework : Infrastructure logicielle.

Graphe : Ensemble de nœuds et d'arêtes.

Intégration de données : Solution permettant à un utilisateur de récupérer des données issues de différentes sources, de les combiner, de les analyser, de les manipuler et de les ré-analyser afin de créer de nouveaux ensembles de données (*Lapatas et al, 2015*).

Méthylation : Processus par lequel un groupement Méthyle est ajouté à l'ADN. La méthylation des gènes peut affecter l'expression génique.

Multi-omiques : Se dit d'une analyse qui exploite conjointement des données issues de différents omiques.

Neo4j : Système de gestion de base de données orienté graphe.

Omique : Néologisme référant aux différents champs de recherche en biologie (par exemple, génomique pour l'étude du génome, protéomique pour l'étude du protéome, …).

Séquençage : Méthode d'acquisition de données permettant de déterminer l'enchaînement des composants d'une macro-molécule (séquence d'acides nucléiques pour l'ADN, séquence d'acides ribonucléiques pour l'ARN, séquence d'acides aminés pour les protéines, …).

Single-omique : Se dit d'une analyse qui se base sur un seul type d'omique.

Introduction

Les récentes avancées en matière de séquençage et d'acquisition de données ont permis l'augmentation du volume et de la diversité des données collectées en biologie, ainsi que le développement de nouvelles potentialités, comme par exemple la médecine de précision. Ainsi, de plus en plus d'objets biologiques différents, classés par "omiques", sont susceptibles d'être mesurés et analysés.

Mais face à la grande hétérogénéité des données biologiques portant des informations différentes et complémentaires, les outils d'analyse et de prédiction existants ne sont pas toujours adaptés. Si l'étude individuelle de chaque type de données est très informative, leur analyse simultanée peut permettre de révéler de nouveaux motifs dans les données ou mettre en évidence une signature corrélée à un phénotype.

C'est pourquoi la question de l'intégration de données est une problématique de plus en plus étudiée par la recherche, notamment en bioinformatique. Nous définissons la notion d'intégration de données en tant que solution permettant à un utilisateur de récupérer des données issues de différentes sources, de les combiner, de les analyser, de les manipuler et de les ré-analyser afin de créer de nouveaux ensembles de données (selon *Lapatas et al, 2015*).

L'intégration de données hétérogènes est la problématique abordée durant mon stage au Laboratoire Bordelais de Recherche en Informatique (LaBRI), au sein de l'équipe Bench to Knowledge and Beyond, spécialisée dans la modélisation et la visualisation de systèmes complexes à l'aide de graphes. L'objectif est de produire une preuve de concept de NeOmics, un outil visant à permettre la manipulation et l'intégration de données issues de différents omiques.

Ce rapport de stage présente les différentes étapes de mon travail dans le cadre de ce projet NeOmics. La première partie décrit les diverses données exploitées en bioinformatique et développe les problématiques traitées durant ce stage. Dans un second temps, un rapide état de l'art des outils et méthodes de manipulation et d'intégration de données hétérogènes est dressé. La troisième partie explique les choix réalisés en termes de données, de question biologique et d'outils utilisés pour la preuve de concept de NeOmics. La quatrième partie présente la méthodologie d'intégration mise en place, et enfin la dernière partie de ce rapport décrit et commente les résultats obtenus.

1) Diversité des données et des méthodes en bioinformatique et le problème de l'intégration des données

1.1) Diversité des données en biologie

1.1.1) Données omiques

Grâce aux récentes avancées en matière d'acquisition de données en biologie, et en particulier au séquençage haut-débit (NGS), le volume et la diversité des données collectées ont largement augmenté depuis quelques années, bénéficiant à de nombreux champs de recherche en biologie. Ces différents champs de recherche sont appelés "omiques". Ce néologisme fait référence au suffixe utilisé pour nommer différents types de recherche sur des objets biologiques spécifiques : génomique pour l'étude de l'ADN et du génome, transcriptomique pour l'étude de l'ARN, protéomique pour l'étude des protéines, etc …

Chaque omique se concentre donc sur un objet biologique spécifique. Cependant, ces objets ne sont pas indépendants et interagissent entre eux. En ce sens, la figure 1 représente différents types d'omiques sous forme de réseau d'interaction multi-couche.

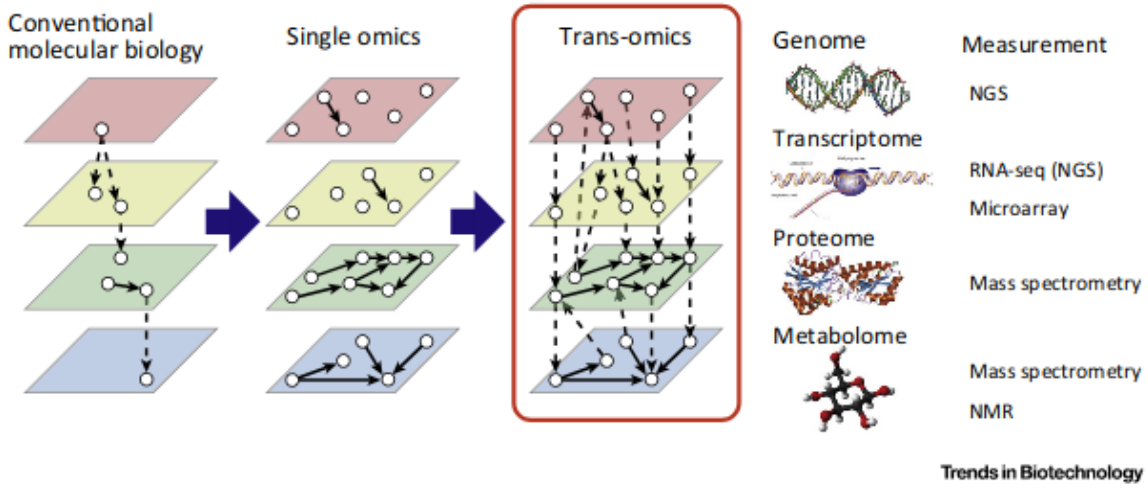


Figure 1 : Omiques et réseau d'interaction trans-omique (Yugi et al, 2016)

De nombreuses méthodes d'analyse existent pour exploiter chaque type d'omiques séparément, mais peu d'outils ont déjà été développés pour l'analyse conjointe de différents types de données. Pourtant, ce type d'analyse *multi-omiques* (par opposition à *single-omique*, analyse se concentrant sur un seul type de données) permettrait de prendre en considération les interactions ainsi que les informations complémentaires portées par chacun des omiques et améliorer les qualités prédictives des modèles biologiques *in silico*. Ainsi, analyser ensemble les différents omiques par des approches de fouille de données pourrait permettre de révéler des structures plus fines dans les données tout en améliorant leur interprétation biologique.

1.1.2) Métadonnées : conditions expérimentales, annotations

Outre les données omiques classiquement analysées par des approches de bioinformatique, de nombreuses informations supplémentaires sont disponibles grâce aux différentes bases de données (BDD) publiques (par exemple, celles du NCBI - National Center for Biotechnology Information, l'ENA - European Nucleotide Archive, ou encore TCGA - The Cancer Genome Atlas) qui accumulent les connaissances biologiques obtenues par la recherche.

Ces connaissances sont appelées annotations et peuvent décrire la fonction d'une protéine, par exemple ou encore la position d'un gène dans le génome, etc…

Les annotations sont diverses et si des efforts ont été faits pour mettre en place des modèles de données unifiés (par exemple, le consortium Gene Ontology (*Ashburner et al, 2000*) pour la description des gènes et des produits géniques), il reste difficile d'exploiter de manière optimale toutes les connaissances disponibles.

Il faut également prendre en compte les conditions expérimentales pour lesquelles les données omiques ont été acquises : tissus biologiques considérés, données cliniques des patients pour la recherche médicale, protocole d'acquisition des données, ...

Bilan de la partie 1.1 : Les données acquises en biologie sont hétérogènes et complémentaires : les omiques mesurent différents objets biologiques, les annotations sont une accumulation des connaissances biologiques acquises par la recherche et la comparaison de ces données selon différentes conditions expérimentales ouvrent de nouvelles voies pour la compréhension des relations entre le phénotype et le génotype.

L'exploitation conjointe des différents omiques (analyses multi-omiques) et des métadonnées est essentielle pour révéler des structures dans les données qui ne seraient pas détectées si chaque omique était analysé seul (analyses single-omiques), ainsi que pour améliorer la véracité biologique des modèles biologiques.

1.2) Analyses single-omique : méthodes et paramétrage

Les données omiques collectées doivent être analysées par différents outils bioinformatiques. Les analyses single-omiques sont les plus répandues. Il existe plusieurs types d'analyse en fonction du type d'omique considéré, et au sein d'un même type d'analyse, de nombreux outils implémentant différentes méthodes d'analyse. Cependant, le choix d'une méthode de prédiction est difficile, chaque approche ayant ses forces et ses limites (par exemple, *Audoux et al, 2017* ou *De Smet et al, 2010* ou *Gadgil, 2008*).

S'il est très intéressant d'avoir accès à différentes méthodes d'analyse, la comparaison de ces méthodes est difficile à réaliser (Kamali et al, 2015). Ainsi, la plupart des études scientifiques privilégient une seule méthode d'analyse en argumentant ce choix par des références prises dans la littérature (s'appuyant souvent sur une comparaison à partir d'un jeu de données particulier) plutôt qu'en se basant sur une réelle comparaison des méthodes disponibles appliquées à leurs propres données (Gardner et al, 2016).

Au sein d'une même méthode, le choix des paramètres d'exécution est déterminant et peut avoir un réel impact sur les résultats obtenus. Là encore, il est difficile de trouver le paramétrage adéquat, car il n'existe généralement pas de consensus permettant de déterminer les valeurs des paramètres en fonction des données étudiées. En pratique, une même méthode va être exécutée plusieurs fois avec différents paramètres, et un choix plus ou moins arbitraire des paramètres à appliquer va être effectué en fonction de ce qui est observé dans les résultats.

Bilan de la partie 1.2 : Du fait de la diversité des méthodes d'analyse existantes et des différentes manières de les paramétrer, nous sommes amenés à faire des choix plus ou moins arbitraires et à produire de nombreux jeux de résultats qu'il est difficile de comparer.

1.3) NeOmics, un nouvel outil exploitant les graphes pour l'intégration de données hétérogènes et de méthodes

1.3.1) Débuts du projet NeOmics et structure d'accueil

Face à la diversité des données et des méthodes, le développement d'un outil permettant d'intégrer des données de différents types et les résultats de différentes méthodes d'analyses serait bénéfique.

C'est dans ce contexte qu'a été lancé le projet NeOmics, pour le développement d'un outil destiné à l'intégration de données massives et hétérogènes et à l'intégration de méthodes. NeOmics est développé au Laboratoire Bordelais de Recherche en Informatique (LaBRI), dans l'équipe Bench to Knowledge and Beyond, spécialisée dans la modélisation et la visualisation de systèmes complexes à l'aide de graphes, et dans l'algorithmique pour la bioinformatique.

Le projet a débuté en 2018 par un stage réalisé par Ludovic Léauté, alors étudiant en master de bioinformatique, et encadré par Raluca Uricaru et Patricia Thébault. Au cours de son stage, Ludovic Léauté a créé une première preuve de concept (POC dans la suite de ce document, de l'anglais *Proof Of Concept*) de l'outil NeOmics. NeOmics exploite les graphes (ensemble de nœuds et d'arêtes) pour représenter des données hétérogènes et leurs interactions, pour permettre leur exploitation conjointe. Quelques résultats de ce premier POC seront présentés plus en détail dans la partie 2.1.

Au vu de la très grande diversité des données et des analyses en bioinformatique, il s'agit d'un projet ambitieux, et si un POC a déjà été réalisé, il ne couvre qu'une partie de la problématique globale. Il est donc poursuivi par le stage que je réalise actuellement, puis fera l'objet d'une thèse qui commencera en fin d'année 2019.

1.3.2) Sujet de stage et objectifs

Ce stage est la continuation du projet commencé en 2018, et son objectif est la création d'un POC pour l'intégration et la fouille de données multi-omiques représentées dans une base de données orientée graphes (GDB, pour *Graph DataBase*, dans la suite de ce document).

Nous définissons la notion d'intégration de données en tant que solution permettant à un utilisateur de récupérer des données issues de différentes sources, de les combiner, de les analyser, de les manipuler et de les réanalyser afin de créer de nouveaux ensembles de données (selon *Lapatas et al, 2015*).

Pour la réalisation de la première version de NeOmics, Ludovic Léauté s'est attaqué aux problématiques de représentation et manipulation de données très hétérogènes. Cependant, NeOmics ne permet actuellement pas de créer de nouveaux ensembles de données à partir des différentes informations stockées dans la BDD.

C'est pourquoi j'ai abordé durant ce stage la problématique de l'intégration des données par la combinaison et l'analyse conjointe de données hétérogènes. La solution NeOmics reposant sur la représentation des données sous forme de nœuds et d'arêtes, des algorithmes de théorie des graphes peuvent être envisagés pour analyser les données représentées dans la BDD.

Ici, la question est de développer une nouvelle méthode d'intégration de données hétérogènes. Nous avons donc dû sélectionner un sujet d'étude, une question biologique et des données à partir desquelles élaborer notre POC. Ces données doivent être suffisamment

génériques afin que les méthodes développées pour ce POC soient réutilisables pour différentes analyses et données. Les choix des données et de la question biologique traitées sont donc importants et devront être réfléchi.

Plusieurs livrables sont attendus : les scripts permettant d'analyser les données choisies, les scripts permettant de créer le graphe sur lequel se basera la solution NeOmics, les requêtes et les scripts permettant d'analyser le graphe et de retourner un résultat par intégration des données. J'ai également produit un poster sur mon travail, que j'ai présenté à la communauté scientifique durant les Journées Ouvertes de la Biologie, Informatique et Mathématiques (JOBIM) début juillet 2019 (voir Annexe 1).

Bilan de la partie 1.3 : Ce stage a pour objectif de produire un POC de NeOmics, un outil destiné à l'intégration de méthodes et de données hétérogènes issues de différents omiques. La solution proposée se basera sur l'exploitation des graphes. Le POC sera construit en réponse à une question biologique, mais sur des données suffisamment génériques pour convenir à différentes analyses et problématiques.

La suite de ce rapport présente les différentes étapes de mon stage. Dans un premier temps, je décris l'état de l'art sur les questions de représentation de données hétérogènes - notamment en présentant la première preuve de concept de NeOmics, réalisée par Ludovic Léauté en 2018 - ainsi que l'état de l'art concernant les outils d'intégration de méthodes.

Dans un deuxième temps, les choix réalisés pour le nouveau POC de NeOmics sont présentés : question biologique (détection de sous-type de cancer), données utilisées (expression génique, expression de micro-ARN et niveau de méthylation), et outils de prédiction multi-omiques existants et exploités par la méthode NeOmics.

Puis, nous verrons en détail la méthodologie utilisée par ce nouveau POC pour l'intégration de données hétérogènes et la détection de sous-type de cancer (signature). Cette méthodologie étant basée sur l'exploitation des graphes, les étapes de construction et d'analyse du graphe seront décrites.

Enfin, les résultats produits par NeOmics seront présentés et interprétés.

2) Intégration de données hétérogènes et de méthodes, un état de l'art

A l'ère du big data et du séquençage haut-débit, la problématique de l'intégration de données hétérogènes en bioinformatique est de plus en plus étudiée (*Dolinski et al, 2015*).

Dans cette partie, nous passerons en revue quelques outils existants pour l'intégration de données et/ou l'intégration de méthodes, à commencer par la première preuve de concept de NeOmics, réalisée par Ludovic Léauté durant son stage au LaBRI.

2.1) Graphes et données hétérogènes : NeOmics, une première preuve de concept (stage de Ludovic Léauté)

L'objectif du projet est de produire un outil d'intégration de données hétérogènes. NeOmics doit donc comprendre une partie base de données, de façon à pouvoir stocker et manipuler les données. Cependant, la grande variété des données rend impossible l'utilisation d'une BDD relationnelle classique. En effet, dans une BDD SQL classique, il faut nécessairement prévoir un modèle de données strict, et l'ajout d'un nouveau type de données implique un effort de maintenance important. Ainsi, les BDD relationnelles sont peu évolutives. L'utilisation d'une BDD relationnelle n'est donc pas adaptée à NeOmics, qui se veut capable d'intégrer facilement de nouveaux types de données.

Ludovic Léauté s'est donc intéressé aux BDD NoSQL (Not-only SQL), qui sont généralement plus évolutives et requièrent un modèle de données moins strict. Les différents omiques étant liés dans un réseau d'interaction complexe (voir Figure 1), l'utilisation d'une BDD NoSQL orientée graphes (Graph DataBase, GDB) semble naturelle.

En informatique, un graphe est formellement défini par un ensemble de nœuds et d'arêtes reliant ces nœuds. L'intérêt des bases de données orientées graphes est de pouvoir représenter des objets sous forme de nœuds et leurs interactions sous forme d'arêtes dans un schéma évolutif, où chaque nœud et arête peut comporter un ou plusieurs attributs.

L'utilisation des graphes permet également de s'affranchir des opérations de jointure coûteuses en ressource en se basant plutôt sur des méthodes de parcours de graphe, plus performantes. De plus, il est possible d'exploiter les différentes méthodes d'analyse de graphes qui pourraient s'avérer intéressantes pour détecter des motifs d'intérêt dans les données (plus court chemin entre deux nœuds, détection de communauté, etc…).

Ludovic Léauté a donc décidé de baser la solution NeOmics sur l'exploitation d'une GDB, et a choisi le Système de Gestion de Base de Données (SGBD dans la suite de ce document) Neo4j, SGBD open source développé par Neo Technology depuis 2000. Neo4j utilise un langage de requête spécifique, Cypher.

Pour son POC de NeOmics, Ludovic Léauté a mis en place un modèle de données permettant d'organiser les objets biologiques et métadonnées les plus utilisés en bioinformatique. Ce modèle de données est présenté dans la figure 2 ci-dessous.

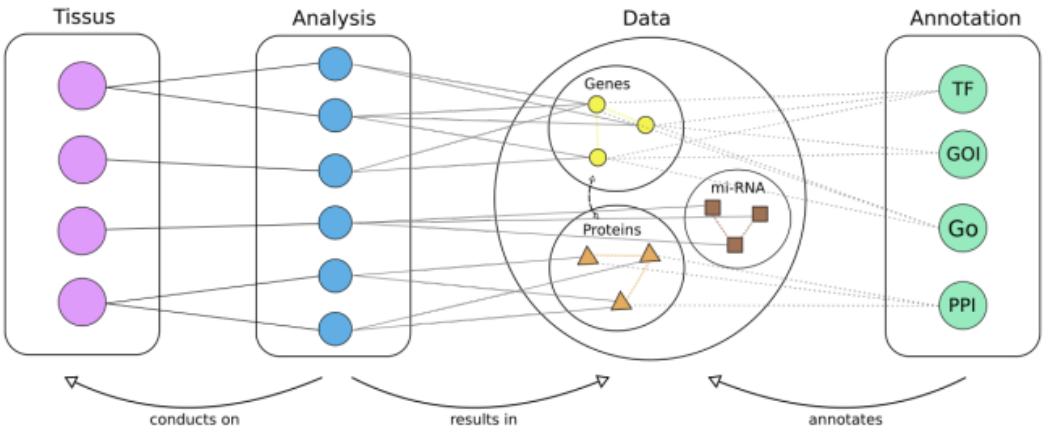


Figure 2 : Modèle de données utilisé pour le POC de NeOmics développé par Ludovic Léauté

Dans ce modèle de données, gènes, protéines et métabolites sont représentés en tant que nœuds et leurs interactions (annotations tirées des bases de données publiques) sous forme d'arêtes de différents types. Le modèle de données prend également en compte les données expérimentales et le type d'analyse dont sont issues les données.

Dans ce POC de NeOmics, Ludovic Léauté a notamment comparé l'expression de différents gènes (un gène pouvant être sous-exprimé DOWN ou sur-exprimé UP selon les conditions expérimentales) dans l'hippocampe de souris sous deux conditions expérimentales : régime déficient en oméga-3 (DEF) et régime équilibré (BAL). Une analyse Rank Product (RP) a été réalisée pour voir la différence d'expression de gènes par rapport à un échantillon de référence dans ces deux conditions expérimentales. Les résultats de cette analyse ont été stockés dans le graphe Neo4j et ont donc pu être requêtés afin de pouvoir comparer de manière plus visuelle la différence d'expression de gènes entre ces deux conditions. La figure 3 présente le graphe résultant d'une requête Cypher demandant l'ensemble des gènes présentant une différence d'expression (UP ou DOWN) pour les deux conditions expérimentales (DEF ou BAL).

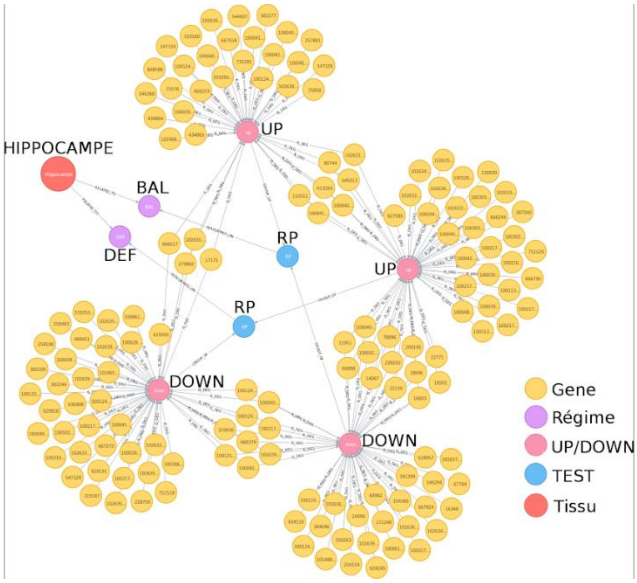


Figure 3 : Gènes différentiellement exprimés (analyse RankProduct) dans l'hippocampe sous deux conditions expérimentales (BAL, régime équilibré et DEF, régime déficient en oméga-3)

Ici, on peut rapidement comparer l'expression des gènes sous les deux conditions, et voir quels gènes présentent une différence d'expression (UP sous une condition expérimentale et DOWN dans l'autre).

Dans ce cas, on compare les résultats d'une méthode d'analyse (RankProduct RP) sous deux conditions expérimentales. Cependant, si on augmente le nombre de paramètres variants (on ajoute par exemple une nouvelle méthode d'analyse d'expression différentielle, ou un nouveau tissu), la taille du graphe augmente ainsi que sa complexité et il devient très difficilement interprétable "à la main".

On ne peut pas ici parler d'intégration de données au sens strict, car si les données sont manipulables grâce aux requêtes Cypher, il n'existe pas de méthode permettant de combiner les différents résultats pour produire un résultat consensus. Au-delà d'une comparaison 2 à 2, cette solution n'est plus adaptée.

Bilan de la partie 2.1 : L'utilisation d'une base de données orientée graphe telle que Neo4j a permis de représenter et de manipuler des données très hétérogènes. Cependant, si ce POC de NeOmics s'avère très informatif pour des comparaisons 2 à 2 de petits jeux de données, ajouter plus de facteurs de variation peut rendre le graphe illisible et non-interprétable. C'est pourquoi il est nécessaire de mettre en place une méthodologie d'intégration de données capable de générer un jeu de résultats interprétable à partir d'un graphe complexe et très connecté.

2.2) Intégration de méthodes : les méta-méthodes

Nous avons vu dans la partie 1.2 que la diversité des méthodes et la difficulté à les comparer rendait souvent nécessaire de produire différents jeux de résultats amenant à des conclusions différentes.

Une des solutions à cette problématique serait de produire des résultats consensus à partir des résultats de prédiction de différentes méthodes d'analyse. Ce genre de méthodologie d'intégration de méthodes est appelé méta-méthode, méta-approche ou approche consensus.

Les méta-méthodes se basent sur les résultats de différentes méthodes de prédiction pour produire un résultat plus complet et plus robuste. Pour intégrer les résultats de différents prédicteurs, les méta-approches capitalisent sur différentes stratégies : sélection des meilleurs résultats de chaque méthode (par exemple, *Xia et al, 2010*) ; sélection des résultats communs à différentes méthodes (par exemple, *Tsolis et al, 2013*) ; approches de machines *learning* - auquel cas on parle de méthode d'ensemble (par exemple, *Rapakoulia et al, 2014*) ; approches de *clustering* consensus (*Yu et al, 20017*) ; etc…

Les méta-méthodes sont de plus en plus utilisées pour l'analyse de données biologiques : prédiction de troubles protéiques (*Deng et al, 2009* ; *Ishida et al, 2008*), analyse d'enrichissement de gènes (*Väremo et al, 2013 ; Alhamdoosh et al, 2017*), prédiction d'interactions protéine-protéine (*Xia et al, 2010*), etc…

L'outil iPF (Integrative Phenotyping Framework, *Kim et al, 2015*) utilise une approche de *clustering* consensus pour intégrer des données de différents omiques, cependant, l'intégration n'est pas simultanée, la stratégie reposant sur une série de comparaison/fusion deux à deux.

Ainsi, si plusieurs outils capitalisant sur une stratégie de méta-méthode existent, ils sont souvent spécifiques à un type de données ou un type d'analyse et à une seule question biologique et ne permettent pas à un utilisateur de manipuler et fouiller ses données de manière simple et visuelle.

Bilan de la partie 2.2 : L'intégration de méthodes est une problématique importante en bioinformatique et permet d'obtenir un résultat consensus plus robuste à partir de différents outils de prédiction. Les outils existants sont spécialisés pour une question biologique ou un type de données, et ne permettent pas de mener différentes analyses. Ainsi, je n'ai identifié aucun outil permettant à un utilisateur de mettre en œuvre un raisonnement analytique en fonction des différentes hypothèses qu'il souhaite tester.

3) NeOmics, une nouvelle preuve de concept basée sur l'intégration de données multi-omiques pour la prédiction de sous-type de cancer

L'objectif du projet NeOmics est de produire un outil permettant d'intégrer des données hétérogènes, et notamment des données issues de divers omiques afin de prendre en compte, durant les analyses, la diversité et la complémentarité des informations issues de chaque type de données. NeOmics doit également être capable d'intégrer les résultats de divers outils de prédiction, afin de produire un résultat consensus plus robuste et/ou d'identifier les forces et faiblesses de ces outils.

Pour aborder cette large problématique, nous avons choisi de commencer par réaliser un nouveau POC de NeOmics, celui-ci devant être suffisamment générique pour pouvoir être par la suite adapté à différents types de données biologiques et différentes analyses.

Cette partie explique les choix réalisés pour constituer la base de ce POC :
→ choix du type d'analyse à réaliser,
→ question biologique et données à traiter,
→ méthodes existantes permettant de répondre à cette question biologique,
→ système de gestion de base de données le mieux adapté à notre problématique.

3.1) Le partitionnement de données, une méthode d'analyse sur laquelle se baser pour la preuve de concept de NeOmics

Le partitionnement de données, ou *clustering* en anglais, est une méthode d'analyse de données qui vise à diviser un jeu de données en différents groupes homogènes, de telle sorte que les objets classés au sein d'un même groupe (ou *cluster*) sont plus proches entre eux qu'avec les objets d'un autre cluster.

Les approches de clustering sont diverses et plusieurs mesures de distance peuvent être utilisées pour caractériser la similarité entre des objets.

Le clustering est utilisé par la communauté scientifique dans de très nombreux domaines. En bioinformatique, il peut par exemple servir à déterminer les gènes qui vont être affectés de la même manière par une maladie/une condition expérimentale (*gene expression clustering*), regrouper les patients affectés de la même manière par une maladie (*disease subtyping*), regrouper des gènes partageant des fonctions similaires (*functional gene clustering*), …

Si l'objectif de ce stage est de produire un nouveau POC de NeOmics et non pas l'outil final, il est important que les problématiques traitées par ce POC soient généralisables et donc suffisamment génériques. Etant donné la diversité des questions biologiques pouvant être traitées par des approches de clustering, nous avons choisi de baser notre POC sur le partitionnement de données hétérogènes.

Bilan de la partie 3.1 : Les algorithmes de *clustering* sont nombreux et utilisés dans différents domaines. Ainsi, il est intéressant d'aborder la question du partitionnement de données pour le POC de NeOmics afin que les solutions proposées soient génériques et applicables à diverses questions biologiques.

3.2) Données et question biologique : prédiction de sous-types de cancers

Un article paru en 2018 dans Nucleic Acids Research (*Rappoport et Shamir, 2018*) passe en revue et compare différents algorithmes de partitionnement de données multi-omiques. Les auteurs ont testé 9 outils d'intégration de données multi-omiques sur des données issues de trois omiques : l'expression de gènes, l'expression de micro-ARN (miARN), et la méthylation de l'ADN.

Le choix de ces 3 omiques est intéressant, car la méthylation et l'expression des miARN ont tous les deux un impact sur l'expression protéique (la méthylation impactant le processus de transcription et les miARN impactant le processus de traduction) et donc sur le phénotype d'un individu.

Pour tester et comparer les résultats des 9 méthodes d'intégration de données multi-omiques, Rappoport et Shamir ont travaillé sur la question biologique de la prédiction de sous-type de cancer. En effet, un même cancer peut se développer sous différents types moléculaires, selon les mutations et les gènes impactés. Le regroupement des tumeurs en fonction de leur profil d'expression génique par des méthodes de clustering permet la prédiction de ces sous-types de cancer, répondant chacun différemment aux traitements (*Desmedt et al, 2008*). L'étude des différents types moléculaires est ainsi cruciale pour permettre la mise en place de traitements mieux adaptés grâce à la médecine de précision.

L'idée est donc ici de considérer les mesures d'expression génique, d'expression de miARN et de méthylation des individus comme une signature, utilisée pour distinguer les différents sous-types de cancer. Les tumeurs sont classées dans différents groupes, ou clusters, grâce au partitionnement de données, chaque cluster étant une prédiction d'un sous-type moléculaire de cancer.

Rappoport et Shamir ont donc testé 9 méthodes d'intégration de données multi-omiques sur 10 cancers différents, avec pour chaque cancer entre 170 et 630 patients pour lesquels les 3 omiques considérés ont été mesurés. Les cancers étudiés sont : leucémie (AML), côlon (COAD), sein (BIC), poumons (LUSC), foie (LIHC), glioblastome (GBM), rein (KIRC), sarcome (SARC), ovaires (OV) et mélanome (SKCM).

Les données utilisées sont publiques et disponibles à l'adresse : http://acgt.cs.tau.ac.il/multi_omic_benchmark/download.html, originellement issues de la base de données The Cancer Genome Atlas (TCGA).

Le tableau 1 montre l'organisation des données d'expression génique. Quel que soit l'omique considéré, les échantillons sont présentés en colonne et les objets mesurés en ligne.

Tableau 1 : Données d'expression génique

	TCGA.AB.2803.03	TCGA.AB.2805.03	TCGA.AB.2806.03	TCGA.AB.2807.03
X..100130426	0	0.6693	0	2.006
X..100133144	0	1.3387	14.1884	9.3247
X..100134869	0	0	16.3654	13.5154
X..10357	92.5926	92.6305	119.599	178.4211

Les échantillons sont identifiés grâce à un code mis en place par TCGA qui est décrypté dans la figure 4 : le *barcode* permet d'identifier, entre autres, l'individu duquel l'échantillon provient ainsi que le type de tissu prélevé (tumeur primaire, métastase, etc…).

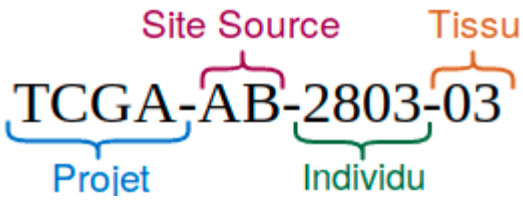


Figure 4 : Barcode d'un échantillon

Dans le tableau 1, c'est l'expression génique qui est mesurée : les identifiants dans la première colonne sont ceux des gènes. Pour les données de méthylation, les identifiants des lignes correspondent aux positions génomiques considérées et pour l'expression des miARN, l'identifiant des microARN considérés.

En génomique, un des problèmes majeurs pour l'analyse des données est la disproportion entre le nombre d'échantillons disponibles et le nombre d'observations. Selon le cancer et la taille de la cohorte étudiée, le nombre de gènes mesurés peut être jusqu'à 100 fois supérieur au nombre d'individus.

En plus de ces données omiques, nous avons accès aux données cliniques des patients : âge au diagnostic, stage pathologique, antécédents, décès, etc… Ces informations sont en partie renseignées par les médecins, et s'il existe une trame commune entre les différents cancers, un tri des données est nécessaire, ainsi que des connaissances cliniques pour les appréhender.

Nous avons choisi de baser le POC de NeOmics sur la détection de sous-type de cancer en suivant la publication de Rappoport et Shamir. En effet, la prédiction de sous-type de cancer se base sur une analyse de clustering, ce qui correspond au besoin de généricité présenté dans la partie 3.1. De plus, cela nous permet de nous affranchir :

→ Du choix des données, puisque celles utilisées par Rappoport et Shamir sont disponibles publiquement,

→ D'une lourde étape de bibliographie pour identifier, étudier et comparer les différentes méthodes de clustering multi-omique existantes.

Bilan de la partie 3.2 : La prédiction de sous-type moléculaire de cancer peut se faire par clustering des individus, le plus couramment en fonction de leur profil d'expression. Il est également possible d'utiliser plusieurs omiques. La prédiction de sous-type de cancer est la question biologique à laquelle nous allons tenter de répondre pour le POC de NeOmics, en étudiant 10 cancers différents à partir de données issues de 3 omiques : expression génique, expression de miARN et méthylation. Les données utilisées sont issues de la publication de Rappoport et Shamir, 2018.

3.3) Méthodes d'analyse de données multi-omiques

Dans leur publication, Rappoport et Shamir présentent un état de l'art des différentes méthodes de *clustering* de données multi-omiques existantes, et ils proposent une classification de ces méthodes dans 5 grandes catégories, récapitulées dans la figure 5.

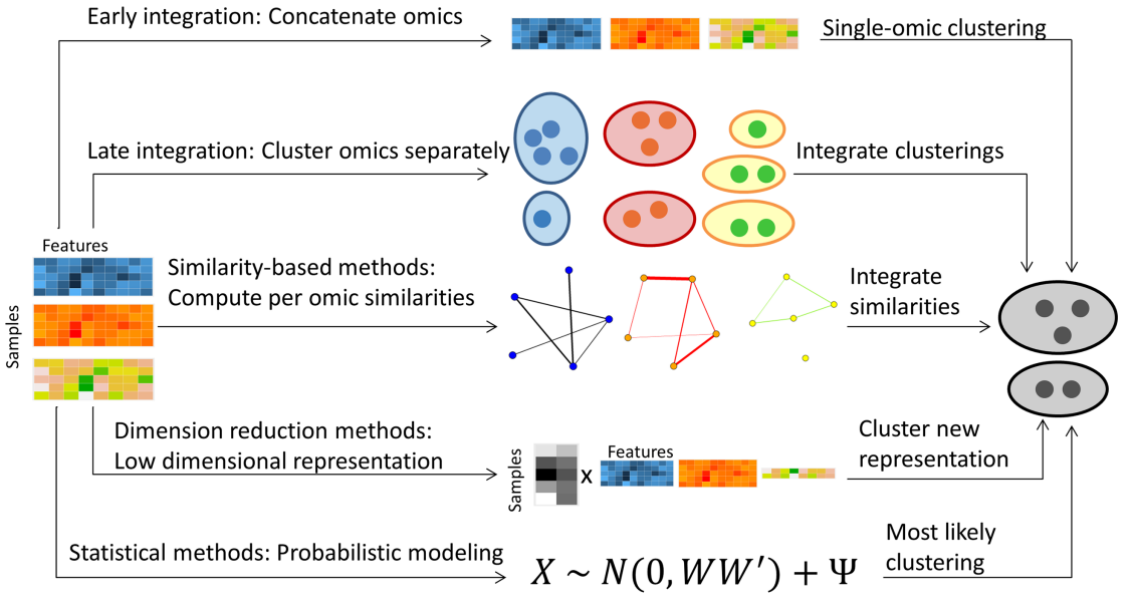


Figure 5 : Approches de clustering multi-omique
Source : *Rappoport et Shamir, 2018*

La première catégorie, "*Early Integration*" ou Intégration Précoce, concerne les outils reposant sur une stratégie de concaténation des différentes matrices omiques en une seule grande matrice, ensuite clusterisée d'un bloc par un algorithme de clustering single-omique. Ainsi, il est possible d'utiliser les algorithmes de clustering existants. Cependant, ce type d'approche présente l'inconvénient d'augmenter le nombre de dimensions (nombre d'observations) dans les données. De plus, elle peut donner plus de poids aux omiques présentant le plus grand nombre d'observations, si la matrice n'est pas normalisée au préalable.

Les méthodes dites "*Late Integration*", Intégration Tardive, commencent par clusteriser indépendamment chacun des omiques. Les clusterings de chaque omique sont alors intégrés pour produire un clustering multi-omique. L'intérêt de cette méthode est qu'elle laisse la possibilité d'utiliser des méthodes de clustering différentes sur chaque omique. Cependant, en n'utilisant que des solutions de clustering dans la phase d'intégration, il est possible de perdre des signaux qui sont faibles dans chaque omique séparément, mais potentiellement communs aux différents omiques.

Les méthodes basées sur les réseaux de similarité, calculent la similarité ou la distance entre chaque échantillon afin de partitionner les données. Pour chaque omique est construit un réseau de similarité. Ils sont ensuite fusionnés pour produire un réseau de similarité multi-omique, lequel sera alors analysé par clustering.

Certaines méthodes capitalisent plutôt sur la réduction de dimension des données, souvent grâce à la factorisation matricielle (analyse canonique des corrélations CCA, factorisation de matrices non négatives NMF, régression des moindres carrés partiels PLS, …). La matrice réduite, décrivant les principaux facteurs de variation observés dans les données est ensuite utilisée pour clusteriser les individus, en utilisant un algorithme de partitionnement single-omique.

Enfin, Rappoport et Shamir proposent la catégorie des méthodes statistiques : celles-ci modélisent la distribution probabiliste des données. Les échantillons sont classés dans des groupes définis par la distribution des données. L'avantage de ces méthodes est que la distribution peut être définie en utilisant les connaissances biologiques dans le modèle. De plus, il est possible, pour chaque échantillon, de calculer la probabilité de son appartenance à un cluster.

Rappoport et Shamir ont testé 9 méthodes de clustering multi-omiques issues de ces 5 grandes catégories d'approches. La comparaison de ces méthodes n'étant pas l'objectif du POC de NeOmics, nous avons sélectionné 4 méthodes à tester, avec le souci de représenter différentes familles de méthodes :

→ **PINS**, Perturbation clustering for data integration and disease subtyping (*Nguyen et al, 2017*) : classé dans la catégorie "Intégration Tardive", intègre des clustering single-omique en se basant sur une matrice de connectivité construite à partir des résultats des différents clusterings (Rand Index Ajusté élargi à plus de 2 clusterings). C'est cette matrice de connectivité qui est exploitée pour obtenir un clustering multi-omique. PINS est développé en langage R (package "*PINSPlus*").

→ **SNF**, Similarity Network Fusion (*Wang et al, 2014*) : classé dans la catégorie des algorithmes basés sur la similarité, SNF calcule des matrices de similarité et des matrices de similarité locale (*k* plus proches voisins) inter-individus pour chaque omique. Les différentes matrices de similarité et de similarité locale sont alors utilisées itérativement pour obtenir une matrice de similarité intégrée, "*fused network*". Cette matrice est alors partitionnée pour obtenir un clustering multi-omique. SNF est développé en langage R (package "*SNFtool*").

→ **rMKL**, regularized Multiple Kernel Learning (*Speicher et Pfeifer, 2015)* : classé dans la catégorie des méthodes basées sur la similarité, rMKL capitalise également sur la réduction de dimension des données. En effet, il effectue une réduction dimensionnelle sur les omiques d'entrée de telle sorte que les similarités (définies à l'aide de l'approche multi-kernel LPP, Locality Preserving Projections) entre chaque échantillon et ses voisins les plus proches soient maintenues en faible dimension. Cette nouvelle représentation est alors clusterisée. rMKL est développé en langage Matlab.

→ **MultiCCA**, Multiple Canonical Corelation Analysis (*Witten et Tibshirani, 2009*) : cet algorithme capitalisant sur la réduction de dimension, est une extension de l'analyse canonique classique, étendue à plus de deux jeux de données. Le principe repose sur la construction de combinaisons linéaires des variables des différents omiques de manière à

maximiser leur corrélation. Les variables canoniques alors calculées sont utilisées pour le clustering des individus. MultiCCA est développé en langage R (package "*PMA*").

En complément de ces 4 méthodes testées par Rappoport et Shamir dans leur publication, nous avons choisi d'ajouter une cinquième méthode à tester pour le POC de NeOmics : **NEMO**, Neighborhood based multi-omics clustering (*Rappoport et Shamir, 2019*). NEMO reprend la méthodologie de SNF, mais ajoute une méthode de gestion des données manquantes. Ainsi, les patients pour lesquels les 3 omiques n'ont pas été mesurés peuvent être pris en compte pour l'analyse, ce qui n'est pas le cas pour les autres méthodes sélectionnées.

Pour le POC de NeOmics, nous allons tester ces 5 méthodes, à la fois en single et multi-omique. Les résultats single-omique pourront être utilisés pour l'intégration des omiques, afin de produire un *clustering multi-omique NeOmics*, qui pourra être comparé aux résultats multi-omiques produits par chacune des méthodes.

Bilan de la partie 3.3 : Plusieurs méthodes d'intégration de données multi-omiques existent déjà. Nous en avons sélectionné 5 (une méthode Intégration Tardive, trois méthodes basées sur le calcul de la similarité et une méthode de réduction de dimension) pour le POC de NeOmics.

Ces méthodes vont d'abord être exécutées en single-omique et leurs résultats vont être intégrés par NeOmics pour produire un clustering multi-omique. Les 5 méthodes sélectionnées seront ensuite exécutées en multi-omique et les clusterings produits pourront être comparé à ceux de NeOmics.

3.4) Bases de données NoSQL : Neo4j

Comme présenté dans la partie 2.1, le principe de NeOmics est basé sur l'utilisation des graphes pour la représentation de données hétérogènes. Ce nouveau POC de NeOmics consacré à l'intégration de données multi-omiques exploite également l'organisation des données sous forme de graphe et l'utilisation du système de gestion de base de données dédié aux graphes, Neo4j.

L'intérêt de l'utilisation d'une BDD NoSQL par rapport à une BDD relationnelle classique réside principalement par la flexibilité du modèle de données. En effet, l'organisation en tables, clés primaires et étrangères des BDD relationnelles impose que le modèle de données soit bien défini et prévu pour toutes les manipulations de données envisagées. L'utilisation de BDD NoSQL présente beaucoup moins de contraintes en ce sens. Or NeOmics se doit d'être évolutif, puisque son objectif est de traiter des données hétérogènes de n'importe quel type.

Parmi les BDD NoSQL, les BDD orientées graphes permettent à la fois de représenter facilement et visuellement l'information, comme démontré dans la partie 2.1. De plus, les algorithmes de théorie des graphes sont un outil supplémentaire d'analyse des données, rendu disponible par cette organisation en nœuds et arêtes.

Enfin, les BDD graphes sont plus performantes pour le traitement de données fortement connectées car elles ne nécessitent pas d'opération de jointures coûteuses en ressources, comme c'est le cas pour les BDD relationnelles.

Nous avons donc décidé de poursuivre le développement de NeOmics avec Neo4j, l'un des premiers SGBD orientés graphes, mais aussi l'un des plus évolués et robustes. Ce SGBD en licence libre est capable de gérer une grosse volumétrie de données (ce qui est indispensable

pour les analyses multi-omiques), sans nécessité de construire un modèle de données cadré, et dispose de son propre langage de requête, Cypher. Ce SGBD propose également des implémentations des principaux algorithmes de théorie des graphes.

Neo4j stocke les données sous forme de nœuds et d'arêtes. Les nœuds représentent des entités et peuvent porter un ou plusieurs labels qui regroupent les entités du même type. Il est également possible d'ajouter des propriétés (ou attributs) à chaque nœud. Les nœuds, même au sein d'un même label, peuvent porter chacun des attributs différents.

Les arêtes représentent des relations entre les nœuds. Les arêtes possèdent obligatoirement un type de relation. Elles peuvent, comme les nœuds, être associées à des propriétés.

La figure 6 présente un graphe simple de Neo4j. Dans cette figure, les rectangles sont les nœuds (portant les labels "Person", ou "Movie"). Ils possèdent des propriétés ("name", "born", "title", "released"). Les relations sont de deux types : ACTED_IN, qui porte un attribut pour le rôle joué, et DIRECTED.

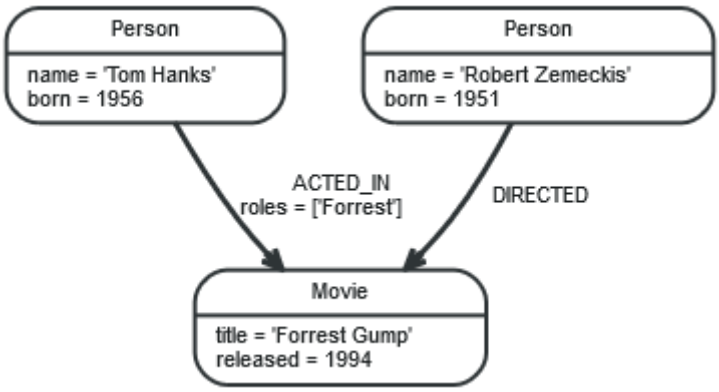


Figure 6 : Exemple de graphe avec Neo4j
Source : Documentation Neo4j.
Chapter 2. Graph database concepts - The Neo4j Getting Started Guide v3.5. Available at: https://neo4j.com/docs/getting-started/current/graphdb-concepts/#graphdb-example-graph (Accessed: 25 July 2019).

Pour construire, interroger et manipuler un tel graphe, il faut utiliser le langage de requête de Neo4j, Cypher. Ce langage est assez intuitif : il fonctionne avec un système de clause, pour déterminer l'action à effectuer et de motifs, pour indiquer les nœuds et arêtes à considérer.

Par exemple, pour créer le graphe de la figure 6, les requêtes sont les suivantes :

```
# Créer les nœuds "Person"
CREATE (tom:Person { name:"Tom Hanks", born:1956 })
CREATE (robert:Person { name:"Robert Zemeckis", born:1951 })
# Créer le nœud "Movie"
CREATE (forrestGump:Movie { title:"Forrest Gump",released:1994 })
# Créer la relation entre Tom Hanks et Forrest Gump
CREATE (tom)-[:ACTED_IN { roles: ["Forrest"]}]->(forrestGump)
# Créer la relation entre Robert Zemeckis et Forrest Gump
CREATE (robert)-[:DIRECTED]->(forrestGump)
```

Ou pour savoir dans quels films a joué Tom Hanks :

```
MATCH(:Person {name:"Tom Hanks"})-[:ACTED_IN]-(m:Movie) RETURN m
# Idem que :
MATCH (p:Person)-[:ACTED_IN]-(m:Movie) WHERE p.name = "Tom Hanks"
RETURN m
```

Il est possible d'exécuter des requêtes Cypher via Python ou encore R grâce à différents frameworks. Pour le POC de NeOmics, le graphe a été construit et interrogé depuis Python en utilisant le package Py2neo. Celui-ci propose des fonctions intégrées pour manipuler le graphe sans utiliser le langage Cypher, ainsi qu'une API pour interroger la BDD directement en Cypher.

Bilan de la partie 3.4 : Neo4j est un SGBD orienté graphe utilisant un langage de requête spécifique appelé Cypher et basé sur une utilisation de clauses (similaires à celles utilisées en SQL) et de motifs, représentant les nœuds et les arêtes à interroger. Neo4j est performant et propose divers algorithmes de théorie des graphes pour analyser les données. Les BDD Neo4j sont accessibles depuis Python grâce au package Py2neo.

4) Preuve de concept : analyses, modèle de données et méthodes d'intégration

Nous avons donc déterminé une question biologique qui est la détection de sous-type de cancer, ce qui passe par une approche de clustering : les patients sont regroupés selon leur sous-type de cancer en s'appuyant sur des données omiques. Pour réaliser cette étude, nous avons sélectionné des jeux de données portant sur 10 cancers différents et comportant de 170 à 630 patients pour lesquels trois omiques ont été mesurés : expression de gène, expression de miARN et méthylation.

Afin d'intégrer ces différentes données omiques, deux solutions sont envisagées : la première sera une méta-méthode se basant uniquement sur les regroupements multi-omiques construits par les outils d'intégration existants présentés précédemment ; la deuxième se basant sur des regroupements produits à partir de chaque omique individuellement, et intégrera chaque partition afin de produire un clustering multi-omique.

Dans cette partie, nous allons présenter la méthodologie utilisée pour le POC de NeOmics.

4.1) Construction du graphe

4.1.1) Exécution de 5 méthodes de prédiction de sous-types de cancer : choix et paramètres

NeOmics se veut être un outil d'intégration de méthodes et de données hétérogènes. Nous avons choisi de baser ce POC sur le clustering, puisque ce type d'analyse peut être utilisé pour répondre à de nombreuses questions biologiques.

Nous avons identifié, dans la partie 3.3, cinq outils d'intégration et de clustering de données multi-omiques : PINS, SNF, rMKL, MultiCCA et NEMO.

Afin de comparer la solution d'intégration de NeOmics à d'autres clustering multi-omiques produits avec les mêmes données, nous avons exécuté ces 5 méthodes sur nos jeux de données multi-omiques.

Nous avons également besoin de résultats de clustering obtenus par analyse de chaque omique séparément (clusterings single-omiques) que nous pourrons intégrer en un seul clustering multi-omique. Or, à l'exception de MultiCCA, qui ne peut être exécuté que sur des données multi-omiques, les outils d'intégration présentés précédemment peuvent également produire un clustering à partir d'un seul omique. Nous avons donc choisi d'utiliser PINS, SNF, rMKL et NEMO pour produire les résultats single-omiques à utiliser par NeOmics pour ses analyses. L'utilisation des mêmes outils nous permet de gagner du temps (identification, étude et exécution de nouvelles méthodes) et également de limiter les facteurs de variation dans un soucis d'interprétabilité des résultats. Cependant, n'importe quelle méthode de clustering single-omique pourrait être utilisée, y compris des méthodes spécialisées pour le clustering d'un omique spécifique.

La figure 7 présente les données intégrées au graphe de NeOmics. Le graphe, dont le modèle de données sera présenté dans la sous-partie suivante, est construit grâce à un script Python communiquant avec Neo4j grâce au package Py2neo. Dans ce graphe sont stockés les résultats de clustering single-omique (1 clustering par omique et par méthode testée), les résultats de clustering multi-omique (1 clustering par méthode testée) ainsi que les données cliniques disponibles pour chaque patient.

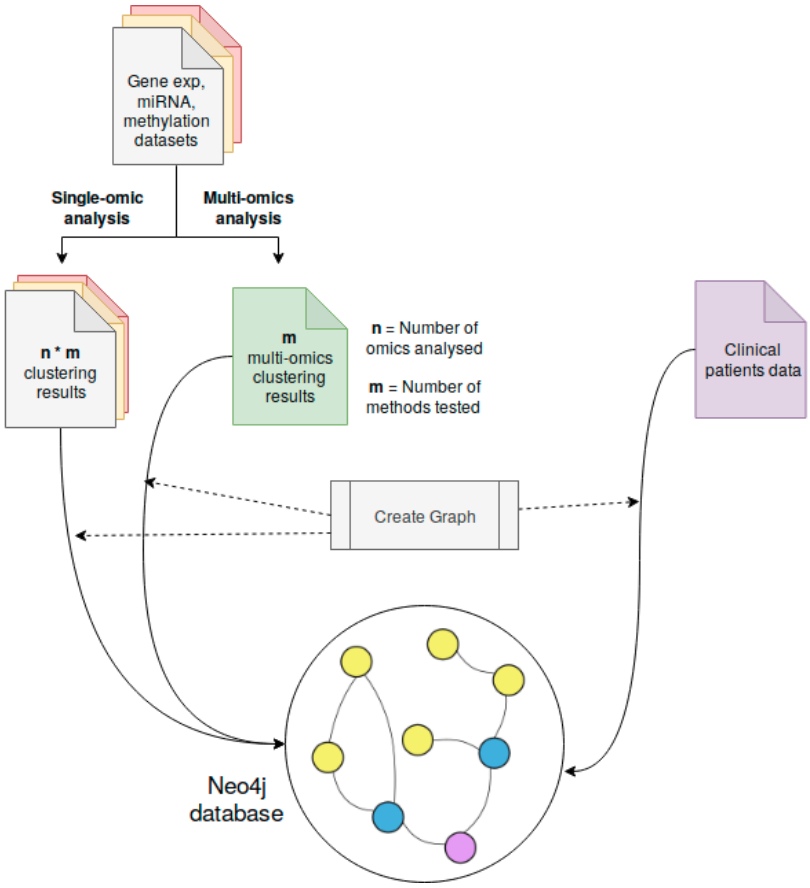


Figure 7 : Données intégrées au graphe NeOmics

Avant de pouvoir commencer les différentes analyses, les données ont été prétraitées, la première étape étant celle de la sélection des échantillons. En effet, les tumeurs et métastases sont deux tissus cancéreux qui ne partagent pas les mêmes caractéristiques (omiques ou physiologiques). Si l'analyse des différents types de tissus simultanément aurait été possible (et fait l'objet d'une option pouvant être spécifiée dans ce POC), nous avons décidé de travailler

uniquement sur un type de tissu par cancer, afin de faciliter l'interprétation biologique des résultats.

Ainsi, les analyses ont été menées uniquement sur les tumeurs primaires (code "01" dans le barcode de l'échantillon, voir figure 4) pour tous les cancers, sauf la leucémie pour lequel le tissu étudié est le sang (code "03") et le mélanome pour lequel le tissu métastatique a été sélectionné (code "06") puisqu'il est le plus présent dans les données.

Pour les analyses multi-omiques, sauf pour NEMO qui gère les données manquantes, nous avons sélectionné les patients pour lesquels les 3 omiques ont été séquencés. Pour les analyses single-omique, en revanche, tous les patients ont été analysés, sauf pour les cancers BIC et OV, méthode PINS, où seuls les patients mesurés pour les 3 omiques ont été analysés, du fait de problèmes d'exécutions dus à un manque de mémoire lorsque le programme était lancé avec les jeux de données complets.

Pour le prétraitement des données, et les paramètres d'exécution utilisés pour l'exécution des différentes méthodes, nous avons suivi les recommandations des auteurs ou utilisé les paramètres par défaut.

Tous les clusterings single et multi-omiques ont ainsi pu être générés, mis à part le clustering multi-omique de PINS pour le cancer du sein (BIC), à cause d'un manque de mémoire vive sur l'ordinateur de calcul. Les résultats sont disponibles sur Github (https://github.com/galadrielbriere/Neomics/tree/master/results/raw_results).

4.1.2) Modèle de données

Le modèle de données établi pour la représentation des résultats de clustering et des métadonnées est assez simple, et est présenté dans la figure 8. Nous distinguons 3 grand types de nœuds : les nœuds "Patient", le nœuds "Cluster" et les nœuds "Metadata" (métadonnées).

Chaque patient et chaque cluster retourné par les méthodes d'analyse sont stockés sous forme de nœuds. Les patients sont reliés par une relation "PART_OF" ("fait partie de") aux clusters dans lesquels ils ont été classés par les différentes méthodes de clustering. Pour rappel, les différents clusters obtenus correspondent à des sous-types de cancer potentiels.

Les nœuds Metadata représentent les métadonnées associées aux différents patients, et sont reliés par des relations spécifiques : par exemple, la relation "IS_PATHO_STAGE" permet de relier les patients à leur stade pathologique correspondant ("Stage IIA", "Stage IIIB", …) ou encore la relation "IS_PATHO_M" pour savoir si le patient a développé ou non des métastases ("MX", absence de données ; "M0", sans métastase, …).

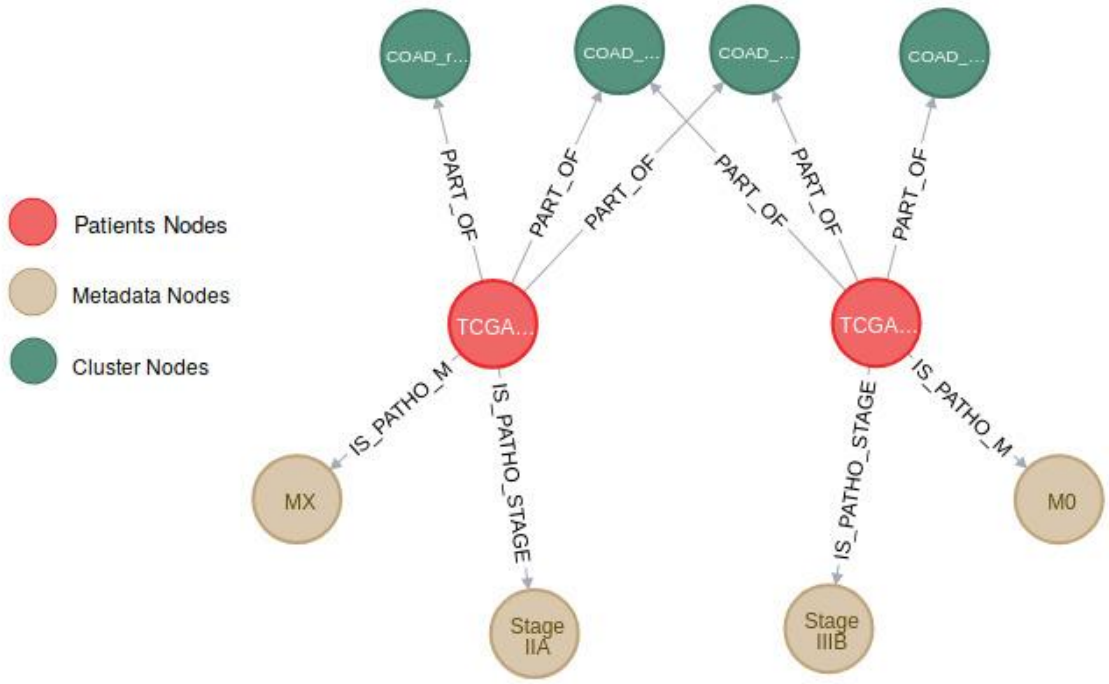


Figure 8 : Modèle de données simplifié

Des attributs ont été associés à chaque type de nœuds pour les distinguer. Les nœuds Patient possèdent un attribut "tcga_sample_id", qui est le même identifiant que celui utilisé dans les données. Les nœuds Cluster ont un attribut "clust_id", qui est un identifiant unique caractérisant le cancer sur lequel le *cluster* a été calculé, la méthode ayant produit le clustering, l'omique étudié ("exp", "mirna", "met" ou "multiomics") et le numéro du cluster. Par exemple, l'identifiant peut être "*COAD_PINS_mirna_cluster2*". Bien que cet attribut à lui seul récapitule les caractéristiques du clustering, ce n'est pas la meilleure manière de stocker ces informations, et il ne s'agit là que d'une manière de produire un identifiant unique et clair.

En effet, des informations comme le type d'omique ou la méthode utilisée pour produire un cluster sont communes à plusieurs nœuds et seront potentiellement souvent requêtées pour manipuler les données. Il faut donc stocker ces informations de telle sorte qu'elles soient facilement accessibles tant en termes de praticité pour écrire les requêtes Cypher qu'en termes de performances et de temps de calcul. Or, Neo4j doit parcourir les nœuds et relations pour accéder aux propriétés. Ce moyen de stocker l'information n'est donc pas le plus rapide.

Par exemple, on pourrait vouloir obtenir la liste des patients communs au cluster 1 de la méthode PINS exécutée pour l'omique 'expression de gènes' et au cluster 2 de la méthode SNF exécutée pour le même omique. Dans le cas où les informations sur l'omique et la méthode utilisés seraient stockés dans des attributs, la requête Cypher à exécuter serait la suivante :

```
MATCH (c1:Cluster {id:1, method: "PINS", omic: "expression"}) <-[:PART_OF]-
(p:Patient)->[:PART_OF]-(c2:Cluster {id:2, method: "SNF", omic:
"expression"})
RETURN p
```

Pour cette requête, Neo4j devrait parcourir tous les paires nœuds Cluster du graphe présentant au moins un nœud Patient en commun jusqu'à trouver les deux nœuds correspondants aux attributs spécifiés. Dans le cas de gros graphes, elle peut donc être coûteuse en ressources.

Une autre manière de stocker les informations sur les omiques et les méthodes est d'utiliser des nœuds spécifiques, comme l'a fait Ludovic Léauté pour le premier POC de

NeOmics (voir figure 2, nœud “RP”, méthode Rank Product, par exemple). Dans le cas où les informations sur les omiques ou sur la méthode étaient stockées via des nœuds, les nœuds Cluster seraient reliés aux nœuds Omic et Method correspondants. La requête Cypher pour interroger le graphe serait alors :

```
MATCH (:Omic {name: "expression"})<-[:FROM_OMIC]-(c1:Cluster {id: 1}) -
[:FROM_METHOD]->(:Method {name: "PINS"})
MATCH (:Omics {name:"expression"})<-[:FROM_OMIC]-(c2:Cluster {id: 2}) -
[:FROM_METHOD]->(:Method {name: "SNF"})
WITH c1, c2
MATCH (c1)<-[:PART_OF]-(p:Patient)-[:PART_OF]->(c2)
RETURN p
```

Dans ce cas, NeOmics parcourt les nœuds de type Omic et Method pour trouver les nœuds Cluster recherchés. Ce type de représentation complexifie grandement les requêtes.

Enfin, il est possible d’attribuer des labels aux nœuds. Par exemple, un nœud “Cluster” serait “étiqueté” avec les méthodes et omiques à partir duquel il a été produit. On aurait alors la requête suivante :

```
MATCH (:Cluster:Expression:PINS {id:1})<-[:PART_OF]-(p:Patient) -
[:PART_OF]->(:Cluster:Expression:SNF {id: 2})
RETURN p
```

Les labels étant automatiquement indexés par Neo4j, le SGBD trouve les nœuds correspondants sans avoir à parcourir le graphe. De plus, la requête Cypher reste simple et lisible.

J’ai donc choisi de stocker les informations communes à de nombreux nœuds et susceptibles d’être souvent requêtés sous forme de labels. Chaque nœud “Cluster” comporte donc un label pour l’omique utilisé et un label pour la méthode. Chaque nœud “Patient” possède un label “Tissue” pour indiquer le tissu duquel les données proviennent.

Les nœuds stockant les métadonnées sont de différents types selon le paramètre clinique stocké (“PathoStage” pour les nœuds représentant les différents stades pathologiques, par exemple), mais possèdent tous un label “Metadata”.

Bilan de la partie 4.1 : Plusieurs méthodes de clustering ont été exécutées, à la fois en single et multi-omique, aboutissant à de nombreux clusterings différents. Tous les résultats de clustering ont été stockés dans un graphe, où les patients sont représentés par des nœuds et reliés par une arête spécifique aux nœuds “Cluster” dans lesquels ils ont été classés. Chaque nœud “Cluster” est caractérisé avec des labels pour indiquer la méthode par lequel il a été produit ainsi que l’omique qui a été considéré. Des métadonnées cliniques sont également associées à chaque patient, stockées sous forme de nœud.

4.2) Analyse du graphe : méthodologie

A ce stade, nous avons donc obtenu de nombreux résultats de clustering : pour chaque cancer, un clustering single-omique par omique et par méthode (4 méthodes single-omique * 3 omiques = 12 clusterings différents), et un clustering multi-omique par méthode (5 méthodes multi-omiques = 5 clusterings différents), et nous avons stocké ces 17 résultats de clustering

dans un graphe. Nous voulons maintenant intégrer les différents résultats pour mettre en évidence les prédictions soutenues par plusieurs méthodes.

4.2.1) Création d'arêtes "SUPPORT" par méthode entre chaque paire de patients

Chacun des clusterings est différent dans la manière dont sont regroupés les patients ou même dans le nombre de clusters obtenus. Afin d'intégrer les différents résultats, nous pouvons déterminer quels patients ont été regroupés ensemble par les différentes méthodes de clustering et omiques. En effet, une des approches courantes des méta-méthodes est l'intersection des résultats de différentes méthodes de prédiction.

Si l'intersection simple des résultats serait possible pour deux clusterings présentant le même nombre de clusters, elle n'est pas adaptée à la comparaison de nombreux clusterings où le nombre de clusters varie.

Ainsi, j'ai plutôt mis en place deux nouveaux types d'arêtes :

→ **les arêtes SUPPORT,** qui permettent de récapituler pour chaque méthode les omiques ayant permis d'aboutir à la même prédiction. Ces arêtes seront présentées dans cette sous partie.

→ **les arêtes d'INTÉGRATION,** qui permettent de résumer les informations contenues dans l'ensemble des arêtes SUPPORT. Ce type d'arête sera présenté dans la sous-partie suivante (partie 4.2.2).

Les arêtes SUPPORT relient les patients ayant été groupés dans le même cluster pour une certaine méthode. Le principe de construction de ces nouveaux types d'arêtes est présenté dans la figure 9 ci-dessous.

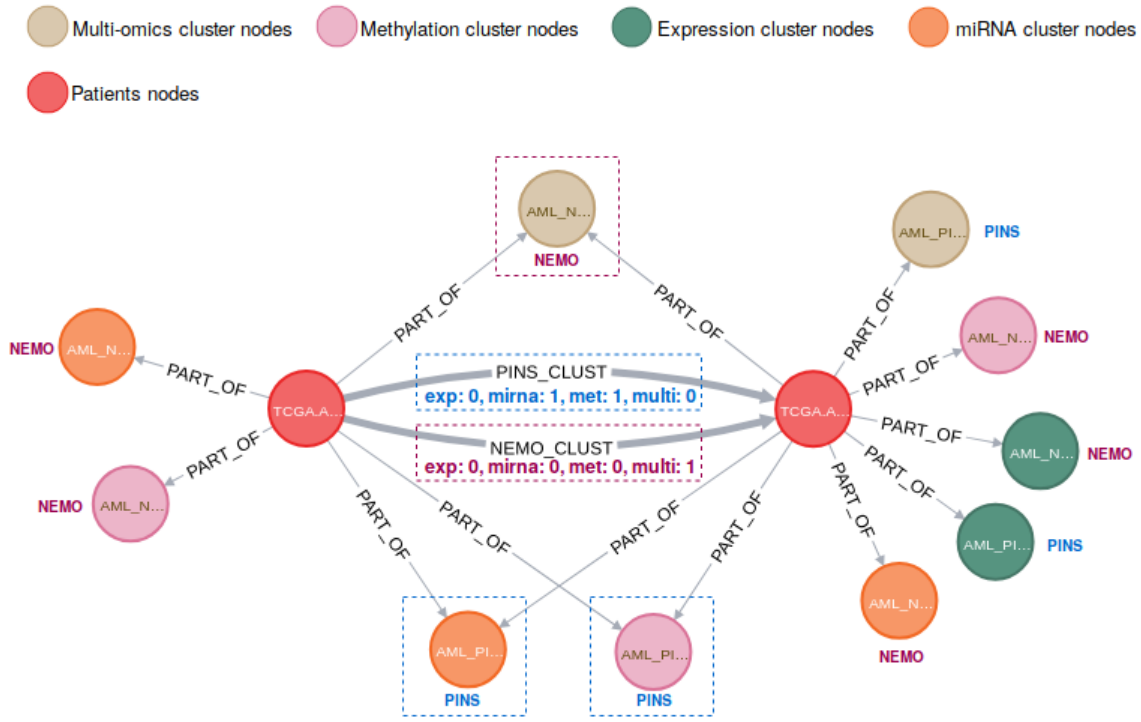


Figure 9 : Création d'arêtes SUPPORT entre les paires de patients classés ensemble dans au moins un résultat de clustering

Dans cette figure, les nœuds “Cluster” ont été colorés en fonction de leur label “omique” (en beige, les clusters multi-omiques, en rose les clusters expression de gènes, etc…). Le label concernant la méthode identifiée est spécifié à côté de chaque nœud “Cluster”. Ici, on ne s’intéresse qu’aux deux méthodes PINS et NEMO, dans un souci de clarté. Les patients, nœuds rouges, sont reliés aux clusters dans lesquels ils ont été classés grâce à la relation “PART_OF”.

On remarque que les deux patients considérés ont été groupés ensemble par :
→ La méthode PINS pour l’expression de miARN (nœud orange encadré en bleu)
→ La méthode PINS pour la méthylation (nœud rose encadré en bleu)
→ La méthode NEMO pour le multi-omiques (nœud beige encadré en bordeaux)

Cette paire de patients est donc supportée par 3 résultats de clustering, issus de deux méthodes différentes. Une arête “PINS_CLUST” et une arête “NEMO_CLUST” sont créées entre les deux patients, pour indiquer les méthodes ayant groupé les deux patients ensemble : ce sont les arêtes SUPPORT.

Ces arêtes SUPPORT portent chacune quatre attributs : “exp” pour l’expression de gène, “mirna” pour l’expression de miARN, “met” pour la méthylation et “multi” pour le multi-omique. Ces attributs permettent de savoir pour quels omiques les méthodes ont regroupé la paire de patients considérée (1 si la paire est dans le même cluster, 0 sinon).

Ainsi, dans l’exemple de la figure 9, la paire ayant été regroupée par PINS avec l’omique d’expression de miARN et avec la méthylation, l’arête “PINS_CLUST” voit ses attributs “mirna” et “met” passer à la valeur 1. La méthode NEMO, qui a regroupé la paire uniquement pour le clustering multi-omique, a tous ses attributs à 0, sauf “multi” passé à 1.

Ce genre d’arête est créé pour chaque méthode et pour chaque paire de patients. Une même paire de patients peut donc être reliée par au maximum 5 arêtes SUPPORT (5 méthodes qui auraient toutes classées les deux patients ensemble). Les arêtes SUPPORT sont créées depuis un script Python, grâce au package Py2neo.

4.2.2) Choix des données à intégrer et création d’une arête “d’INTEGRATION”

À ce stade, nous avons donc créé pour chaque méthode une arête SUPPORT spécifique permettant de relier les patients regroupés ensemble dans au moins un résultat de clustering, et récapitulant via ses attributs, quels omiques supportent la paire considérée.

Nous voulons maintenant permettre à un utilisateur d’utiliser ces informations pour produire un clustering intégré, c’est-à-dire construit à partir des résultats de clustering d’une ou plusieurs méthodes, un ou plusieurs omiques, … L’utilisateur doit pouvoir choisir simplement quelles informations (omiques et/ou méthodes) intégrer : pour cela, j’ai écrit en Python une fonction *create_rel_to_query* (disponible sur Github : https://github.com/galadrielbriere/Neomics/blob/master/dev/Neomics_Gala/graph_analysis/generalized_cluster_fusion.py).

Cette fonction prend en entrée deux listes : une spécifiant les méthodes à intégrer, l’autre spécifiant les omiques à intégrer. Par exemple, si on veut intégrer les résultats de clustering single-omique PINS et NEMO, les deux listes à passer à la fonction sont : [“PINS”, “NEMO”] et [“exp”, “mirna”, “met”]. Ou encore, si on veut intégrer les résultats de clustering de méthylation et d’expression de gènes de PINS uniquement : [“PINS”] et [“met”, “exp”].

La fonction *create_rel_to_query* créée une nouvelle relation spécifique en fonction de la demande de l'utilisateur. Cette arête, que nous appelons arête "d'INTÉGRATION", présente un seul attribut : *nb_support*, pour indiquer le nombre de supports (méthodes et omiques) pour chaque paire de patients. Le nombre de supports est calculé en fonction de ce qui est intégré. Les arêtes d'INTÉGRATION sont donc spécifiques à une intégration et sont stockées dans le graphe. Le principe de la création de l'arête d'INTÉGRATION est présenté dans la figure 10.

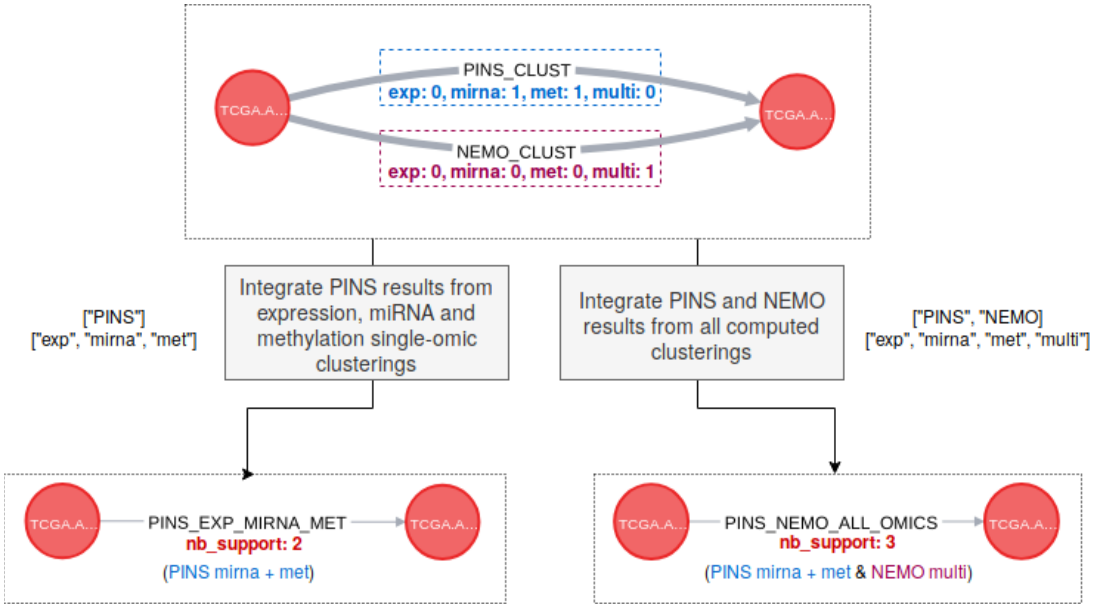


Figure 10 : Création d'une arête d'INTÉGRATION spécifique aux données intégrées par l'utilisateur

Comme le montre cette figure, une nouvelle arête est créée entre chaque paire de patients classés dans le même cluster par les méthodes et omiques considérés. Par exemple, on peut choisir d'intégrer les 3 clusterings single-omiques de PINS (flèche de gauche). Dans ce cas, comme PINS a classé la paire dans le même cluster pour les clusterings miARN et méthylation, mais pas pour le clustering expression de gène, le nombre de support pour la paire est de 2.

Dans le cas où on intègre tous les clusterings de PINS et NEMO (flèche de droite), le nombre de support est de 3 (clusterings PINS miRNA et méthylation et clustering de NEMO multi-omique).

La fonction *create_rel_to_query* qui créée les arêtes d'INTÉGRATION prend également un argument pour indiquer si tous les omiques à intégrer doivent supporter une paire de patient (condition ET) ou si les omiques doivent être considérés de manière distincte (condition OU). Le comportement de la fonction selon ce paramètre est décrit dans la figure 11.

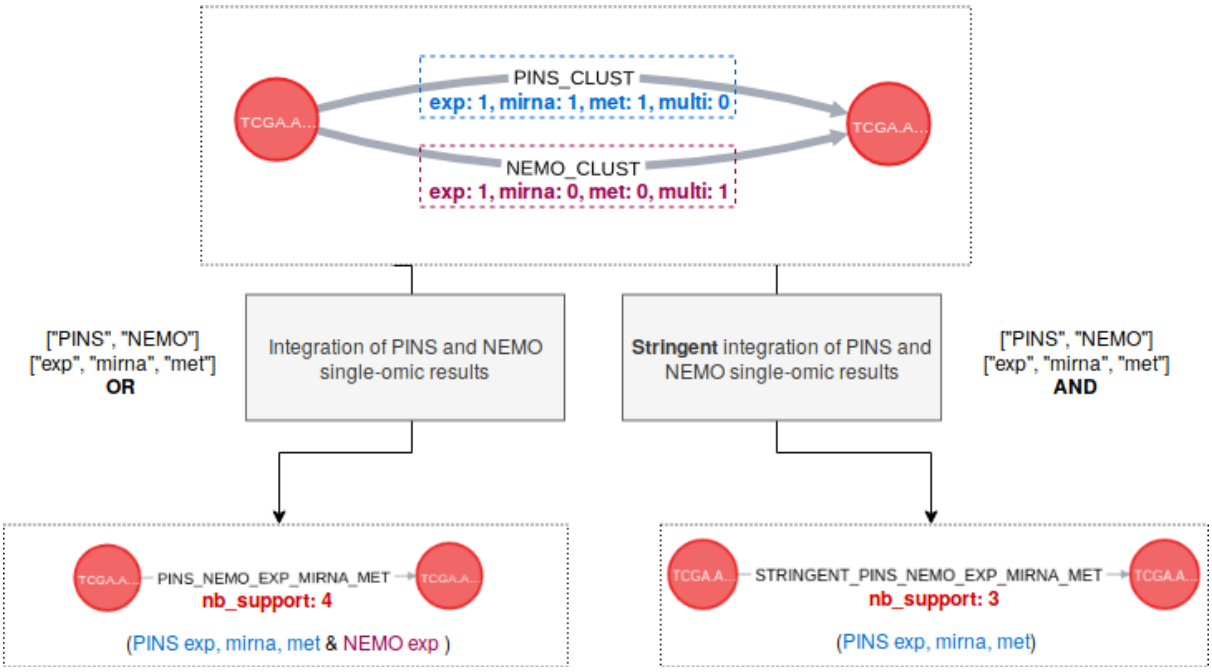


Figure 11 : Comportement de la fonction create_rel_to_query : omiques considérés séparément (flèche de gauche OR) et omiques considérés ensemble (flèche de droite AND)

Dans cette figure, les deux patients ont été clusterisés ensemble par les 3 clusterings single-omiques de PINS, le clustering sur l'expression de gène de NEMO et le clustering multi-omique de NEMO.

Dans le cas où on considère chaque omique séparément (flèche de gauche, condition OU), le nombre de support de l'arête d'intégration est de 4 (les 3 clusterings single-omique de PINS et le clustering expression de gène de NEMO).

Si on désire que les trois omiques à intégrer portent la même information, c'est-à-dire que les patients soient clusterisés ensemble par les 3 omiques (flèche de droite, condition ET), alors le nombre de support n'est plus que de 3. En effet, les clusterings single-omiques sur l'expression de miARN et de méthylation de NEMO n'ont pas classé les 2 patients considérés dans les mêmes clusters. Seuls les 3 clusterings single-omiques de PINS sont donc comptabilisés dans le nombre de supports.

En termes de requête, la différence entre les deux cas de figures est la suivante :

```
# On recherche les patients clusterisés ensembles par PINS sur les 3 omiques
avec une condition OU :
MATCH (p1:Patient)-[r:PINS_CLUST]-(p2:Patient)
WHERE r.exp=1 OR r.mirna=1 OR r.met=1
RETURN p1, r, p2

# On recherche les patients clusterisés ensembles par PINS sur les 3 omiques
avec une condition ET :
MATCH (p1:Patient)-[r:PINS_CLUST]-(p2:Patient)
WHERE r.exp=1 AND r.mirna=1 AND r.met=1
RETURN p1, r, p2
```

La deuxième requête est beaucoup plus stricte que la première, puisqu'elle impose que les 3 clusterings single-omiques de PINS aient classés les patients dans le même cluster. La

première requête, moins stricte, retourne toutes les paires de patients clusterisés ensemble par PINS pour au moins un omique.

4.2.3) Nombre de supports optimal : nombre de patients et complexité du graphe

La fonction *create_rel_to_query* permet donc à un utilisateur de créer une arête récapitulative spécifique, l'arête d'INTÉGRATION, afin de relier toutes les paires de patients clusterisés au moins une fois ensemble (selon les critères spécifiés par l'utilisateur). On obtient donc un graphe très connecté, puisque chaque paire de patient peut potentiellement être connectée par une arête d'intégration. Cependant, l'attribut *nb_support* de ces arêtes d'intégration peut être utilisé comme un poids, afin de filtrer les arêtes d'intégration en définissant un nombre minimal de supports. La figure 12 présente différents graphes, obtenus en filtrant les arêtes d'intégration sur le nombre de support.

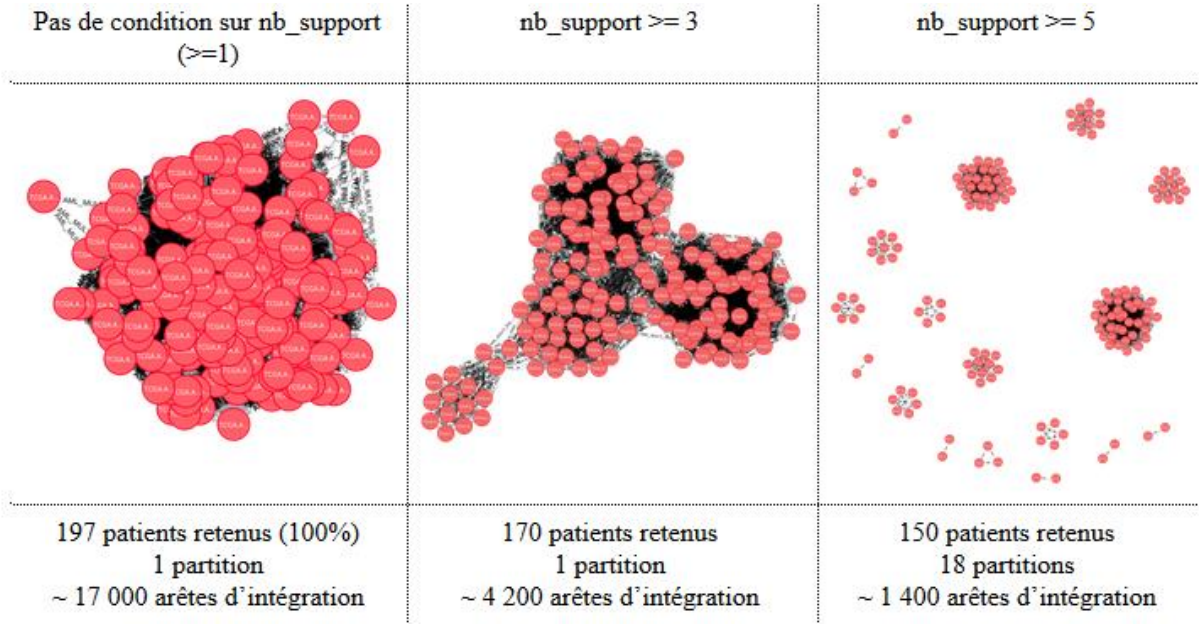


Figure 12 : Allure des graphes d'intégration selon le nombre minimum de supports demandé (cancer AML, intégration des clusterings multi-omiques de toutes les méthodes)

Requête : `MATCH (p1:Patient:AML)-[r:AML_MULTI_PINS_SNF_MKL_NEMO_MCCA]-(p2:Patient:AML) WHERE r.nb_support >= {x} RETURN p1, r, p2`

Comme le montre cette figure, en filtrant les arêtes sur le nombre de supports, les paires d'individu les moins stables entre les omiques et les méthodes sont éliminées. Si ce comportement est intéressant puisqu'il permet de ne conserver que les liens les plus robustes, il impose également une perte d'information si le nombre de supports seuil choisi est grand. Les patients n'étant jamais appariés avec un nombre de supports suffisant sont omis par l'analyse, et ne seront donc pas classés dans un cluster. Il y a donc ici un compromis à faire entre robustesse et perte des données.

En augmentant le nombre de supports accepté, on remarque également la formation de groupes de patients partageant plus de connections entre eux qu'avec les autres nœuds du graphe. La suppression des arêtes les moins stables laisse donc se former dans le graphe des clusters d'individus, classés ensemble par plusieurs omiques et méthodes. Plus le nombre de supports seuil est grand, plus le graphe est déconnecté. Si le nombre de supports est trop grand, alors les groupes se détachent et forment des partitions (des sous-graphes déconnectés) de trop

petite taille pour être informatives, par exemple des petites partitions regroupant uniquement 2 patients (voir figure 12, troisième graphe). Là encore, un compromis doit être fait.

Afin de déterminer la valeur optimale du nombre de supports seuil en fonction du nombre de patients conservés et du nombre et de la taille des partitions (il est intéressant d'avoir un certain nombre de partitions, *ie.* de groupes de patients connectés, si ces partitions sont de taille suffisante), j'ai écrit la fonction *find_optimal_nb_support* (*generalized_cluster_fusion.py*).

Cette fonction prend en entrée plusieurs paramètres :
→`nb_sup_init`, le nombre de support initial à tester
→`min_pop_size`, la taille minimale de la population à conserver (en % du total de la population multi-omique, *ie.* les patients pour lesquels les trois omiques ont été mesurés)
→`min_part_size`, la taille minimale d'une partition pour qu'elle soit considérée comme informative (en % du total de la population multi-omique)

À partir d'un nombre de supports initial fourni par lutilisateur, l'algorithme incrémente le nombre de supports de 1 tant que le (nombre de patients conservés) moins (la somme de la taille des partitions regroupant moins de `min_part_size`% de la population) vaut plus que `min_pop_size`%. Ce critère permet de s'assurer qu'on garde suffisamment d'individus dans l'analyse, même en enlevant les partitions trop petites pour être informatives.

Le nombre optimal de supports est donc le nombre le plus grand tel que les relations les moins robustes ne sont pas considérées mais que la quantité d'informations conservée est suffisante.

4.2.4) Algorithmes de détection de communauté

En filtrant le graphe d'intégration sur le nombre de supports, comme présenté dans la figure 12, avec un seuil approprié, des groupes de patients fortement interconnectés se distinguent. Il s'agit des patients ayant été classés dans les mêmes clusters pour plusieurs analyses. Un tel graphe peut être soumis à un algorithme de clustering de graphe afin de distinguer des groupes de nœuds plus fortement connectés au sein du groupe qu'avec les nœuds d'autres groupes. L'objectif est ici de créer des partitions distinctes d'individus classés de la même manière par les différents outils de prédiction, qui sont donc des individus susceptibles de présenter un même sous-type de cancer, en utilisant un algorithme de clustering de graphe.

Il existe plusieurs stratégies de détection de communautés : certains algorithmes détectent les arêtes inter-communautés et les retirent du réseau, d'autres agglomèrent les nœuds/communautés similaires de manière récursive, ou d'autres encore tentent de maximiser une certaine mesure de qualité (fonction objectif).

Il n'existe pas de définition consensuelle d'un "bon" cluster dans un graphe, ainsi de nombreuses mesures de qualité de clustering peuvent être utilisées pour qualifier les communautés d'un graphe. Selon la métrique de qualité choisie par une méthode de partitionnement de graphe, les clusters produits sont différents.

Une comparaison de différents algorithmes de clustering de graphe a été présentée durant le Symposium International sur les Algorithmes Web de 2015 (*Creusefond, 2015*). Parmi les algorithmes testés par ce chercheur, nous avons choisi de nous pencher plus spécifiquement sur l'algorithme de Louvain (*Blondel et al, 2008*) et l'algorithme Markov Clustering (*Van Dongen, 2000*). Ces deux méthodes ont été choisies car elles capitalisent sur

des stratégies différentes, présentent de bons résultats et peuvent être exécutées sur de grands graphes. De plus l'algorithme de Louvain a été implémenté directement dans Neo4j, ce qui facilite les développements. Le clustering de Markov n'ayant pas été implémenté dans Neo4j, j'ai utilisé le module Python *Markov Clustering* développé par G. Allard et mis à disposition sur Github (https://github.com/GuyAllard/markov_clustering, consulté le 7 août 2019).

L'algorithme de détection de communautés de Louvain se base sur le calcul de la modularité, une valeur comprise entre -1 et 1 qui compare la densité des arêtes au sein et au dehors d'une communauté. Cet algorithme renvoie le clustering avec la plus grande valeur de modularité. Il est possible de considérer le nombre de supports porté par les arêtes d'intégration comme un poids, utilisé par l'algorithme de Louvain.

L'algorithme de clustering de Markov utilise les marches aléatoires sur le graphe pour détecter les communautés. En considérant qu'il y a plus d'arêtes au sein d'un cluster et moins d'arêtes entre les clusters, il est plus probable de rester au sein d'un même cluster que de changer de cluster au cours d'une marche aléatoire. En simulant plusieurs marches aléatoires sur le graphe grâce aux chaînes de Markov, il est possible de voir où le flux se concentre, et donc quels sont les clusters. Cet algorithme prend en considération le poids des arêtes.

Exploiter ces algorithmes de détection de communauté permet donc de construire un nouveau clustering en se basant sur les arêtes d'intégration filtrées reliant les différents nœuds patients.

En pratique, ces deux algorithmes de détection de communautés sont appelés via Python. Pour l'algorithme de Louvain, une requête Cypher permettant l'exécution de l'algorithme est envoyée directement à Neo4j. Cette requête se présente sous la forme suivante:

```
# On définit les objets à clusteriser, c'est-à-dire tous les nœuds patients atteints du cancer considéré
objects = "MATCH (p:Patient:Cancer) RETURN DISTINCT id(p) as id"

# On définit les relations sur lesquelles doit se baser le clustering des nœuds, c'est à dire les arêtes d'intégration avec un nombre de support suffisant, et on retourne le nombre de supports comme un poids.
condition = "MATCH (p1:Patient:Cancer)-[r:REL_NAME]-(p2:Patient:Cancer)
WHERE r.nb_support >= opt_nb_support RETURN id(p1) as source, id(p2) as
target, r.nb_support as weight"


# On peut maintenant écrire la requête pour exécuter l'algorithme de Louvain.
louvain = "CALL algo.louvain(" + objects + "," + condition + "
{weightProperty:'weight', writeProperty:'community'})"
# Le nombre de supports porté par chaque arête d'intégration est considéré comme un poids, utilisé par l'algorithme de Louvain.
# La propriété 'community' sera ajoutée aux nœuds 'Patient' pour identifier la communauté à laquelle ils appartiennent.

# Exécution de la requête
graph.run(louvain)
```

Pour le clustering de Markov, non implémenté sur Neo4j, il faut d'abord récupérer le sous-graphe d'intérêt et le rendre exploitable pour Python avec le package Networkx :

```
# On récupère les noeuds et les arêtes d'intérêt pour l'analyse, c'est-à-dire uniquement les nœuds Patient du cancer considéré et les arêtes d'intégration les reliant avec un nombre de support suffisant. cypher.run renvoie un résultat sous la forme d'une liste de dictionnaires contenant toutes les données des nœuds et arêtes retenus par la requête.
subgraph_data = cypher.run("MATCH (p1:Patient:Cancer)-[r:REL_NAME]-(p2:Patient:Cancer) WHERE r.nb_support >= opt_nb_support RETURN p1, r, p2")

# On transforme les données en graphe pouvant être lu par Python, grâce au package Networkx
subgraph = subgraph_data.get_graph()

# On créer une matrice de taille n*n (n = nombre de nœuds) pour stocker le nombre de supports soutenant chaque paire de patients, avec le package Networkx (nx)
matrix = nx.to_spicy_sparse_matrix(subgraph, weight = "nb_support")

# On utilise le package Markov Clustering (mc) pour clusteriser le graphe décrit par la matrice
result = mc.run_mcl(matrix)

# On récupère les clusters détectés par l'algorithme
clusters = mc.get_clusters(result)
```

Il est possible que ces algorithmes de détection de communauté retournent des clusters de trop petite taille pour être exploitables. C'est pourquoi l'utilisateur doit définir un paramètre pour indiquer la taille minimale d'un cluster pour qu'il soit retenu. Ce seuil est défini par défaut à 5% de la population multi-omique. Cela signifie que toutes les communautés rassemblant moins de 5% de cette population seront retirés des résultats.

Si le nombre total de patients retenus dans l'ensemble des clusters principaux (ceux de taille > 5%) est plus petit que le nombre minimum de patients accepté (paramètre *min_pop_size*, voir partie 4.2.3), alors le nombre de supports à utiliser pour l'analyse du graphe est décrémenté de 1 par rapport au nombre de supports optimal calculé. En effet, en passant à un nombre de supports plus faible, davantage de patients seront retenus dans le graphe d'intégration et les nœuds seront plus interconnectés, permettant ainsi aux algorithmes de détection de communauté de former des clusters de plus grande taille. Tant que le nombre total de patients clusterisés est inférieur au seuil, le nombre de support est décrémenté.

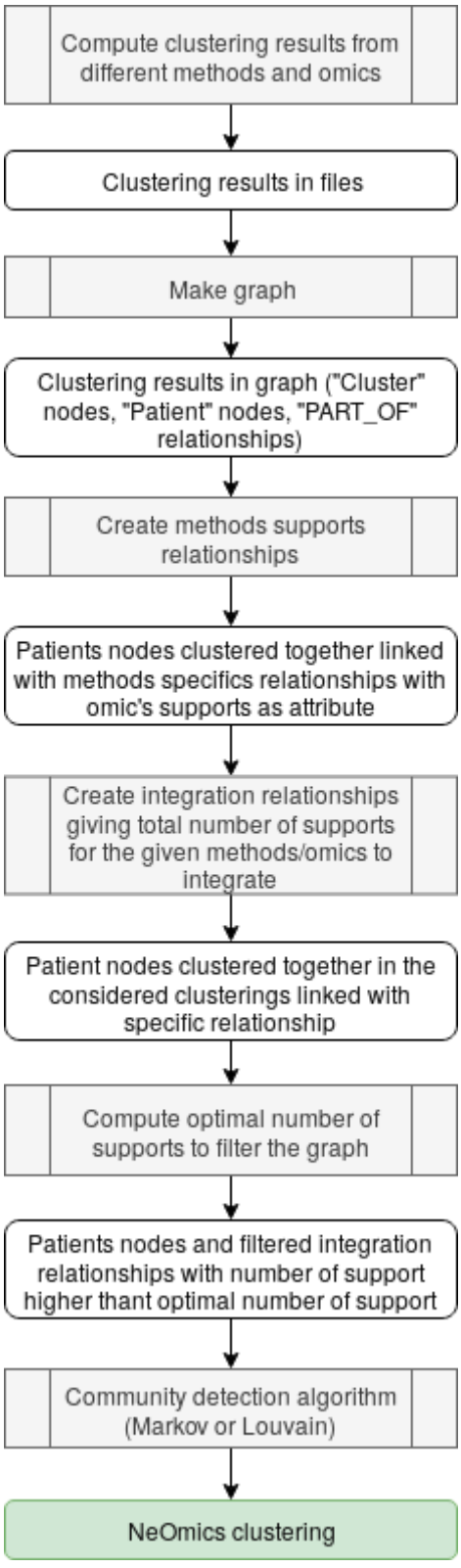


Figure 13 : Récapitulatif de la méthodologie

Bilan de la partie 4.2 : La méthodologie d'intégration de NeOmics est donc une méta-méthode qui se base sur la construction d'un graphe contenant à l'origine tous les résultats de clustering obtenus par des méthodes diverses, pour chaque type de données à intégrer. Des arêtes support sont créées entre chaque nœud pour définir si une paire de patients a été groupée dans le même cluster par une ou plusieurs méthodes d'analyses.

Une arête d'intégration spécifique est construite selon les données à intégrer et récapitule le nombre de méthodes et types de données supportant les différentes paires de patients.

Les arêtes d'intégration sont filtrées pour ne garder que les arêtes les plus robustes, c'est-à-dire celles présentant le plus grand nombre de supports.

Un algorithme de détection de communauté est ensuite utilisé afin de produire un clustering intégré.

4.3) Analyse du graphe : intégration des clusterings multi-omiques et single-omiques

Après avoir calculé tous les résultats de clustering single et multi-omiques avec les outils d'intégration choisis présentés dans la partie 3.3, les résultats ont été stockés dans Neo4j et les arêtes de support de chaque méthode ont été construites. Deux intégrations ont ensuite été effectuées en suivant la méthodologie présentée précédemment : une intégration des clusterings multi-omiques produits par les méthodes testées (intégration de méthodes) et une intégration des clusterings single-omiques des méthodes testées (intégration simultanée des méthodes et des omiques).

4.3.1) Intégration des méthodes : création d'un clustering consensus à partir de plusieurs clusterings multi-omiques

La première intégration testée avec NeOmics est l'intégration des différentes méthodes multi-omiques. Ici, on s'appuie uniquement sur les clusterings multi-omiques retournés par PINS, SNF, MCCA, rMKL et NEMO. On réalise donc une intégration NeOmics *a posteriori,* puisque l'intégration des omiques a déjà été réalisée par les 5 outils de prédiction, et ce sont uniquement les résultats de ces analyses multi-omiques qui sont intégrés.

Ainsi, pour construire les arêtes d'intégration, seules les arêtes support ayant un attribut "multi" = 1 sont prises en compte. Les arêtes d'intégration sont construites en appelant la fonction "*create_rel_to_query*" (*generalized_cluster_fusion.py*) avec les arguments pour indiquer : le cancer sur lequel travailler, les méthodes et les omiques à considérer et le nom à donner aux arêtes d'intégration. Par exemple, pour l'intégration des résultats multi-omiques sur le cancer AML (cancer du sang), la fonction est appelée ainsi :

```
create_rel_to_query("AML", ["PINS", "SNF", "rMKL", "MCCA", "NEMO"],
["multi"], "AML_MULTI_PINS_SNF_MKL_NEMO_MCCA")
```

La fonction *create_rel_to_query* crée donc les arêtes d'intégration correspondantes aux paramètres passés. La valeur maximale que peut prendre le nombre de supports pour ces arêtes d'intégration est donc de 5 (paire ayant été clusterisée ensemble par les 5 méthodes).

Les arêtes d'intégration sont ensuite filtrées sur leur nombre de supports, le nombre de supports optimal étant obtenu grâce à la fonction "*find_optimal_nb_support"*, déjà présentée dans la partie 4.2.3.

Une fois le graphe filtré, un algorithme de détection de communauté est exécuté pour produire les clusters NeOmics intégrés.

Le graphe filtré est présenté dans la figure 14. À gauche dans cette figure est représenté le graphe d'intégration simplement filtré avec le nombre de support optimal. À droite, il s'agit du même graphe dont les nœuds ont été colorés différemment pour chaque communauté détectée. Ici, on a donc obtenu après clustering du graphe (méthode Louvain), 5 communautés distinctes. Ces clusters correspondent à des groupes de patients clusterisés ensembles par plusieurs des méthodes et omiques qui ont été intégrées, donc des prédictions communes à plusieurs analyses.

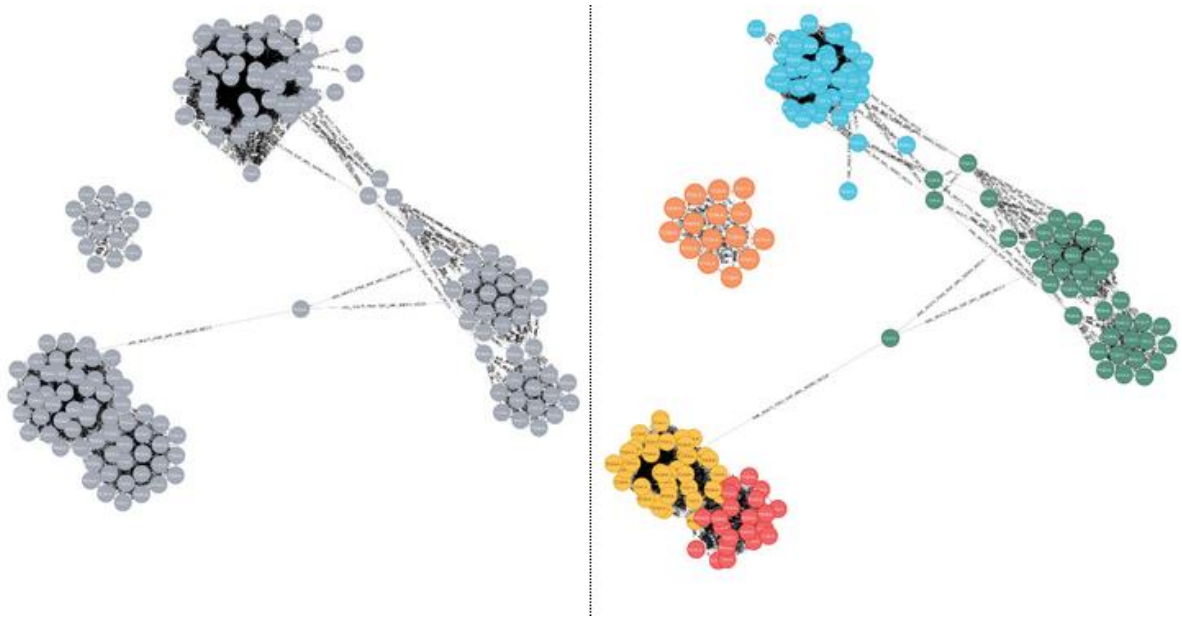


Figure 14 : Graphe d'intégration filtré (à gauche) et graphe d'intégration filtré avec coloration des nœuds par communauté (à droite).
Intégration des clusterings multi-omiques, cancer AML, nb_support >= 4, clustering Louvain.

Ce nouveau clustering NeOmics est également stocké dans un fichier au format classique de résultats de clustering (une ligne par patient avec identifiant du patient et numéro du cluster auquel il appartient).

4.3.2) Intégration des omiques : création d'un clustering consensus à partir de plusieurs clusterings single-omiques

Après l'intégration des omiques *a posteriori*, grâce aux résultats de clustering donnés par les outils de prédictions multi-omiques existants, une intégration des données de clustering single-omique a été réalisée.

Deux versions de cette intégration ont été imaginées : une intégration des omiques "simple" et une intégration "stricte". La différence entre ces deux intégrations réside dans la création des arêtes d'intégration. L'intégration simple crée une arête d'intégration en considérant tous les omiques séparément. L'intégration stricte impose, pour créer une arête d'intégration entre deux nœuds "Patient", que les trois omiques supportent cette même paire. Ce paramétrage a déjà été présenté dans la partie 4.2.2 et dans la figure 11.

L'appel à la fonction *create_rel_to_query* s'est fait en utilisant le paramètre par défaut *or_q = 1* (*or_q* pour "*query OR*") pour l'intégration simple :

```
create_rel_to_query("AML", ["PINS", "SNF", "rMKL", "NEMO"], ["exp", "mirna", "met"], "AML_EXP_MIRNA_MET_PINS_SNF_MKL_NEMO")
```

Notons que la méthode MCCA n'est ici pas appelée, puisque cet outil de prédiction ne permet pas de réaliser de clustering single-omique.

Pour l'intégration stricte, le paramètre *or_q* est passé à 0, pour imposer qu'une arête d'intégration soit créée si et seulement si les trois omiques considérés ont chacun permis pour

au moins une méthode de clusteriser une paire dans le même groupe. L'appel à la fonction *create_rel_to_query* est donc :

```
create_rel_to_query("AML", ["PINS", "SNF", "rMKL", "NEMO"], ["exp", "mirna", "met"], "AML_EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT", 0)
```

Une fois les arêtes d'intégration construites, les deux sous-graphes ont été filtrés sur le nombre de supports (en déterminant pour les deux graphes leur nombre de supports optimal respectif) puis soumis à un algorithme de clustering de graphe. La figure 15 ci-dessous présente l'allure des communautés pour les deux intégrations (clustering Louvain).

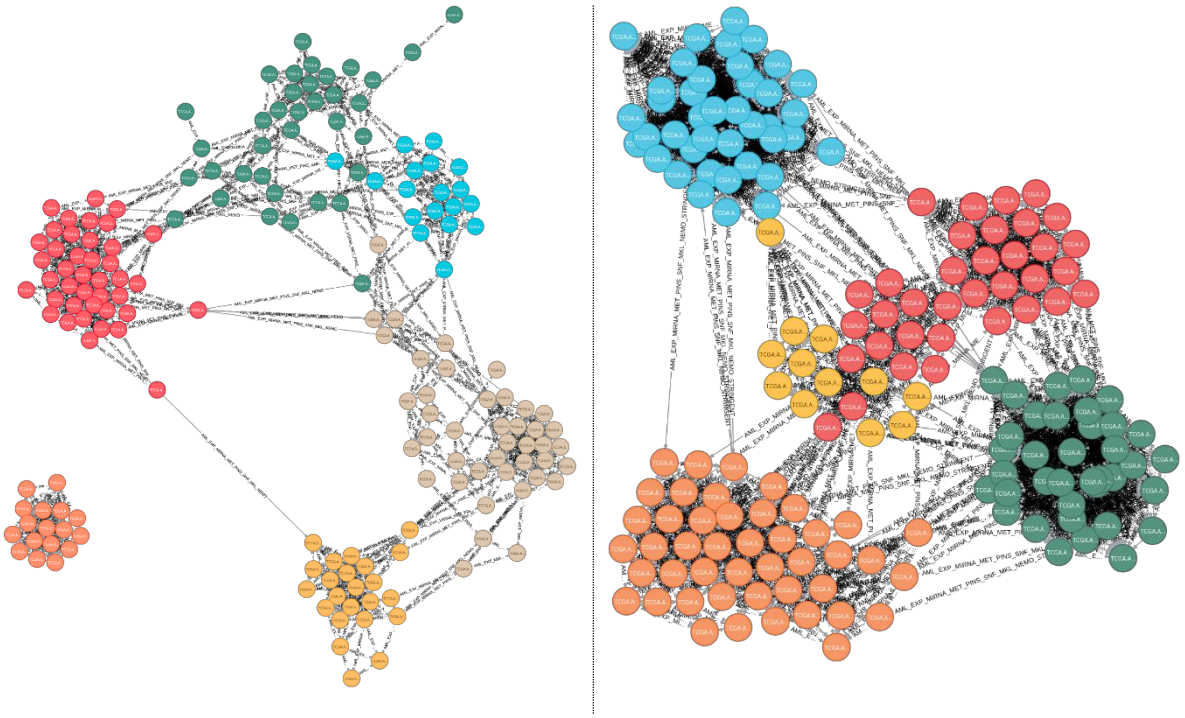


Figure 15 : Graphes d'intégration filtrés et colorés par communauté pour l'intégration simple à gauche (nb_support >= 8) et l'intégration stricte à droite (nb_support >= 3).
Intégration des clusterings single-omique, cancer AML, clustering Louvain.

Bilan de la partie 4.3 : La méthodologie NeOmics a été appliquée sur trois cas d'intégration:

• **Intégration *a posteriori*** : des clusterings multi-omiques sont produits par les outils de prédiction multi-omiques existants et exploités pour produire un clustering multi-omique consensus.

• **Intégration *a priori*** : des clusterings sont produits indépendamment pour chaque omique et sont exploités conjointement pour produire un clustering multi-omique consensus

→ **Simple :** toutes les paires de patients clusterisés ensemble pour au moins un omique sont prises en compte pour la création des arêtes d'intégration.

→ **Stricte :** seules les paires de patients clusterisés ensemble au moins une fois pour chaque omique sont prises en compte pour la création des arêtes d'intégration.

5) Résultats : métriques graphes et interprétation biologique des clusters

Nous avons donc à ce stade produit de nouveaux résultats de clustering, grâce à la méthodologie présentée dans la partie précédente. Afin d'évaluer l'efficacité de la méthode NeOmics, il faut analyser les résultats produits et définir des métriques permettant de comparer les résultats obtenus par NeOmics et ceux des autres outils de prédiction testés.

Deux grandes catégories de métriques ont été choisies : des métriques permettant de décrire le comportement du graphe d'intégration en fonction des paramètres d'exécution choisis, et des métriques d'interprétation biologique permettant de discuter la validité biologique des clusters produits.

5.1) Exigence (stringence) des requêtes, nombre de supports seuil, algorithme de clustering de graphe : l'impact sur les résultats NeOmics

5.1.1) Variations du nombre de supports et impact sur le graphe d'intégration

Comme présenté dans la partie 4.2.3, il est important de définir un nombre de supports seuil approprié pour filtrer le graphe en gardant uniquement les arêtes les plus robustes tout en évitant de perdre trop de données. En effet, un nombre de supports seuil trop haut implique une perte de nœuds (d'individus) dans l'analyse et la création de communautés de taille potentiellement trop petite par les algorithmes de clustering de graphe. Afin de définir un nombre de supports optimal à utiliser, on se base sur différents critères. Le nombre de nœuds retenus après le filtre sur le nombre de supports doit être suffisant. Le nombre de partitions, c'est-à-dire de sous graphes déconnectés, est directement corrélé avec le nombre de clusters produits par les algorithmes de clustering de graphe (une partition donnera au minimum un communauté distincte). L'apparition de partitions dans le graphe par filtre des arêtes d'intégration est donc une propriété intéressante pour définir un nombre de supports adéquat, mais un nombre de partitions très grand implique la détection de communautés petites et donc non informatives.

La figure 16 ci-dessous analyse l'impact de deux métriques en fonction du nombre de supports minimum imposé : le nombre de nœuds et le nombre de partitions dans le graphe d'intégration filtré. Il s'agit ici d'une moyenne calculée avec les données issues des 9 cancers. Des graphes similaires ont été produits pour chaque cancer et sont disponibles en annexe (annexes 2 à 10).

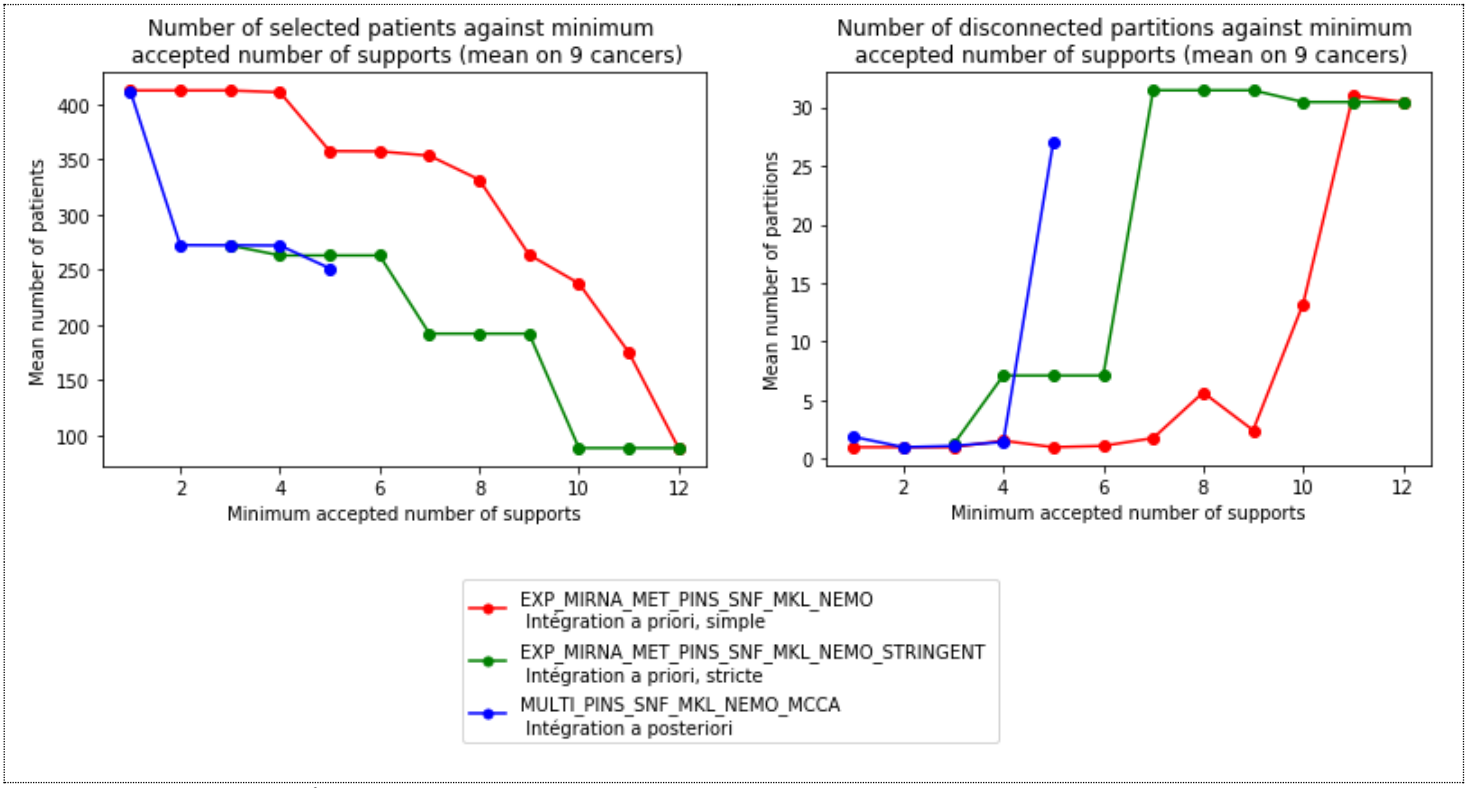


Figure 16 : Évolution de nombre de patients retenus et du nombre de partitions dans le graphe d'intégration filtré sur différentes valeurs de nombre de supports. Moyenne sur 9 cancers.

Notons que pour l'intégration *a priori*, version stricte (courbes vertes), on impose que les trois omiques soient en accord pour créer une arête d'intégration. Le nombre de supports ne peut donc varier que de 3 en trois, avec un minimum de (3 omiques * 1 méthode) = 3 et un maximum de (3 omiques * 4 méthodes) = 12. Pour l'intégration *a priori*, version simple (courbes rouges), les omiques peuvent être en désaccord : le nombre de support peut varier par pas de 1, avec un minimum à 1 et un maximum à 12. Pour l'intégration *a posteriori* (courbes bleues), le nombre de support varie également de 1 en 1, avec un minimum à 1 et un maximum à 5 (5 méthodes ont produit 5 clusterings multi-omiques sur lesquels on se base pour l'intégration).

Comme on peut le voir dans le premier graphe de la figure 16, le nombre de patients retenus dans le graphe d'intégration diminue lorsque le nombre de supports augmente : il faut choisir un nombre de supports seuil approprié pour ne pas diminuer excessivement la population. Il est intéressant de comparer les résultats pour l'intégration a priori des deux versions : simple (courbe rouge) et stricte (courbe verte). Pour les mêmes données intégrées, on voit que le fait d'imposer que les différents omiques soient en accord implique une perte de données importantes, puisque près de 35% des nœuds sont rejetés des analyses pour l'intégration stricte pour un nombre de supports à 3 uniquement. Pour l'intégration *a posteriori* des clusterings multi-omiques, on remarque un palier entre les nombres de supports 2 et 4 : cela montre un certain accord des algorithmes de prédiction. Ce palier est également retrouvé pour chaque cancer individuellement (*cf.* annexes 2 à 10). Cela signifie que 3 des 5 méthodes de prédiction testées sont généralement en accord sur le clustering d'une paire de patients. Il serait également intéressant de savoir si les 3 méthodes en accord sont toujours les mêmes, mais cette information n'étant pas stockée dans les arêtes d'intégration, de nouveaux développements sont nécessaires et n'ont pour le moment pas encore été réalisés par manque de temps.

Le deuxième graphe présenté figure 16 présente le nombre de partitions, c'est-à-dire de sous graphes déconnectés dans le graphe d'intégration, en fonction du nombre de supports. On remarque que le nombre de partitions augmente brutalement en fonction d'un nombre de supports seuil (5 pour l'intégration *a posteriori*, 7 pour l'intégration *a priori* stricte et 10 pour l'intégration *a priori* simple). Le nombre de partitions maximum obtenu peut atteindre jusqu'à plus de 30 partitions, ce qui aboutirait à au moins autant de clusters, ce qui n'est pas un comportement souhaitable. Cela correspond par exemple au troisième graphe d'intégration déjà présenté figure 12, avec beaucoup de petites communautés rassemblant un très petit nombre de patients.

Ce sont ces deux métriques qui ont permis de définir un nombre de supports optimal pour filtrer le graphe, comme cela a été expliqué dans la partie 4.2.3. En moyenne sur les 9 cancers pour lesquels des résultats ont été obtenus, le nombre de supports optimal vaut 8.7 pour l'intégration *a priori* simple, 4.3 pour l'intégration *a priori* stricte et 4 pour l'intégration *a posteriori*.

5.1.2) Nombre de supports optimal, nombre de supports effectif et clustering du graphe d'intégration filtré

Pour rappel, lors du clustering du graphe par des algorithmes de détection de communauté, si les clusters créés sont d'une taille insuffisante en termes de patients, ils sont considérés comme non informatif et les patients appartenant à ces clusters ne sont pas pris en compte. Si le nombre total de patients clusterisés dans des clusters principaux est plus petit qu'un seuil donné par l'utilisateur, alors l'analyse est considérée comme non concluante et le nombre de supports choisi est décrémenté. C'est pourquoi le nombre de supports optimal retourné par la fonction "*find_optimal_nb_support*" ne correspond pas forcément au nombre de supports final choisi pour les analyses. Ce nombre de supports "effectif" dépend donc de l'algorithme de détection de communautés utilisé et de sa capacité à créer des clusters suffisamment gros pour être conservés.

Le tableau 2 ci-dessous récapitule, pour chaque cancer et chaque type d'intégration le nombre de supports optimal et le nombre de supports réellement choisi pour le clustering du graphe (pour les deux méthodes, Louvain et Markov).

Tableau 2 : Nombre de supports optimal (retournés par la fonction find_optimal_nb_support) et nombre de supports choisi pour la détection des communautés

	EXP_MIRNA_MET_PINS_SNF_MKL_NEMO Intégration *a priori,* simple			EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT Intégration *a priori*, stricte			MULTI_PINS_SNF_MKL_NEMO_MCCA Intégration *a posteriori*		
	Optimal	Louvain	Markov	Optimal	Louvain	Markov	Optimal	Louvain	Markov
AML	8	**8**	7	3	3	3	4	4	4
COAD	9	**8**	7	3	**3**	3 (202/209)	4	**4**	3
GBM	9	**9**	8	6	3	3	4	**4**	3
KIRC	9	**9**	8	6	3	3	4	**4**	3
LIHC	9	**9**	6	3	**3**	3 (292/348)	4	4	4
LUSC	8	**8**	7	3	**3**	3 (306/324)	4	4	4
OV	9	**9**	8	6	3	3	4	4	4
SARC	9	**9**	8	6	**6**	3	4	4	4
SKCM	8	**8**	6	3	**3**	3 (318/333)	4	**4**	3

Dans ce tableau, les nombres de supports en rouge indiquent que, le seuil choisi n'a pas permis de produire une analyse qui classe suffisamment de patients par rapport à la limite fixée par l'utilisateur. Les chiffres entre parenthèses indiquent le nombre de patients retournés par l'analyse, contre le nombre de patients minimal accepté pour que l'analyse soit validée. Par exemple, pour le cancer du côlon COAD, l'analyse *a priori* version stricte avec la méthode de clustering Markov a été exécutée avec un nombre de supports de 3 (nombre minimal de supports possibles pour respecter la contrainte "accord des 3 omiques"). Or, l'utilisation de ce nombre de supports n'a pas permis de maintenir un nombre suffisant de patients dans l'analyse, puisque seulement 202 patients ont été classés dans des clusters, contre les 209 acceptés au minimum (95% de la population multi-omique totale). Le tableau 3, qui sera discuté un peu plus tard dans cette partie, présente le nombre de patients retenus par analyse par rapport au pourcentage minimum accepté de la population multi-omique totale.

Comme on peut le constater dans le tableau 2, le nombre de supports choisi varie, pour la même analyse, en fonction de l'algorithme de clustering de graphe choisi. On remarque notamment que la méthode Markov semble plus stricte que celle de Louvain, puisque pour le même type d'intégration, le nombre de supports choisis pour Markov est inférieur ou égal à celui choisi pour Louvain. Cela signifie que l'algorithme de Markov a tendance à créer des communautés plus petites, dont certaines seront de taille trop petite pour être considérées dans l'analyse, et donc supprimées. Ainsi, il faut diminuer le nombre de supports seuil pour obtenir une analyse informative du point de vue du nombre de patients clusterisés. L'algorithme de Louvain semble donc moins strict sur l'assignation d'un individu à une communauté, et davantage susceptible de produire un clustering intéressant en utilisant un nombre de supports égal ou proche du nombre de supports optimal calculé. La différence de comportement de ces deux algorithmes peut notamment être observée dans la figure 17.

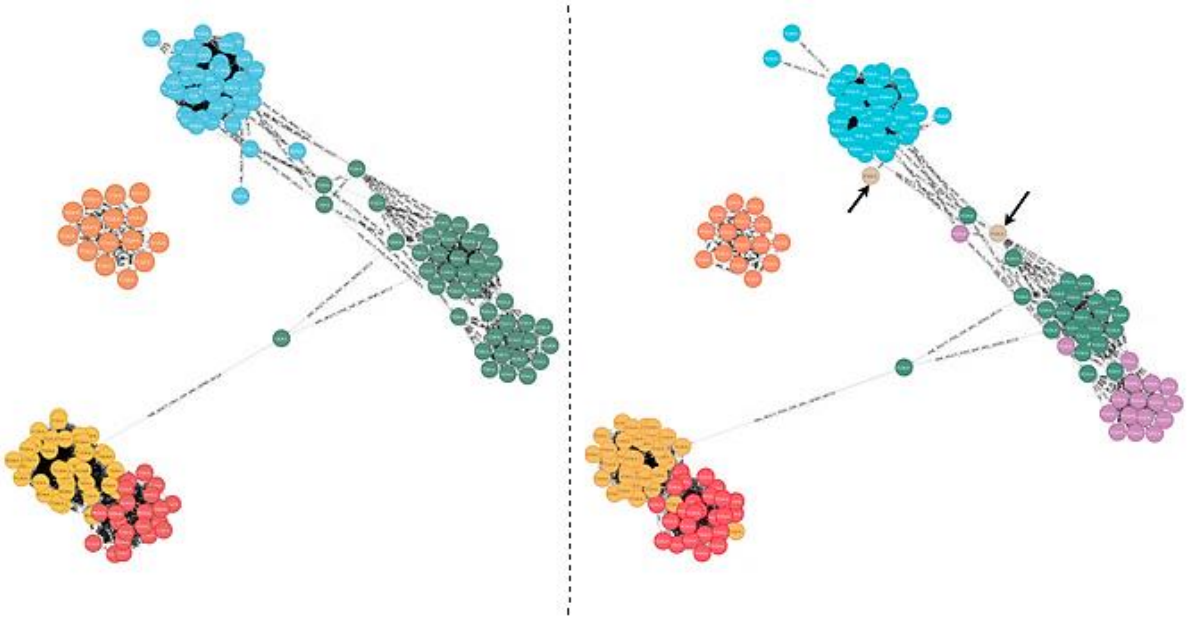


Figure 17 : Graphe d'intégration clusterisé par Louvain (à gauche) et Markov (à droite).
Intégration des clusterings multi-omiques, cancer AML, nb_support >= 4.

Dans cette figure, on voit que l'utilisation du clustering de Markov permet de détecter une "sous-communauté" au sein de la communauté colorée en vert, détectée par Louvain. On remarque également que deux patients sont exclus des communautés (nœuds indiqués par une flèche sur le graphe) de Markov, alors qu'ils sont bien classés dans une communauté pour l'exécution avec Louvain.

Cette figure illustre bien la raison pour laquelle le nombre de supports choisi pour le clustering de Markov est généralement plus bas que celui choisi pour Louvain : les communautés détectées par Markov sont susceptibles d'être plus petites, et l'appariement d'un nœud à une communauté est plus strict. Certains nœuds se retrouvent donc non classés (en réalité, ils sont classés dans des clusters de taille 1) et sont retirés des résultats.

Le tableau 3 récapitule le nombre de patients classés pour les différentes analyses. Les cases en rouges représentent les analyses pour lesquelles, malgré un nombre de supports seuil le plus bas possible, le nombre d'individus retournés est plus petit que le seuil minimum imposé (dans nos analyses, 95% de la population multi-omique).

Tableau 3 : Nombre de patients clusterisés par cancer et par analyse, seuil accepté et population totale

	EXP_MIRNA_MET_PINS_SNF_MKL_NEMO Intégration *a priori*, simple		EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT Intégration *a priori*, stricte		MULTI_PINS_SNF_MKL_NEMO_MCCA Intégration *a posteriori*		Min accepted pop (95% pop multi)	Pop multi
	Louvain	Markov	Louvain	Markov	Louvain	Markov		
AML	177	180	170	170	170	168	161	**170**
COAD	245	240	209	202	219	220	209	**220**
GBM	262	272	274	267	261	269	260	274
KIRC	176	210	183	176	176	176	174	**183**
LIHC	352	361	365	292	367	351	348	**367**
LUSC	349	336	324	306	341	337	324	**341**
OV	279	313	287	287	287	284	272	**287**
SARC	253	257	250	257	255	247	244	257
SKCM	355	362	333	318	351	351	333	**351**

Les cases colorées en vert indiquent que le nombre d'individus clusterisés par l'analyse est plus grand que la taille totale de la population multi-omique, c'est à dire que le nombre d'individus pour lesquels les 3 omiques ont été séquencés. Notons que pour l'analyse *a priori* version stricte, il n'est pas possible de surpasser la taille de la population multi-omique, puisque cette version impose que les 3 omiques soient en accord, et donc, qu'ils aient été mesurés. Il en va de même pour l'intégration *a posteriori*, puisque tous les algorithmes ayant servi à produire les clusters d'entrée, à l'exception de NEMO, ne gèrent pas les données manquantes et donc ne traitent que les individus multi-omiques.

On voit donc que l'analyse *a priori*, version simple, permet de clusteriser des patients pour lesquels seulement un ou deux omiques a été séquencé. En effet, les clusterings fournis en entrée de l'analyse sont des clusterings single-omique : tous les individus dont l'omique concerné a été mesuré peuvent être pris en compte, indépendamment des autres omiques. Si un individu pour lequel seul un omique a été mesuré se retrouve clusterisé de la même manière par les différentes méthodes, son nombre de supports est incrémenté et vaut au maximum 4 (car 4 méthodes de prédiction single-omique ont été utilisées). Les individus pour lesquels deux omiques ont été séquencés peuvent porter un nombre de supports maximum de 8 (2 omiques * 4 méthodes de prédiction). Ces individus apportent donc des informations supplémentaires à

l'analyse, peuvent être pris en compte avec un nombre de supports seuil approprié. Par exemple, le choix d'un nombre de supports seuil à 4 pourrait permettre de classifier les patients pour lesquels seul un omique a été séquencé. Dans notre cas, et comme le montre le tableau 2 présenté précédemment, le nombre de supports seuil choisi vaut au minimum 6, ce qui permet de classer les patients pour lesquels au moins deux omiques ont été mesurés. En d'autres termes, on peut dire que la méthode d'intégration *a priori*, version simple, permet une certaine gestion des données manquantes.

Outre cette caractéristique intéressante de l'intégration *a priori*, version simple, l'intérêt de cette approche est qu'elle n'impose pas que les omiques soient en accord. En effet, les omiques portant des informations différentes et complémentaires, il n'est pas évident qu'ils présentent les mêmes caractéristiques ni qu'ils varient de la même manière. La contrainte de l'accord des 3 omiques est une contrainte très forte, qui explique d'ailleurs les faibles valeurs de nombre de supports retenus pour l'analyse *a priori*, version stricte.

5.1.3) Communautés détectées, coefficient de clustering et centralité : petit-monde ?

Jusqu'ici, nous avons donc décrit le comportement du graphe d'intégration par rapport au nombre de supports utilisé pour le filtrer, et nous avons décrit l'influence de la méthode de clustering de graphe utilisée sur le nombre de patients retenus dans les analyses et sur le nombre de supports à appliquer pour respecter la contrainte de la taille de la population analysée. Nous voulons maintenant caractériser les clusters produits par ces deux méthodes de détection de communauté, Louvain et Markov.

En 1998, Watts et Strogatz introduisent la notion de réseau "petit-monde" (*Watts et Strogatz, 1998*). S'il n'existe pas de définition stricte de cette notion, les graphes petits-mondes présentent deux caractéristiques principales : d'une part, la distance moyenne entre toute paire de nœuds du graphe doit être faible (en théorie des graphes, la longueur d'un chemin étant le nombre d'arêtes à parcourir pour passer d'un nœud à un autre), et d'autre part, les nœuds doivent être très connectés à leurs voisins immédiats. En pratique, la notion de réseau petit-monde est utilisée dans de nombreux domaines : de l'étude des réseaux sociaux à celle des interactions protéiques ou du transport aérien, elle permet d'étudier l'influence d'un nœud dans un graphe, l'efficacité du transfert d'informations d'un nœud à l'autre, …

Dans notre cas, il est intéressant de caractériser la tendance des clusters détectés par Louvain ou Markov à avoir un comportement de graphe petit-monde. En effet, idéalement, tous les nœuds au sein d'une même communauté devraient pouvoir être reliés entre eux par un chemin court, puisqu'il est souhaitable que tous les nœuds d'un même cluster soient fortement connectés : cela signifie qu'ils ont été clusterisés ensemble par un nombre suffisant de méthodes et d'omiques.

Nous avons donc utilisé deux métriques pour caractériser la tendance petit-monde :
→ Le coefficient de clustering, pour mesurer le regroupement des nœuds (leur degré de connexion avec leurs voisins), dont les valeurs possibles sont comprises entre 0 et 1.
→ La centralité de proximité (*closeness centrality*), pour mesurer la distance moyenne entre les nœuds, dont les valeurs possibles sont également comprises entre 0 et 1.

Ces deux métriques ont été implémentées sur Neo4j, leur calcul a donc pu être réalisé via une requête Cypher. Plus leur valeur est haute, plus le graphe présente une tendance petit-monde.

Notons que pour pouvoir caractériser un graphe de petit-monde, il faudrait le comparer à un graphe avec les mêmes propriétés (nombre de nœuds et d'arêtes) mais généré de manière aléatoire, et caractériser la significativité des différences en terme de longueur moyenne des chemins entre deux nœuds et de coefficient de clustering. Par manque de temps, cela n'a pas été réalisé. Nous parlons donc ici d'une *tendance* du graphe à présenter des propriétés petit-monde.

La figure 18 présente les résultats du calcul de ces deux métriques sur différents graphes d'intégration :

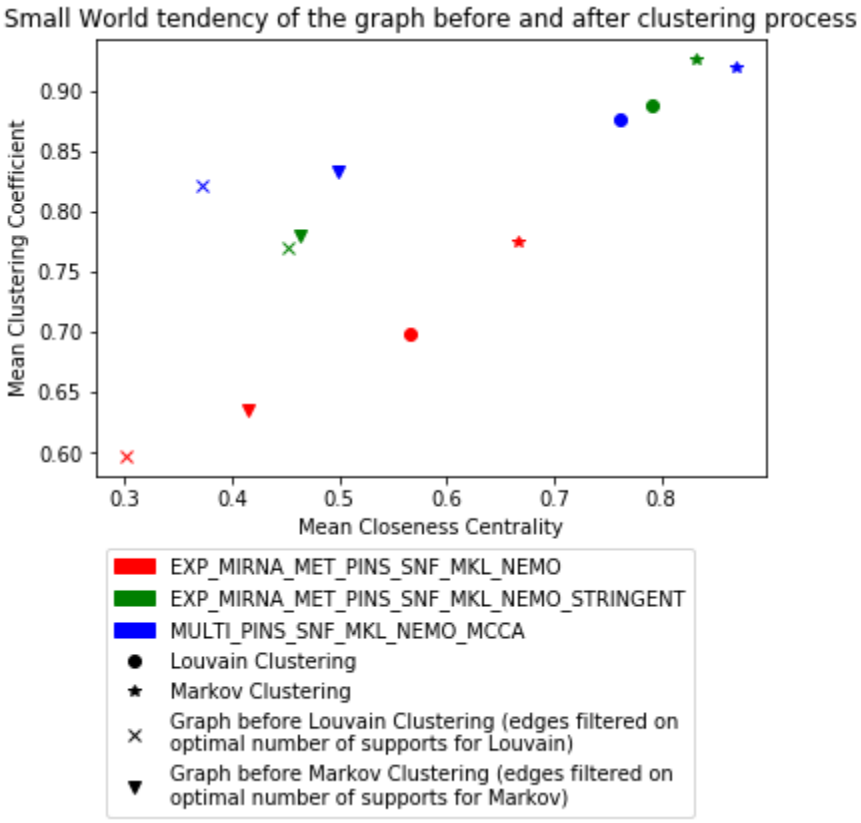


Figure 18 : Tendance petit-monde des graphes d'intégration et des clusterings (moyenne sur 9 cancers)

Plusieurs mesures ont été réalisées. Tout d'abord, j'ai calculé les valeurs de coefficient de clustering et de centralité moyennes des graphes d'intégrations filtrés sur les nombres de supports efficients utilisé. Sur ce graphique, les croix (respectivement, les triangles) représentent donc la valeur de ces deux métriques pour le graphe d'intégration filtré avec le nombre de supports utilisé pour le clustering via la méthode Louvain (respectivement, méthode Markov). Il s'agit ici des valeurs moyennes sur les 9 cancers testés, mais ces résultats ont également été calculés pour chaque cancer et sont présentés dans les annexes 2 à 10.

Les points (respectivement, les étoiles), représentent la moyenne des valeurs de ces deux métriques calculées sur les clusters générés par Louvain (respectivement, Markov), le tout moyenné sur les 9 cancers testés. En d'autres termes, on compare la tendance du graphe d'intégration avant et après clustering.

Plus les points se retrouvent en haut et à gauche du graphique, plus le graphe se rapproche d'un graphe petit-monde (coefficient de clustering et centralité de proximité maximisés).

On peut ainsi vérifier que les algorithmes de détection de communauté créent des clusters cohérents, puisqu'on observe bien une augmentation du coefficient de clustering et de la centralité de proximité.

Ici, on doit se contenter de comparer le graphe d'intégration avant clustering de Louvain (respectivement Markov) avec le graphe d'intégration après clustering via la méthode Louvain (respectivement Markov) pour un type d'intégration donné. En effet, les facteurs de variations sont nombreux pour les autres comparaisons : type d'intégration, nombre de supports utilisé pour filtrer le graphe, nombre de patients dans le graphe d'intégration, nombre d'arêtes dans le graphe d'intégration, … Une comparaison plus poussée n'a donc pas pu être réalisée pour le moment.

On pourra simplement remarquer qu'utiliser la méthode de clustering de Markov amène à des clusters plus robustes en termes de métriques petit-monde, sans conclure sur les raisons de cette meilleure performance : nombre de supports choisi pour filtrer le graphe (et donc nombre de nœuds et d'arêtes dans le graphe) ou efficacité de la méthode de partitionnement de données.

Bilan de la partie 5.1 : Nous avons commencé l'analyse des résultats par l'étude de plusieurs métriques graphes sur le graphe d'intégration et les clusters finaux. Nous avons décrit l'importance du choix du nombre de supports seuil choisi pour filtrer le graphe d'intégration et du choix de l'algorithme de clustering de graphe utilisé. La méthode Markov semble plus stricte que la méthode Louvain, ce qui impose parfois d'utiliser un nombre de supports seuil plus faible pour limiter la perte de données.

Nous avons également souligné un atout important de l'intégration *a priori* simple : en n'imposant pas un accord des trois omiques et en choisissant un nombre de supports approprié, il est possible de prendre en compte les individus pour lesquels tous les omiques n'ont pas été mesurés. Parmi les méthodes d'intégration testées, seul l'outil NEMO permet une certaine gestion des données manquantes.

Enfin, nous avons qualifié la qualité des clusters produits par les algorithmes de détection de communauté en utilisant la notion de petit-monde.

5.2) Interprétation biologique des résultats : pertinence biologique des clusters

Nous avons donc produit de nouveaux clusters NeOmics en faisant 3 types d'intégration différentes. Nous voulons maintenant comparer ces différents clusterings multi-omiques aux clusterings produits par les méthodes de prédiction utilisées et qualifier la pertinence biologique des clusters.

Pour cela, nous avons utilisé deux métriques, également utilisées par Rappoport et Shamir dans leur revue des méthodes d'intégration de données multi-omiques : les courbes de survie des clusters, et leur enrichissement en certains labels cliniques.

5.2.1) Courbe de survie

La première métrique pour attester de la pertinence biologie proposée par Rappoport et Shamir est l'analyse de survie. Ce type d'analyse, très utilisée dans l'étude des cancers mais aussi dans de nombreux autres domaines, consiste à calculer le taux de survie des individus malades en fonction du temps.

Dans le cadre de la prédiction de sous-type de cancer, il a été montré que le type moléculaire de cancer est lié à la survie observée chez les patients atteints. La figure 19 ci-dessous (*Fallahpour et al, 2017*), montre par exemple les différentes courbes de survie observées pour différents types moléculaires du cancer : Luminal A, Luminal B, HER2 et Triple Négatif.

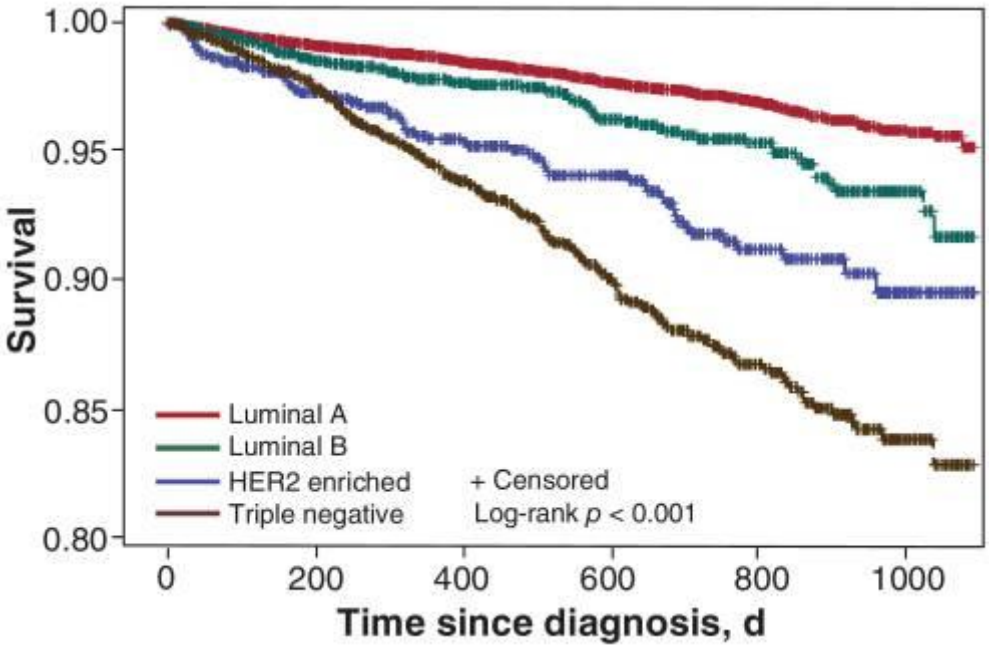


Figure 19 : Taux de survie en fonction du type moléculaire de cancer du sein
Source : *Fallahpour et al, 2017*

Afin de déterminer si les courbes de survie étaient significativement différentes selon le type moléculaire présenté, les auteurs ont utilisé le test statistique de log-rank.

Ainsi, la pertinence biologique des clusters produits peut être mesurée en réalisant une analyse de survie sur les clusters produits, et en regardant si les courbes de survie présentent des différences significatives en calculant la p-valeur du test de log-rank.

Les données de survie se présentent sous la forme suivante :

Tableau 4 : Organisation des données utilisées pour l'analyse de survie

Patient ID	Survival	Death
TCGA.AB.2934.03	28	0
TCGA.AB.2840.03	577	1
TCGA.AB.2850.03	304	0

Les données sont rassemblées dans un tableau où chaque ligne correspond à un patient, identifié par son code TCGA. La colonne *Death* indique si l'individu en question est toujours vivant (0) ou s'il est décédé (1). La colonne *Survival* indique le nombre de jours écoulés entre le moment où l'individu a été diagnostiqué comme présentant un cancer, et le jour où l'observation mort/survie a été réalisée.

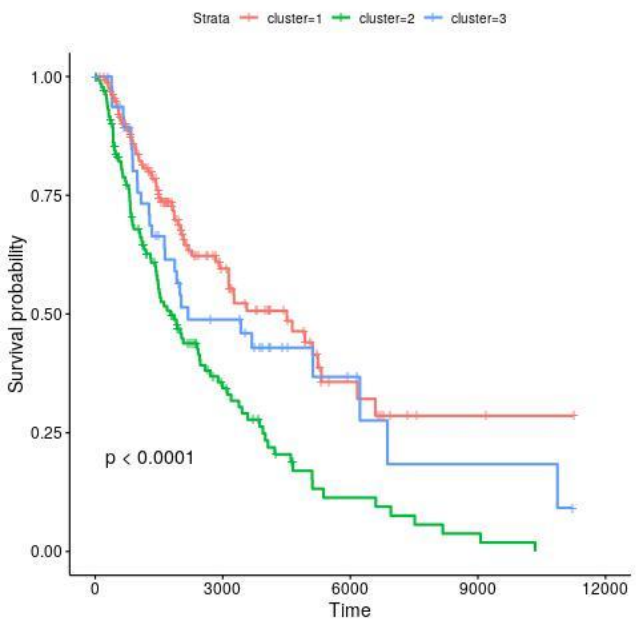
Le temps mesuré ici est donc la durée entre le diagnostic et l'événement "mort du patient". Dans le cas où le patient est toujours vivant, l'événement n'est pas encore survenu, la durée de survie mesurée n'apporte donc qu'une information partielle. On peut ici parler de censure (et plus précisément, de *censure à droite*) des données, au sens statistique. Le test de log-rank prend ce phénomène en considération et permet de traiter de telles données.

L'hypothèse nulle du test de log-rank est que les groupes testés présentent les mêmes courbes de survie.

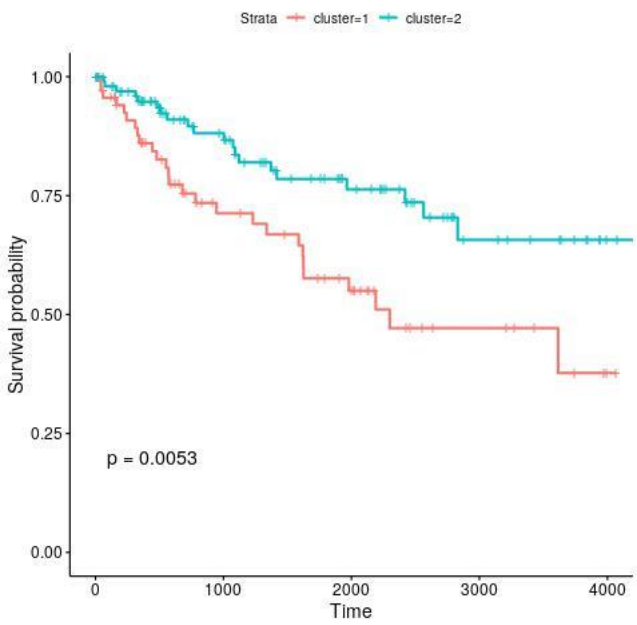
Pour cette analyse, nous avons suivi la méthodologie proposée par Rappoport et Shamir pour leur comparaison des outils de prédiction et utilisé le package R *Survival*, qui implémente divers algorithmes pour les analyses de survie, et notamment le test de log-rank.

Pour chaque clustering, la p-valeur du test de log-rank a été calculée. Afin d'avoir des résultats plus robustes, d'autres tests de log-rank ont été réalisés avec permutations : les assignations des patients à leur cluster ont été permutées aléatoirement, et pour chaque permutation, la p-valeur du test de log-rank a été calculée. On réalise ensuite un test binomial en regardant le nombre de p-valeurs permutées inférieures (et donc meilleures) à la vraie p-valeur calculée sur les données non permutées. Plus le nombre de p-valeurs permutées inférieures à la p-valeur d'origine est élevé, plus la permutation des assignations patients/cluster a permis de détecter des courbes de survie différenciées, et donc moins le clustering est pertinent. Les permutations et les tests binomiaux sont exécutés tante que l'intervalle de confiance à 95% du test binomial dépasse 0.05, ou bien jusqu'à 100 000 permutations. Une fois un de ces deux critères atteints, la probabilité de succès du test binomial est retournée et peut être interprétée comme une p-valeur : si le nombre de succès est important, la p-valeur retournée est haute, et les clusters ne sont pas pertinents vis-à-vis de l'analyse de survie. Si le nombre de succès est faible, la p-valeur retournée est faible et le clustering est pertinent.

Nous avons donc réalisé cette analyse sur tous les clusterings produits, que ce soit par les méthodes d'intégration de données multi-omiques testées ou les résultats de NeOmics. Nous avons considéré que les clusters étaient pertinents si la p-valeur retournée était inférieure à 0.05. Tous les résultats ainsi que les courbes de survie sont disponibles sur Github (https://github.com/galadrielbriere/Neomics/tree/master/results/survival). Deux exemples de courbes obtenues sont également présentés dans la figure 20.



Cancer SKCM, méthode MCCA, clustering multi-omique



Cancer KIRC, méthode NeOmics Markov, clustering *a posteriori*

Figure 20 : Courbes de survie par cluster et p-valeurs du test de log-rank.

Une analyse de survie ayant été réalisée pour chaque clustering (pour chaque omique et chaque méthode), les possibilités d'analyse et de comparaison sont nombreuses.

Les résultats obtenus seront commentés et interprétés dans leur globalité dans les parties 5.2.3 et 5.2.4.

5.2.2) Enrichissement en labels cliniques

La deuxième métrique proposée par Rappoport et Shamir pour attester de la pertinence biologique des clusters est l'enrichissement de ceux-ci en labels cliniques.

Les labels cliniques sont les métadonnées cliniques disponibles pour chaque patient : stade pathologique, âge au diagnostic, type histologique des tumeurs, etc… L'idée est que les patients atteints par un même sous-type de cancer présenteront aussi les mêmes caractéristiques cliniques, dans une certaine mesure. Ainsi, en regardant l'abondance de certains labels cliniques dans certains clusters et en remarquant que cet enrichissement est statistiquement significatif, alors on peut en déduire que le clustering produit est robuste et pertinent.

Les métadonnées sont disponibles grâce au projet TCGA sous la forme de fichiers tabulés rassemblant les annotations des médecins. Si une trame commune existe entre les différents cancers, les labels cliniques d'un cancer à l'autre varient (par exemple, historique de fumeur pour le cancer du poumon, présence de polypes pour le cancer du côlon, etc…). Les données peuvent être manquantes ou ambiguës, et l'expertise d'un spécialiste clinicien nous a été nécessaire, d'une part pour choisir les labels cliniques pertinents à traiter et d'autre part pour définir le niveau de précision à utiliser. Des précisions sur les labels cliniques choisis sont présentées en annexe 11.

Les labels cliniques pour lesquels les données étaient manquantes pour plus de la moitié des individus ont été retirés des analyses.

Afin de déterminer si les clusters sont enrichis en paramètres cliniques, nous avons, pour chacun des labels, réalisé un test du χ^2 (pour les variables discrètes) ou un test de Kruskal-Wallis (pour les variables numériques). Comme pour l'analyse de survie, nous avons permuté les assignations individu/cluster pour compter le nombre de permutations pour lesquelles la p-valeur permutée était plus faible que la p-valeur calculée avec les bonnes assignations et exécuté un test binomial jusqu'à ce que son intervalle de confiance à 95% ne dépasse pas 0.05 ou jusqu'à 100 000 itérations. On retourne alors la probabilité de succès calculée par le test binomial : plus il y a de succès, plus la permutation a permis d'obtenir des p-valeurs basses, et donc moins le clustering est enrichi par rapport au paramètre clinique étudié. Au contraire, plus la probabilité de succès est faible, plus le clustering est enrichi, et donc pertinent biologiquement.

L'enrichissement est considéré comme significatif si la p-valeur calculée est inférieure à 0.05.

Les résultats sont retournés pour chaque clustering dans un fichier contenant tous les paramètres cliniques pris en compte et leur p-valeur d'enrichissement associée. Ils sont disponibles sur Github (https://github.com/galadrielbriere/Neomics/tree/master/results/clinical_enrich).

5.2.3) Résultats pour l'intégration a posteriori

Grâce à ces deux métriques (survie et enrichissement en labels cliniques), dont les valeurs ont été calculées pour tous les clusterings obtenus, il est possible de comparer les clusterings produits par les différents outils de prédiction testés et les clusterings produits par NeOmics.

La figure 21 ci-dessous présente les résultats de survie et d'enrichissement en labels cliniques pour les clusterings multi-omiques produits par NEMO, SNF, PINS, SNF et MCCA et pour les clusterings produits par NeOmics pour l'intégration *a posteriori.* Ici, on compare donc les clusterings NeOmics (Louvain et Markov) avec les clusterings qui ont servi à produire ces résultats intégrés.

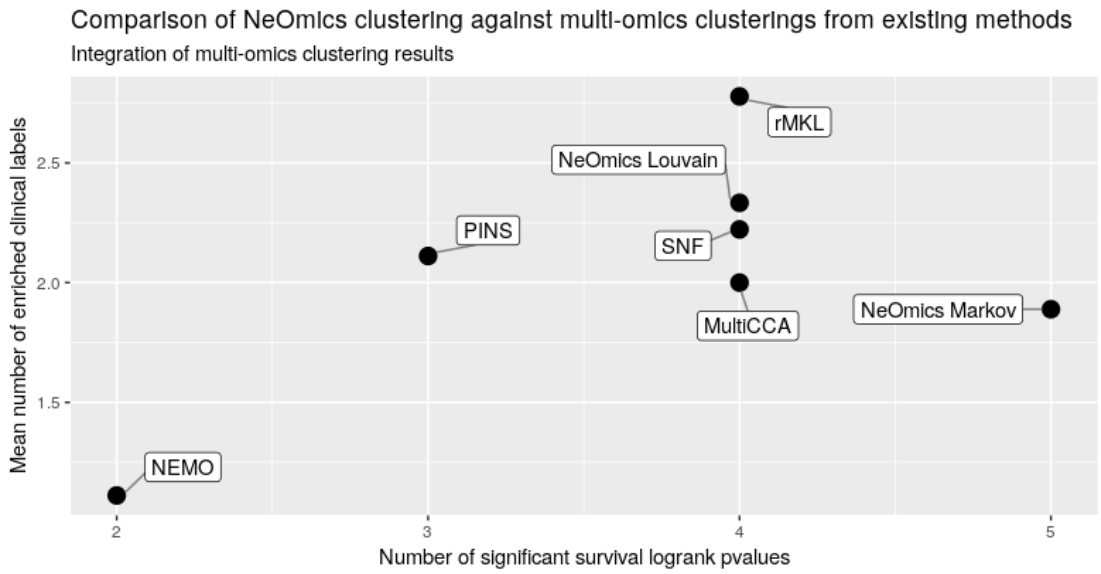


Figure 21 : Pertinence biologique des clusterings multi-omiques produits par différentes méthodes (moyenne sur 9 cancers) - intégration a posteriori

En abscisse, on représente le nombre de p-valeurs significatives de l'analyse de survie (1 p-valeur par cancer, résultats issus de 9 cancers). En ordonnée, on représente le nombre

moyen de labels cliniques enrichis (sur 9 cancers). Plus une méthode se retrouve en haut à droite sur ce graphique, plus les clusterings qu'elle a produit pour les différents cancers testés sont pertinents biologiquement.

La méthode la plus efficace du point de vue de l'enrichissement en labels cliniques est la méthode rMKL, qui a produit des clusters enrichis en moyenne sur environ 2.7 paramètres cliniques, contre un peu plus de 1 pour NEMO, la méthode la moins performante ici. Les autres méthodes ont produit des clusterings enrichis en moyenne pour 1.8 à 2.3 labels.

Sur les 9 cancers testés, NeOmics Markov a permis de produire des clusters significativement différents concernant l'analyse de survie pour 5 cancers, ce qui est la meilleure performance ici, puisque les autres méthodes ont produit des clusterings avec une p-valeur significative pour seulement 4, 3 ou 2 cancers.

On peut donc voir ici que, globalement, l'intégration des clusterings multi-omiques des différentes méthodes de prédiction existantes a permis de produire des clusterings plus significatifs d'un point de vue biologique, même si la meilleure performance en terme d'enrichissement en labels cliniques est celle de l'algorithme rMKL.

Il faut cependant noter qu'il s'agit ici d'une moyenne sur 9 cancers et que les résultats par cancer montrent une grande variabilité des performances. Je vous invite donc à consulter les annexes 2 à 10, qui rassemblent les résultats par cancer.

5.2.4) Résultats pour les intégrations a priori

Le même type de résultats a été produit pour l'intégration *a priori*. Ici, ce sont les clusterings single-omique de chaque méthode qui ont permis de produire les clusterings multi-omiques NeOmics, qui sont comparés aux clusterings multi-omiques des méthodes de prédiction existantes.

La figure 22 présente les résultats pour l'intégration *a priori*, version simple.

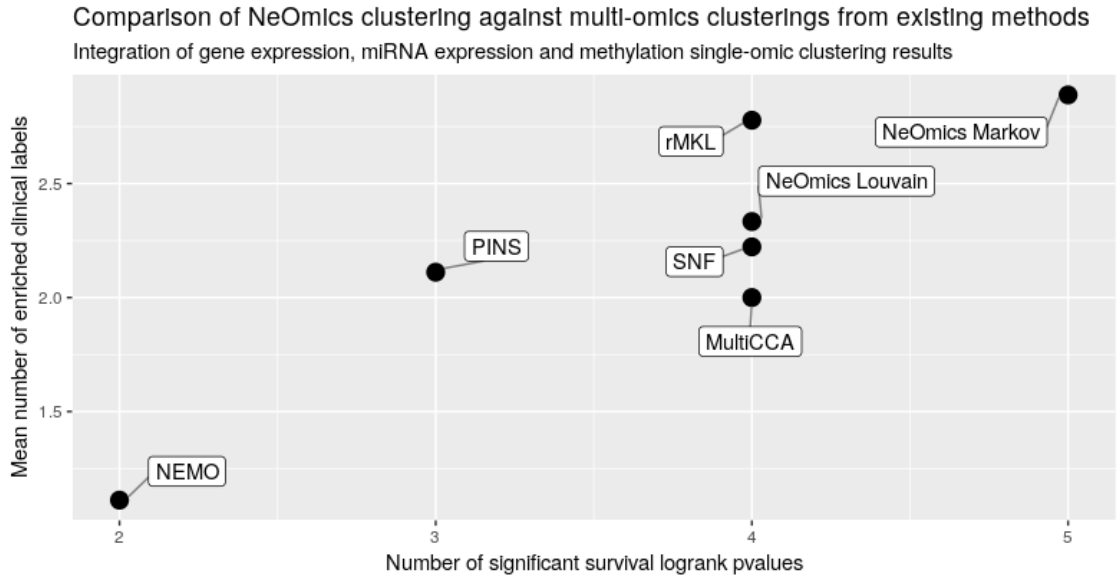


Figure 22 : Pertinence biologique des clusterings multi-omiques produits par différentes méthodes (moyenne sur 9 cancers) - intégration a priori simple

On peut voir que pour cette intégration, si les performances de NeOmics Louvain n'ont pas varié par rapport à l'intégration *a posteriori* (mêmes performances en moyenne, mais performances différentes par cancer entre les deux intégrations, *cf.* annexes 2 à 10), les performances de NeOmics Markov sont meilleures en termes d'enrichissement en labels cliniques. L'intégration *a priori*, version simple combinée à l'utilisation de l'algorithme de Markov a permis de générer les clusterings les plus robustes en moyenne, tant au niveau de l'analyse de survie que pour l'enrichissement en labels cliniques.

L'intégration *a priori* version stricte, dont les résultats sont présentés dans la figure 23, montre des résultats beaucoup plus mitigés, avec une baisse de performance à la fois pour NeOmics Markov et NeOmics Louvain, la baisse la plus importante étant observée pour le nombre de p-valeurs de survie significatives dans les clusterings NeOmics Markov : sur les 9 clusterings produits, seul un clustering présente une p-valeur significative. Cela peut être une conséquence liée au fait que certains clusterings de Markov sont déjà considérés comme non significatifs car ne rassemblant pas assez d'individus par rapport au minimum accepté (*cf.* partie 5.1.2, tableau 3, chiffres en rouges).

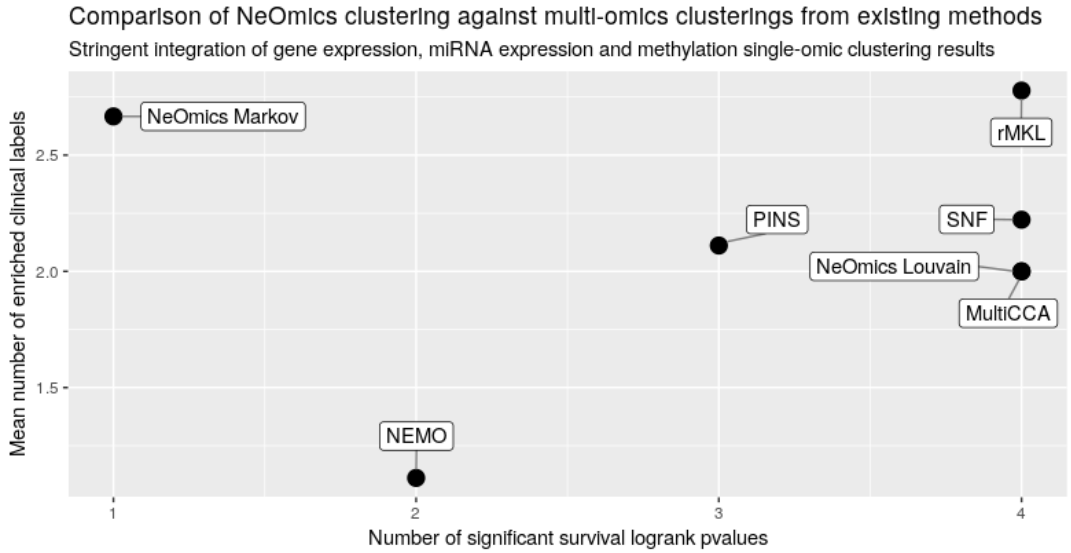


Figure 23 : Pertinence biologique des clusterings multi-omiques produits par différentes méthodes (moyenne sur 9 cancers) - intégration a priori stricte

Bilan de la partie 5.2 : Nous avons défini deux métriques pour attester de la validité biologique des clusterings produits : analyse des courbes de survie de chaque cluster et enrichissement des clusters en labels cliniques. Ces deux métriques nous permettent de comparer les clusterings issus des différents types d'intégration de NeOmics aux clusters multi-omiques produits par les outils de prédiction existants.

Il apparaît que le meilleur clustering produit est celui de NeOmics, pour l'intégration *a priori* version simple, et en utilisant l'algorithme de détection de communautés de Markov.

5.3) Synthèse : interprétation des différences de performance entre les différentes méthodes d'intégration de NeOmics

Comment expliquer les différences de performance observées pour chaque type d'intégration ?

Tout d'abord, nous pensons que l'intégration *a priori* version simple est l'intégration qui présente le plus de sensibilité, ce qui explique ses bonnes performances. En effet, c'est l'intégration pour laquelle l'étendue du nombre de supports utilisables est le plus élargi, puisqu'il peut varier de 1 à 12, contrairement à l'intégration *a priori* version stricte qui varie par pas de 3, de 3 à 12. L'intégration *a posteriori* ne permet qu'une variation du nombre de supports seuil de 1 à 5. Ce champ de variation large permet de filtrer le graphe d'intégration de manière optimale, et donc d'obtenir de meilleurs résultats.

Il faut également rappeler que l'intégration *a priori* version stricte pose une lourde contrainte pour le filtre des arêtes d'intégration, puisque les omiques doivent être en accord. On perd ainsi rapidement un grand nombre de données en voulant augmenter le nombre de supports, ce qui diminue la sensibilité de la méthode. Il faut remettre en question cette contrainte imposée, puisque, d'un point de vue biologique, il n'est pas évident que les différents omiques varient de la même manière.

D'après les résultats présentés précédemment, il semble que l'intégration *a priori* des omiques soit plus performante que l'intégration *a posteriori*. Notons d'ailleurs qu'outre les meilleurs résultats, cette méthode d'intégration permet une certaine gestion des données manquantes, ce qui est très avantageux pour exploiter le maximum des données disponibles. En effet, l'omique le plus mesuré en cancérologie reste l'expression génique et les autres types de données ne sont pas toujours disponibles.

De plus, l'intégration *a priori* nécessite uniquement des clusterings single-omique pour fonctionner. Dans notre étude, nous avons fait le choix d'utiliser des méthodes multi-omiques en leur fournissant des données single-omiques afin de ne pas faire varier les méthodes dans notre comparaison. Or, il existe de nombreux algorithmes de clustering spécialisés pour un seul omique, qui produisent potentiellement des résultats de clustering single-omique meilleurs que ceux produits avec les outils d'intégration testés. Une marge d'amélioration des résultats NeOmics est donc probable si on se base sur des clusterings single-omiques de meilleure qualité, produits par des méthodes spécialisées.

5.4) Perspectives

Plusieurs points d'amélioration et perspectives sont envisagés pour cette preuve de concept de NeOmics.

Tout d'abord, nous pensons que l'analyse de survie pour attester de la pertinence biologique des clusters peut être améliorée. En effet, cette analyse ne prend pas en compte un certain nombre de facteurs qui influent sur la survie des individus : le stade pathologique du cancer, le stade pathologique au moment du diagnostic, le traitement prescrit, l'âge de l'individu, … Ces facteurs sont susceptibles d'avoir un effet sur la survie des individus, et devraient donc être pris en compte pour améliorer la sensibilité des prédictions dans de futurs développements. Il existe d'ailleurs une méthode d'analyse de survie pouvant prendre en compte plusieurs facteurs de variation : la régression multivariée de Cox, en anglais *multivariate Cox's regression* (*Christensen, 1987*) et son implémentation dans Neomics constitue une piste intéressante.

Nous aimerions également pousser l'étude des résultats pour mieux comprendre les différences de performances observées selon les différents types d'intégration. En effet, en se basant sur les résultats produits par type de cancer disponibles dans les annexes 2 à 10, la méthode d'intégration la plus performante n'est pas toujours la même selon le cancer étudié. D'après cette observation, il serait pertinent dans de nouveaux développements de déterminer les facteurs qui influent sur les performances d'un type d'intégration : qualité des données omiques, taille de la population, qualité des clusterings sur lesquels se base la solution NeOmics, nombre de supports choisi, etc…

Il serait également intéressant de tester la méthode d'intégration *a priori* en fournissant des clusterings single-omique produits par des méthodes de clustering spécifiques au omique considéré.

Le fonctionnement méta-méthode de NeOmics pose également une contrainte importante : il nécessite de générer des clusterings à analyser et intégrer. Cela signifie qu'il faut prendre en compte le fait que toutes les méthodes testées vont nécessiter un certain temps d'exécution, auquel s'ajoute le temps de construction du graphe, des arêtes supports, des arêtes d'intégration, et du clustering de graphe. Les étapes les plus longues sont la création des arêtes support et des arêtes d'intégration, puisqu'il faut considérer toutes les paires possibles de patients et se connecter régulièrement à la base de données pour créer les arêtes. Afin de diminuer le temps d'exécution total, j'ai tenté une parallélisation du processus, puisque chaque paire de patients peut être traitée indifféremment, mais cette parallélisation a provoqué des problèmes de connexion à la base de données Neo4j. Des efforts supplémentaires doivent donc être fournis pour répondre à ce problème.

Bien sûr, le passage de NeOmics d'une preuve de concept à un outil stable et distribué à la communauté scientifique demandera aussi des développements, notamment concernant l'automatisation de la création du graphe selon les demandes de l'utilisateur (types de nœuds et arêtes à créer, attributs portés, …). Cette automatisation permettra aux utilisateurs de traiter de nouvelles questions biologiques et données en utilisant les méthodes d'intégration de NeOmics.

Conclusion

Nous avons produit une preuve de concept pour NeOmics, un outil visant à permettre la manipulation et l'intégration de données hétérogènes grâce à l'exploitation de Neo4j, une BDD orientée graphes. Ce POC complète celui produit en 2018, centré sur la question de la représentation et de la manipulation des données.

Le nouveau POC de NeOmics traite de la problématique de l'intégration des données, en se focalisant sur la question biologique de la détection de sous-type de cancer, un des challenges actuels de la médecine de précision. Trois omiques ont été intégrés, via les résultats de différentes méthodes d'intégration existantes (intégration *a posteriori*), ou via des clusterings single-omiques (intégration *a priori*). Les résultats sont très prometteurs, et montrent d'ores et déjà les avantages d'une intégration *a priori.* Ces résultats permettent ainsi d'envisager d'intégrer automatiquement des données de cancérologie, et l'exploitation de nouvelles données permettraient d'affiner et renforcer les prédictions.

Si la question biologique est ici bien définie, l'utilisation de résultats de clusterings pour produire les résultats NeOmics permet d'élargir les problématiques pouvant être traitées par notre outil. En effet, la force de NeOmics est qu'il est capable de se baser sur n'importe quels résultats de clustering, quelques soient les algorithmes de clustering utilisés initialement, le nombre de clusters ou le type de données d'origine.

Comme mentionné en début de mémoire, l'aspect générique de l'approche de NeOmics était une spécification importante. Ce nouveau POC ouvre de nombreuses perspectives et pourra être adapté à différentes données et question biologiques. Coté utilisateur, il pourrait permettre à terme une prise en main par un biologiste pour aborder différentes questions biologiques au moyen d'une interface pour formuler facilement des requêtes Cypher. Un premier pas dans cette direction a été réalisé grâce au stage de master 1 de deux étudiants que j'ai co-encadré, avec la réalisation d'une interface clique-bouton permettant la création facile et totalement personnalisable de requêtes Cypher sans avoir à connaître ce langage. Ce projet vise à permettre aux biologistes de créer leurs propres graphes et de les manipuler facilement pour répondre à leurs problématiques de recherche.

Ces différentes perspectives font partie des futurs développements que je souhaite réaliser au cours de ma thèse, qui débutera en octobre 2019.

Allard, G. (2019) '*Markov Clustering in Python'*. Available at: https://github.com/GuyAllard/markov_clustering (Accessed: 7 August 2019).

Alhamdoosh, M. *et al.* (2017) 'Combining multiple tools outperforms individual methods in gene set enrichment analyses', *Bioinformatics*, 33(3), pp. 414–424. doi: 10.1093/bioinformatics/btw623.

Ashburner, M. *et al.* (2000) 'Gene ontology: tool for the unification of biology. The Gene Ontology Consortium', *Nature Genetics*, 25(1), pp. 25–29. doi: 10.1038/75556.

Audoux, J. *et al.* (2017) 'SimBA: A methodology and tools for evaluating the performance of RNA-Seq bioinformatic pipelines', *BMC bioinformatics*, 18(1), p. 428. doi: 10.1186/s12859-017-1831-5.

Ben-Dor, A., Shamir, R. and Yakhini, Z. (1999) 'Clustering Gene Expression Patterns', *Journal of Computational Biology*, 6(3–4), pp. 281–297. doi: 10.1089/106652799318274.

Blondel, V. D. *et al.* (2008) 'Fast unfolding of communities in large networks', *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10), p. P10008. doi: 10.1088/1742-5468/2008/10/P10008.

Creusefond, J. (2015) 'A comparison of graph clustering algorithms' *International Symposium on WebAlGorithms, Jun 2015, Deauville, France.* hal-01171341 (poster disponible à l'adresse : http://iswag-symposium.org/2015/pdfs/A0-7.pdf)

Christensen, E. (1987) 'Multivariate survival analysis using Cox's regression model', *Hepatology*, 7(6), pp. 1346–1358. doi: 10.1002/hep.1840070628.

Deng, X., Eickholt, J. and Cheng, J. (2009) 'PreDisorder: ab initio sequence-based prediction of protein disordered regions', *BMC Bioinformatics*, 10(1), p. 436. doi: 10.1186/1471-2105-10-436.

Desmedt, C. *et al.* (2008) 'Biological Processes Associated with Breast Cancer Clinical Outcome Depend on the Molecular Subtypes', *Clinical Cancer Research*, 14(16), pp. 5158–5165. doi: 10.1158/1078-0432.CCR-07-4756.

De Smet, R. and Marchal, K. (2010) 'Advantages and limitations of current network inference methods', *Nature Reviews Microbiology*, 8(10), pp. 717–729. doi: 10.1038/nrmicro2419.

Dolinski, K. and Troyanskaya, O. G. (2015) 'Implications of Big Data for cell biology', *Molecular Biology of the Cell*, 26(14), pp. 2575–2578. doi: 10.1091/mbc.E13-12-0756.

European Nucleotide Archive (no date) *European Nucleotide Archive - EMBL-EBI*. Available at: https://www.ebi.ac.uk/ena (Accessed: 22 July 2019).

Fallahpour, S. *et al.* (2017) 'Breast cancer survival by molecular subtype: a population-based analysis of cancer registry data', *CMAJ Open*, 5(3), pp. E734–E739. doi: 10.9778/cmajo.20170030.

Gadgil, M. (2008) 'A Population Proportion approach for ranking differentially expressed genes', *BMC Bioinformatics*, 9(1), p. 380. doi: 10.1186/1471-2105-9-380.

Gardner, P. P. *et al.* (2016) 'A meta-analysis of bioinformatics software benchmarks reveals that publication-bias unduly influences software accuracy', *bioRxiv*, p. 092205. doi: 10.1101/092205.

Ishida, T. and Kinoshita, K. (2008) 'Prediction of disordered regions in proteins based on the meta approach', *Bioinformatics*, 24(11), pp. 1344–1348. doi: 10.1093/bioinformatics/btn195.

Kamali, A. H. *et al.* (2015) 'How to test bioinformatics software?', *Biophysical Reviews*, 7(3), pp. 343–352. doi: 10.1007/s12551-015-0177-3.

Kim, S. *et al.* (2015) 'Integrative phenotyping framework (iPF): integrative clustering of multiple omics data identifies novel lung disease subphenotypes', *BMC Genomics*, 16(1), p. 924. doi: 10.1186/s12864-015-2170-4.

Lapatas, V. *et al.* (2015) *Data integration in biological research: An overview*. doi: 10.1186/s40709-015-0032-5.

National Center for Biotechnology Information (no date) *National Center for Biotechnology Information*. Available at: https://www.ncbi.nlm.nih.gov/ (Accessed: 22 July 2019).

National Cancer Institute (2018) *The Cancer Genome Atlas Program*, *National Cancer Institute*. Available at: https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga (Accessed: 22 July 2019).

Nguyen, T. *et al.* (2017) 'A novel approach for data integration and disease subtyping', *Genome Research*, 27(12), pp. 2025–2039. doi: 10.1101/gr.215129.116.

Rapakoulia, T. *et al.* (2014) 'EnsembleGASVR: a novel ensemble method for classifying missense single nucleotide polymorphisms', *Bioinformatics*, 30(16), pp. 2324–2333. doi: 10.1093/bioinformatics/btu297.

Rappoport, N. and Shamir, R. (2018) 'Multi-omic and multi-view clustering algorithms: review and cancer benchmark', *Nucleic Acids Research*, 46(20), pp. 10546–10562. doi: 10.1093/nar/gky889.

Rappoport, N. and Shamir, R. (2019) 'NEMO: cancer subtyping by integration of partial multi-omic data', *Bioinformatics*. doi: 10.1093/bioinformatics/btz058.

Speicher, N. K. and Pfeifer, N. (2015) 'Integrating different data types by regularized unsupervised multiple kernel learning with application to cancer subtype discovery', *Bioinformatics*, 31(12), pp. i268–i275. doi: 10.1093/bioinformatics/btv244.

Tsolis, A. C. *et al.* (2013) 'A Consensus Method for the Prediction of "Aggregation-Prone" Peptides in Globular Proteins', *PLOS ONE*, 8(1), p. e54175. doi: 10.1371/journal.pone.0054175.

Väremo, L., Nielsen, J. and Nookaew, I. (2013) 'Enriching the gene set analysis of genome-wide data by incorporating directionality of gene expression and combining statistical hypotheses and methods', *Nucleic Acids Research*, 41(8), pp. 4378–4391. doi: 10.1093/nar/gkt111.

Van Dongen, S. M. (2000) '*Graph clustering by flow simulation'*. PhD Thesis. University of Utrecht, Netherlands.

Wang, B. *et al.* (2014) 'Similarity network fusion for aggregating data types on a genomic scale', *Nature Methods*, 11(3), pp. 333–337. doi: 10.1038/nmeth.2810.

Watts, D. J. and Strogatz, S. H. (1998) 'Collective dynamics of "small-world" networks', *Nature*, 393(6684), pp. 440–442. doi: 10.1038/30918.

Witten, D. M. and Tibshirani, R. J. (2009) 'Extensions of Sparse Canonical Correlation Analysis with Applications to Genomic Data', *Statistical Applications in Genetics and Molecular Biology*, 8(1), pp. 1–27. doi: 10.2202/1544-6115.1470.

Xia, J.-F., Zhao, X.-M. and Huang, D.-S. (2010) 'Predicting protein–protein interactions from protein sequences using meta predictor', *Amino Acids*, 39(5), pp. 1595–1599. doi: 10.1007/s00726-010-0588-1.

Yu, Z., Wong, H.-S. and Wang, H. (2007) 'Graph-based consensus clustering for class discovery from gene expression data', *Bioinformatics*, 23(21), pp. 2888–2896. doi: 10.1093/bioinformatics/btm463.

Yugi, K. *et al.* (2016) 'Trans-Omics: How To Reconstruct Biochemical Networks Across Multiple "Omic" Layers', *Trends in Biotechnology*, 34(4), pp. 276–290. doi: 10.1016/j.tibtech.2015.12.013.

Annexes

1. Poster présenté pour les Journées Ouvertes Biologie, Informatique et Mathématiques (Juillet 2019)

Voir page suivante.

NeOmics

Development of a new method to integrate heterogeneous data and methods

Galadriel BRIERE [1*], Ludovic LÉAUTÉ [1], Élodie DARBO [2], Raluca URICARU [1], Patricia THÉBAULT [1]

(1) Univ. Bordeaux, CNRS UMR 5800, LaBRI, F-33000 Bordeaux, France
(2) ACTION - Actions for OnCogenesis understanding and Target Identification in ONcology
* Corresponding author : galadriel.briere@gmail.com

Abstract : Thanks to the recent advances in high throughput methods, many types of omics data are now routinely collected. Methods to compute single-omic dataset clusterings are widely used and have proven to be efficient for biological and medical research. However, using all available omic datasets to cluster data might reveal sharper structures as each omic carries different and complementary information. Herein, we are developing **NeOmics**, a tool that aims at integrating multi-omics data using results by combining multiple omics clusterings. NeOmics helps you to both integrate heterogeneous data and combine methods, resulting in a consensus result. We used NeOmics to predict cancer subtypes, analyzing data from 3 different omics and 10 cancer types.

Data and existing analysis methods

3 omics : gene and miRNA expression, methylation data
10 cancer types, from 170 to 630 patients
4 methods : PINS [1], SNF [2], rMKL [3], MCCA [4]

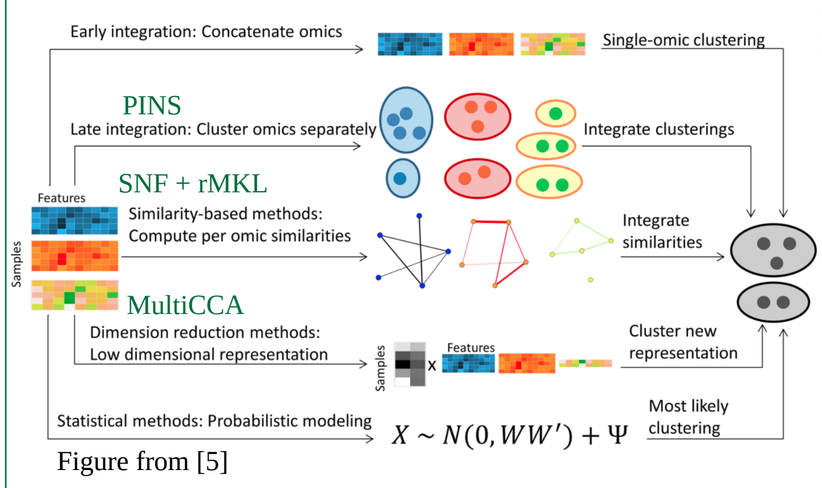
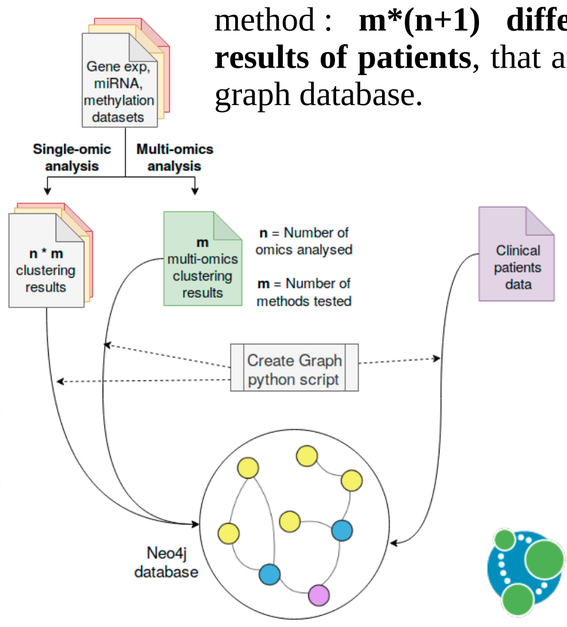


Figure from [5]

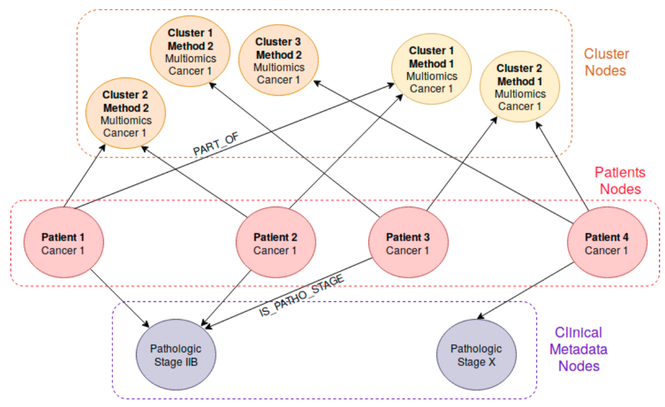
Pipeline

Computation of single and multi-omics analysis for each method : **m*(n+1) different clustering results of patients**, that are integrated in a graph database.



Data Model

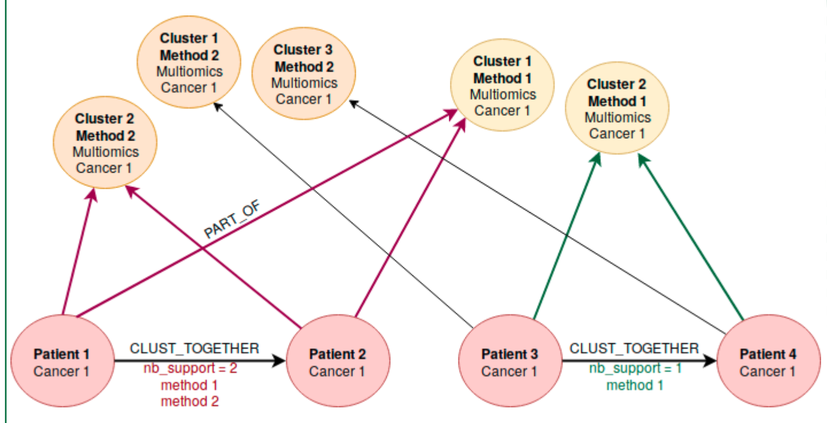
Clustering results and metadata are represented using specific nodes and relationships.



NeOmics relies on a graph-oriented database **Neo4j** to make advantage of:

- Evolutive data model
- Handling highly connected data
- Intuitive and efficient query language Cypher

Creation of a consensus multi-omic clustering : integration of methods



An edge labeled “CLUST_TOGETHER” is created between **a pair of patients if they belong to the same cluster in at least one multi-omics clustering result**. The name of the used method becomes a property of the edge. Moreover, the number of different methods supporting the pair is also given as a number of support.

The graph can be queried using the “CLUST_TOGETHER” relationship :

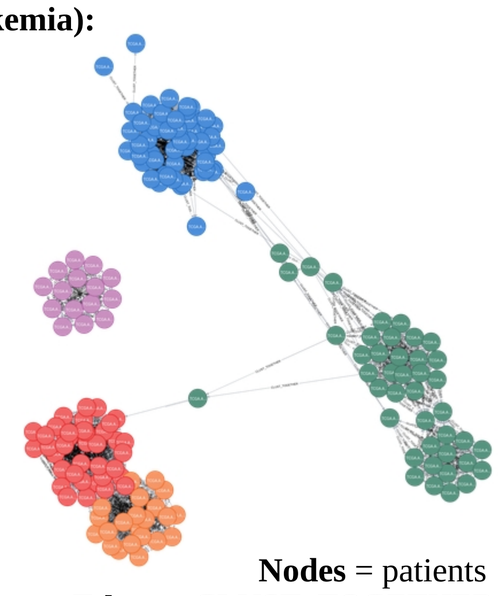
```
MATCH (p1:Patient)-[r:CLUST_TOGETHER]-(p2:Patient)
WHERE r.nb_support >=3
RETURN p1, r, p2
```

Pairs of patients are returned if **the patients constituting the pair belong to the same cluster for at least 3 different methods** (over 4).

This Cypher query returns a complex subgraph where groups of patients clustered together in several methods are **densely connected** : those groups of patients can be detected using **community detection algorithms** such as the Louvain algorithm or the Markov Cluster algorithm.

The communities returned by Louvain or Markov Clusters algorithms are **consensus clusters**, as they have been computed using the common results of the tested methods.

Louvain communities for AML cancer (leukemia):



Nodes = patients
Edges = CLUST_TOGETHER relationship with number of support >= 3

Comparing clusterings : survival rate & clinical label enrichment

Clinical Labels Enrichment

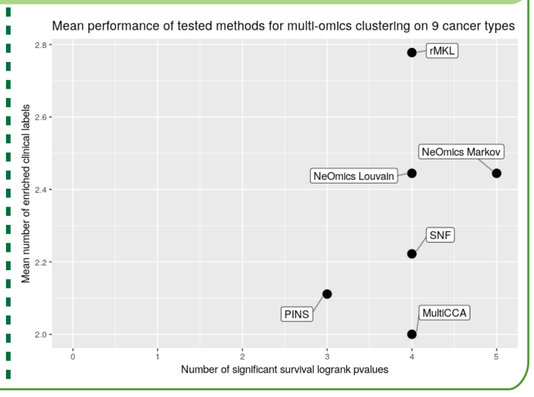
Pan-cancer and cancer-specific clinical labels (pathologic stage, age at diagnosis, smoking history for lung cancer, presence of polyps for colon cancer, …) have been selected to interpret results.

The **enrichment of these clinical labels in the clusters have been computed** using χ^2 test for discrete parameters and Krukal-Wallis test for numeric parameters. **Permutation tests** have been carried out to estimate p-values as described in [5].

Survival Analysis

It has been shown that molecular cancer subtype is a significant contributor to the hazard of death.

We measured **differential survival rates between the computed clusters** using the **log-rank test** as described in [5].



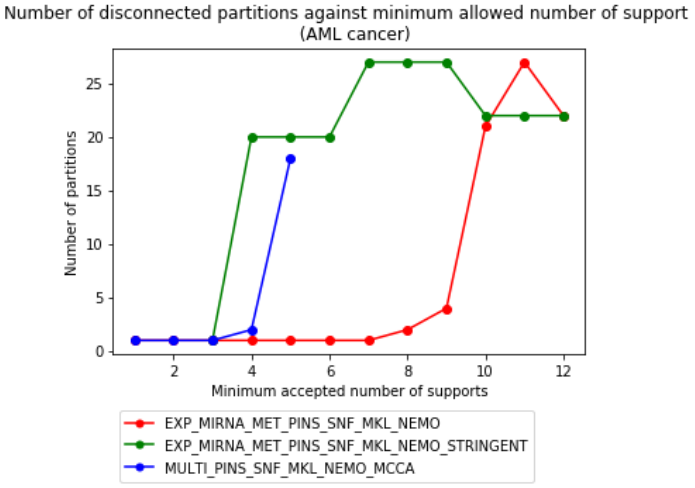
Conclusion : NeOmics tool aims at integrating methods and heterogeneous data such as different omics measurements and biological knowledge on the objects considered. NeOmics allows you to query and visualize data to find out patterns of interests. The strength of NeOmics is given by the representation of heterogeneous data through graphs, adapting graph theory algorithms to an evolutive data model and helping to address various biological and methodological questions.

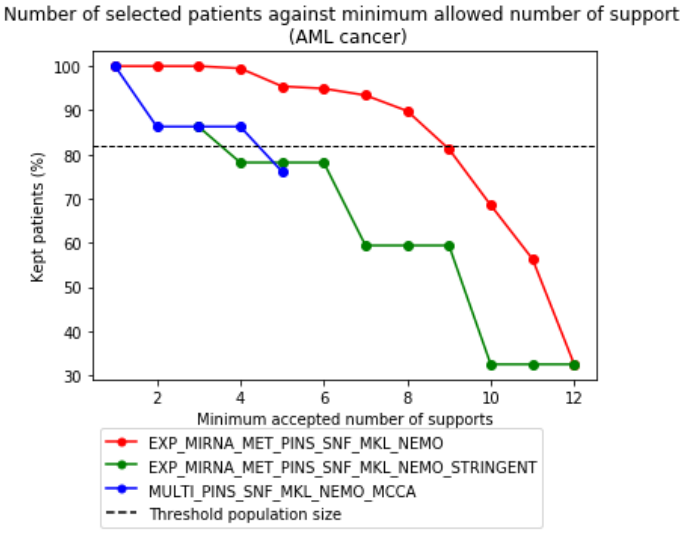
From single-omic to multi-omics clusters ? The actual version of NeOmics performs integration of methods and calculate consensus results. In order to simultaneously integrate omics and methods within NeOmics, we will develop a new approach using multiple single-omic clustering results from various methods. This approach will be based on the number of different omics for which a pair of patient have been classified in the same group, using the results of all the tested methods. Louvain and Markov graph clustering algorithms will then identify groups of patients supported by various omics datatypes and methods, resulting in a consensus multi-omic clustering.

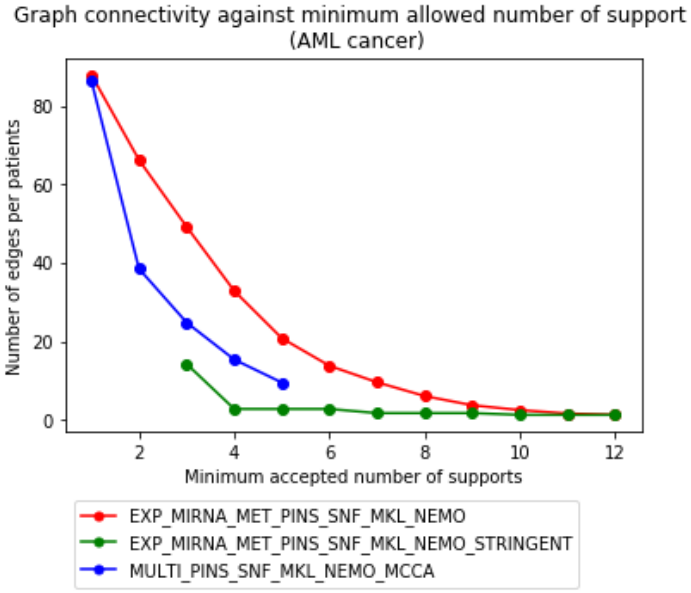
Acknowledgments : This work is supported by the LaBRI (UMR 5800).

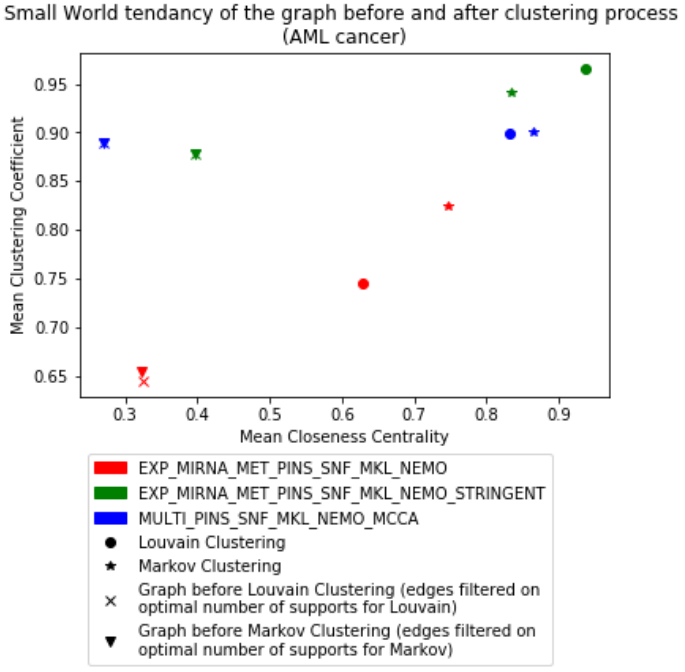
[1] NGUYEN, Tin, TAGETT, Rebecca, DIAZ, Diana, et al. A novel approach for data integration and disease subtyping. Genome research, 2017, vol. 27, no 12, p. 2025-2039.
[2] WANG, Bo, MEZLINI, Aziz M., DEMIR, Feyyaz, et al. Similarity network fusion for aggregating data types on a genomic scale. Nature methods, 2014, vol. 11, no 3, p. 333.
[3] SPEICHER, Nora K. et PFEIFER, Nico. Integrating different data types by regularized unsupervised multiple kernel learning with application to cancer subtype discovery. Bioinformatics, 2015, vol. 31, no 12, p. i268-i275.
[4] WITTEN, Daniela M. et TIBSHIRANI, Robert J. Extensions of sparse canonical correlation analysis with applications to genomic data. Statistical applications in genetics and molecular biology, 2009, vol. 8, no 1, p. 1-27.
[5] RAPPOPORT, Nimrod et SHAMIR, Ron. Multi-omic and multi-view clustering algorithms: review and cancer benchmark. Nucleic acids research, 2018, vol. 46, no 20, p. 10546-10562.

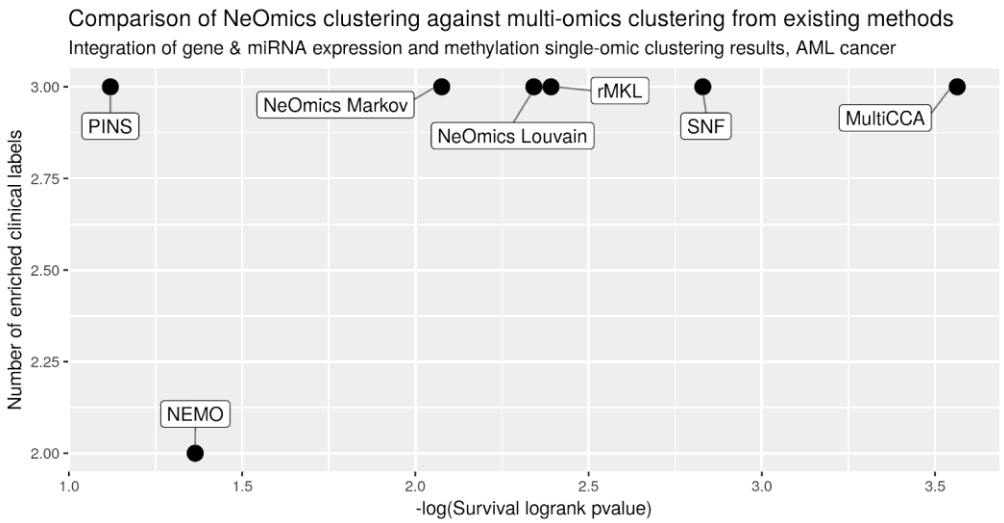
2. Résultats pour la leucémie (AML)

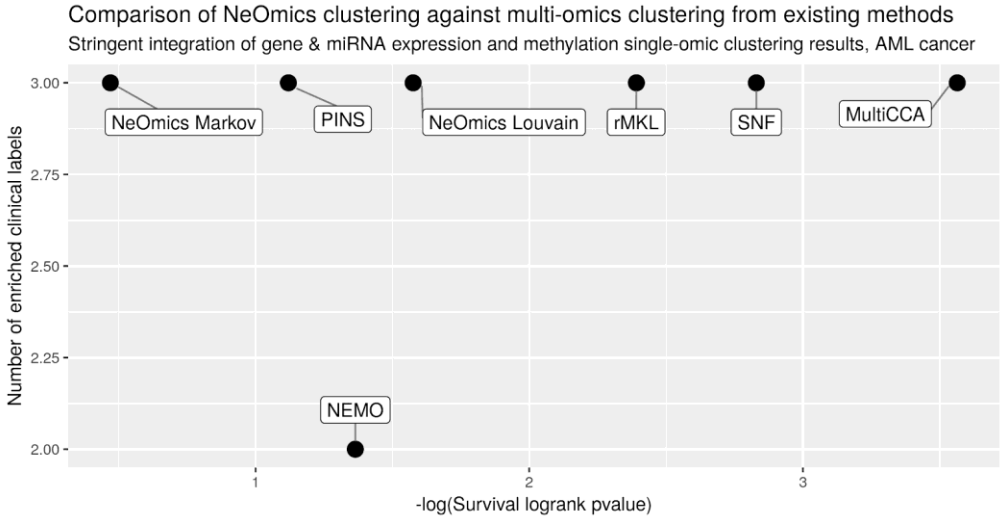


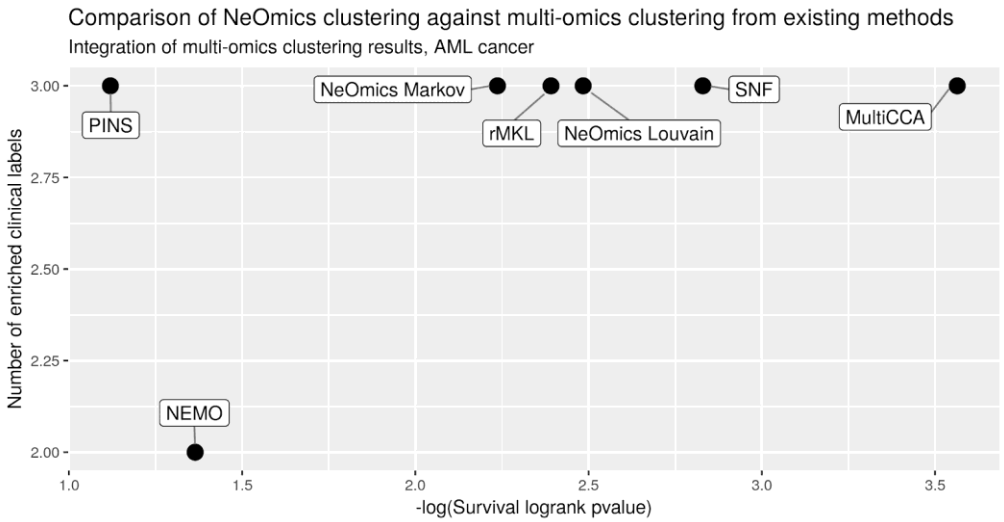




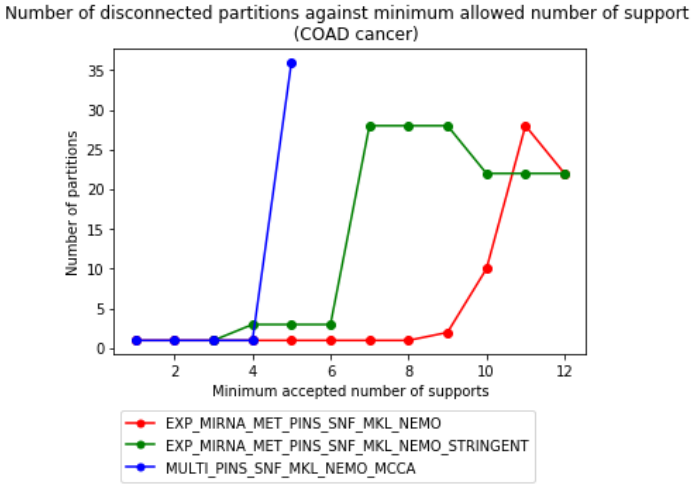


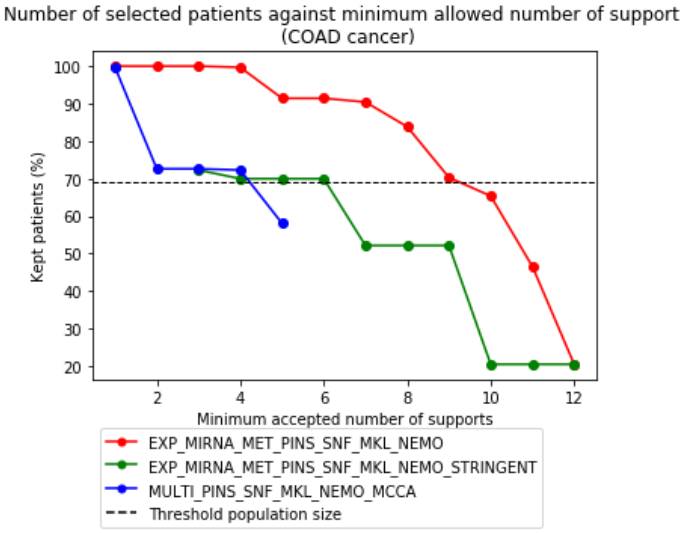

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of gene & miRNA expression and methylation single-omic clustering results, AML cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
PINS
NeOmics Markov
NeOmics Louvain
rMKL
SNF
MultiCCA
NEMO

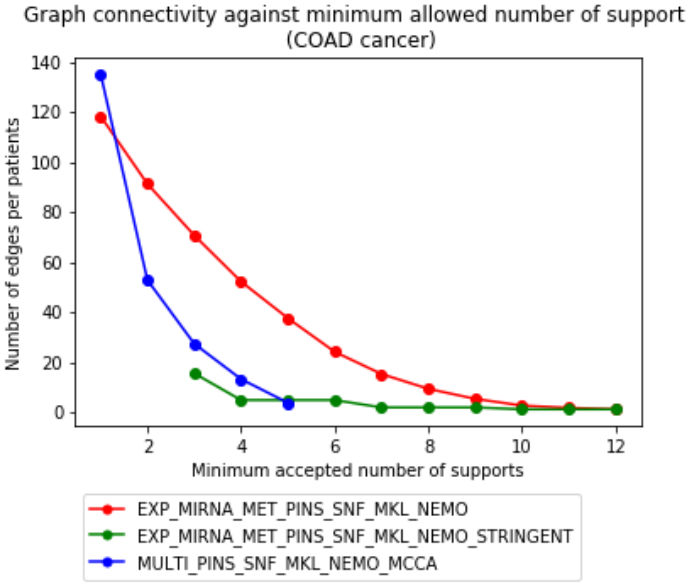

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Stringent integration of gene & miRNA expression and methylation single-omic clustering results, AML cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NeOmics Markov
PINS
NeOmics Louvain
rMKL
SNF
MultiCCA
NEMO

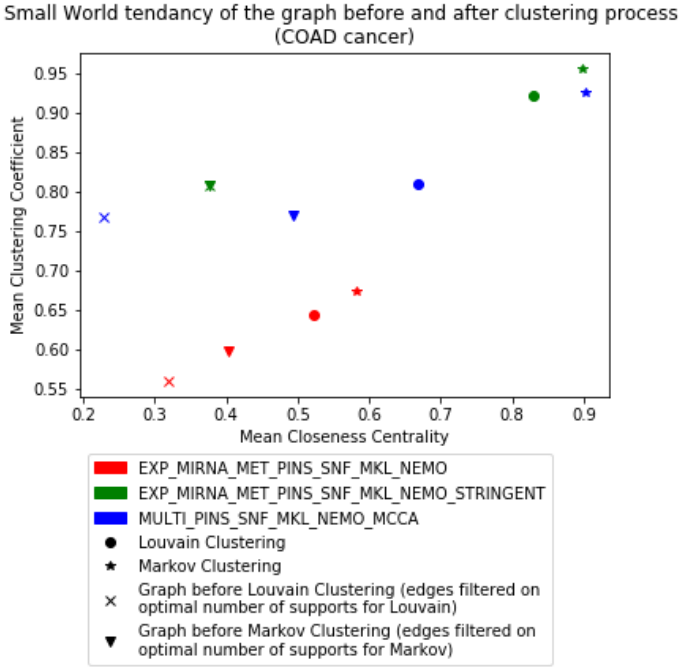

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of multi-omics clustering results, AML cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
PINS
NeOmics Markov
rMKL
NeOmics Louvain
SNF
MultiCCA
NEMO

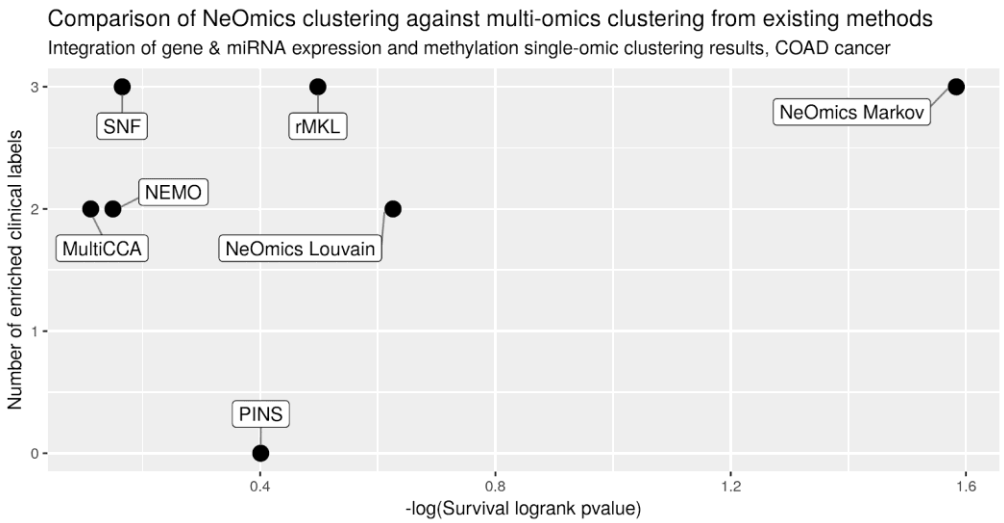
3. Résultats pour le cancer du côlon (COAD)

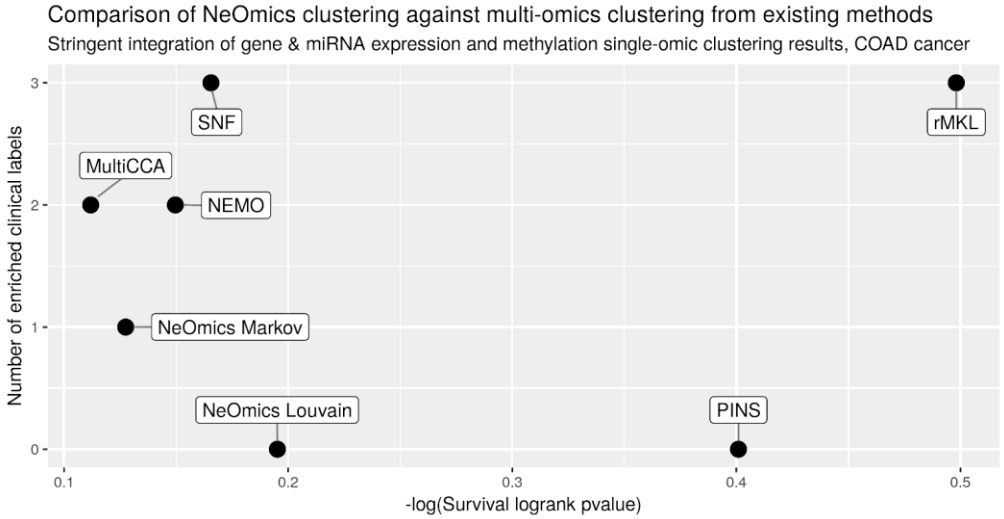


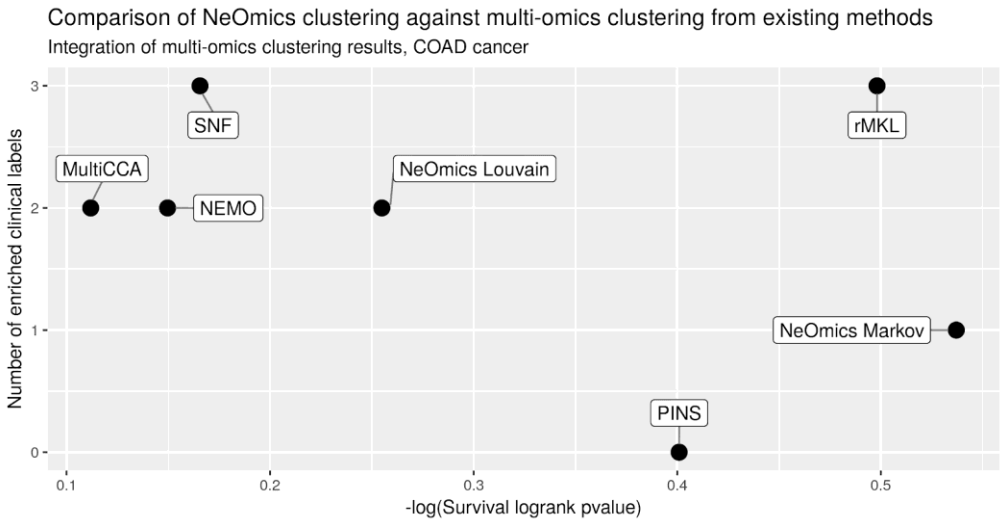


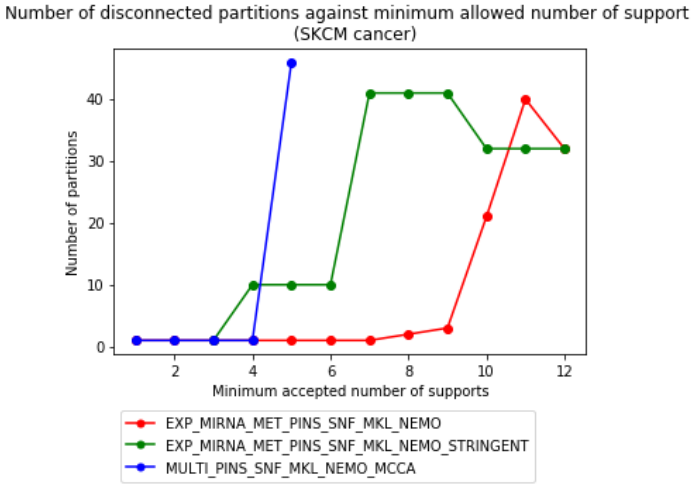


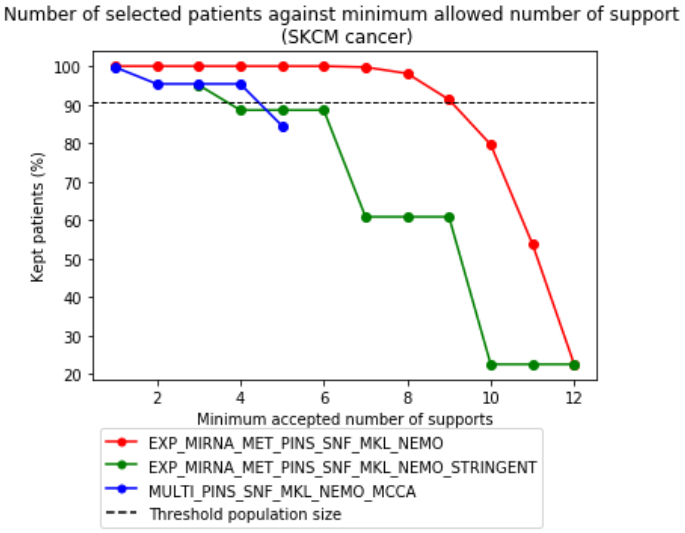


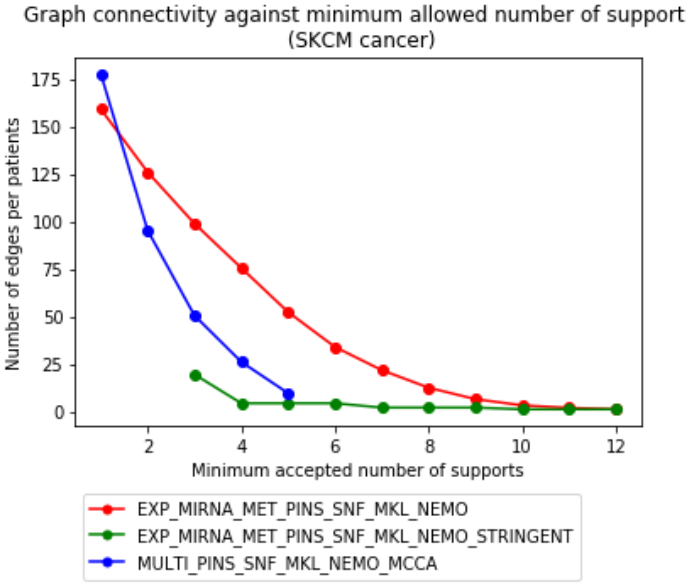

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of gene & miRNA expression and methylation single-omic clustering results, COAD cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
SNF
rMKL
NeOmics Markov
NEMO
MultiCCA
NeOmics Louvain
PINS

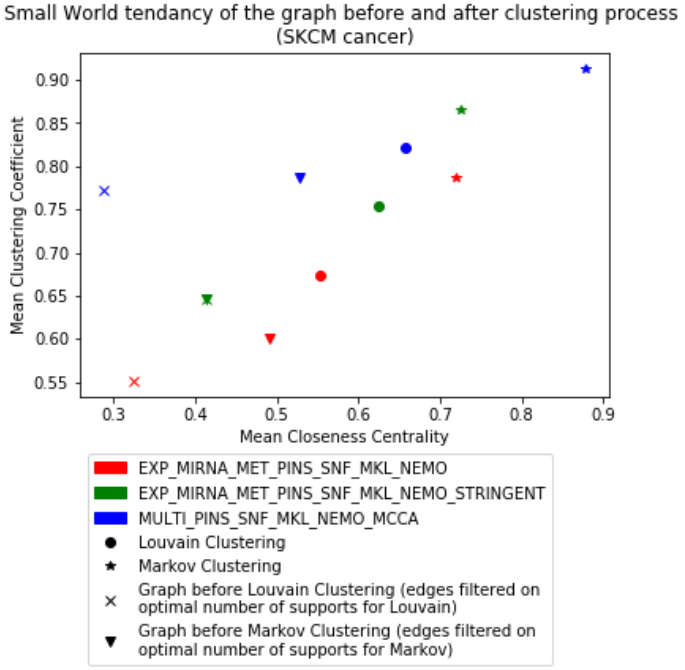

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Stringent integration of gene & miRNA expression and methylation single-omic clustering results, COAD cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
SNF
rMKL
MultiCCA
NEMO
NeOmics Markov
NeOmics Louvain
PINS

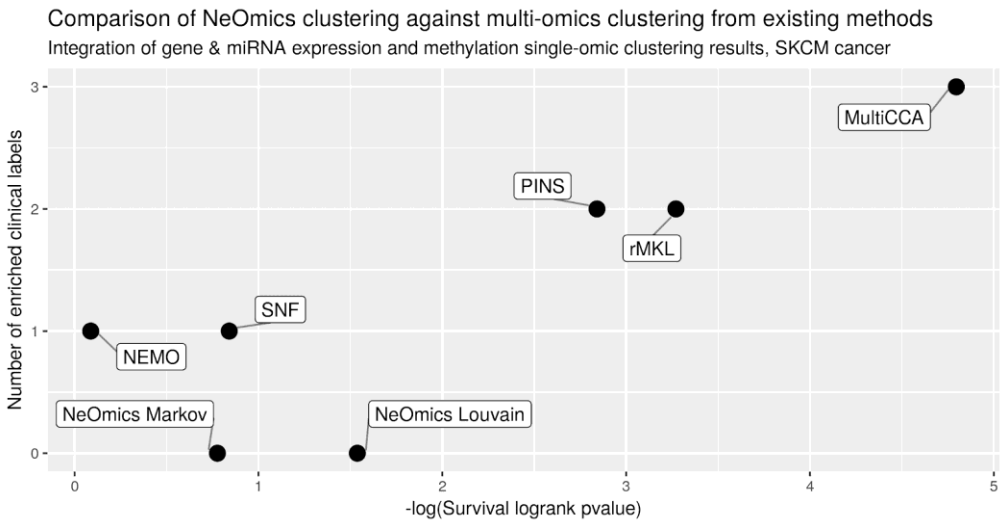

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of multi-omics clustering results, COAD cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
SNF
rMKL
MultiCCA
NeOmics Louvain
NEMO
NeOmics Markov
PINS

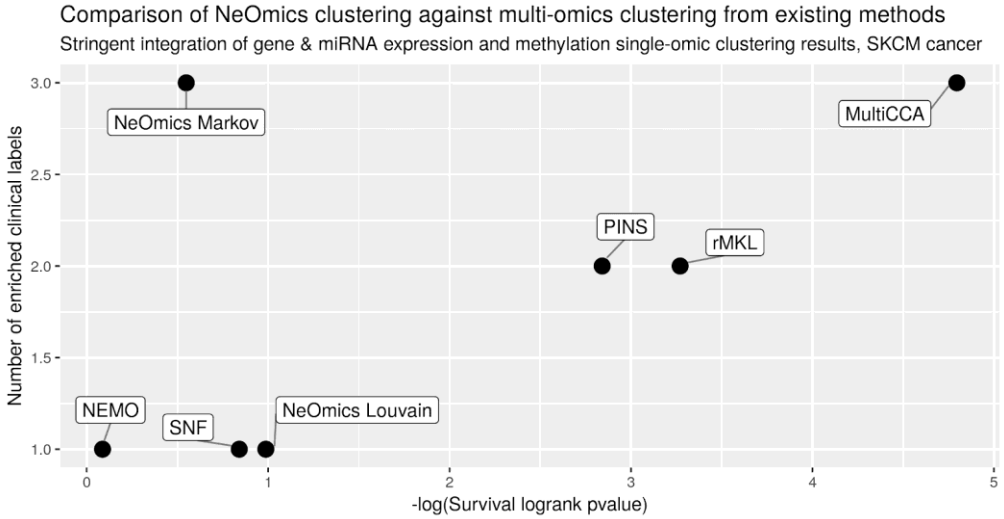

Number of disconnected partitions against minimum allowed number of support (SKCM cancer)
Number of partitions
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

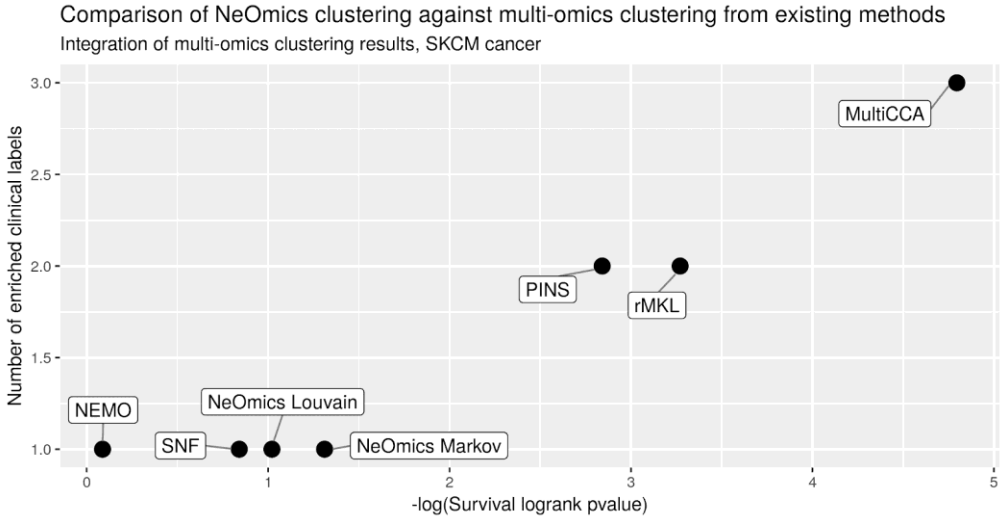

Number of selected patients against minimum allowed number of support (SKCM cancer)
Kept patients (%)
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Threshold population size

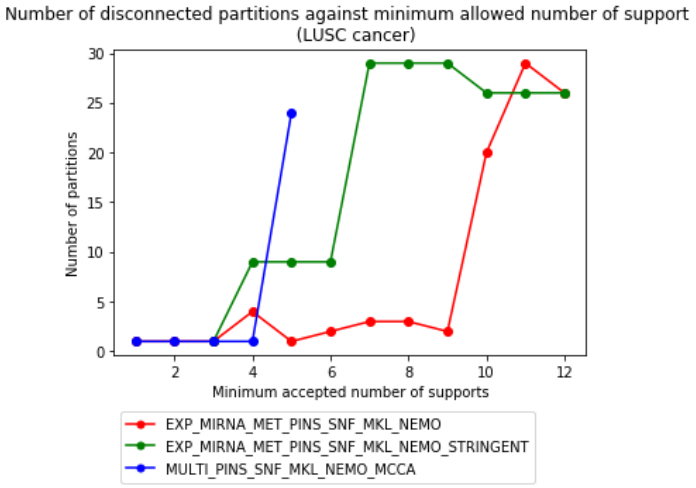

Graph connectivity against minimum allowed number of support (SKCM cancer)
Number of edges per patients
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

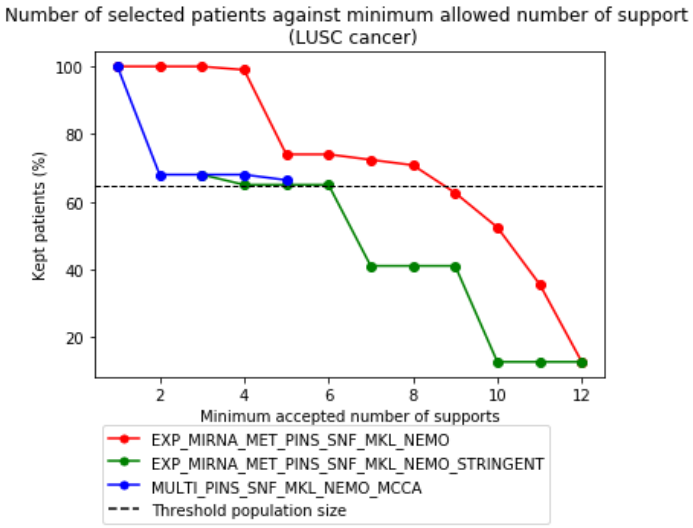

Small World tendancy of the graph before and after clustering process (SKCM cancer)
Mean Clustering Coefficient
Mean Closeness Centrality
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Louvain Clustering
Markov Clustering
Graph before Louvain Clustering (edges filtered on optimal number of supports for Louvain)
Graph before Markov Clustering (edges filtered on optimal number of supports for Markov)

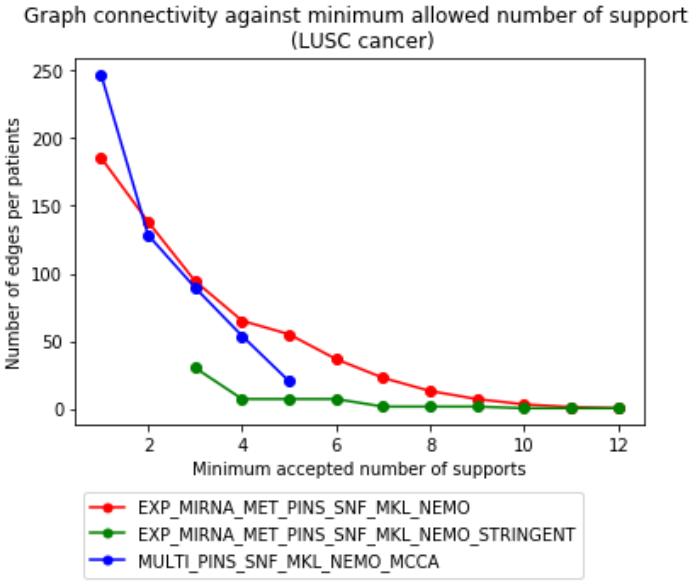

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of gene & miRNA expression and methylation single-omic clustering results, SKCM cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
MultiCCA
PINS
rMKL
SNF
NEMO
NeOmics Markov
NeOmics Louvain

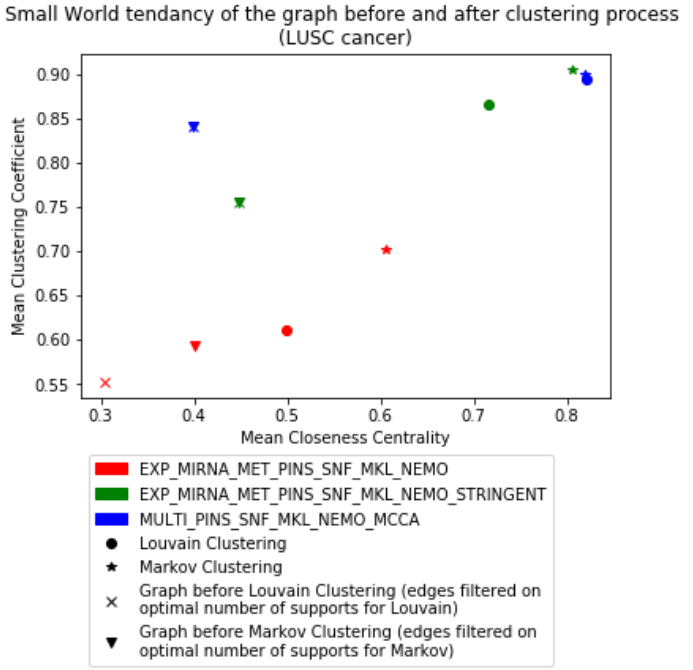

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Stringent integration of gene & miRNA expression and methylation single-omic clustering results, SKCM cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NeOmics Markov
MultiCCA
PINS
rMKL
NEMO
SNF
NeOmics Louvain

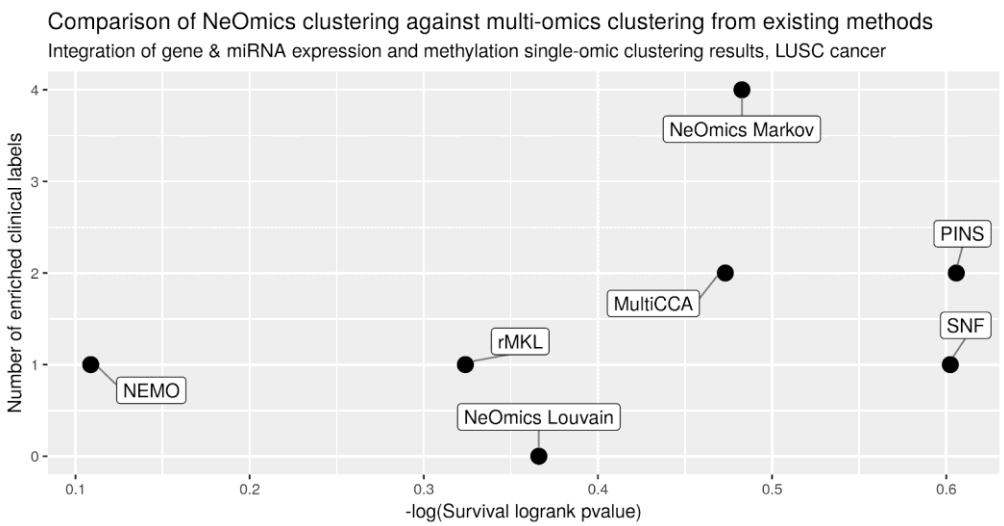

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of multi-omics clustering results, SKCM cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
MultiCCA
PINS
rMKL
NEMO
NeOmics Louvain
SNF
NeOmics Markov

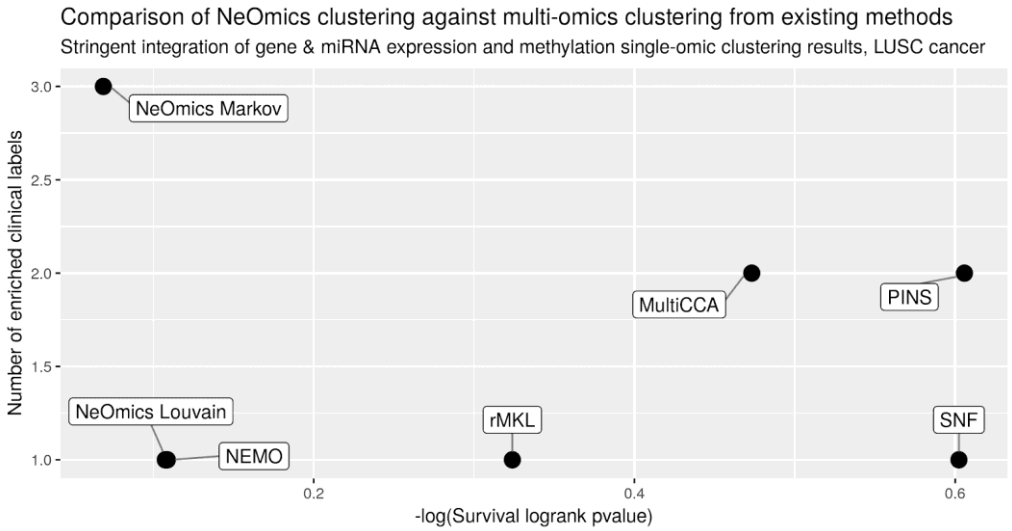

Number of disconnected partitions against minimum allowed number of support (LUSC cancer)
Number of partitions
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

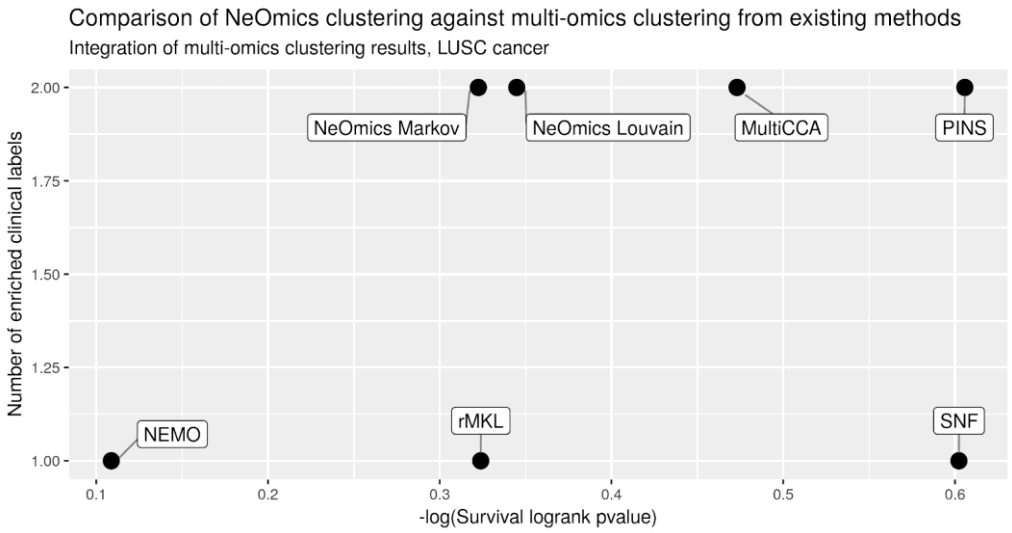

Number of selected patients against minimum allowed number of support (LUSC cancer)
Kept patients (%)
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Threshold population size

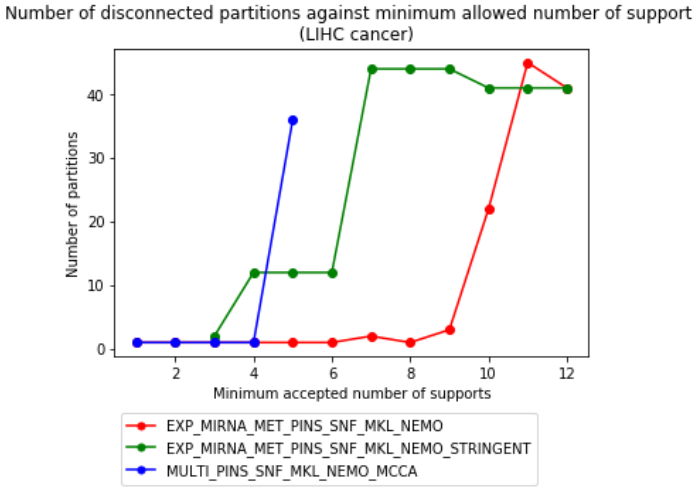

Graph connectivity against minimum allowed number of support (LUSC cancer)
Number of edges per patients
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

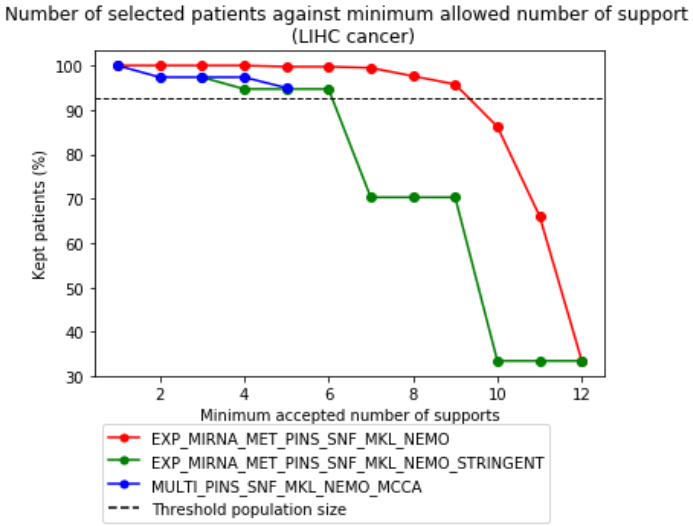

Small World tendancy of the graph before and after clustering process (LUSC cancer)
Mean Clustering Coefficient
Mean Closeness Centrality
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Louvain Clustering
Markov Clustering
Graph before Louvain Clustering (edges filtered on optimal number of supports for Louvain)
Graph before Markov Clustering (edges filtered on optimal number of supports for Markov)

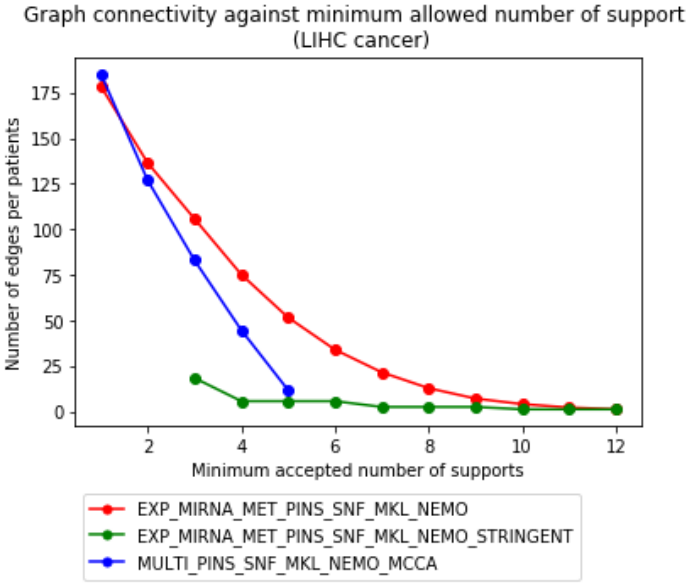

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of gene & miRNA expression and methylation single-omic clustering results, LUSC cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NeOmics Markov
PINS
MultiCCA
SNF
rMKL
NEMO
NeOmics Louvain

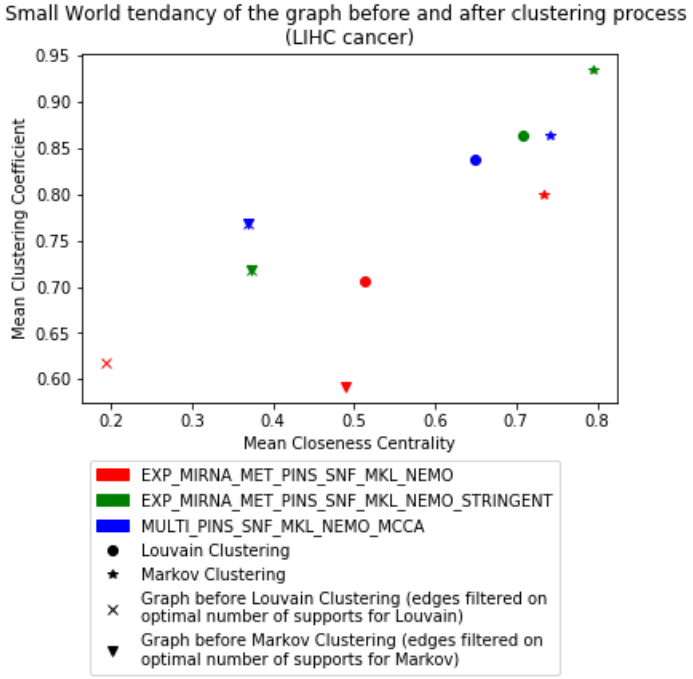

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Stringent integration of gene & miRNA expression and methylation single-omic clustering results, LUSC cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NeOmics Markov
MultiCCA
PINS
NeOmics Louvain
NEMO
rMKL
SNF

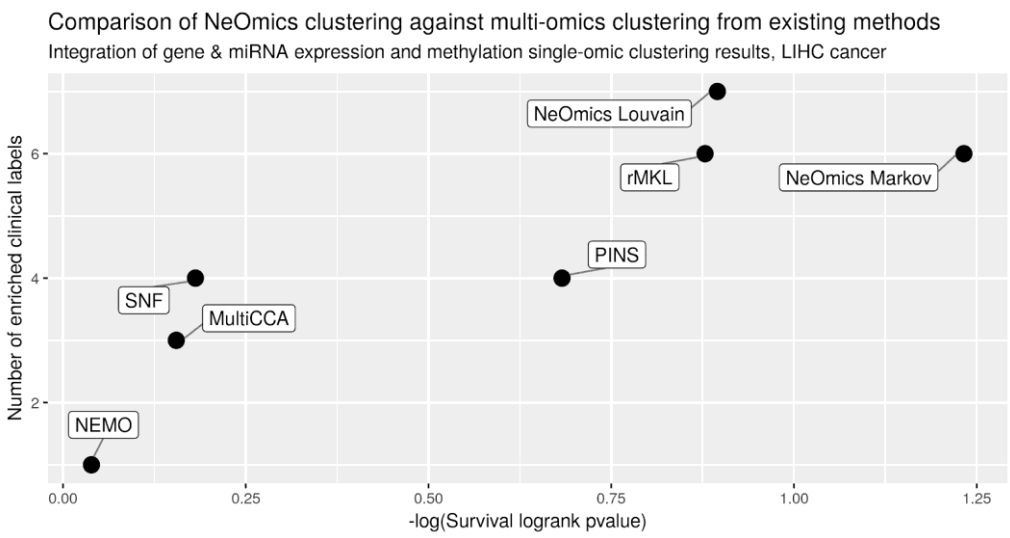

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of multi-omics clustering results, LUSC cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NeOmics Markov
NeOmics Louvain
MultiCCA
PINS
NEMO
rMKL
SNF

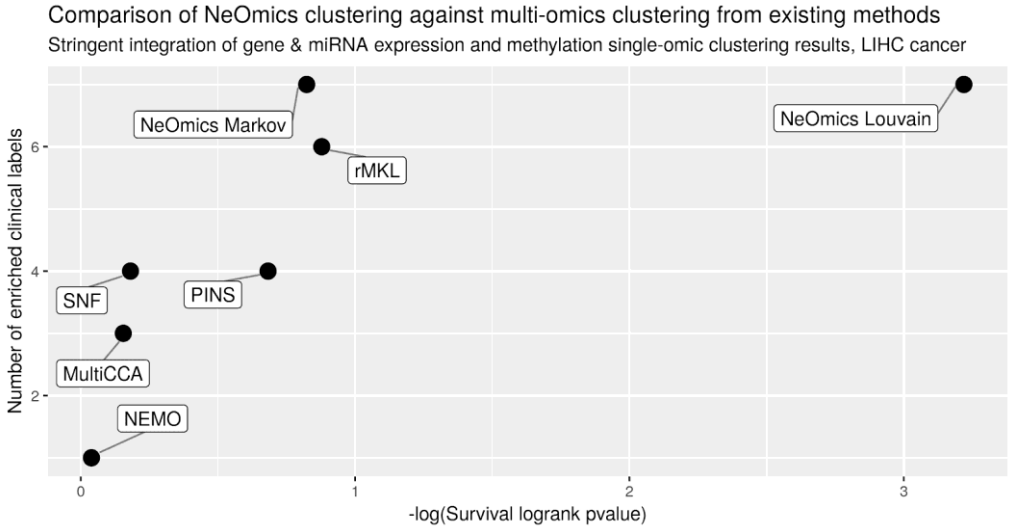

Number of disconnected partitions against minimum allowed number of support (LIHC cancer)
Number of partitions
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

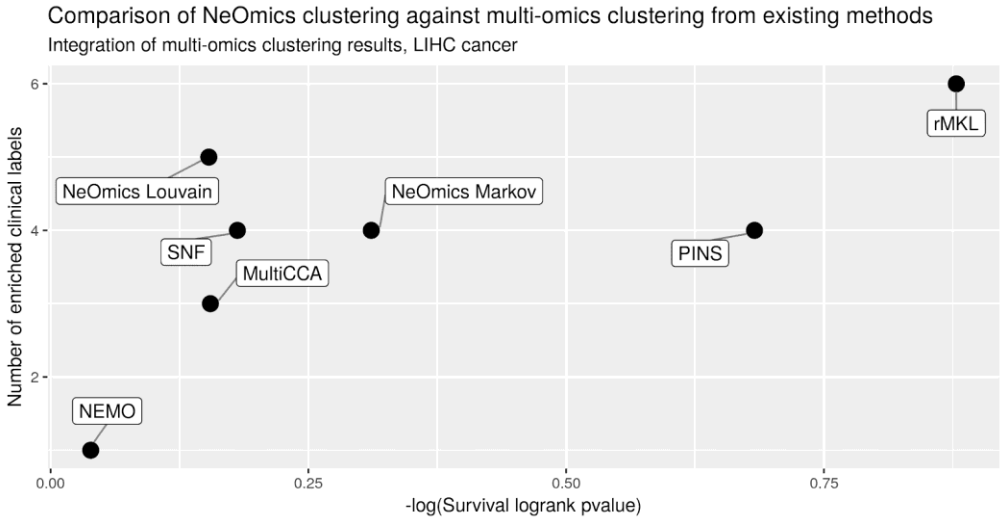

Number of selected patients against minimum allowed number of support (LIHC cancer)
Kept patients (%)
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Threshold population size

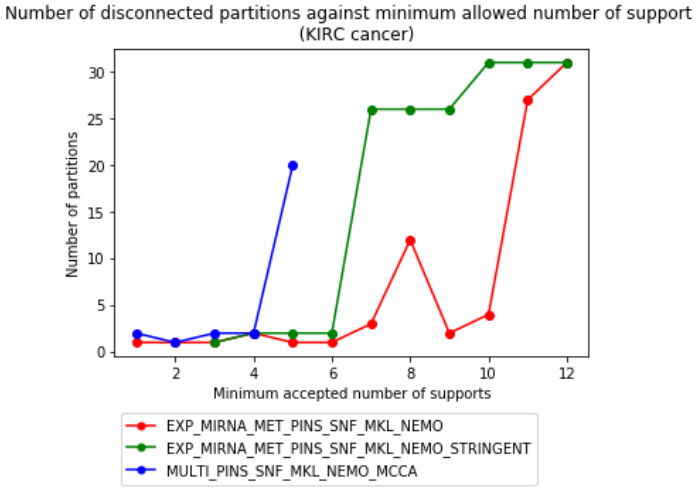

Graph connectivity against minimum allowed number of support (LIHC cancer)
Number of edges per patients
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

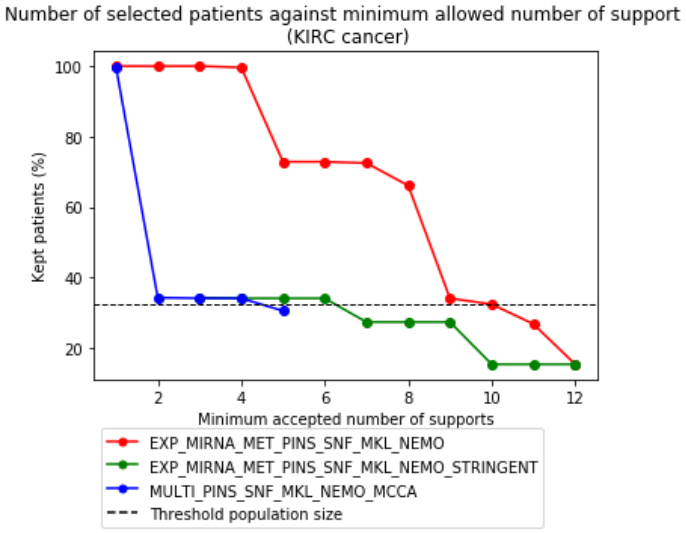

Small World tendancy of the graph before and after clustering process (LIHC cancer)
Mean Clustering Coefficient
Mean Closeness Centrality
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Louvain Clustering
Markov Clustering
Graph before Louvain Clustering (edges filtered on optimal number of supports for Louvain)
Graph before Markov Clustering (edges filtered on optimal number of supports for Markov)

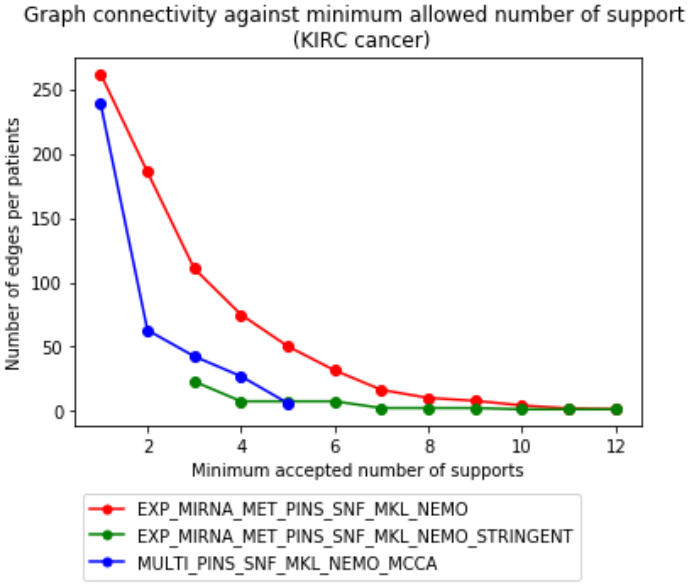

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of gene & miRNA expression and methylation single-omic clustering results, LIHC cancer
NeOmics Louvain
rMKL
NeOmics Markov
PINS
SNF
MultiCCA
NEMO
Number of enriched clinical labels
-log(Survival logrank pvalue)

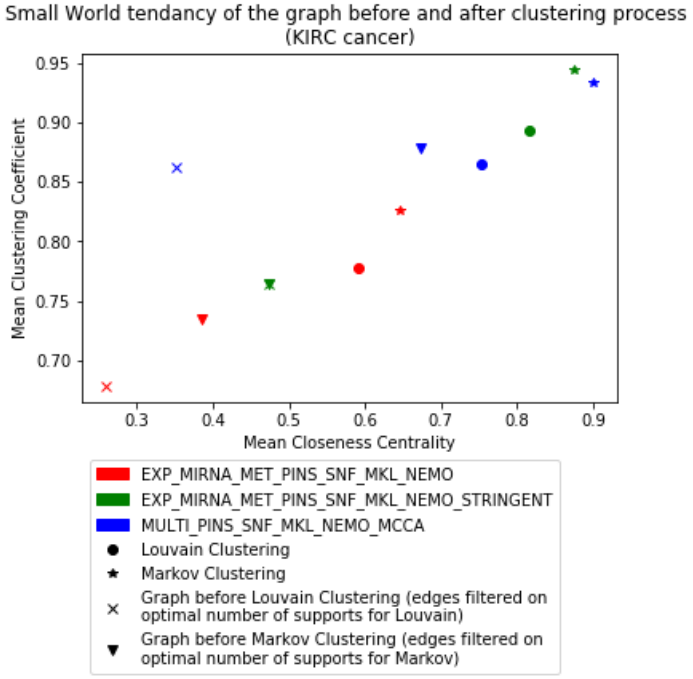

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Stringent integration of gene & miRNA expression and methylation single-omic clustering results, LIHC cancer
NeOmics Markov
NeOmics Louvain
rMKL
SNF
PINS
MultiCCA
NEMO
Number of enriched clinical labels
-log(Survival logrank pvalue)

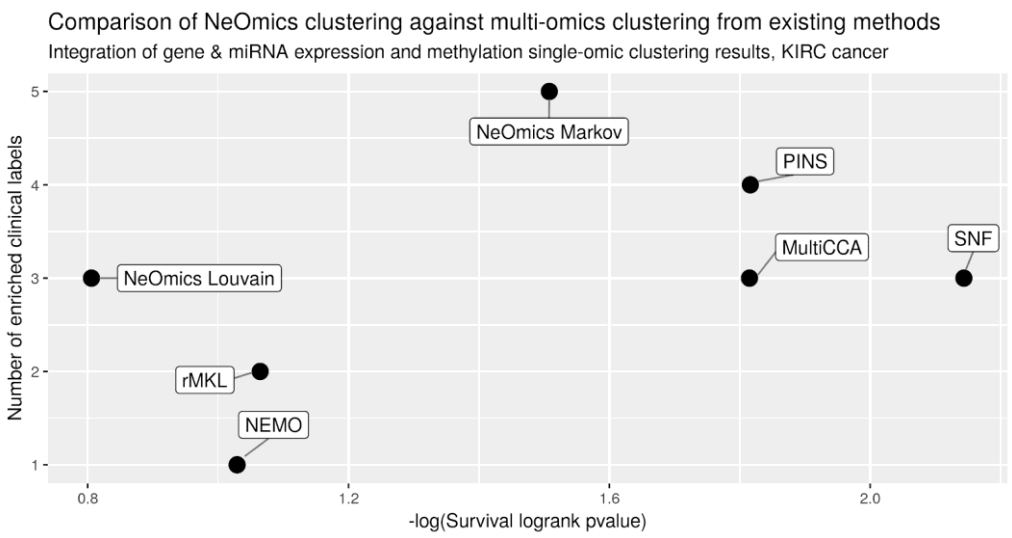

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of multi-omics clustering results, LIHC cancer
rMKL
NeOmics Louvain
NeOmics Markov
SNF
PINS
MultiCCA
NEMO
Number of enriched clinical labels
-log(Survival logrank pvalue)

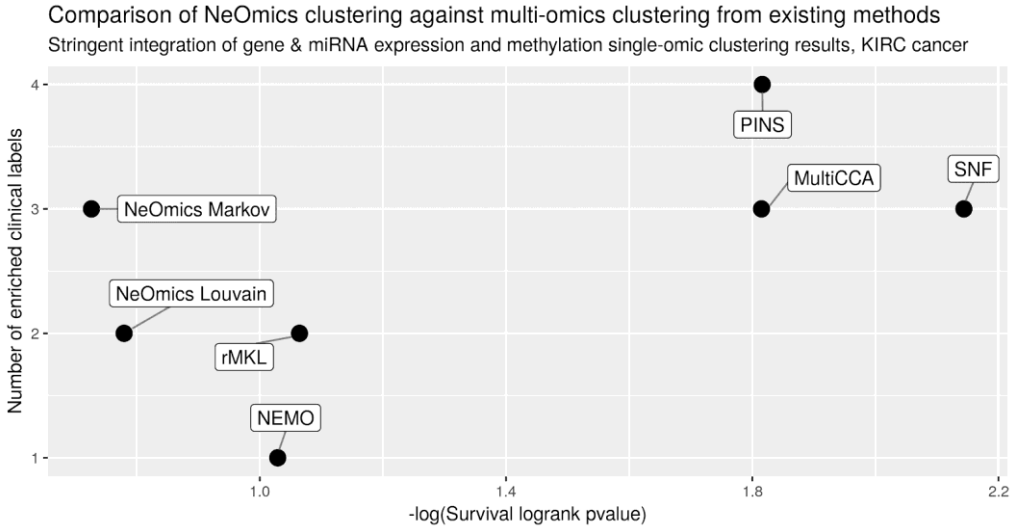

Number of disconnected partitions against minimum allowed number of support (KIRC cancer)
Number of partitions
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

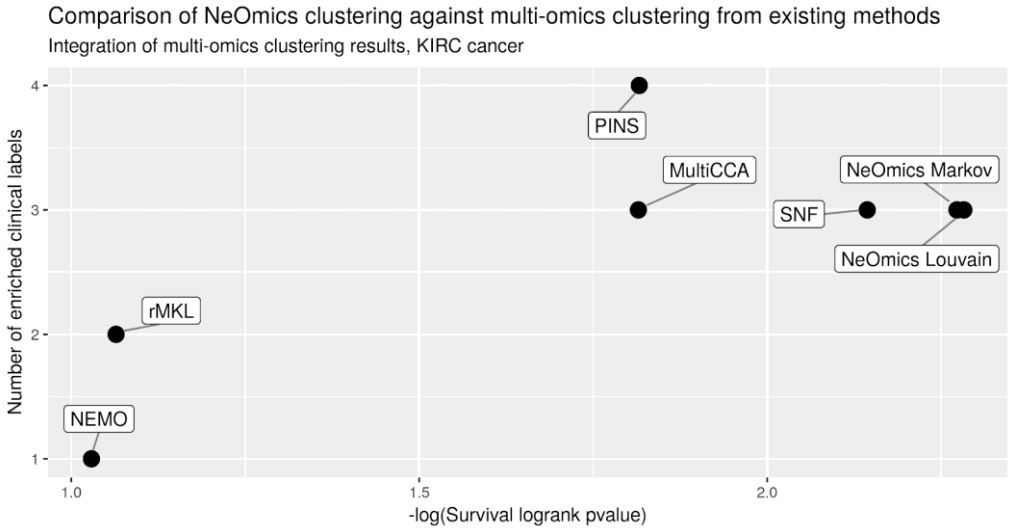

Number of selected patients against minimum allowed number of support (KIRC cancer)
Kept patients (%)
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Threshold population size

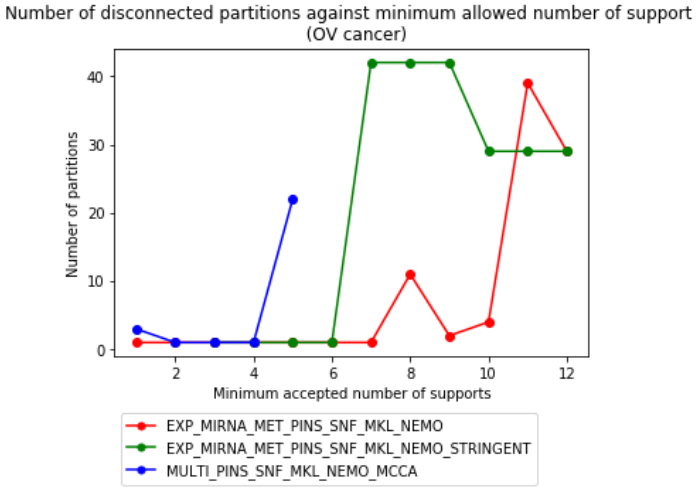

Graph connectivity against minimum allowed number of support (KIRC cancer)
Number of edges per patients
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

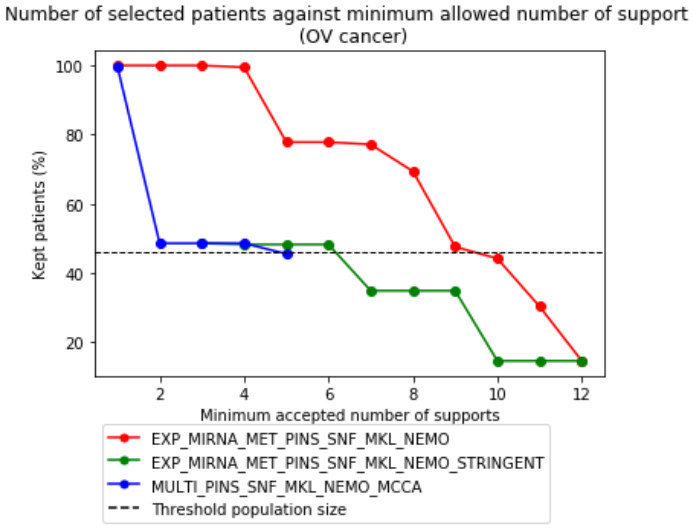

Small World tendancy of the graph before and after clustering process (KIRC cancer)
Mean Clustering Coefficient
Mean Closeness Centrality
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Louvain Clustering
Markov Clustering
Graph before Louvain Clustering (edges filtered on optimal number of supports for Louvain)
Graph before Markov Clustering (edges filtered on optimal number of supports for Markov)

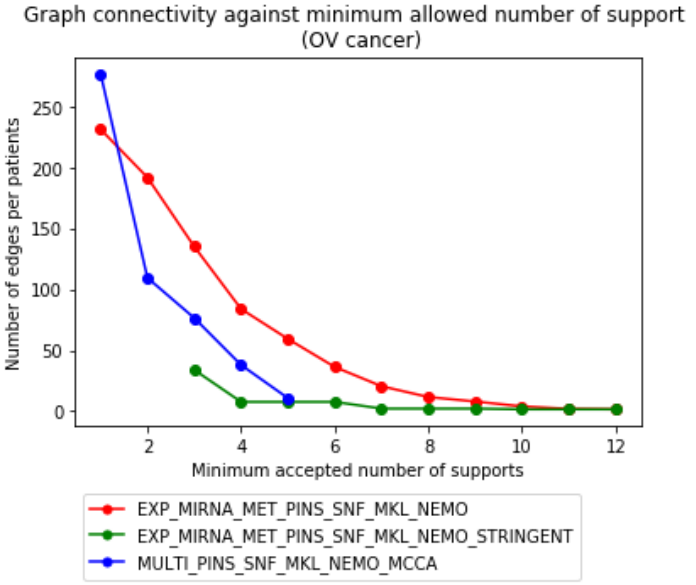

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of gene & miRNA expression and methylation single-omic clustering results, KIRC cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NeOmics Markov
PINS
MultiCCA
SNF
NeOmics Louvain
rMKL
NEMO

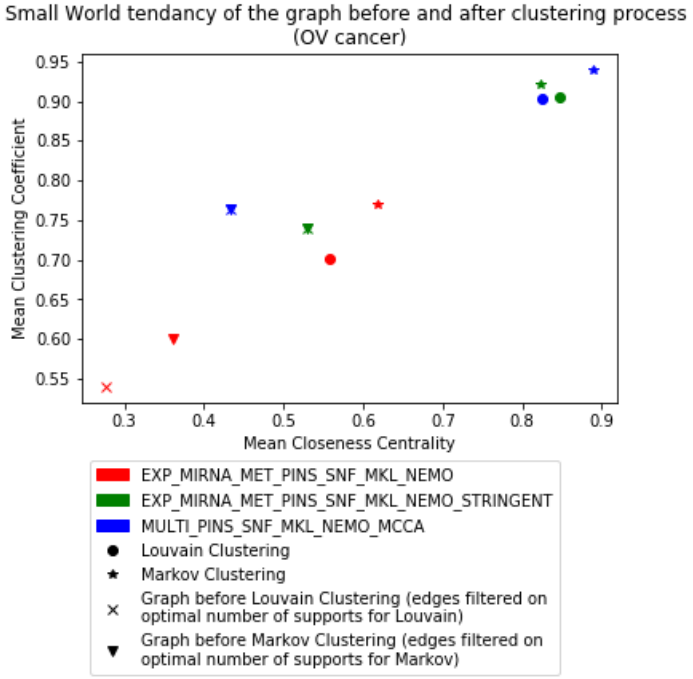

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Stringent integration of gene & miRNA expression and methylation single-omic clustering results, KIRC cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
PINS
MultiCCA
SNF
NeOmics Markov
NeOmics Louvain
rMKL
NEMO

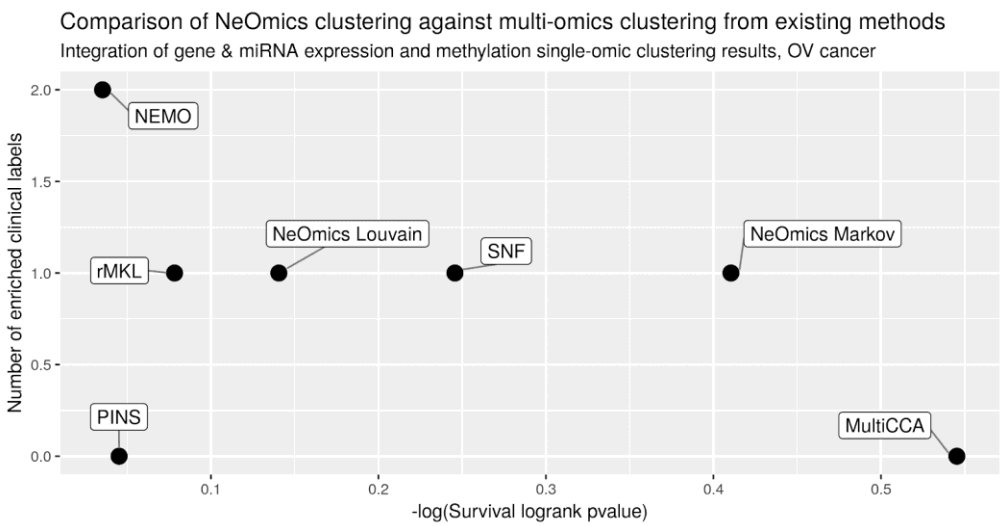

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of multi-omics clustering results, KIRC cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
PINS
MultiCCA
NeOmics Markov
SNF
NeOmics Louvain
rMKL
NEMO

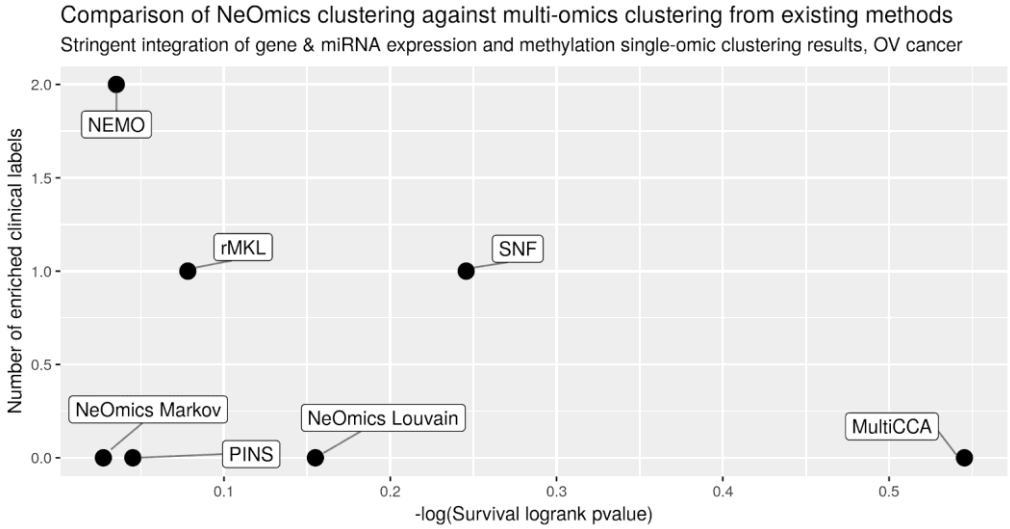

Number of disconnected partitions against minimum allowed number of support (OV cancer)
Number of partitions
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

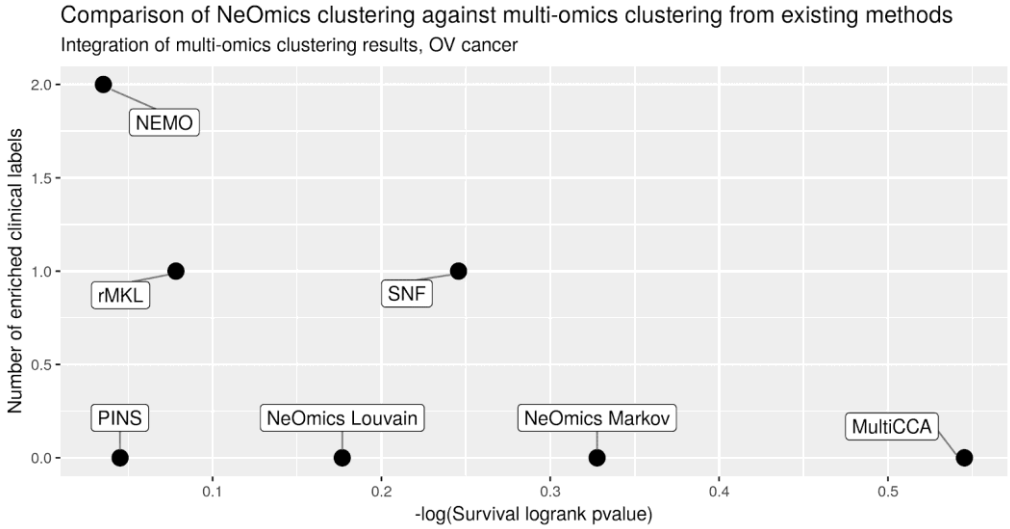

Number of selected patients against minimum allowed number of support (OV cancer)
Kept patients (%)
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Threshold population size

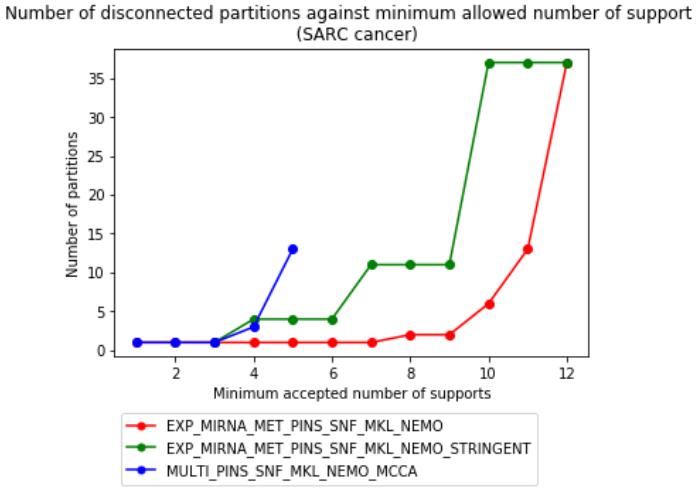

Graph connectivity against minimum allowed number of support (OV cancer)
Number of edges per patients
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

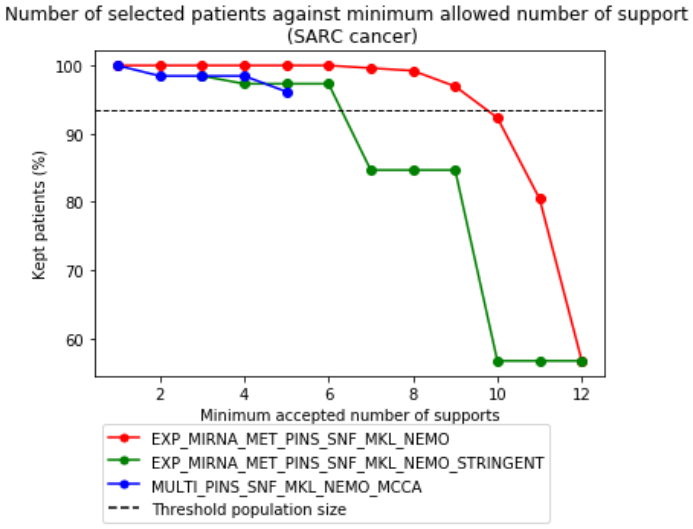

Small World tendancy of the graph before and after clustering process (OV cancer)
Mean Clustering Coefficient
Mean Closeness Centrality
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Louvain Clustering
Markov Clustering
Graph before Louvain Clustering (edges filtered on optimal number of supports for Louvain)
Graph before Markov Clustering (edges filtered on optimal number of supports for Markov)

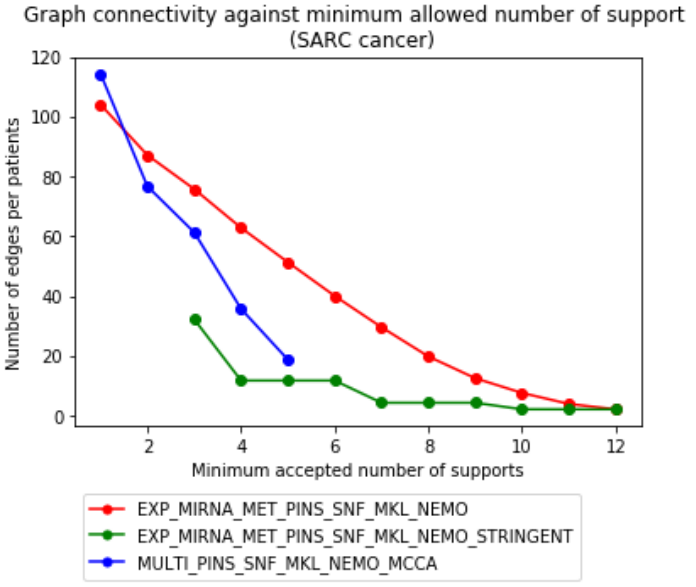

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of gene & miRNA expression and methylation single-omic clustering results, OV cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NEMO
rMKL
NeOmics Louvain
SNF
NeOmics Markov
PINS
MultiCCA

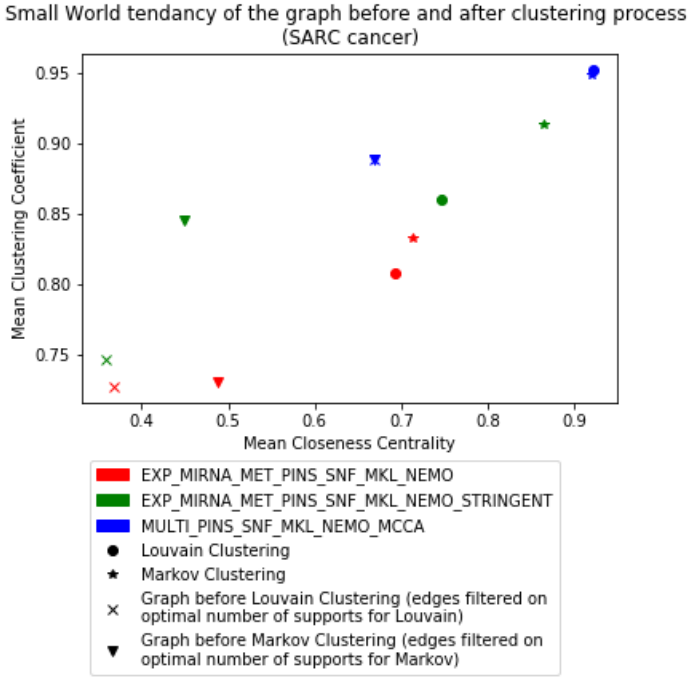

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Stringent integration of gene & miRNA expression and methylation single-omic clustering results, OV cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NEMO
rMKL
SNF
NeOmics Markov
NeOmics Louvain
PINS
MultiCCA

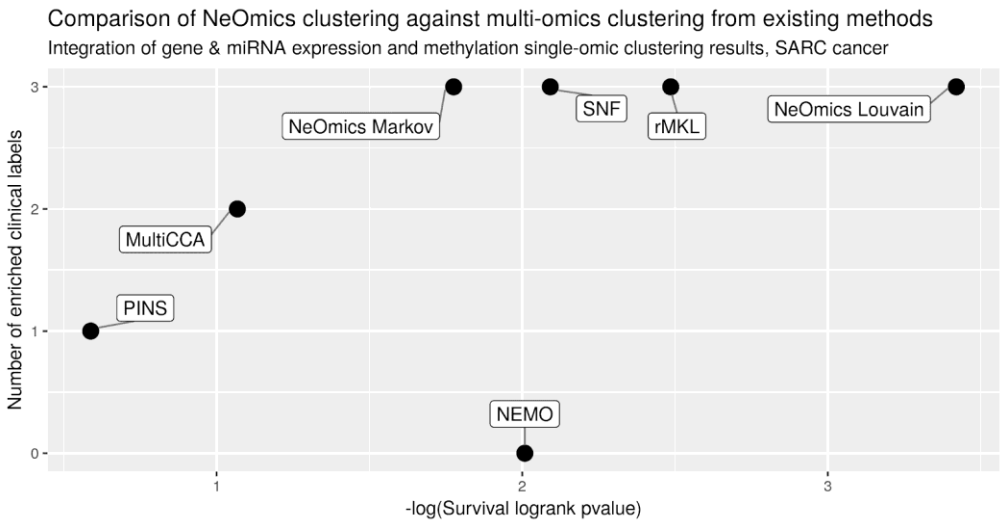

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of multi-omics clustering results, OV cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NEMO
rMKL
SNF
PINS
NeOmics Louvain
NeOmics Markov
MultiCCA

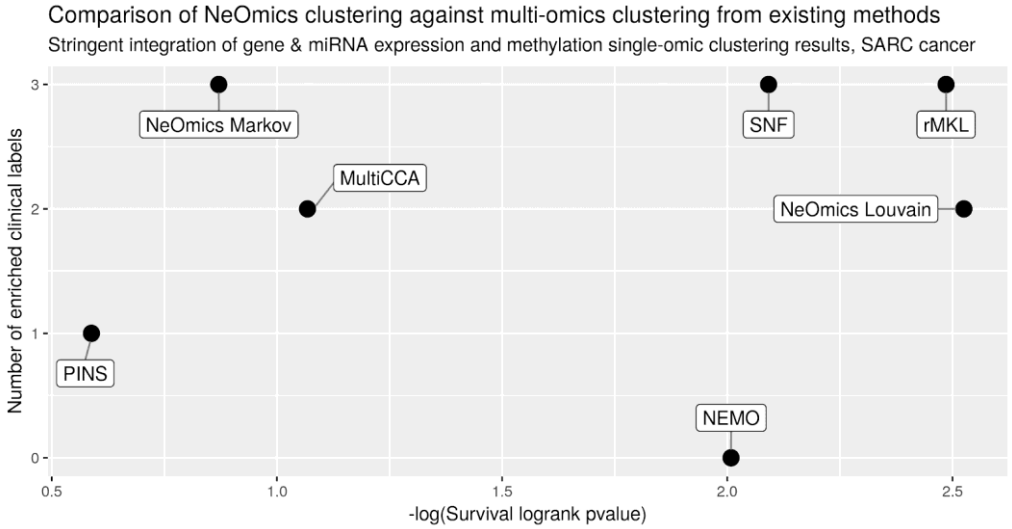

Number of disconnected partitions against minimum allowed number of support
(SARC cancer)
Number of partitions
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

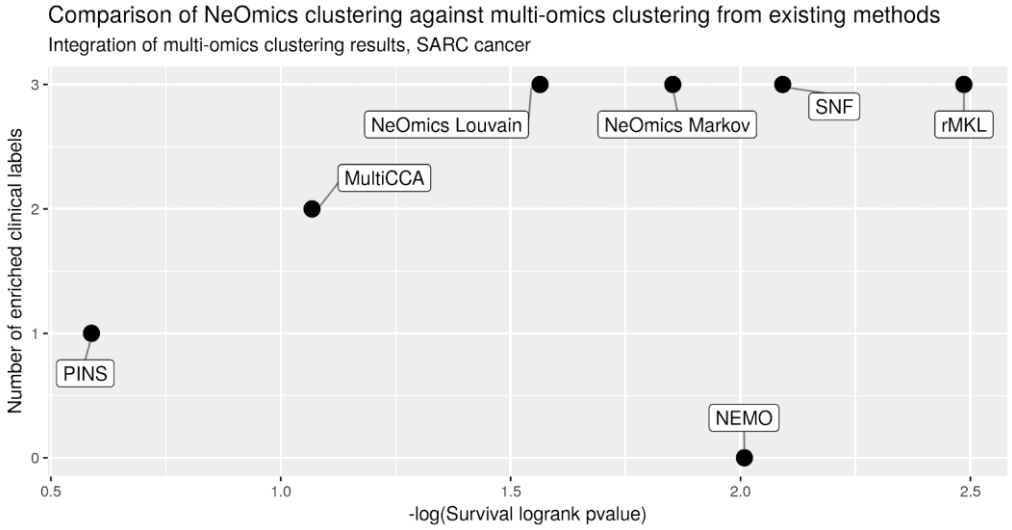

Number of selected patients against minimum allowed number of support
(SARC cancer)
Kept patients (%)
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Threshold population size

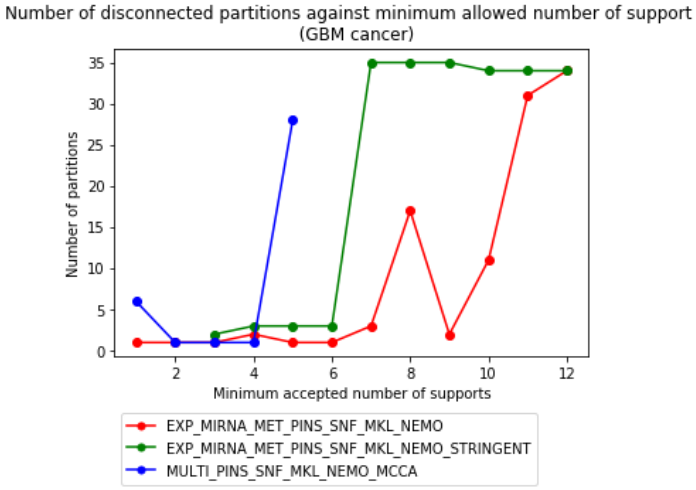

Graph connectivity against minimum allowed number of support
(SARC cancer)
Number of edges per patients
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

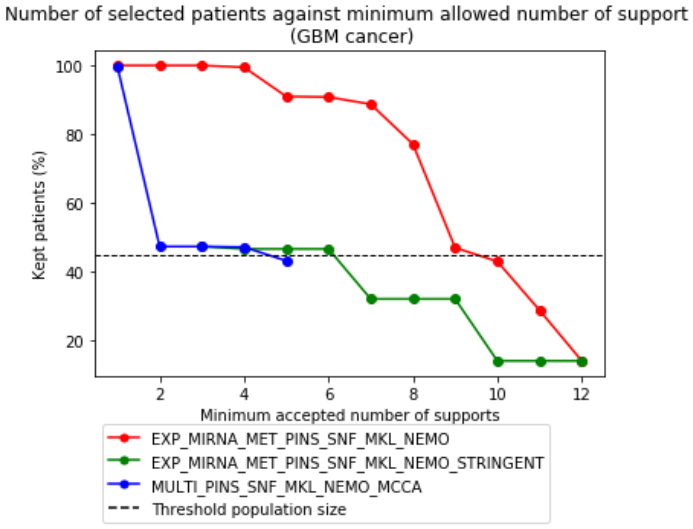

Small World tendancy of the graph before and after clustering process
(SARC cancer)
Mean Clustering Coefficient
Mean Closeness Centrality
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Louvain Clustering
Markov Clustering
Graph before Louvain Clustering (edges filtered on optimal number of supports for Louvain)
Graph before Markov Clustering (edges filtered on optimal number of supports for Markov)

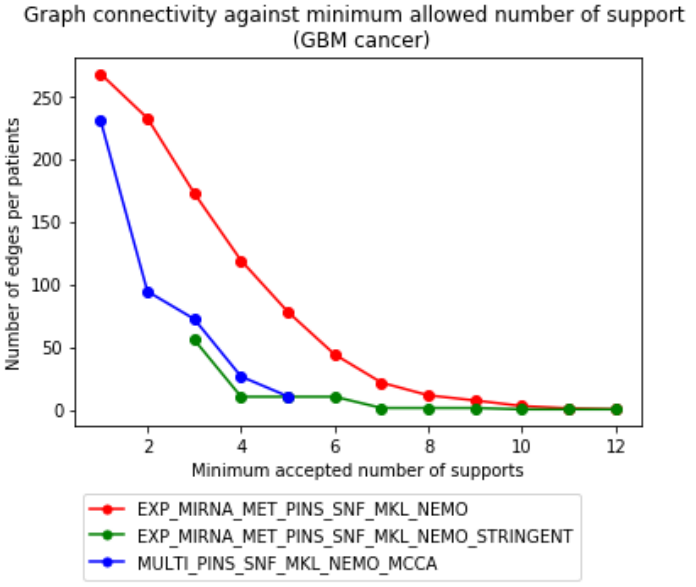

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of gene & miRNA expression and methylation single-omic clustering results, SARC cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NeOmics Markov
SNF
rMKL
NeOmics Louvain
MultiCCA
PINS
NEMO

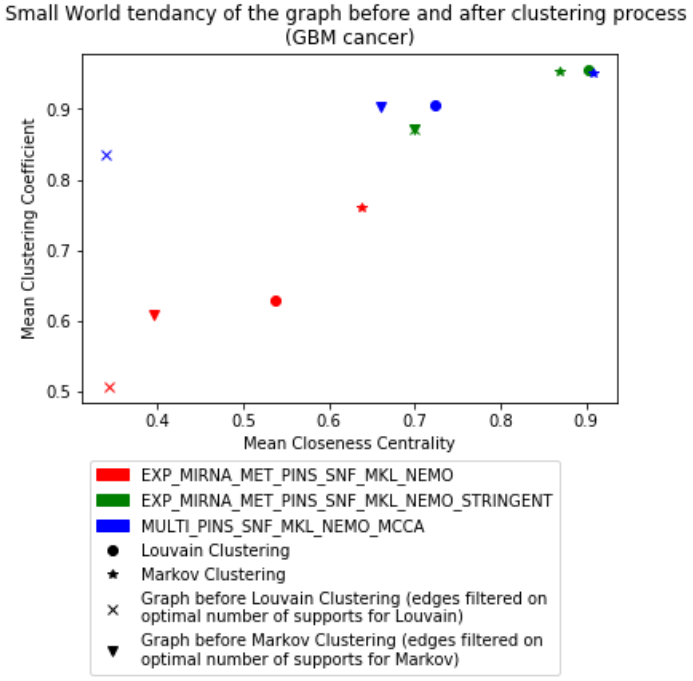

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Stringent integration of gene & miRNA expression and methylation single-omic clustering results, SARC cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NeOmics Markov
SNF
rMKL
MultiCCA
NeOmics Louvain
PINS
NEMO

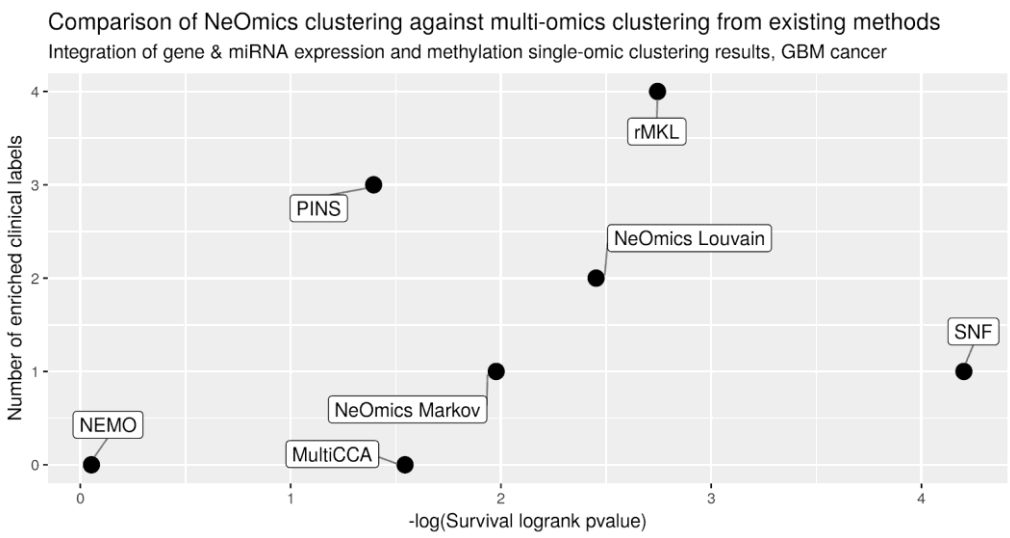

Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of multi-omics clustering results, SARC cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
NeOmics Louvain
NeOmics Markov
SNF
rMKL
MultiCCA
PINS
NEMO

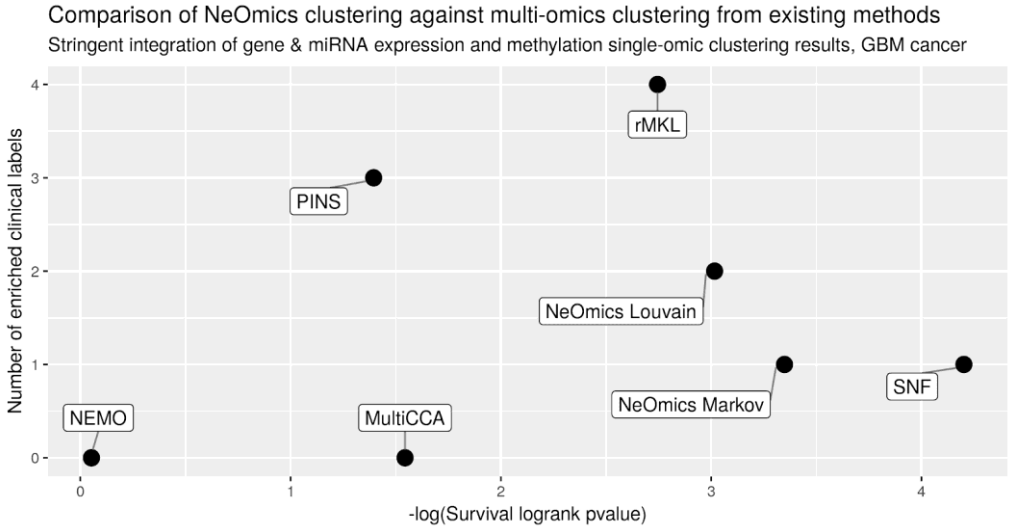

Number of disconnected partitions against minimum allowed number of support (GBM cancer)
Number of partitions
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA

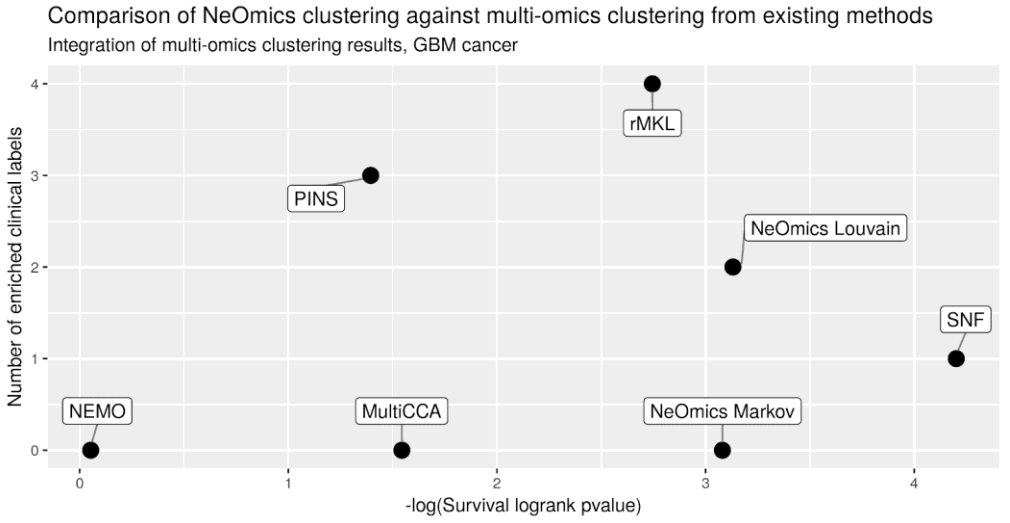

Number of selected patients against minimum allowed number of support (GBM cancer)
Kept patients (%)
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Threshold population size


Graph connectivity against minimum allowed number of support (GBM cancer)
Number of edges per patients
Minimum accepted number of supports
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA


Small World tendancy of the graph before and after clustering process (GBM cancer)
Mean Clustering Coefficient
Mean Closeness Centrality
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO
EXP_MIRNA_MET_PINS_SNF_MKL_NEMO_STRINGENT
MULTI_PINS_SNF_MKL_NEMO_MCCA
Louvain Clustering
Markov Clustering
Graph before Louvain Clustering (edges filtered on optimal number of supports for Louvain)
Graph before Markov Clustering (edges filtered on optimal number of supports for Markov)


Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of gene & miRNA expression and methylation single-omic clustering results, GBM cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
rMKL
PINS
NeOmics Louvain
SNF
NeOmics Markov
NEMO
MultiCCA


Comparison of NeOmics clustering against multi-omics clustering from existing methods
Stringent integration of gene & miRNA expression and methylation single-omic clustering results, GBM cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
rMKL
PINS
NeOmics Louvain
NeOmics Markov
SNF
NEMO
MultiCCA


Comparison of NeOmics clustering against multi-omics clustering from existing methods
Integration of multi-omics clustering results, GBM cancer
Number of enriched clinical labels
-log(Survival logrank pvalue)
rMKL
PINS
NeOmics Louvain
SNF
NEMO
MultiCCA
NeOmics Markov

Labels cliniques communs à tous les cancers : genre, âge au diagnostic, pathologie M, pathologie N, pathologie T, stade pathologique, type histologique, nouvel événement néoplasme, grade histologique du néoplasme

Auxquels s'ajoutent, selon le cancer, les labels cliniques suivants :

LUSC (poumons) : historique fumeur, nombre de paquets fumés par an
SKCM (peau) : niveau de Clark du mélanome, indicateur d'ulcération
AML (leucémie) : catégorie de risque cytogénétique, code morphologique
COAD (colon) : présence de polypes, historique polypes
GBM (cerveau) : antécédents gliomes
KIRC (rein) : résultats hémoglobine, résultats palettes (qualitatif), résultats serum calcium, compte globules blancs
LIHC (foie) : type d'inflammation du tissu hépatique, résultats albumine, résultats foetoprotéines, score Ishak, résultats créatinine
OV (ovaires) : pas de paramètre clinique supplémentaire
SARC (sarcome) : pas de paramètre clinique supplémentaire

Pour le cancer du sein, les paramètres cliniques ont été sélectionnés, mais les résultats pour ce cancer n'ont pas été générés (manque de mémoire pour PINS) :

BIC (sein) : PAM50, statut récepteurs œstrogène, statut récepteurs progestérone, niveau récepteurs œstrogènes, niveau récepteurs progestérone

Les labels cliniques utilisés par Rappoport et Shamir dans leur étude sont indiqués en bleu. Les autres labels cliniques ont été ajoutés en complément pour l'étude NeOmics.