# ÉLÉMENT DE PORTFOLIO 02

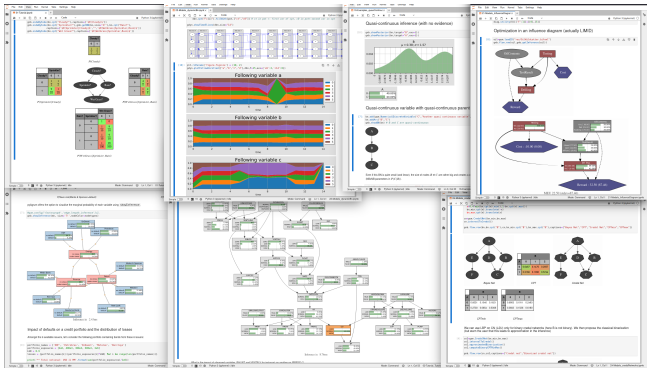
# Logiciel ou bibliothèque logicielle

**Titre de l'élément :** aGrUM /pyAgrum

**URL de l'élément :** `https://www.agrum.org`

## 1 DÉFINITION DE CET ÉLÉMENT

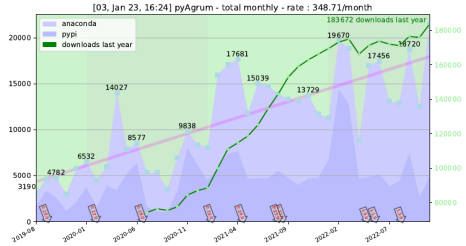
aGrUM/pyAgrum est une bibliothèque de modélisation, de calculs et d'apprentissage dans les modèles graphiques probabilistes. Les modèles graphiques permettent de manipuler des représentations probabilistes du comportement d'un système complexe en modélisant les interactions (probabilistes) entre les variables de description de ce système. Naturellement, l'approche probabiliste complexe (en grande dimension) implique un grand nombre de problèmes NP-difficiles. Toutefois, la factorisation de la loi jointe que proposent les modèles graphiques rend cette approche utilisable dans beaucoup de cas pratiques (le pire des cas restant bien sûr exponentiel en mémoire et en temps). Toutes les tâches probabilistes, de la prédiction au diagnostic en passant par la simulation et le *monitoring* sont alors accessibles. Manipuler des modèles graphiques est aussi un atout dans une démarche de validation, d'explication voire de recherche causale. aGrUM/pyAgrum a pour but de mettre à disposition l'ensemble des outils nécessaires autant pour exploiter ces capacités que pour proposer de nouveaux algorithmes.



*Une session de travail avec Jupyter et pyAgrum*

## 2 MOTIVATIONS DU CHOIX DE CET ÉLÉMENT

Développée au sein de l'équipe, cette bibliothèque *open-source* de calcul scientifique reçoit une reconnaissance aussi bien académique (plus de $60$ citations depuis $2017$, cf. ci-dessous) que publique en terme de téléchargements ($15\,000$ téléchargements mensuels en moyenne sur l'année $2022$, uniquement sur les $2$ moyens de téléchargement les plus importants de *packages* python : `pypi` et `anaconda` ; cf. ci-contre).




*Téléchargements mensuels depuis 2019*

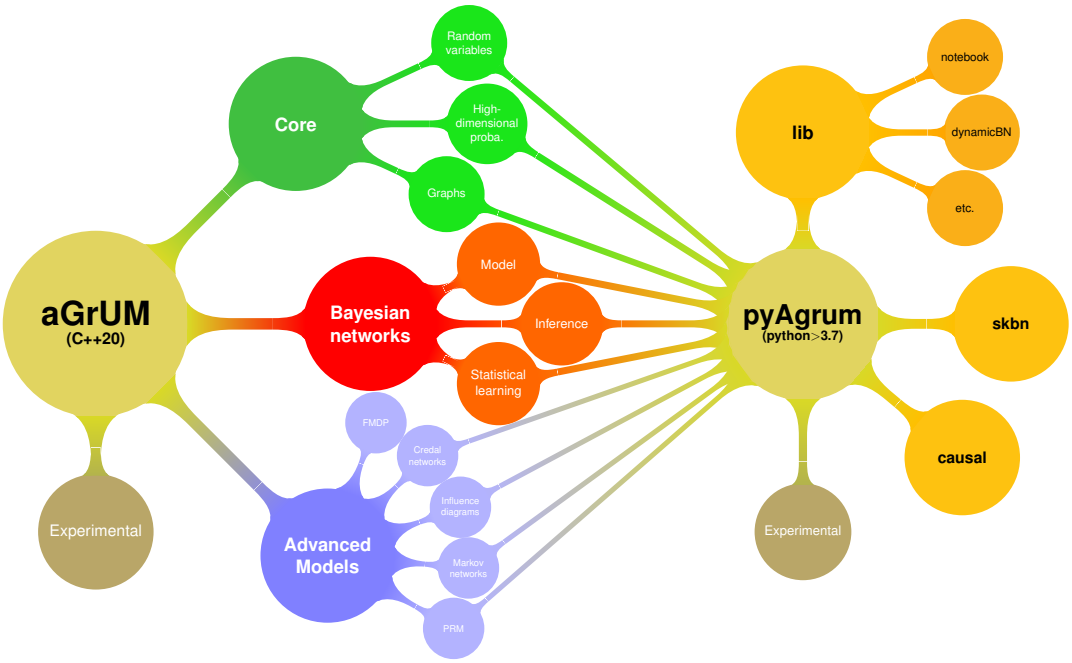
## 3 PRÉSENTATION DE CET ÉLÉMENT

La bibliothèque C++ aGrUM s'est structurée dans l'équipe autour de codes mis en commun pour des raisons pratiques de développement des algorithmes et des benchmarks utiles pour les sujets de recherche. De plus en plus complète et cohérente, s'étant dotée d'outils de contrôle de qualité (tests unitaires, outil de distribution et de gestion de versions des codes, documentation, etc.), cette bibliothèque est devenue assez mature pour sortir du laboratoire et gagner en visibilité en tant que bibliothèque open-source (LGPL). Toutefois, le choix du langage C++, même si pertinent en terme d'efficacité, handicapait le possible intérêt pour d'autres utilisateurs. Il a donc été décidé de fournir pyAgrum, un *wrapper* python permettant d'accéder aux implémentations rapides et efficaces d'aGrUM à partir d'un code python. Enfin, un effort important a été fourni, afin de rendre disponible pyAgrum sur les principaux systèmes de distribution de package, sur le site de documentation `readthedocs`, etc.

## 3.1 Périmètre

L'architecture d'aGrum est basée sur la mise à disposition de structures de données qui permettent de construire efficacement les éléments principaux des modèles graphiques probabilistes : variables aléatoires discrètes, potentiels (qui sont des tenseurs de rang quelconque, dont les dimensions sont explicitement désignées par des variables aléatoires) et graphes.

Principalement dédiée aux réseaux bayésiens, aGrUM implémente de nombreuses fonctionnalités pour l'inférence, l'apprentissage, etc. autour de ce modèle. Néanmoins, de proche en proche, les *modèles avancés* ont pris une importance croissante. Il est maintenant aussi facile de manipuler des *Markov Random Fields* (modèle graphique non orienté), des diagrammes d'influence (extension décisionnelle), des réseaux de croyances (des polytopes de réseaux bayésiens), des processus de décision markoviens factorisés (autre extension décisionnelle) et des modèles relationnels (des extension des réseaux bayésien à la logique du premier ordre).




*Composantes d'aGrUM et pyAgrum*

PyAgrum, *wrapper* Python, donne accès à la majorité de ces composantes sous la forme d'une API générée par SWIG. Des codes simplifiant et/ou 'pythonisant' (sic) les codes utilisateurs ont été produits dans `pyAgrum.lib`. Enfin, pyAgrum est désormais suffisamment mature pour permettre des développements propres comme `pyAgrum.skbn` qui donne la possibilité d'utiliser les réseaux bayésiens comme classifieurs dans l'écosystème de scikit-learn ou `pyAgrum.causal` qui implémente les modèles causaux et les calculs de probabilité par intervention de Pearl [**?**]. D'autre développements sont en cours comme une librairie de gestion des `ConditionalLinearGaussian`, des `ContinuousTime BayesianNetwork`, etc.

En permettant un accès simple non seulement aux objets de haut niveau mais également aux composants élémentaires, aGrUM/pyAgrum a pour ambition de pouvoir intéresser autant des chercheurs dans le domaine des modèles graphiques intéressés à développer de nouveaux algorithmes, que des utilisateurs des modèles graphiques intéressés par leur capacité à décrire, à prédire, à modéliser, à expliquer des systèmes complexes. aGrUM/pyAgrum sont bien entendu utilisés dans l'équipe pour tout travail de recherche sur ces domaines (d'où les bulles *Experimental* dans la figure).

## 3.2 Visibilité

Aujourd'hui, avec les statistiques d'utilisation déjà citées plus haut, il est indéniable qu'aGrUM/pyAgrum a atteint une reconnaissance académique et publique en tant que bibliothèque bien installée dans le domaine des outils opensource sur les modèles graphiques probabilistes. Sa présence sur internet se décline en un site propre, un repository gitlab (et un clone github), un groupe linkedin, un projet research-gate, une disponibilité sur pip et anaconda, une documentation sur readthedocs, des groupes d'utilisateurs discord, gitter, des projets communs avec d'autres librairies (otagrum avec openturns), etc.

# 4 RÉFÉRENCES

Une liste d'articles académiques citant aGrUM/pyAgrum est disponible à l'adresse suivante :
https ://www.lip6.fr/Pierre-Henri.Wuillemin/biblioAgrum