Aix*Marseille université
Socialement engagée

NNT/NL : xxxxAIXMxxxx/xxxED184

THÈSE DE DOCTORAT

Soutenue à Aix-Marseille Université
le 06/03/2023 par

Axel JOURNE

Apprentissage de réseaux bayésiens et réseaux causaux avec savoirs incertains d'experts

Discipline
Informatique

École doctorale
ED 184 - MATHÉMATIQUES ET INFORMATIQUE DE MARSEILLE

Laboratoire/Partenaires de recherche
Laboratoire d'Informatique et Systèmes (LIS)
ENGIE Lab CRIGEN, CSAI

Composition du jury

Philippe LERAY Nantes Université	Rapporteur
Salem BENFERHAT Université d'Artois	Rapporteur
Véronique DELCROIX Université Polytechnique Hauts-de-France	Examinatrice
Pierre-Henri WUILLEMIN Sorbonne Université	Examinateur
Djamal HABET Aix-Marseille Université	Président du jury
Christophe GONZALES Aix-Marseille Université	Directeur de thèse
Ahmed MABROUK ENGIE Lab CRIGEN	Co-directeur de thèse

ENGIE Lab CRIGEN

LIS LABORATOIRE D'INFORMATIQUE & SYSTÈMES

Affidavit

Je soussigné, Axel JOURNE, déclare par la présente que le travail présenté dans ce manuscrit est mon propre travail, réalisé sous la direction scientifique de Christophe GONZALES et Ahmed MABROUK, dans le respect des principes d'honnêteté, d'intégrité et de responsabilité inhérents à la mission de recherche. Les travaux de recherche et la rédaction de ce manuscrit ont été réalisés dans le respect à la fois de la charte nationale de déontologie des métiers de la recherche et de la charte d'Aix-Marseille Université relative à la lutte contre le plagiat.

Ce travail n'a pas été précédemment soumis en France ou à l'étranger dans une version identique ou similaire à un organisme examinateur.

Fait à Stains le 15/12/2022



Cette œuvre est mise à disposition selon les termes de la Licence Creative Commons Attribution - Pas d'Utilisation Commerciale - Pas de Modification 4.0 International.

Liste de publications et participation aux conférences

Liste des publications réalisées dans le cadre du projet de thèse :

1. Gonzales, C., Journe, A., and Mabrouk, A. (2021). Constraint-Based Bayesian Network Structure Learning using Uncertain Experts' Knowledge. Proceedings of The 34th Conference of The Florida Artificial Intelligence Research Society.
2. Gonzales, C., Journe, A., and Mabrouk, A. (2021). Apprentissage de structure de réseau bayésien sous contraintes en présence de connaissances d'experts. Actes des 15èmes Journées d'Intelligence Artificielle Fondamentale.
3. Gonzales, C., Journe, A., and Mabrouk, A. (2022). A Hybrid Algorithm for Learning Causal Networks using Uncertain Experts' Knowledge. Proceedings of The 11th Conference on Probabilistic Graphical Models.

Participation aux conférences et écoles d'été au cours de la période de thèse :

1. La 24ème édition de l'École des Jeunes Chercheurs en Informatique Mathématique (EJCIM 2020), 2020.
2. The 34th Conference of The Florida Artificial Intelligence Research Society (FLAIRS 34), 2021.
3. Les 15èmes Journées d'Intelligence Artificielle Fondamentale (JIAF 15), 2021.
4. The 11th Conference on Probabilistic Graphical Models (PGM 11), 2022.

Résumé

Dans cette thèse, nous nous intéressons à l'apprentissage de la structure de réseaux bayésiens et de modèles causaux. Ces derniers sont une classe de modèles graphiques probabilistes permettant de modéliser efficacement des systèmes complexes. Il existe principalement trois catégories d'algorithme pour l'apprentissage de la structure de réseaux bayésiens : les algorithmes sous contraintes, qui utilisent des tests d'indépendance, les algorithmes à base de score et les algorithmes hybrides. Ces techniques d'apprentissage automatique utilisent des données récupérées sur les systèmes que l'on souhaite modéliser. Cependant, pour un système donné, les experts du domaine peuvent également fournir d'importantes informations. Ces dernières viennent avec les incertitudes naturelles de l'expert qu'il faut alors intégrer à l'apprentissage pour une utilisation optimale. Le travail de cette thèse porte sur l'intégration des connaissances d'expert incertaines dans l'apprentissage de la structure des réseaux bayésiens. Selon la nature des connaissances et l'approche d'apprentissage utilisée, cette intégration peut revêtir de nombreuses formes. Nous nous intéresserons ici à deux cas. Le premier concerne des connaissances sur les indépendances entre les variables avec des algorithmes sous contraintes. Nous verrons qu'il est possible d'intégrer les connaissances incertaines directement dans les tests d'indépendances. Le deuxième cas concerne l'apprentissage de réseaux bayésiens causaux, dans lesquels le graphe représente un schéma causal complet en plus des indépendances conditionnelles. Nous verrons qu'il est possible d'intégrer des connaissances causales dans le cadre d'un apprentissage hybride. Dans les deux cas, des expérimentations permettent de comprendre le comportement des nouveaux algorithmes proposés ainsi que de montrer leur intérêt dans l'écosystème des algorithmes déjà existants.

Mots clés : système complexe, modèle graphique probabiliste, réseau bayésien, modèle causal, apprentissage de structure, connaissance

Abstract

In this thesis, we are interested in learning the structure of Bayesian and causal networks. Bayesian networks are a class of probabilistic graphical models that can be used to efficiently model complex systems. There are mainly three categories of algorithms for learning the structure of Bayesian networks: constraint-based algorithms, which use independence tests, score-based algorithms, and hybrid algorithms. These machine learning techniques use data generated by the systems we wish to model. Moreover, in a given system, the experts of the domain can also provide important information. However, this information comes with the natural uncertainties of the expert that must be integrated into the learning process for an optimal use. The work in this thesis focuses on the integration of uncertain expert knowledge into Bayesian network structure learning. Depending on the nature of the knowledge and the learning approach used, this integration can take many forms. We will focus here on two cases. The first one concerns knowledge about the independence between random variables of the probability distribution using constraint-based algorithms. We prove that it is possible to intelligently integrate the knowledge and its uncertainties directly into the independence tests. The second case concerns the learning of causal Bayesian networks, in which the graph represents a complete causal scheme in addition to conditional independences. We will show that it is possible to integrate causal knowledge into hybrid learning while allowing to optimize the elicitation of the expert knowledge. In both cases, experiments are conducted including comparisons with other algorithms in the literature to understand the behavior of the proposed new algorithms as well as to show their utility in the ecosystem of already existing algorithms.

Keywords: complex system, probabilistic graphical model, Bayesian network, causal model, structure learning, knowledge

Remerciements

Selon moi, une thèse est avant tout un long effort d'apprentissage, de remise en question et de perfectionnement, après lequel on apprécie encore plus ce que l'on apprend après. C'est un voyage difficile, mais j'ai heureusement eu la chance d'être encadré et entouré par de nombreuses personnes à qui sont destinés ces présents remerciements.

Tout d'abord, à mes directeurs de thèse, Christophe GONZALES et Ahmed MABROUK, pour leur bienveillance, leur disponibilité, leur immense patience et toutes les choses qu'ils ont bien voulu partager à travers leur encadrement. Vous m'avez accueilli dès mon stage à la fin de mon cursus d'ingénieur, et m'avez permis de concrétiser ce beau projet. Pour cela je vous prie de croire à ma plus sincère gratitude.

Je tiens à remercier Pierre-Henri WUILLEMIN et Djamel HABET pour leur aide précieuse et leurs conseils pertinents en tant que membres de comité de suivi de thèse. Merci également à Philippe LERAY pour m'avoir fait découvrir les réseaux bayésiens dans le cadre de mon cursus d'ingénieur à Polytech'Nantes, puis, bien plus tard, pour avoir accepté d'être rapporteur de cette thèse aux côtés de Salem BENFERHAT que je remercie bien évidemment. Enfin, je dis par ailleurs d'avance merci à Véronique DELCROIX, ainsi que tous les membres du jury déjà cités dans ce paragraphe, devant lequel je soutiendrai cette thèse seulement quelques mois après avoir écrit ces lignes.

Cette thèse n'aurait pas non plus pu voir le jour sans l'appui d'Engie, qui l'a financé et m'a permis de profiter de ses locaux pour travailler dans les meilleures conditions (même si une certaine épidémie en a décidé autrement pendant un temps). Et puisque qu'une entreprise n'est rien sans ses employés, merci à tous les collègues que j'ai pu côtoyer au cours de ces 3 années. J'y ai rencontré des personnes sympathiques et intéressantes, avec qui j'ai pris un grand plaisir à échanger. J'ai une pensée particulière aux personnes du laboratoire CSAI au CRIGEN, et encore plus à Yolène BERROU et Siamak SOLAT, doctorants en même temps que moi, avec qui les interactions ont été riches d'échanges d'expériences et de sympathie. Je pense également à la communauté des doctorants du CRIGEN, qui m'a donné l'occasion de décompresser, mais également d'entrevoir l'élaboration d'une thèse dans des domaines scientifiques très variés.

Aix-Marseille Université est l'autre entité qui m'a permis de concrétiser ce beau projet. Je remercie l'école doctorale d'informatique et de mathématiques pour leur efficacité et leur réactivité à mes nombreuses questions, ainsi que le Laboratoire d'Informatique et Systèmes de l'université, qui a accepté d'héberger ma thèse.

J'ai eu la chance d'avoir pu compter sur le soutien indéfectible de ma famille. Merci à ma mère et à mon père, pour avoir longuement écouté mes complaintes, mes inquiétudes, mes déceptions, mais aussi mes joies et mes espoirs. Je remercie mon

grand-père, pour son grand intérêt pour mon travail, et je remercie également ma grand-mère, qui m'a toujours soutenu tout au long de mes études. Merci à mon frère et ma sœur pour leurs encouragements si fraternels, et merci à mon beau-père pour sa disponibilité en cas de problème (et ses gâteaux).

Enfin, durant les heures les plus dures, j'ai toujours pu compter sur mes amis, à Nantes, en Ile-de-France, à Épernay et même en Allemagne, pour décompresser et me changer les idées. Merci à vous tous du fond du cœur.

Merci à tous ceux que je n'ai pas pu citer dans ces quelques lignes et qui m'ont aidé d'une manière ou d'une autre à l'élaboration de ma thèse.

Et puisque qu'il n'y a pas de bonne ou de mauvaise situation, je dis merci la vie.

Table des matières

Affidavit 2

Liste de publications et participation aux conférences 3

Résumé 4

Abstract 5

Remerciements 6

Table des matières 8

Table des figures 11

Liste des tableaux 13

Introduction 15

1 État de l'art 20
- 1.1 Introduction . . . 20
- 1.2 Réseaux bayésiens . . . 21
  - 1.2.1 Définitions . . . 21
  - 1.2.2 Indépendance . . . 26
  - 1.2.3 Différents types de connexions . . . 27
  - 1.2.4 D-séparation . . . 29
  - 1.2.5 Couverture de Markov . . . 31
  - 1.2.6 I-map, D-map et P-map . . . 32
  - 1.2.7 Classe d'équivalence de Markov . . . 34
- 1.3 Causalité et réseaux bayésiens causaux . . . 35
  - 1.3.1 Représentation Graphique de la causalité . . . 36
  - 1.3.2 Intervention . . . 38
  - 1.3.3 Identifiabilité . . . 40
- 1.4 Conclusion . . . 43

2 Apprentissage des réseaux bayésiens 44
- 2.1 Introduction . . . 44
- 2.2 Apprentissage des paramètres . . . 45
  - 2.2.1 Définitions et Hypothèses . . . 45
  - 2.2.2 Approche statistique . . . 47

2.2.3 Approche bayésienne . . . 49
2.3 Apprentissage de la structure . . . 50
2.3.1 Hypothèses . . . 51
2.3.2 Tests d'indépendance . . . 52
2.3.3 Algorithmes . . . 54
2.3.4 Apprentissage avec des connaissances . . . 76
2.3.5 L'apprentissage de réseaux bayésiens causaux . . . 87
2.4 Conclusion . . . 91

3 Algorithme d'apprentissage sous contraintes de structure de réseau bayésien à l'aide de connaissances incertaines 93
3.1 Introduction . . . 93
3.2 Hypothèses . . . 94
3.2.1 Hypothèses d'apprentissage . . . 94
3.2.2 Les connaissances . . . 95
3.3 Test d'indépendances . . . 95
3.3.1 Modèle pour le test marginal . . . 96
3.3.2 Développement du test d'indépendance marginale . . . 99
3.3.3 Modèle pour le test conditionnel . . . 105
3.3.4 Développement du test d'indépendance conditionnelle . . . 107
3.4 Algorithme . . . 109
3.4.1 Problèmes rencontrés et résolus . . . 109
3.4.2 Pseudo-code . . . 110
3.5 Expérimentations . . . 112
3.5.1 Paramètres expérimentaux . . . 112
3.5.2 Résultats . . . 114
3.6 Conclusion . . . 120

4 Algorithme hybride d'apprentissage de structure de réseaux bayésiens causaux à l'aide de connaissances incertaines 121
4.1 Introduction . . . 121
4.2 Hypothèses . . . 123
4.2.1 Hypothèses d'apprentissage . . . 123
4.2.2 Les connaissances . . . 123
4.3 Algorithme . . . 125
4.3.1 Structure générale . . . 125
4.3.2 Étape sous contraintes . . . 126
4.3.3 Étape à base de score . . . 131
4.3.4 Étape interactive . . . 135
4.4 Expérimentations . . . 139
4.4.1 Paramètres expérimentaux . . . 139
4.4.2 Résultats . . . 141
4.5 Conclusion . . . 149

Conclusion 150

Bibliographie 155

Index 163

Table des figures

1 Schéma d'une éolienne. 16

1.1 Représentation graphique d'un réseau social. 22
1.2 Exemples de graphes. Les graphes 1.2a et 1.2b ne sont pas des DAG alors que le graphe 1.2c en est un. 22
1.3 Exemple de réseau bayésien avec ses CPD. 25
1.4 Types de connexion entre trois nœuds d'un réseau bayésien 28
1.5 Réseau bayésien Asia 31
1.6 Exemple de couverture de Markov. Le nœud concerné est X, en jaune, la couverture de Markov est en orange et les autres points sont en rouge. 32
1.7 Exemples de tentatives de P-map infructueuses. 33
1.8 (a) Le DAG du réseau bayésien Asia. (b) Le CPDAG du réseau bayésien Asia. 35
1.9 Exemple de modèles. (a) Modèle markovien (b) Modèle semi-markovien (c) modèle non-markovien 37
1.10 Représentation graphique d'une intervention sur H. (a) Le réseau causal d'origine. (b) Le réseau causal sur lequel l'intervention sur H a eu lieu. 39
1.11 Exemple d'effet causal non identifiable 40
1.12 Exemple de DAG sur lequel on peut appliquer le critère backdoor. . . . 41
1.13 Exemple de DAG sur lequel on peut appliquer le critère frontdoor . . . 42

2.1 Trois DAG différents compatibles avec le même ordre $X \prec Y \prec Z$. . . . 68
2.2 Transformer le graphe complet en PDAG ne permet pas d'aboutir à un graphe consistent avec les connaissances imposées. 81
2.3 Modélisation de l'environnement d'apprentissage incluant la structure, les données et les connaissances des experts. 84
2.4 Modélisation de l'environnement d'apprentissage incluant la structure et les connaissances des experts. 84
2.5 Modélisation détaillée de l'environnement d'apprentissage incluant la structure et les connaissances des experts. 85
2.6 Arbre de décision pour calculer $P(O_j^i | g_j, \gamma^i)$ pour $O_j^i \neq \emptyset$. 86

3.1 Le réseau bayésien soulignant la relation entre $(XY)^{(n)}$ et la variable $\mathbb{I}$. . 97
3.2 Transformation de probabilité $\varphi_k(\cdot)$. 99
3.3 Le réseau bayésien soulignant la relation entre les échantillons et les connaissances des experts. 99
3.4 Le réseau bayésien pour le test d'indépendance conditionnelle. 105

3.5 (a) Le DAG du réseau bayésien Asia. (b) Le squelette obtenu avec l'algorithme PCSDC. (c) Les connaissances utilisées par l'algorithme PCSDC. 116

4.1 Représentation schématique de la transmission d'une éolienne. 124
4.2 Illustration de l'incertitude de l'orientation causale entre deux variables. 124
4.3 Arbre de décision utilisé pour l'étape à base de score de l'algorithme. . 132
4.4 Exemple d'exécution de l'étape à base de score de notre algorithme. En rouge les arcs et arêtes modifiés pendant cette étape. 134
4.5 (a) Le DAG du réseau bayésien Asia. (b) Un PDAG et les poids de chaque arc et arête entraînant une modification de SHD. 140

Liste des tableaux

2.1 Récapitulatif des algorithmes à base de score cités dans la sous-section 2.3.3.2. (1. RG = Recherche Gloutonne) 71
2.2 Quelques indépendances utilisées dans l'algorithme GSA. 85

3.1 Liste des réseaux bayésiens utilisés dans les expérimentations. 112
3.2 F-scores de PCS, PCSDC et PCSe pour le réseau bayésien Alarm en fonction du pourcentage de véracité et de la taille de la base de données. 114
3.3 F-scores de PCS, PCSDC et PCSe pour différents réseaux bayésiens de référence avec des bases de données de 500 enregistrements en fonction du pourcentage de véracité. 115
3.4 F-scores de PCS, PCSDC et PCSe pour le réseau bayésien Alarm en fonction du paramètre γ_k et de la taille de la base de données. 117
3.5 F-scores de PCS, PCSDC et PCSe pour différents réseaux bayésiens de référence avec des bases de données de 500 enregistrements en fonction du paramètre γ_k. 118
3.6 F-scores de PCSe pour le réseau bayésien Alarm en fonction de γ et ρ. 119
3.7 F-scores de PCSe pour le réseau bayésien Alarm en fonction de γ_k et ρ_k. 120

4.1 Modifications des orientations temporaires. 130
4.2 Mesures utilisées pour déterminer la précision d'un graphe partiel. . . 141
4.3 SHD de GSA, CAIRN.2, CAIRN.3 et PCSDC pour le réseau bayésien Alarm en fonction de p_{XY} et de la taille de la base de données. 142
4.4 SHD relatif de GSA, CAIRN.2, CAIRN.3 et PCSDC en fonction de p_{XY} et du réseau bayésien. 143
4.5 F-score du squelette de GSA, CAIRN et PCSDC pour le réseau bayésien Alarm en fonction de p_{XY} et de la taille de la base de données. 144
4.6 SHD de GSA, CAIRN.2, CAIRN.3 et PCSDC pour le réseau bayésien Alarm en fonction de la véracité et de la taille de la base de données. 145
4.7 SHD relatif de GSA, CAIRN.2, CAIRN.3 et PCSDC en fonction de la véracité et du réseau bayésien. 145
4.8 SHD de GSA, CAIRN.2, CAIRN.3 et PCSDC pour le réseau bayésien Alarm en fonction de la taille de la base de connaissance $|K|$ et de la taille de la base de données. 146
4.9 SHD relatif de GSA, CAIRN.2, CAIRN.3 et PCSDC en fonction de la taille de la base de connaissance $|K|$ et du réseau bayésien. 146
4.10 Pourcentage d'orientation de chaque étape de CAIRN pour le réseau bayésien Alarm en fonction de p_{XY} et de la taille de la base de données. 147

4.11 Pourcentage d'orientation de chaque étape de CAIRN en fonction de p_{XY} et du réseau bayésien. 148
4.12 Précision des graphes de GSA, CAIRN et PCSDC pour le réseau bayésien Alarm en fonction de la taille de la base de données. 148
4.13 Précision des graphes de GSA, CAIRN et PCSDC en fonction du réseau bayésien. 148

Introduction

L'énergie éolienne est l'énergie électrique produite en captant et en transformant l'énergie du vent. C'est une énergie renouvelable, elle occupe donc une place importante dans le plan de transition énergétique français, ainsi que dans la lutte contre le dérèglement climatique. Chaque année, la part de l'éolien dans le mixe énergétique français augmente, et a atteint 7,9% de la production électrique totale en 2020 [1], ce qui représente 39,7 TWh. La puissance installée représente quant à elle 17,6 GW en 2020, et l'objectif pour 2023 est fixé entre 21,8 et 26 GW [2]. Cela se traduit par des éoliennes de plus en plus nombreuses sur le territoire français : on en comptait 2006 en 2020 [3].

Il existe plusieurs types d'éoliennes en fonction des usages. Pour la production électrique, elles peuvent adopter de nombreuses tailles et formes selon l'emplacement qu'elles occupent. Les plus courantes sont les éoliennes verticales. Elles sont constituées d'un mât, surmonté d'une nacelle. L'énergie cinétique du vent est captée par des pales qui tournent autour d'un axe vertical parallèle au sens du vent. Les éoliennes verticales terrestres ont une puissance située entre 1,8 et 3 MW, avec un mât pouvant atteindre 100m et un rotor pouvant aller jusqu'à 110 m de diamètre. La majeure partie du mécanisme est logée à l'intérieur de la nacelle. La rotation des pales entraine un axe puis une série de mécanismes dans un multiplicateur. Ce dernier est composé d'une multitude d'engrenages qui permettent de transformer la vitesse de rotation lente et à haut couple que le vent induit sur le rotor en une rotation rapide et à faible couple. Cette dernière est ensuite utilisée par une génératrice pour produire de l'électricité. Ce mécanisme d'apparence simple cache en réalité une grande complexité. On peut en effet trouver dans la nacelle de nombreuses autres pièces, telles que les systèmes de contrôle de l'orientation des pales, de la nacelle, ou les systèmes de freinage et de refroidissement.

1. RTE - Bilan électrique 2020
2. Programmations Pluriannuelles de l'Énergie - 2019-2023
3. SDES - Quatrième trimestre 2020

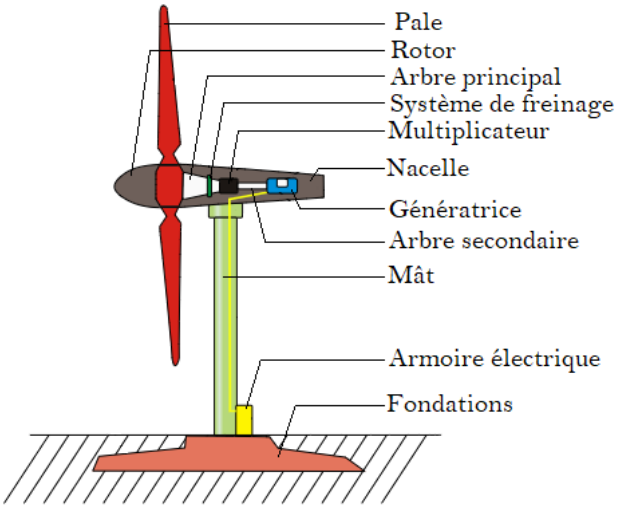


FIGURE 1 – Schéma d'une éolienne.

Engie est une entreprise française du secteur de l'énergie. À ce titre, elle possède une partie du parc éolien français, d'une puissance de 2056 MW. La transition énergétique pousse à une optimisation des moyens de production énergétique, et donc une optimisation des parcs éoliens. Mais la gestion d'autant de dispositifs complexes est un problème difficile, car elle demande une supervision constante ainsi que de fréquentes prises de décision. En effet, la défaillance d'un composant peut engendrer un temps d'arrêt important et des coûts de maintenance élevés. Pour assister les opérateur dans leur travail, il est possible de représenter ces systèmes avec des modèles. Ceux-ci doivent cependant respecter quelques contraintes.

Tout d'abord, ces modèles doivent permettre d'accomplir de la maintenance prédictive et du diagnostic d'anomalie. En effet, traiter un problème avant qu'il ne survienne est le meilleur moyen d'assurer l'optimisation de l'utilisation du système. Mais puisque tous les problèmes ne sont pas évitables, il est également nécessaire de faire du diagnostic d'anomalie, qui consiste à identifier les acteurs clefs responsables d'une défaillance du système. Ensuite, les maintenances ou réparations doivent être faites par des personnes. Il y a donc une nécessité d'interaction entre le modèle et ces personnes, ce qui implique que le modèle doit être aisément interprétable. Lorsque celui-ci donne une réponse, les opérateurs doivent pouvoir comprendre pourquoi cette réponse a été donnée. Enfin, ces opérateurs ont eux-mêmes des connaissances sur les systèmes, et ces connaissances doivent pouvoir être intégrées au modèle et participer à l'élaboration des réponses. En effet, ces connaissances peuvent être une source d'informations importante et utile, et permettre d'améliorer significativement le modèle et ses réponses.

Parmi les modèles qu'il est possible d'utiliser avec toutes ces contraintes, les réseaux bayésiens sont des outils ayant déjà montré de bons résultats et ont été utilisés pour de nombreuses applications. Les réseaux bayésiens sont des modèles graphiques probabilistes développés depuis plus d'une trentaine d'années. Ils sont constitués

d'un graphe couplé à une distribution de probabilité et leurs utilisations reposent donc principalement sur le calcul des probabilités. Grâce à cette spécificité, les informations utilisées peuvent être fortement incertaines, puisque cette incertitude est fondamentalement prise en compte par le modèle sous la forme de probabilités. En retour, les réponses fournies par le modèle sont également nuancées, ce qui permet à l'opérateur qui interagit avec le modèle de considérer un large spectre de possibilités de réponse et prendre la meilleure décision compte tenu des probabilités de chaque réponse.

Un réseau bayésien est composé de deux éléments : le graphe, désigné en tant que **structure**, et la distribution de probabilité, désignée en tant que **paramètres**. Ces deux éléments sont intrinsèquement liés. En effet, la structure est composée de nœuds, chacun représentant une variable aléatoire de la distribution, et les indépendances conditionnelles de cette distribution sont encodées dans la structure, au gré de ses arcs. Pour utiliser de tels modèles, il est donc nécessaire d'avoir chacun de ses deux éléments constitutifs. Il arrive que ceux-ci puissent être obtenus en partie manuellement. Une entreprise comme Engie possède une expertise dans les systèmes qu'elle gère, et peut donc utiliser des connaissances d'experts pour construire le modèle. Mais du fait de la complexité des systèmes tels que les éoliennes, la tâche s'avère fastidieuse et permet rarement d'obtenir un modèle complet. Une autre solution est alors d'obtenir le modèle à partir de bases de données, ce qui nécessite alors des techniques d'apprentissage automatique spécifiques aux réseaux bayésiens.

L'apprentissage automatique des réseaux bayésiens est un problème difficile. Tout d'abord, nous devons distinguer l'apprentissage de la structure de l'apprentissage des paramètres. Les approches sous contraintes consistent à utiliser des tests statistiques pour identifier les indépendances conditionnelles entre les variables de la distribution, pour ensuite construire la structure en fonction des résultats des tests. Les approches à base de score mesurent la qualité des structures grâce à un score, et cherchent la structure ayant la plus grande qualité vis-à-vis des données à disposition. Enfin, il existe des approches hybrides qui combinent les deux précédentes pour tirer profit de leurs avantages respectifs.

Cependant, en n'utilisant que les données, on ne met pas en valeur l'expertise de l'entreprise, ce qui est donc une perte. Une autre façon d'apprendre un réseau bayésien consiste donc à construire le modèle grâce à un apprentissage par les données assisté avec des connaissances d'experts. Cela permet de valoriser non seulement les données, mais aussi l'expertise de l'entreprise, et ainsi obtenir de meilleurs modèles. On appelle **élicitation** le fait de recueillir les connaissances de l'expert sur un système pour les exploiter dans la construction d'un modèle. Ces connaissances peuvent prendre de très nombreuses formes et porter sur divers aspects des modèles. De plus, ces connaissances peuvent être certaines ou incertaines selon la prise en compte de l'incertitude de l'expert lors de l'élicitation. Compte tenu de la diversité existante des connaissances, il n'existe pas une unique façon de les intégrer dans l'apprentissage. Au contraire, chaque forme de connaissances demande souvent des méthodes spécifiques. Dans le cadre des travaux d'Engie, il est donc intéressant de développer des algorithmes spécifiques pour utiliser les connaissances disponibles.

Un réseau bayésien représente grâce à sa structure les indépendances conditionnelles entre les variables de sa distribution. Les indépendances conditionnelles relèvent du lien statistique. Il est cependant possible d'aller plus loin en considérant les relations causales entre les variables. Ce sont des liens plus profonds, qui relèvent du mécanisme régissant les relations entre les variables. Les notions de causes et d'effets sont des notions naturelles pour l'esprit humain. Lorsqu'on pense à deux éléments dépendants, une relation causale est très souvent à l'origine de cette dépendance. Ainsi, la structure d'un réseau bayésien peut, en plus de représenter les indépendances conditionnelles, aussi représenter les relations causales entre les variables. On parle alors de réseau bayésien causal.

La prise en compte des relations causales et l'utilisation des réseaux bayésiens causaux en général permet de répondre à plus de questions que l'utilisation des indépendances conditionnelles seules. Les modèles causaux permettent notamment de visualiser des **interventions**, et donc l'impact sur d'autres variables d'un changement des valeurs d'une ou plusieurs variables. Et ces modèles permettent également de formaliser le raisonnement **contrefactuel**, ce qui permet d'imaginer et comprendre les répercussions de règles imaginaires sur des mondes alternatifs. Cependant, ces nouvelles possibilités se font au prix d'un apprentissage plus difficile. Toutes les relations causales ne sont pas toujours identifiables avec les données uniquement. Dans ce cadre, les connaissances d'experts revêtent une importance décuplée, puisqu'elles permettent non seulement d'améliorer l'apprentissage, mais également de le compléter.

L'objectif des travaux présentés dans cette thèse est donc de développer des algorithmes d'apprentissage de structure de réseaux bayésiens et de réseaux bayésiens causaux, en utilisant différents types de connaissances à disposition. Au cours de ces trois années de recherche, nous avons développé deux algorithmes. Ces deux algorithmes partagent des hypothèses communes à propos des données. D'abord, nous travaillerons principalement sur des données discrètes. Cela signifie que les données sont issues de distributions de probabilités de variables discrètes. De plus, nous utiliserons l'hypothèse de fidélité qui assure que la distribution sur laquelle nous travaillons admet l'existence d'un graphe pouvant en représenter toutes les indépendances conditionnelles. Cette hypothèse pourra être discutée et partiellement relâchée ultérieurement. Enfin, nous utiliserons également l'hypothèse de suffisance causale, qui nous permet d'être sûrs que toutes les relations entre les variables peuvent être expliquées avec des relations causales.

Nous prendrons également d'autres hypothèses, spécifiques à chaque algorithme. D'abord, le modèle recherché : dans un cas, nous chercherons uniquement le squelette d'un réseau bayésien. Cela laisse alors la liberté de continuer l'orientation et la complétion de la structure par d'autres moyens. Dans l'autre cas, nous chercherons un réseau bayésien causal. Ce dernier imposera des contraintes plus fortes sur l'apprentissage de la structure, et son utilisation en combinaison avec d'autres algorithmes sera plus limitée. L'autre divergence importante porte sur les connaissances utilisées. Le premier algorithme doit permettre d'utiliser des connaissances portant sur les indépendances et dépendances elles-mêmes. Ces connaissances peuvent être

très utiles dans le cadre d'apprentissage sous contraintes. C'est d'ailleurs dans cette catégorie que se situe ce premier algorithme. Son innovation repose principalement sur le fait que des connaissances incertaines sont utilisables dans le cadre d'un algorithme sous contraintes. Le deuxième algorithme s'intéresse à l'apprentissage causal. Il doit permettre d'utiliser des connaissances incertaines portant donc sur les relations causales. Ici, l'objectif est donc d'intégrer des connaissances incertaines sur les relations causales lors d'un apprentissage complet, tout en optimisant l'utilisation de ces connaissances durant l'intégralité de l'algorithme. Cela nous amène à une dernière divergence, qui est la structure de l'algorithme. Le premier algorithme est constitué d'une unique étape sous contraintes, alors que le second est divisé en trois étapes qui visent chacune à se compléter pour aboutir à un graphe le plus complet possible.

Pour présenter l'ensemble de ces travaux, cette thèse est divisée de la manière suivante. Les deux premiers chapitres sont dédiés à l'état de l'art. Dans le chapitre 1, nous étudierons les réseaux bayésiens. Nous présenterons un ensemble de définitions et de propriétés permettant de comprendre comment ces modèles peuvent représenter des systèmes complexes. Une partie de ce chapitre sera dédiée à la causalité et aux réseaux bayésiens causaux. Nous verrons plus spécifiquement ce que représente la causalité dans le cadre des travaux de Judea Pearl. Dans le chapitre 2, nous nous intéresserons à l'apprentissage des réseaux bayésiens et réseaux bayésiens causaux. Après un bref aperçu de l'apprentissage des paramètres, nous verrons comment il est possible d'utiliser des données pour apprendre une structure, et ce par le biais d'exemples pour chacune des trois approches citées plus haut : les approches sous contraintes, les approches à base de score et les approches hybrides. Dans ce chapitre, nous verrons également les propositions de la littérature pour intégrer les connaissances, certaines et incertaines, dans ces algorithmes d'apprentissage de structure. Enfin, une partie de ce chapitre est dédiée à l'apprentissage causal, avec les difficultés que ce problème implique. Les chapitres 3 et 4 sont, quant à eux, destinés à la présentation des deux algorithmes développés lors de cette thèse, et adoptent un plan similaire. Dans le chapitre 3, nous verrons l'algorithme sous contraintes d'apprentissage de structure utilisant des connaissances d'experts incertaines. Nous commencerons par voir les hypothèses utilisées et la nature des connaissances. Nous verrons ensuite les développements mathématiques au cœur de cette méthode, puis comment nous avons conçu l'algorithme pour accueillir ces développements. Des expérimentations sont présentées à la fin de ce chapitre pour en présenter les avantages. Le chapitre 4 est consacré quant à lui à l'algorithme hybride d'apprentissage causal utilisant des connaissances incertaines. Nous détaillerons encore une fois les hypothèses utilisées, puis nous verrons chacune des étapes de cet algorithme et des expérimentations montrerons le fonctionnement et les avantages de l'algorithme. Enfin, une conclusion viendra clore cette thèse.

1 État de l'art

Sommaire

1.1 Introduction 20
1.2 Réseaux bayésiens 21
1.2.1 Définitions 21
1.2.2 Indépendance 26
1.2.3 Différents types de connexions 27
1.2.4 D-séparation 29
1.2.5 Couverture de Markov 31
1.2.6 I-map, D-map et P-map 32
1.2.7 Classe d'équivalence de Markov 34
1.3 Causalité et réseaux bayésiens causaux 35
1.3.1 Représentation Graphique de la causalité 36
1.3.2 Intervention 38
1.3.3 Identifiabilité 40
1.4 Conclusion 43

1.1 Introduction

Les réseaux bayésiens, ainsi que les réseaux bayésiens causaux, sont des modèles graphiques probabilistes permettant de représenter des systèmes complexes par le biais de graphes et de probabilités. Ils permettent également de mener des raisonnements poussés en se basant sur les éléments graphiques et probabilistes. Les réseaux bayésiens causaux, en particulier, relèvent en plus de la logique causale, ce qui permet de conduire des raisonnements encore plus avancés.

De tels modèles demandent une formalisation importante que nous allons voir dans ce premier chapitre. Nous commencerons par donner les définitions des éléments constitutifs d'un réseau bayésien, c'est-à-dire le graphe et la distribution de probabilité. Nous parcourrons ensuite un ensemble de propriétés permettant aux réseaux bayésiens de fonctionner correctement. Enfin, une dernière partie sera consacrée aux réseaux bayésiens causaux, leurs définitions et leurs propriétés.

1.2 Réseaux bayésiens

1.2.1 Définitions

Les réseaux bayésiens sont des modèles graphiques probabilistes. Cela signifie que ces modèles s'appuient sur deux éléments : un graphe et une distribution de probabilité. Avant de donner la définition précise d'un réseau bayésien, intéressons-nous d'abord à la définition de ces deux éléments constitutifs.

Définition 1.2.1 (Graphe (BERGE 1963)) *Un graphe est un couple* $G = (\mathbf{V}, \mathbf{E})$ *avec :*
- $\mathbf{V}$ *un ensemble de sommets, aussi appelés nœuds*
- $\mathbf{E}$ *un ensemble d'arêtes reliant des couples de sommets*

En d'autres termes, il s'agit d'une structure constituée de nœuds reliés par des arêtes. Les graphes sont des outils intuitifs très utilisés pour la modélisation. Pour mieux adapter le graphe au système que l'on souhaite modéliser, il est possible de spécifier les propriétés du graphe. Par exemple, on peut avoir toutes, certaines ou aucune arête orientée dans le graphe, ce qui nous donne un graphe orienté, partiellement orienté ou non orienté.

Intéressons-nous au vocabulaire lié aux graphes orientés et partiellement orientés. Une arête orientée, appelée **arc**, est symbolisée par une flèche qui peut être dirigée vers l'un ou l'autre des nœuds. Si une arête est dirigée d'un nœud X vers un nœud Y, nous dirons dans ce manuscrit que X est un **parent** de Y et que Y est un **enfant** de X. On note $Pa_G(X)$ l'ensemble des parents de X dans G et $Ch_G(X)$ l'ensemble des enfants de X dans G. Lorsqu'il n'y aura pas de confusion possible sur le graphe, nous pourrons raccourcir les notations en $Pa(X)$ et $Ch(X)$. Lorsqu'il existe une succession d'arêtes dirigées, toutes dans le même sens, d'un nœud X vers un nœud Y, nous parlons de **chemin** dont X est la source et Y est le but. Lorsque la source et le but sont confondus, le chemin est appelé **circuit**. Les équivalents des chemins et des circuits dans les graphes non orientés sont respectivement les **chaînes** et les **cycles**. On peut également utiliser ces deux derniers termes dans des graphes orientés dès lors que l'orientation n'importe pas. De plus, on appelle **ancêtre** d'un nœud X tout nœud Y tels qu'il existe un chemin de Y vers X, et **descendant** tout nœud Z tel qu'il existe un chemin de X vers Z. Les ancêtres et descendants de X sont respectivement notés $An_G(X)$ et $Desc_G(X)$, ou plus simplement $An(X)$ et $Desc(X)$ lorsqu'il n'y a pas de méprise possible sur le graphe. Enfin, on appelle **squelette** d'un graphe G le graphe non orienté formé en remplaçant tous les arcs par des arêtes.

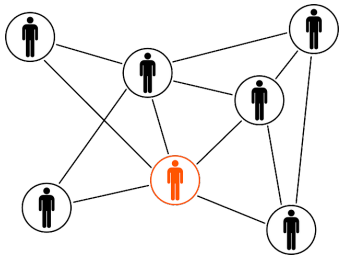


FIGURE 1.1 – Représentation graphique d'un réseau social.

Exemple 1.2.1 *Les graphes peuvent être utilisés pour représenter une grande diversité de systèmes. Par exemple, les réseaux sociaux sont modélisables sous la forme de graphes, ce qui permet de facilement déterminer les propriétés et paramètres du réseau grâce à des mesures faites directement sur le graphe.*

On peut maintenant définir une catégorie bien précise de graphes, les **DAG**.

Définition 1.2.2 (DAG) *Un DAG, pour Directed Acyclic Graph en anglais, est un graphe totalement orienté sans circuit.*

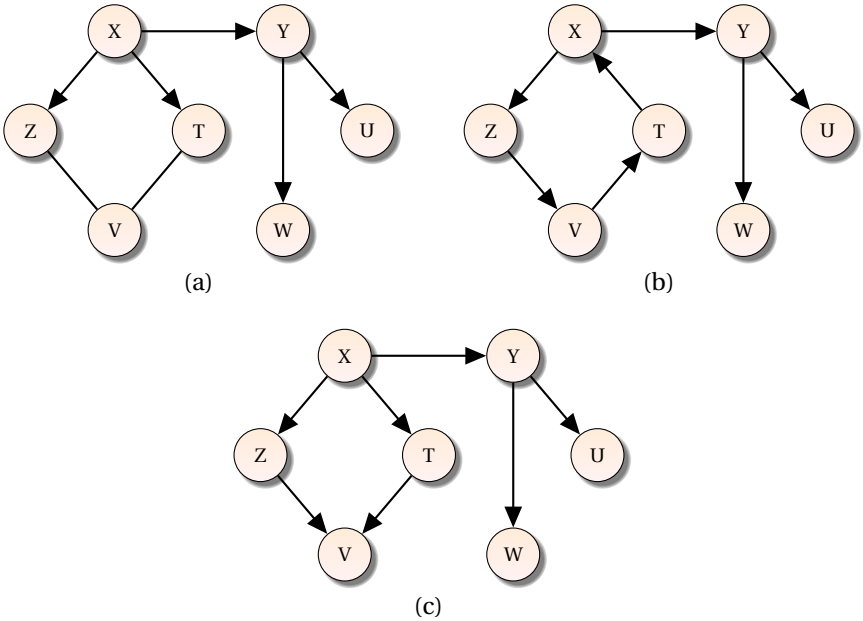


FIGURE 1.2 – Exemples de graphes. Les graphes 1.2a et 1.2b ne sont pas des DAG alors que le graphe 1.2c en est un.

Exemple 1.2.2 *Nous pouvons voir sur la figure 1.2 trois exemples de graphes. Les deux premiers ne sont pas des DAG. Pour le graphe 1.2a, la présence d'arêtes non orientées indique qu'il ne s'agit pas d'un DAG. Le graphe 1.2b présente des arcs uniquement, mais comporte un circuit. Enfin, le graphe 1.2c remplit toutes les conditions pour être un DAG.*

Un réseau bayésien est constitué d'un graphe et d'une distribution de probabilité. Définissons maintenant ce deuxième composant.

Définition 1.2.3 (Distribution de probabilités (KOLMOGOROV 1933)) *Soit Ω un ensemble d'événements, appelé univers, et S l'ensemble des parties de Ω. Une distribution de probabilité P est une application de S dans $[0,1]$ qui satisfait les conditions suivantes :*

- *$P(x) \geq 0$ pour tout élément $x \in S$*
- *$P(\Omega) = 1$*
- *Si $x, y \in S$ et $x \cap y = \emptyset$, alors $P\left(x \cup y\right) = P\left(x\right) + P\left(y\right)$*

Si en plus, pour tout élément x de S, on a $P(x) > 0$, on dit que la distribution est positive.

La première condition établit qu'une probabilité ne peut pas être inférieure à 0. La deuxième condition stipule qu'il est certain qu'au moins un des événements possibles survienne. Et enfin, la dernière condition spécifie que la probabilité de deux ensembles d'événements disjoints est la somme des probabilités des deux ensembles d'événements.

Exemple 1.2.3 *Prenons le cas classique d'un lancer de pièce non pipée. Les deux événements possible sont « pile » et « face », dans ce cas $\Omega = \{\text{«}pile\text{»}, \text{«}face\text{»}\}$. L'ensemble des parties de Ω est $S = \{\emptyset, \text{«}pile\text{»}, \text{«}face\text{»}, \Omega\}$. On a alors :*

- *$P(\emptyset) = 0$*
- *$P(\{\text{«}pile\text{»}\}) = 0,5$*
- *$P(\{\text{«}face\text{»}\}) = 0,5$*
- *$P(\{\text{«}pile\text{»}, \text{«}face\text{»}\}) = 1$*

La dernière probabilité indique que la pièce tombe bien soit sur « pile », soit sur « face ».

Dans l'exemple précédent, le résultat du lancer de pièce, qui peut valoir « pile » ou « face », est appelé **variable aléatoire**. On parle de variable aléatoire **discrète** lorsque l'univers est fini, et **continue** sinon. Une distribution de probabilité peut parfois concerner plusieurs variables aléatoires. C'est le cas, par exemple, si on s'intéresse au résultat du lancer de plusieurs pièces simultanément. Dans ce cas, S est égale à l'ensemble des parties de tous les univers, et les probabilités sont définies sur des événements qui joignent les différentes variables aléatoires. Une distribution de probabilité définie sur plusieurs variables aléatoires est appelée **distribution jointe**, et **distribution jointe** lorsque la distribution ne concerne qu'une seule variable. Il est possible de passer d'une loi jointe à une loi marginale en utilisant la **formule des probabilités totales**.

Théorème 1.2.1 (Formule des probabilités totales) *Soit X et Y deux variables aléatoires discrètes, alors on a :*

$$P(X) = \sum_{Y} P(X, Y) \tag{1.1}$$

Enfin, on peut également s'intéresser aux probabilités d'une variable aléatoire sachant l'état d'une ou plusieurs autres variables aléatoires. On parle alors de **distribution de probabilité conditionnelle**, qu'on note $P(X|\mathbf{Y})$ pour la probabilité de X sachant un ensemble de variables $\mathbf{Y}$. Cette probabilité peut se calculer à partir de la probabilité jointe entre X et $\mathbf{Y}$ et la probabilité jointe de l'ensemble $\mathbf{Y}$ seul :

$$P(X|\mathbf{Y}) = \frac{P(X,\mathbf{Y})}{P(\mathbf{Y})}$$

Il est alors possible de réécrire une distribution jointe en fonction des probabilités conditionnelles, c'est ce qu'on appelle la **formule des probabilités composées**.

Théorème 1.2.2 (Formule des probabilités composées) *Soit $X_1, ..., X_n$ n variables aléatoires, alors on a :*

$$P(X_1, ..., X_n) = P(X_1) \times P(X_2|X_1) \times P(X_3|X_2, X_1) \times ... \times P(X_n|X_{n-1}, ..., X_2, X_1) \quad (1.2)$$

De plus, les lois conditionnelles de deux variables peuvent être reliées l'une à l'autre grâce au **théorème de Bayes**.

Théorème 1.2.3 (Théorème de Bayes (BAYES et PRICE 1763)) *Soit X et Y deux variables aléatoires de loi marginale positive, alors on a :*

$$P(Y|X) = \frac{P(X|Y) \times P(Y)}{P(X)} \quad (1.3)$$

Un réseau bayésien est donc l'association d'un graphe et d'une distribution de probabilité où chaque nœud du graphe va représenter une variable aléatoire. En d'autres termes, les nœuds du graphe vont correspondre exactement aux variables aléatoires de la distribution jointe. Nous pouvons maintenant définir le réseau bayésien de manière plus formelle.

Définition 1.2.4 (Réseau bayésien (PEARL 1988)) *On appelle réseau bayésien un couple $B = (G, \Theta)$ où $G = (\mathbf{V}, \mathbf{E})$ est un DAG et $\Theta = \{P(X|Pa(X))\}_{X \in \mathbf{V}}$ l'ensemble des distributions de probabilité conditionnelle (CPD) des nœuds/variables X de G sachant leurs parents $Pa(X)$ dans G. Le réseau bayésien encode la probabilité jointe sur $\mathbf{V}$ définie par*

$$P(\mathbf{V}) = \prod_{X \in \mathbf{V}} P(X|Pa(X)) \quad (1.4)$$

La formule 1.4 est appelée **factorisation** de la distribution encodée par le réseau bayésien.

Ainsi, un réseau bayésien possède une structure sous la forme d'un DAG et des paramètres sous la forme de distributions de probabilité conditionnelle associées à chaque nœud. Dans la suite de ce mémoire, nous parlerons indifféremment de variables et de nœuds, les deux étant équivalents dans un réseau bayésien. Prenons un exemple concret de réseau bayésien pour mieux comprendre.

Exemple 1.2.4 *Cet exemple est inspiré d'un exemple du livre de* PEARL *2009. La figure 1.3 montre le réseau bayésien qui encode une distribution de probabilité sur 4 variables aléatoires : Nuage (N), Pluie (P), Arrosage automatique (A) et Herbe mouillée (H). Chaque variable est binaire : elle peut prendre la valeur "vraie" (v) ou "faux" (f). La valeur de N peut influencer la valeur de P et A, et ces deux dernières peuvent influencer la valeur de H. L'absence d'arc entre P et A ou entre C et H montre l'absence d'influence directe entre les variables de chaque paire. On associe à chaque variable une table de probabilité conditionnelle qui décrit la probabilité de la variable sachant l'état de ses parents. On voit ici un des avantages des réseaux bayésiens. Ceux-ci sont une représentation compacte d'une distribution jointe. En effet, si on voulait représenter toutes les probabilités de cette dernière, il faudrait utiliser* 16 *paramètres (*$2^4 = 16$*). Sachant que les probabilités somment à* 1*, il est possible de n'en spécifier que 15. Mais sous la forme d'un réseau bayésien, il ne faut plus que* 9 *paramètres (*$1+2+2+4=9$*, toujours en utilisant le fait que les probabilités somment à* 1 *pour chaque variable). De plus, les probabilités conditionnelles sont plus simples à manipuler pour l'homme que les distributions jointes, donc chacun des paramètres qu'il est nécessaire de spécifier est plus intuitif.*

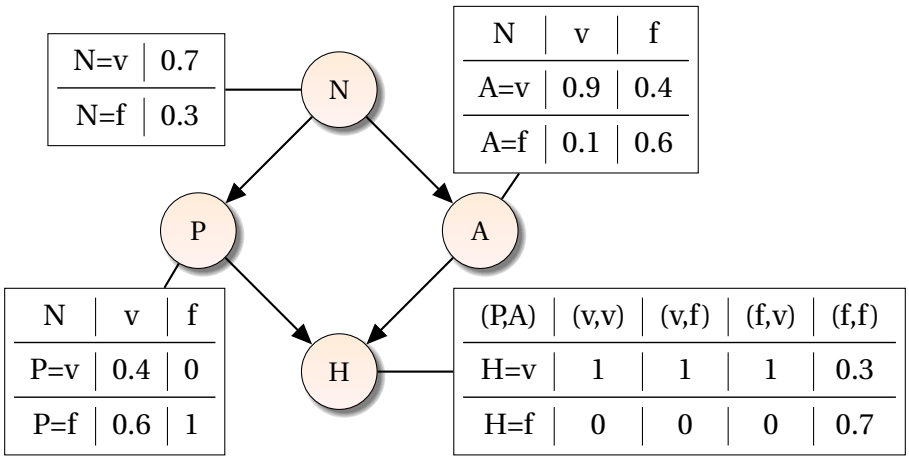


FIGURE 1.3 – Exemple de réseau bayésien avec ses CPD.

Un réseau bayésien est un outil de raisonnement probabiliste. Il est possible de donner des requêtes au réseau, c'est-à-dire des demandes d'informations, souvent en échange d'autres informations appelées **informations élémentaires**. Elles prennent la forme de variables aléatoires, généralement notées **E** et dont l'instanciation est notée **e**. Celles-ci vont venir modifier les probabilités de certains nœuds. Le raisonnement consiste donc propager l'information élémentaire pour modifier les probabilités qui doivent l'être, puis renvoyer le résultat de la requête soumise. Différentes requêtes sont possibles. On retrouve notamment, avec **V** l'ensemble des variables :

— les requêtes de probabilités conditionnelles, où on recherche la probabilité d'un groupe de variables **X** sachant un ensemble disjoint d'informations élémentaires **E** : $P(\mathbf{X}|\mathbf{E}=\mathbf{e})$

— les requêtes de l'explication la plus probable (MPE pour *Most Probable Explanation* en anglais) cherchent à trouver les valeurs de l'ensemble des variables $\mathbf{X} = \mathbf{V} - \mathbf{E}$ les plus probables sachant un ensemble d'informations élémentaires $\mathbf{E}$: $MPE(\mathbf{X}|\mathbf{e}) = \text{argmax}_{\mathbf{x}\in\mathbf{X}} P(\mathbf{x}, \mathbf{e})$
— les requêtes de *maximum a posteriori* (MAP) sont une généralisation des requêtes MPE. Dans ces dernières, on cherche l'instanciation de probabilité maximum pour toutes les variables, informations élémentaires mises à part. Ici, on cherche l'instanciation de probabilité maximum pour un sous-ensemble réduit de variables aléatoires. Si on note $\mathbf{X}$ ce sous-ensemble, et $\mathbf{Y} = \mathbf{V} - \mathbf{X} - \mathbf{E}$, alors la requête MAP recherche : $MAP(\mathbf{X}|\mathbf{e}) = argmax_{\mathbf{x}\in\mathbf{X}} \sum_{\mathbf{Y}} P(\mathbf{x}, \mathbf{Y}|\mathbf{e})$

1.2.2 Indépendance

La notion d'indépendance est une notion importante en probabilité et fondamentale pour les réseaux bayésiens. Elle désigne un type de relation existant entre deux variables.

Définition 1.2.5 (Indépendance) *On dit que deux variables aléatoires X et Y sont indépendantes, noté $X \perp Y$, si et seulement si on a : $P(X, Y) = P(X) \times P(Y)$*

Une propriété importante relie la notion d'indépendance et de probabilité conditionnelle : deux variables aléatoires X et Y sont indépendantes si et seulement si $P(X|Y) = P(X)$ et $P(Y|X) = P(Y)$. En d'autres termes, lorsque deux variables aléatoires sont indépendantes, les probabilités marginales et conditionnelles sont égales.

La notion d'indépendance entre deux variables, appelée indépendance marginale, peut être étendue au conditionnement.

Définition 1.2.6 (Indépendance conditionnelle) *On dit que deux variables aléatoires X et Y sont indépendantes sachant une troisième Z, noté $X \perp Y|Z$, si et seulement si on a : $P(X, Y|Z) = P(X|Z) \times P(Y|Z)$.*

Identiquement à l'indépendance marginale, X et Y sont deux variables aléatoires indépendantes sachant Z si et seulement si on a $P(X|Y, Z) = P(X|Z)$ et $P(Y|X, Z) = P(Y|Z)$.

Il est important de noter que deux variables peuvent être indépendantes conditionnellement, mais dépendantes marginalement. L'état de dépendance ou d'indépendance entre deux variables peut donc dépendre de la connaissance d'autres variables. Enfin, il est ci-dessus question d'indépendance entre deux variables sachant une unique troisième variable. Mais il est évidemment envisageable de considérer des ensembles de variables **X**, **Y** et **Z**. Dans ce cas, l'indépendance entre **X** et **Y** sera observée lorsque nous aurons des connaissances sur chacune des variables de l'ensemble **Z**.

Ces deux notions sont fondamentales pour comprendre le lien étroit existant entre la structure du réseau bayésien et la distribution jointe qu'il encode. En effet, un réseau bayésien va permettre de raisonner sur la distribution de probabilité, à partir

d'information sur certaines variables et dans le but d'obtenir de nouvelles informations sur d'autres variables. Le réseau bayésien va être construit de telle sorte que sa structure puisse retranscrire (au moins en partie, comme nous le verrons plus tard) les indépendances de la distribution que l'on veut représenter. Les indépendances encodées dans le réseau bayésien dictent le cheminement de l'information au sein du graphe. En d'autres termes, l'information peut circuler d'un nœud à un autre si les nœuds sont dépendants.

Avant d'aller plus loin dans l'étude de cette connexion existante entre le réseau bayésien et la distribution de probabilité, il est donc nécessaire de s'intéresser davantage à l'ensemble des indépendances conditionnelles de la distribution de probabilité. Pour cela, définissons un modèle d'indépendances.

Définition 1.2.7 (Modèle d'indépendance (DAWID 1979)) *Un modèle d'indépendance est un couple $M = (\mathbf{V}, I)$, où V est un ensemble de variables aléatoires et I est une relation assignant une valeur de vérité à chaque prédicat de la forme « $\mathbf{X}$ est indépendant de $\mathbf{Y}$ sachant $\mathbf{Z}$ », noté $I(\mathbf{X},\mathbf{Y}|\mathbf{Z})$, où $\mathbf{X},\mathbf{Y},\mathbf{Z} \subset \mathbf{V}$ sont trois ensembles disjoints. La relation I est appelée relation d'indépendance.*

À partir d'une distribution de probabilité jointe, il est possible de construire un modèle d'indépendances, notée $I(P)$. Ce modèle permet de représenter l'ensemble des indépendances qu'on peut trouver entre les variables d'une distribution jointe. Cette représentation est compacte, il n'est pas nécessaire de lister exhaustivement toutes les indépendances, celles-ci peuvent être retrouvées à l'aide d'un ensemble de 4 règles. Toutes les distributions correctement construites (qui respectent la définition 1.2.3) admettent un modèle d'indépendances avec ces 4 règles :

— Symétrie : $(\mathbf{X} \perp \mathbf{Y}|\mathbf{Z}) \Rightarrow (\mathbf{Y} \perp \mathbf{X}|\mathbf{Z})$
— Décomposition : $(\mathbf{X} \perp \mathbf{Y},\mathbf{W}|\mathbf{Z}) \Rightarrow (\mathbf{X} \perp \mathbf{Y}|\mathbf{Z})$
— Union faible : $(\mathbf{X} \perp \mathbf{Y},\mathbf{W}|\mathbf{Z}) \Rightarrow (\mathbf{X} \perp \mathbf{Y}|\mathbf{Z},\mathbf{W})$
— Contraction : $(\mathbf{X} \perp \mathbf{W}|\mathbf{Z},\mathbf{Y}) \wedge (\mathbf{X} \perp \mathbf{Y}|\mathbf{Z}) \Rightarrow (\mathbf{X} \perp \mathbf{Y},\mathbf{W}|\mathbf{Z})$

Un modèle d'indépendance qui respecte ces quatre propriétés est un **semi-graphoïde**. Lorsque la distribution est positive, on peut rajouter une cinquième règle :

— Intersection : $(\mathbf{X} \perp \mathbf{Y}|\mathbf{Z},\mathbf{W}) \wedge (\mathbf{X} \perp \mathbf{W}|\mathbf{Z},\mathbf{Y}) \Rightarrow (\mathbf{X} \perp \mathbf{Y},\mathbf{W}|\mathbf{Z})$

Un modèle d'indépendance qui vérifie ces cinq propriétés est alors appelé **graphoïde** (DAWID 1979).

1.2.3 Différents types de connexions

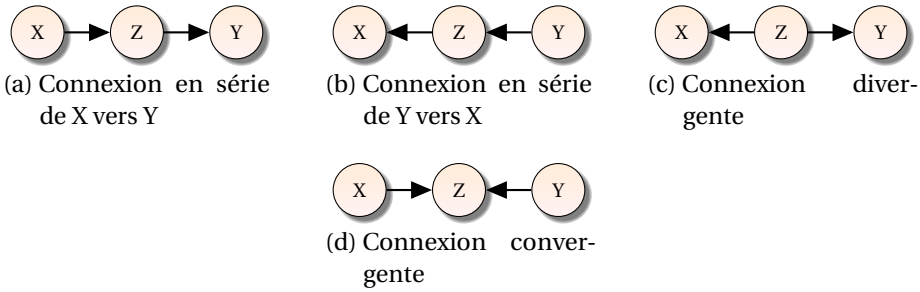
Comme dit précédemment, il est important de prendre en compte les indépendances de la distribution au sein du réseau bayésien. La structure même du réseau bayésien va, elle aussi, encoder des indépendances de sorte que l'information pourra ou non circuler d'un nœud à un autre. Ces indépendances forment également un modèle d'indépendance, qu'on va noter $I(G)$ pour un graphe G. Parmi les indépendances de la distribution d'un réseau bayésien qu'on peut retrouver dans sa structure, il y a les indépendances dites locales, déduites grâce à la **propriété locale de Markov**.

Théorème 1.2.4 (Propriété locale de Markov (KIIVERI, SPEED et CARLIN 1984)) *Soit* $B = (G, \Theta)$ *un réseau bayésien avec* $G = (\mathbf{V}, \mathbf{E})$ *le graphe et* Θ *l'ensemble des CPD. Pour une variable* $X \in \mathbf{V}$*, notons* $ND(X)$ *l'ensemble des* ***non-descendants*** *de* X *(*X *et ses parents exclus). Alors* B *encode dans sa structure les indépendances suivantes :*

$$\forall X \in \mathbf{V}, X \perp ND(X) | Pa(X) \tag{1.5}$$

Ces indépendances ne sont pas les seules qu'il est possible de retrouver dans le graphe. Pour découvrir les autres, une solution est d'utiliser les cinq axiomes du graphoïde. On obtient alors l'intégralité des indépendances encodées dans la structure $I(G)$. Il existe un autre outil permettant de trouver toutes ces indépendances. Pour comprendre ce dernier, qui sera introduit à la sous-section suivante, il est nécessaire de présenter quelques sous-structures de base des réseaux bayésiens.

Il existe trois sous-structures à trois nœuds servant de briques à la construction d'un DAG. Ces trois structures sont la connexion en série (dans un sens ou dans l'autre), la connexion convergente et la connexion divergente.



(a) Connexion en série de X vers Y

(b) Connexion en série de Y vers X

(c) Connexion divergente

(d) Connexion convergente

FIGURE 1.4 – Types de connexion entre trois nœuds d'un réseau bayésien

Sur les figures 1.4a et 1.4b, la connexion en série représente une indépendance de X et Y sachant Z. Les arêtes elles-mêmes montrent que X et Z sont dépendantes marginalement et conditionnellement l'une de l'autre, et de même pour Z et Y. Mais dans ce cas, la connaissance de la valeur exacte de Z empêche la circulation de l'information entre X et Y. En effet, intuitivement, X nous donne des informations sur Y en passant par Z. Donc si on connaît déjà l'état de Z, alors X ne peut plus rien nous apprendre sur Z et donc sur Y. Ainsi, la connexion en série représente une dépendance marginale entre X et Y, mais une indépendance entre X et Y sachant Z.

Sur la figure 1.4c, on observe une connexion divergente. Ici encore, cette sous-structure représente une indépendance entre X et Y sachant Z, comme pour la connexion en série. Intuitivement, à partir de Z, on peut avoir des informations sur X et Y. De même, à partir de X ou Y, on peut avoir des informations sur Z. Donc l'information peut circuler dans la connexion divergente. Cependant, si on a déjà des informations sur Z, alors ni X ni Y ne peuvent nous en fournir davantage et l'information ne peut plus circuler.

Enfin, la figure 1.4d représente une connexion convergente. Ici, les mécanismes sous-jacents sont différents. La sous-structure représente une indépendance marginale entre X et Y, mais une dépendance conditionnelle entre X et Y sachant Z, à l'inverse des deux connexions précédentes. Autrement dit, si on ne connait pas Z, alors l'information ne peut pas circuler entre X et Y. Par contre, si on connait Z, alors la variable peut servir de « pont » entre X et Y, et l'information peut circuler. Toujours pour donner l'intuition derrière ce mécanisme, X et Y sont deux variables qui ne partagent pas d'origine commune, ce qui explique qu'elles soient indépendantes. Par contre, si on connait Z, alors la connaissance sur X (respectivement Y) peut contraindre les possibilités de Y (respectivement X). En d'autres termes, l'information peut circuler dans la structure.

On voit donc que parmi ces trois structures, une se démarque particulièrement dans les indépendances qu'elle représente, la connexion convergente. Cette sous-structure est également appelée **v-structure**. La v-structure est centrale aussi bien dans la compréhension des réseaux bayésiens que dans l'apprentissage de leur structure. Elle permet entre autre de définir la propriété qui permet de généraliser la correspondance entre les indépendances du graphe et les indépendances d'une distribution de probabilité. Un nœud au centre d'une v-structure est appelé **collision** (*collider* en anglais).

Grâce à ces sous-structures, nous pouvons maintenant définir un critère important pour lire les indépendances d'un DAG, la **d-séparation**.

1.2.4 D-séparation

La d-séparation est une propriété très importante permettant de lire graphiquement les indépendances et dépendances entre les variables du réseau bayésien. Nous avons vu précédemment comment lire les indépendances d'une structure de trois variables. Mais il est possible de généraliser cette lecture à plus de variables, et même deux groupes de variables, séparés par plusieurs autres variables. Pour cela, intéressons-nous dans un premier temps à la notion de chaîne active et chaîne bloquée dans un graphe orienté.

Définition 1.2.8 (Chaîne active) *Soit une chaîne $C = (X_i)_{i \in I}$ dans un graphe $G = (\mathbf{V}, \mathbf{E})$ et $\mathbf{Z} \in \mathbf{V}$ un sous-ensemble de nœuds. On dit que C est une chaîne active sachant $\mathbf{Z}$ si et seulement si :*

- *pour toute v-structure sur C, le nœud central de la v-structure ou un de ses descendants appartient à $\mathbf{Z}$*
- *aucun élément de C qui n'est pas au centre d'une v-structure n'est dans $\mathbf{Z}$*

Une chaîne qui n'est pas active sachant $\mathbf{Z}$ est dite bloquée par $\mathbf{Z}$.

Cette définition reprend l'intuition donnée dans la section précédente sur la circulation de l'information, avec $\mathbf{Z}$ l'ensemble des nœuds sur lesquels on a des connaissances. Ici, l'information peut circuler dans une chaîne active, mais ne le peut pas dans une chaîne bloquée. En effet, si on a des connaissances sur le nœud central d'une v-structure, on a vu précédemment que cela permettait à l'information de passer,

contrairement au cas où on n'a aucune connaissance. C'est donc une condition nécessaire pour avoir une chaîne active. S'il n'y a pas de v-structure, alors on peut appliquer le même raisonnement que pour les connexions en série ou divergente : pour que l'information circule, il ne faut pas de connaissances sur les nœuds intermédiaires.

On peut maintenant donner la définition de la d-séparation.

Définition 1.2.9 (d-séparation (PEARL 1988)) *Soit $G = (\mathbf{V}, \mathbf{E})$ un DAG, pour tout triplet $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$ de sous-ensembles disjoints de $\mathbf{V}$, on dit que $\mathbf{X}$ est d-séparé de $\mathbf{Y}$ par $\mathbf{Z}$ si et seulement si toute chaîne $(X_i)_{i \in \{1 \dots p\}}$, avec p la longueur de la chaîne, $X_1 \in \mathbf{X}$ et $X_p \in \mathbf{Y}$, est bloquée par $\mathbf{Z}$.*

On peut évidemment prendre le cas particulier où X et Y sont des singletons, la propriété de d-séparation indique alors que les deux variables du réseau bayésien sont indépendantes conditionnellement à l'ensemble de variables $\mathbf{Z}$.

De la même façon que la notion de chaîne active existe aux côtés de celles de chaîne bloquée, la notion de d-connexion existe aux côtés de celle de d-séparation.

Définition 1.2.10 (d-connexion) *Soit $G = (\mathbf{V}, \mathbf{E})$ un DAG, pour tout triplet $(\mathbf{X}, \mathbf{Y}, \mathbf{Z})$ de sous-ensembles disjoints de $\mathbf{V}$, on dit que $\mathbf{X}$ est d-connecté de $\mathbf{Y}$ par $\mathbf{Z}$ si et seulement s'il existe une chaîne $(X_i)_{i \in \{1 \dots p\}}$ avec $X_1 \in \mathbf{X}$ et $X_p \in \mathbf{Y}$ active par rapport à $\mathbf{Z}$.*

La notion de d-connexion s'apparente quant à elle à la notion de dépendance conditionnelle et permet de lire celles de la distribution directement dans le graphe.

Exemple 1.2.5 *Sur la figure 1.5, nous pouvons voir le réseau bayésien Asia issu de l'article de* LAURITZEN *et* SPIEGELHALTER *1988. C'est un réseau bayésien largement utilisé dans la littérature. Il représente la distribution du diagnostic d'une personne pour une maladie des poumons, la dyspnée. Sur le graphe de ce réseau bayésien, nous voyons que les nœuds asia et x-ray sont marginalement d-connectés, c'est-à-dire que lorsque aucune variable n'est connue, alors les deux nœuds sont d-connectés. En effet, la chaîne asia→tub→either→x-ray ne comporte pas de v-structure (either n'est pas au centre d'une v-structure sur ce chemin). Sachant la variable tub, les variables asia et x-ray deviennent d-séparées, car le chemin est bloqué par la variable connue. Enfin, les variables asia et smoke sont marginalement d-séparées, car les deux chemins asia→tub→either←lung←smoke et*
asia→tub→either→dysp←bronc←smoke présentent respectivement une v-structure sur either et dysp.

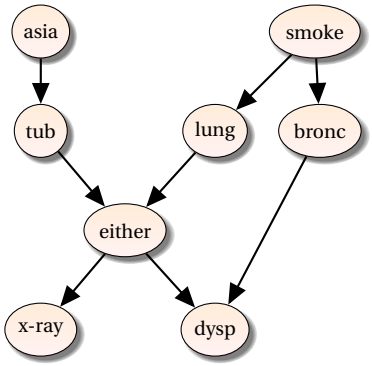


FIGURE 1.5 – Réseau bayésien Asia

La d-séparation est donc un critère permettant de lire toutes les indépendances contenue dans $I(G)$ directement dans le graphe. Grâce à ce critère, nous pouvons définir un ensemble propre à chaque nœud d'un DAG, appelé **couvertude de Markov**.

1.2.5 Couverture de Markov

Lorsque deux variables X et Y sont d-séparées dans le DAG par un ensemble de variable $\mathbf{Z}$, alors l'instanciation de $\mathbf{Z}$, c'est-à-dire le fait le fixer la valeur des variables aléatoire constitutives de $\mathbf{Z}$, permet de rendre X et Y indépendantes. Ainsi, la probabilité de X sachant Y et $\mathbf{Z}$ ne varie pas selon la valeur de Y. Dans le DAG d'un réseau bayésien, il est possible, pour toute variable, de trouver un ensemble de variables $\mathbf{Z}$ qui permette de d-séparer X de toutes les autres variables. Cet ensemble est appelé **couverture de Markov**.

Définition 1.2.11 (Couverture de Markov (PEARL 1988)) *Soit $G = (\mathbf{V}, \mathbf{E})$ un DAG, on appelle couverture de Markov de $X \in \mathbf{V}$ dans G, notée $MB_G(X)$, l'ensemble minimal pour lequel $X \perp Y | MB_G(X)$ pour tout $Y \in \mathbf{V} - \{X\} - MB_G(X)$.*

Cet ensemble permet donc d'isoler un nœud du reste du graphe. Ainsi, il n'est pas nécessaire d'avoir des informations sur les variables qui n'appartiennent pas à $X \cup MB(X)$ pour déterminer la distribution de la variable X. De plus, dans un réseau bayésien, cet ensemble est facilement identifiable puisqu'il correspond exactement à l'union des parents, des enfants et des autres parents des enfants (appelés **époux**).

Exemple 1.2.6 *Dans le graphe de la figure 1.6, on constate que si on instancie toutes les variables orange, alors il est possible de rendre indépendante la variable X du reste des variables du graphe. On peut aussi aisément voir pourquoi il est nécessaire de prendre en compte les époux. En effet, les variables X, Z et Y forment une v-structure. Si on n'instancie que Z, alors le chemin entre X et Y est actif. Il est donc requis d'instancier également Y pour bloquer le chemin entre X et les variables au-delà de Y.*

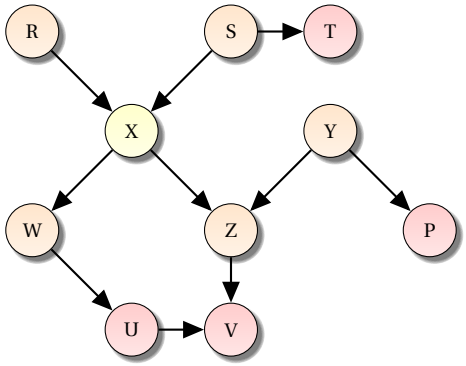


FIGURE 1.6 – Exemple de couverture de Markov. Le nœud concerné est X, en jaune, la couverture de Markov est en orange et les autres points sont en rouge.

Maintenant que nous nous sommes intéressés aux modèles d'indépendance d'un graphe, nous pouvons étudier sa relation avec le modèle d'indépendance d'une distribution encodée par un réseau bayésien.

1.2.6 I-map, D-map et P-map

L'ensemble des indépendances retranscrites dans le DAG d'un réseau bayésien forme un semi-graphoïde, voir un graphoïde lorsque la distribution est positive. Cette structure est de la même nature que l'ensemble des indépendances contenues dans une distribution. Ces deux objets sont donc comparables, et cette comparaison va nous permettre de spécifier plus précisément la relation entre une distribution de probabilité et le réseau bayésien utilisé pour la représenter.

Définition 1.2.12 (I-map, D-map et P-map (PEARL 1986)) *Soit P une distribution de probabilité et G un DAG. On dit que G est :*
- *une **I-map** si $I(G) \subseteq I(P)$*
- *une **D-map** si $I(G) \supseteq I(P)$*
- *une **P-map** si $I(G) = I(P)$*

Remarque 1.2.1 *Soit un ensemble de variables* **V**. *Le graphe complet G_c, c'est-à-dire le graphe où tous les nœuds sont reliés les uns aux autres, n'encode aucune indépendance. Ce graphe est donc une I-map pour l'ensemble des distributions sur* **V**. *Au contraire, le graphe vide G_v, sans aucune arête, n'encode que des indépendances. C'est donc une D-map pour l'ensemble des distributions sur* **V**.

Si le DAG G d'un réseau bayésien est une I-map d'une distribution P, alors en partant de la formule des probabilités composées, on peut retrouver la factorisation de la distribution du réseau bayésien. En d'autres termes, on a :

Théorème 1.2.5 *Soit G un DAG et P une distribution de probabilité, on a $I(G) \subseteq I(P)$ si et seulement si on peut réécrire la distribution de la façon suivante : $P(X_1, ..., X_n) = \prod_i P(X_i | Pa_G(X_i))$.*

On dit alors que **P** ***factorise selon*** **G**.

Une P-map permet de retranscrire toutes les indépendances d'une distribution, c'est donc une propriété très enviable lorsqu'on souhaite construire un modèle qui représente une distribution de probabilité. On peut alors se demander s'il existe une P-map pour toutes les distributions. Malheureusement, la réponse est négative. On peut en effet assez facilement construire une distribution telle que l'ensemble des indépendances conditionnelles ne soit pas représentable par un DAG (voir exemple 1.2.7). Il est alors nécessaire de recourir à des I-maps, qui permettent toujours de factoriser les distributions. Mais comme on l'a dit dans la remarque 1.2.1, le graphe complet est une I-map, évidemment très peu informative. Le graphe choisi doit donc être une I-map, mais qui se rapproche également le plus possible de la P-map. En d'autres termes, on cherche un graphe qui encode le plus possible d'indépendances. On appelle ce genre de graphe une **I-map minimale**.

Définition 1.2.13 (I-map minimale) *Soit P une distribution de probabilité. On appelle I-map minimale un DAG G telle que G est une I-map de P et que la suppression d'un seul arc du graphe G implique que G n'est plus une I-map.*

Exemple 1.2.7 *Il n'est pas toujours possible de trouver une P-map pour toutes les distributions de probabilité. En voici un exemple. Soit P une distribution sur les variables $\{X, Y, Z, T\}$. Les indépendances sont les suivantes :*

— $X \perp T | Y, Z$

— $Y \perp Z | X, T$

La figure 1.7a montre une première tentative pour représenter cet ensemble d'indépendances, mais n'est pas correcte, car X et T ne doivent pas être marginalement indépendants. La figure 1.7b présente une autre tentative, tout aussi incorrecte, car cette fois-ci Y et Z sont dépendants conditionnellement à X et T. On pourrait continuer à énumérer des tentatives comme cela pendant longtemps, mais en vain : il n'existe simplement pas de DAG qui soit une P-map de la distribution P.

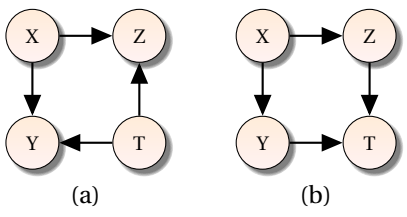


FIGURE 1.7 – Exemples de tentatives de P-map infructueuses.

Pour terminer cette sous-partie, il est nécessaire d'insister sur la dissociation qui est faite ici. Les modèles des indépendances de la distribution et du graphe ne sont pas systématiquement égaux, et il faut donc choisir le graphe dont l'ensemble des indépendances permet de représenter au mieux celui de la distribution pour espérer représenter cette dernière. En d'autres termes, il y a bien d'une part le réseau bayésien,

avec sa structure et ses CPD (ses paramètres), et d'un autre côté la distribution de probabilité qu'on cherche à modéliser grâce au réseau bayésien.

Notons également que différents graphes peuvent encoder le même modèle d'indépendance. Ces graphes, et les réseaux bayésiens dont ils font partie, peuvent être regroupés dans une classe d'équivalence, appelé **classe d'équivalence de Markov**, dont nous allons dorénavant voir les détails.

1.2.7 Classe d'équivalence de Markov

Un réseau bayésien encode grâce à sa structure un ensemble d'indépendances. Ces indépendances peuvent être retrouvées grâce à la notion de d-séparation abordée précédemment. Mais comme on l'a vu dans la sous-section 1.2.3, les sous-structures en série de X vers Y, de Y vers X ou divergente partagent les mêmes indépendances conditionnelles. Avec des graphes de tailles quelconques, on peut également trouver des graphes qui décrivent exactement les mêmes indépendances. On dit alors que les réseaux bayésiens associés à ces graphes sont **équivalents au sens de Markov**.

Définition 1.2.14 (Équivalence de Markov (VERMA et PEARL 1990)) *Deux réseaux bayésiens B_1 et B_2 sont dits équivalents au sens de Markov, noté $B_1 \equiv B_2$, si et seulement si $I(B_1) = I(B_2)$.*

Par abus de notation, on pourra également appliquer cette relation aux graphes et dire que deux graphes sont équivalents au sens de Markov.

Ainsi, dans les graphes de deux réseaux bayésiens équivalents au sens de Markov, toutes les relations pouvant être établies grâce à la d-séparation dans un graphe pourront se retrouver dans l'autre graphe. Comme toute relation d'équivalence, l'équivalence de Markov respecte les propriétés suivantes, avec B_1, B_2 et B_3 trois réseaux bayésiens :

— réfléxivité : $B_1 \equiv B_1$
— symétrie : $B_1 \equiv B_2 \iff B_2 \equiv B_1$
— transitivité : $B_1 \equiv B_2 \wedge B_2 \equiv B_3 \Rightarrow B_1 \equiv B_3$

À cette relation d'équivalence, on peut évidemment joindre une classe d'équivalence appelée **classe d'équivalence de Markov** qui est l'ensemble des réseaux bayésiens équivalents entre eux au sens de Markov, ou, toujours pas abus de notation, l'ensemble des graphes de ces réseaux bayésiens.

Toujours dans la sous-section 1.2.3, nous avons vu que la v-structure représentait des indépendances différentes des autres sous-structures, et celle-ci ne peut donc pas être remplacée par une autre sous-structure sans changer l'ensemble d'indépendances. Cela signifie que tous les graphes d'une même classe d'équivalence de Markov partagent ces v-structures. On peut également remarquer que deux squelettes différents mènent forcément à des ensembles d'indépendances différents, donc de la même manière les graphes d'une même classe d'équivalence partageront le même squelette. Ainsi, on peut construire un graphe partiellement orienté (**PDAG**) spécifique commun à tous les DAG d'une classe d'équivalence. Ce graphe représente la classe

d'équivalence et est construit en gardant le squelette et tous les arcs non inversibles (appartenant à une v-structure ou dont l'inversion engendre une nouvelle v-structure), et en changeant tous les arcs inversibles en arêtes. Un tel PDAG est appelé **CPDAG** pour *Completed* PDAG.

Exemple 1.2.8 *La figure 1.8 montre le CPDAG du réseau bayésien Asia. Il ne subsiste que les v-structures et l'arc de either vers x-ray, qui ne peut être orienté dans l'autre sens sous peine de faire apparaître de nouvelles v-structures.*

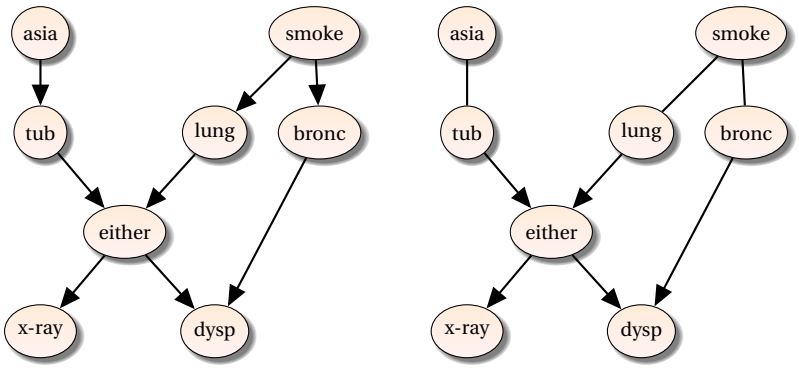


FIGURE 1.8 – (a) Le DAG du réseau bayésien Asia. (b) Le CPDAG du réseau bayésien Asia.

1.3 Causalité et réseaux bayésiens causaux

Comme nous l'avons vu précédemment, deux structures de réseaux bayésiens peuvent représenter les mêmes indépendances et donc être équivalents au sens de Markov. Les différences vont principalement résider dans l'orientation de certaines arêtes qui n'appartiennent pas à une v-structure et dont l'inversion n'ajoute aucune v-structure. Dans l'utilisation classique d'un réseau bayésien, on peut de ce fait choisir arbitrairement l'orientation de ces arêtes. Mais il existe une sous-classe de réseaux bayésiens qui ne permet pas une telle liberté.

On parle de **réseau bayésien causal** lorsqu'en plus des indépendances, la structure du réseau supporte également une sémantique causale. Dans ce cas, un arc de X vers Y signifie que X est une cause de Y. Ainsi, il n'est plus possible d'inverser les arcs aussi facilement, car quel que soit l'arc, que son inversion crée une v-structure ou non, son inversion affectera la sémantique causale du réseau bayésien. Avec cette nouvelle sémantique, le réseau bayésien acquiert de nouvelles capacités liées à la causalité. Pour étudier ces facultés, intéressons-nous d'abord à la causalité elle-même. Un historique plus détaillé peut être retrouvé dans l'épilogue de PEARL 2009. Nous verrons ensuite un ensemble de notions gravitant autour de la causalité.

Les premières traces de l'étude du concept de causalité, par Aristote notamment, relie cette notion à l'idée de but. Aristote parlait en effet de cause finale, qui est l'intention recherchée, l'objectif. Cette interprétation a survécu pendant longtemps, jusqu'à Galilée qui a mis la cause en opposition à la description : la description vient en premier, et répond à la question "Comment ?", puis vient la cause qui répond à la question "Pourquoi ?". La description est ce qui intéresse les scientifiques : ce sont les équations qui régissent le monde. Au contraire, la cause est la raison pour laquelle les choses arrivent. Cette idée va être développée par Humes qui va même suggérer que la notion de causalité est superflue en science, et que seule l'explication du mécanisme, le comment, importe. Enfin, Bertrand Russell dira même que la causalité est un reliquat des temps passés, uniquement préservé par le fait que sa présence n'est pas gênante. Cette idée est notamment soutenue par le fait que la causalité n'est pas retranscrite dans les lois de la physique. Une équation comme $f = m \times a$ n'a pas de direction privilégiée, et peut-être réécrite de différentes manières, mettant en avant l'une ou l'autre des grandeurs en jeu. Cette idée sera également partagée par Pearson qui verra la corrélation comme l'aboutissement final de la description de la relation entre deux choses, et qui suggère donc que la causalité n'a pas sa place dans le discours scientifique.

Cette idée que la causalité ne peut rien apporter à la science persistera, spécifiquement dans les publications scientifiques, où aucune mention à la causalité ne pourra être relevée. Mais paradoxalement, comme le relève Judea Pearl, le vocabulaire de la causalité continuera d'exister dans les discussions informelles. Cela signifie que l'idée de cause et d'effet est restée un outil de compréhension important pour les chercheurs, même si celui-ci n'apparait pas dans les résultats. Et c'est principalement ceci qui poussera Pearl à étudier la causalité, dans le but de redonner à la causalité la place qui lui revient selon lui.

1.3.1 Représentation Graphique de la causalité

Pour bien comprendre et utiliser la causalité, il est nécessaire de la représenter. Comme nous l'avons vu, les équations de la physique sont symétriques, contrairement à une relation de cause à effet. Il est donc nécessaire d'aller plus loin dans la représentation des mécanismes physiques. Et comme nous l'avons vu au tout début de ce manuscrit, les graphes sont des outils de modélisation très pratiques et très intuitifs. C'est donc vers ceux-ci que Pearl s'est tourné dans son étude de la causalité, et il a découvert qu'ils étaient des outils puissants pour la représentation de la causalité.

Les modèles utilisés se rapprochent énormément des réseaux bayésiens vus plus tôt, à quelques nuances près. Plutôt que de représenter les mécanismes causaux avec des tables de probabilités conditionnelles, les mécanismes seront ici représentés par des équations : chaque variable est le résultat d'une fonction qui a en argument ses causes et une perturbation extérieure. Cette relation est de la forme :

$$X = f_X(Pa(X), \epsilon_X) \tag{1.6}$$

où ϵ_X est la perturbation associée à la variable X. La fonction est déterministe, mais la perturbation suit une distribution de probabilité arbitraire $P(\epsilon_X)$, ce qui permet de préserver un aspect probabiliste et donc de parler de distribution pour les variables. Un modèle construit avec de telles fonctions est appelé **modèle structurel**. Comme pour les réseaux bayésiens, il est possible de construire un graphe $G = (\mathbf{V}, \mathbf{E})$ associé au modèle, où un arc de Y vers X appartient à $\mathbf{E}$ si Y apparait dans la fonction de X. Si G est acyclique, alors on dit que le modèle structurel est un modèle **semi-markovien**, et G est un DAG. La perturbation représente des facteurs extérieurs que le modélisateur a choisi de ne pas intégrer au modèle. Lorsque qu'en plus de l'acyclicité, toutes les perturbations sont indépendantes les une des autres, alors le modèle est dit **markovien**.

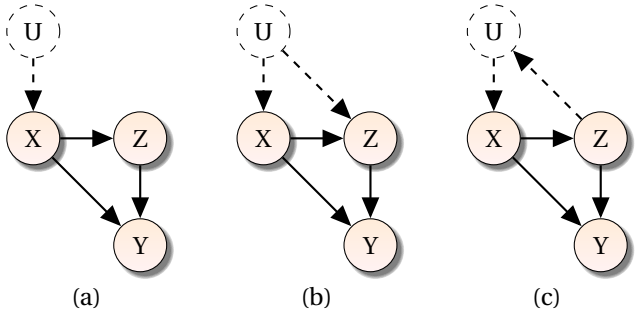


FIGURE 1.9 – Exemple de modèles. (a) Modèle markovien (b) Modèle semi-markovien (c) modèle non-markovien

Dans un modèle markovien, il est possible de ne pas représenter les perturbations dans le graphe. Par contre, lorsque les perturbations ne sont pas indépendantes les unes des autres, alors elles créent une dépendance entre les variables impactées par ces perturbations. La relation engendrée par cette dépendance n'implique pas de relation causale entre les deux variables, et ne peut donc pas être représentée dans le DAG par un unique arc. On ajoute alors au DAG un nœud qui représente une variable latente, modélisant des mécanismes de la nature non observés qui sont la cause de deux variables.

Il est intéressant de noter que tout modèle semi-markovien peut se rapporter sans perte à un modèle où chaque variable non observée n'a aucun parent et est parent d'exactement deux variables observées. C'est donc sous cette forme que sont généralement représentés les modèles semi-markovien.

Théorème 1.3.1 (VERMA 1993) *Tout modèle avec des variables latentes arbitraires (c'est-à-dire pouvant posséder autant de parents et autant de variables filles que nécessaire) peut être transformé en modèle semi-markovien. Cette transformation sauvegarde les relations d'indépendance et de causalité des variables observées.*

Mais ces modèles ne sont pas suffisants pour décrire et manipuler complètement la causalité. Pour cela, nous avons besoin d'un autre outil, appelé l'**intervention**.

1.3.2 Intervention

La causalité, contrairement à la corrélation, est asymétrique. Cela signifie que quelque chose différencie la cause de l'effet et que la relation a un sens. C'est ce sens que nous cherchons à décrire. Intuitivement, on sait par exemple que la pluie cause l'herbe mouillée, et pas l'inverse. Et si on veut s'en assurer, une expérience simple consiste à arroser l'herbe, puis voir si cela provoque la pluie. En d'autres termes, on agit sur le système et fixe artificiellement la variable "Herbe mouillée" pour observer l'impact de cette action sur le système, puis en tirer des conclusions. Une telle opération est appelée **intervention**.

L'intervention est un outil très important pour l'étude de la causalité. Elle diffère du conditionnement en ce sens que le conditionnement est une manipulation faite sur les observations : on pose un filtre sur les données pour n'observer que celles qui nous intéressent, par exemple celle où l'herbe est mouillée. Une intervention est une manipulation sur le système lui-même : on fixe la valeur d'une ou plusieurs variables pour voir l'impact que cela a sur le système.

L'intervention permet notamment de définir les notions de cause et d'effet :

Définition 1.3.1 (Cause et effet) *On dit que X est une cause de Y, et que Y est un effet de X lorsqu'il est possible de modifier la valeur de Y par une intervention sur la variable X.*

Avec l'observation, l'intervention est une des deux activités nécessaires à la recherche. Heureusement, l'observation peut être décrite par de nombreux outils mathématiques, ce qui permet de tirer diverses connaissances venant de base de données observationnelles. Les probabilités en font partie : il est possible d'exprimer la probabilité qu'il ait plu lorsqu'on observe que l'herbe est mouillée, à travers la quantité $P(P|H)$, en reprenant les notations de la figure 1.3. À l'inverse, l'intervention n'est pas autant mise en valeurs par les outils mathématiques existants, et il n'est pas évident d'exprimer la probabilité qu'il pleuve sachant qu'on a mouillé l'herbe. C'est pourquoi des chercheurs, dont Pearl, ont développé un outil pour ce but, matérialisé sous la forme de l'opérateur "$do()$" (GOLDSZMIDT et PEARL 1992). Par exemple, la probabilité qu'il ait plu sachant qu'on a mouillé l'herbe peut s'écrire : $P(P|do(H = v))$.

Comme on l'a dit, l'intervention revient à fixer artificiellement la valeur d'une variable, indépendamment des autres variables. Or les variables qui déterminent normalement ses valeurs sont ses causes. Donc graphiquement, intervenir sur une variable revient à supprimer les arêtes entre une variable et ses causes, comme présenté sur la figure 1.10. Dans cet exemple, les variables N, P, A, H et G correspondent respectivement à Nuage, Pluie, Arrosage, Herbe et Glissade.

Pour parler plus aisément de ces graphes tronqués, avec G un DAG, on notera :

— $G_{\overline{X}}$ le graphe obtenu en supprimant de G toutes arêtes pointant vers X

— $G_{\underline{X}}$ le graphe obtenu en supprimant de G toutes arêtes sortant de X

— $G_{\overline{\underline{X}}}$ le graphe obtenu en supprimant de G toutes arêtes entrant et sortant de X

L'opérateur $do()$ n'est pas suffisant en lui-même, il faut aussi l'accompagner de règles de calcul, comme la commutativité et associativité pour l'addition. Dans le cas

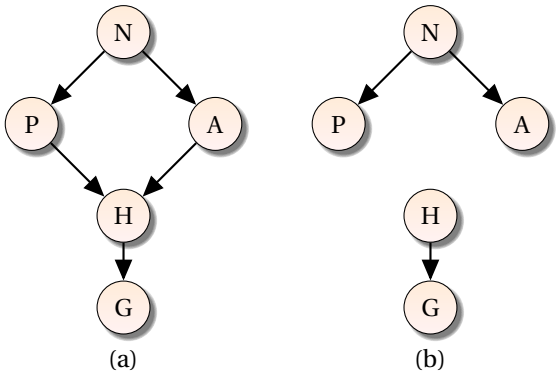


FIGURE 1.10 – Représentation graphique d'une intervention sur H. (a) Le réseau causal d'origine. (b) Le réseau causal sur lequel l'intervention sur H a eu lieu.

de l'opérateur $do(\,)$, ces règles s'appellent **do-calculus**, et sont au nombre de 3. Elles permettent de manipuler algébriquement les interventions pour que celles-ci soient pleinement utilisables. Pour simplifier, toutes les variables ci-dessous sont réduites à un singleton, mais ces règles fonctionnent également avec des ensembles de variables.

Théorème 1.3.2 (PEARL 1995) *Soit G un DAG associé à un modèle semi-markovien et P la distribution de probabilité induite par le modèle. Pour tout sous-ensemble disjoint de variables X, Y, Z et W, on a :*

- ***Règle 1*** *: Insertion et suppression d'observations*
  Si $(Y \perp Z|X,W)_{G_{\overline{X}}}$ *alors* $P(Y|do(X),Z,W) = P(Y|do(X),W)$
- ***Règle 2*** *: Permutations d'interventions et observations*
  Si $(Y \perp Z|X,W)_{G_{\overline{X}\underline{Z}}}$ *alors* $P(Y|do(X),do(Z),W) = P(Y|do(X),Z,W)$
- ***Règle 3*** *: Insertion et suppression d'interventions*
  Si $(Y \perp Z|X,W)_{G_{\overline{X}\,\overline{Z(W)}}}$ *alors* $P(Y|do(X),do(Z),W) = P(Y|do(X),W)$
  où $Z(W)$ est l'ensemble des nœuds de Z qui ne sont pas des ancêtres d'un nœud de W dans $G_{\overline{X}}$

La première règle indique que la d-séparation implique toujours une indépendance avec des variables qui ont subi une intervention, et donc qu'il est possible d'ignorer certaines variables (ici la variable Z) dans le calcul de la probabilité conditionnelle.

La deuxième règle indique qu'une intervention est équivalente à un conditionnement si la variable sur laquelle on intervient (Z) est un ancêtre de la variable étudiée (Y) et qu'il n'existe pas de chemin actif alternatif entre les deux. On peut donc remplacer $do(Z)$ par Z dans ce cas, et inversement.

La troisième règle indique que si une variable sur laquelle on fait une intervention devient indépendante d'une seconde variable, alors l'intervention n'a aucun effet sur la seconde variable. On peut donc supprimer l'intervention du calcul de la probabilité conditionnelle, ou au contraire l'ajouter si les conditions sont réunies.

Ces trois règles permettent donc d'ajouter et supprimer des opérateurs $do(\,)$, ainsi que modifier l'expression d'une probabilité en présence d'un tel opérateur. En somme, ces opérateurs permettent de manipuler des expressions incluant des interventions.

1.3.3 Identifiabilité

La distribution $P(Y|do(X))$ est appelé **effet causal** de X sur Y et représente l'impact de la valeur de X sur celle de Y, et donc l'influence causale pouvant exister à travers le mécanisme physique qui relie ces deux variables. Connaître cette distribution permet de rendre compte de l'impact causal d'une variable sur une autre : est-ce que c'est une cause majeure ou un facteur mineur? Pour déterminer l'effet causal, la façon la plus directe consiste à utiliser une base de données dite **interventionnelle**. C'est une base de données obtenue en mesurant les variables d'un système sur lequel on a effectué une intervention, dans notre cas l'intervention $do(X)$. Mais l'obtention de bases de données interventionnelle soulève diverses problématiques de plusieurs ordres :

— économique, lorsque l'intervention coûte trop cher. Par exemple donner un salaire mensuel de 10000 € à un pays entier pour observer l'effet sur l'économie
— éthique, lorsque l'intervention n'est pas éthiquement faisable. Par exemple, obliger des personnes à fumer uniquement pour étudier l'effet de la cigarette
— pratique, lorsque l'intervention n'est simplement pas réalisable. Par exemple, stopper totalement l'émission de dioxyde de carbone pour observer l'effet sur le climat.

Il n'est donc pas toujours possible de mener des expériences permettant d'accéder à un système sous intervention, et il faut alors se satisfaire de bases de données observationnelles, c'est-à-dire des bases de données qui ont été relevées sur des systèmes soumis à aucune intervention. Malheureusement, il n'est pas toujours possible de déterminer l'effet causal entre deux variables à partir d'une base de données observationnelle. Par exemple, sur la figure 1.11, la variable Y est soumise à l'influence directe de X, mais également à celle de la variable non observée U à travers la variable X, ces deux influences se combinent et ne sont pas distinguables l'une de l'autre.

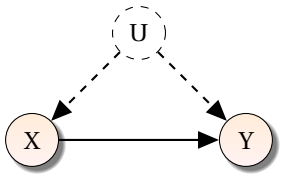


FIGURE 1.11 – Exemple d'effet causal non identifiable

Le problème de détermination de l'effet causal entre deux variables dans un modèle markovien ou semi-markovien s'appelle **problème d'identifiabilité**. Dans un modèle markovien, le problème se résout assez simplement grâce au théorème suivant :

Théorème 1.3.3 *Soit un DAG G associé à un modèle markovien M dans lequel un sous-ensemble de variables* **V** *sont observées, l'effet causal* $P(Y|do(X))$ *est identifiable si*

$\{X \cup Y \cup Pa(X)\} \subseteq \mathbf{V}$. *On a alors :*

$$P(Y|do(X)) = \sum_{Z_i \in Pa(X)} P(Y|X, Z_i)P(Z_i) \tag{1.7}$$

Le problème d'identifiabilité est plus complexe dans le cas d'un modèle semi-markovien. En effet, comme vu sur la figure 1.11, des variables latentes peuvent empêcher de déterminer l'influence causale d'une variable sur une autre. Heureusement, il existe un ensemble de critères, le plus souvent graphique, permettant de savoir si un effet causal est identifiable ou non.

Le premier critère est appelé critère ***backdoor***.

Définition 1.3.2 (Critère backdoor) *Un ensemble de variables* $\mathbf{Z}$ *satisfait le critère backdoor relatif à une paire ordonnée de variables* (X, Y) *dans un DAG* G *si :*

— *aucun nœud dans* $\mathbf{Z}$ *n'est un descendant de* X

— *et* $\mathbf{Z}$ *bloque tous les chemin entre* X *et* Y *qui contiennent un arc vers* X

Théorème 1.3.4 (PEARL 1993) *Si un ensemble de variables* $\mathbf{Z}$ *satisfait le critère backdoor relatif à* (X, Y)*, alors l'effet causal de* X *sur* Y *est identifiable et donné par la formule :*

$$P(Y|do(X)) = \sum_{\mathbf{Z}} P(Y|X, \mathbf{Z})P(\mathbf{Z}) \tag{1.8}$$

Exemple 1.3.1 *Dans la figure 1.12, on observe que le nœud* Z *n'est pas un descendant de* X *et que* Z *bloque le seul chemin entre* X *et* Y *qui ne contienne pas d'arc sortant de* X*. On peut donc utiliser le critère backdoor. Le terme anglais backdoor, "porte dérobée" en français, fait référence au fait que* Z *bloque tous les passages alternatifs au chemin direct qui pourraient permettre de véhiculer l'influence de* X *sur* Y*.*

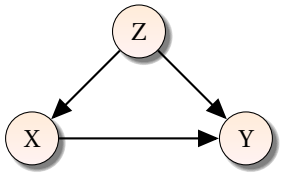


FIGURE 1.12 – Exemple de DAG sur lequel on peut appliquer le critère backdoor.

Un autre critère proche est le critère ***frontdoor***. Pour celui-ci, nous allons définir un **chemin *backdoor*** de X vers Y, qui est un ensemble d'arcs consécutifs entre X et Y, étant une chaîne active par rapport à l'ensemble vide, et tel qu'il ne soit pas un chemin de X vers Y.

Définition 1.3.3 (Critère frontdoor) *Un ensemble de variables* $\mathbf{Z}$ *satisfait le critère frondoor relatif à une paire ordonnée de variables* (X, Y) *dans un DAG* G *si :*

— $\mathbf{Z}$ *bloque tous les chemins dirigés de* X *vers* Y

— *il n'y a pas de chemin backdoor de* X *vers* $\mathbf{Z}$

— *tous les chemins backdoor de* $\mathbf{Z}$ *vers* Y *sont bloqués par* X

Théorème 1.3.5 (PEARL 1995) *Si un ensemble de variables* $\mathbf{Z}$ *satisfait le critère frontdoor relatif à* (X,Y) *et si* $P(X,\mathbf{Z})>0$*, alors l'effet causal de* X *sur* Y *est identifiable et donné par la formule :*

$$P(Y|do(X)) = \sum_{\mathbf{Z}} P(\mathbf{Z}|X) \sum_{X'} P(Y|X',\mathbf{Z})P(X') \tag{1.9}$$

Exemple 1.3.2 *Dans la figure 1.13, on voit qu'il n'est pas possible d'appliquer le critère backdoor car la variable* U *n'est pas observée, avoir cette variable dans l'expression de l'effet causal empêcherait donc sa résolution. Heureusement, on observe que le nœud* Z *bloque le chemin de* X *vers* Y*, qu'il n'y a pas de chemin backdoor de* X *vers* Z *(la chaîne* (Z,Y,U,X) *est bloquée à cause de la v-structure sur* Y*) et le chaîne* (Z,X,U,Y) *est bloquée par* X*. On peut donc appliquer le critère frontdoor. Une façon de comprendre ce critère est de le voir comme l'application successive de deux critères backdoor. Le premier pour calculer l'effet causal de* Z *sur* Y*, et le second pour déterminer l'effet causal de* X *sur* Z*. En décomposant l'effet causal de* X *sur* Y *de la sorte, on arrive à l'exprimer sans opérateur* $do()$ *et sans variable non observée.*

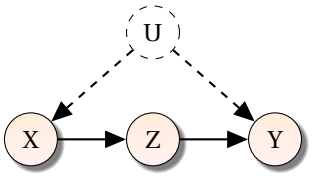


FIGURE 1.13 – Exemple de DAG sur lequel on peut appliquer le critère frontdoor

Les deux critères ci-dessus sont suffisants, mais pas nécessaires. Cela signifie que si on arrive à appliquer ces critères, alors on peut prouver que l'effet causal est identifiable. Mais si on ne parvient pas à les appliquer, alors cela ne prouve pas que l'effet causal n'est pas identifiable et la question reste donc sans réponse.

Néanmoins, on a vu précédemment que les règles du do-calculus (1.3.2) permettaient de manipuler des expressions incluant des opérateurs $do()$. Or l'objectif de la résolution du problème d'identifiabilité est justement de calculer l'effet d'une intervention à partir de données observationnelles, autrement dit, de calculer une probabilité incluant un opérateur $do()$ à partir de grandeurs qui en sont dépourvues. De ce fait, si on arrive, avec une séquence de transformations qui utilisent les règles du do-calculus, à se défaire de l'opérateur $do()$, alors on aura montré que l'effet causal est identifiable. On tient donc ici un critère suffisant. En outre, une preuve que ce critère est nécessaire peut être trouvée dans SHPITSER et PEARL 2006.

Théorème 1.3.6 (SHPITSER et PEARL 2006) *Un effet causal est identifiable dans un modèle semi-markovien* M *associé à un DAG* G *si et seulement s'il existe une séquence*

finie de transformations, chacune étant une application d'une règle du do-calculus du théorème 1.3.2, permettant de réduire l'effet causal à une expression probabiliste standard, c'est-à-dire sans opérateur $do()$ et n'impliquant que des variables observées.

Bien que ce critère soit nécessaire et suffisant, il comporte un défaut non négligeable. En plus d'être purement algébrique, il ne donne pas la méthode pour trouver la séquence de transformations qui permet d'appliquer le critère, ce qui en fait un critère difficilement utilisable.

Il existe pour cela d'autres critères que nous allons voir, sans rentrer dans les détails. D'abord, on peut citer le critère suffisant mais non nécessaire de GALLES et PEARL 1995, qui établit en réalité 4 conditions chacune suffisante pour établir l'identifiabilité d'un effet causal. Les cas décrits par les critères backdoor et frontdoor sont des sous-ensembles des cas décris par ce nouveau critère, qui est donc plus général. Il existe un critère encore plus large, mais toujours suffisant uniquement, proposé par TIAN et PEARL 2002, et qui s'applique sur des effets causaux sur une ou plusieurs variables. Ce critère fournit également l'expression de l'effet causal lorsque celui-ci est identifiable. Enfin, un critère nécessaire et suffisant a été trouvé par SHPITSER et PEARL 2006, qui ferme enfin la question de l'identifiabilité pour un graphe causal totalement défini et donne aussi son expression lorsque l'identifiabilité est prouvée.

1.4 Conclusion

En conclusion, dans ce chapitre, nous avons défini les réseaux bayésiens, qui sont une classe de modèles graphiques probabilistes. Ces réseaux bayésiens sont composés d'un graphe et d'une distribution de probabilité. Les deux éléments sont étroitement liés à travers les indépendances conditionnelles. En effet, la distribution de probabilité possède des indépendances conditionnelles entre ses variables, et celles-ci peuvent être retrouvées graphiquement dans la structure grâce à la notion de d-séparation.

La caractérisation des indépendances des graphes permet de définir une relation d'équivalence entre ceux-ci, et donc une classe d'équivalence, appelée classe d'équivalence de Markov. Tous les graphes équivalents au sens de Markov partagent les mêmes indépendances conditionnelles, et peuvent donc tous représenter une distribution ayant ces mêmes indépendances.

Il est également possible de prendre en compte les relations causales dans le graphe. Dans ce cas, deux graphes différents ne peuvent plus être équivalents, puisqu'ils représentent des relations causales différentes. La causalité permet de mener des raisonnements plus complexes, mais implique de nouvelles difficultés, comme le problème d'identifiabilité.

Cependant, avant d'utiliser un réseau bayésien ou un réseau causal pour représenter un système, il est nécessaire d'identifier la structure de son graphe et les paramètres de sa distribution qui permettront la représentation la plus fidèle possible. Ce processus est l'apprentissage du réseau bayésien (ou du réseau bayésien causal), et c'est le sujet du chapitre suivant.

2 Apprentissage des réseaux bayésiens

Sommaire

2.1 Introduction . . . 44
2.2 Apprentissage des paramètres . . . 45
2.2.1 Définitions et Hypothèses . . . 45
2.2.2 Approche statistique . . . 47
2.2.3 Approche bayésienne . . . 49
2.3 Apprentissage de la structure . . . 50
2.3.1 Hypothèses . . . 51
2.3.2 Tests d'indépendance . . . 52
2.3.3 Algorithmes . . . 54
2.3.3.1 Algorithmes sous contraintes . . . 54
2.3.3.2 Algorithmes à base de score . . . 62
2.3.3.3 Algorithmes hybdrides . . . 70
2.3.4 Apprentissage avec des connaissances . . . 76
2.3.4.1 Utilisation de connaissances certaines . . . 76
2.3.4.2 Utilisation de connaissances incertaines . . . 78
2.3.4.3 Algorithme PCDC . . . 79
2.3.4.4 Algorithme GSA . . . 81
2.3.5 L'apprentissage de réseaux bayésiens causaux . . . 87
2.3.5.1 L'algorithme PC et l'inférence causale . . . 87
2.3.5.2 Les algorithmes d'inférence causale . . . 89
2.4 Conclusion . . . 91

2.1 Introduction

Dans le chapitre précédent, nous avons abordé les réseaux bayésiens et leur interprétation. Ils représentent une distribution de probabilité, ses indépendances et ses paramètres. Lorsque le système est complexe, la distribution associée peut posséder de très nombreuses variables, ce qui motive encore plus le recours aux réseaux bayésiens. Mais pour cela, une étape de construction du réseau (manuellement et/ou partir des données) s'avère cruciale.

La construction du réseau se déroule en deux étapes. Il est d'abord nécessaire de choisir le DAG du réseau bayésien. Comme évoqué précédemment, ce DAG doit

représenter fidèlement les indépendances qui se trouvent dans la distribution de probabilité. Il faut donc choisir avec soin les arcs. Une fois que la structure est déterminée, on obtient alors les relations de parentalité de chaque nœud. Cela nous permet de passer à la deuxième étape, la création des CPD. Ces CPD représentent les probabilités conditionnelles des variables sachant leurs parents, d'où la nécessité d'avoir la structure au préalable. Encore une fois, ces probabilités doivent être choisies avec soin, car elles définissent la distribution jointe qui sera contenue dans le réseau bayésien, d'après la formule 1.2.

Historiquement, la structure comme les paramètres des premiers réseaux bayésiens étaient renseignés manuellement. Les chercheurs faisaient appel à des experts du système qu'ils souhaitaient modéliser pour obtenir les informations requises. Néanmoins, lorsque que le nombre de variables est élevé, la taille du graphe et le nombre de distributions possibles sont trop importants, et la tâche devient fastidieuse. Pour ces raisons, la recherche scientifique s'est vite orientée vers l'automatisation de la construction des réseaux bayésiens, en s'appuyant sur des bases de données. C'est ce qu'on appelle **l'apprentissage automatique du réseau bayésien**.

Ce chapitre est dédié à cette problématique. Nous verrons tout d'abord l'apprentissage des paramètres, puis l'apprentissage de la structure. Préalablement à chaque sujet, les hypothèses utilisées seront présentées.

2.2 Apprentissage des paramètres

2.2.1 Définitions et Hypothèses

Cette première partie est dédiée à l'apprentissage des paramètres du réseau bayésien. Ces paramètres correspondent aux tables de probabilité conditionnelle associées à chaque variable du réseau : ce sont les probabilités de la variable sachant ses parents. Pour apprendre ces paramètres, nous allons utiliser une base de données qui contiennent une série d'observations. Ces dernières permettent de calculer les fréquences d'apparition des phénomènes qu'on cherche à modéliser, et diverses méthodes permettent d'en extraire des probabilités qui rempliront les CPD.

Mais avant de découvrir ces méthodes, il est nécessaire d'introduire quelques notations. Soit un réseau bayésien $B = (G, \Theta)$ avec $G = (\mathbf{V}, \mathbf{E})$ le graphe et Θ l'ensemble des distributions de probabilité conditionnelle. Pour chaque variable $X_i \in \mathbf{V}, i \in \{1, \ldots, n\}$, on note r_i et q_i respectivement le nombre d'états de la variable X_i et le nombre d'états des parents $Pa(X_i)$ de X_i. Par exemple, si X_3 est une variable binaire qui possède deux parents X_1 et X_2 binaires également, alors on a $r_3 = 2$ et $q_3 = r_1 \times r_2 = 2 \times 2 = 4$. On note $x_{i,k}$ la $k^{ième}$ valeur de la variable X_i, et de la même façon $pa(X_i)_j$ la $j^{ième}$ valeur que peut prendre l'ensemble de variables $Pa(X_i)$.

Dans une base de données, un ensemble d'observations $\mathbf{S}$ de taille N décrit les états observés des variables du système. Chaque observation est représentée par une ligne dans la base de données. Pour la $l^{ième}$ ligne, on observe l'état $x_i^{(l)}$ de la variable X_i. On utilise $N_{i,j,k}$ pour parler du nombre d'observations dans $\mathbf{S}$ pour lesquelles

on a $X_i = x_{i,k}$ et $Pa(X_i) = pa(X_i)_j$. Grâce à cette base de données, on va chercher à déterminer les paramètres Θ. Plus précisément, on note :

— θ_i l'ensemble des paramètres de la variable X_i, selon la distribution de probabilité $P(X_i|Pa(X_i))$

— $\theta_{i,j}$ l'ensemble des paramètres de la variable X_i sachant que ses parents ont pris les valeurs $pa(X_i)_j$, donc selon la probabilité $P(X_i|Pa(X_i) = pa(X_i)_j)$

— $\theta_{i,j,k}$ la probabilité que la variable X_i prenne la valeur $x_{i,k}$ sachant que ses parents $Pa(X_i)$ ont pris la valeur $pa(X_i)_j$, et donc $\theta_{i,j,k} = P(X_i = x_{i,k}|Pa(X_i) = pa(X_i)_j))$

De plus, relevons le fait qu'une base de données finie n'est qu'une représentation partielle de la réalité, et ne peut donc nous fournir qu'une approximation de Θ, notée $\hat{\Theta}$.

Il nous faut également définir quelques termes de statistiques bayésiennes. D'abord, une **loi *a priori*** est la loi de probabilité d'une grandeur avant toute étude des observations de la base de données. Elle ne repose donc pas sur les observations $\mathbf{S}$, mais sur nos connaissances ou sur des observations antérieures $\mathbf{S}'$. On note généralement cette loi par $\pi(\theta)$ pour un paramètre θ. L'objectif des statistiques bayésiennes est de combiner une loi *a priori* avec l'information contenue dans les observations $\mathbf{S}$ pour déterminer une **loi *a posteriori***, notée $\pi(\theta|\mathbf{S})$, en utilisant le théorème de Bayes (théorème 1.2.3) :

$$\pi(\theta|\mathbf{S}) = \frac{P(\mathbf{S}|\theta)\pi(\theta)}{\int_\theta P(\mathbf{S}|\theta)\pi(\theta)d\theta} \tag{2.1}$$

La grandeur $P(\mathbf{S}|\theta)$ est appelée **vraisemblance** et se note souvent $\mathscr{L}(\mathbf{S}|\theta)$. Elle représente la "probabilité" d'observer les échantillons de $\mathbf{S}$ sachant le paramètre θ. Notons également que pour simplifier les calculs, c'est souvent le logarithme de la vraisemblance, la **log-vraisemblance**, qui est utilisé et qui se note $\mathscr{LL}(\mathbf{S}|\Theta)$.

Pour utiliser les méthodes décrites ci-dessous, il nous faut également définir un ensemble d'hypothèses permettant de mener à bien les calculs :

— Les données sont **discrètes**

— **Indépendance des échantillons** : nous supposerons que toutes les observations répertoriées dans la base de données sont **indépendantes et identiquement distribuées** (I.I.D). En d'autres termes, les observations ne sont pas dépendantes les unes des autres et les lois de probabilités qui régissent le système ne changent pas entre les observations. Cette hypothèse permet notamment de déterminer la vraisemblance (et donc la log-vraisemblance) grâce à la formule suivante :

$$\mathscr{L}(\mathbf{S}|\theta) = \prod_{l=1}^{N} P_\theta(\mathbf{x}^{(l)}) \tag{2.2}$$

— **Indépendance des paramètres** : nous supposerons que les paramètres eux-mêmes sont indépendants les uns des autres. Cette indépendance peut être observée à deux échelles. L'indépendance globale des paramètres indique que les paramètres $\{\theta_i\}_{i=1,\dots,n}$ sont indépendants les uns des autres, ce qui permet

d'écrire :

$$P(\Theta) = \prod_{i=1}^{n} P(\theta_i) \tag{2.3}$$

De même, d'un point de vue local, cette hypothèse stipule que les paramètres $\{\theta_{i,j}\}_{j=1,\dots,q_i}$ d'une variable X_i sont indépendants les uns des autres, ce qui permet d'écrire :

$$P(\theta_i) = \prod_{j=1}^{q_i} P(\theta_{i,j}) \tag{2.4}$$

— **Modularité des paramètres** : nous supposerons que les paramètres d'une variable X_i ne dépendent que de ses parents et pas des autres parties du graphe. En d'autres termes, pour deux graphes G et G' différents tels que $Pa_G(X_i) = Pa_{G'}(X_i)$, on a $P(\theta_i|G) = P(\theta_i|G')$.
— **Hypothèse de Dirichlet** : nous supposerons que les lois *a priori* des paramètres $\theta_{i,j}$ suivent une distribution de Dirichlet.

2.2.2 Approche statistique

Débutons avec une première méthode issue de l'approche statistique. La méthode par **maximum de vraisemblance**, noté **MV**, vise comme son nom l'indique à trouver les paramètres Θ qui maximisent la vraisemblance :

$$\hat{\Theta}^{MV} = \underset{\Theta}{\operatorname{argmax}}\, \mathscr{L}(\mathbf{S}|\Theta)$$

Grâce à cette méthode, les paramètres estimés sont obtenus par la formule suivante :

$$\hat{\theta}^{MV}_{i,j,k} = \frac{N_{i,j,k}}{\sum_{k'=1}^{r_i} N_{i,j,k'}} \tag{2.5}$$

Cette méthode est en réalité assez intuitive. Lorsqu'on observe que sur 100 lancés de pièce, on a 50 "pile", on en déduit intuitivement que la probabilité de pile est $\frac{1}{2}$. Mais allons plus loin que l'intuition, et démontrons ce résultat.

Démonstration

Soit $\mathbf{x}^{(l)} = \{x_1^l, x_2^l, \dots, x_n^l\}$ la $l^{ième}$ observation de notre base de données. La vraisemblance de cette observation conditionnellement aux paramètres Θ est :

$$\begin{aligned} P(\mathbf{X} = \mathbf{x}^{(l)}|\Theta) &= P(X_1 = x_1^{(l)}, X_2 = x_2^{(l)}, \dots, X_n = x_n^{(l)}|\Theta) \\ &= \prod_{i=1}^{n} P(X_i = x_i^{(l)}|Pa(X_i) = pa(X_i)^{(l)}, \Theta) \\ &= \prod_{i=1}^{n} \theta_{i,j(l),k(l)} \end{aligned}$$

Ici, $\theta_{i,j(l),k(l)}$ désigne le paramètre lorsque X_i prend la valeur $x_i^{(l)}$ observée à la $l^{ème}$ ligne et $Pa(X_i)$ prend les valeurs $pa(X_i)^{(l)}$ observées à cette même ligne. Pour

l'ensemble des données, de la même manière, on a :

$$\mathscr{L}(\mathbf{S}|\theta) = \prod_{l=1}^{N} P(\mathbf{X} = \mathbf{x}^{(l)}|\Theta) = \prod_{i=1}^{n}\prod_{l=1}^{N} \theta_{i,j(l),k(l)} \tag{2.6}$$

En parcourant toute la base de données, pour i, j, k fixés, le terme $\theta_{i,j,k}$ apparaît par définition $N_{i,j,k}$ fois. Donc on peut réécrire l'équation 2.6 de la manière suivante :

$$\mathscr{L}(\mathbf{S}|\theta) = \prod_{i=1}^{n}\prod_{j=1}^{q_i}\prod_{k=1}^{r_i} \theta_{i,j,k}^{N_{i,j,k}} \tag{2.7}$$

Ainsi, on peut réécrire la log-vraisemblance :

$$\mathscr{LL}(\mathbf{S}|\theta) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{i,j,k} log(\theta_{i,j,k}) \tag{2.8}$$

Les paramètres $\theta_{i,j,k}$ sont des probabilités, et donc somment à 1. On a donc également :

$$\sum_{k=1}^{r_i} \theta_{i,j,k} = 1 \Longleftrightarrow \theta_{i,j,r_i} = 1 - \sum_{k=1}^{r_i-1} \theta_{i,j,k}$$

Cela nous permet de réécrire une nouvelle fois la log-vraisemblance :

$$\mathscr{LL}(\mathbf{S}|\Theta) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\left(\sum_{k=1}^{r_i-1} N_{i,j,k} log(\theta_{i,j,k}) + N_{i,j,r_i} log\left(1 - \sum_{k=1}^{r_i-1}\theta_{i,j,k}\right)\right)$$

Rappelons que nous cherchons la valeur de $\theta_{i,j,k}$ qui maximise la vraisemblance (ou de manière équivalente, la log-vraisemblance). Un manière de trouver cette valeur consiste à calculer la dérivée de la fonction à maximiser et de trouver la valeur qui annule la dérivée. La dérivée de la log-vraisemblance par rapport à un paramètre $\theta_{i,j,k}$ est :

$$\frac{\partial \mathscr{LL}(\mathbf{S}|\Theta)}{\partial \theta_{i,j,k}} = \frac{N_{i,j,k}}{\theta_{i,j,k}} - \frac{N_{i,j,r_i}}{1 - \sum_{k'=1}^{r_1-1}\theta_{i,j,k'}} = \frac{N_{i,j,k}}{\theta_{i,j,k}} - \frac{N_{i,j,r_i}}{\theta_{i,j,r_i}}$$

Ainsi, pour tout $k \in \{1, \ldots, r_1 - 1\}$, la valeur $\theta_{i,j,k}$ qui maximise la vraisemblance vérifie :

$$\frac{N_{i,j,k}}{\theta_{i,j,k}} - \frac{N_{i,j,r_i}}{\theta_{i,j,r_i}} = 0 \Longleftrightarrow \frac{N_{i,j,k}}{\theta_{i,j,k}} = \frac{N_{i,j,r_i}}{\theta_{i,j,r_i}} = \frac{\sum_{k'=1}^{r_i} N_{i,j,k'}}{\sum_{k'=1}^{r_i}\theta_{i,j,k'}} = \sum_{k'=1}^{r_1} N_{i,j,k'}$$

On a donc, pour tout $k \in \{1, \ldots, r_i\}$:

$$\hat{\theta}_{i,j,k} = \frac{N_{i,j,k}}{\sum_{k'=1}^{r_i} N_{i,j,k'}} \tag{2.9}$$

■

Cette méthode est simple, puisqu'elle consiste à compter le nombre d'occurrences d'un événement dans la base de données. Mais en contrepartie, elle nécessite une base de données de taille suffisante pour retranscrire correctement les distributions de probabilité. Pour imager ce point, si on reprend l'exemple des lancés de pièce, on voit bien qu'avec 5 lancés, nous ne sommes pas certains d'avoir la bonne distribution. Cela est encore plus explicite avec un lancé de dés, qui permet 6 valeurs possibles, et qui nécessite plus d'observations. En poussant ce raisonnement plus loin, on voit également apparaître un problème. Certains événements pourraient ne pas être observés du tout, sans qu'ils soient pour autant impossibles. Cette méthode conduirait à une probabilité nulle en cas d'absence de la combinaison d'un ou plusieurs états des variables en question au sein de la base données. Cela pose de nombreuses difficultés au moment de l'utilisation des réseaux bayésiens pour réaliser de l'inférence probabiliste.

Pour remédier à ce problème, il est possible d'inclure des pseudo-comptes sur tous les événements pour assurer une présence "virtuelle" a minima. Pour cela, nous allons utiliser un *a priori* de Dirichlet qui assurera une probabilité non nulle à l'événement. L'utilisation d'un *a priori* est au cœur de l'approche bayésienne que nous allons voir dans la sous-section suivante.

2.2.3 Approche bayésienne

L'approche bayésienne vise à prendre en compte une distribution *a priori* issue de connaissances ou de bases de données antérieures pour calculer une distribution *a posteriori*. Cette dernière est comme "mise à jour" grâce à l'*a priori*, et se veut donc plus fidèle à la réalité. La première méthode de cette approche est le **maximum *a posteriori***. L'objectif n'est plus de calculer les paramètres qui maximisent la vraisemblance des données, mais de calculer ceux qui maximisent la probabilité *a posteriori*.

$$\hat{\Theta}^{MAP} = \underset{\Theta}{\operatorname{argmax}}\, \pi(\Theta|\mathbf{S})$$

La probabilité *a posteriori* $\pi(\Theta|\mathbf{S})$ est obtenu grâce au théorème de Bayes 2.1. Notons que dans cette formule, le dénominateur n'est pas fonction de Θ. On peut donc réécrire l'équation 2.1 de la façon suivante :

$$\pi(\Theta|\mathbf{S}) \propto \mathscr{L}(\mathbf{S}|\Theta)\pi(\Theta)$$

où le symbole $\propto$ signifie "proportionnel".

Une des difficulté de l'approche bayésienne est le calcul de la loi *a posteriori*. Or une solution pour faciliter ce calcul est l'utilisation de **lois conjuguées**.

Définition 2.2.1 (Loi conjuguée) *Une famille F de lois sur Θ est dite conjuguée si, pour toute distribution π appartenant à cette famille, la loi $\pi(\Theta|\mathbf{S})$ appartient également à cette famille.*

Dans notre cas, nous savons grâce à l'équation 2.7 que la vraisemblance suit une loi multinomiale. Or dans ce cas, une loi *a priori* conjuguée est la loi de **Dirichlet**. En d'autres termes, si la loi des observations sachant les paramètres suit une loi multinomiale $M(\Theta)$, alors l'utilisation d'une loi de Dirichlet $D(\alpha)$ comme *a priori* nous permet d'affirmer que la loi *a posteriori* sera également une loi de Dirichlet.

Ainsi, on a :

$$\pi(\Theta) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \left(\theta_{i,j,k}\right)^{\alpha_{i,j,k}-1}$$

Ce qui nous donne une loi *a posteriori*, d'après ROBERT 1994 :

$$\pi(\Theta|\mathbf{S}) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \left(\theta_{i,j,k}\right)^{N_{i,j,k}+\alpha_{i,j,k}-1}$$

Maintenant, en posant $N'_{i,j,k} = N_{i,j,k} + \alpha_{i,j,k} - 1$, on retrouve la même équation que 2.7. On peut alors faire le même raisonnement, ce qui nous donne l'estimation suivante :

$$\hat{\theta}^{MAP}_{i,j,k} = \frac{N_{i,j,k} + \alpha_{i,j,k} - 1}{\sum_{k'=1}^{r_i} N_{i,j,k'} + \alpha_{i,j,k'} - 1} \tag{2.10}$$

Il existe une autre méthode issue de l'approche bayésienne. La méthode de l'**espérance *a posteriori*** propose de s'intéresser à l'espérance de la loi a posteriori. Cette méthode nous donne l'estimation suivante, d'après ROBERT 1994 :

$$\hat{\theta}^{EAP}_{i,j,k} = \frac{N_{i,j,k} + \alpha_{i,j,k}}{\sum_{k'=1}^{r_i} N_{i,j,k'} + \alpha_{i,j,k'}} \tag{2.11}$$

2.3 Apprentissage de la structure

Dans la section précédente, nous nous sommes intéressés à l'apprentissage automatique des paramètres du réseau bayésien à partir de données complètes. Les paramètres sont les probabilités d'une variable sachant ses parents, il est donc nécessaire de connaître les parents de chaque variable et donc de connaître la structure, avant de réaliser l'apprentissage de ces paramètres. C'est le sujet de cette section.

L'apprentissage automatique de structure a pour but de découvrir, généralement grâce aux données, et parfois des connaissances d'experts, le DAG du réseau bayésien qui permet de modéliser le système étudié. Pour cela, on part d'un ensemble de variables, sur lesquelles on a des connaissances, et on utilise un algorithme qui exploite ces données pour aboutir à la structure du réseau bayésien. Les connaissances peuvent quant à elles prendre de nombreuses formes. Elles peuvent aider au choix des variables à introduire dans le réseau bayésien ou à l'apprentissage lui-même.

Ce problème est une tâche difficile. D'abord, il s'agit d'un problème classé *NP-difficile* (CHICKERING 1996), ce qui signifie qu'il n'existe pas de méthode connue pour résoudre ce problème de façon exacte dans des temps polynomiaux. C'est pourquoi l'apprentissage de structure est rapidement devenu un champ de recherche actif, avec

de nombreuses publications scientifiques chaque année. Devant la difficulté d'utiliser des méthodes exactes, des méthodes approchées sont bien souvent utilisées, avec l'inconvénient de ne pas être assuré de trouver la solution la plus optimale. Enfin, il est fréquemment nécessaire d'avoir de larges bases de données pour envisager un apprentissage efficace.

Les méthodes d'apprentissage automatique de la structure sont généralement divisées en 3 catégories :

— **les méthodes sous contraintes** (*constraint-based*) : ces méthodes exploitent des tests d'indépendances pour construire le modèle d'indépendance du réseau bayésien et ainsi définir le graphe associé à ce modèle
— **les méthodes à base de score** (*score-based)* : ces méthodes utilisent un score dans le but d'explorer un espace de recherche, pouvant être constitué de DAG, de CPDAG ou d'autres structures. Elles évaluent chaque élément rencontré pour se diriger dans l'espace et trouver l'élément qui correspond le plus aux données.
— **les méthodes hybrides** : ces méthodes combinent des éléments des deux approches précédentes pour exploiter leurs avantages respectifs et pallier les défauts de chacune. Elles sont généralement constituées d'étapes issues des méthodes sous contraintes ou à base de score.

Dans cette partie sur l'apprentissage de la structure, nous commencerons avec les algorithmes qui n'utilisent que des données, sans connaissances d'experts pour les aider. Dans cette catégorie, nous verrons successivement les algorithmes sous contraintes, à base de score, et hybrides. Nous verrons ensuite des algorithmes qui permettent l'utilisation de connaissances d'experts. Enfin, nous nous intéresserons à la fin de cette partie à l'apprentissage causal, qui diffère un peu de l'apprentissage de réseaux bayésiens classiques.

Comme pour l'apprentissage des paramètres, avant de découvrir ces méthodes, nous devons faire quelques hypothèses.

2.3.1 Hypothèses

Les hypothèses suivantes permettent de définir un cadre dans lequel les méthodes proposées sont valides. Mais parmi les méthodes décrites dans cette section, nous verrons que certaines d'entre elles permettent de relâcher des hypothèses et ainsi rendre l'utilisation de ces méthodes plus générale, mais en donnant un résultat parfois de nature différente.

La première hypothèse est l'hypothèse de **fidélité** (*faithfulness*). Celle-ci nous assure que la distribution sur laquelle nous travaillons admet l'existence d'une P-map. Plus précisément, on sait qu'il existe une I-map pour cette distribution. Cette hypothèse nous assure qu'il existe un graphe tel que toutes les indépendances de la distribution sont décrites par la propriété locale de Markov (1.2.4). Ce graphe est donc à la fois une I-map et une D-map, et par conséquent une P-map. Cette hypothèse nous permet de fixer un objectif concret et d'être sûr que cet objectif est atteignable.

La deuxième hypothèse est l'hypothèse de **suffisance causale** (*causal sufficiency*), aussi appelée **complétude** (*completeness*). On dit qu'un ensemble de variables **V** est

suffisant causalement pour un ensemble d'observation **S** si et seulement si pour chaque cause Y commune à plusieurs variables de **V**, Y appartient aussi à **V** ou alors Y est constant pour toute la base de données. Cette hypothèse nous permet d'être sûr qu'il n'existe pas de variable cachée qui rendrait dépendantes deux variables sans que cette dépendance puisse être causalement expliquée.

Enfin, nous supposerons également que les données sont discrètes, de la même manière que pour l'apprentissage des paramètres.

2.3.2 Tests d'indépendance

Les algorithmes sous contraintes utilisent les indépendances de la distribution de probabilité pour contraindre la structure et ainsi construire un graphe qui exprime les mêmes indépendances que la distribution. Dans la pratique, ces indépendances sont inconnues, et doivent par conséquent être déterminées grâce à la base de données.

En effet, lorsqu'une base de données est distribuée selon une distribution de probabilité et que la base de données est assez grande, il est possible de retrouver les indépendances entre les variables directement dans les données. Pour cela, on utilise des tests d'indépendance. Notons X et Y deux variables dont on veut déterminer si elles sont dépendantes marginalement ou non. Les tests d'indépendances mettent en confrontation deux hypothèses, notées H_0 et H_1, respectivement appelées hypothèse nulle et contre-hypothèse :

— $\mathbf{H_0}$: X et Y sont marginalement indépendants
— $\mathbf{H_1}$: X et Y ne sont pas marginalement indépendants

Le principe général des tests d'indépendances est de confronter les deux hypothèses et de savoir laquelle est la plus plausible au regard des données. Pour cela, il est nécessaire de spécifier une valeur, α, appelé risque de première espèce, et qui correspond à la probabilité de rejeter H_0 sahant que H_0 est vraie. Ce risque correspond donc à la probabilité de rejeter l'hypothèse d'indépendance sachant que les variables sont véritablement indépendantes.

Pour étudier les données et prendre notre décision, nous allons considérer deux modèles. Notons N_x et N_y le nombre d'occurrences des observations $X = x$ et $Y = y$ et N_{xy} le nombre d'occurences de $(X = x, Y = y)$ dans les données. Enfin, notons N la taille de la base de données. Ainsi, on construit d'une part le modèle observé $(M)_o$ et le modèle théorique en cas d'indépendance $(M)_t$. Pour chaque modèle, on peut associer des probabilités à l'évènement $(X = x, Y = y)$. Pour $(M)_o$, on observe avec les données que :

$$P(X = x, Y = y) = \theta_{xy} = \frac{N_{xy}}{N}$$

Dans le modèle théorique en cas d'indépendance $(M)_t$, on a :

$$P(X = x, Y = y) = P(X = x) \times P(Y = y) = \theta_x \times \theta_y = \frac{N_x}{N} \times \frac{N_y}{N}$$

Donc si il y a effectivement une indépendance, on a l'égalité :

$$\frac{N_{xy}}{N} = \frac{N_x}{N} \times \frac{N_y}{N}$$

qu'on peut réécrire :

$$N_{xy} = \frac{N_x \times N_y}{N}$$

Un des tests les plus connus est le test du χ^2 (SAPORTA 1990). Il demande de calculer la statistique :

$$d^2 = \sum_{x \in dom(X)} \sum_{y \in dom(Y)} \frac{\left(N_{xy} - \frac{N_x \times N_y}{N}\right)^2}{\frac{N_x \times N_y}{N}} \tag{2.12}$$

En cas d'indépendance d^2 suit une loi du χ^2 à $(|dom(X)|-1) \times (|dom(Y)|-1)$ degrés de liberté. Pour utiliser ce test, il faut donc fixer le risque de première espèce α puis déterminer la **p-value** de la statistique d^2 calculée en utilisant la formule 2.12 avec les données observées. La p-value représente la probabilité que la statistique soit au moins aussi grande sachant l'indépendance. Donc si p-value<α, alors la probabilité d'obtenir cette statistique sachant l'indépendance est trop faible, ce qui permet de rejeter l'hypothèse de l'indépendance et en conclure que les variables sont dépendantes. A l'inverse, si p-value>α, on estimera par simplification que les variables sont indépendantes.

Ce test peut être étendu à l'indépendance conditionnelle. Si on veut tester l'indépendance entre deux variables X et Y sachant une ensemble de variables $\mathbf{Z}$, alors il faudra calculer la statistique d^2 avec la formule 2.13. En cas d'indépendance, la statistique suit cette fois une loi du χ^2 à $(|dom(X)|-1) \times (|dom(Y)|-1) \times \prod_{Z \in \mathbf{Z}} |dom(Z)|$ degrés de liberté.

$$d^2 = \sum_{x \in dom(X)} \sum_{y \in dom(Y)} \sum_{\mathbf{z} \in dom(\mathbf{Z})} \frac{\left(N_{xy\mathbf{z}} - \frac{N_{x\mathbf{z}} \times N_{y\mathbf{z}}}{N_{\mathbf{z}}}\right)^2}{\frac{N_{x\mathbf{z}} \times N_{y\mathbf{z}}}{N_{\mathbf{z}}}} \tag{2.13}$$

Enfin, un autre test a été mis au point pour obtenir des résultats plus robustes (SAPORTA 1990), le test G^2. Celui-ci suit exactement le même principe que le test χ^2, mais la statistique est cette fois-ci dérivée du rapport entre la vraisemblance du modèle observé et celle du modèle théorique sachant l'indépendance.

$$g^2 = 2ln\left(\frac{\prod_{xy} \theta_{xy}^{N_{xy}}}{\prod_{xy} \left(\theta_{x\cdot} \times \theta_{\cdot y}\right)^{N_{xy}}}\right) \tag{2.14}$$

où $\theta_{x\cdot} = \sum_y \theta_{xy}$ et $\theta_{\cdot y} = \sum_x \theta_{xy}$. Ici, les paramètres θ_{xy} peuvent être évalués par exemple par maximum de vraisemblance.

En cas d'indépendance, la statistique g^2 suit une loi du χ^2 à $(|dom(X)|-1) \times (|dom(Y)|-1)$. Ce test s'utilise donc comme le test χ^2, et peut également être étendu au cas conditionnel. Notons de plus que cette statistique peut être calculée d'une

autre façon, avec l'équation 2.15, les deux formules étant strictement égale.

$$g^2 = 2 \sum_{x \in dom(X)} \sum_{y \in dom(Y)} N_{xy} ln \left(\frac{N_{xy}}{\frac{N_x \times N_y}{N}} \right) \tag{2.15}$$

2.3.3 Algorithmes

2.3.3.1 Algorithmes sous contraintes

Commençons par les algorithmes sous contraintes. Ces algorithmes vont chercher à construire le modèle d'indépendance de la distribution qui a généré les données. Du fait de l'hypothèse de fidélité, nous savons que nous recherchons une P-map, ce qui signifie que toutes les indépendances trouvées dans la base de données se traduisent par une relation de d-séparation dans le graphe. Nous allons donc employer ces indépendances comme des contraintes que doit respecter le graphe pour aboutir au résultat recherché. Cela nous permet de sélectionner les arêtes qui doivent être dans le graphe, et ainsi d'obtenir le squelette.

Du fait de leur spécificité en termes d'indépendance, les v-structures ont une place importante dans l'orientation des arêtes. Grâce à l'hypothèse de suffisance causale, on sait qu'une indépendance marginale entre deux variables séparées par une troisième variable ne peut signifier qu'une v-structure. Celles-ci pourront alors être identifiées.

D'après ce qui précède, il est possible d'obtenir le squelette et les v-structures. Ces éléments sont suffisants pour obtenir le CPDAG du graphe recherché. Et c'est là une spécificité importante des algorithmes sous contraintes : ils ne permettent pas de distinguer les DAG au sein d'une même classe d'équivalence. En d'autres termes, le résultat de ces algorithmes est un CPDAG, représentant d'une classe d'équivalence qui retranscrit toutes les indépendances trouvées dans les données.

Il est aussi important de noter qu'un graphe orienté seulement avec des v-structures n'en fait pas toujours un CPDAG. Il faut orienter toutes les arêtes invariables dans la classe d'équivalence de Markov. Heureusement, les arêtes invariables qui n'appartiennent pas à des v-structures sont facilement identifiables en partant de ces v-structures. Un ensemble de quatre règles permettent de les identifier, appelées règles de Meek. Tout d'abord, voyons ces règles.

Les règles de Meek

Les règles d'orientation ont d'abord été identifiées par VERMA et PEARL 1992. Puis MEEK 1995 a montré que ces règles étaient nécessaires et suffisantes, ce qui signifie qu'elles permettent d'identifier toutes les arêtes invariables en plus des v-structures. On compte quatre règles :

— R_1 : Si il y a deux arêtes $X \rightarrow Y$ et $Y - Z$ et que X et Z sont non adjacents, alors on peut orienter $Y - Z$ en $Y \rightarrow Z$

— R_2 : Si il y a un chemin $X \rightarrow Y \rightarrow Z$ et une arête $X - Z$, alors on peut orienter $X - Z$ en $X \rightarrow Z$

— R_3 : Si on a $X - Z \rightarrow Y$ et $X - T \rightarrow Y$, une arête $X - Y$ et que Z et T sont non adjacents, alors on peut orienter $X - Y$ en $X \rightarrow Y$
— R_4 : Si on a $X - Z \rightarrow T$ et $X - T \rightarrow Y$, une arête $X - Y$ et que Z et Y sont non adjacents, alors on peut orienter $X - Y$ en $X \rightarrow Y$

Notons que lorsqu'on part d'un graphe partiellement orienté avec des v-structures uniquement, la règle R_4 n'est plus nécessaire.

Maintenant que nous avons tous les éléments pour découvrir les algorithmes sous contraintes, passons au premier exemple, l'algorithme **IC**.

Algorithme IC (VERMA et PEARL 1990)

Datant de 1991, l'algorithme IC est une sorte de prototype des algorithmes sous contraintes, laissant des libertés dans la manière de faire les tests d'indépendance. Il est constitué de 3 étapes, retranscrites dans l'algorithme 1.

Algorithm 1 Algorithme IC

Entrée : un ensemble d'observation **S**, un ensemble de variables **V**
Sortie : un graphe partiellement orienté G

1: $G \leftarrow (\emptyset, \mathbf{V})$
2: **Pour tout** $(X, Y) \in \mathbf{V}^2$ **Faire**
3: chercher un ensemble de séparation $Sepset_{XY} \subset \mathbf{V}$ tel que $X \perp Y | Sepset_{XY}$
4: **Fin Pour**
5: **Pour tout** $(X, Y) \in \mathbf{V}^2$ telle qu'aucun $Sepset_{XY}$ n'a pu être trouvé **Faire**
6: Ajouter une arête à G entre les nœuds X et Y
7: **Fin Pour**
8: **Pour tout** $(X, Y) \in \mathbf{V}^2$ non adjacent avec un voisin Z en commun **Faire**
9: **Si** $Z \notin Sepset_{XY}$ **Alors**
10: Orienter $X \rightarrow Z \leftarrow Y$
11: **Fin Si**
12: **Fin Pour**
13: Orienter autant d'arêtes que possible par application des règles de Meek.
14: **Retourner** G

La première étape, de la ligne 2 à 7, consiste à rechercher les indépendances de la distribution. Cette étape part d'un graphe vide, donc sans aucune arête, et ajoute les arêtes lorsqu'aucune indépendance n'est trouvée, ni marginale ni conditionnelle. Pour toutes les paires de variables pour lesquelles une indépendance a été trouvée, l'ensemble de séparation, c'est-à-dire l'ensemble de conditionnement qui a été utilisé pour trouver l'indépendance, est sauvegardé.

La deuxième étape, ligne 8 à 12, oriente les v-structures. Une v-structure indique une dépendance conditionnelle. Le nœud situé au milieu de la structure ne doit donc pas appartenir à l'ensemble de séparation des deux autres nœuds. C'est pour cette raison que les ensembles de séparation sont sauvegardés. En parcourant toutes les

structures candidates, on peut trouver les v-structures. À la fin de cette étape, on a donc un graphe partiellement orienté, où les seuls arcs sont ceux des v-structures.

Enfin, la dernière étape, consiste à trouver tous les arcs invariables, grâce aux règles de Meek. On obtient donc un CPDAG, et une classe d'équivalence associée à ce CPDAG. Pour obtenir un DAG, il suffit d'orienter toute les arêtes qui ne sont pas encore orientées. La seule contrainte est de ne pas créer de nouvelle v-structure ou de circuit.

Comme on peut le voir, cet algorithme ne donne pas de méthode systématique pour trouver les indépendances. Ainsi, si le nombre de variables est trop grand, une recherche naïve peut s'avérer longue et fastidieuse. C'est pourquoi les auteurs de SPIRTES et C. GLYMOUR 1991 ont mis au point une méthode pour trouver ces indépendances, ce qui a donné l'algorithme **PC**.

Algorithme PC (SPIRTES et C. GLYMOUR 1991)

L'algorithme PC reprend les trois étapes, mais donne une méthode efficace pour parcourir les paires de nœuds et trouver les indépendances lors de la première étape. Cette première étape est retranscrite dans l'algorithme 2.

Algorithm 2 Algorithme PC

Entrée : un ensemble d'observations **S**, un ensemble de variables **V**
Sortie : un graphe non orienté G

$G \leftarrow (\mathbf{V}, \mathbf{E})$
Sepset$\leftarrow \emptyset$
$m \leftarrow -1$
Répéter
  $m \leftarrow m+1$
  Répéter
    Sélectionner une nouvelle paire ordonnée (X,Y) adjacente telle que $|adj(X) \setminus Y| \geq m$
    Répéter
      Sélectionner un nouvel ensemble $\mathbf{Z} \subseteq adj(X) \setminus Y$ tel que $|\mathbf{Z}| = m$
      Si $X \perp Y | \mathbf{Z}$ **Alors**
        Supprimer l'arête entre X et Y dans G
        $Sepset_{XY} \leftarrow Sepset_{XY} \cup \mathbf{Z}$
      Fin Si
    Jusqu'à ce que tous les ensembles $\mathbf{Z} \subseteq adj(X)$ tels que $|\mathbf{Z}| = m$ aient été testés ou qu'une indépendance ait été trouvée
  Jusqu'à ce que toutes les paires ordonnées (X,Y) adjacentes telles que $|adj(X) \setminus Y| \geq m$ aient été testées
Jusqu'à ce que $\forall X \in \mathbf{V}$, $|adj(X)| \leq m$
Retourner G

Contrairement à l'algorithme IC, l'algorithme PC part d'un graphe complet, où tous les nœuds sont adjacents les uns aux autres. Il va ensuite faire les tests d'indépendance dans un ordre précis. Il va commencer par découvrir les indépendances marginales. À chaque indépendance trouvée, l'arête entre les nœuds indépendants est supprimée. Une fois toutes les paires étudiées, il recommence à faire de nouveaux tests d'indépendance entre les nœuds encore adjacents, en prenant cette fois un ensemble de conditionnement de taille 1. L'ensemble de conditionnement est constitué de variables adjacentes à l'une des variables dont l'indépendance est étudiée. Une fois que toutes les indépendances conditionnelles avec un ensemble de conditionnement de taille 1 ont été étudiées, on recommence avec des ensemble de conditionnement de taille 2, puis 3, etc. La taille de conditionnement maximale est l'un des paramètres de l'algorithme.

Les autres étapes de l'algorithme sont les mêmes que IC.

Cet algorithme va donc faire les tests d'indépendance dans un ordre précis, qui dépend de l'ordre dans lequel les variables sont ordonnées en entrée de l'algorithme. Le problème est donc que le résultat est sensible à l'ordre donné en entrée. C'est ce qu'ont voulu corriger COLOMBO et M. H. MAATHUIS 2014, grâce à leur algorithme **PC stable**.

Algorithme PC stable (COLOMBO et M. H. MAATHUIS 2014)

Dans l'article COLOMBO et M. H. MAATHUIS 2014, les auteurs proposent une série de modifications permettant de rendre indépendants le résultat et l'ordre d'entrée des variables. Ces modifications concernent toutes les étapes de PC, mais nous nous intéresserons ici seulement à la modification de la première étape, retranscrite dans l'algorithme 3.

Dans l'algorithme, les ensembles de conditionnement utilisés pour les tests d'indépendance dépendent des adjacences du graphe au moment où le test est effectué. Dans PC stable, les auteurs proposent de sauvegarder les adjacences au début d'un nouveau cycle. Ainsi, pour une même taille d'ensemble de conditionnement, les variables de conditionnement ne dépendent plus des tests qui ont été effectués avant, ce qui permet de rendre le résultat de cette étape stable en fonction de l'ordre d'entrée des variables.

Cet algorithme est resté comme une référence des algorithmes sous contraintes. Mais d'autres modifications ont été apportées au fil du temps, ce qui a donné naissance à d'autres versions de cet algorithme. On peut notamment mentionner RAMSEY, SPIRTES et J. ZHANG 2006 qui ont travaillé sur les hypothèses de l'algorithme, ce qui a donné l'algorithme **CPC**, pour *Conservative* PC.

Algorithme CPC (RAMSEY, SPIRTES et J. ZHANG 2006)

L'idée derrière l'algorithme Conservative PC est de remettre en question l'hypothèse de fidélité. Les auteurs proposent de se concentrer sur deux implications de la fidélité,

Algorithm 3 Algorithme PC-stable

Entrée : un ensemble d'observations **S**, un ensemble de variables **V**
Sortie : un graphe non orienté G

1: $G \leftarrow (\mathbf{V}, \mathbf{E})$
2: Sepset$\leftarrow \emptyset$
3: $m \leftarrow -1$
4: **Répéter**
5: $m \leftarrow m+1$
6: **Pour tout** $X \in \mathbf{V}$ **Faire**
7: $adj_X \leftarrow \{Y \in \mathbf{V} : (X,Y) \in E\}$
8: **Fin Pour**
9: **Répéter**
10: Sélectionner une nouvelle paire ordonnée (X,Y) adjacente telle que $|adj_X \backslash Y| \geq m$
11: **Répéter**
12: Sélectionner un nouvel ensemble $\mathbf{Z} \subseteq adj_X \backslash Y$ tel que $|\mathbf{Z}| = m$
13: **Si** $X \perp Y | \mathbf{Z}$ **Alors**
14: Supprimer l'arête entre X et Y dans G
15: $Sepset_{XY} \leftarrow Sepset_{XY} \cup \mathbf{Z}$
16: **Fin Si**
17: **Jusqu'à ce que** tous les ensembles $\mathbf{Z} \subseteq adj_X$ tels que $|\mathbf{Z}| = m$ on été testés ou qu'une indépendance ait été trouvée
18: **Jusqu'à ce que** toutes les paires ordonnées (X,Y) adjacentes telles que $|adj_X \backslash Y| \geq m$ ont été testées
19: **Jusqu'à ce que** $\forall X \in \mathbf{V}$, $|adj_X| \leq m$
20: **Retourner** G

qui sont donc des hypothèses moins fortes que la fidélité. La première implication est la **fidélité d'adjacence**.

Théorème 2.3.1 (Fidélité d'adjacence) *Pour un ensemble de variables* **V** *dont la structure peut être représentée par un DAG G, si* $X, Y \in \mathbf{V}$ *sont adjacents dans G, alors ils sont dépendants conditionnellement à tous les ensembles* $\mathbf{V} \backslash \{X, Y\}$.

La deuxième implication est celle de **fidélité d'orientation**.

Théorème 2.3.2 (Fidélité d'orientation) *Pour un ensemble de variables* **V** *dont la structure peut être représentée par un DAG G, soit* $\langle X, Y, Z \rangle$ *trois variables telles que X et Z sont adjacents, Y et Z sont également adjacents, mais X et Y ne le sont pas.*

— *si* $X \rightarrow Z \leftarrow Y$, *alors X et Y sont dépendants sachant tout sous-ensemble de* $\mathbf{V} \backslash \{X, Y\}$ *contenant Z*
— *sinon, X et Y sont dépendants conditionnellement à tout sous-ensemble de* $\mathbf{V} \backslash \{X, Y\}$ *qui ne contient pas Z*

Les auteurs proposent de relâcher l'hypothèse de fidélité d'orientation. La règle de PC permettant d'orienter les v-structures n'est alors plus correcte et peut amener à des erreurs. Pour corriger cela, une nouvelle procédure pour l'étape d'orientation des v-structures a été proposée, et est retranscrite dans l'algorithme 4.

Algorithm 4 Algorithme Conservative PC (CPC)

Entrée : un graphe non orienté $G = (\mathbf{V}, \mathbf{E})$
Sortie : un graphe partiellement orienté marqué

1: **Pour tout** $(X, Y) \in \mathbf{V}^2$ non adjacent avec un voisin Z en commun **Faire**
2: $PPa_X \leftarrow \{T \in \mathbf{V} : (X, T) \in \mathbf{E} \wedge T \neq Z\}$
3: $PPa_Y \leftarrow \{T \in \mathbf{V} : (Y, T) \in \mathbf{E} \wedge T \neq Z\}$
4: **Si** Z n'est dans aucun sous-ensemble $\boldsymbol{\nu}$ de PPa_X ou PPa_Y pour lequel $X \perp Y | \boldsymbol{\nu}$ **Alors**
5: Orienter $X \rightarrow Z \leftarrow Y$
6: **Sinon Si** Z est dans tous les sous-ensembles $\boldsymbol{\nu}$ de PPa_X ou PPa_Y pour lesquels $X \perp Y | \boldsymbol{\nu}$ **Alors**
7: Laisser $X - Z - Y$ non orienté
8: **Sinon**
9: Marquer le triplet (X, Y, Z) comme "infidèle"
10: **Fin Si**
11: **Fin Pour**

L'adjectif "*conservative*" prend tout son sens ici. L'algorithme va se montrer plus prudent dans l'orientation des v-structures, et s'abstenir de faire des orientations sur lesquelles des doutes existeraient.

Les algorithmes vus jusqu'ici utilisent plus ou moins le même principe. Ils utilisent des tests d'indépendance pour construire un graphe non orienté puis orientent les arêtes suivants des règles précises. Mais il existe également dans la littérature des algorithmes sous contraintes différents qui exploitent la notion de couverture de Markov pour tenter de diminuer le nombre de tests d'indépendance effectués. Nous allons découvrir ici quelques exemples de tels algorithmes permettant de découvrir la couverture de Markov d'une variable. Le premier algorithme est l'algorithme GSMB (MARGARITIS et THRUN 1999), pour *Grow-Shrink Markov Blankets* ("Grow-Shrink" signifiant "Grandir-Rétrécir" en français, et "Markov Blanket" signifiant "couverture de Markov").

Algorithme GSMB (MARGARITIS et THRUN 1999)

L'algorithme GSMB prend en argument une variable qu'on notera X et un ensemble de variables qu'on notera $\mathbf{V}$ tel que $X \in \mathbf{V}$. Son objectif est de trouver et renvoyer la couverture de Markov de X, notée ici $MB(X)$. Pour cela, comme le laisse penser son nom, l'algorithme s'articule autour de deux étapes : une étape de croissance et une étape d'élagage.

Durant la première étape, l'algorithme part d'un ensemble $MB_{tmp}(X)$ vide et va le remplir progressivement. Pour cela, on énumère toutes les variables Y de $\mathbf{V}\backslash X$. Pour chacune d'entre elles, l'algorithme teste l'indépendance entre X et Y sachant les variables déjà dans $MB_{tmp}(X)$. En effet, si une variable est indépendante de X sachant les variables de $MB_{tmp}(X)$, alors on est sûr qu'elle n'appartient pas à $MB(X)$. Mais ce faisant, il est fort probable que des faux positifs soient ajoutés dans $MB_{tmp}(X)$. Par exemple, si X et Z sont indépendants sachant Y, mais dépendants marginalement, alors il suffit que Z soit étudiée avant Y pour être ajoutée à $MB_{tmp}(X)$.

Pour cette raison, l'algorithme possède une deuxième étape, où on va supprimer les éléments en trop dans $MB_{tmp}(X)$. Pour cela, l'algorithme va énumérer les éléments de $MB_{tmp}(X)$. Pour chaque élément Y de $MB_{tmp}(X)$, l'indépendance entre X et Y sachant $MB_{tmp}(X)\backslash Y$ est de nouveau testée. En cas d'indépendance, on peut être sûr que cette variable n'appartient pas à $MB(X)$.

Ces deux étapes sont résumées succinctement dans l'algorithme 5.

Les auteurs proposent également une optimisation qui nécessite d'ordonner les variables de $\mathbf{V}\backslash\{X\}$ en fonction de leur proximité avec X. Pour évaluer cette "proximité", ils ont utilisé l'information mutuelle marginale, donnée par la formule 2.16, et qui permet d'estimer l'information commune partagée par deux variables aléatoires.

$$I(X;Y) = \sum_{x,y} \theta_{xy} log\left(\frac{\theta_{xy}}{\theta_x \theta_y}\right) \tag{2.16}$$

Cet ordonnancement est fait avant la première étape. Malheureusement, l'information mutuelle marginale étant utilisée, les époux pourraient se retrouver en mauvaise position dans cet ordre. En effet, ceux-ci ne partagent pas une grande proximité marginale, mais seulement une proximité conditionnelle sachant leurs enfants. Cette heuristique peut donc conduire à ajouter trop de variables aléatoires durant la première étape, ce qui a pour conséquence d'augmenter le temps de calcul, et de diminuer la fiabilité des tests d'indépendance (TSAMARDINOS, ALIFERIS et STATNIKOV 2003). C'est pourquoi TSAMARDINOS, ALIFERIS et STATNIKOV 2003 ont proposé un autre algorithme, IAMB, pour *Incremental Association Markov Blanket*.

Algorithme IAMB (TSAMARDINOS, ALIFERIS et STATNIKOV 2003)

L'algorithme IAMB utilise les mêmes deux étapes que GSMB, à la différence près que l'ordonnancement sur les variables se fait à chaque modification de $MB_{tmp}(X)$ durant l'étape de croissance. À chaque boucle de la première étape, on évalue l'information mutuelle entre X et Y sachant toutes les variables déjà dans $MB_{tmp}(X)$. La variable qui sera testée sera celle qui aura l'information mutuelle la plus grande. Cela permet d'ajouter d'abord les variables les plus proches (parents et enfants), puis les époux.

En 2005, YARAMAKALA et MARGARITIS 2005 ont proposé une nouvelle version optimisée de cet algorithme, appelée *Fast-IAMB*.

Algorithm 5 Algorithme GSMB

Entrée : un ensemble de variables **V**, une variable $X \in \mathbf{V}$, un ensemble d'observations **S**

Sortie : la couverture de Markov de X notée $MB(X)$

1: $MB_{tmp} \leftarrow \emptyset$
2: **Tant que** $\exists Y \in \mathbf{V} \backslash \{X\}$ telle que $Y \not\perp X | MB_{tmp}$ **Faire**
3: choisir une telle variable Y
4: $MB_{tmp} \leftarrow MB_{tmp} \cup \{Y\}$
5: **Fin Tant que**
6: **Tant que** $\exists Y \in MB_{tmp}$ telle que $Y \perp X | MB_{tmp}$ **Faire**
7: choisir une telle variable Y
8: $MB_{tmp} \leftarrow MB_{tmp} \backslash \{Y\}$
9: **Fin Tant que**
10: **Retourner** MB_{tmp}

Algorithme Fast-IAMB (YARAMAKALA et MARGARITIS 2005)

L'algorithme Fast-IAMB (6) est supposé faire moins de tests d'indépendance que IAMB, conduisant ainsi à une exécution plus rapide. De plus, les tests sont également censés avoir des ensembles de conditionnement plus petits, ce qui augmente la fiabilité des tests. Pour arriver à ce résultat, les auteurs ont d'abord légèrement changé la structure de l'algorithme, puisque celui-ci passe maintenant par plusieurs itérations d'alternance entre les étapes de croissance et d'élagage (modification déjà proposée dans l'algorithme Inter-IAMB par TSAMARDINOS, ALIFERIS et STATNIKOV 2003). Mais la principale optimisation vient du fait qu'à chaque étape de croissance, l'algorithme va ajouter non pas une mais plusieurs variables.

Pour cela, il faut sélectionner un ensemble de variables candidates avant chaque étape de croissance. Cette sélection se fait en ne choisissant que les variables de $\mathbf{V} \backslash (X \cup MB_{tmp}(X))$ dépendantes de X sachant $MB_{tmp}(X)$. Plusieurs variables parmi ces candidats seront ajoutées à $MB_{tmp}(X)$. L'étape d'élagage se fera comme pour IAMB, en supprimant les variables indépendantes de X sachant le nouvel ensemble $MB_{tmp}(X)$.

La question qui persiste est de savoir combien d'éléments sont ajoutés lors de l'étape de croissance. Pour cela, les auteurs ont utilisé une mesure qui se base sur la taille de la base de données et la taille de domaine des variables en jeux. Lorsque cette mesure dépasse un seuil, on estime que prendre plus de variables diminuerait la fiabilité des tests d'indépendances de l'étape d'élagage. On arrête alors d'ajouter de nouvelles variables à $MB_{tmp}(X)$ et on passe à l'étape d'élagage suivante.

Enfin, deux autres différences existent entre Fast-IAMB et IAMB. D'abord, là où les auteurs à l'origine de IAMB utilisaient le test du χ^2 pour faire les tests d'indépendance, c'est plutôt le test du G^2 qui est utilisé dans l'algorithme Fast-IAMB. De plus, l'ordonnancement des variables, qui se fait maintenant sur l'ensemble des variables

candidates, ne se fait plus avec l'information mutuelles conditionnelles, mais avec la p-value du test G^2.

Algorithm 6 Algorithme Fast-IAMB

Entrée : un ensemble de variables **V**, une variable $X \in \mathbf{V}$, un ensemble d'observations **S**
Sortie : la couverture de Markov de X $MB(X)$

1: $MB_{tmp} \leftarrow \emptyset$
2: $\mathbf{C} \leftarrow \{Y | Y \in \mathbf{V} \backslash \{X\} \wedge Y \not\perp\!\!\!\perp X\}$
3: **Tant que** $\mathbf{C} \neq 0$ **Faire**
4: $\langle Y_1, \cdots, Y_{|\mathbf{C}|} \rangle \leftarrow \mathbf{C}$ trié selon la p-value du test G^2 entre les variable de **C** et X sanchant MB_{tmp}
5: $donnée_insuffisantes \leftarrow FALSE$
6: **Pour tout** $i = 1$ à $|\mathbf{C}|$ **Faire**
7: **Si** $\frac{N}{r_{Y_i} \times r_X \times r_{MB_{tmp}}} \leq k$ **Alors**
8: $MB_{tmp} \leftarrow MB_{tmp} \cup \{Y_i\}$
9: **Sinon**
10: $donnée_insuffisantes \leftarrow TRUE$
11: Retourner à la ligne 15
12: **Fin Si**
13: **Fin Pour**
14: **Pour tout** $Y \in MB_{tmp}$ **Faire**
15: **Si** $Y \perp X | MB_{tmp} \backslash Y$ **Alors**
16: $MB_{tmp} \leftarrow MB_{tmp} \backslash \{Y\}$
17: **Fin Si**
18: **Fin Pour**
19: **Si** $donnée_insuffisantes = TRUE$ et aucune variable n'a été enlevée durant l'étape d'élagage **Alors**
20: **Retourner** MB_{tmp}
21: **Sinon**
22: $\mathbf{C} \leftarrow \{Y | Y \in \mathbf{V} \backslash (\{X\} \cup MB_{tmp}) \wedge Y \not\perp\!\!\!\perp X | MB_{tmp}\}$
23: **Fin Si**
24: **Fin Tant que**

2.3.3.2 Algorithmes à base de score

Voyons maintenant les algorithmes à base de score. Comme leur nom l'indique, ces algorithmes utilisent un score pour évaluer les solutions aux problèmes d'apprentissage de réseaux bayésiens. Les solutions sont rencontrées dans un espace de recherche qui est parcouru à l'aide d'une stratégie spécifique. On a donc ici trois composants des algorithmes à base de score : le score, l'espace de recherche et l'heuristique qui permet d'explorer l'espace de recherche. Les algorithmes à base de score se différen-

cient les uns des autres dans les composants qu'ils utilisent. Dans cette partie, nous analyserons indépendamment chacun des composants en donnant des exemples de variétés de ceux-ci et des exemples d'algorithmes qui les utilisent.

Les scores

Commençons avec les scores. L'algorithme va évoluer au sein d'un espace de recherche constitué de solutions potentielles. Pour se déplacer dans cet espace, il va évaluer les solutions par lesquelles il passe. Les scores sont là précisément pour ces évaluations. Ils permettent de jauger une solution ou une partie d'une solution pour savoir si la direction est la bonne, ce qui se traduit par une amélioration globale des scores des solutions.

Définition 2.3.1 (Score) *On appelle score une fonction prenant en argument une structure et une base de données et renvoyant un nombre réel.*

$$score(G : \mathbf{S}) : \mathscr{G} \times \mathbf{S} \rightarrow \mathscr{R}$$

Il existe deux propriétés enviables pour les scores. La première est la propriété de **décomposabilité**. Elle permet de décomposer le score global d'une structure en scores locaux associés à chaque nœud sachant ses parents. Cette propriété apporte un avantage calculatoire important, car lors d'un changement mineur de la structure, il devient possible de ne recalculer que les scores locaux des nœuds dont l'ensemble des parents a changé. Il n'est pas nécessaire de recalculer le score pour les parties du graphe qui restent inchangées.

Définition 2.3.2 (Décomposabilité (KOLLER et FRIEDMAN 2009)) *Un score est dit décomposable si et seulement si le score d'un DAG G peut être réécrit de la façon suivante :*

$$score(G : \mathbf{S}) = \sum_i score(X_i | Pa_G(X_i) : \mathbf{S})$$

La deuxième propriété intéressante est celle d'équivalence. Lorsque les relations causales ne sont pas prises en compte, nous avons vu que des graphes d'une même classe d'équivalence de Markov représentent la même distribution. Il n'est donc pas nécessaire de faire de distinction entre ces graphes. La propriété d'équivalence permet justement d'attribuer un même score à tous les graphes d'une même classe d'équivalence. En d'autres termes :

Définition 2.3.3 (Équivalence(KOLLER et FRIEDMAN 2009)) *Un score est dit équivalent si et seulement si, pour tous graphes G_1 et G_2 :*

$$G_1 \equiv G_2 \Rightarrow score(G_1 : \mathbf{S}) = score(G_2 : \mathbf{S})$$

Découvrons maintenant les principaux scores existant dans la littérature. Nous allons commencer par la famille des scores fondés sur la vraisemblance. Le premier

score envisageable de cette famille est la vraisemblance elle-même. Rappelons que la vraisemblance correspond à la probabilité des données sachant le modèle. Ici le modèle est le graphe qu'on cherche à évaluer. Donc on évalue la capacité des données à correspondre avec le graphe. Ce score est calculé à l'aide de l'équation suivante :

$$\mathscr{L}(\mathbf{S}|G) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} (\theta_{i,j,k})^{N_{i,j,k}} \tag{2.17}$$

Pour manipuler des valeurs plus faibles, il est courant d'utiliser le logarithme de la vraisemblance plutôt que la vraisemblance elle-même. On obtient alors l'expression suivante :

$$\mathscr{LL}(\mathbf{S}|G) = \sum_{i=1}^{n} \sum_{j=1}^{q_i} \sum_{k=1}^{r_i} N_{i,j,k} log(\theta_{i,j,k}) \tag{2.18}$$

Malheureusement, ce score est très peu utilisé. En effet, la vraisemblance ne permet aucun contrôle sur la complexité de la structure. Or avec la vraisemblance, plus une structure est complexe, plus son score est élevé. On se retrouverait alors avec des structures complexes, avec beaucoup de paramètres, et donc difficile à utiliser. C'est ici qu'intervient le principe de parcimonie, aussi appelé rasoir d'Occam. Ce principe suggère de trouver le graphe qui peut s'adapter à la distribution, mais qui soit en même temps le moins complexe possible en limitant le nombre de paramètres nécessaires au modèle. L'utilisation de ce principe implique la présence d'un terme de pénalité, qui permet de réduire les scores des structures complexes. On peut par exemple soustraire à la log-vraisemblance la dimension du graphe, comme c'est le cas pour le score AIC proposé par AKAIKE 1970.

$$AIC(G:\mathbf{S}) = \mathscr{LL}(\mathbf{S}|G) - dim(G) \tag{2.19}$$

où $dim(G)$ est égale au nombre de paramètres indépendants du réseau bayésien. Une autre solution, le score BIC, a été proposée par SCHWARZ 1978 :

$$BIC(G:\mathbf{S}) = \mathscr{LL}(\mathbf{S}|G) - \frac{1}{2} dim(G) log(N) \tag{2.20}$$

avec N la taille de la base de données.

Ces solutions utilisent la vraisemblance pour évaluer les structures. Mais comme pour les paramètres, il est possible d'utiliser une distribution *a posteriori* $\pi(G|\mathbf{S})$ pour évaluer une structure G à la lumière d'un ensemble d'observations $\mathbf{S}$. C'est le principe des scores **bayésiens**. La solution pour évaluer une structure, selon ces méthodes, est de calculer la distribution *a posteriori* des structures en prenant en compte une distribution *a priori* sur ces mêmes structures. C'est la méthode utilisée par le score

BD proposé par COOPER et HERSKOVITS 1992 :

$$\begin{aligned} BD(G:\mathbf{S}) = \pi(G|\mathbf{S}) &= \frac{\mathscr{L}(\mathbf{S}|G)\times\pi(G)}{P(\mathbf{S})} \propto P(G,\mathbf{S}) \\ &= \int_{\Theta} P(\mathbf{S},G,\Theta)d\Theta \\ &= \int_{\Theta} P(\mathbf{S}|G,\Theta)P(\Theta|G)\pi(G)d\Theta \end{aligned}$$

Cette dernière ligne a été obtenue par application de la formule des probabilités composées 1.2. Le terme $\pi(G)$, qui est l'*a priori* sur les structures, est indépendant des paramètres. On peut donc réécrire :

$$BD(G:\mathbf{S}) = \pi(G)\int_{\Theta} P(\mathbf{S}|G,\Theta)P(\Theta|G)d\Theta \tag{2.21}$$

Il est également possible d'utiliser cette expression sans approximation, en utilisant un *a priori* de Dirichlet sur les paramètres. De cette manière, CHICKERING et HECKERMAN 1996 ont montré que :

$$BD(G:\mathbf{S}) = \pi(G)\prod_{i=1}^{n}\prod_{j=1}^{q_i}\frac{\Gamma(\alpha_{i,j})}{\Gamma(N_{i,j}+\alpha_{i,j})}\prod_{k=1}^{r_i}\frac{\Gamma(N_{i,j,k}+\alpha_{i,j,k})}{\Gamma(\alpha_{i,j,k})} \tag{2.22}$$

avec Γ la fonction Gamma et $\alpha_{i,j,k}$ les hyperparamètres de l'*a priori* de Dirichlet.

Malheureusement, le score BD ne respecte pas la propriété d'équivalence. C'est pourquoi HECKERMAN, GEIGER et CHICKERING 1995 ont proposé une nouvelle version de ce score, appelée **BDe**, qui est décomposable et équivalent. Il s'obtient en spécifiant les hyperparamètres de l'*a priori* de Dirichlet de la manière suivante :

$$\alpha_{i,j,k} = N' \times P(X_i = x_{i,k}, Pa(X_i) = pa(X_i)_j|G_c)$$

avec G_c le graphe complet et N' un nombre d'exemples équivalents définis par l'utilisateur. Ce N' permet de donner plus ou moins d'importance aux *a priori* en leur donnant une hypothétique place plus ou moins importante dans la base de données.

Mais cette version du score BD demande toujours de spécifier les paramètres $P(X_i = x_{i,k}, Pa(X_i) = pa(X_i)_j|G_c)$ ce qui peut s'avérer être une tâche fastidieuse. C'est pourquoi une troisième version du score va apparaître grâce aux travaux de BUNTINE 1991, qui propose de remplacer les paramètres $P(X_i = x_{i,k}, Pa(X_i) = pa(X_i)_j|G_c)$ par $\frac{1}{r_i q_i}$ et ainsi obtenir les *a priori* suivants :

$$\alpha_{i,j,k} = \frac{N'}{r_i q_i}$$

Ce nouveau score est appelé **BDeu** et ne possède plus qu'un seul paramètre N'.

Enfin, il existe un dernier score issue de BD, appelé score **K2**. COOPER et HERSKOVITS 1992 ont montré qu'en utilisant des hyperparamètres $\alpha_{i,j,k} = 1$, on peut réécrire le

score BD de la manière suivante :

$$K2(G:\mathbf{S}) = \prod_{i=1}^{n} \prod_{j=1}^{q_i} \frac{(r_i-1)!}{(N_{i,j}+r_i-1)!} \prod_{k=1}^{r_i} N_{i,j,k}! \tag{2.23}$$

Pour terminer cette partie sur les principaux scores de la littérature, on peut également citer les scores fondés sur la notion de **MDL** pour *Minimum Description Length* en anglais, soit Longueur de Description Minimum en français (RISSANEN 1978). Cette notion permet de mesurer la complexité d'un modèle en comptant le nombre de bits qu'il faut pour représenter ce modèle. Plusieurs versions de scores utilisant cette notion ont été présentées dans la littérature, dont on citera la version de BOUCKAERT 1993 :

$$MDL_{Bouckaert}(G:\mathbf{S}) = \mathscr{LL}(\mathbf{S}|G) - |A_G| log(N) - c \times Dim(B) \tag{2.24}$$

où $|A_G|$ est le nombre d'arcs contenu dans le graphe G et c est le nombre de bits utilisés pour stocker un paramètre du réseau bayésien.

Ces scores permettent d'évoluer au sein d'un espace de solutions. La nature de cet espace varie également d'un algorithme à l'autre, comme nous allons le découvrir maintenant.

Les espaces de recherche

Rappelons que l'objectif d'un algorithme à base de score est de parcourir un espace à la recherche de la solution optimale. Les espaces de recherche sont des ensembles de solutions dans lesquels vont évoluer les algorithmes. Et bien que l'objectif des algorithmes soit de trouver un DAG, les espaces de recherche peuvent varier selon la nature des solutions qu'ils contiennent.

Selon la nature des solutions possibles, il faut également adapter la façon de se déplacer dans l'espace de recherche de ces solutions. Pour chaque algorithme d'apprentissage à base de score, on va donc définir une structure de voisinage qui correspond à tous les éléments qu'on pourra atteindre à partir d'une solution particulière. Ce voisinage peut être atteint à l'aide d'opérations propres aux espaces de recherche.

On va notamment distinguer 3 espaces de recherches différents.

Le premier est **l'espace des DAG**. C'est l'espace le plus intuitif. On cherche un DAG, on va donc évaluer des DAG pour trouver celui qui correspond le plus à notre distribution. Différents ensembles d'opérations sont possibles pour cet espace. Le premier est celui de l'algorithme K2 (COOPER et HERSKOVITS 1992) et l'algorithme B (BUNTINE 1991). Ces algorithmes ne permettent qu'une seule opération qui est l'ajout d'arcs. On part donc du graphe vide, et on ajoute les arcs les plus intéressants selon ce qu'indique le score.

Mais il apparaît vite que certains arcs qui, à première vue, peuvent paraître intéressant sont finalement sous-optimaux, et qu'il faille les inverser ou même les supprimer. BOUCKAERT 1995 a donc proposé d'autres opérations en plus de l'addition : la suppression et l'inversion. Bien que cette dernière opération soit accessible par suppression

puis addition, elle est nécessaire car la structure intermédiaire pourrait être de qualité inférieure, et ainsi difficilement envisagée par l'algorithme.

Ce type d'espace possède plusieurs problèmes. D'abord, le nombre de DAG possibles pour un ensemble de nœuds de taille n est donné par la formule récursive $G(0) = 1$, $G(n) = \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} 2^{i(n-1)} G(n-i)$ (ROBINSON 1977). Par exemple, pour $n = 10$, on compte approximativement $4,2 \times 10^{18}$ structures différentes. Même pour des graphes de petite taille (quelques dizaines de nœuds), l'espace des DAG est donc très grand, ce qui nuit à la complexité des algorithmes. De plus, lorsque l'aspect causal des relations entre les variables n'est pas important, alors grâce à la propriété d'équivalence des scores, évoluer entre des DAG d'une même classe d'équivalence peut être futile. Une façon de remédier à ces problèmes est de passer à un espace de plus petite taille.

Le second espace qu'on peut trouver est **l'espace des classes d'équivalence**, comme utilisé dans les algorithmes GES (CHICKERING 2002b) et FGES (RAMSEY, M. GLYMOUR, SANCHEZ-ROMERO et al. 2017). En effet, les DAG au sein d'une classe d'équivalence présentent les mêmes indépendances, et peuvent donc être intervertis, à condition bien sûr que les relations causales ne soient pas importantes dans la modélisation. En outre, pour un même nombre de noeuds, l'espace des CPDAG est plus petit que celui des DAG. Pour ces raisons, il peut s'avérer plus intéressant d'évoluer dans l'espace des classes d'équivalence plutôt que celui des DAG.

Pour évoluer dans ces espaces, il faut un ensemble d'opérations spécifique, par exemple celui proposé par CHICKERING 2002a qui s'applique sur un CPDAG et permet d'assurer de changer de CPDAG, et donc de modèle d'indépendance et de classe d'équivalence. Ces opérations sont au nombre de 6 :

— **InsertU** : Ajout d'une arête non orientée entre deux nœuds X et Y non adjacents
— **DeleteU** : Suppression d'une arête non orientée entre deux nœuds X et Y
— **InsertD** : Ajout d'un arc du nœud X vers le nœud Y non adjacent
— **DeleteD** : Suppression d'un arc du nœud X vers le nœud Y
— **ReverseD** : Inversion d'un arc du nœud X vers le nœud Y
— **MakeV** : Ajout d'une v-structure entre trois nœuds (X, Y, Z) tels que la chaîne non orientée $X - Y - Z$ existe et que X et Z ne sont pas adjacents

Nous pouvons remarquer que l'application de l'une de ces règles sur un CPDAG ne permet pas forcément d'avoir un autre CPDAG, mais aboutit souvent à un PDAG. Rappelons que la différence entre un CPDAG et un PDAG est que ce dernier peut avoir des arêtes, invariables dans la classe d'équivalence, non orientés. Pour cela, les auteurs de CHICKERING 2002b proposent d'appliquer successivement 2 procédures après chaque utilisation d'une opération : PDAG_vers_DAG puis DAG_vers_CPDAG. Ces procédures, comme leurs noms l'indiquent, proposent respectivement d'obtenir un DAG à partir d'un PDAG et un CPDAG à partir d'un DAG (voir CHICKERING 2002a pour plus de détails).

Enfin, un autre espace possible est **l'espace des ordres**, comme proposé par L. D. CAMPOS et J. PUERTA 2001 ou TEYSSIER et KOLLER 2005. Pour découvrir cet espace de recherche, définissons d'abord la notion d'**ordre**.

Définition 2.3.4 (Relation d'ordre strict) *On appelle relation d'ordre strict, ou plus simplement ordre strict, une relation $\prec$ sur un ensemble* **E** *qui, pour tout $x, y, z \in$* **E**, *présente les deux propriétés suivantes :*

— ***irréflexivité*** *: $x \not\prec x$*
— ***asymétrie*** *: $x \prec y \Rightarrow x \not\prec y$*
— ***transitivité*** *: $x \prec y \wedge y \prec z \Rightarrow x \prec z$*

On dit alors que l'ensemble **E** *est un ensemble strictement ordonné.*

Dans le reste de ce manuscrit, par soucis de simplicité, nous parlerons d'ordre pour désigner la notion d'ordre strict, bien que les deux notions soient en réalité différentes. Dans le contexte des réseaux bayésiens, l'ensemble sur lequel va être appliqué la relation est l'ensemble des variables **V** et l'ordre lui-même représente la relation de *non-ancestralité*. C'est-à-dire :

$$X \prec Y \Rightarrow Y \text{ n'est pas un ancêtre de } X \text{ dans } G \tag{2.25}$$

On dit qu'un ordre $\prec$ est **compatible** avec un DAG G lorsque l'implication 2.25 est vérifiée. Un ordre peut être compatible avec plusieurs DAG, et plusieurs ordres peuvent être compatibles avec un même DAG, comme l'atteste l'exemle 2.3.1.

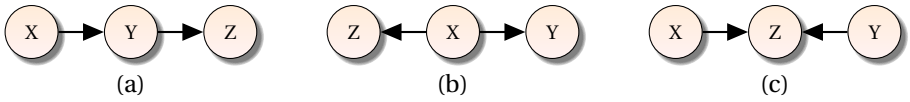


FIGURE 2.1 – Trois DAG différents compatibles avec le même ordre $X \prec Y \prec Z$.

Exemple 2.3.1 *Sur la figure 2.1, on peut voir trois DAG étant tous les trois compatibles avec l'ordre $X \prec Y \prec Z$. De plus les ordres $X \prec Y \prec Z$ et $X \prec Z \prec Y$ sont tous les deux compatibles avec le DAG 2.1b.*

Pour utiliser l'espace des ordres, il est nécessaire de définir un voisinage et donc un ensemble d'opérations. Généralement, la seule opération utilisée pour cet espace est la permutation de deux éléments. Par exemple, toujours pour trois variables, on passe de $X \prec Y \prec Z$ à $X \prec Z \prec Y$ en permutant les éléments Y et Z.

Le voisinage d'un ordre peut notamment être restreint, comme proposé par L. D. CAMPOS et J. PUERTA 2001. Si on assigne un indice $i \in \{0, \ldots, n\}$ à chaque variable telle que $X \prec Y \Rightarrow i(X) < i(Y)$, alors la restriction de rayon r revient à ne permettre une permutation entre X et Y que lorsque $|i(X) - i(Y)| \leq r$. Ceci diminue la taille du voisinage et donc accélère le processus de recherche.

Cet espace de recherche possède plusieurs autres avantages. D'abord, il est plus petit que l'espace des DAG. En effet, pour n variables, il existe $n!$ relations d'ordre possibles. Ensuite, la permutation est une modification de grande amplitude lorsqu'elle est rapportée aux DAG. En d'autres termes, une permutation peut apporter de grands changements dans les solutions possibles et cela aide à éviter les maxima locaux.

Enfin, limiter les solutions aux graphes compatibles avec un ordre permet d'assurer automatiquement la condition d'acyclicité.

Cependant, pour évaluer un ordre, il est nécessaire de passer par l'évaluation des DAG compatibles avec cet ordre. En d'autres termes, un ordre est évalué de la manière suivante :

$$score(\prec : \mathbf{S}) = \max_{G \in \mathcal{G}_\prec}(f(G : \mathbf{S})) \tag{2.26}$$

où $\mathcal{G}_\prec$ l'ensemble des graphes compatibles avec l'ordre $\prec$ et f une fonction d'évaluation des DAG. Il faut donc trouver le meilleur graphe compatible avec $\prec$. Cela signifie qu'il nécessaire de revenir à un espace restreint des DAG, et donc subir les inconvénients de ces espaces.

Maintenant que nous avons vu quel pouvait être les espaces de recherche, il nous faut également une stratégie pour parcourir ces espaces. On appelle ces stratégies des **heuristiques**.

Les heuristiques

L'heuristique est la stratégie qui permet de se déplacer efficacement au sein de l'espace de recherche. Une stratégie "efficace" cherche deux choses :

— éviter les maxima locaux. Un algorithme peut s'arrêter à un maximum local, c'est-à-dire une solution meilleure que tous ses voisins, mais moins bonne qu'un autre élément de l'espace.
— arriver à une solution optimale le plus rapidement possible en termes de temps de calcul et/ou nombre d'opérations

L'heuristique la plus simple est appelée **escalade** (*hill-climbing* en anglais)ou **recherche gloutonne** (*greedy search* en anglais). À chaque étape, cette méthode va évaluer chaque solution du voisinage, sélectionner le meilleur voisin puis choisir ce voisin comme nouvelle solution. Ensuite, elle recommence à évaluer le voisinage, choisir le meilleur voisin, et ce jusqu'à ce qu'il n'y ait plus de voisin meilleur que la solution courante. Les ensembles d'opérations sont un premier moyen de faire varier cette stratégie, comme on l'a déjà vu. Par exemple, l'algorithme B (BUNTINE 1991) propose de n'utiliser qu'une seule opération, l'addition d'arcs. L'algorithme K2 n'utilise que l'opération d'addition d'arcs également, mais propose d'utiliser un ordre $\prec$ sur les variables et suppose les ensembles parents-nœuds indépendants les uns des autres, ce qui permet de se contenter de trouver les parents de chaque nœud, les uns après les autres. L'algorithme GES (CHICKERING 2002b) propose pour sa part une variation de la stratégie d'escalade divisée en deux étapes : une étape d'addition, puis une étape de suppression, avec donc seulement deux opérateurs.

Le principal défaut de la recherche gloutonne est sa sensibilité aux maxima locaux. C'est un défaut classique dans les problèmes d'optimisation, et de nombreuses solutions existent et peuvent être appliquées à l'apprentissage de réseaux bayésiens. Parmi elles, on peut citer celles dont parle BOUCKAERT 1995 :

— **la liste tabou** (GLOVER 1986), qui permet d'interdire de défaire les dernières opérations appliquées

— **le redémarrage aléatoire**, qui consiste à refaire plusieurs fois la recherche gloutonne en partant de solutions de départ différentes
— **le recuit simulé** (KIRKPATRICK, GELATT JR et VECCHI 1983), qui est une méta-heuristique dont le principe général est de permettre d'accéder à des solutions parfois moins bonnes pour sortir de maxima locaux

D'autres méthodes s'éloignent plus sensiblement de la recherche gloutonne, mais sont toujours issues des solutions aux problèmes d'optimisation et ont été appliquées à l'apprentissage de réseaux bayésiens :

— les **algorithmes bio-inspirés** tels que les algorithmes génétiques (LARRANAGA, POZA, YURRAMENDI et al. 1996), les colonies de fourmis ou les colonies d'abeilles
— **A*** (YUAN et MALONE 2013), qui est un algorithme de recherche générale
— **VSN** (L. D. CAMPOS et J. PUERTA 2001), pour *Variable Neighborhood Search* en anglais, ou "recherche à voisinage variable" en français, qui permet d'explorer un voisinage plus étendu en accédant à des solutions distantes de plus d'une opération
— **B&B** (C. P. d. CAMPOS, ZENG et Q. JI 2009), pour *Branch and Bound*, ou "séparation et évaluation" en français, qui est une façon de parcourir l'espace de recherche en mettant de côté des ensembles de solutions jugées mauvaises.

2.3.3.3 Algorithmes hybdrides

Enfin, il existe une dernière catégorie d'algorithmes, appelés **algorithmes hybrides**. Ceux-ci sont une combinaison de méthodes issues des paradigmes sous contraintes et à base de score. L'objectif de cette combinaison est d'exploiter les avantages de chaque paradigme pour obtenir un algorithme qui surpasse ceux des catégories sous contraintes et à base de score.

Nous allons ici faire une liste non exhaustive des algorithmes hybrides existants, permettant seulement de se faire une idée des possibilités. En premier lieu, le premier algorithme de ce genre est l'algorithme Constraint-Bayesian de SINGH et VALTORTA 1993.

Algorithme *Constraint-Bayesian* (SINGH et VALTORTA 1993)

L'algorithme Constraint-Bayesian a pour but d'exploiter l'algorithme K2 vu précédemment, qui a la capacité de trouver un DAG à partir d'un ordre. Son pseudo-code est disponible dans les algorithmes 7 et 8. Il est composé de deux étapes qui se répètent alternativement. L'algorithme démarre avec un graphe complet G_1 et un entier $ord = 0$. La première étape utilise l'algorithme PC en exécutant des tests d'indépendance d'ordre ord seulement lors de la phase de construction du squelette (ligne 4 à 17). Les autres phases de l'algorithme PC vont s'effectuer sur une copie du graphe G_1 obtenu après les tests, de sorte à préserver ce dernier (ligne 18 à 20). Après orientation des v-structures et propagation des orientations, les dernières orientations sont obtenues par l'utilisation du score K2 (pas l'algorithme, juste le score donné par l'équation 2.23), ce qui permet d'obtenir un nouveau DAG (ligne 21 à 33).

Algorithme	Score	Espace	Heuristique	Référence
K2	K2	DAG	RG[1] (addition)	COOPER et HERSKOVITS 1992
Algorithme B	K2	DAG	RG (addition)	BUNTINE 1991
RG	BDe	DAG	RG (addition, suppression, inversion)	HECKERMAN, GEIGER et CHICKERING 1995
$RG_{Bouckaert}$	K2	DAG	Liste Tabou	BOUCKAERT 1995
AG	K2	DAG	Algorithme génétique	LARRANAGA, POZA, YURRAMENDI et al. 1996
HCSO	K2	Ordre	RG	L. D. CAMPOS et J. PUERTA 2001
VNSO	K2	Ordre	SVN	L. D. CAMPOS et J. PUERTA 2001
ACO-K2SN	K2	Ordre	Colonie de fourmis	L. M. d. CAMPOS, GÁMEZ et J. M. PUERTA 2002
GES	BIC	Classe d'équivalence	RG (addition, suppression)	CHICKERING 2002b
BCO	K2	DAG	Colonie d'abeilles	J. JI, WEI et C. LIU 2013
B&B	BIC/AIC	DAG	Branch&Bound	C. P. d. CAMPOS, ZENG et Q. JI 2009
A*	BIC	DAG	A*	YUAN et MALONE 2013
FGES	BIC	Classe d'équivalence	RG (addition, suppression)	RAMSEY, M. GLYMOUR, SANCHEZ-ROMERO et al. 2017

TABLEAU 2.1 – Récapitulatif des algorithmes à base de score cités dans la sous-section 2.3.3.2.
(1. RG = Recherche Gloutonne)

La deuxième étape commence avec un ordre complet extrait du DAG obtenu précédemment. Cet ordre va servir comme point de départ pour l'utilisation de l'algorithme K2, ce qui permet d'obtenir un nouveau DAG (ligne 34). Ces deux étapes se répètent jusqu'à ce qu'un DAG obtenu à la fin d'une itération $n+1$ ne soit pas meilleur que celui obtenu à l'itération n (ligne 35 à 43). À chaque itération, l'entier ord est incrémenté, ce qui augmente l'ordre des tests d'indépendance de l'étape sous contraintes. Notons que le graphe utilisé pour la construction du squelette est lui toujours le graphe G_1 qui se modifie à chaque itération.

Algorithme *Sparse Candidate* (FRIEDMAN, NACHMAN et PE'ER 1999)

L'algorithme *Sparse Candidate* ("Candidat épars" en français) propose également l'alternance d'une étape sous contraintes et d'une étape à base de score. Pour une itération n, la première étape vise à trouver les candidats potentiels $\mathbf{C}_X^n$ pour le rôle de parents pour chaque nœud X. La deuxième étape vise ensuite à chercher le meilleur DAG G_n dans l'espace des structures qui respectent les contraintes trouvées à la première étape, donc telles que $\forall X \in \mathbf{V},\ Pa_{G_n}(X) \subset \mathbf{C}_X^n$.

Le DAG obtenu à la fin d'une itération est évalué. Les parents de chaque nœud sont automatiquement considérés comme des candidats potentiels pour l'itération suivante, ce qui signifie qu'à l'itération $n+1$, on a forcément $Pa_{G_n}(X) \subset \mathbf{C}_X^{n+1}$. Cela permet d'assurer la convergence de l'algorithme, puisque la structure obtenue à l'étape n devient une structure légale pour l'étape $n+1$. Ainsi la structure G_{n+1} sera au moins aussi bonne que la structure G_n. L'algorithme se termine lorsqu'aucune structure n'est meilleure que celle déjà trouvée à l'itération précédente.

Pour la sélection des candidats, les auteurs proposent d'utiliser l'information mutuelle (2.16), en faisant l'hypothèse que cette mesure permet d'évaluer la "proximité" entre deux nœuds. Pour l'étape à base de score, les auteurs expliquent qu'il est possible d'utiliser une simple recherche gloutonne. Mais ils proposent également une heuristique dite de "diviser et conquérir", qui consiste à diviser l'ensemble des variables $\mathbf{V}$ en sous-ensembles sur lesquels il sera plus simple de trouver la structure optimale.

Cet algorithme peut être vu comme une heuristique générale, tant la liberté est permise dans son implémentation. Celle proposée par les auteurs possède notamment trois défauts : une estimation des candidats qui n'est pas mathématiquement fondée, la nécessité pour l'utilisateur de renseigner un paramètre et une dispersion uniforme imposée sur le graphe. C'est pourquoi TSAMARDINOS, L. BROWN et ALIFERIS 2006 vont reprendre cette heuristique pour proposer l'algorithme **MMHC**, censé corriger ces défauts.

Algorithm 7 Algorithme Constraint-Bayesian

Entrée : un ensemble d'observations **S**, un ensemble de variables **V**, un entier u
Sortie : $\Pi = \{\pi_X\}_{X \in \mathbf{V}}$ et $Prob$ le score du meilleur graphe trouvé

1: $G_1 \leftarrow$ un graphe complet non orienté, Sepset$\leftarrow \emptyset$, $ord \leftarrow 0$
2: $Prob_{old} \leftarrow 0$ et $\pi_{X\,old} \leftarrow \{\}$, $\forall X \in \mathbf{V}$
3: **Répéter**
4: Choisir une nouvelle paire ordonnée (X, Y) adjacente t.q. $|adj(X) \backslash Y| \geq ord$
5: **Répéter**
6: Choisir un nouvel ensemble $\mathbf{Z} \subseteq adj(X) \backslash Y$ t.q. $|\mathbf{Z}| = ord$
7: **Si** $X \perp Y | \mathbf{Z}$ **Alors**
8: Supprimer l'arête entre X et Y dans G_1
9: $Sepset_{XY} \leftarrow Sepset_{XY} \cup \mathbf{Z}$
10: **Fin Si**
11: **Jusqu'à ce que** tous les ensembles $\mathbf{Z} \subseteq adj(X)$ tels que $|\mathbf{Z}| = m$ ont été testés ou qu'une indépendance a été trouvée
12: **Jusqu'à ce que** toutes les paires ordonnées (X, Y) adjacentes telles que $|adj(X) \backslash Y| \geq ord$ ont été testées
13: **Si** $deg(G_1) > u$ **Alors**
14: $ord \leftarrow ord + 1$ et retourner à la ligne 4
15: **Fin Si**
16: $G \leftarrow G_1$
17: Orienter les v-structures dans G à la manière de l'algorithme PC
18: Orienter autant d'arêtes dans G que possible par application des règles de Meek.
19: $\pi_X \leftarrow Pa_G(X)$, $\forall X \in \mathbf{V}$
20: **Pour tout** arête non orientée $X - Y$ dans G **Faire**
21: **Si** l'orientation $X \rightarrow Y$ introduit un circuit **Alors**
22: $\pi_X \leftarrow \pi_X \cup \{Y\}$
23: **Sinon**
24: $val_X \leftarrow g(X, \pi_X) \times g(Y, \pi_Y \cup \{X\})$, $val_Y \leftarrow g(Y, \pi_Y) \times g(X, \pi_X \cup \{Y\})$
25: **Si** $val_X > val_Y$ **Alors**
26: $\pi_Y \leftarrow \pi_Y \cup \{X\}$
27: **Sinon Si** $val_X > val_Y$ **Alors**
28: $\pi_X \leftarrow \pi_X \cup \{Y\}$
29: **Fin Si**
30: **Fin Si**
31: **Fin Pour**
32: $D \leftarrow$DAG formé grâce aux ensembles π_X, $\forall X \in \mathbf{V}$; $\prec \leftarrow$ un ordre compatible avec D
33: π'_X, $\forall X \in \mathbf{V} \leftarrow$ les nouveaux ensembles de parents obtenus avec l'algorithme K2
Suite à l'algorithme 8...

Algorithm 8 Algorithme Constraint-Bayesian - Partie 2

34: $Prob_{new} \leftarrow \prod_{X \in \mathbf{V}} g(X, \pi'_X)$
35: **Si** $Prob_{new} > Prob_{old}$ **Alors**
36: $Prob_{old} \leftarrow Prob_{new}$; $\pi_{X\,old} \leftarrow \pi'_X$, $\forall X \in \mathbf{V}$
37: $ord \leftarrow ord + 1$ et retourner à la ligne 4 (Algorithme 7)
38: **Sinon**
39: **Retourner** $\pi_{X\,old}$, $\forall X \in \mathbf{V}$ et $Prob_{old}$
40: **Fin Si**

Algorithme MMHC (TSAMARDINOS, L. BROWN et ALIFERIS 2006)

L'algorithme MMHC peut être vu comme une version particulière de l'algorithme *Sparse Candidate*. Les deux principales différences reposent dans l'étape sous contraintes et dans le fait que l'algorithme n'utilise qu'une seule fois les deux étapes, sans itération.

Concernant l'étape sous contraintes, les auteurs ont développé l'algorithme MMPC, retranscrit ici avec le pseudo-code 9. Comme pour l'algorithme précédent, il consiste à trouver, pour chaque variable $X \in \mathbf{V}$, les variables les plus proches. Mais dans le cas de MMPC, cet ensemble désigne les parents et les enfants et est noté $\mathbf{CPC}_X$. Ce dernier va être construit de la manière suivante : itérativement, on va ajouter la variable Y la plus fortement associée à la variable X sachant celles qui sont déjà dans $\mathbf{CPC}_X$, en s'arrêtant lorsque toutes les variables restantes sont complètement dissociées de X. La variable Y est choisie par MaxMin, c'est-à-dire qu'on va choisir la variable qui donne la plus forte valeur (max) de minimum d'association en fonction des sous-ensembles de $\mathbf{CPC}_X$. En d'autres termes : $Y = \text{argmax}_{Y \in \mathbf{V}} \min_{\boldsymbol{\nu} \subseteq \mathbf{CPC}} Assoc(X, Y|\boldsymbol{\nu})$

On va ensuite enlever les variables de $\mathbf{CPC}_X$ pour lesquelles on trouve une indépendance entre elle et X sachant un sous-ensemble de $\mathbf{CPC}_X$. Enfin, on remarque que si $Y \in \mathbf{CPC}_X$, alors X doit appartenir à $\mathbf{CPC}_Y$. Donc les ensembles **CPC** passent par une nouvelle étape qui consiste à supprimer Y de $\mathbf{CPC}_X$ si $X \notin \mathbf{CPC}_Y$. Les auteurs proposent d'utiliser la p-valeur des tests d'indépendance comme mesure de l'association entre deux variables, combinée avec la statistique du test si les p-valeurs sont égales.

Pour l'étape à base de score, une recherche gloutonne est utilisée avec les opérations d'addition, de suppression et d'inversion d'arcs, mais en ne permettant l'addition d'arcs entre X et Y que lorsque $X \in \mathbf{CPC}_Y$.

Enfin, comme l'algorithme précédent, il s'agit ici de l'implémentation d'un modèle. Les différentes parties de l'algorithme peuvent être remplacées par d'autres méthodes. Par exemple, les auteurs proposent d'utiliser l'heuristique tabou en plus de la recherche gloutonne lors de l'étape à base de score, et d'autres heuristiques pourraient très bien être envisagées. Une autre version a été par exemple proposée par NANDY, HAUSER et M. MAATHUIS 2018, il s'agit de l'algorithme **ARGES**.

Algorithm 9 Algorithme MMPC

Entrée : un ensemble d'observations **S**, un ensemble de variables **V**
Sortie : **CPC** les parents et enfants de chaque variable de **V** dans chaque réseau bayésien fidèle à la distribution de **S**

Pour tout $X \in \mathbf{V}$ **Faire**
  $\mathbf{CPC}_X \leftarrow \emptyset$
  Répéter
    $F, assocF = MaxMinHeuristique(X, \mathbf{CPC}_X)$
    Si $assocF \neq 0$ **Alors**
      $\mathbf{CPC} \leftarrow \mathbf{CPC}_X \cup F$
    Fin Si
  Jusqu'à ce que CPC ne change plus
  Pour tout $Y \in \mathbf{CPC}_X$ **Faire**
    Si $\exists \boldsymbol{\nu} \subseteq \mathbf{CPC}$ tel que $Y \perp X | \boldsymbol{\nu}$ **Alors**
      $\mathbf{CPC}_X \leftarrow \mathbf{CPC} \backslash \{Y\}$
    Fin Si
  Fin Pour
Fin Pour
Pour tout $X \in \mathbf{V}$ **Faire**
  Pour tout $Y \in \mathbf{CPC}_X$ **Faire**
    Si $X \notin \mathbf{CPC}_Y$ **Alors**
      $\mathbf{CPC}_X \leftarrow \mathbf{CPC}_X \backslash Y$
    Fin Si
  Fin Pour
Fin Pour
Retourner $\mathbf{CPC} = \{\mathbf{CPC}_X\}_{X \in \mathbf{V}}$

Algorithm 10 MaxMinHeuristique

Entrée : une variable X, l'ensemble de variable $\mathbf{CPC}_X$
Sortie : le maximum parmi toutes les variables de l'association minimum avec la variable X sachant $\mathbf{CPC}_X$ et la variable qui atteint ce maximum

$assocF \leftarrow \max_{Y \in \mathbf{V}} \min_{\boldsymbol{\nu} \subseteq \mathbf{CPC}} Assoc(X, Y | \boldsymbol{\nu})$
$F \leftarrow \operatorname{argmax}_{Y \in \mathbf{V}} \min_{\boldsymbol{\nu} \subseteq \mathbf{CPC}} Assoc(X, Y | \boldsymbol{\nu})$
Retourner F et $assocF$

Algorithme ARGES (NANDY, HAUSER et M. MAATHUIS 2018)

L'algorithme ARGES est une variation de l'algorithme MMHC, dans lequel la recherche gloutonne est remplacée par un algorithme GES. Ici, la partie à base de score s'appuie donc sur l'espace des classes d'équivalence de Markov et se divise en deux

étapes, une étape durant laquelle on ajoute des arcs et une autre durant laquelle on en supprime. La différence avec le travail de CHICKERING 2002b réside dans une utilisation préalable de l'algorithme MMPC permettant d'obtenir un squelette, restreignant ainsi l'espace de recherche parcouru lors de l'utilisation de l'algorithme GES.

De plus, la première étape de GES est adaptée pour prendre en compte ces contraintes. MMPC permet de renvoyer un squelette, et c'est cette structure qui est utilisée comme contrainte. Les auteurs ont défini un ensemble d'arêtes admissibles. Une arête entre deux nœuds non adjacents X et Y est admissible lorsqu'il existe une arête entre X et Y dans le squelette ou lorsqu'il existe un nœud Z tel que la structure $X - Z - Y$ existe. La deuxième étape correspond exactement à celle de l'algorithme GES.

On voit ainsi que les algorithmes hybrides se résument souvent à l'utilisation d'une étape sous contraintes et d'une étape à base de score. Ce modèle offre une grande liberté du fait de la possibilité de choisir les algorithmes utilisés à chaque étape.

2.3.4 Apprentissage avec des connaissances

Comme nous l'avons vu, du fait de leur complexité, la structure de réseaux bayésiens ne peut pas toujours être entièrement construite manuellement, et c'est pourquoi leur apprentissage automatique est un domaine de recherche actif. Cependant, si demander à un expert de construire totalement un graphe est difficile, il est toujours possible de lui demander de fournir des connaissances partielles sur ce graphes. De nombreuses études ont montré que l'exploitation de telles connaissances en plus des données peut améliorer l'apprentissage des modèles.

Ces connaissances d'experts peuvent revêtir de nombreuses formes et natures. Elles peuvent être certaines ou incertaines et impacter différentes parties des algorithmes d'apprentissage, de leur initialisations à leur conclusions.

Dans cette partie, nous verrons plusieurs d'exemples d'algorithmes capables d'exploiter des connaissances d'experts. Ces algorithmes pourront appartenir aux trois grandes catégories d'algorithmes : les algorithmes sous contraintes, à base de score et hybrides. Nous verrons ensuite de manière plus approfondie deux algorithmes en particulier, qui seront utiles pour la partie contribution de ce manuscrit.

2.3.4.1 Utilisation de connaissances certaines

Tout d'abord, étudions l'utilisation de connaissances certaines dans l'apprentissage de réseaux bayésiens. Il s'avère que nous avons déjà vu des exemples d'algorithmes utilisant de telles connaissances. En effet, l'algorithme K2 prend en entrée un ordre complet sur les variables. Bien qu'il existe des heuristiques pour chercher cet ordre, il est possible de demander à un expert de le fournir. Pour l'algorithme K2, l'ordre permet de contraindre l'espace de recherche en limitant les parents potentiels de chaque nœud.

Pour tous les algorithmes à base de score, l'utilisation de connaissances certaines se fait toujours selon le même principe : contraindre l'espace de recherche pour diminuer sa taille ainsi que la taille des voisinages. Cela permet non seulement d'accélérer la

recherche, mais également d'améliorer la qualité du modèle appris. De nombreuses formes de connaissances sont utilisables de cette façon. D'abord, au lieu d'utiliser un ordre complet, il est possible d'utiliser un ordre partiel, c'est à dire un ordre qui ne permet pas de comparer toutes les variables deux à deux entre elles. Même si ce type de connaissances contient moins d'information, il est toujours utile de les exploiter.

Par exemple, L. M. d. CAMPOS et CASTELLANO 2007 proposent deux algorithmes dont un appartient à la catégorie des algorithmes sous contraintes, et l'autre appartient à celle des algorithmes à base de score et est inspiré par la recherche gloutonne dans l'espace des DAG. Ce dernier algorithme est capable d'exploiter entre autres des ordres partiels. Dans ce cas, ces connaissances servent à empêcher la création de chemin dirigé. En effet, pour deux variables X et Y, si l'ordre partiel $\prec_{part}$ impose $X \prec_{part} Y$ alors il ne peut pas avoir de chemin de Y vers X.

L'algorithme à base de score de L. M. d. CAMPOS et CASTELLANO 2007 permet d'utiliser d'autres types de connaissances. Des connaissances sur la présence (ou absence) d'arcs ou d'arêtes pourront par exemple interdire ou imposer la présence d'arcs ou d'arêtes dans le graphe. En outre, il est également possible de prendre en compte des relations d'ancestralité. Si l'expert énonce qu'une variable X est l'ancêtre d'une variable Y, alors le graphe final devra forcément avoir un chemin dirigé de X vers Y.

On peut trouver dans l'article de C. P. d. CAMPOS, ZENG et Q. JI 2009 les mêmes principes, mais avec un type de connaissances supplémentaire. Les auteurs proposent d'utiliser également des connaissances sur le **degré maximum** du graphe. Cette notion correspond au nombre maximum de voisins que peut avoir un nœud. En pratique, il est souvent préférable d'avoir des graphes épars, avec un degré maximum faible. Et lorsqu'on sait que le modèle recherché est effectivement épars, alors on peut artificiellement limiter le nombre de voisins possibles, ce qui a en plus l'avantage d'accélérer l'apprentissage, puisque les modèles manipulés sont moins complexes.

La majorité de ces connaissances sont également utilisables dans le cadre d'algorithmes sous contraintes. L'algorithme sous contraintes proposé par L. M. d. CAMPOS et CASTELLANO 2007 permet de les prendre en compte. Nous verrons les détails de cet algorithme un peu plus tard dans ce manuscrit. Nous pouvons également parler de l'algorithme de HU, MO, X. ZHANG et al. 2012, fondé sur la structure de l'algorithme **SLA** (CHENG, GREINER, KELLY et al. 2002). De manière similaire à l'algorithme GSMB, cet algorithme comporte une étape de croissance et une étape d'élagage, mais ces étapes permettent de trouver la structure globale du graphe, et plus seulement le voisinage d'un nœud. HU, MO, X. ZHANG et al. 2012 ont proposé de l'adapter pour prendre en compte des connaissances sur des ordres partiels, sur des relations d'ancestralité et sur des présences ou absences d'arcs. Ces connaissances permettent alors de diminuer le nombre de tests d'indépendance qu'il est nécessaire d'effectuer, ce qui permet non seulement d'accélérer l'apprentissage, mais également de limiter l'impact des erreurs des tests d'indépendance lorsque ceux-ci ne sont pas fiables.

Enfin, si ces connaissances sont applicables à des algorithmes sous contraintes et des algorithmes à base de score, alors elles le sont aussi pour les algorithmes hybrides. On peut notamment citer les travaux de A. LI et BEEK 2018 sur ce sujet.

Au final, grâce à la certitude sur la valeur de vérité de ces connaissances, celles-ci sont facilement exploitables en contraignant le graphe. Malheureusement, les experts peuvent avoir des doutes, des incertitudes et faire des erreurs. Lorsque c'est le cas, il devient nécessaire de trouver des compromis entre ce qui est trouvé grâce aux données et ce qui indiqué par l'expert.

2.3.4.2 Utilisation de connaissances incertaines

L'exploitation de connaissances certaines s'appuyait, à quelques détails près, sur le même principe pour les algorithmes sous contraintes et à base de score. Cependant, pour les connaissances incertaines, il va être nécessaire d'utiliser les spécificités de chaque catégorie d'algorithme pour réussir à faire les compromis recherchés et correctement intégrer les connaissances dans l'apprentissage.

Débutons avec les algorithmes à base de score. Ceux-ci sont particulièrement adaptées à l'introduction de connaissances incertaines. En effet, l'utilisation de connaissances en tant que probabilité (et donc intrinsèquement incertaines) est au cœur de l'approche des scores bayésiens tels que BD (COOPER et HERSKOVITS 1992) ou BDe (HECKERMAN, GEIGER et CHICKERING 1995), où les connaissances peuvent être insérées sous la forme d'*a priori*. Pourtant, transposer les connaissances d'un expert sur la structure en *a priori* n'est pas une tâche aisée. C'est pourquoi RICHARDSON et DOMINGOS 2003 ont proposé une méthode pour directement introduire une distribution de probabilité sur la présence ou l'absence d'arcs ou d'arêtes dans le score.

Sans rentrer dans les détails de leur solution, la méthode consiste à calculer la probabilité $P(G|\mathbf{S}, E)$ au lieu de $P(G|\mathbf{S})$ comme c'est habituellement fait pour le score BDeu. La variable E représente le modèle des experts, en d'autres termes un ensemble de probabilités sur la présence ou l'absence d'arcs ou d'arêtes. Il est possible de prendre en compte les connaissances de plusieurs experts simultanément, représentées par des paramètres p_a, p_d, p_r, p_b et p_c, qui sont les probabilités respectives que : un expert rajoute un arc, un expert supprime un arc, un expert inverse un arc, un expert ajoute une arête là où il n'y avait rien, et la probabilité qu'un expert ajoute une arête là où il y avait un arc (suppression de l'orientation).

Une méthode similaire a été utilisée par AMIRKHANI, RAHMATI, LUCAS et al. 2017. La méthode exacte utilisée dans cette article sera détaillée un peu plus tard dans ce manuscrit.

Si l'utilisation de connaissances incertaines dans les algorithmes à base de score est plutôt naturelle, il n'en est pas de même avec les algorithmes sous contraintes. LUIS, SUCAR et MORALES 2010 ont tenté d'apporter une solution à ce problème en permettant de combiner les résultats des tests d'indépendance par les résultats de tests d'indépendance venant de problèmes similaires, sur les mêmes variables, par le biais de technique d'apprentissage par transfert. Ici, il ne s'agit donc pas exactement de connaissances d'experts, mais plutôt de connaissances venant d'autres problèmes déjà résolus.

Enfin, H. LI et GUO 2018 ont pour leur part proposé un algorithme hybride permettant de considérer des connaissances incertaines. Les connaissances concernent,

comme pour RICHARDSON et DOMINGOS 2003 et AMIRKHANI, RAHMATI, LUCAS et al. 2017, la présence ou l'absence d'arcs ou d'arêtes. Dans leur solution, chaque connaissance exprimée par un expert est accompagnée par une probabilité. L'idée est d'agglomérer les probabilités de chaque expert. L'algorithme en lui-même est un algorithme hybride classique, avec une étape reprenant PC et une étape reprenant une recherche gloutonne dans l'espace des DAG avec le score BIC. Durant l'étape sous contraintes, les connaissances sur les arêtes sont utilisées après l'étape de construction du squelette, celles sur les arcs après l'étape d'orientation. À chaque fois, un tirage aléatoire est fait selon une distribution uniforme entre 0 et 1, et la décision de valider ou invalider les structures trouvées est prise en fonction des probabilités obtenues après agglomération des connaissances. Concernant l'étape à base de score, c'est l'algorithme de AMIRKHANI, RAHMATI, LUCAS et al. 2017 qui est utilisé.

Nous avons survolé rapidement les algorithmes décrits dans cette partie. L'objectif pour la suite sera de donner deux exemples concrets d'algorithmes utilisant des connaissances. Ainsi, nous commencerons par voir l'algorithme sous contraintes proposé par L. M. d. CAMPOS et CASTELLANO 2007, que nous appellerons **PCDC**, puis l'algorithme à base de score de AMIRKHANI, RAHMATI, LUCAS et al. 2017, que nous appellerons **GSA**.

2.3.4.3 Algorithme PCDC

Le premier algorithme que nous traiterons est l'algorithme sous contraintes de L. M. d. CAMPOS et CASTELLANO 2007. Cet algorithme est capable de prendre en compte des connaissances certaines à propos de l'absence ou présence d'arcs ou arêtes et à propos d'ordre partiel sur les variables. Ces connaissances sont considérées comme des contraintes sur le graphe qu'il faut obligatoirement respecter. Les auteurs supposent que ces connaissances sont cohérentes entre elles. Par exemple, il n'est pas possible d'avoir un ensemble d'arcs imposés qui forment un circuit. L'algorithme est une adaptation de PC permettant de prendre en compte ces connaissances, il comporte donc les étapes de construction de squelette, d'orientation des v-structures et de propagation des orientations qu'on peut retrouver dans l'algorithme PC.

Nous allons commencé par définir plus formellement les connaissances utilisées dans cet algorithme. Celles-ci sont organisées sous la forme de plusieurs graphes. Il y a d'abord les graphes qui reprennent explicitement les connaissances fournies par l'expert, G_e, G_a et G_o, qui représentent respectivement les contraintes d'existence, d'absence et d'ordre. Soit $\mathbf{V}$ un ensemble de variables aléatoires.

— $G_e = (\mathbf{V}, \mathbf{E}_e)$ avec $\mathbf{E}_e = \{X - Y | X - Y \text{ est imposé}\} \cup \{X \rightarrow Y | X \rightarrow Y \text{ est imposé}\}$
— $G_a = (\mathbf{V}, \mathbf{E}_a)$ avec $\mathbf{E}_a = \{X - Y | X - Y \text{ est interdit}\} \cup \{X \rightarrow Y | X \rightarrow Y \text{ est interdit}\}$
— $G_o = (\mathbf{V}, \mathbf{E}_o)$ avec $\mathbf{E}_o = \{X \rightarrow Y | X \prec Y \text{ est imposé}\}$

Les contraintes sur l'ordre peuvent également se traduire de la façon suivante : $X \prec Y \Leftrightarrow Y \not\rightsquigarrow X$, où $Y \not\rightsquigarrow X$ signifie qu'il n'existe pas de chemin de Y vers X. Dans la suite, nous utiliserons également la notation $Y \not\rightsquigarrow X \in G$ pour dire qu'il n'existe pas de chemin de Y vers X dans le graphe G.

Grâce à cette modélisation des connaissances, il est possible de construire d'autres

graphes qui représentent également des connaissances implicites, conséquences des précédentes. G_{re} est le graphe qui représente les présences d'arcs et d'arêtes imposées directement plus celles imposées indirectement par certaines absences d'arcs. G_{au} représente les absences d'arêtes seulement. G_{ai} représente les arcs qui ne peuvent exister que dans un sens, s'ils existent. Et enfin, G_{eot} représente la fermeture transitive des graphes G_o et G_{re}, c'est-à-dire les arcs qui, s'ils existent, ne peuvent être orientés autrement sous peine de créer un circuit.

— $G_{re} = (\mathbf{V}, \mathbf{E}_{re})$ avec $\mathbf{E}_{re} = \{X \rightarrow Y | X \rightarrow Y \in \mathbf{E}_e\} \cup \{Y \rightarrow X | X - Y \in \mathbf{E}_e \wedge X \rightarrow Y \in \mathbf{E_a}\} \cup \{X - Y | X - Y \in \mathbf{E}_e \wedge X \rightarrow Y \notin \mathbf{E}_a \wedge Y \rightarrow X \notin \mathbf{E}_a\}$
— $G_{au} = (\mathbf{V}, \mathbf{E}_{au})$ avec $\mathbf{E}_{au} = \{X - Y | X - Y \in \mathbf{E}_a\}$
— $G_{ai} = (\mathbf{V}, \mathbf{E}_{ai})$ avec $\mathbf{E}_{ai} = \{X \rightarrow Y | Y \rightarrow X \in \mathbf{E}_a\}$
— $G_{eot} = (\mathbf{V}, \mathbf{E}_{eot})$ avec $\mathbf{E}_{eot} = \{X \rightarrow Y | X \rightsquigarrow Y \in G_{re} \cup G_o\}$

Grâce à ces graphes, il est possible de contraindre l'apprentissage. Ces contraintes s'appliquent de multiples façons. Nous distinguerons quatre niveaux d'application des contraintes, qui se traduiront par des modifications apportées à différents moments de l'algorithme de PC tel qu'il a été présenté (1 et 2). Le résultat est donné sous la forme des algorithmes 11 et 12. Ci-dessous, nous notons H le graphe modifié lors de l'exécution de l'algorithme.

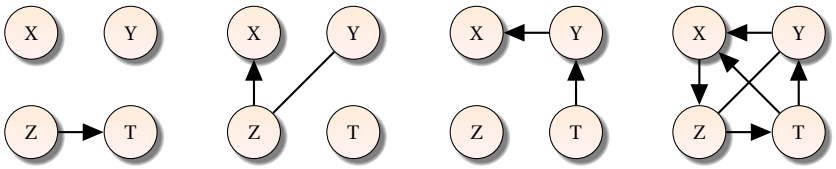
D'abord, le premier niveau concerne l'initialisation du graphe. Dans l'algorithme original de PC, le graphe initial est un graphe complet. Dans le cas présent, il est possible d'élaguer une première fois le graphe initial à l'aide des connaissances. Il est notamment possible d'appliquer les orientations des graphes G_{ai} et G_{eot}. Il aurait été intéressant de supprimer également les arêtes présentes dans G_{au}, mais cela empêcherait de faire des tests d'indépendance dont le résultat permettrait d'orienter des v-structures. Il faut donc attendre la fin de l'étape de construction du squelette pour supprimer ces arêtes, une fois que tous les tests auront été faits.

Ensuite, le deuxième niveau concerne les ensembles de conditionnement utilisés pour faire les tests d'indépendance. Si deux nœuds X et Y sont non-adjacents dans un DAG, alors ils seront indépendants sachant les parents de X ou de Y dans le graphe. Dans la version originale de PC, il n'y a aucune orientation lorsque les tests sont effectués, il n'est donc pas possible de savoir quels sont les parents potentiels des autres variables. Ici, les connaissances nous permettent au moins de savoir quelles sont les variables qui ne seront pas les parents d'une autre. Les auteurs proposent donc de restreindre les ensembles conditionnels, pour un test sur la paire ordonnée (X, Y), aux variables de l'ensemble $\{Z \in adj_H(X) \backslash \{Y\} | X \not\rightsquigarrow Z \in G_{re} \cup G_o \wedge Z \rightarrow X \notin \mathbf{E}_a \wedge Z - X \notin \mathbf{E}_a\}$.

Le troisième niveau concerne les conditions nécessaires à respecter pour modifier le graphe. Ces conditions visent à garder un graphe cohérent avec les connaissances imposées. L'algorithme PC apporte des modifications au graphe durant trois étapes. Il faut donc vérifier qu'à chaque fois qu'une modification est effectuée, celle-ci ne contredit pas les connaissances apportées par l'expert. La première étape est la construction du squelette. Pour PCDC, cette étape est renommée "construction du PDAG" car le graphe obtenu peut être partiellement orienté. Lors de cette étape, des arcs ou des arêtes peuvent être supprimés. On va autoriser la suppression d'une

arête entre X et Y si $X \perp Y \wedge X \rightarrow Y \notin \mathbf{E}_e \wedge X - Y \notin \mathbf{E}_e$. De même, on autorisera la suppression d'un arc entre X et Y si $X \perp Y \wedge X \rightarrow Y \notin \mathbf{E}_e \wedge Y \rightarrow X \notin \mathbf{E}_e \wedge X - Y \notin \mathbf{E}_e$. La deuxième étape de modification est l'orientation des v-structures. Dans l'algorithme PCDC, on peut orienter une v-structure $X - Z - Y$ avec X et Y non-adjacents si $Z \notin Sepset(X,Y) \wedge X \rightarrow Z \notin \mathbf{E}_a \wedge Y \rightarrow Z \notin \mathbf{E}_a \wedge Z \not\rightarrow X \in H \cup G_o \wedge Z \not\rightarrow Y \in H \cup G_o$. Enfin, la troisième étape de modification du graphe est la propagation des orientations. Il s'agit comme précédemment d'orientation d'arêtes, donc les conditions sont les mêmes : si une règle de Meek indique qu'il faut orienter une arête de X vers Y, alors on peut le faire si $X \rightarrow Y \notin \mathbf{E}_a \wedge Y \not\rightarrow X \in H \cup G_o$.

Enfin le dernier niveau concerne un problème de compatibilité entre les connaissances. Même si on suppose qu'elles sont consistantes entre elles, le graphe temporaire qui est contraint par les connaissances peut ne pas être cohérent avec ces dernières, comme le témoigne la figure 2.2, issue de l'article de L. M. d. CAMPOS et CASTELLANO 2007. Ces cas peuvent n'être que transitoires donc il n'est pas prudent de supprimer un arc pour rétablir directement la cohérence car cette suppression pourrait être inutile et dommageable. Les auteurs conseillent d'attendre la fin de l'étape de construction du PDAG pour supprimer les arcs impliqués. L'avantage est que les incohérences peuvent être aisément identifiées, car elles apparaissent si et seulement si le graphe $H \cup G_o$ possède un circuit, et il suffit de supprimer un arc de ce circuit. La dernière modification de l'algorithme PC pour obtenir l'algorithme PCDC consiste donc à supprimer tous les circuits du graphe $H \cup G_o$ à la fin de l'étape de construction du PDAG.



(a) G_e (b) G_a (c) G_o (d) Le graphe initial.

FIGURE 2.2 – Transformer le graphe complet en PDAG ne permet pas d'aboutir à un graphe consistent avec les connaissances imposées.

2.3.4.4 Algorithme GSA

Nous pouvons maintenant passer au second algorithme que nous verrons en détail. Il s'agit d'un algorithme à base de score permettant d'utiliser des connaissances incertaines à propos de présence ou d'absence d'arcs venant de multiples experts. L'heuristique utilisée par l'algorithme est celle de la recherche gloutonne, permettant d'explorer l'espace des DAG. L'insertion des connaissances se fait par le biais d'un score, qui repose sur un score BDeu modifié à cet effet.

De la même manière que dans l'algorithme PCDC, il est nécessaire d'adopter une modélisation pour correctement insérer les connaissances. D'abord, les auteurs sup-

Algorithm 11 Algorithme PCDC - Construction du PDAG

Entrée : un ensemble d'observations **S**, un ensemble de variables **V**, des connaissances sous forme de graphe G_e, G_a et G_o
Sortie : un graphe non orienté G

1: $G \leftarrow (\mathbf{V}, \mathbf{E_c})$
2: $G_{re} = (\mathbf{V}, \mathbf{E}_{re})$ avec $\mathbf{E}_{re} = \{X \rightarrow Y | X \rightarrow Y \in \mathbf{E}_e\} \cup \{Y \rightarrow X | X - Y \in \mathbf{E}_e \wedge X \rightarrow Y \in \mathbf{E_a}\} \cup \{X - Y | X - Y \in \mathbf{E}_e \wedge X \rightarrow Y \notin \mathbf{E}_a \wedge Y \rightarrow X \notin \mathbf{E}_a\}$
3: $G_{au} = (\mathbf{V}, \mathbf{E}_{au})$ avec $\mathbf{E}_{au} = \{X - Y | X - Y \in \mathbf{E}_a\}$
4: $G_{ai} = (\mathbf{V}, \mathbf{E}_{ai})$ avec $\mathbf{E}_{ai} = \{X \rightarrow Y | Y \rightarrow X \in \mathbf{E}_a\}$
5: $G_{eot} = (\mathbf{V}, \mathbf{E}_{eot})$ avec $\mathbf{E}_{eot} = \{X \rightarrow Y | X \rightsquigarrow Y \in G_{re} \cup G_o\}$
6: $H \leftarrow G \cup G_{eot} \cup G_{ai}$
7: Sepset$\leftarrow \emptyset$, $m \leftarrow -1$
8: **Répéter**
9: $m \leftarrow m + 1$
10: **Répéter**
11: Sélectionner une nouvelle paire ordonnée (X, Y) adjacente telle que $|adj(X) \backslash Y| \geq m$
12: **Répéter**
13: Sélectionner un nouvel ensemble $\mathbf{Z} \subseteq adj(X) \backslash Y$ tel que $|\mathbf{Z}| = m \wedge X \not\rightsquigarrow \mathbf{Z} \in G_{re} \cup G_o \wedge \mathbf{Z} \rightarrow X \notin \mathbf{E}_a \wedge \mathbf{Z} - X \notin \mathbf{E}_a$
14: **Si** $X \perp Y | \mathbf{Z} \wedge ((X - Y \in H \wedge X \rightarrow Y \notin \mathbf{E}_e \wedge X - Y \notin \mathbf{E}_e) \vee (X \rightarrow Y \wedge X \rightarrow Y \notin \mathbf{E}_e \wedge Y \rightarrow X \notin \mathbf{E}_e \wedge X - Y \notin \mathbf{E}_e))$ **Alors**
15: Supprimer l'arête $X - Y$ dans H
16: $Sepset_{XY} \leftarrow Sepset_{XY} \cup \mathbf{Z}$
17: **Fin Si**
18: **Jusqu'à ce que** tous les ensembles $\mathbf{Z} \subseteq adj(X)$ tels que $|\mathbf{Z}| = m$ respectant les conditions des connaissances aient été testés ou qu'une indépendance ait été trouvée
19: **Jusqu'à ce que** toutes les paires ordonnées (X, Y) adjacentes telles que $|adj(X) \backslash Y| \geq m$ aient été testées
20: **Jusqu'à ce que** $\forall X \in \mathbf{V}$, $|adj(X)| \leq m$
21: **Pour tout** $X - Y \in \mathbf{E}_a$ tel que $X - Y \in H$ **Faire**
22: Supprimer l'arête $X - Y$ dans H
23: **Fin Pour**
24: **Tant que** $H \cup G_o$ possède des circuits **Faire**
25: Sélectionner un circuit $\mathscr{C}$
26: Supprimer un arc $X \rightarrow Y \in \mathscr{C}$ tel que $X \rightarrow Y \in \mathbf{E}_{ai}$
27: **Fin Tant que**
28: **Retourner** H

posent qu'il y a un ordre fixe entre les paires de nœuds, et entre les nœuds de chaque paire. Par exemple, (X, Y) existe alors que (Y, X) n'existe pas. Cela permet de définir sans ambiguïté les statuts des arêtes entre les nœuds : $\leftarrow$, $\rightarrow$ ou $\not\rightarrow$.

Algorithm 12 Algorithme PCDC - Orientation

Entrée : un ensemble d'observations **S**, un ensemble de variables **V**, des connaissances sous forme de graphe G_e, G_a et G_o
Sortie : un graphe partiellement orienté G

1: **Pour tout** $(X, Y) \in \mathbf{V}^2$ non adjacent avec un voisin Z en commun **Faire**
2: **Si** $Z \notin Sepset(X, Y) \wedge X \rightarrow Z \notin \mathbf{E}_a \wedge Y \rightarrow Z \notin \mathbf{E}_a \wedge Z \not\rightarrow X \in H \cup G_o \wedge Z \not\rightarrow Y \in H \cup G_o$ **Alors**
3: Orienter $X \rightarrow Z \leftarrow Y$
4: **Fin Si**
5: **Fin Pour**
6: Orienter autant d'arêtes que possible par application des règles de Meek en vérifiant à chaque orientation d'arête $X - Y$ que $X \rightarrow Y \notin \mathbf{E}_a \wedge Y \not\rightarrow X \in H \cup G_o$.

7: **Retourner** G

Prenons R experts. Pour $n = |\mathbf{V}|$, il y a $N = n \times (n-1)$ paires de nœuds, notées g_j. L'opinion du i^e expert à propos de la j^e arête est notée $O^i_j \in \{\emptyset, \leftarrow, \rightarrow, \not\rightarrow\}$ où $\emptyset$ désigne l'absence de connaissance émise par l'expert. O^i désigne l'ensemble des connaissances émises par le i^e expert, alors que O_j désigne l'ensemble des connaissances concernant la j^e arête. Pour modéliser l'incertitude des experts, les auteurs proposent d'utiliser trois paramètres :

— γ_1 : la probabilité de détecter une arête existante avec la bonne orientation
— γ_2 : la probabilité de détecter une arête existante avec la mauvaise orientation
— γ_3 : la probabilité de correctement détecter l'absence d'arête entre deux nœuds

Pour un expert i en particulier, on notera γ^i_1, γ^i_2 et γ^i_3 ses paramètres utilisés pour toutes les connaissances qu'il fournit.

Exemple 2.3.2 *Si on considère un expert i avec les paramètres suivants : $\gamma^i_1 = 0,6$, $\gamma^i_2 = 0,1$ et $\gamma^i_3 = 0,8$, alors il est possible de construire la matrice de confusion suivante :*

$$\begin{array}{c} \\ \rightarrow \\ \leftarrow \\ \not\rightarrow \end{array} \begin{array}{c} \begin{array}{ccc} \rightarrow & \leftarrow & \not\rightarrow \end{array} \\ \begin{pmatrix} 0,6 & 0,1 & 0,3 \\ 0,1 & 0,6 & 0,3 \\ 0,1 & 0,1 & 0,8 \end{pmatrix} \end{array}$$

où chaque ligne correspond à un statut possible de la paire de nœuds j et chaque colonne représente l'opinion O^i_j. Notons que lorsque l'expert à tort sur l'absence d'une arête (c'est-à-dire lorsque l'expert pense qu'il y a une arête là où il n'y en a pas en réalité), alors il choisit une orientation entre $\rightarrow$ et $\leftarrow$. Sans plus d'information, ces deux possibilités sont supposées équiprobables, d'où l'égalité entre les deux premières colonnes de la dernière ligne.

Une fois que nous savons comment les connaissances sont exprimées par les experts, il faut concevoir un modèle permettant de les accueillir. Pour cela, il est possible de

représenter l'environnement de l'apprentissage sous la forme d'un réseau bayésien. Dans ce modèle, les auteurs ont représenté les variables suivantes :

— G : la structure
— $\mathbf{S}$: les données
— K : les informations extérieurs qui déterminent la structure
— O : les connaissances des experts
— γ : les incertitudes des experts

En recherchant des indépendances logiques entre ces variables, il est possible de construire à la main des réseaux bayésiens, desquels on pourra tirer d'autres indépendances qui permettront le développement d'un score. Le premier modèle est présenté avec la figure 2.3. Dans ce modèle, les connaissances de chaque expert sont agglomérées. Pour voir les indépendances entre les connaissances des experts, il est nécessaire de détailler une partie du modèle, comme présenté dans la figure 2.4. Cela permet de faire apparaître les variables O^i et γ^i. Enfin, il est possible de détailler encore davantage ce modèle pour cette fois apparaître les variables O^i_j et les variables g_j, ce qui donne cette fois-ci le modèle représenté par la figure 2.5.

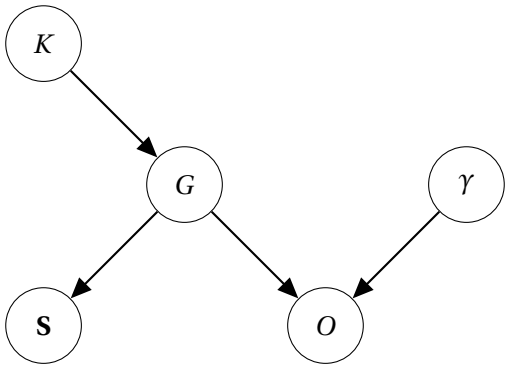


FIGURE 2.3 – Modélisation de l'environnement d'apprentissage incluant la structure, les données et les connaissances des experts.

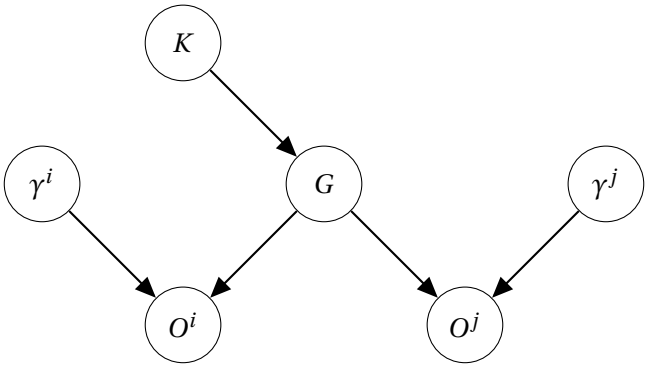


FIGURE 2.4 – Modélisation de l'environnement d'apprentissage incluant la structure et les connaissances des experts.

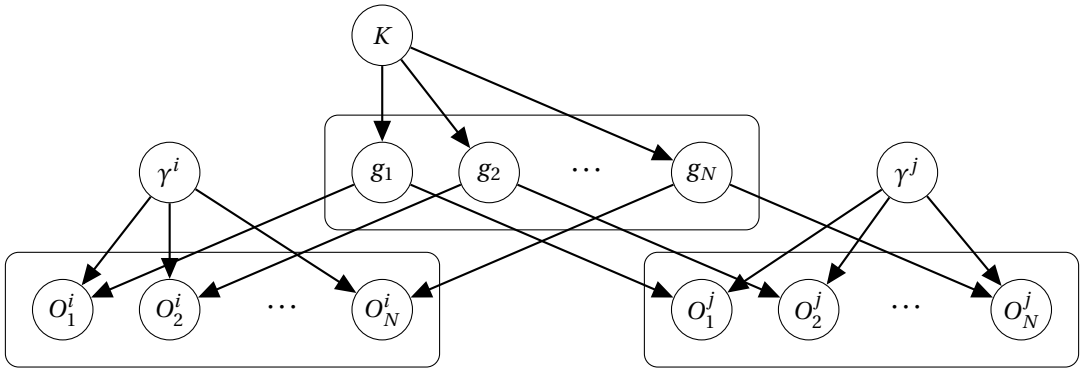


FIGURE 2.5 – Modélisation détaillée de l'environnement d'apprentissage incluant la structure et les connaissances des experts.

À partir de ces graphes, et grâce à la d-séparation, il est possible d'énoncer une liste d'indépendances qui seront nécessaires pour le développement d'un score par la suite. Certaines indépendances peuvent sembler superflues, mais l'objectif est de décrire exactement les indépendances dont nous aurons besoin.

Numéro	Indépendance	Figure
1	$G \perp \gamma$	2.3
2	$\mathbf{S} \perp \gamma \mid G$	2.3
3	$O \perp \mathbf{S} \mid G, \gamma$	2.3
4	$O^i \perp O^j \mid G, \gamma$	2.4
5	$O^j \perp \gamma^i \mid \mathbf{S}, \gamma^j$	2.4
6	$O_x^j \perp O_y^j \mid \mathbf{S}, \gamma^j$	2.5
7	$O_x^j \perp g_y \mid g_x, \gamma^j$	2.5

TABLEAU 2.2 – Quelques indépendances utilisées dans l'algorithme GSA.

Les connaissances des experts ont été modélisées et positionnées dans l'environnement de l'apprentissage. Les auteurs peuvent maintenant développer un score permettant la prise en compte des connaissances dans l'apprentissage. Pour cela, ils proposent de reprendre la méthodologie utilisée pour obtenir le score BDeu. L'objectif est de calculer la probabilité de *a posteriori* de G sachant $\mathbf{S}$, O et γ.

$$P(G|\mathbf{S}, O, \gamma) \propto P(G, \mathbf{S}, O, \gamma) = P(\gamma)P(G|\gamma)P(\mathbf{S}|G, \gamma)P(O|G, \mathbf{S}, \gamma) \tag{2.27}$$

D'après l'indépendance 1 dans la table 2.2, on a $P(G|\gamma) = P(G)$. D'après l'indépendance 2 dans cette même table, on a $P(\mathbf{S}|G, \gamma) = P(\mathbf{S}|G)$. Et d'après l'indépendance 3, on a $P(O|G, \mathbf{S}, \gamma) = P(O|G, \gamma)$. Par ailleurs, $P(\gamma)$ ne dépend pas de la structure du graphe, il peut donc être ignoré du calcul du score. On a donc :

$$P(G, \mathbf{S}, O, \gamma) = P(G)P(\mathbf{S}|G)P(O|G, \gamma) \tag{2.28}$$

$P(G)$ est l'*a priori* sur les structures. Les auteurs le considèrent comme uniforme, ce qui nous permet de l'omettre de l'expression du score. Le terme $P(\mathbf{S}|G)$ est le même que celui calculé dans le score BDeu, on peut donc utiliser le score BDeu directement. Et enfin, pour déterminer le dernier terme, nous allons utiliser les indépendances 4,5,6,7 de la table 2.2.

$$\begin{aligned} P(O|G,\gamma) &= \prod_{i=1}^{R} P(O^i|G,\gamma^1,\cdots,\gamma^i,\cdots,\gamma^R) \\ &= \prod_{i=1}^{R} P(O^i|G,\gamma^i) \\ &= \prod_{j=1}^{N}\prod_{i=1}^{R} P(O^i_j|g_1,\cdots,g_j,\cdots,g_N,\gamma^i) \\ &= \prod_{j=1}^{N}\prod_{i=1}^{R} P(O^i_j|g_j,\gamma^i) \end{aligned} \tag{2.29}$$

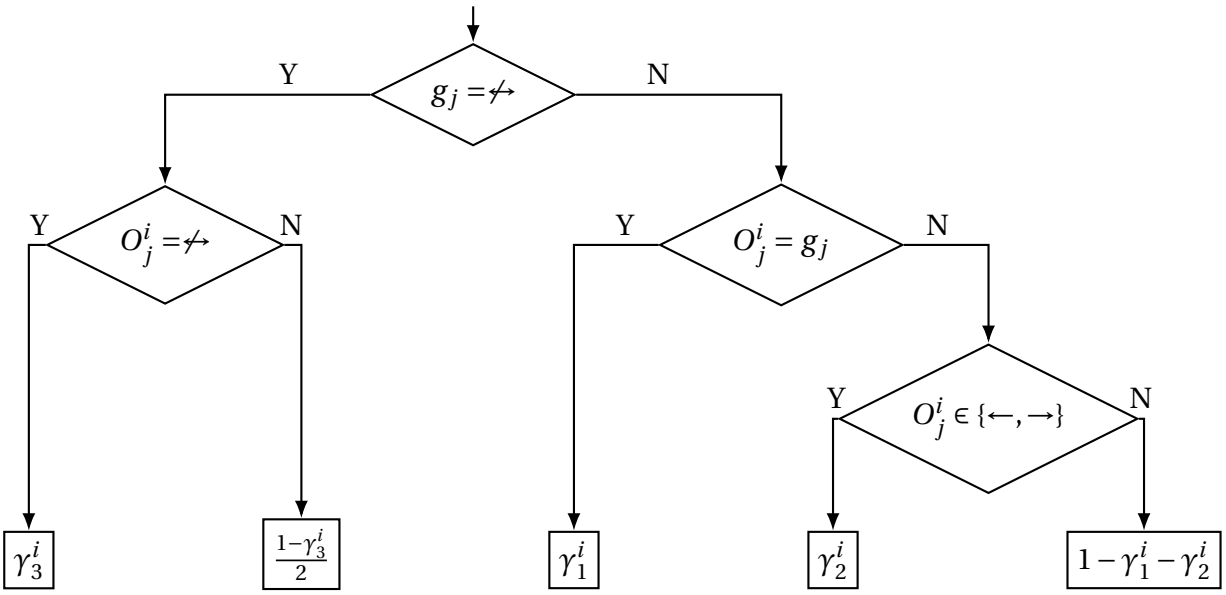


FIGURE 2.6 – Arbre de décision pour calculer $P(O^i_j|g_j,\gamma^i)$ pour $O^i_j \neq \emptyset$.

Le terme $P(O^i_j|g_j,\gamma^i)$ de l'équation 2.29 est déterminé à l'aide de l'arbre de décision de la figure 2.6. Enfin, comme pour le score BDeu original, c'est le logarithme de cet expression est qui utilisé. On obtient alors le score suivant :

$$score_{GSA}(G:\mathbf{S},O,\gamma) = log\,(P(\mathbf{S}|G)) + \sum_{j=1}^{N}\sum_{i=1}^{R} log\left(P(O^i_j|g_j,\gamma^i)\right) \tag{2.30}$$

La partie du score liée aux connaissances est décomposable, tout comme le score

BDeu. Donc ce nouveau score est décomposable. On perd cependant la propriété d'équivalence, mais cela est voulu puisqu'on cherche justement à différencier des DAG en fonction des connaissances, même au sein d'une même classe d'équivalence. Ainsi, $score_{GSA}$ peut s'utiliser de la même façon que le score BDeu.

2.3.5 L'apprentissage de réseaux bayésiens causaux

Nous terminerons ce chapitre en parlant de l'apprentissage des réseaux bayésiens causaux. Pour cela, revenons à un apprentissage sans connaissances d'experts. Pour apprendre un réseau bayésien causal, il faut découvrir les relations causales entre les variables et trouver leurs orientations. Il existe donc deux problèmes dans la littérature. Le problème de **découverte causale** consiste à chercher l'existence et, le cas échéant, l'orientation de la relation causale. Lorsque l'existence de la relation causale est présupposée et que seule l'orientation de celle-ci est recherchée, on parle alors d'**inférence causale**[2].

Nous avons déjà vu des algorithmes permettant de faire de l'apprentissage causal. C'est par exemple le cas de l'algorithme PC, qui renvoie à la fin de son exécution un CPDAG, et dont les arcs portent une signification causale. L'algorithme PC est donc capable, pour une partie des paires de variables seulement, de montrer qu'une relation causale existe et d'en donner l'orientation, ce qui correspond au problème de découverte causale.

L'apprentissage d'un réseau bayésien dans un contexte causal est un problème plus difficile que dans un contexte non causal. En effet, sans causalité, tous les DAG appartenant à la classe d'équivalence d'un CPDAG sont interchangeables. Il est donc possible de choisir arbitrairement l'orientation de chaque arête du CPDAG, à condition bien-sûr de rester dans la bonne classe d'équivalence. Cependant, ceci n'est plus possible dans un contexte causal, puisque chaque arête doit être orientée minutieusement, selon la relation causale existante dans le système. L'orientation de ces arêtes, qui sous certaines conditions décrivent des relations causales, relève donc du problème d'inférence causale.

Dans cette sous-section, nous tenterons d'abord d'expliquer pourquoi un algorithme comme PC permet de trouver des relations causales. Nous verrons ensuite d'autres algorithmes d'inférence causale, ainsi que leurs contraintes.

2.3.5.1 L'algorithme PC et l'inférence causale

Rappelons que si l'algorithme PC peut apprendre un réseau bayésien classique, c'est grâce à la propriété locale de Markov (1.2.4). Cette propriété dit que pour un réseau bayésien $\mathscr{B}$, une variable est indépendante de ses non-descendants sachant ses parents. Cette propriété relie donc la notion de parent et d'indépendance, et c'est sur cela que s'appuie l'algorithme PC, ainsi que beaucoup d'autres algorithmes sous contraintes.

2. Le problème d'inférence causale désigne aussi parfois la résolution de requêtes sur des modèles causaux (PEARL 2010)

Mais pour apprendre la structure d'un réseau bayésien causal, cette propriété n'est pas suffisante. Dans un tel modèle, les parents doivent être considérés comme les causes directes d'une variable, et cela nécessite deux hypothèses.

La première hypothèse est celle de suffisance causale, qu'on a déjà vu pour l'apprentissage de réseaux bayésiens classiques. Ici, cette hypothèse va nous permettre de nous placer dans le cadre spécifique des modèles markoviens. Rappelons que les modèles structurels impliquent des termes d'erreurs pour chaque variable observée. La première hypothèse stipule que ces termes d'erreurs sont indépendants les uns des autres et donc qu'on utilise un modèle markovien. En d'autres termes, on peut considérer les termes d'erreurs comme du bruit dans les distributions de probabilité des variables. Les modèles markoviens peuvent alors être assimilés à des réseaux bayésiens ayant comme structure un DAG.

La deuxième hypothèse permet de justifier la signification causale obtenue par l'algorithme. Elle se résume par la maxime de Reichenbach (REICHENBACH 1956) : "Il n'y a pas de corrélation sans causalité". Il faut comprendre ici que s'il existe une dépendance marginale et conditionnelle entre deux variables, alors soit l'une est la cause de l'autre, soit il existe une troisième, cachée, qui est la cause commune des deux premières. Dans un modèle structurel, une variable cachée qui est parente de deux variables observées peut être vue comme une relation de dépendance entre les termes d'erreurs de deux variables observées. Or d'après la première hypothèse, cela n'est pas possible. On peut donc reformuler cette hypothèse de la manière suivante : si X et Y sont dépendants marginalement et conditionnellement, alors X est la cause de Y ou Y est la cause de X.

Ainsi, lorsque que PC trouve un arc sortant d'une variable X vers une variable Y, on peut considérer que X est la cause de Y. Ce raisonnement fonctionne pour l'algorithme PC, mais il peut également fonctionner pour presque tous les algorithmes sous contraintes, du moment qu'ils permettent de trouver des orientations. En effet, ils ont comme point commun de s'appuyer sur les dépendances et indépendances pour trouver les orientations entre les variables, et grâce aux indépendances, ces orientations peuvent être considérées comme causales.

Cela permet notamment de reformuler la propriété locale de Markov en **propriété causale de Markov**, où $Ca(X)$ désigne les causes de X :

Théorème 2.3.3 (Propriété causale de Markov (PEARL et VERMA 1991)) *Soit $B = (G, \Theta)$ un réseau bayésien causal avec $G = (\mathbf{V}, \mathbf{E})$ le graphe et Θ l'ensemble des CPD. Pour une variable $X \in \mathbf{V}$, notons $Ca(X)$ l'ensemble de ces causes directes, et $ND(X)$ l'ensemble des **non-descendants** de X (X et ses causes directes exclus). Alors B encode dans sa structure les indépendances suivantes :*

$$\forall X \in \mathbf{V}, X \perp ND(X) | Ca(X) \tag{2.31}$$

2.3.5.2 Les algorithmes d'inférence causale

En utilisant un algorithme sous contraintes, on a vu qu'il n'était pas toujours possible d'orienter toutes les arêtes. Et dans la plupart des cas, cet état de fait ne changera pas. La plupart seulement, car il existe effectivement certain cas où l'orientation de l'arête entre deux variables peut être trouvée, uniquement avec des données observationnelles. C'est notamment le cas des modèles **ANM**, pour *Additive Noise Model*, ou Modèle à Bruit Additif en français.

Ces modèles se placent dans le cadre des modèles structurels. Rappelons une fois de plus qu'un modèle structurel représente les relations entre les variables par des fonctions qui ont comme arguments les parents (ou ici les causes) et un bruit : $X = f(Y, e_X)$. Dans les modèles ANM, la relation peut s'écrire sous la forme suivante $X = f(Y) + e_X$, avec e_X une variable indépendante de Y. C'est l'addition de e_X pour obtenir X qui justifie les termes de "bruit additif".

Avant de présenter les méthodes, il est nécessaire de parler d'une hypothèse importante : l'indépendance de la cause et du mécanisme de l'effet. Cette hypothèse stipule que la distribution de la cause (par exemple $P(Y)$ dans les paragraphes précédents) est indépendante de la distribution de l'effet sachant la cause ($P(X|Y)$). Il est important de noter qu'il ne s'agit pas d'indépendance entre des variables aléatoires, mais bien entre des distributions de probabilité, ce qui a une sémantique bien différente. Cette hypothèse est notamment nécessaire pour le modèle ANM, puisque le bruit, qui fait partie du mécanisme de l'effet, doit être indépendant de la cause.

Comme dit précédemment, faire l'hypothèse d'un modèle ANM permet de rendre identifiable une partie des orientations. Cela a d'abord permis l'émergence de méthodes pour les données continues (SHIMIZU, P. O. HOYER, HYVÄRINEN et al. 2006), mais ces méthodes ont été rapidement adaptées pour les données discrètes. C'est par exemple le cas de la méthode présentée par PETERS, JANZING et SCHÖLKOPF 2010. Dans leur article, les auteurs s'intéressent à l'orientation d'une arête entre deux variables X et Y. Ils envisagent alors deux modèles :

— le modèle où X est la cause de Y, et où on a donc : $Y = f_{XY}(X) + e_Y$, avec $X \perp e_Y$
— le modèle où Y est la cause de X, et où on a donc : $X = f_{YX}(Y) + e_X$, avec $Y \perp e_X$

Pour déterminer les fonctions f_{XY} et f_{YX}, les auteurs proposent d'utiliser une régression. On peut ainsi générer des données pour les bruits e_X et e_Y, grâce aux fonctions trouvées. Pour chaque ligne l de la base de données, on a :

$$e_Y^{(l)} = Y^{(l)} - f_{XY}(X^{(l)}) \text{ et } e_X^{(l)} = X^{(l)} - f_{YX}(Y^{(l)}) \tag{2.32}$$

Il est ensuite possible de faire des tests d'indépendance pour vérifier le sens dans lequel le bruit est indépendant de la cause. Ici, l'implémentation est libre. Les auteurs proposent d'utiliser le test du χ^2. Grâce aux résultats d'indépendance, il est possible de conclure de la manière suivante :

— Si $e_X \not\perp Y$ et $e_Y \perp X$, alors on peut inférer "*X est la cause de Y*"
— Si $e_X \perp Y$ et $e_Y \not\perp X$, alors on peut inférer "*Y est la cause de X*"
— Si $e_X \not\perp Y$ et $e_Y \not\perp X$ ou si $e_X \perp Y$ et $e_Y \perp X$, alors il n'est pas possible de décider.

D'autres méthodes similaires ont été présentées dans la littérature. Celles-ci vont garder le même schéma, c'est-à-dire chercher des fonctions par régression, puis chercher l'indépendance entre les bruits et les causes potentielles. C'est dans la manière de chercher les indépendances qu'on va trouver de la diversité. Par exemple, les auteurs de l'article BUDHATHOKI et VREEKEN 2018 proposent d'utiliser l'entropie de Shannon (SHANNON 1948). Cette grandeur est donnée pour une variables aléatoires X par la formule 2.33. Et lorsque que ces deux variables aléatoires sont indépendantes, on trouve $H(X,Y) = H(X) + H(Y)$.

$$H(X) = -\sum_{X} P(X) log(P(X)) \tag{2.33}$$

Les auteurs montrent également le théorème 2.3.4. Ainsi, pour tester l'orientation d'une relation causale, il suffit de tester l'inégalité 2.34. En cas d'égalité, il n'est pas possible de décider.

Théorème 2.3.4 (BUDHATHOKI et VREEKEN 2018) *Si des observations, issues d'une distribution jointe $P(X,Y)$ de deux variables aléatoires X et Y, sont induites par un modèle ANM avec $X \to Y$, alors :*

$$H(X) + H(e_Y) < H(Y) + H(e_X) \tag{2.34}$$

On voit dans les conclusions ci-dessus qu'il reste encore une possibilité que l'on ne parvienne pas à trouver l'orientation. En effet, ces méthodes ne sont pas exactes. Par exemple, les fonctions f_{XY} et f_{YX} sont trouvées par régression. De plus, on peut pointer du doigt les hypothèses assez restrictives qui ne permettent pas d'appliquer ces méthodes à beaucoup de cas, à commencer par les données catégoriques (lorsque les valeurs ne sont pas numériques).

D'autres méthodes légèrement moins restrictives sont apparues, relaxant l'hypothèse des modèles ANM. Dans l'article de BUDHATHOKI et VREEKEN 2017, les auteurs se contentent de l'hypothèse d'indépendance de la cause et du mécanisme. Une des interprétations possibles pour cette indépendance entre deux distributions se fait par le biais de la **complexité de Kolmogorov**.

Définition 2.3.5 (Complexité de Kolmogorov (KOLMOGOROV 1965)) *Pour un objet s (un nombre, une image, une chaîne de caractères par exemple), la complexité de Kolmogorov, notée $K(s)$, est la longueur en bit du plus petit algorithme permettant de produire l'objet.*

La complexité de Kolmogorov est une grandeur théorique issue de la théorie de l'information. Elle n'est pas calculable, mais permet de rendre compte de la complexité d'un objet par rapport à un autre. La complexité de Kolmogorov a été adaptée aux distributions de probabilité (GRÜNWALD et al. 2008).

Définition 2.3.6 (Complexité de Kolmogorov d'une distribution de probabilité) *Soit une distribution de probabilité P, la complexité de Kolmogorov de P, notée $K(P)$, est la longueur, en bits, du plus petit algorithme permettant de produire la probabilité $P(x)$ avec une précision q en ayant en entrée les valeurs x et q.*

L'indépendance de deux variables a notamment en conséquence le théorème 2.3.5 en termes de complexité de Kolmogorov.

Théorème 2.3.5 (STEGLE, JANZING, K. ZHANG et al. 2010) *Soit X et Y deux variables aléatoires. Si X est la cause de Y, alors :*

$$K(P(X)) + K(P(X|Y)) \leq K(P(Y)) + K(P(X|Y)) \tag{2.35}$$

Comme dit précédemment, la complexité de Kolmogorov n'est pas calculable, il faut donc trouver une approximation pour l'utiliser. Les auteurs de BUDHATHOKI et VREEKEN 2017 proposent la **complexité stochastique**, qui est une variation du principe de MDL (vu plus tôt en tant que score). Sans rentrer dans les détails des calculs, pour inférer l'orientation de la relation causale, il suffit de tester l'inégalité 2.35 en utilisant la complexité stochastique au lieu de la complexité de Kolmogorov, avec une impossibilité de prendre une décision s'il y a une égalité.

D'autres auteurs (F. LIU et CHAN 2016) proposent de directement mesurer l'indépendance entre la cause et le mécanisme en considérant $P(X)$, $P(Y|X)$, $P(Y)$ et $P(X|Y)$ comme des variables aléatoires et en calculant la distance de corrélation entre $P(X)$ et $P(Y|X)$ et entre $P(Y)$ et $P(X|Y)$. Plus la distance de corrélation est grande, moins les variables sont indépendantes. Ainsi, on infère l'orientation de la relation pour le couple cause/mécanisme de l'effet qui a la plus petite distance de corrélation.

Pour conclure sur cette sous-partie, on a vu que les algorithmes sous contraintes étaient capables de nous apporter des résultats fondés, mais pas pour toutes les arêtes d'un graphe. Pour les arêtes restantes, des solutions d'inférences causales existent mais comportent des défauts. D'abord, il est souvent nécessaire d'être sûr de l'existence de la relation causale entre les deux variables. De plus, ces méthodes ne sont pas toujours adaptées à tous les types de données, et ne sont pas non plus toujours fiables.

2.4 Conclusion

Dans ce chapitre, nous avons étudié l'apprentissage de réseaux bayésiens à l'aide de données discrètes. Nous avons vu que cet apprentissage se déclinait en deux parties, l'une concernant la structure et l'autre les paramètres. Il est apparu que l'apprentissage de la structure était la partie la plus compliquée. Pour ce problème, de nombreuses approches ont été développées en plus de vingt ans. Nous avons vu que la majorité de ces approches pouvaient être réparties dans trois catégories principales : les méthodes sous contraintes, les méthodes à base de score et les méthodes hybrides. Pour chaque

catégorie, des exemples ont été donnés et ont permis de donner une idée précise de la richesse de la littérature sur ce sujet.

Nous avons également vu que l'apprentissage de la structure pouvait être faite avec l'appui de connaissances d'experts en plus des données. Une grande diversité de connaissances possibles a pu être observée. Ces connaissances peuvent être certaines, ou au contraire posséder un paramètre d'incertitude. Deux exemples d'algorithmes utilisant des connaissances d'experts ont pu être présentés en détail : PCDC, qui est un algorithme sous contraintes utilisant des connaissances certaines, et GSA qui est un algorithme à base de score utilisant des connaissances incertaines.

Enfin, nous avons observé que l'apprentissage de réseau bayésien causal était un exercice plus difficile que l'apprentissage de réseau bayésien classique. Les algorithmes classiques permettent rarement d'apprendre l'intégralité des orientations des relations causales. Nous avons aussi vu que des algorithmes avaient été mis au point pour faciliter l'apprentissage, mais que ceux-ci possèdent des contraintes fortes. Cela montre également la difficulté de l'apprentissage causal.

Maintenant que nous connaissons l'état général de la littérature à propos des réseaux bayésiens et de l'apprentissage de leurs structures, nous pouvons passer aux contributions de cette thèse. Les deux prochains chapitres présenteront chacun un nouvel algorithme d'apprentissage de structure utilisant des connaissances d'experts incertaines. Le premier algorithme s'intéresse à l'apprentissage de réseaux bayésiens classiques alors que le deuxième s'intéresse plus particulièrement à l'apprentissage causal.

3 Algorithme d'apprentissage sous contraintes de structure de réseau bayésien à l'aide de connaissances incertaines

Sommaire

3.1 Introduction . . . 93
3.2 Hypothèses . . . 94
3.2.1 Hypothèses d'apprentissage . . . 94
3.2.2 Les connaissances . . . 95
3.3 Test d'indépendances . . . 95
3.3.1 Modèle pour le test marginal . . . 96
3.3.2 Développement du test d'indépendance marginale . . . 99
3.3.3 Modèle pour le test conditionnel . . . 105
3.3.4 Développement du test d'indépendance conditionnelle . . . 107
3.4 Algorithme . . . 109
3.4.1 Problèmes rencontrés et résolus . . . 109
3.4.2 Pseudo-code . . . 110
3.5 Expérimentations . . . 112
3.5.1 Paramètres expérimentaux . . . 112
3.5.2 Résultats . . . 114
3.6 Conclusion . . . 120

3.1 Introduction

Nous avons vu dans le chapitre précédent que des connaissances d'experts pouvaient améliorer le résultat de l'apprentissage d'un réseau bayésien. Il existe de nombreuses sortes de connaissances possibles, et encore plus de façons d'intégrer ces connaissances à l'apprentissage. Les connaissances certaines sont les plus simples à utiliser, puisqu'elles agissent comme des contraintes strictes. Cependant, les connaissances incertaines possèdent un paramètre en plus : leur incertitude. Ce paramètre supplémentaire les rend mécaniquement plus générales.

L'utilisation de l'incertitude provient de facto de l'existence de connaissances fausses. En effet, si l'erreur est impossible dans les connaissances, alors il n'y a pas

d'intérêt de prendre en compte l'incertitude. Lorsqu'il s'agit de prendre en compte l'incertitude, il est donc surtout question de gérer l'erreur, autant du côté des données que de l'expert.

Les erreurs dans les connaissances proviennent essentiellement d'une circulation imparfaite de l'information. En effet, l'expert connaît le modèle étudié, et cette connaissance lui a été transmise sous forme d'information venant soit directement du modèle étudié (l'expert étudie alors lui-même le modèle en question), soit par d'autres sources de connaissances (d'autres experts ou un livre). Dans tous les cas, l'information doit circuler jusqu'à l'expert, puis entre l'expert et la personne (ou l'interface) qui intégrera les connaissances à l'apprentissage.

Si la circulation de l'information est parfaite, alors aucune erreur ne peut apparaître. On se retrouve alors avec des connaissances sans erreur, et il n'est pas nécessaire de prendre en compte l'incertitude. Mais dans la réalité, la circulation de l'information est imparfaite, et des erreurs peuvent apparaître lors de la transmission de l'information d'un acteur à un autre.

Il est donc intéressant d'utiliser des connaissances incertaines. Comme on l'a vu plus tôt, ces connaissances sont principalement utilisées dans le cadre d'algorithmes à base de score, et jamais à notre connaissance avec des algorithmes sous contraintes. Or les algorithmes à base de score souffrent des problèmes de maximum locaux, problèmes que ne connaissent pas les algorithmes sous contraintes.

Le principal problème des algorithmes sous contraintes vient de la faiblesse des tests d'indépendances. Si la base de données est petite ou que les données sont de mauvaise qualité, alors des tests d'indépendance donneront le mauvais résultat. Mais c'est justement là que les connaissances peuvent être utiles. En apportant plus d'information aux tests d'indépendances, on peut diminuer la quantité d'erreurs, et donc améliorer la qualité des graphes appris.

C'est pour l'ensemble de ces raisons que nous proposons dans cette partie un algorithme d'apprentissage sous contraintes de structure de réseau bayésien utilisant des connaissances d'experts incertaines. Dans la suite de ce chapitre, nous aborderons d'abord les hypothèses ainsi que la nature précise des connaissances utilisées. Puis nous décrirons la façon exacte dont les connaissances sont intégrées directement au test d'indépendance. Nous verrons ensuite comment nous avons adapté l'algorithme PC pour pouvoir correctement utiliser ce nouveau test d'indépendance. Enfin, nous verrons les résultats d'une série d'expérimentations venant confirmer l'efficacité de l'algorithme.

3.2 Hypothèses

3.2.1 Hypothèses d'apprentissage

Commençons avec les hypothèses faites dans le cadre de cet algorithme. Ce dernier se place dans la catégorie des algorithmes sous contraintes. On fait donc les deux hypothèses classiques de fidélité et de suffisance causale. Rappelons que la première

assure que la distribution sur laquelle nous travaillons admet l'existence d'une P-map. La deuxième nous permet d'être sûr qu'il n'existe pas de variable cachée qui rendrait dépendantes deux variables sans que cette dépendance puisse être causalement expliquée.

Nous faisons également l'hypothèse de données discrètes, nous supposerons donc que les données sont de la bonne nature.

Enfin, la dernière hypothèse porte sur la quantité d'indépendance que l'on trouve dans la structure des réseaux appris. Si on note $\mathbb{I}$ la variable aléatoire binaire représentant l'indépendance ($\mathbb{I} = 1$) ou la dépendance ($\mathbb{I} = 0$) de deux variables quelconques du réseau bayésien, la distribution de probabilité de $\mathbb{I}$ représente les probabilités *a priori* que deux variables quelconques soient dépendantes ou indépendantes. Pour notre algorithme, nous nous placerons dans le cadre de réseaux bayésiens "creux", pour lesquels les indépendances sont plus nombreuses que les dépendances. Cela se manifeste par des structures avec moins d'arêtes. Nous traduisons ce caractère creux par une probabilité $P(\mathbb{I} = 1) = 0,95$.

3.2.2 Les connaissances

Comme dit précédemment, dans la littérature, il est possible de trouver des algorithmes utilisant des connaissances de nombreuses natures différentes. Il est donc important de décrire la nature des connaissances utilisées. Nous utiliserons des connaissances incertaines portant sur les dépendances ou indépendances marginales ou conditionnelles entre les variables aléatoires du réseau bayésien. En d'autres termes, soit deux variables aléatoires $\mathbb{X}$ et $\mathbb{Y}$, chaque expert pourra nous donner son opinion sur l'état de dépendance ou d'indépendance marginale entre ces deux variables, ou conditionnellement à un ensemble $\mathbb{Z}$. En tant que connaissances incertaines, nous ferons intervenir un paramètre d'incertitude associé à chaque connaissance disponible. Nous n'avons pas de connaissances pour tous les triplets $(\mathbb{X}, \mathbb{Y}, \mathbb{Z})$, donc ce n'est qu'une partie des indépendances et dépendances pour lesquelles nous aurons des connaissances.

Enfin, un autre paramètre que l'incertitude pourra être utilisé : il s'agit de la confiance en l'expert. En effet, nous verrons que nous pourrons également modéliser et faire varier la confiance que nous accordons à l'expert à propos de sa capacité à donner des connaissances correctes.

3.3 Test d'indépendances

L'objectif principal de cet algorithme sous contraintes est de prendre en compte des connaissances incertaines. Or les algorithmes sous contraintes tentent de construire la structure à l'aide de l'ensemble d'indépendances qu'elle doit satisfaire. Les indépendances conditionnelles sont donc au cœur des algorithmes sous contraintes, et c'est pourquoi nous avons intégré les connaissances directement dans le test d'indépendance.

Pour intégrer les connaissances directement au test d'indépendance, nous allons commencer par construire un modèle dans lequel les connaissances et les données existent. Ce modèle nous permettra de voir le lien entre les connaissances et les données. Il prend la forme d'un réseau bayésien construit manuellement, ce qui nous permet de modéliser et prendre en compte les indépendances entre les différents composants du modèle.

Grâce au réseau bayésien ainsi construit, nous pourrons développer un test d'indépendance qui s'appuie sur un rapport de vraisemblances, de manière similaire au test G^2. Cette vraisemblance inclura les connaissances des experts sous la forme de probabilité. Les calculs montrent que ce nouveau test peut en fait s'appuyer largement sur le résultat d'un test G^2 classique, avec la manifestation des connaissances sous la forme d'un facteur pouvant altérer l'issue du test.

Dans la suite de cette section, nous verrons comment nous avons construit le modèle. Nous verrons ensuite comment nous avons mené les développements nécessaires pour obtenir le test d'indépendance marginale, puis le test d'indépendance conditionnelle. À la fin de ce chapitre, nous analyserons les résultats d'expérimentations faites avec ce nouvel algorithme.

3.3.1 Modèle pour le test marginal

Pour modifier correctement le test d'indépendance, nous avons modélisé ce dernier sous la forme d'un réseau bayésien. Commençons par en détailler la structure et les paramètres.

Soient $\mathbb{X}$ et $\mathbb{Y}$ deux variables aléatoires, $dom(\mathbb{X}) = \{x_1, \ldots, x_r\}$ et $dom(\mathbb{Y}) = \{y_1, \ldots, y_s\}$ leurs domaines respectifs et $P(\mathbb{X}, \mathbb{Y})$ leur distribution jointe. Notons $\mathbf{S} = \{(XY)^{(n)}\}_{n=1}^{N}$ un ensemble de N variables aléatoires distribuées selon $P(\mathbb{X}, \mathbb{Y})$. Ainsi, une base de données représentatives des variables $\mathbb{X}$ et $\mathbb{Y}$ est une observation de toutes les variables de $\mathbf{S}$.

L'indépendance ou la dépendance des variables $\mathbb{X}$ et $\mathbb{Y}$ est représentée par une variable aléatoire $\mathbb{I}$ qui vaut 1 en cas d'indépendance et 0 en cas de dépendance. Nous pouvons remarquer que lorsque $\mathbb{I}$ est inconnu, toutes les variables $(XY)^{(n)}$ sont mutuellement indépendantes. A l'inverse, si $\mathbb{I}$ est connu, ces variables deviennent mutuellement dépendantes. En effet, sachant $\mathbb{I} = 1$, si on observe un sous-ensemble $\mathbf{S}' \subset \mathbf{S}$ de variables $(XY)^{(n)}$ dont les valeurs montreraient une dépendance entre les variables $\mathbb{X}$ et $\mathbb{Y}$, il est très probable que le reste des variables dans $\mathbf{S} \backslash \mathbf{S}'$ permettent d'aboutir à l'indépendance entre $\mathbb{X}$ et $\mathbb{Y}$. En représentant cette relation sous la forme d'un réseau bayésien, on obtient la figure 3.1, présentant donc une v-structure.

Définissons maintenant les tables de probabilités de ce réseau bayésien. D'abord, notons $P(\mathbb{X} = x, \mathbb{Y} = y) = \theta_{(xy)}$ avec $x \in \{x_1, \ldots, x_r\}$ et $y \in \{y_1, \ldots, y_s\}$. Pour tout $x \in \{x_1, \ldots, x_r\}$, notons $\theta_{(x.)} = \sum_{y=y_1}^{y_s} \theta_{(xy)}$ et, pour tout $y \in \{y_1, \ldots, y_s\}$, notons $\theta_{(.y)} = \sum_{x=x_1}^{x_r} \theta_{(xy)}$. Cela nous permet de définir la distribution $P(\mathbb{I} | (XY)^{(1)}, \ldots, (XY)^{(N)})$. En effet, grâce à

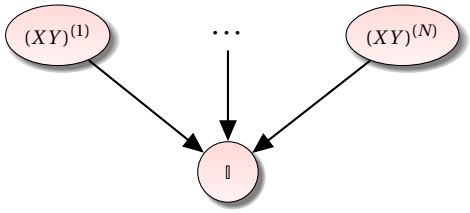


FIGURE 3.1 – Le réseau bayésien soulignant la relation entre $(XY)^{(n)}$ et la variable $\mathbb{I}$.

la formule de Bayes, on a :

$$P(\mathbb{I}|(XY)^{(1)},\dots,(XY)^{(N)}) = \frac{P((XY)^{(1)},\dots,(XY)^{(N)}|\mathbb{I}) \times P(\mathbb{I})}{P((XY)^{(1)},\dots,(XY)^{(N)})}$$

Par d-séparation entre les variables $(XY)^{(n)}$, nous obtenons l'égalité suivante pour le numérateur : $P((XY)^{(1)},\dots,(XY)^{(N)}) = \prod_{n=1}^{N} P((XY)^{(n)})$.

Concernant le numérateur, nous pouvons approcher $P((XY)^{(1)},\dots,(XY)^{(N)}|\mathbb{I})$ avec $\prod_{n=1}^{N} P((XY)^{(n)}|\mathbb{I})$. Notez que cette approximation correspond précisément à celle utilisée dans les tests χ^2 et G^2 classiques. Ainsi, nous trouvons l'approximation suivante :

$$P(\mathbb{I}|(XY)^{(1)},\dots,(XY)^{(N)}) \approx P(\mathbb{I}) \times \prod_{n=1}^{N} \frac{P((XY)^{(n)}|\mathbb{I})}{P((XY)^{(n)})} \tag{3.1}$$

Définissons également $P((XY)^{(n)} = (xy)^{(n)}) = \theta_{(xy)^{(n)}}$. Par abus de notation, nous pouvons également noter $\theta_{(x^{(n)}.)} = \sum_{y=y_1}^{y_s} \theta_{(x^{(n)}y)}$ et $\theta_{(.y^{(n)})} = \sum_{x=x_1}^{x_r} \theta_{(xy^{(n)})}$. Ainsi, puisque $\mathbb{I} = 1$ signifie que $\mathbb{X}$ et $\mathbb{Y}$ sont indépendants, nous trouvons que $P((XY)^{(n)} = (xy)^{(n)}|\mathbb{I} = 1) = \theta_{(x^{(n)}.)} \times \theta_{(.y^{(n)})}$. En reprenant l'équation 3.1, on a :

$$P(\mathbb{I} = 1|(XY)^{(1)} = (xy)^{(1)},\dots,(XY)^{(N)} = (xy)^{(N)}) \approx P(\mathbb{I} = 1) \times \prod_{n=1}^{N} \frac{\theta_{(x^{(n)}.)} \times \theta_{(.y^{(n)})}}{\theta_{(xy)^{(n)}}}$$

Soit $\mathbf{s} = \{(xy)^{(n)}\}_{n=1}^{N}$ un échantillon d'observations des variables $\{(XY)^{(n)}\}_{n=1}^{N}$. Notons N_{xy} le nombre de variables $(XY)^{(n)}$ dont les valeurs sont exactement égales à (xy) dans $\mathbf{s}$. Alors :

$$P(\mathbb{I} = 1|(XY)^{(1)} = (xy)^{(1)},\dots,(XY)^{(N)} = (xy)^{(N)}) \approx P(\mathbb{I} = 1) \times \prod_{xy} \left(\frac{\theta_{(x.)} \times \theta_{(.y)}}{\theta_{(xy)}}\right)^{N_{xy}}$$

Ainsi, puisque $P(\mathbb{I} = 0|(XY)^{(1)},\dots,(XY)^{(N)}) + P(\mathbb{I} = 1|(XY)^{(1)},\dots,(XY)^{(N)}) = 1$, on a :

$$P(\mathbb{I} = i|(xy)^{(1)},\dots,(xy)^{(N)}) \approx \begin{cases} 1 - P(\mathbb{I} = 1) \times \prod_{xy} \left(\frac{\theta_{(x.)} \times \theta_{(.y)}}{\theta_{(xy)}}\right)^{N_{xy}} & \text{si } i = 0 \\ P(\mathbb{I} = 1) \times \prod_{xy} \left(\frac{\theta_{(x.)} \times \theta_{(.y)}}{\theta_{(xy)}}\right)^{N_{xy}} & \text{si } i = 1. \end{cases} \tag{3.2}$$

Il est simple de voir que ces équations ne donnent aucune contrainte sur la valeur de $P(\mathbb{I}=1)$. Cette probabilité doit refléter nos connaissances initiales à propos de l'indépendance de $\mathbb{X}$ et $\mathbb{Y}$. C'est ici que nous allons utiliser l'hypothèse discutée plus tôt et adopter la valeur $P(\mathbb{I}=1)=0.95$.

Maintenant, supposons que K experts apportent leurs connaissances à propos de l'état de $\mathbb{I}$ et donc l'indépendance entre $\mathbb{X}$ et $\mathbb{Y}$, avec des assertions e_k telles que "Je pense que $\mathbb{X}$ et $\mathbb{Y}$ sont indépendants" ou "Je crois que $\mathbb{X}$ influence directement $\mathbb{Y}$". Soit O_k une variable aléatoire qui représente l'opinion de l'expert k (ici, $O_k=0$ et $O_k=1$ signifient respectivement une dépendance et une indépendance). Ainsi, e_k est une observation de la variable O_k et $P(e_k|O_k)$ est un vecteur de taille $|dom(O_k)|=2$ avec une valeur égale à 1 et l'autre égale à 0. En effet, dans l'expression $P(e_k|O_k)$, seule O_k est une variable aléatoire, e_k n'étant qu'une observation, une information et, à ce titre, e_k n'a pas de dimension. Ainsi $P(e_k|O_k=0)=1$ s'interprète comme la probabilité que le $k$ème expert nous ait informé qu'il penche pour une dépendance entre $\mathbb{X}$ et $\mathbb{Y}$ sachant qu'il pense qu'il y a effectivement une dépendance. Évidemment, les experts ne sont pas forcément certains de leurs expertises et ils accordent donc une certaine confiance dans leur jugement. Nous supposons ici qu'ils sont capables d'estimer cette confiance dans leur identification de l'état de $\mathbb{I}$. Notons γ_k cette confiance. Par exemple, l'expert k peut estimer être sûr à 70% que son jugement est correct. Dans ce cas, $\gamma_k=0.7$ et donc $P(O_k=i|\mathbb{I}=i)=\gamma_k$ pour $i\in\{0,1\}$. Les opinions des experts sont fondées sur leurs expériences avec $\mathbb{X}$ et $\mathbb{Y}$. Par conséquent, elles ne sont pas indépendantes. Cependant, du moment que la valeur de $\mathbb{I}$ est connue, les connaissances d'un ou plusieurs experts n'apportent plus aucune information supplémentaire sur les opinions des autres experts. D'après cela, les relations entre $\mathbb{I}$ et les variables O_k peuvent être encodées par un réseau bayésien contenant uniquement des arcs $\mathbb{I}\rightarrow O_k$. On pourrait rajouter ces arcs directement dans le réseau bayésien de la figure 3.1 afin de tenir compte du savoir d'experts dans l'exploitation statistique des échantillons. Toutefois, les distributions $P(O_k|\mathbb{I})$, autrement dit les valeurs γ_k, ne sont pas nécessairement aisées à estimer par les experts. En effet, les difficultés d'estimation des probabilités de la part des humains sont connues (TVERSKY et KAHNEMAN 1992). Pour corriger ce biais, nous proposons d'appliquer une transformation aux γ_k. Pour éviter toute ambiguïté, nous introduisons une nouvelle variable E_k utilisant cette transformation : $P(E_k=i|\mathbb{I}=i)=\varphi_k(\gamma_k)$ et c'est cette variable que nous exploiterons dans nos tests statistiques. Notons que $\varphi_k(\cdot)$ peut également inclure notre propre perception quant à la justesse du savoir de l'expert k. Dans ce manuscrit, nous suggérons d'utiliser la fonction logistique suivante de $[0,1]\mapsto[0,1]$:

$$\varphi_k(\gamma_k)=\frac{1}{\beta_k-1}\left[\frac{1+\beta_k}{1+\beta_k^{1-2\gamma_k}}-1\right], \tag{3.3}$$

avec β_k un paramètre strictement positif. Cette transformation est plutôt générique, comme le montre la figure 3.2 dans laquelle, pour simplifier, $\rho_k=2\log\beta_k$.

A partir de ce qui précède, pour prendre en compte les connaissances des experts et leurs incertitudes, nous pouvons étendre le réseau bayésien de la figure 3.1 comme

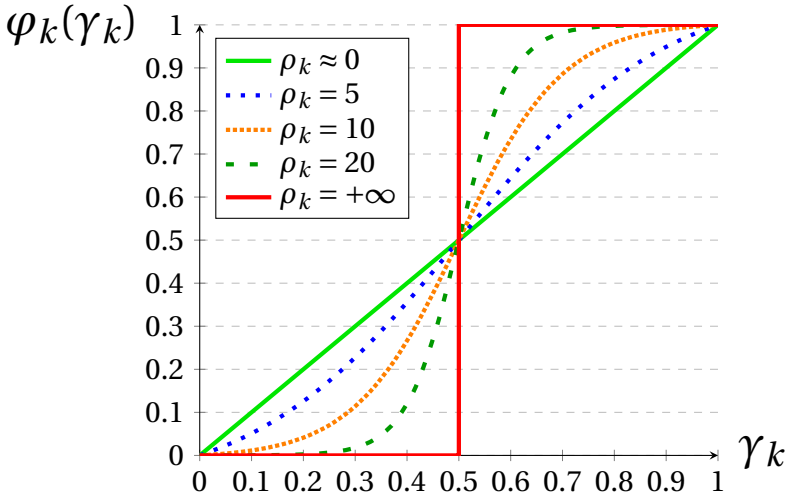


FIGURE 3.2 – Transformation de probabilité $\varphi_k(\cdot)$.

indiqué sur la figure 3.3. Les évidences e_k peuvent être introduites dans le réseau en tant que $P(e_k|E_k) = P(e_k|O_k)$.

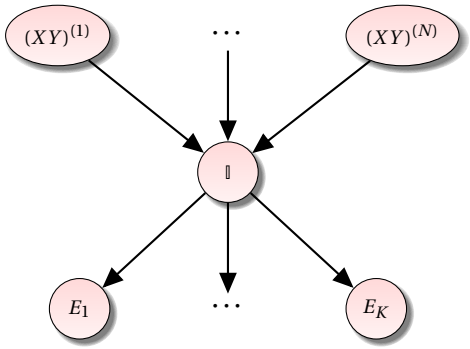


FIGURE 3.3 – Le réseau bayésien soulignant la relation entre les échantillons et les connaissances des experts.

3.3.2 Développement du test d'indépendance marginale

Le test d'indépendance G^2 est un test basé sur le rapport de vraisemblances noté $\mathscr{LR}(\mathbf{s}|\mathbf{e})$ et exprimé comme suit :

$$\mathscr{LR}(\mathbf{s}|\mathbf{e}) = \frac{\mathscr{L}(\mathbf{s}|\mathbf{e})}{\mathscr{L}(\mathbf{s}|\mathbb{I}=1,\mathbf{e})} \tag{3.4}$$

avec $\mathscr{L}(\mathbf{s}|\mathbf{e})$ la vraisemblance *a posteriori* et $\mathscr{L}(\mathbf{s}|\mathbb{I}=1,\mathbf{e})$ la vraisemblance sous l'hypothèse d'indépendance.

Commençons par calculer la vraisemblance *a posteriori* $\mathscr{L}(\mathbf{s}|\mathbf{e})$. Soit un échantillon

$\mathbf{s} = \{(xy)^{(n)}\}_{n=1}^{N}$ avec N observations des variables $\{(XY)^{(n)}\}_{n=1}^{N}$ sachant des connaissances d'experts $\mathbf{e} = \{e_1, \ldots, e_K\}$, on a :

$$\mathscr{L}(\mathbf{s}|\mathbf{e}) = \frac{P(\mathbf{s},\mathbf{e})}{P(\mathbf{e})}.$$

D'après le réseau bayésien donné dans la figure 3.3, il est possible de calculer le numérateur.

$$\begin{aligned} P(\mathbf{s},\mathbf{e}) &= \sum_{\mathbb{I}} \sum_{E_1,\ldots,E_K} \left[\left(\prod_{n=1}^{N} P((xy)^{(n)}) \right) \times P(\mathbb{I}|(xy)^{(1)},\ldots,(xy)^{(N)}) \times \left(\prod_{k=1}^{K} P(E_k|\mathbb{I}) P(e_k|E_k) \right) \right] \\ &= \sum_{\mathbb{I}} \left[\left(\prod_{n=1}^{N} P((xy)^{(n)}) \right) \times P(\mathbb{I}|(xy)^{(1)},\ldots,(xy)^{(N)}) \times \left(\sum_{E_1,\ldots,E_K} \prod_{k=1}^{K} P(E_k|\mathbb{I}) P(e_k|E_k) \right) \right] \end{aligned}$$

Notons ϵ_0 et ϵ_1 :

$$\begin{aligned} \epsilon_0 &= \sum_{E_1,\ldots,E_K} \prod_{k=1}^{K} P(E_k|\mathbb{I}=0) P(e_k|E_k) \\ \epsilon_1 &= \sum_{E_1,\ldots,E_K} \prod_{k=1}^{K} P(E_k|\mathbb{I}=1) P(e_k|E_k) \end{aligned}$$

Ainsi, en utilisant 3.2, nous avons donc :

$$\begin{aligned} P(\mathbf{s},\mathbf{e}) &= \left(\prod_{n=1}^{N} P((xy)^{(n)}) \right) \times P(\mathbb{I}=0|(xy)^{(1)},\ldots,(xy)^{(N)}) \times \epsilon_0 \\ &\quad + \left(\prod_{n=1}^{N} P((xy)^{(n)}) \right) \times P(\mathbb{I}=1|(xy)^{(1)},\ldots,(xy)^{(N)}) \times \epsilon_1 \\ &= \left[\left(\prod_{xy} \theta_{(xy)}^{N_{xy}} \right) - P(\mathbb{I}=1) \times \prod_{xy} \left(\theta_{(x.)} \times \theta_{(.y)} \right)^{N_{xy}} \right] \times \epsilon_0 + \left[P(\mathbb{I}=1) \times \prod_{xy} \left(\theta_{(x.)} \times \theta_{(.y)} \right)^{N_{xy}} \right] \times \epsilon_1 \end{aligned}$$

Soit en réarrangeant les termes :

$$P(\mathbf{s},\mathbf{e}) = \epsilon_0 \left(\prod_{xy} \theta_{(xy)}^{N_{xy}} \right) + \left(\prod_{xy} \left(\theta_{(x.)} \times \theta_{(.y)} \right)^{N_{xy}} \right) \times P(\mathbb{I}=1) \times [\epsilon_1 - \epsilon_0] \tag{3.5}$$

Calculons maintenant $P(e)$. D'après le réseau bayésien de la figure 3.3, on a :

$$\begin{aligned} P(\mathbf{e}) &= \sum_{\mathbb{I}} \sum_{E_1,\dots,E_K} P(\mathbb{I}) \prod_{k=1}^{K} P(E_k|\mathbb{I}) P(e_k|E_k) \\ &= P(\mathbb{I}=0) \times \epsilon_0 + P(\mathbb{I}=1) \times \epsilon_1 \\ &= (1 - P(\mathbb{I}=1)) \times \epsilon_0 + P(\mathbb{I}=1) \times \epsilon_1 \\ &= \epsilon_0 + P(\mathbb{I}=1) \times [\epsilon_1 - \epsilon_0]. \end{aligned}$$

On observe que $P(e)$ ne dépend que de nos connaissances de base sur l'indépendance entre $\mathbb{X}$ et $\mathbb{Y}$ ($P(\mathbb{I}=1)$) et des connaissances des experts (ϵ_0 et ϵ_1).

En utilisant l'équation 3.5 et l'expression de $P(\mathbf{e})$ ci-dessus, on trouve que :

$$\mathscr{L}(\mathbf{s}|\mathbf{e}) = \frac{\epsilon_0}{P(\mathbf{e})} \times \left(\prod_{xy} \theta_{(xy)}^{N_{xy}} \right) + \left(1 - \frac{\epsilon_0}{P(\mathbf{e})} \right) \times \left(\prod_{xy} \left(\theta_{(x.)} \times \theta_{(.y)} \right)^{N_{xy}} \right) \tag{3.6}$$

Une vraisemblance est une probabilité d'observer un certain échantillon $\mathbf{s}$ sachant une distribution de probabilité P sur la population. Ainsi, on peut voir $\mathscr{L}(\mathbf{s}|\mathbf{e})$ comme un mélange de vraisemblances de deux populations, une avec la distribution $P_0(\mathbb{X},\mathbb{Y}) = \{\theta_{(xy)} : x \in dom(\mathbb{X}), y \in dom(\mathbb{Y})\}$ et l'autre avec la distribution $P_1(\mathbb{X},\mathbb{Y}) = \{\theta_{(x.)} \times \theta_{(.y)} : x \in dom(\mathbb{X}), y \in dom(\mathbb{Y})\}$. Chaque vraisemblance, et donc chaque distribution, est plus ou moins mise en avant par les connaissances des experts, en fonction de leurs opinions.

Notons $\mathscr{L}_{P_d}(\mathbf{s}|\mathbf{e})$, avec $d \in \{0,1\}$, la valeur de la vraisemblance en fonction de P_d, i.e. :

$$\mathscr{L}_{P_0}(\mathbf{s}|\mathbf{e}) = \prod_{xy} \theta_{(xy)}^{N_{xy}}$$

$$\mathscr{L}_{P_1}(\mathbf{s}|\mathbf{e}) = \prod_{xy} \left(\theta_{(x.)} \times \theta_{(.y)} \right)^{N_{xy}}$$

Ainsi, nous pouvons mettre en avant les deux vraisemblances :

$$\mathscr{L}(\mathbf{s}|\mathbf{e}) = \frac{\epsilon_0}{P(\mathbf{e})} \times \mathscr{L}_{P_0}(\mathbf{s}|\mathbf{e}) + \left(1 - \frac{\epsilon_0}{P(\mathbf{e})} \right) \times \mathscr{L}_{P_1}(\mathbf{s}|\mathbf{e}) \tag{3.7}$$

Cette interprétation possède cependant un problème. En effet, si on prend le cas d'un expert seul et si celui-ci s'exprime en faveur d'une dépendance entre les variables $\mathbb{X}$ et $\mathbb{Y}$, alors on aura $\frac{\epsilon_0}{P(e)} > 1$ et donc $1 - \frac{\epsilon_0}{P(e)} < 0$. Or d'après l'interprétation donnée ci-dessus, on aurait alors une population négative, ce qui n'est pas satisfaisant. Heureusement, il est possible de trouver une autre expression interprétable lorsque 3.6 ne l'est pas. Pour cela, on peut d'abord facilement montrer que si $\frac{\epsilon_0}{P(e)} > 1$, alors $\epsilon_0 > \epsilon_1$. Or $P(e)$ peut également se réécrire de la manière suivante :

$$P(\mathbf{e}) = \epsilon_1 + P(\mathbb{I}=1) \times [\epsilon_0 - \epsilon_1] \tag{3.8}$$

L'égalité 3.8 est équivalente, pour $P(\mathbf{e}) \neq 0$, à :

$$1 = \frac{\epsilon_1}{P(\mathbf{e})} + \frac{P(\mathbb{I}=1) \times [\epsilon_0 - \epsilon_1]}{P(\mathbf{e})}$$

Dès lors que $\epsilon_0 > \epsilon_1$, on a donc $\frac{\epsilon_1}{P(\mathbf{e})} < 1$ et $1 - \frac{\epsilon_1}{P(\mathbf{e})} > 0$. Il devient alors possible de se servir de ces dernières équations pour obtenir une nouvelle interprétation. Réécrivons d'abord l'équation 3.2 de la manière suivante :

$$P(\mathbb{I}=i|(xy)^{(1)},\ldots,(xy)^{(N)}) \approx \begin{cases} P(\mathbb{I}=0) \times \prod_{n=1}^{N} \left(\frac{P((xy)^{(n)}|\mathbb{I}=0)}{P((xy)^{(n)})}\right)^{N_{xy}} & \text{si } i=0 \\ 1 - P(\mathbb{I}=0) \times \prod_{n=1}^{N} \left(\frac{P((xy)^{(n)}|\mathbb{I}=0)}{P((xy)^{(n)})}\right)^{N_{xy}} & \text{si } i=1. \end{cases} \tag{3.9}$$

En utilisant cette dernière expression et en procédant de la même manière que pour 3.5, on aboutit à une nouvelle expression pour $P(\mathbf{s},\mathbf{e})$:

$$P(\mathbf{s},\mathbf{e}) = \epsilon_1 \left(\prod_{n=1}^{N} P((xy)^{(n)})\right) + \left(\prod_{n=1}^{N} P((xy)^{(n)}|\mathbb{I}=0)\right) \times P(\mathbb{I}=0) \times [\epsilon_0 - \epsilon_1] \tag{3.10}$$

Il en découle alors également une nouvelle expression pour la vraisemblance $\mathscr{L}(\mathbf{s}|\mathbf{e})$:

$$\mathscr{L}(\mathbf{s}|\mathbf{e}) = \frac{\epsilon_1}{P(\mathbf{e})} \times \left(\prod_{n=1}^{N} P((xy)^{(n)})\right) + \left(1 - \frac{\epsilon_1}{P(\mathbf{e})}\right) \times \left(\prod_{n=1}^{N} P((xy)^{(n)}|\mathbb{I}=0)\right) \tag{3.11}$$

On peut alors de nouveau identifier deux populations associées aux distributions indéfinies $P'_0(\mathbb{X},\mathbb{Y})$ et $P'_1(\mathbb{X},\mathbb{Y})$ et de ce fait écrire :

$$\mathscr{L}(\mathbf{s}|\mathbf{e}) = \frac{\epsilon_1}{P(\mathbf{e})} \times \mathscr{L}_{P'_0}(\mathbf{s}|\mathbf{e}) + \left(1 - \frac{\epsilon_1}{P(\mathbf{e})}\right) \times \mathscr{L}_{P'_1}(\mathbf{s}|\mathbf{e}) \tag{3.12}$$

avec

$$\mathscr{L}_{P'_0}(\mathbf{s}|\mathbf{e}) = \prod_{n=1}^{N} P((xy)^{(n)})$$

$$\mathscr{L}_{P'_1}(\mathbf{s}|\mathbf{e}) = \prod_{n=1}^{N} P((xy)^{(n)}|\mathbb{I}=0)$$

Comme nous l'avons dit précédemment, si $\frac{\epsilon_0}{P(e)} > 1$, alors l'équation 3.6 ne peut pas être interprétée en termes de population. Cependant, dans ce cas, on a alors $\frac{\epsilon_1}{P(e)} < 1$ et c'est l'équation 3.11 qui peut être interprétée. Notons que les deux expressions sont bien entendu strictement égales.

Maintenant que nous avons calculé la vraisemblance *a posteriori*, calculons la vraisemblance sachant l'indépendance entre $\mathbb{X}$ et $\mathbb{Y}$, notée $\mathscr{L}(\mathbf{s}|\mathbb{I}=1,\mathbf{e})$, qui est égale

à :

$$\mathscr{L}(\mathbf{s}|\mathbb{I}=1,\mathbf{e}) = \frac{P(\mathbf{s},\mathbb{I}=1,\mathbf{e})}{P(\mathbb{I}=1,\mathbf{e})}$$

Avec la même méthode que précédemment, on a :

$$\begin{aligned} P(\mathbf{s},\mathbb{I}=1,\mathbf{e}) &= \sum_{E_1,\ldots,E_K}\left[\left(\prod_{n=1}^{N} P((xy)^{(n)})\right)\times P(\mathbb{I}=1|(xy)^{(1)},\ldots,(xy)^{(N)})\times\left(\prod_{k=1}^{K} P(E_k|\mathbb{I}=1)P(e_k|E_k)\right)\right] \\ &= \left[P(\mathbb{I}=1)\times\prod_{xy}\left(\theta_{(x.)}\times\theta_{(.y)}\right)^{N_{xy}}\right]\times\epsilon_1 \end{aligned}$$

On a également :

$$\begin{aligned} P(\mathbb{I}=1,\mathbf{e}) &= \sum_{E_1,\ldots,E_K} P(\mathbb{I}=1)\prod_{k=1}^{K} P(E_k|\mathbb{I}=1)P(e_k|E_k) \\ &= P(\mathbb{I}=1)\times\epsilon_1 \end{aligned}$$

Ce qui nous donne :

$$\mathscr{L}(\mathbf{s}|\mathbb{I}=1,\mathbf{e}) = \prod_{xy}\left(\theta_{(x.)}\times\theta_{(.y)}\right)^{N_{xy}}$$

Nous avons maintenant la vraisemblance *a posteriori* $\mathscr{L}(\mathbf{s}|\mathbf{e})$ et la vraisemblance sachant l'indépendance $\mathscr{L}(\mathbf{s}|\mathbb{I}=1,\mathbf{e})$. Nous avons donc le rapport de vraisemblances $\mathscr{LR}(\mathbf{s}|\mathbf{e})$. Et puisqu'il existe deux expressions pour $\mathscr{L}(\mathbf{s}|\mathbf{e})$, il en est de même pour $\mathscr{LR}(\mathbf{s}|\mathbf{e})$. Lorsque $\frac{\epsilon_0}{P(e)} \leq 1$, alors on peut utiliser l'expression suivante :

$$\begin{aligned} \mathscr{LR}(\mathbf{s}|\mathbf{e}) &= \frac{\mathscr{L}(\mathbf{s}|\mathbf{e})}{\mathscr{L}(\mathbf{s}|\mathbb{I}=1,\mathbf{e})} \\ &= \frac{\epsilon_0}{P(\mathbf{e})}\times\mathscr{LR}_{P_0}(\mathbf{s}|\mathbf{e}) + \left(1-\frac{\epsilon_0}{P(\mathbf{e})}\right)\times\mathscr{LR}_{P_1}(\mathbf{s}|\mathbf{e}) \end{aligned} \tag{3.13}$$

avec

$$\begin{aligned} \mathscr{LR}_{P_0}(\mathbf{s}|\mathbf{e}) &= \frac{\prod_{xy}\theta_{(xy)}^{N_{xy}}}{\prod_{xy}\left(\theta_{(x.)}\times\theta_{(.y)}\right)^{N_{xy}}} \\ \mathscr{LR}_{P_1}(\mathbf{s}|\mathbf{e}) &= \frac{\prod_{xy}\left(\theta_{(x.)}\times\theta_{(.y)}\right)^{N_{xy}}}{\prod_{xy}\left(\theta_{(x.)}\times\theta_{(.y)}\right)^{N_{xy}}}. \end{aligned}$$

Dans la continuité des premières interprétations, le rapport de vraisemblances qui nous intéresse est un mélange de deux rapports de vraisemblances, un sur la population de distribution P_1 et l'autre sur la population de distribution P_0. Or le rapport de vraisemblances $\mathscr{LR}_{P_1}(\mathbf{s}|\mathbf{e})$ est toujours égal à 1 et n'apporte donc aucune information pour notre test. Ainsi, ce nouveau test repose essentiellement sur $\mathscr{LR}_{P_0}(\mathbf{s}|\mathbf{e})$.

Et lorsque $\frac{\epsilon_0}{P(e)} > 1$, alors on utilise l'expression :

$$\mathscr{LR}(\mathbf{s}|\mathbf{e}) = \frac{\epsilon_1}{P(\mathbf{e})} \times \mathscr{LR}_{P'_0}(\mathbf{s}|\mathbf{e}) + \left(1 - \frac{\epsilon_1}{P(\mathbf{e})}\right) \times \mathscr{LR}_{P'_1}(\mathbf{s}|\mathbf{e}) \tag{3.14}$$

avec

$$\mathscr{LR}_{P'_0}(\mathbf{s}|\mathbf{e}) = \frac{\prod_{n=1}^{N} P((xy)^{(n)})}{\prod_{n=1}^{N} P((xy)^{(n)}|\mathbb{I}=1)}$$
$$\mathscr{LR}_{P'_1}(\mathbf{s}|\mathbf{e}) = \frac{\prod_{n=1}^{N} P((xy)^{(n)}|\mathbb{I}=0)}{\prod_{n=1}^{N} P((xy)^{(n)}|\mathbb{I}=1)}.$$

Cette dernière équation peut être interprétée en termes de population de la même manière que 3.13, avec les populations des distributions P'_0 et P'_1. Cependant, ces distributions ne sont pas aussi bien définies que celles de l'équation 3.13, et il ne sera donc pas possible d'effectuer les mêmes calculs. Ainsi, l'équation 3.21 nous permet de compléter l'interprétation du rapport de vraisemblances, mais ne sera pas utilisée dans la suite du développement du test d'indépendance marginale. Rappelons une nouvelle fois que les deux expressions sont strictement égales, les calculs sont donc équivalents pour les deux cas.

Remarquons que lorsqu'aucune connaissance e_k n'est disponible, nous retrouvons $\epsilon_0 = \epsilon_1 = P(\mathbf{e}) = 1$, ce qui correspond aux vraisemblances traditionnellement utilisées dans les tests du G^2 classique. De la même manière, lorsque nous avons les mêmes niveaux de confiance pour chaque expert et quand les experts n'ont pas de connaissances ou donnent juste une estimation 50-50% sur l'indépendance et la dépendance, alors $\epsilon_0 = \epsilon_1$. Or dès lors que $\epsilon_0 = \epsilon_1$, on obtient $P(\mathbf{e}) = \epsilon_0$, ce qui nous redonne également les vraisemblances traditionnellement utilisées dans les tests du G^2 classique.

Rappelons maintenant qu'avec le test du G^2 classique, si on suppose que la vraisemblance sous la contre-hypothèse est proche de celle sous l'hypothèse nulle, leur rapport devrait être proche de 1. Par conséquent, sous cette hypothèse, à un niveau de risque (de première espèce) α, le test du G^2 classique consiste à calculer le seuil au-dessus duquel seul un pourcentage α de l'ensemble de tous les échantillons possibles va produire un rapport plus grand que ce seuil.

Notons $\eta = \epsilon_0 / P(\mathbf{e})$. D'après l'équation 3.13, nous pouvons constater que lorsque $\mathscr{LR}(\mathbf{S}|\mathbf{e})$ tend vers 1, c'est-à-dire en supposant que l'hypothèse nulle est la bonne, $\mathscr{LR}_{P_0}(\mathbf{S}|\mathbf{e})$ tend également vers 1. Or on a également :

$$2\ln\big(\mathscr{LR}_{P_0}(\mathbf{S}|\mathbf{e})\big) = 2\ln\left(\frac{\prod_{xy}\theta_{(xy)}^{N_{xy}}}{\prod_{xy}\left(\theta_{(x.)} \times \theta_{(.y)}\right)^{N_{xy}}}\right) \sim \chi^2_k \tag{3.15}$$

avec $k = (|dom(\mathbb{X})| - 1) \times (|dom(\mathbb{Y})| - 1)$. Sachant que le logarithme est strictement croissant, nous avons :

$$P(\mathscr{LR}_{P_0}(\mathbf{S}|\mathbf{e}) > c) = \alpha\eta \Longleftrightarrow P(2\ln\big(\mathscr{LR}_{P_0}(\mathbf{S}|\mathbf{e})\big) > C_{k,\alpha\eta}) = \alpha\eta$$

où $C_{k,\alpha\eta}$ est le quantile $(1-\alpha\eta)$ de la distribution χ^2 avec k degrés de liberté.

Ainsi, notre test, en se basant sur un échantillon $\mathbf{s}$, peut juger l'indépendance entre les deux variables $\mathbb{X}$ et $\mathbb{Y}$ en prenant en compte des connaissances d'experts simplement en effectuant le test G^2 classique avec un risque de première espèce $\alpha_e = \alpha \times \frac{\epsilon_0}{P(\mathbf{e})}$ à la place de α comme pour le test usuel.

3.3.3 Modèle pour le test conditionnel

Nous savons maintenant comment mener un test d'indépendance marginale en prenant en compte les connaissances des experts. Passons maintenant aux tests d'indépendance conditionnelle qui prennent en compte de la même façon les connaissances des experts.

Soit $\mathbb{Z}$ un ensemble de variables aléatoires. Nous souhaitons tester l'indépendance entre $\mathbb{X}$ et $\mathbb{Y}$ sachant $\mathbb{Z}$. Comme pour le cas marginal, nous introduisons les variables aléatoires $\{(\mathbf{Z})^{(n)}\}_{n=1}^N$ distribuées selon $P(\mathbb{Z})$. En ajoutant ces variables dans le réseau bayésien représentant le test d'indépendance, nous obtenons un nouveau réseau donné par la figure 3.4.

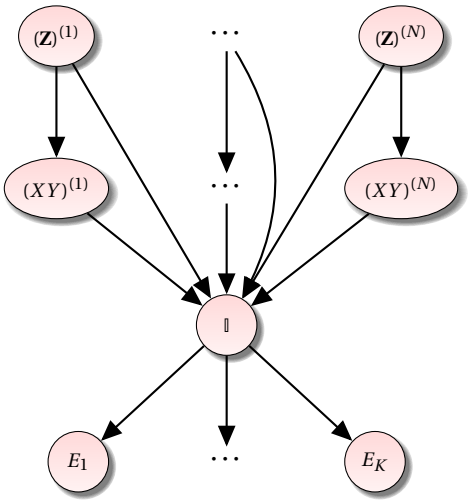


FIGURE 3.4 – Le réseau bayésien pour le test d'indépendance conditionnelle.

Soit $dom(\mathbb{Z}) = \{\mathbf{z}_1,\ldots,\mathbf{z}_t\}$ et $P(\mathbb{Z}) = \{\theta_{(\mathbf{z})} : \mathbf{z} \in \{\mathbf{z}_1,\ldots,\mathbf{z}_t\}\}$. Notons également pour les probabilités conditionnelles entre $\mathbb{X}$ et $\mathbb{Y}$ sachant $\mathbb{Z}$: $P(\mathbb{X},\mathbb{Y}|\mathbb{Z}) = \{\theta_{(xy|\mathbf{z})} : x \in \{x_1,\ldots,x_r\}, y \in \{y_1,\ldots,y_s\}, \mathbf{z} \in \{\mathbf{z}_1,\ldots,\mathbf{z}_t\}\}$. Enfin, donnons aux variables $(\mathbf{Z})^{(n)}$ les distributions $\{\theta_{(\mathbf{z})} : \mathbf{z} \in \{\mathbf{z}_1,\ldots,\mathbf{z}_t\}\}$.

Les probabilités conditionnelles de $\mathbb{I}$ sachant $(XY)^{(1)},\ldots,(XY)^{(N)},(\mathbf{Z})^{(1)},\ldots,(\mathbf{Z})^{(N)}$ peuvent être calculées de la même façon que dans le cas marginal. En appliquant la formule de Bayes, on a :

$$P(\mathbb{I}|\{(XY)^{(n)},(\mathbf{Z})^{(n)}\}_{n=1}^N) = \frac{P(\{(XY)^{(n)}\}_{n=1}^N|\mathbb{I},\{(\mathbf{Z})^{(n)}\}_{n=1}^N) \times P(\mathbb{I}|\{(\mathbf{Z})^{(n)}\}_{n=1}^N)}{P(\{(XY)^{(n)}\}_{n=1}^N|\{(\mathbf{Z})^{(n)}\}_{n=1}^N)}$$

Par d-séparation, $P(\{(XY)^{(n)}\}_{n=1}^N|\{(\mathbf{Z})^{(n)}\}_{n=1}^N) = \prod_{n=1}^N P((XY)^{(n)}|(\mathbf{Z})^{(n)})$.
On peut approcher $P(\{(XY)^{(n)}\}_{n=1}^N|\mathbb{I},\{(\mathbf{Z})^{(n)}\}_{n=1}^N)$ avec $\prod_{n=1}^N P((XY)^{(n)}|\mathbb{I},(\mathbf{Z})^{(n)})$. Notons que $P(\mathbb{I}|\{(\mathbf{Z})^{(n)}\}_{n=1}^N)$ est la certitude que nous avons qu'il existe une indépendance conditionnelle entre $\mathbb{X}$ et $\mathbb{Y}$ lorsque les seules informations disponibles sont les observations de $\mathbb{Z}$ dans $\mathbf{s}$. Sans aucune connaissance de $\mathbb{X}$ ou $\mathbb{Y}$, on peut supposer que $\{(\mathbf{Z})^{(n)}\}_{n=1}^N$ ne fournit pas plus d'informations sur $\mathbb{I}$, ce qui nous donne $P(\mathbb{I}|\{(\mathbf{Z})^{(n)}\}_{n=1}^N) \approx P(\mathbb{I})$.

Ainsi, nous avons :

$$P(\mathbb{I}|\{(XY)^{(n)},(\mathbf{Z})^{(n)}\}_{n=1}^N) \approx P(\mathbb{I}) \times \prod_{n=1}^N \frac{P((XY)^{(n)}|\mathbb{I},(\mathbf{Z})^{(n)})}{P((XY)^{(n)}|(\mathbf{Z})^{(n)})}$$

Nous avons déjà défini $P((XY)^{(n)} = (xy)^{(n)}|(\mathbf{Z})^{(n)} = \mathbf{z}) = \theta_{(xy|z)^{(n)}}$. Pour tout $x \in \{x_1,\dots,x_r\}$, notons $\theta_{(x.|z)} = \sum_{y=y_1}^{y_s} \theta_{(xy|\mathbf{z})}$ et, pour tout $y \in \{y_1,\dots,y_s\}$, notons $\theta_{(.y|z)} = \sum_{x=x_1}^{x_r} \theta_{(xy|\mathbf{z})}$. Par abus de notation, notons également $\theta_{(x.|\mathbf{z})^{(n)}} = \sum_{y=y_1}^{y_s} \theta_{(x^{(n)}y|\mathbf{z}^{(n)})}$ et $\theta_{(.y|\mathbf{z})^{(n)}} = \sum_{x=x_1}^{x_r} \theta_{(xy^{(n)}|\mathbf{z}^{(n)})}$. Enfin, puisque $\mathbb{I}=1$ signifie que $\mathbb{X}$ et $\mathbb{Y}$ sont conditionnellement indépendants sachant $\mathbb{Z}$, nous avons $P((XY)^{(n)} = (xy)^{(n)}|\mathbb{I}=1,(\mathbf{Z})^{(n)} = \mathbf{z}^{(n)}) = \theta_{(x.|\mathbf{z})^{(n)}} \times \theta_{(.y|\mathbf{z})^{(n)}}$.

Cela nous donne donc :

$$P(\mathbb{I}=1|\{(XY)^{(n)} = (xy)^{(n)},(\mathbf{Z})^{(n)} = (\mathbf{z})^{(n)}\}_{n=1}^N) \approx P(\mathbb{I}=1) \times \prod_{n=1}^N \frac{\theta_{(x.|\mathbf{z})^{(n)}} \times \theta_{(.y|\mathbf{z})^{(n)}}}{\theta_{(xy|\mathbf{z})^{(n)}}}$$

Soit $\mathbf{s} = \{(xy\mathbf{z})^{(n)}\}_{n=1}^N$ un échantillon d'observations des variables $\{(XY\mathbf{Z})^{(n)}\}_{n=1}^N$. Notons $N_{xy\mathbf{z}}$ le nombre de variables $(XY\mathbf{Z})^{(n)}$ dont les valeurs sont exactement égales à $(xy\mathbf{z})$ dans l'échantillon $\mathbf{s}$. Ainsi :

$$P(\mathbb{I}=1|\{(XY)^{(n)} = (xy)^{(n)},(\mathbf{Z})^{(n)} = (\mathbf{z})^{(n)}\}_{n}^N) \approx P(\mathbb{I}=1) \times \prod_{xy\mathbf{z}} \left(\frac{\theta_{(x.|\mathbf{z})} \times \theta_{(.y|\mathbf{z})}}{\theta_{(xy|\mathbf{z})}}\right)^{N_{xy\mathbf{z}}}$$

Ainsi, puisque $P(\mathbb{I}=0|\{(XY)^{(n)},(\mathbf{Z})^{(n)}\}_{n=1}^N) + P(\mathbb{I}=1|\{(XY)^{(n)},(\mathbf{Z})^{(n)}\}_{n=1}^N) = 1$, nous avons :

$$P(\mathbb{I}=i|\{(xy)^{(n)},(\mathbf{Z})^{(n)}\}_{n=1}^N) \approx \begin{cases} 1 - P(\mathbb{I}=1) \times \prod_{xy\mathbf{z}} \left(\frac{\theta_{(x.|\mathbf{z})} \times \theta_{(.y|\mathbf{z})}}{\theta_{(xy|\mathbf{z})}}\right)^{N_{xy\mathbf{z}}} & \text{si } i=0 \\ P(\mathbb{I}=1) \times \prod_{xy\mathbf{z}} \left(\frac{\theta_{(x.|\mathbf{z})} \times \theta_{(.y|\mathbf{z})}}{\theta_{(xy|\mathbf{z})}}\right)^{N_{xy\mathbf{z}}} & \text{si } i=1. \end{cases} \tag{3.16}$$

Encore une fois, nous pouvons constater que l'équation ci-dessus ne donne aucune contrainte sur la valeur de $P(\mathbb{I}=1)$. Nous choisirons encore une fois la valeur $P(\mathbb{I}=1) = 0.95$.

Les connaissances des experts sont pour leur part ajoutées de la même manière que dans le cas marginal.

3.3.4 Développement du test d'indépendance conditionnelle

Maintenant que nous avons développé les distributions de probabilité importantes, nous pouvons passer au calcul du rapport de vraisemblances comme dans le cas marginal. Ainsi, nous allons commencer par calculer la vraisemblance *a priori* $\mathscr{L}(\mathbf{s}|\mathbf{e})$ grâce à son expression :

$$\mathscr{L}(\mathbf{s}|\mathbf{e}) = \frac{P(\mathbf{s},\mathbf{e})}{P(\mathbf{e})}.$$

Grâce au réseau bayésien de la figure 3.4, nous avons :

$$\begin{aligned} P(\mathbf{s},\mathbf{e}) &= \sum_{\mathbb{I}} \sum_{E_1,\dots,E_K} \left[\left(\prod_{n=1}^{N} P((\mathbf{z})^{(n)}) P((xy|\mathbf{z})^{(n)}) \right) \times P(\mathbb{I}|\{(xy)^{(n)},(\mathbf{z})^{(n)}\}_{n=1}^{N}) \times \left(\prod_{k=1}^{K} P(E_k|\mathbb{I}) P(e_k|E_k) \right) \right] \\ &= \sum_{\mathbb{I}} \left[\left(\prod_{n=1}^{N} P((\mathbf{z})^{(n)}) P((xy|\mathbf{z})^{(n)}) \right) \times P(\mathbb{I}|\{(xy)^{(n)},(\mathbf{z})^{(n)}\}_{n=1}^{N}) \times \left(\sum_{E_1,\dots,E_K} \prod_{k=1}^{K} P(E_k|\mathbb{I}) P(e_k|E_k) \right) \right] \end{aligned}$$

On définit comme précédemment ϵ_0 et ϵ_1 de la manière suivante :

$$\begin{aligned} \epsilon_0 &= \sum_{E_1,\dots,E_K} \prod_{k=1}^{K} P(E_k|\mathbb{I}=0) P(e_k|E_k) \\ \epsilon_1 &= \sum_{E_1,\dots,E_K} \prod_{k=1}^{K} P(E_k|\mathbb{I}=1) P(e_k|E_k) \end{aligned}$$

Avec ceci, on obtient :

$$\begin{aligned} P(\mathbf{s},\mathbf{e}) &= \left[\left(\prod_{n=1}^{N} P((\mathbf{z})^{(n)}) P((xy|\mathbf{z})^{(n)}) \right) \times P(\mathbb{I}=0|\{(xy)^{(n)},(\mathbf{z})^{(n)}\}_{n=1}^{N}) \times \epsilon_0 \right] + \\ &\quad \left[\left(\prod_{n=1}^{N} P((\mathbf{z})^{(n)}) P((xy|\mathbf{z})^{(n)}) \right) \times P(\mathbb{I}=1|\{(xy)^{(n)},(\mathbf{z})^{(n)}\}_{n=1}^{N}) \times \epsilon_1 \right] \\ &= \prod_{\mathbf{z}} \theta_{(\mathbf{z})}^{N_{\mathbf{z}}} \times \left[\epsilon_0 \left(\prod_{xy\mathbf{z}} \theta_{(xy|\mathbf{z})}^{N_{xy\mathbf{z}}} \right) + P(\mathbb{I}=1) \times [\epsilon_1 - \epsilon_0] \times \left(\prod_{xy\mathbf{z}} \left(\theta_{(x.|\mathbf{z})} \times \theta_{(.y|\mathbf{z})} \right)^{N_{xy\mathbf{z}}} \right) \right] \end{aligned}$$

Comme pour le cas marginal, nous avons également :

$$P(\mathbf{e}) = \sum_{\mathbb{I}} \sum_{E_1,\dots,E_K} P(\mathbb{I}) \prod_{k=1}^{K} P(E_k|\mathbb{I}) P(e_k|E_k) = \epsilon_0 + P(\mathbb{I}=1) \times [\epsilon_1 - \epsilon_0]$$

On arrive ainsi à l'expression suivante :

$$\mathscr{L}(\mathbf{s}|\mathbf{e}) = \prod_{\mathbf{z}} \theta_{(\mathbf{z})}^{N_{\mathbf{z}}} \times \left[\frac{\epsilon_0}{P(\mathbf{e})} \times \left(\prod_{xy\mathbf{z}} \theta_{(xy|\mathbf{z})}^{N_{xy\mathbf{z}}} \right) + \left(1 - \frac{\epsilon_0}{P(\mathbf{e})} \right) \times \left(\prod_{xy\mathbf{z}} \left(\theta_{(x.|z)} \times \theta_{(.y|\mathbf{z})} \right)^{N_{xy\mathbf{z}}} \right) \right] \quad (3.17)$$

Cette vraisemblance peut également être réécrite en fonction de ϵ_1, de la même

manière que pour le test marginal :

$$\mathscr{L}(\mathbf{s}|\mathbf{e}) = \prod_{n=1}^{N} P((\mathbf{z})^{(n)}) \times \left[\frac{\epsilon_1}{P(\mathbf{e})} \times \left(\prod_{n=1}^{N} P((xy|\mathbf{z})^{(n)}) \right) + \left(1 - \frac{\epsilon_1}{P(\mathbf{e})} \right) \times \left(\prod_{n=1}^{N} P((xy|\mathbf{z})^{(n)} | \mathbb{I} = 0) \right) \right] \tag{3.18}$$

Passons à présent au calcul de la vraisemblance sous l'hypothèse nulle.

$$\mathscr{L}(\mathbf{s}|\mathbb{I} = 1, \mathbf{e}) = \frac{P(\mathbf{s}, \mathbb{I} = 1, \mathbf{e})}{P(\mathbb{I} = 1, \mathbf{e})}$$

$$\begin{aligned} P(\mathbf{s}, \mathbb{I} = 1, \mathbf{e}) &= \sum_{E_1,\ldots,E_K} \left(\prod_{n=1}^{N} P((\mathbf{z})^{(n)}) P((xy|\mathbf{z})^{(n)}) \right) \times P(\mathbb{I} = 1 | \{(xy)^{(n)}, (\mathbf{z})^{(n)}\}_{n=1}^{N}) \\ &\quad \times \left(\prod_{k=1}^{K} P(E_k|\mathbb{I} = 1) P(e_k|E_k) \right) \\ &= \prod_{\mathbf{z}} \theta_{(\mathbf{z})}^{N_{\mathbf{z}}} \times \epsilon_1 \times \left[P(\mathbb{I} = 1) \times \prod_{xy\mathbf{z}} \left(\theta_{(x.|\mathbf{z})} \times \theta_{(.y|\mathbf{z})} \right)^{N_{xy\mathbf{z}}} \right] \end{aligned}$$

$$P(\mathbb{I} = 1, \mathbf{e}) = \sum_{E_1,\ldots,E_K} P(\mathbb{I} = 1) \prod_{k=1}^{K} P(E_k|\mathbb{I} = 1) P(e_k|E_k) = P(\mathbb{I} = 1) \times \epsilon_1$$

Avec ces dernières expressions, nous obtenons :

$$\begin{aligned} \mathscr{LR}(\mathbf{s}|\mathbf{e}) &= \frac{\mathscr{L}(\mathbf{s}|\mathbf{e})}{\mathscr{L}(\mathbf{s}|\mathbb{I} = 1, \mathbf{e})} && (3.19) \\ &= \frac{\epsilon_0}{P(\mathbf{e})} \times \mathscr{LR}_{P_0}(\mathbf{s}|\mathbf{e}) + \left(1 - \frac{\epsilon_0}{P(\mathbf{e})} \right) \times \mathscr{LR}_{P_1}(\mathbf{s}|\mathbf{e}) && (3.20) \end{aligned}$$

avec :

$$\mathscr{LR}_{P_0}(\mathbf{s}|\mathbf{e}) = \frac{\prod_{xy\mathbf{z}} \theta_{(xy|\mathbf{z})}^{N_{xy\mathbf{z}}}}{\prod_{xy\mathbf{z}} \left(\theta_{(x.|\mathbf{z})} \times \theta_{(.y|\mathbf{z})} \right)^{N_{xy\mathbf{z}}}}$$

$$\mathscr{LR}_{P_1}(\mathbf{s}|\mathbf{e}) = \frac{\prod_{xy\mathbf{z}} \left(\theta_{(x.|\mathbf{z})} \times \theta_{(.y|\mathbf{z})} \right)^{N_{xy\mathbf{z}}}}{\prod_{xy|\mathbf{z}} \left(\theta_{(x.|\mathbf{z})} \times \theta_{(.y|\mathbf{z})} \right)^{N_{xy\mathbf{z}}}}.$$

Encore une fois, le rapport de vraisemblances peut lui aussi être exprimé en fonction de ϵ_1, pour l'interprétation, lorsque $\frac{\epsilon_0}{P(e)} > 1$.

$$\mathscr{LR}(\mathbf{s}|\mathbf{e}) = \frac{\epsilon_1}{P(\mathbf{e})} \times \mathscr{LR}_{P_0'}(\mathbf{s}|\mathbf{e}) + \left(1 - \frac{\epsilon_1}{P(\mathbf{e})} \right) \times \mathscr{LR}_{P_1'}(\mathbf{s}|\mathbf{e}) \tag{3.21}$$

avec

$$\mathscr{LR}_{P'_0}(\mathbf{s}|\mathbf{e}) = \frac{\prod_{n=1}^{N} P((xy|\mathbf{z})^{(n)})}{\prod_{n=1}^{N} P((xy|\mathbf{z})^{(n)}|\mathbb{I}=1)}$$
$$\mathscr{LR}_{P'_1}(\mathbf{s}|\mathbf{e}) = \frac{\prod_{n=1}^{N} P((xy|\mathbf{z})^{(n)}|\mathbb{I}=0)}{\prod_{n=1}^{N} P((xy|\mathbf{z})^{(n)}|\mathbb{I}=1)}.$$

Identiquement au raisonnement mené sur le cas marginal, notre test, en s'appuyant sur un échantillon $\mathbf{s}$, peut juger l'indépendance entre les variables $\mathbb{X}$ et $\mathbb{Y}$ conditionnellement à un ensemble de variables $\mathbb{Z}$ en prenant en compte des connaissances d'experts simplement en effectuant le test conditionnel G^2 classique avec un risque de première espèce $\alpha_e = \alpha \times \frac{\epsilon_0}{P(\mathbf{e})}$ en lieu et place de α comme dans le test usuel.

3.4 Algorithme

3.4.1 Problèmes rencontrés et résolus

En théorie, nous pouvons directement utiliser notre nouveau test d'indépendance au sein des algorithmes d'apprentissage de structure de réseaux bayésiens sous contraintes (PC, IC, PC-stable, etc.). Toutefois, l'utilisation des connaissances incertaines soulève deux problèmes majeurs :

1. la présence des bases de données de taille réduite
2. la divergence entre les connaissances d'experts et les résultats obtenus avec les tests d'indépendance

Concernant le premier problème, les algorithmes d'apprentissage ignorent généralement les résultats retournés par le test d'indépendance lorsque la base de données est trop petite, car les résultats obtenus ne sont pas assez fiables. En revanche, l'utilisation des savoirs d'experts dans ce genre de situations demeure tout de même possible, et on peut donc s'appuyer sur ces connaissances pour conclure sur l'indépendance étudiée. Pour cela, notons d la décision de considérer que « $\mathbb{X}$ et $\mathbb{Y}$ sont indépendants sachant $\mathbb{Z}$ » et soient u_1 et u_0 les utilités que l'on associe à une telle décision lorsque l'expert a respectivement raison et tort. En d'autres termes, u_1 et u_0 représentent notre perception du gain et de la perte associés à la prise de décision d lorsque celle-ci se fonde uniquement sur le savoir d'expert (nous supposons que $u_1 > 0$ et $u_0 < 0$). En général, l'élicitation d'utilités telles que u_1 et u_0 est difficile à réaliser. En revanche, leur ratio $T = -u_0/u_1$ est beaucoup plus facile à estimer (Keeney et Raiffa 1976). Enfin, notons par ϵ_1 et ϵ_0 les probabilités que les experts aient respectivement raison et tort. Si notre critère pour prendre nos décisions est la maximisation de l'espérance d'utilité, nous devrions considérer $\mathbb{X}$ et $\mathbb{Y}$ conditionnellement indépendants sachant $\mathbb{Z}$ si et seulement si $\epsilon_1 u_1 + \epsilon_0 u_0 > 0$ ou, de manière équivalente, si $\epsilon_1/\epsilon_0 > T$. Nous utilisons précisément cette règle dans l'algorithme 13.

Concernant le deuxième problème, notons que retirer ou garder une arête à une étape d'un algorithme d'apprentissage sous contraintes peut avoir un impact significa-

tif sur les étapes suivantes. De telles décisions doivent donc être prises avec prudence. Quand les conclusions avec et sans connaissances d'experts sont identiques, nous pouvons être confiants dans la décision qui en résulte. Au contraire, quand les conclusions diffèrent, nous devrions suivre la conclusion du test qui inclut les savoirs d'experts, puisque ce test prend en compte plus d'informations. Cependant, il ne serait pas prudent de complètement ignorer la conclusion obtenue sans les connaissances d'experts. Ainsi, dans l'algorithme 13, nous construisons un graphe G qui sera le squelette appris, mais nous proposons également de maintenir en parallèle un autre graphe G_{adj} utilisé pour déterminer les *sepset*, c'est-à-dire les ensembles de séparation qui rendent des variables indépendantes conditionnellement à cet ensemble. Le graphe G_{adj} prend en compte les désaccords entre les conclusions avec et sans connaissances d'experts. En d'autres termes, le graphe G est le graphe appris et est maintenu grâce aux tests d'indépendance qui prennent en compte les connaissances des experts. Et le graphe G_{adj} est le graphe duquel on prend les adjacences des nœuds lorsqu'on forme les ensembles de conditionnement pour les tests d'indépendance et est maintenu grâce aux tests d'indépendance sans les connaissances des experts.

3.4.2 Pseudo-code

Ces variations combinées dans l'algorithme PC-stable (COLOMBO et M. H. MAATHUIS 2014) résultent en un algorithme que l'on nomme **PCSe**. Celui-ci est partiellement décrit dans l'algorithme 13, où seule l'étape de construction du squelette est représentée. Le reste de l'algorithme se développe de la même manière que les algorithmes PC et PC-stable, avec une étape d'orientation des v-structures et une étape de propagation des orientations.

Décrivons un peu plus en détail l'algorithme 13. Comme nous le voyons, les deux graphes G et G_{adj} maintenus parallèlement sont initialisés dès le début, ligne 2 et 3. On part naturellement de graphes complets. À partir de la ligne 13, les connaissances commencent à être utilisées, d'abord avec l'extraction des connaissances depuis la base de connaissances, puis avec le calcul du facteur η. La solution décrite dans la sous-partie précédente lorsque la base de données est trop petite est appliquée de la ligne 15 à 18. Ensuite, à la ligne 20, c'est le test G^2 classique, qui est effectué pour déterminer la p-valeur. On compare ensuite cette p-valeur au risque de première espèce α et au risque de première espèce modifié $\alpha \times \eta$ pour décider respectivement des suppressions de l'arête dans le graphe G_{adj} et G. L'ensemble de séparation n'est sauvegardé qu'en cas de suppression dans le graphe G. Enfin, ligne 30, la condition de sortie de la boucle est adaptée aux tests faits dans l'algorithme.

Algorithm 13 Algorithme PCSe

Entrée : Un ensemble d'observations **S**, connaissances d'experts **K**, niveau de risque α, connaissances *a priori* $P(\mathbb{I})$, seuil T
Sortie : un graphe non orienté G et un ensemble de $Sepsets$

1: $\mathbf{V} \leftarrow$ toutes les variables/colonnes de **S**
2: $G = (\mathbf{V}, \mathbf{E}) \leftarrow$ graphe complet non orienté
3: $G_{adj} = (\mathbf{V}, \mathbf{E}_{adj}) \leftarrow$ graphe complet non orienté
4: $m \leftarrow -1$; $Sepset \leftarrow \emptyset$
5: **Répéter**
6: $m \leftarrow m+1$
7: **Pour tout** X_i dans **V Faire**
8: $\mathbf{adj}_i \leftarrow \{X_j \in \mathbf{V} : (X_i, X_j) \in \mathbf{E}_{adj}\}$
9: **Fin Pour**
10: **Pour tout** $(X_i, X_j) \in \mathbf{E}_{adj}$ t.q. $|\mathbf{adj}_i| \geq m+1$ **Faire**
11: **Répéter**
12: Choisir un ensemble $\mathbf{Z} \subseteq \mathbf{adj}_i \setminus \{X_j\}$ t.q. $|\mathbf{Z}| = m$
13: Extraire de **K** la connaissance sur $(X_i, X_j|\mathbf{Z})$
14: Créer le réseau bayésien de la figure 3.4 et calculer ses paramètres θ, ϵ_0, ϵ_1, $P(\mathbf{e})$ et $\eta = \epsilon_0 / P(\mathbf{e})$
15: **Si S** est trop petit pour faire le test du G^2 **Alors**
16: **Si** $\epsilon_1/\epsilon_0 > T$ **Alors**
17: Supprimer (X_i, X_j) dans **E** et $\mathbf{E}_{adj}$
18: **Fin Si**
19: **Sinon**
20: $p \leftarrow$ p-valeur du test G^2 classique
21: **Si** $p \geq \alpha$ **Alors**
22: Supprimer (X_i, X_j) dans $\mathbf{E}_{adj}$
23: **Fin Si**
24: **Si** $p \geq \alpha \times \eta$ **Alors**
25: Supprimer (X_i, X_j) dans **E**
26: $Sepset(X_i, X_j) \leftarrow \mathbf{Z}$
27: $Sepset(X_j, X_i) \leftarrow \mathbf{Z}$
28: **Fin Si**
29: **Fin Si**
30: **Jusqu'à ce que** $p \geq \alpha$ et $p \geq \alpha \times \eta$, ou tous les **Z** t.q. $|\mathbf{Z}| = m$ ont été considérés
31: **Fin Pour**
32: **Jusqu'à ce que** toutes les variables $X_i \in \mathbf{V}$ sont telles que $|\mathbf{adj}_i| \leq m$
33: **Retourner** G, $Sepset$

3.5 Expérimentations

3.5.1 Paramètres expérimentaux

Dans cette section, nous allons décrire le comportement de l'algorithme PCSe et démontrer son efficacité en le comparant à d'autres algorithmes. Pour cela, nous avons exécuté l'algorithme sur des données générées depuis des réseaux bayésiens connus et couramment utilisés dans la littérature (voir 3.1 [1]). Pour la comparaison, nous avons également exécuté et observé les résultats de deux autres algorithmes :

— PCS (COLOMBO et M. H. MAATHUIS 2014), que nous avons déjà décrit au chapitre 2, dans la section 2.3.3.1

— PCSDC, qui est une version de l'algorithme PCDC L. M. d. CAMPOS et CASTELLANO 2007 que nous avons décris au chapitre 2, section 2.3.4.3, mais que nous avons modifié pour lui octroyer la gestion de l'ordre sur les variables (à la manière de PCS)

L'algorithme PCS n'utilise pas de connaissance, et doit donc permettre d'évaluer l'impact des connaissances sur l'apprentissage. L'algorithme PCSDC utilise quant à lui des connaissances certaines, et va donc permettre d'évaluer l'impact de l'incertitude sur l'apprentissage.

Nom	Nœuds	Arêtes	Paramètres	Référence
Alarm	37	46	509	BEINLIC, SUERMONDT, CHAVEZ et al. 1989
Andes	223	338	1157	CONATI, GERTNER, VANLEHN et al. 1997
Asia	8	8	18	LAURITZEN et SPIEGELHALTER 1988
Child	20	25	230	SPIEGELHALTER et COWELL 1992
Hailfinder	56	66	2656	AGRAMSON, J. BROWN, MURPHY et al. 1996
Insurance	27	52	984	BINDER, KOLLER, RUSSEL et al. 1997
Sachs	11	17	178	SACHS, PEREZ, PE'ER et al. 2005
Survey	6	6	21	SCUTARI et DENIS 2021
Win95pts	76	112	574	https://www.bnlearn.com/bnrepository/

TABLEAU 3.1 – Liste des réseaux bayésiens utilisés dans les expérimentations.

Les connaissances ont été générées automatiquement. Pour cela, nous avons sélectionné aléatoirement 50% des informations structurelles contenues dans le graphe

1. https://www.bnlearn.com/bnrepository/

original (présences et absences d'arêtes). Ces informations ont été traduites sous la forme de dépendance ou indépendance. Une partie de ces informations est ensuite utilisée comme connaissances, alors que l'autre partie est inversée pour être utilisée comme connaissance incorrecte. Cela permet de tester le comportement des algorithmes en présence d'erreurs dans les connaissances. Notons que pour permettre la comparaison avec l'algorithme PCSDC, nous nous sommes restreints à un seul expert. Comme précisé plus tôt, nous avons utilisé la valeur $P(\mathbb{I}) = 0.95$. Cela retranscrit le fait qu'en pratique, les réseaux bayésiens ont souvent des structures assez peu denses, c'est-à-dire qu'ils ont peu d'arcs, induisant ainsi beaucoup d'indépendances. Enfin, le seuil T a été fixé à 0.05. Concernant les connaissances utilisées pour PCSDC, il s'agit de connaissances purement structurelles (portant sur l'absence et la présence d'arêtes). Elles sont donc directement issues des informations sélectionnées aléatoirement.

PCSe, PCS et PCSDC sont comparés sur la base des squelettes qu'ils produisent. Pour cela, les squelettes obtenus sont comparés à celui du réseau bayésien original utilisé pour générer la base de données. Pour mesurer la qualité des squelettes, nous nous sommes appuyés sur des grandeurs couramment utilisées en apprentissage automatique, et plus précisément en classification. Lorsque l'objectif est de répartir des objets entre deux classes, une classe positive et négative, alors il est possible de distinguer quatre grandeurs :

— le nombre de **vrais positifs**, noté **VP**, décompte les éléments correctement classés dans la classe positive
— le nombre de **faux positifs**, noté **FP**, décompte les éléments incorrectement classés dans la classe positive
— le nombre de **faux négatifs**, noté **FN**, décompte les éléments incorrectement classés dans la classe négative
— le nombre de **vrais négatifs**, noté **VN**, décompte les éléments correctement classés dans la classe négative

Avec ces deux grandeurs, il est possible d'exprimer le **rappel** et la **précision**, qui représentent respectivement la capacité d'un algorithme à correctement classer les positifs, et sa capacité à éviter les négatifs. Ils se calculent de la manière suivante :

$$rappel = \frac{VP}{VP+FN} \text{ et } precision = \frac{VP}{VP+FP} \tag{3.22}$$

Enfin, ces deux grandeurs peuvent elles-mêmes être utilisées pour calculer une autre grandeur, appelée **F-score**. Cette grandeur a pour objectif d'agglomérer les deux précédentes, et se calcule de la manière suivante :

$$\text{F-score} = \frac{2 \times precision \times rappel}{precision + rappel} \tag{3.23}$$

Dans les expérimentations, les vrais positifs seront les arêtes existantes dans le graphe original et présentes dans le graphe appris, les faux positifs seront les arêtes absentes dans le graphe original et présentes dans le graphe appris, et les faux négatifs seront les arêtes présentes dans le graphe original et absentes dans le graphe appris.

Le F-score sera la mesure utilisée pour comparer les résultats de chaque algorithme.

Pour rendre la comparaison des résultats plus complète, nous avons fait varier les paramètres suivants :

— la taille de la base de données,
— la véracité, qui représente le pourcentage d'assertions correctes venant de l'expert,
— la confiance γ_k de l'expert pour donner une assertion correcte,
— le paramètre ρ_k pour la transformation de la confiance de l'expert (voir la figure 3.2 et l'équation (3.3)).

Enfin, pour chaque ensemble de paramètres ci-dessus, 100 itérations des expériences ont été menées et les tableaux 3.2 à 3.7 montrent les moyennes des F-scores obtenus sur ces 100 itérations.

Toutes les expériences ont été menées sur un ordinateur muni d'un processeur Intel Core i7 de 1.9GHz avec 16Go de mémoire RAM tournant sur Windows 10. Les algorithmes ont été implémentés en python et en utilisant la bibliothèque pyAgrum (DUCAMP, GONZALES et WUILLEMIN 2020).

3.5.2 Résultats

Taille	500			1000		
Véracité (%)	PCS	PCSDC	PCSe	PCS	PCSDC	PCSe
0	0.629	0.000	0.622	0.706	0.000	0.698
25	0.629	0.167	0.640	0.706	0.174	0.718
50	0.629	0.336	0.660	0.706	0.348	0.740
75	0.629	0.507	0.679	0.706	0.522	0.754
100	0.629	0.836	0.716	0.706	0.861	0.761
Taille	5000			10000		
Véracité (%)	PCS	PCSDC	PCSe	PCS	PCSDC	PCSe
0	0.837	0.000	0.828	0.888	0.000	0.882
25	0.837	0.174	0.842	0.888	0.174	0.884
50	0.837	0.348	0.846	0.888	0.348	0.888
75	0.837	0.522	0.856	0.888	0.521	0.893
100	0.837	0.918	0.842	0.888	0.939	0.902

TABLEAU 3.2 – F-scores de PCS, PCSDC et PCSe pour le réseau bayésien Alarm en fonction du pourcentage de véracité et de la taille de la base de données.

Commençons avec les tableaux 3.2 et 3.3. Ces tableaux présentent le F-score mesuré sur les squelettes obtenus avec les algorithmes PCS, PCSDC et PCSe en fonction de la véracité. Le tableau 3.2 a été obtenu avec le graphe Alarm et fait également varier la taille de la base de données utilisée. Le tableau 3.3 a pour sa part été obtenu avec une taille de base de données de 500 échantillons, mais fait varier le réseau bayésien

	PCS	PCSDC	PCSe	PCS	PCSDC	PCSe	PCS	PCSDC	PCSe
Véracité (%)		Alarm			Andes			Asia	
0	0.629	0.000	0.622	0.640	0.000	0.603	0.679	0.000	0.623
25	0.629	0.167	0.640	0.640	0.167	0.638	0.679	0.166	0.646
50	0.629	0.336	0.660	0.640	0.330	0.669	0.679	0.397	0.668
75	0.629	0.507	0.679	0.640	0.493	0.702	0.679	0.644	0.712
100	0.629	0.836	0.716	0.640	0.839	0.745	0.679	0.839	0.730
Véracité (%)		Child			Hailfinder			Insurance	
0	0.830	0.000	0.766	0.522	0.000	0.496	0.603	0.000	0.597
25	0.830	0.200	0.780	0.522	0.117	0.500	0.603	0.177	0.625
50	0.830	0.351	0.803	0.522	0.233	0.524	0.603	0.329	0.632
75	0.830	0.502	0.846	0.522	0.360	0.535	0.603	0.506	0.637
100	0.830	0.915	0.877	0.522	0.727	0.539	0.603	0.824	0.657
Véracité (%)		Sachs			Survey			Win95pts	
0	0.802	0.000	0.784	0.542	0.064	0.301	0.478	0.000	0.456
25	0.802	0.231	0.819	0.542	0.341	0.461	0.478	0.165	0.490
50	0.802	0.494	0.840	0.542	0.551	0.598	0.478	0.332	0.502
75	0.802	0.722	0.858	0.542	0.762	0.720	0.478	0.499	0.544
100	0.802	0.918	0.888	0.542	0.791	0.740	0.478	0.766	0.564

TABLEAU 3.3 – F-scores de PCS, PCSDC et PCSe pour différents réseaux bayésiens de référence avec des bases de données de 500 enregistrements en fonction du pourcentage de véracité.

utilisé pour générer les données. Pour les deux tableaux, le paramètre γ_k est fixé à 0.8, ce qui se traduit par de l'incertitude dans les connaissances. Le paramètre ρ_k a été fixé à 14.

Démarrons avec des observations générales concernant PCS et PCSDC. On peut voir que le F-score de l'algorithme PCS est constant en fonction de la véracité. Cela signifie que ce paramètre n'affecte pas l'algorithme, ce qui s'explique simplement par le fait que cet algorithme n'utilise pas de connaissance. Cette observation sera évidemment la même pour toutes les autres expérimentations de ce chapitre. Ensuite, on remarque également que l'algorithme PCSDC obtient un F-score de 0 lorsque la véracité des connaissances est à 0%. Cela signifie que le nombre de vrais positifs est systématiquement à 0 avec ces paramètres, et donc que l'algorithme ne peut trouver aucune arête correctement. L'explication est la suivante : lorsque la véracité est à 0%, alors les connaissances portant sur des absences d'arêtes recouvrent l'intégralité des arêtes existantes dans le vrai graphe. Ainsi, il ne peut y avoir aucun vrai positif. La figure 3.5 donne un exemple de ce cas. Dans le graphe de la figure 3.5c, une arête pleine symbolise une connaissance sur la présence de l'arête, alors qu'une arête en pointillé symbolise une connaissance sur son absence.

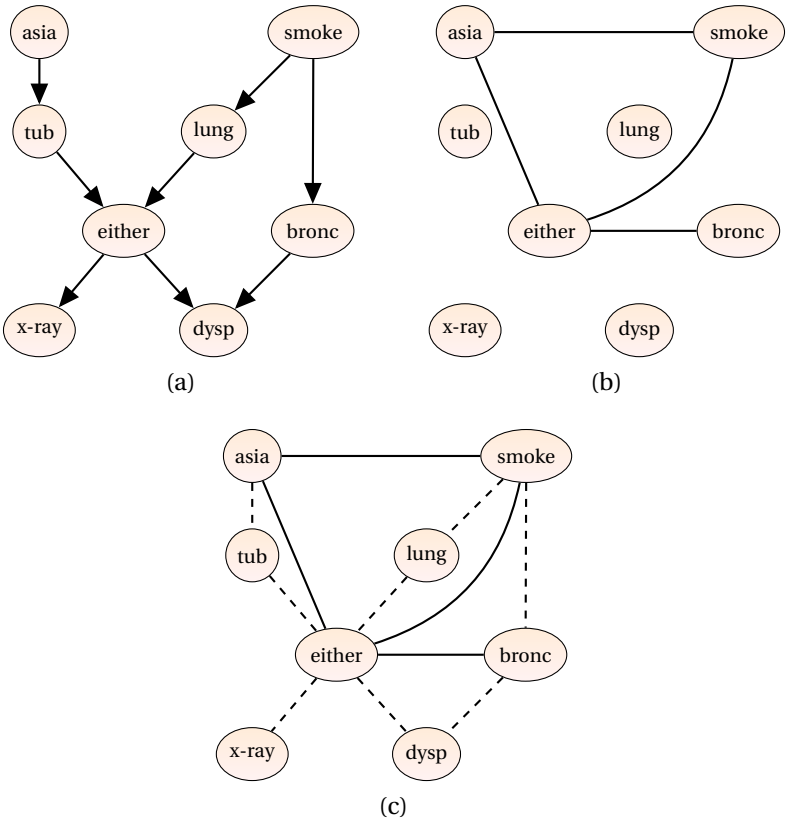


FIGURE 3.5 – (a) Le DAG du réseau bayésien Asia. (b) Le squelette obtenu avec l'algorithme PCSDC. (c) Les connaissances utilisées par l'algorithme PCSDC.

Passons à la comparaison entre les différents algorithmes. Lorsque la véracité est à 0%, on voit que PCS obtient un meilleur F-score que les autres algorithmes. Dans le tableau 3.2, la différence entre les F-scores se situe entre 0,006 et 0,009. Dans le tableau 3.3, elle varie entre 0,006 et 0,064 (sauf pour Survey, avec 0,241, qui est un très petit réseau bayésien). Ces différences sont peu significatives, mais soulignent tout de même le fait que PCS fait légèrement moins d'erreurs avec ce paramètre que PCSe. Cela s'explique simplement par le fait qu'aucune connaissance correcte n'a été donnée à PCSe à ce moment, ce qui implique plus d'erreurs et aucune assistance comme devraient le faire des connaissances correctes. Néanmoins, ces différences restent très faibles. Cela signifie que l'impact des connaissances erronées sur l'apprentissage est faible.

À partir d'une véracité de 25%, le F-score de PCSe dépasse dans certain cas celui de PCS. Encore une fois la différence n'est pas significative, entre 0.022 et 0.005 pour une véracité à 25% sur les deux tableaux et lorsque PCSe dépasse PCS. Mais cela souligne tout de même un effet positif lorsque des connaissances correctes sont fournies. Plus la véracité est élevée, plus l'impact positif des connaissances est mis en valeur. Par ailleurs, l'écart entre le F-score de PCSDC et celui de PCS ou PCSe est beaucoup plus grand lorsque la proportion d'erreurs est grande. Cela signifie que les erreurs ont

beaucoup plus d'impact sur PCSDC que sur PCSe. L'explication vient simplement du fait que PCSDC ne prend pas en compte l'incertitude. L'algorithme applique donc systématiquement les connaissances, même lorsque celles-ci sont fausses et que les données les contredisent clairement. Au contraire, PCSe prend en compte cette incertitude et permet de remettre en question les connaissances incorrectes au regard des données. Par exemple, quand les données montrent clairement une indépendance entre deux variables et que l'expert suppose sans certitude une dépendance, alors l'algorithme est capable de renoncer à suivre l'avis de l'expert. Ce mécanisme accorde à PCSe une robustesse face aux erreurs dans les connaissances.

Cependant, dès lors que la véracité des connaissances atteint 100% et donc qu'aucune erreur ne subsiste dans les connaissances, alors le F-score de PCSDC dépasse celui de PCSe et PCS. Cela signifie que les connaissances correctes ont un impact positif plus important sur l'apprentissage de PCSDC que sur celui de PCSe. L'explication vient une fois encore de l'incertitude des connaissances. De la même façon que pour des connaissances incorrectes, il se peut que PCSe remette en question des connaissances correctes au regard des données. C'est notamment le cas lorsque les données sont peu nombreuses ou de mauvaises qualités, et que les tests d'indépendance aboutissent à des conclusions erronées avec une p-valeur laissant peu de place au doute.

En conclusion, pour ces premiers tableaux, on voit que la prise en compte de l'incertitude des connaissances est à double tranchant : cela permet de diminuer l'impact des erreurs, mais donc également diminuer l'impact des connaissances correctes. Nous voyons par ailleurs que ces conclusions restent les mêmes en fonction du réseau bayésien appris ou de la taille de la base de données.

Taille	500			1000		
γ_k	PCS	PCSDC	PCSe	PCS	PCSDC	PCSe
0.5	0.629±0.031	0.629±0.031	0.629±0.031	0.706±0.041	0.706±0.041	0.706±0.041
0.6	0.629±0.031	0.507±0.007	0.639±0.032	0.706±0.041	0.522±0.000	0.721±0.040
0.7	0.629±0.031	0.507±0.007	0.657±0.031	0.706±0.041	0.522±0.000	0.737±0.041
0.8	0.629±0.031	0.507±0.007	0.679±0.034	0.706±0.041	0.522±0.000	0.754±0.038
0.9	0.629±0.031	0.507±0.007	0.695±0.033	0.706±0.041	0.522±0.000	0.767±0.037
Taille	5000			10000		
γ_k	PCS	PCSDC	PCSe	PCS	PCSDC	PCSe
0.5	0.837±0.024	0.837±0.024	0.837±0.024	0.888±0.014	0.888±0.014	0.888±0.014
0.6	0.837±0.024	0.522±0.001	0.844±0.024	0.888±0.014	0.521±0.002	0.891±0.013
0.7	0.837±0.024	0.522±0.001	0.852±0.024	0.888±0.014	0.521±0.002	0.893±0.012
0.8	0.837±0.024	0.522±0.001	0.856±0.022	0.888±0.014	0.521±0.002	0.893±0.013
0.9	0.837±0.024	0.522±0.001	0.857±0.023	0.888±0.014	0.521±0.002	0.890±0.014

TABLEAU 3.4 – F-scores de PCS, PCSDC et PCSe pour le réseau bayésien Alarm en fonction du paramètre γ_k et de la taille de la base de données.

Passons aux tableaux 3.4 et 3.5. Dans ceux-ci, nous avons fixé le pourcentage de

	PCS	PCSDC	PCSe	PCS	PCSDC	PCSe	PCS	PCSDC	PCSe
γ_k	Alarm			Andes			Asia		
0.5	0.629±0.031	0.629±0.031	0.629±0.031	0.640±0.010	0.640±0.010	0.640±0.010	0.679±0.073	0.679±0.073	0.679±0.073
0.6	0.629±0.031	0.507±0.007	0.639±0.032	0.640±0.010	0.493±0.003	0.659±0.011	0.679±0.073	0.644±0.068	0.694±0.068
0.7	0.629±0.031	0.507±0.007	0.657±0.031	0.640±0.010	0.493±0.003	0.683±0.013	0.679±0.073	0.644±0.068	0.711±0.072
0.8	0.629±0.031	0.507±0.007	0.679±0.034	0.640±0.010	0.493±0.003	0.702±0.011	0.679±0.073	0.644±0.068	0.712±0.084
0.9	0.629±0.031	0.507±0.007	0.695±0.033	0.640±0.010	0.493±0.003	0.708±0.011	0.679±0.073	0.644±0.068	0.705±0.087
γ_k	Child			Hailfinder			Insurance		
0.5	0.830±0.030	0.830±0.030	0.830±0.030	0.522±0.014	0.522±0.014	0.522±0.014	0.603±0.034	0.603±0.034	0.603±0.034
0.6	0.830±0.030	0.502±0.012	0.844±0.028	0.522±0.014	0.360±0.012	0.525±0.014	0.603±0.034	0.506±0.005	0.614±0.033
0.7	0.830±0.030	0.502±0.012	0.848±0.028	0.522±0.014	0.360±0.012	0.530±0.015	0.603±0.034	0.506±0.005	0.624±0.032
0.8	0.830±0.030	0.502±0.012	0.846±0.030	0.522±0.014	0.360±0.012	0.535±0.015	0.603±0.034	0.506±0.005	0.637±0.031
0.9	0.830±0.030	0.502±0.012	0.836±0.027	0.522±0.014	0.360±0.012	0.537±0.015	0.603±0.034	0.506±0.005	0.647±0.030
γ_k	Sachs			Survey			Win95pts		
0.5	0.802±0.048	0.802±0.048	0.802±0.048	0.542±0.134	0.542±0.134	0.542±0.134	0.478±0.028	0.478±0.028	0.478±0.028
0.6	0.802±0.048	0.722±0.047	0.820±0.048	0.542±0.134	0.762±0.078	0.599±0.147	0.478±0.028	0.499±0.002	0.497±0.028
0.7	0.802±0.048	0.722±0.047	0.840±0.049	0.542±0.134	0.762±0.078	0.648±0.141	0.478±0.028	0.499±0.002	0.520±0.031
0.8	0.802±0.048	0.722±0.047	0.858±0.050	0.542±0.134	0.762±0.078	0.720±0.126	0.478±0.028	0.499±0.002	0.544±0.030
0.9	0.802±0.048	0.722±0.047	0.867±0.046	0.542±0.134	0.762±0.078	0.764±0.097	0.478±0.028	0.499±0.002	0.562±0.030

TABLEAU 3.5 – F-scores de PCS, PCSDC et PCSe pour différents réseaux bayésiens de référence avec des bases de données de 500 enregistrements en fonction du paramètre γ_k.

véracité à 75% et avons laissé ρ_k à 14. Nous avons fait varier γ puis calculé le F-score pour le réseau bayésien Alarm en fonction de la taille de la base de données pour le tableau 3.4, et avons fait varier le réseau bayésien avec des bases de données de 500 échantillons pour le tableau 3.5. Nous pouvons d'abord remarquer trois choses. Premièrement, comme précédemment et pour les mêmes raisons, le F-score de PCS est constant. Deuxièmement, le F-score des trois algorithmes est identique lorsque $\gamma = 0,5$. Cela indique que les connaissances n'ont aucun impact sur l'apprentissage, puisque les algorithmes utilisant des connaissances arrivent aux mêmes résultats que celui n'en utilisant pas. Cela s'explique simplement par le fait qu'une connaissance avec $\gamma = 0,5$ signifie que l'expert ne sait pas si son assertion est vraie ou fausse, ce qui correspond donc à une absence de connaissance. Plus précisément, pour PCSe, $\gamma = 0.5$ implique que $\epsilon_0 = \epsilon_1$, et donc que la connaissance de l'expert n'est prise en compte, d'après l'équation 3.13. Enfin, le F-score de PCSDC augmente entre $\gamma = 0,5$ et $\gamma = 0,6$, puis reste constant pour $0,6 \leq \gamma \leq 0,9$. C'est simplement une conséquence de la prise en compte déterministe des connaissances de la part de PCSDC. On fait varier la certitude des connaissances, or celle-ci n'est pas utilisée par PCSDC, donc le F-score ne varie pas.

Comparons maintenant les différents algorithmes entre eux. Nous pouvons voir dans le tableau 3.4 que le F-score de PCSe est supérieur aux autres quelle que soit la taille de la base de données. En particulier, lors du passage de $\gamma = 0,5$ à $\gamma = 0,6$,

le F-score de PCSDC diminue alors que celui de PCSe augmente. Cela signifie que lorsqu'on utilise des connaissances partiellement fausses, il est plus intéressant de tirer profit de l'incertitude. Cela corrobore les conclusions faites précédemment : la prise en compte de l'incertitude rend PCSe plus robuste fasse aux erreurs. De plus, nous pouvons observer concernant PCSe que plus γ est élevé, plus le F-score l'est aussi. Cela s'explique par le fait qu'il y a, avec les paramètres utilisés ici, plus de connaissances correctes qu'incorrectes. Donc plus la certitude est haute, plus la partie correcte des connaissances aura un impact positif, qui contrebalancera l'impact de la partie incorrecte des connaissances. Dans le tableau 3.5, les mêmes conclusions peuvent être tirées pour presque tous les réseaux bayésiens.

De manière générale, on observe grâce à ces deux tableaux qu'en présence d'erreurs dans les connaissances utilisées par les algorithmes, il est avantageux de prendre en considération l'incertitude. Lorsque l'incertitude n'est pas disponible, nous voyons également que pour les grands réseaux bayésiens, il est plus intéressant de ne pas prendre du tout en compte les connaissances plutôt que de risquer d'appliquer des connaissances erronées.

γ_k	$\rho_k = 1$	$\rho_k = 5$	$\rho_k = 10$	$\rho_k = 15$	$\rho_k = 25$	$\rho_k = 50$
0.6	0.708±0.039	0.711±0.040	0.718±0.042	0.724±0.041	0.732±0.045	0.771±0.041
0.7	0.715±0.041	0.720±0.040	0.728±0.043	0.739±0.046	0.771±0.041	0.818±0.032
0.8	0.721±0.042	0.726±0.042	0.742±0.046	0.766±0.042	0.803±0.038	0.831±0.028
0.9	0.728±0.045	0.738±0.044	0.764±0.042	0.791±0.038	0.818±0.032	0.840±0.030
1.0	0.873±0.027	0.873±0.027	0.873±0.027	0.873±0.027	0.873±0.027	0.873±0.027

TABLEAU 3.6 – F-scores de PCSe pour le réseau bayésien Alarm en fonction de γ et ρ.

Le tableau 3.6 présente le F-score des résultats de l'algorithme PCSe uniquement. L'objectif est ici de montrer la variation du comportement de l'algorithme en fonction de la variation du paramètre ρ. Nous avons donc les résultats obtenus sur le réseau bayésien Alarm avec des bases de données de 1000 échantillons et une véracité des connaissances de 100%. On observe très clairement que, quelle que soit la valeur de γ utilisée, plus ρ est élevé, plus les résultats sont bons. Puisque toutes les connaissances sont correctes, cela signifie que plus ρ est élevé, plus ces connaissances ont un impact important sur l'apprentissage. L'explication se trouve grâce au graphique 3.2. Ce graphique montre l'évolution de la fonction φ en fonction de γ, avec différentes valeurs de ρ. On observe que plus la valeur de ρ est élevée, plus la croissance de la fonction $\varphi()$ est rapide. Ainsi, avec une même certitude donnée par l'expert, l'impact des connaissances sera bien plus fort avec une valeur ρ haute qu'avec une valeur basse. Ce paramètre se présente donc comme un levier supplémentaire permettant de modifier le comportement de l'algorithme et donc de l'adapter quelle que soit la situation.

Le tableau 3.7 montre le F-score de l'algorithme PCSe lorsque $\rho = 1000$ (une valeur élevée), ainsi que le F-score de l'algorithme PCSDC. Avec une telle valeur de ρ, la fonction $\varphi()$ présente une croissance très rapide. Ainsi, même avec une incertitude

très prononcée (γ proche de 0,5), les connaissances auront un impact fort sur l'apprentissage. En d'autres termes, l'incertitude des connaissances a moins d'impact, dans le sens où la conclusion des tests d'indépendances varient peu en fonction de l'incertitude lorsque celle-ci varie entre 0,6 et 0.9. C'est ce qu'on observe dans le tableau, où les résultats de PCSe et PCSDC sont très proches.

γ_k	PCSDC	PCSe avec $\rho_k = 1000$
0.6	0.861±0.027	0.854±0.028
0.7	0.861±0.027	0.854±0.028
0.8	0.861±0.027	0.854±0.028
0.9	0.861±0.027	0.854±0.028
1.0	0.861±0.027	0.873±0.027

TABLEAU 3.7 – F-scores de PCSe pour le réseau bayésien Alarm en fonction de γ_k et ρ_k.

3.6 Conclusion

Dans ce chapitre, nous avons vu un algorithme d'apprentissage de structure de réseaux bayésiens sous contraintes permettant l'intégration de connaissances d'experts incertaines. L'algorithme s'articule autour d'un test d'indépendance *G*2 modifié spécialement pour prendre en compte ces connaissances. Pour modifier ce test, nous l'avons modélisé sous la forme d'un réseau bayésien, ce qui nous a permis d'intégrer correctement les connaissances. Bien que l'algorithme soit fondé sur PC-stable, des spécificités permettant d'optimiser l'utilisation des connaissances ont été ajoutées.

Les expérimentations menées ont permis de montrer plusieurs points. D'abord, il est clair que l'utilisation des connaissances améliore la qualité des modèles appris. Ensuite, on peut également observer que l'utilisation de l'incertitude en présence d'erreurs dans les connaissances améliore également la qualité de l'apprentissage. Enfin, l'algorithme proposé est non seulement robuste face à une grande proportion d'erreurs dans la base de connaissances, mais permet en plus de s'adapter à diverses situations, ce qui est un avantage lorsqu'un agent humain est impliqué dans le processus d'apprentissage.

4 Algorithme hybride d'apprentissage de structure de réseaux bayésiens causaux à l'aide de connaissances incertaines

Sommaire

4.1 Introduction . . . 121
4.2 Hypothèses . . . 123
4.2.1 Hypothèses d'apprentissage . . . 123
4.2.2 Les connaissances . . . 123
4.3 Algorithme . . . 125
4.3.1 Structure générale . . . 125
4.3.2 Étape sous contraintes . . . 126
4.3.3 Étape à base de score . . . 131
4.3.4 Étape interactive . . . 135
4.4 Expérimentations . . . 139
4.4.1 Paramètres expérimentaux . . . 139
4.4.2 Résultats . . . 141
4.5 Conclusion . . . 149

4.1 Introduction

Dans le chapitre précédent, nous avons vu un nouvel algorithme sous contraintes permettant d'utiliser des connaissances d'experts à propos de dépendances et d'indépendances pour améliorer l'apprentissage de la structure. Une des limites des algorithmes sous contraintes, qui reposent essentiellement sur des tests d'indépendances, est leur fragilité face à des données peu nombreuses ou de mauvaise qualité. Les connaissances utilisées dans le cadre de l'algorithme PCSe permettent de repousser cette limite.

De surcroît, les algorithmes sous contraintes possèdent une autre limite importante, cette fois-ci en terme d'apprentissage causal. En effet, comme beaucoup d'algorithmes sous contraintes, PC ne peut apprendre la structure causale (le squelette et

les orientations causales de chaque arête) que jusqu'au CPDAG. À la fin, il reste donc potentiellement de nombreuses arêtes non orientées. Or, de par la nature des connaissances utilisées dans PCSe, celles-ci ne permettent pas d'outrepasser cette limite. Pour aller au-delà, il est donc nécessaire d'utiliser une autre forme de connaissance.

En résumé, lorsque l'objectif est d'apprendre un réseau bayésien causal, deux problèmes surviennent : la qualité du CPDAG trouvé (au niveau de son squelette et de ses orientations), et l'incomplétude de ses orientations causales. Concernant le premier problème, nous avons déjà vu, grâce à l'algorithme PCSe, que des connaissances d'experts sur les indépendances et dépendances pouvaient être d'une aide précieuse. Mais il est tout à fait possible d'utiliser des connaissances de natures différentes, par exemple, des connaissances structurelles qui portent directement sur l'existence des arêtes et sur leurs orientations. Une autre solution, compatible avec la première, est l'utilisation d'un algorithme à base de score en plus de l'étape sous contraintes. L'algorithme devient alors hybride, et la deuxième étape permet de parfaire l'apprentissage du CPDAG.

L'incomplétude causale du résultat est un problème plus difficile. En effet, un algorithme sous contraintes ne peut théoriquement pas trouver la structure causale complète avec certitude, et il en est de même pour les méthodes à base de score. Ainsi, un algorithme hybride n'est d'aucune aide pour ce problème. En fait, en l'absence d'hypothèse permettant d'utiliser des algorithmes d'inférence causale, il est même théoriquement impossible d'orienter les arêtes de la structure au-delà du CPDAG juste à partir des données, quelque soit l'approche utilisée (PEARL 2009). Pour aller plus loin, il est donc nécessaire d'avoir recours à de nouvelles sources d'informations, par exemple des connaissances d'expert portant directement sur l'orientation des relations causales.

L'algorithme d'apprentissage présenté ci-dessous doit donc permettre de combiner des données et des connaissances pour en tirer le plus d'informations possible. À l'issue des deux premières étapes de l'algorithme, nous obtiendrons un graphe causal partiellement orienté. Bien que le graphe ne soit pas complet, il peut tout de même être interprété, notamment par l'expert qui a émis les connaissances initiales. Nous pensons que cette confrontation peut être vertueuse pour l'expert, qui pourrait alors nous apporter des connaissances supplémentaires pour le processus d'apprentissage de la structure. Pour capter ces connaissances, nous proposons une ultime étape reposant sur l'interaction. Dans celle-ci, l'expert, confronté au graphe partiel, sera soumis à une série de questions dont les réponses doivent permettre de compléter les orientations du graphe de manière exhaustive. Cette étape a un double objectif d'optimisation : d'abord l'optimisation de l'élicitation des connaissances de l'expert, pour obtenir le plus de connaissances et ainsi construire le graphe le plus correct possible. Puis une optimisation en termes de nombre de questions, car l'intervention d'un expert peut s'avérer coûteuse pour une entreprise, et il est donc intéressant de minimiser le nombre de questions nécessaires pour compléter le graphe.

Dans ce chapitre, nous verrons donc le deuxième algorithme mis au point lors de cette thèse. Il s'agit d'un algorithme hybride d'apprentissage de réseaux bayésiens causaux utilisant des connaissances d'expert incertaines. Dans la suite de ce chapitre,

nous aborderons d'abord, comme pour le chapitre précédent, les hypothèses et la nature précise des connaissances utilisées. Puis nous verrons comment nous avons adapté l'algorithme PC et l'algorithme de Recherche Gloutonne pour prendre en compte les connaissances des experts sur la causalité. Nous décrirons également la façon avec laquelle une étape interactive a été mise en place pour optimiser l'utilisation des connaissances. Enfin, nous discuterons des résultats d'une série d'expérimentations venant confirmer l'efficacité de l'algorithme.

4.2 Hypothèses

4.2.1 Hypothèses d'apprentissage

Commençons avec les hypothèses utilisées pour cet algorithme. Ce dernier se place dans la catégorie des algorithmes hybrides, car il est constitué d'une étape sous contraintes, suivie d'une étape à base de score. L'algorithme doit donc suivre les hypothèses classiques de fidélité et de suffisance causale. Rappelons que la première stipule que la distribution de probabilité des données d'entrée admet l'existence d'une P-map. La deuxième nous permet d'être sûr qu'il n'existe pas de variable cachée qui rendrait dépendantes deux variables sans que cette dépendance puisse être causalement expliquée avec les variables connues. Il est important de souligner le fait que nous supposons que le modèle n'admet aucune variable latente.

Notons également que l'hypothèse de fidélité va être en partie relâchée pour l'étape sous contraintes. En effet, comme expliqué dans le cadre de l'algorithme CPC (2.3.3.1), il est possible d'employer une hypothèse moins forte (donc moins contraignante) durant l'exécution d'un algorithme similaire à PC. Nous n'utiliserons durant l'étape sous contraintes que l'hypothèse de fidélité d'adjacence, en laissant donc de côté l'hypothèse de fidélité d'orientation. Cette réduction des hypothèses nous permet, à cette étape, d'améliorer les résultats mais également d'être plus prudent lors de l'orientation des arêtes. Ce dernier point est particulièrement important dans le cadre d'un apprentissage causal.

Enfin, nous nous placerons dans le cadre de données discrètes, nous faisons donc l'hypothèse que les données utilisées sont de la bonne nature.

4.2.2 Les connaissances

Dans le cadre d'un apprentissage causal, nous supposerons qu'un expert est capable de nous fournir des connaissances incertaines sur l'orientation des relations causales. Plus précisément, l'incertitude porte sur l'orientation uniquement. En d'autres termes, lorsque l'expert nous donne des informations sur la relation entre deux variables, nous supposerons que ces deux variables sont connectées grâce à une relation causale directe. L'existence de cette relation causale sera considérée comme certaine. Seule son orientation est sujette à l'incertitude émise par l'expert.

La nature particulière des connaissances utilisées est tirée d'expériences concrètes.

En effet, en dehors du cadre de la thèse, lors d'interactions avec des experts travaillant sur les éoliennes, il a pu être constaté que des experts pouvaient nous fournir des connaissances sur des relations causales directes dont l'existence était certaine, mais pour lesquelles l'orientation était incertaine. C'est par exemple le cas pour les variables "Température de la nacelle" et "Température du roulement principal", représentées dans la figure 4.1. La relation causale entre ces deux variables est connue, mais son orientation est incertaine.

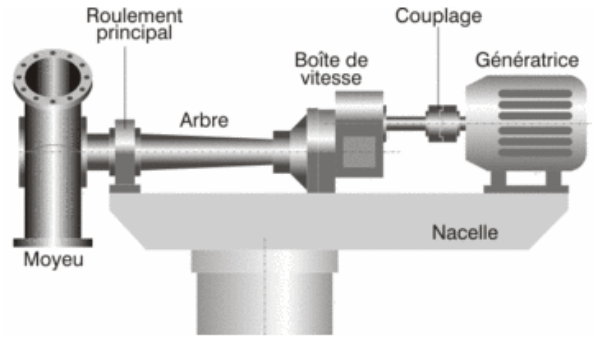


FIGURE 4.1 – Représentation schématique de la transmission d'une éolienne.

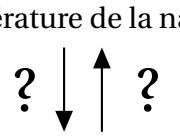


FIGURE 4.2 – Illustration de l'incertitude de l'orientation causale entre deux variables.

Formalisons un peu plus ces connaissances. Elles sont exprimées en tant qu'affirmations de la forme "Je sais qu'il y a une relation causale directe entre X et Y mais je ne suis pas totalement certain de l'orientation". L'incertitude sur l'orientation de la relation causale est exprimée par l'expert sous la forme d'une probabilité. Ainsi, l'ensemble des connaissances **K** exprimées par l'expert peut être représenté par un ensemble de couples $(X \rightarrow Y, p_{XY})$ où p_{XY} est la probabilité donnée par l'expert que l'orientation causale est $X \rightarrow Y$. On suppose que $p_{XY} \geq 0,5$, c'est-à-dire que si l'expert pense qu'il y a une relation causale entre X et Y et que la probabilité qu'il donne est inférieure à $0,5$, alors il doit également penser qu'il y a une probabilité $p_{YX} = 1 - p_{XY} \geq 0.5$ que Y cause X, et seulement $(Y \rightarrow X, p_{YX})$ sera sauvegardé dans **K**. Nous supposerons également que l'ensemble des connaissances est cohérent, ce qui signifie que les orientations causales proposées par l'expert ne forment pas de circuit lorsqu'elles sont combinées.

De plus, les connaissances seront divisées en deux ensembles :

— $\mathbf{K}_a = \{(X \rightarrow Y, p_{XY}) \in \mathbf{K} : p_{XY} = 1\}$, qui représente les connaissances pour lesquelles le paramètre d'incertitude est à 1, ce qui signifie qu'il n'y a aucune incertitude de la part de l'expert sur l'orientation de la relation causale
— $\mathbf{K}_e = \mathbf{K} \backslash \mathbf{K}_a$, qui représente toutes les connaissances pour lesquelles il y a effectivement une incertitude sur l'orientation.

On notera également $G_a = (\mathbf{V}, \mathbf{E}_a)$ avec $\mathbf{E}_a = \{(X \rightarrow Y | (X \rightarrow Y, p_{XY}) \in \mathbf{K}_a\}$ et $G_e = (\mathbf{V}, \mathbf{E}_e)$ avec $\mathbf{E}_e = \{(X - Y | (X \rightarrow Y, p_{XY}) \in \mathbf{K}_e\}$.

Enfin, les connaissances décrites ci-dessus ne sont pas les seules à être utilisées dans cet algorithme. En effet, la spécificité de ce dernier est la présence d'une étape interactive. Durant les deux premières étapes, les connaissances sont obtenues *a priori*. Cependant, les connaissances utilisées durant l'étape interactive sont demandées à l'expert durant l'exécution de l'algorithme. Nous supposerons que cette différence ne change pas la nature des connaissances. Nous utiliserons donc la même formalisation pour l'intégralité de l'algorithme.

4.3 Algorithme

4.3.1 Structure générale

Passons maintenant à l'algorithme. Comme dit précédemment, l'algorithme est divisé en trois étapes :

— une étape sous contraintes, reprenant donc les méthodes des approches sous contraintes
— une étape à base de score, utilisant une approche à base de score et exploitant le résultat de la première étape
— une étape interactive, où le graphe sera affiné avec l'aide de l'expert qui interagira en direct.

Il s'agit donc d'un algorithme hybride. L'objectif ici est de diversifier les méthodes utilisées pour optimiser l'exploitation des données et des connaissances à notre disposition.

La première étape va exploiter les données grâce à des tests d'indépendance, et donc construire le squelette du graphe. Elle permet d'aboutir à une première estimation des v-structures présentes dans le graphe. De manière parallèle, cette étape permet également d'exploiter les connaissances pour le squelette, les premières v-structures et orientations.

La deuxième étape est à base de score. Son objectif est d'affiner le graphe partiel et de trouver la classe d'équivalence de Markov exacte grâce à l'utilisation d'un score. Les connaissances sont exploitées directement grâce au score en utilisant l'algorithme GSA dont nous avons discuté à la sous-section 2.3.4 du chapitre 2.

Enfin, la dernière étape doit servir à compléter les orientations du graphe avec l'objectif d'arriver si possible à un graphe entièrement orienté. Pour cela, une procédure impliquant directement l'expert est répétée. Cette étape se termine lorsque le graphe est entièrement orienté ou que l'expert n'est plus capable de répondre aux questions.

Ainsi, tout au long de l'algorithme, un graphe est construit, modifié et complété. Pour plus de clarté, détaillons tout de suite les entrées et les sorties de chaque étape. La première étape prend en entrées les données et les connaissances *a priori*. Sa sortie est un PDAG, donc un graphe partiellement orienté. Ce PDAG ne correspond pas à une classe d'équivalence précise, toutes les indépendances conditionnelles n'étant pas encore identifiées. La deuxième étape prend en entrées les données, les connaissances et le PDAG obtenu à la fin de la première étape. En sortie, nous restreindrons le résultats à un nouveau PDAG, mais qui correspond cette fois-ci à un CPDAG avec potentiellement des orientations supplémentaires. Ce PDAG contiendra ainsi toutes les indépendances nécessaires à la construction du modèle. Enfin, la dernière étape prend en entrées le PDAG obtenu à la fin de la deuxième étape ainsi que les réponses données par l'expert, et donne en sortie un DAG, donc un graphe complètement orienté, ou un PDAG si l'expert ne peut pas répondre à toutes les questions.

Découvrons maintenant les différentes étapes en détails.

4.3.2 Étape sous contraintes

Commençons avec la première étape. Pour celle-ci, nous avons utilisé un algorithme qui reprend les trois sous-étapes de PC :

— l'apprentissage du squelette. En raison de la présence des connaissances, il s'agit ici de l'apprentissage d'un PDAG
— l'orientation des v-structures
— la propagation des orientations

Commençons avec l'apprentissage du PDAG. Nous avons utilisé ici l'algorithme PC stable, pour l'avantage qu'il donne sur l'ordre d'exécution des tests d'indépendance. De plus, nous avons exploité les connaissances de la même manière que l'algorithme PCDC. Nous partons donc d'un graphe contraint avec les connaissances à notre disposition, c'est-à-dire un graphe complet avec les orientations de G_a. Les arêtes présentes dans les graphes G_a et G_e impliquent l'existence de relations causales entre les nœuds associés, et cette existence est certaine. Il ne doit donc pas être possible de supprimer ces arcs/arêtes.

En plus de l'initialisation du graphe, on restreint, comme pour PCDC, les ensembles de conditionnement des tests d'indépendance. Rappelons que X est indépendant de ses non-descendants sachant ses parents. Donc pour tester l'existence d'un ensemble $\mathbf{Z}$ tel que $X \perp Y|\mathbf{Z}$, il suffit de tester si $X \perp Y|Pa(X)$ ou $X \perp Y|Pa(Y)$. Puisque les connaissances nous permettent potentiellement de savoir quelles sont les variables qui ne seront pas les parents d'une autre, il est possible de restreindre les ensembles de conditionnement utilisés pour les tests d'indépendance. Ainsi, pour le test de l'indépendance entre les variables X et Y sachant un ensemble de variables $\mathbf{Z}$, seront exclues de $\mathbf{Z}$ les variables appartenant à $\text{Desc}_{\mathbf{K}_a}(X)$, où $\text{Desc}_{\mathbf{K}_a}(X)$ représente l'ensemble des variables pour lesquelles l'expert sait avec certitude qu'elles sont des descendantes de X. Ce qui a été détaillé ci-dessus est décrit dans l'algorithme 14.

Après l'apprentissage du PDAG vient l'orientation des v-structures. C'est une étape cruciale, puisque les v-structures sont la base de l'apprentissage causal avec les mé-

Algorithm 14 Apprentissage du PDAG

Entrée : une base de données **S**, un ensemble de connaissances d'expert **K**
Sortie : Un graphe partiellement orienté G, des ensembles de séparation $Sepset$

1: $\mathbf{V} \leftarrow$ toutes les variables/colonnes de $\mathbf{S}$; $\mathbf{E} \leftarrow \{X - Y : X, Y \in \mathbf{V}\}$; $G \leftarrow$ graphe complet $(\mathbf{V}, \mathbf{E})$
2: $\mathbf{E_K} \leftarrow \{A - B : (A \rightarrow B, p_{AB}) \in \mathbf{K}\}$
3: $m \leftarrow -1$; $Sepset \leftarrow \emptyset$
4: **Répéter**
5: $m \leftarrow m + 1$
6: **Pour tout** X dans **V Faire**
7: $\mathbf{adj}_X \leftarrow \{Y \in \mathbf{V} : X - Y \in \mathbf{E}\}$
8: **Fin Pour**
9: **Pour tout** $(X, Y) \in \mathbf{V}^2$ tel que $X - Y \in \mathbf{E} \backslash \mathbf{E_K}$ et $|\mathbf{adj}_X| \geq m + 1$ **Faire**
10: **Répéter**
11: Choisir un nouvel ensemble $\mathbf{Z} \subseteq \mathbf{adj}_X \backslash (\{Y\} \cup \mathrm{Desc}_{\mathbf{K}_a}(X))$ tel que $|\mathbf{Z}| = m$
12: **Si** $X \perp Y | \mathbf{Z}$ **Alors**
13: Supprimer l'arête $X - Y$ de $\mathbf{E}$; $Sepset_{X,Y} \leftarrow \mathbf{Z}$
14: **Fin Si**
15: **Jusqu'à ce que** tous les ensembles $\mathbf{Z} \subset adj(X)$ tels que $|\mathbf{Z}| = m$ aient été testés ou qu'une indépendance ait été trouvée
16: **Fin Pour**
17: **Jusqu'à ce que** toutes les paires ordonnées (X, Y) adjacentes telles que $|adj(X) \backslash Y| \geq m$ aient été testées
18: **Pour tout** $(X \rightarrow Y, p_{XY}) \in \mathbf{K}_a$ **Faire**
19: remplacer $X - Y$ par $X \rightarrow Y$ dans $\mathbf{E}$
20: **Fin Pour**
21: **Retourner** $G, Sepsets$

thodes sous contraintes. Ici, nous souhaitons un apprentissage précis, c'est-à-dire un apprentissage avec peu d'arcs mal orientés. Dans cet objectif, nous avons mis en place une approche prudente des orientations de v-structures. Cette prudence passe par deux outils : l'identification de **conflits** et l'algorithme CPC.

Le premier outil est l'identification de **conflits** dans le graphe. Un conflit est une arête pour laquelle deux orientations sont possibles, chaque orientation étant poussée par l'orientation d'une v-structure potentielle ou par des connaissances de l'expert. On peut identifier deux types de conflits. Les conflits entre deux v-structures potentielles et les conflits entre une v-structure potentielle et une connaissance. Ces deux types de conflits peuvent être exprimés sous la forme de deux ensembles :

— $\boldsymbol{\sigma}_K$: les v-structures qui sont en désaccords avec des connaissances
— $\boldsymbol{\sigma}_V$: les v-structures qui sont en désaccords avec d'autres v-structures

Exemple 4.3.1 *Par exemple, pour trois variables X, Y et Z, l'expert peut exprimer une opinion incertaine sur l'orientation $Y \rightarrow X$ alors que l'analyse des ensembles de*

séparation montre une v-structure $X \rightarrow Y \leftarrow Z$. Cette dernière v-structure sera alors dans l'ensemble $\boldsymbol{\sigma}_K$.

De même, pour quatre autres variables A, B, C et D, l'analyse des ensembles de séparation peut montrer les v-structures $A \rightarrow B \leftarrow C$ et $B \rightarrow C \leftarrow D$. Les deux v-structures iront alors dans l'ensemble $\boldsymbol{\sigma}_V$.

En présence de conflit, il est difficile de décider immédiatement de la bonne orientation. Cependant, ne pas mettre d'orientation du tout est risqué. En effet, durant la propagation des orientations, une structure $X - Z - Y$ sans orientation peut être mal orientée par une v-structure voisine. Pour palier ce problème, nous avons utilisé des **orientations temporaires**. Lorsqu'une v-structure est conflictuelle, alors ses arcs sont orientés temporairement. Il est important de noter qu'un arc peut avoir simultanément deux orientations temporaires opposées. Cela permet de continuer à envisager les deux orientations, sans avoir besoin de décider tout de suite.

Le deuxième outil est la méthode de CPC décrite dans la sous-section 2.3.3.1 du chapitre 2. Rappelons que cette méthode consiste à vérifier la présence du nœud central d'une structure $X - Z - Y$ dans les sous-ensembles $\mathbf{Z} \subseteq \mathbf{V} \setminus \{X, Y\}$ tel que $X \perp Y | \mathbf{Z}$. Si Z n'appartient à aucun ensemble, alors on peut orienter la v-structure. Si Z appartient à tous les sous-ensembles, alors on abandonne la v-structure. Et si Z appartient à certains sous-ensembles, mais pas tous, alors la structure doit être catégorisée comme "infidèle". La dernière condition implique que la sortie de l'algorithme n'est pas un simple PDAG ou CPDAG, puisque certaines structures peuvent être "marquées".

Dans notre cas, nous avons décidé de ne pas garder les marques comme proposé par les auteurs de CPC. À la place, nous avons utilisé la notion d'orientation temporaire définie plus tôt. Lorsqu'une v-structure "infidèle" est détectée, celle-ci est orientée temporairement. Cependant, s'il s'avère que cette v-structure est conflictuelle en plus d'être infidèle, alors les orientations temporaires de la v-structure sont supprimées, et la v-structure est abandonnée. Ainsi, dans la logique de prudence que nous avons adopté pour cet algorithme, nous n'appliquons pas des v-structures pour lesquelles des doutes trop importants subsistent.

Ces méthodes sont décrites dans l'algorithme 15. À la ligne 4, nous utilisons le méta-symbole $\circ$ pour représenter l'absence ou la présence de $\blacktriangleright$ à l'extrémité d'une arête. Ainsi, $X - \circ Y$ représente $X - Y$ ou $X \rightarrow Y$.

Enfin, la dernière sous-étape de l'algorithme PC est la propagation des orientations. Rappelons que cette propagation se fait par l'application de quatre règles issues des travaux de Meek (MEEK 1995), que l'on peut retrouver dans le chapitre 2 (voir 2.3.3.1).

La principale difficulté pour appliquer ces règles au graphe que l'on manipule est la gestion des orientations temporaires. Voyons dans un premier temps les conditions d'application de ces règles. Lorsqu'un arc est nécessaire pour l'application d'une règle, alors cette règle est applicable, quelle que soit la marque "temporaire" ou "définitive" de l'arc. De plus, si l'orientation est temporaire et qu'une orientation temporaire existe dans l'autre sens, alors la règle est toujours applicable. Dans un deuxième temps, voyons l'application de ces règles. L'arête qui s'apprête à être orientée par l'application d'une règle peut être une arête (sans orientation) ou un arc orienté

Algorithm 15 Orientation des v-structures

Entrée : une base de données **S**, un ensemble de connaissances d'experts **K**, un graphe partiellement orienté $G = (\mathbf{V}, \mathbf{E})$
Sortie : Un graphe partiellement orienté G, avec des orientations temporaires et définitives

1: **Pour tout** nœud X in **V Faire**
2: $\mathbf{adj}_X \leftarrow \{Y \in \mathbf{V} : X - Y \in \mathbf{E}\}$
3: **Fin Pour**
4: $\boldsymbol{\sigma} \leftarrow \{(X,Z,Y) \in \mathbf{V}^3 : X \multimap Z \circ\!\!-\!\!\circ Y \in \mathbf{E}, X \circ\!\!-\!\!\circ Y \notin \mathbf{E}$ et il existe $\mathbf{T} \subseteq \mathbf{adj}_X \backslash \{Z\}$ ou $\mathbf{T} \subseteq \mathbf{adj}_Y \backslash \{Z\}$ tel que $X \perp Y | \mathbf{T}\}$
5: $\boldsymbol{\sigma}_K \leftarrow \{(X,Z,Y) \in \boldsymbol{\sigma} : (Z \rightarrow X, p_{ZX}) \in \mathbf{K}_e$ ou $(Z \rightarrow Y, p_{ZY}) \in \mathbf{K}_e\}$
6: $\boldsymbol{\sigma}_V \leftarrow \{(X,Z,Y) \in \boldsymbol{\sigma}$: il existe $X' \in \mathbf{V}$ tel que $(Z,X,X') \in \boldsymbol{\sigma}$ ou $(Z,Y,X') \in \boldsymbol{\sigma}\}$
7: **Pour tout** $(X,Z,Y) \in \boldsymbol{\sigma}$ **Faire**
8: **Si** il existe $\mathbf{T} \subseteq \mathbf{adj}_X$ ou $\mathbf{T} \subseteq \mathbf{adj}_Y$ tel que $Z \in \mathbf{T}$ et $X \perp Y | \mathbf{T}$ **Alors**
9: **Si** $(X,Z,Y) \notin \boldsymbol{\sigma}_K$ et $(X,Z,Y) \notin \boldsymbol{\sigma}_V$ **Alors**
10: Supprimer les arêtes $X - Z$ et $Z - Y$ de **E** (si elles existent toujours)
11: Ajouter les arcs $X \rightarrow Z$ et $Z \leftarrow Y$ dans **E** et les marquer comme "temporaires"
12: **Fin Si**
13: **Sinon Si** $(X,Z,Y) \in \boldsymbol{\sigma}_K$ or $(X,Z,Y) \in \boldsymbol{\sigma}_V$ **Alors**
14: Supprimer les arêtes $X - Z$ et $Z - Y$ de **E** (si elles existent toujours)
15: Ajouter les arcs $X \rightarrow Z$ et $Z \leftarrow Y$ dans **E** et les marquer comme "temporaires"
16: **Sinon Si** $(X,Z,Y) \notin \boldsymbol{\sigma}_K$ et $(X,Z,Y) \notin \boldsymbol{\sigma}_V$ **Alors**
17: Supprimer les arêtes $X - Z$ et $Z - Y$ de **E** (si elles existent toujours)
18: Ajouter les arcs $X \rightarrow Z$ et $Z \leftarrow Y$ dans **E** et les marquer comme "définitifs"
19: **Fin Si**
20: **Fin Pour**
21: **Pour tout** $(X \rightarrow Y, p_{XY}) \in \mathbf{K}_a$ **Faire**
22: Marquer $X \rightarrow Y$ comme définitif
23: **Fin Pour**
24: **Retourner** Un graphe partiellement orienté G et les marques "temporaires" ou "définitives" sur les arcs

temporairement. Grâce à ces modifications, il est possible d'orienter prudemment les arêtes du graphe.

Ainsi, nous avons un PDAG avec des arcs marqués définitivement ou temporairement (algorithme 15 l. 24). Nous allons utiliser une dernière procédure (algorithme 16) pour obtenir un graphe avec des arcs orientés définitivement uniquement. Dans cette procédure, nous supprimons les orientations temporaires pour lesquelles il existe des doutes légitimes, et rendre définitives les orientations temporaires pour lesquelles il n'y a pas de doute. La méthode employée est simple : s'il existe deux orientations temporaires opposées entre deux nœuds, alors nous supprimons les deux orientations et nous ne gardons qu'une arête. S'il n'existe qu'une seule orientation temporaire entre deux nœuds, alors nous pouvons considérer que cette orientation est légitime,

et nous pouvons la transformer en un arc définitif. Évidemment, à ce niveau, si l'ajout d'un arc définitif entraîne la création d'un circuit, alors c'est la contrainte d'acyclicité qui prime et l'arc opposé est ajouté. Ces explications sont résumées dans le tableau 4.1. Dans ce tableau, les arcs en pointillé symbolisent des orientations temporaires. L'arc bidirectionnel en pointillé représente quant à lui la présence de deux orientations temporaires opposées entre une paire de nœuds.

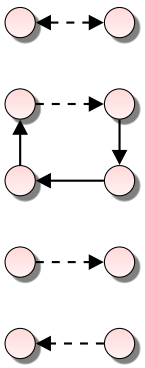
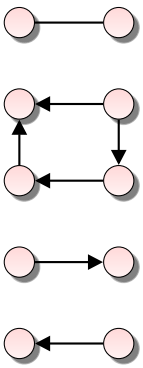
Structure initiale	Structure finale
	

TABLEAU 4.1 – Modifications des orientations temporaires.

Algorithm 16 Conversion du PDAG marqué en PDAG classique.

Entrée : un graphe partiellement orienté $G = (\mathbf{V}, \mathbf{E})$ et les marques "temporaire" ou "définitif" sur les arcs
Sortie : Un graphe partiellement orienté G

1: **Pour tout** $X, Y \in \mathbf{V}^2$ tels que $X \rightarrow Y \in \mathbf{E}$ et $X \leftarrow Y \in \mathbf{E}$ et sont marqués comme "temporaire" **Faire**
2: Remplacer $X \rightarrow Y$ et $X \leftarrow Y$ par $X - Y$ dans $\mathbf{E}$
3: **Fin Pour**
4: **Pour tout** $X, Y \in \mathbf{V}^2$ tels que $X \rightarrow Y \in \mathbf{E}$ et sont marqués comme "temporaire" **Faire**
5: Marquer $X \rightarrow Y$ comme définitif
6: **Si** $X \rightarrow Y$ crée un circuit d'arcs définitifs **Alors**
7: Remplacer $X \rightarrow Y$ par $Y \rightarrow X$ dans $\mathbf{E}$ et le marquer comme définitif
8: **Fin Si**
9: **Fin Pour**
10: **Retourner** Un graphe partiellement orienté G

La procédure décrite dans l'algorithme 16 possède cependant deux défauts, et il est important de les mentionner. D'abord, il est nécessaire d'éviter la création d'un circuit

dans le graphe. Pour cette raison, la procédure nous incite à donner une orientation à une arête qui est l'inverse de l'orientation temporaire initiale. Une conséquence possible est la création d'une nouvelle v-structure, qui n'était pas présente à l'origine. De plus, cela provoque également un deuxième défaut, qui est l'apparition d'une dépendance du résultat à l'ordre de considération des variables. En effet, lorsqu'un circuit est évité, la v-structure qui peut en résulter pourrait apparaître à différents endroits dans le graphe selon l'ordre dans lequel on s'intéresse aux couples des variables pour supprimer les orientations temporaires. Heureusement, comme nous le verrons plus tard dans ce chapitre, ces défauts ne nuisent pas aux performances globales de l'algorithme.

À l'issue de cette étape, nous obtenons un PDAG avec des arcs définitifs uniquement. Grâce à cette méthode sous contraintes, nous avons pu exploiter les indépendances conditionnelles pour obtenir un ensemble partiel et préliminaire des indépendances conditionnelles du graphe ainsi que ses orientations causales. La prochaine étape doit permettre d'affiner le PDAG pour obtenir toutes les indépendances conditionnelles et toutes les orientations causales liées à ces indépendances.

4.3.3 Étape à base de score

Passons maintenant à l'étape à base de score. Au début de cette étape, nous avons un PDAG qui ne représente pas l'intégralité des indépendances conditionnelles de la distribution, et qui n'a pas toutes les orientations causales nécessaires. De plus, ce PDAG n'a que des orientations définitives. À la fin de cette étape, nous obtiendrons ce qu'on appellera un PDAG$^+$.

Définition 4.3.1 *Soit $G = (\mathbf{V}, \mathbf{E})$ un CPDAG, on dit que $G' = (\mathbf{V}, \mathbf{E}')$ est un PDAG$^+$ si et seulement si c'est un graphe ayant toutes les orientations d'un CPDAG et donc retranscrivant toutes les indépendances conditionnelles d'une distribution, mais pouvant avoir des orientations causales supplémentaires.*

Cela signifie qu'à la fin de cette étape, nous aurons toutes les indépendances conditionnelles de la distribution et toutes les orientations causales liées aux v-structures du CPDAG. De plus, nous pourrons également avoir quelques orientations supplémentaires liées aux connaissances de l'expert ainsi que les propagations de ces orientations par les règles de Meek.

Cette étape est à base de score, cela signifie que nous allons utiliser un score pour affiner le PDAG. Cependant, cela nécessite d'avoir un DAG. Nous allons donc commencer par compléter arbitrairement le PDAG obtenu à l'issue de l'étape sous contraintes pour obtenir un DAG sur lequel pourra être calculé un score. Pour cela, nous orientons chaque arête, soit en suivant l'opinion de l'expert s'il existe des connaissances sur celle-ci, soit aléatoirement, et toujours en évitant de créer des circuits. Ainsi, nous avons un DAG qui servira d'état initial pour cette étape. Cette modification du graphe est représentée dans l'algorithme 17.

Comme nous l'avons vu dans le chapitre 2, pour définir une méthode à base de score, il nous faut un score, un espace de recherche avec un ensemble d'opérations et une

Algorithm 17 Conversion du PDAG en DAG arbitraire.

Entrée : un graphe partiellement orienté $G = (\mathbf{V}, \mathbf{E})$
Sortie : Un graphe totalement orienté G

1: **Pour tout** $X, Y \in \mathbf{V}^2$ s.t. $X - Y \in \mathbf{E}$ **Faire**
2: Orienter l'arête $X - Y$ sans créer de circuit et si possible en suivant les connaissances contenues dans $\mathbf{K}_e$
3: **Fin Pour**
4: **Retourner** Un graphe totalement orienté G

heuristique. Nous utilisons le score BDeu modifié de l'algorithme GSA, qui a déjà été présenté dans la sous-section 2.3.4 du chapitre 2 et qui permet de prendre en compte des connaissances. Ici, les connaissances ne concernent que des orientations causales de relations existantes. Cela nous a amené à modifier légèrement l'arbre de décision utilisé dans l'algorithme original. Cet arbre de décision modifié est présenté dans la figure 4.3, où $O_{XY} \in \{X \leftarrow Y, X \rightarrow Y\}$ est l'opinion de l'expert, et $g_{XY} \in \{X \leftarrow Y, X \rightarrow Y\}$ représente les orientations possibles.

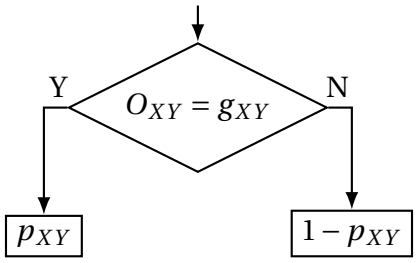


FIGURE 4.3 – Arbre de décision utilisé pour l'étape à base de score de l'algorithme.

L'espace de recherche est celui des DAG, et sera parcouru avec une recherche gloutonne. Cependant, le squelette a été trouvé à l'issue de l'étape sous contraintes, avec les connaissances de l'expert. Il n'est donc pas nécessaire de le modifier. Ainsi, les opérations de suppression et d'addition d'arc ne sont pas utilisées, et seule l'inversion d'arc est possible. Rappelons aussi que dans ce graphe, certaines orientations ont été obtenues durant l'étape sous contraintes, et sont considérées comme définitives, alors que l'autre partie a été choisie arbitrairement ou aléatoirement, uniquement pour avoir un état initial valide pour la méthode à base de score. Ainsi, ce sont ces dernières orientations, et seulement ces orientations, qui pourront être modifiées durant la recherche gloutonne. Les arêtes concernées correspondent exactement aux arêtes orientées dans l'algorithme 17.

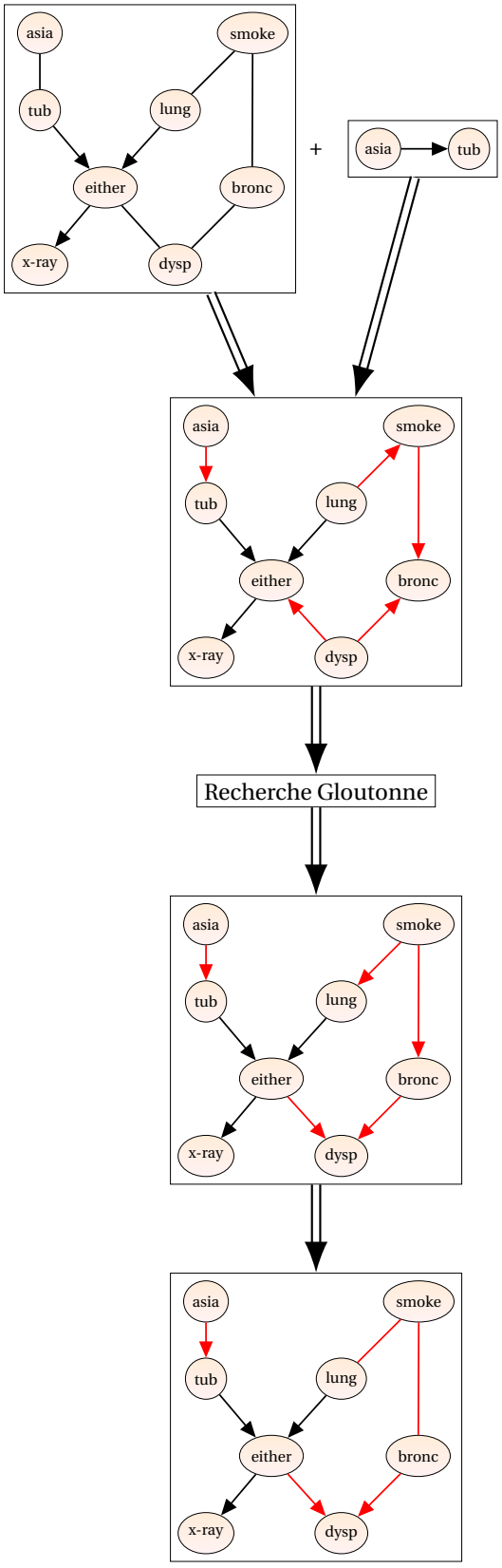
Une fois la convergence atteinte avec la méthode à base de score, nous obtenons un DAG. Cependant, on ne peut pas prouver que toutes les orientations de ce graphe sont causales. En effet, le score utilisé est "équivalent", cela signifie qu'il attribue un même score à tous les DAG d'une même classe d'équivalence de Markov. Tout comme pour les algorithmes sous contraintes, on peut supposer que toutes les orien-

tations propres à cette classe d'équivalences sont causales. Cependant, cela n'est pas possible pour les autres orientations. Ainsi, on va préserver les orientations qui correspondent au CPDAG dans le DAG obtenu. Parmi les orientations restantes, on peut supposer que celles corroborant une connaissance d'expert suffisamment certaine peuvent être considérées comment correctes, et peuvent également être préservées. Ici, une connaissance "suffisamment certaine" peut être définie en utilisant un seuil sur le paramètre p_{XY} de la connaissance. Pour les expérimentations présentées en fin de chapitre, nous avons choisi arbitrairement un seuil de 0,6. Compte tenu des orientations préservées après exécution de la recherche gloutonne, on obtient un PDAG$^+$, et toutes les orientations peuvent être considérées comme définitives. Il est alors possible d'appliquer les règles de Meek usuelles pour obtenir des orientations supplémentaires.

En entrée de cette étape, nous avions un graphe partiellement orienté qui ne représentait que partiellement les indépendances de la distribution. Grâce à l'utilisation d'une méthode à base de score, nous avons pu affiner ce graphe pour obtenir le modèle d'indépendance de la distribution et un graphe avec toutes les orientations causales associées au modèle d'indépendance, plus quelques orientations supplémentaires associées aux connaissances. Un exemple de ce processus est présenté avec la figure 4.4.

Exemple 4.3.2 *La figure 4.4 montre un exemple d'exécution de l'étape à base de score de notre algorithme avec le réseau bayésien Asia (*LAURITZEN *et* SPIEGELHALTER *1988). On peut voir qu'au début de l'étape représentée, une seule v-structure est présente. Une connaissance sur la relation causale de asia vers tub (pour tuberculose) est disponible avec une certitude à 0,8. Durant cette étape à base de score, nous allons d'abord compléter le graphe. Si il existe des connaissances suffisamment certaines, alors nous les suivons. Sinon, l'orientation est aléatoire. L'arête entre asia et tub est donc orientée suivant l'opinion de l'expert, les autres arcs rouges sont orientés aléatoirement, sans créer de circuit. Une fois un DAG obtenu, on peut appliquer la recherche gloutonne avec l'algorithme GSA qui permet de prendre en compte des connaissances incertaines dans le score. On obtient ainsi un nouveau DAG. De ce DAG, on ne garde que le CPDAG et les arcs corroborés par des connaissances. Ici, l'arc asia→tub est corroboré une connaissance suffisamment certaine, donc on peut préserver l'orientation. Dans cet exemple en particulier, appliquer les règles de Meek n'apporte aucune nouvelle orientation.*

Cependant, il reste encore des arêtes non orientées dans le graphe, que l'expert peut potentiellement nous aider à orienter. C'est pourquoi nous proposons pour cet algorithme une dernière étape interactive.



Le PDAG à la sortie de l'étape sous contraintes.

Une connaissance incertaine de l'expert avec $p_{asia \rightarrow tub} = 0,8$.

Le PDAG est complété pour obtenir un DAG. L'arête entre *asia* et *tub* est orientée selon la connaissance disponible.

La recherche gloutonne prenant en compte les connaissances a permis d'aboutir un DAG.

On restreint le graphe au CPDAG du DAG précédent et à l'orientation *asia*→*tub* corroborée par une connaissance. Une propagation en utilisant les règles de Meek peut être appliquée.

FIGURE 4.4 – Exemple d'exécution de l'étape à base de score de notre algorithme. En rouge les arcs et arêtes modifiés pendant cette étape.

4.3.4 Étape interactive

À l'étape précédente, nous avons donc obtenu successivement plusieurs structures. D'abord, à l'issue de la convergence de la recherche gloutonne, nous avions un DAG que nous noterons G^*. Ensuite, nous en avons extrait un CPDAG qui représente une classe d'équivalence $\mathscr{C}$. Enfin, ce CPDAG a été transformé, grâce aux connaissances de l'expert et à une application des règles de Meek, en un PDAG$^+$.

L'objectif de cette nouvelle étape est de compléter ce dernier graphe, c'est-à-dire trouver les orientations causales des arêtes restantes, tout en prenant soin de ne pas créer de nouvelles v-structures. Cette volonté de ne pas créer de nouvelles v-structures se traduit par la préservation de la classe d'équivalence $\mathscr{C}$. En effet, un ensemble de v-structures a été obtenu grâce aux deux premières étapes de l'algorithme, et modifier cet ensemble contredirait le modèle causal partiel que nous avons déjà appris. Ces v-structures sont associées à une classe d'équivalence, et ne pas modifier l'ensemble des v-structures du graphe est donc équivalent à ne pas changer de classe d'équivalence.

Il s'agit ici d'un problème d'inférence causale. Malheureusement, nos hypothèses ne sont pas suffisantes pour utiliser un des algorithmes étudiés à la sous-section 2.3.5 du chapitre 2. Ainsi, seules les connaissances de l'expert peuvent nous permettre d'orienter les dernières arêtes. Mais celles utilisées jusqu'ici ne concernent pas forcément les arêtes restantes. C'est pour cette raison que nous proposons d'utiliser une étape interactive, dans laquelle l'expert pourrait directement répondre à nos questions. Les réponses à ces questions seraient de nouvelles connaissances se rapportant directement aux arêtes à orienter.

Il est également nécessaire de prendre en compte qu'une telle intervention d'un expert peut être coûteuse. C'est pourquoi nous souhaitons également que l'élicitation des connaissances durant cette étape soit la plus efficace possible. En d'autres termes, nous voulons poser le moins de questions possible.

Pour cette étape, nous pouvons donc identifier trois questions :

— comment compléter le graphe tout en restant dans la classe d'équivalence de Markov $\mathscr{C}$?
— comment obtenir les orientations causales ?
— comment minimiser le coût de l'élicitation des connaissances ?

Dans cette sous-section, nous répondrons à chacune de ces questions. Commençons avec la première question.

Comment compléter le graphe en restant dans la classe d'équivalence de Markov $\mathscr{C}$?

Pour répondre à cette question, nous occulterons les problèmes des deux autres questions. Nous cherchons donc à compléter le graphe pour obtenir un DAG appartenant à $\mathscr{C}$. Pour ce problème, nous proposons de répéter l'utilisation de la procédure 18, dont le théorème 4.3.1 montre qu'il permet d'arriver au résultat désiré.

Algorithm 18 Procédure de l'étape interactive

1: $e \leftarrow$ choisir une arête dans **E**
2: $o \leftarrow$ choisir une orientation pour e
3: Appliquer l'orientation o à e
4: Appliquer les règles de Meek pour propager dans le graphe l'orientation nouvellement obtenue

Théorème 4.3.1 *Soient $G_0, \ldots, G_k$ une suite de graphes partiellement orientés tels que G_0 soit un CPDAG et, pour tout $i < k$, G_i est obtenu à partir G_{i-1} en orientant n'importe quelle arête de G_{i-1} dans n'importe quelle direction, puis en appliquant les règles de Meek. Alors $G_0, \ldots, G_k$ sont tous des PDAG$^+$ et peuvent être complétés en un DAG appartenant à la même classe d'équivalence de Markov que G_0.*

Démonstration
Soit G^* le DAG obtenu après convergence du score durant l'étape à base de score. Notons G_0 le CPDAG issu de G^* obtenu en remplaçant, par une arête $X - Y$, les arcs $X \rightarrow Y$ qui n'appartiennent à aucune v-structure ni ne sont contraints par aucune v-structure. Ainsi, G_0 est un CPDAG qui représente une classe d'équivalence $\mathscr{C}$ non vide (elle contient notamment G^*). Choisissons une arête $X - Y$ de G_0 et orientons la arbitrairement $X \rightarrow Y$. Cette arête était non orienté dans le CPDAG, donc son orientation n'est pas contrainte par une v-structure. Donc il existe dans $\mathscr{C}$ au moins un DAG, noté G_{DAG_1} qui possède toutes les orientations de G_0 plus $X \rightarrow Y$. Comme $G_{DAG_1} \in \mathscr{C}$, G_{DAG_1} doit aussi avoir les mêmes v-structures que G_0 et donc toutes les orientations contraintes par $X \rightarrow Y$ dans G_0 se retrouvent aussi G_{DAG_1}. Ainsi, le graphe G_1 obtenu en orientant $X - Y$ de X vers Y est un PDAG$^+$ tel que ses arcs se retrouvent tous dans G_{DAG_1}.

À partir de G_1, il est possible d'appliquer le même raisonnement pour obtenir le graphe G_2 qui possèdera les mêmes propriétés, avec un DAG G_{DAG_2}. Par récurrence, cette propriété reste vraie pour G_i, $i \leq k$.

Le nombre d'arêtes étant fini, en suivant cette procédure, on arrivera nécessairement à un DAG. ■

Ainsi, en utilisant la procédure 18, à partir d'un CPDAG, on est sûr de ne pas changer les indépendances conditionnelles. Après un nombre d'itérations suffisant, on est donc assuré d'obtenir un DAG appartenant à la classe d'équivalence de Markov représentée par le CPDAG initial.

Dans notre cas, nous commençons avec un PDAG$^+$. Mais celui-ci est issu d'un CPDAG, lui même issu d'un DAG, d'après l'étape à base de score. Donc le raisonnement fonctionne de la même manière. De plus, ce raisonnement est indépendant de la manière de sélectionner l'arête qui doit être orientée et son orientation. Donc qu'importe la méthode utilisée, cela ne changera pas les propriétés du résultat. Nous pouvons maintenant passer à la seconde question.

Comment obtenir les orientations causales ?

Comme dit précédemment, nous souhaitons directement poser les questions à l'expert pour obtenir les orientations causales nécessaires. Nous pouvons cependant donner un détail supplémentaire. Les connaissances qui vont être élicitées avec les questions n'ont pas été données plus tôt par l'expert. Cela signifie que ces connaissances peuvent être caractérisées par une forte incertitude. Lors de cette étape, nous supposons que l'expert est toujours capable d'évaluer l'incertitude de ses connaissances, et nous utilisons un seuil pour filtrer les connaissances trop incertaines. Dans les expérimentations qui suivent, ce seuil est fixé à 0,6.

Enfin, comme dit précédemment, cette interaction avec l'expert peut être coûteuse. C'est de ce fait que découle la dernière question, sur laquelle nous passerons un peu plus de temps.

Comment minimiser le coût de l'élicitation des connaissances ?

Nous estimons ici le coût de l'intervention de l'expert en temps. Nous pouvons alors identifier deux facteurs de coût : le temps pour répondre à une question et le nombre de questions. Cependant, il est difficile d'estimer le temps que pourrait mettre l'expert pour répondre à une question. Pour cette raison, nous allons tenter d'agir sur le nombre de questions qui seront posées à l'expert.

Pour minimiser le nombre de questions, une solution consiste à poser les questions dont les réponses seront les plus informatives. Ici, les informations dont nous avons besoin sont les directions des relations causales. Ainsi, pour estimer l'intérêt d'une question, nous pouvons essayer d'évaluer le nombre d'orientations que la réponse à cette question pourrait apporter. En effet, comme l'indique la procédure 18, chaque réponse est suivie de l'application des règles de Meek pour propager l'orientation obtenue. L'objectif est donc de compter le nombre potentiel d'arêtes qui pourront être orientées à la suite d'une réponse. Ainsi, en choisissant l'arête permettant le plus d'orientations, on peut diminuer le temps d'intervention de l'expert.

Voyons maintenant comment estimer le nombre potentiel d'arêtes pouvant être orientées après une réponse. Sachant l'arête $X - Y$ et l'orientation $X \rightarrow Y$, le nombre d'orientations obtenues après propagation, noté $n_{X \rightarrow Y}$, peut être déterminé simplement en appliquant les règles de Meek. Après avoir fait de même pour l'orientation $X \leftarrow Y$, nous avons les deux valeurs $n_{X \rightarrow Y}$ et $n_{X \leftarrow Y}$. En associant à chaque arête $X - Y$ la paire $(n_{X \rightarrow Y}, n_{X \leftarrow Y})$, nous avons donc à résoudre un problème d'optimisation bicritère. Pour cela, nous pouvons adopter une stratégie Maximin, qui consiste comme son nom l'indique à maximiser le minimum des deux critères. De manière plus formelle, on cherche :

$$Max(\{Min(n_{X \rightarrow Y}, n_{X \leftarrow Y}) : X - Y \in E\})$$

Enfin, notons une dernière chose. Pendant l'étape à base de score, lors du passage du DAG au CPDAG, on préserve les orientations corroborées par des connaissances même si elles concernent des arcs qui n'appartiennent ni ne sont contraints par des v-

Algorithm 19 L'étape interactive

Entrée : un graphe partiellement orienté $G = (\mathbf{V}, \mathbf{E})$, un seuil p
Sortie : Un graphe partiellement ou totalement orienté G

1: **Répéter**
2: Soit $X - Y$ une arête maximisant $Max(\{Min(n_{X\rightarrow Y}, n_{X\leftarrow Y}) : X - Y \in E\})$
3: Demander l'opinion de l'expert à propos de l'orientation $X - Y$
4: **Si** $p_{XY} \geq p$, i.e., la certitude de l'expert est suffisamment forte **Alors**
5: Orienter $X - Y$ dans **E** selon l'opinion de l'expert
6: Appliquer les règles de Meek pour propager dans le graphe l'orientation nouvellement obtenue
7: **Fin Si**
8: **Jusqu'à ce que** il ne reste plus d'arête non orientée ou l'expert ne peut plus répondre aux questions
9: **Retourner** le graphe G

structures. Par conséquent, certains de ces arcs, que la recherche gloutonne a orienté dans une direction contraire aux connaissances, vont perdre leurs orientations. Or si ce sont bien des arcs qui n'appartiennent ni ne sont contraints par des v-structures, le choix de l'algorithme n'a rien de causal, et "l'opposition" portée contre les connaissances n'a donc pas de valeur. En d'autres termes, ces connaissances n'ont pas été annulées par les données, et peuvent donc encore être utilisées. Ainsi, une solution pour encore réduire l'intervention de l'expert est d'abord d'utiliser ces connaissances. Elle pourrait en effet orienter d'autres arêtes, ce qui fait autant de questions qui n'auront pas besoin d'être posées.

Nous avons maintenant répondu aux trois questions. Nous avons donc une solution pour compléter de manière causale un graphe partiellement orienté avec l'aide de l'expert de manière efficace. Notons pour terminer que, bien que l'objectif soit d'obtenir un DAG, nous ne pouvons pas garantir que l'expert sera capable de répondre à toutes les questions avec suffisamment d'assurance. Lorsqu'une question reste sans réponse, alors on passe à l'arête suivante. Mais lorsqu'il n'y a plus d'arête sur laquelle poser une question, alors il n'est plus possible d'orienter le graphe avec notre méthode. Cela démontre encore la difficulté du problème de découverte causale.

Nous pouvons formaliser les éléments ci-dessus sous la forme d'un algorithme décrit par le pseudo-code 19.

Dans la prochaine section, nous verrons les expérimentations qui nous ont permis de démontrer l'efficacité de l'algorithme.

4.4 Expérimentations

4.4.1 Paramètres expérimentaux

Dans cette section, nous allons décrire le comportement de notre algorithme, que nous appellerons **CAIRN**, pour *Causal Algorithm with expeRt kNowledge*, ou Algorithme Causal avec des Connaissances d'Expert avec en français. Dans les résultats, nous montrerons parfois plusieurs valeurs pour CAIRN : CAIRN.1, CAIRN.2 et CAIRN.3. Ces valeurs correspondent respectivement aux graphes obtenus à la fin de la première étape (étape sous contraintes), de la deuxième étape (étape à base de score) et de la troisième étape (étape interactive). Nous démontrerons son efficacité en le comparant à d'autres algorithmes. Ces algorithmes ont déjà été présentés dans ce manuscrit. Il s'agit des algorithmes GSA et PCSDC, la version stable de PCDC. Nous étudierons l'impact des connaissances d'experts sur le processus d'apprentissage, qu'elles soient certaines ou incertaines, et montrerons que ces connaissances peuvent avoir un impact positif pour l'apprentissage des relations causales.

Nous avons exécuté les algorithmes de ces expérimentations sur des bases de données générées à partir de 6 réseaux bayésiens différents, tous ayant déjà été utilisés dans le chapitre 3. La taille de ces bases de données varient selon les expériences, allant de 500 à 10 000. Nous avons généré automatiquement les connaissances utilisées, à partir des informations structurelles contenues dans les graphes des réseaux bayésiens utilisés. Les connaissances que nous utilisons ne concernent que des relations causales existantes, mais nous ajoutons volontairement des erreurs dans les connaissances pour tester la robustesse des algorithmes. Ainsi, pour générer les connaissances, nous avons sélectionné aléatoirement une part variable d'arcs qui dépend de la taille du graphe. Pour certains de ces arcs, l'orientation inverse est utilisée, ce qui permet d'ajouter les erreurs. La part d'erreurs dans les connaissances est variable.

Le risque de première espèce α utilisé pour les tests d'indépendances est fixé à 0,05, et le seuil p utilisé à plusieurs reprises dans l'algorithme CAIRN est fixé à 0,6.

Les trois algorithmes sont comparés sur la base du graphe partiellement ou totalement orienté qu'ils parviennent à retourner. La principale mesure utilisée sera la mesure SHD, pour *Structural Hamming Distance*, soit Distance Structurelle de Hamming en français. Cette grandeur est une distance permettant de comparer deux structures graphiques en fonction du nombre minimal d'opérations élémentaires nécessaires pour passer de l'une à l'autre. Cette distance est couramment utilisée avec des DAG en prenant en compte les opérations suivantes : inversion d'arc, ajout d'arc et suppression d'arc. Nous comparons le résultat des algorithmes au DAG original.

Dans la perception du problème que l'on a dans ce chapitre, orienter un arc dans la mauvaise direction est plus grave que de ne pas orienter un arc. En effet, nous souhaitons avant tout éviter d'apprendre des informations incorrectes, car une erreur peut avoir des conséquences plus graves sur la prise de décision qu'une absence d'information. Ainsi, nous souhaitons que la mesure SHD pénalise moins les arcs non orientés que les arcs orientés dans la mauvaise direction. De plus, les opérations utilisées doivent être compatibles avec la comparaison entre un PDAG et un DAG.

Pour cette raison nous avons choisi les opérations suivantes :
- enlever une arête
- ajouter une arête
- enlever une orientation
- ajouter une orientation

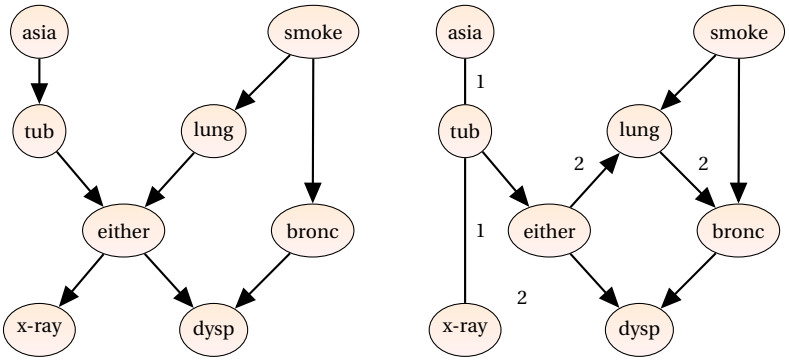


FIGURE 4.5 – (a) Le DAG du réseau bayésien Asia. (b) Un PDAG et les poids de chaque arc et arête entraînant une modification de SHD.

Exemple 4.4.1 *Sur la figure 4.5, nous pouvons voir un exemple de la logique derrière le calcul de la distance structurelle de Hamming. Lorsque qu'il manque une orientation, il faut une seule opération. Lorsqu'il y a une arête en trop, il faut également une seule opération. Mais lorsqu'une orientation est mauvaise ou qu'il y a un arc en trop, alors il faut deux opérations pour corriger les erreurs, ce qui augmente de deux la distance structurelle de Hamming. Ainsi la structure 4.5b a un SHD égal à 8.*

Ce nouvel ensemble d'opérations ne modifie pas les propriétés de cette mesure, elle reste une distance au sens mathématique du terme, c'est-à-dire qu'elle respecte les propriétés de symétrie, de séparation et l'inégalité triangulaire.

Notons que cette distance peut être formalisée autrement. On peut également représenter chaque relation entre deux nœuds X et Y de la manière suivante : $X \circ\!\!-\!\!-\!\!\circ Y$, où $--$ représente "rien" ou une ligne, et chaque $\circ$ représente "rien" ou $>$ (où "rien" est une absence de symbole). Par exemple, pour deux nœuds non adjacents, $--$ ainsi que les deux $\circ$ représentent "rien", alors que pour $X \leftarrow Y$, le $\circ$ de gauche est $>$, le $\circ$ de droite est "rien" et $--$ est une ligne. On suppose ici un ordre fixe entre les paires de nœuds et entre les nœuds de chaque paire, ce qui permet de ne compter chaque paire qu'une seule fois et ainsi ne pas compter (X, Y) et (Y, X) deux fois. De cette manière, le SHD entre deux graphes est la somme des différences entre les symboles $\circ$ et le symbole $--$ dans chacun des graphes. Par exemple, la distance entre le graphe $X \leftarrow Y$ et $X \rightarrow Y$ est $1+1+0=2$, puisque les deux symboles $\circ$ sont différents mais le symbole $--$ est identique.

Mesure	Graphe appris	Graphe original
Vrai positif	$X \rightarrow Y$	$X \rightarrow Y$
Faux positif	$X \rightarrow Y$	$X \leftarrow Y$ ou $X \quad Y$

TABLEAU 4.2 – Mesures utilisées pour déterminer la précision d'un graphe partiel.

D'autres mesures ont également été utilisées ci-dessous. D'abord, les mesures SHD varient beaucoup en fonction du réseau bayésien. Par exemple, une structure avec peu d'arcs impliquera mécaniquement un SHD plus petit. Pour cette raison, lorsque les réseaux bayésiens varient sur un même tableau, nous utiliserons le SHD relatif. En d'autres termes, toutes les valeurs du tableau sont divisées par celle obtenue par CAIRN.3. En outre, nous avons utilisé le F-score du squelette, calculé de la même façon qu'au chapitre 3. Nous avons également utilisé la précision du graphe partiel, déterminée avec les valeurs "vrai positif" et "faux positif" trouvé dans le tableau 4.2. Enfin, nous avons utilisé la proportion d'arcs orientés dans un graphe, qui correspond simplement au nombre d'arcs divisé par le nombre d'arcs et d'arêtes dans le graphe.

Durant ces expérimentations, nous avons fait varier les paramètres suivants :

— la taille de la base de données
— la taille de la base de connaissances
— le pourcentage de connaissances correctes (la véracité)
— le paramètre d'incertitude de l'expert p_{XY}

Chaque valeur ci-dessous correspond à une moyenne obtenue sur 100 exécutions des algorithmes avec des bases de données et des connaissances différentes. Toutes les expériences ont été menées sur un ordinateur muni d'un processeur Intel Core i7 de 1.9GHz avec 16Go de mémoire RAM tournant sur Windows 10. Les algorithmes ont été implémentés en python et en utilisant les bibliothèques NetworkX (HAGBERG et CONWAY 2020) et pyAgrum (DUCAMP, GONZALES et WUILLEMIN 2020).

4.4.2 Résultats

Commençons par regarder les tableaux 4.3, 4.4 et 4.5. Le tableau 4.3 montre le SHD en fonction de l'incertitude et de la taille de la base de données. Le tableau 4.4 montre le SHD en fonction de l'incertitude et du réseau bayésien. Le tableau 4.5 montre pour sa part le F-score du squelette de chaque algorithme en fonction de la certitude et de la base de données (seul CAIRN.1 est montré car le squelette ne change plus après la première étape). Sur chacun de ces tableaux, nous pouvons observer que les résultats varient peu entre 0.6 et 0.9. Cela signifie que la variation de la certitude n'a pas d'impact important sur l'exécution de l'algorithme. En effet, mis à part l'étape à base de score, l'utilisation de la certitude s'appuie principalement sur des seuils. Par exemple, dès le début, dans l'algorithme 14, on différencie les connaissances certaines et incertaines en fonction d'un seuil logiquement placé à 1. Ensuite, dans l'algorithme 15, on décide d'appliquer ou non les connaissances, en utilisant un seuil pour ne pas prendre en

Taille	500				1000			
p_{XY}	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0.5	59.82±13.09	39.21±6.19	39.21±6.19	49.26±5.49	51.78±15.64	32.48±4.94	32.48±4.94	36.20±5.39
0.6	38.92±10.75	40.32±6.33	39.93±6.67	48.79±5.38	31.28±10.85	30.05±5.35	29.35±5.66	39.54±4.84
0.7	39.48±10.55	40.35±6.41	40.04±6.57	48.79±5.38	31.36±10.79	30.09±5.32	29.41±5.31	39.54±4.84
0.8	39.54±10.48	40.15±6.45	39.70±6.64	48.79±5.38	31.54±10.94	30.18±5.36	29.46±5.66	39.54±4.84
0.9	39.54±10.38	40.07±6.29	39.76±6.54	48.79±5.38	32.02±10.99	30.16±5.37	29.61±5.50	39.54±4.84
1.0	54.16±9.12	43.56±5.62	43.16±5.70	48.79±5.38	49.32±7.97	35.28±4.61	34.78±5.08	39.54±4.85
Taille	5000				10000			
p_{XY}	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0.5	51.92±16.37	22.21±4.65	22.21±4.65	24.21±4.84	51.34±18.01	17.99±4.41	17.99±4.41	18.95±4.49
0.6	28.14±11.40	22.56±5.31	21.82±5.50	36.52±5.69	26.14±12.73	18.81±5.83	18.13±5.88	36.08±5.62
0.7	28.24±11.43	22.60±5.37	21.80±5.36	36.52±5.69	25.92±12.86	18.79±5.82	17.91±5.79	36.08±5.62
0.8	28.32±11.31	22.54±5.35	21.58±5.62	36.52±5.69	26.12±12.96	18.79±5.82	17.94±5.82	36.08±5.62
0.9	28.54±11.22	22.56±5.37	21.88±5.58	36.52±5.69	26.38±12.93	18.79±5.82	17.86±5.85	36.08±5.62
1.0	49.58±8.97	31.57±5.38	30.78±5.33	36.52±5.69	50.02±9.45	30.71±5.04	30.08±5.36	36.08±5.62

TABLEAU 4.3 – SHD de GSA, CAIRN.2, CAIRN.3 et PCSDC pour le réseau bayésien Alarm en fonction de p_{XY} et de la taille de la base de données.

compte les connaissances incertaines. Cette dernière méthode est également utilisée pour l'étape interactive. Ainsi, dans les tableaux, ailleurs qu'au niveau des seuils, il existe peu de différence dans le comportement de l'algorithme. On voit d'ailleurs sur le tableau 4.5, pour CAIRN.1, que la variation des résultats en fonction de la certitude lorsque celle-ci est comprise entre 0.5 et 0.9 est totalement inexistante. Pour ce même tableau, on voit une variation entre 0.9 et 1.0, ce qui souligne la distinction faite entre les connaissances certaines et incertaines. Ainsi, on retrouve bien l'utilisation exclusive de seuil durant la première étape.

Si on regarde maintenant les deux autres algorithmes à propos de ce point, on voit que ceux-ci ne présentent pas non plus de variations importantes. Pour PCSDC, ces résultats sont attendues. En effet, l'algorithme ne prend pas du tout en compte l'incertitude. Cependant, GSA prend en compte l'incertitude, sans qu'il soit question de seuil, durant toute son exécution. Malgré ça, les variations restent minimes en dehors des extrema, signe que l'incertitude n'a pas énormément d'influence. L'étape à base de score de l'algorithme CAIRN utilisant la même approche que GSA, il est donc normal de retrouver ce même constat sur CAIRN.

Continuons avec l'étude du tableau 4.5. Rappelons que ce tableau ne s'intéresse qu'au squelette des graphes appris. Dans notre algorithme, le squelette est déterminé à la première étape de l'algorithme, CAIRN.1. Il s'agit d'une étape sous contraintes. Or nous voyons sur le tableau 4.5 deux choses. D'une part, plus la base de données est grande, plus le F-score associé au squelette de CAIRN est élevé, passant de 0.85 pour 500 échantillons, à 0.95 pour 10 000. Cela signifie que la taille de la base de données influence l'apprentissage. Cela n'est pas surprenant, ce phénomène est bien

RB	Alarm				Asia			
p_{XY}	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0.5	2.338±0.737	1.000±0.209	1.0±0.209	1.090±0.218	1.854±1.23	1.000±0.254	1.0±0.254	1.401±0.442
0.6	1.290±0.552	1.034±0.243	1.0±0.252	1.674±0.261	0.997±1.03	1.090±0.493	1.0±0.508	1.816±0.639
0.7	1.295±0.524	1.037±0.246	1.0±0.246	1.675±0.261	0.977±1.02	1.094±0.495	1.0±0.559	1.822±0.641
0.8	1.312±0.524	1.044±0.248	1.0±0.260	1.692±0.264	1.032±1.04	1.103±0.499	1.0±0.569	1.838±0.647
0.9	1.304±0.513	1.031±0.246	1.0±0.255	1.669±0.260	1.006±1.01	1.069±0.483	1.0±0.547	1.780±0.626
1.0	1.611±0.292	1.026±0.175	1.0±0.173	1.186±0.185	1.211±0.654	1.056±0.360	1.0±0.363	1.205±0.424
RB	Insurance				Sachs			
p_{XY}	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0.5	1.754±0.503	1.000±0.144	1.0±0.144	1.481±0.166	0.854±0.250	1.000±0.041	1.0±0.041	1.117±0.158
0.6	1.167±0.386	1.018±0.178	1.0±0.185	1.675±0.239	0.866±0.416	1.162±0.458	1.0±0.539	1.162±0.371
0.7	1.167±0.356	1.019±0.178	1.0±0.187	1.676±0.239	0.865±0.416	1.161±0.457	1.0±0.549	1.161±0.371
0.8	1.165±0.393	1.024±0.179	1.0±0.189	1.683±0.240	0.893±0.429	1.198±0.472	1.0±0.542	1.198±0.382
0.9	1.160±0.392	1.019±0.179	1.0±0.183	1.680±0.240	0.881±0.423	1.181±0.465	1.0±0.526	1.181±0.377
1.0	1.202±0.217	1.008±0.162	1.0±0.166	1.115±0.166	1.060±0.414	1.100±0.362	1.0±0.382	1.209±0.386
RB	Child				Survey			
p_{XY}	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0.5	2.012±0.838	1.000±0.197	1.0±0.197	1.840±0.368	2.856±1.11	1.000±0.878	1.0±0.878	1.586±0.458
0.6	2.340±1.22	1.145±0.871	1.0±0.866	5.358±1.05	1.732±0.846	1.014±0.776	1.0±0.772	1.433±0.535
0.7	2.389±1.25	1.168±0.889	1.0±0.872	5.469±1.07	1.743±0.835	1.000±0.781	1.0±0.781	1.413±0.528
0.8	2.227±1.17	1.089±0.829	1.0±0.829	5.099±0.996	1.720±0.816	1.014±0.767	1.0±0.773	1.423±0.531
0.9	2.419±1.26	1.181±0.899	1.0±0.881	5.526±1.08	1.689±0.795	1.035±0.720	1.0±0.716	1.408±0.526
1.0	1.099±0.226	1.033±0.228	1.0±0.221	1.310±0.256	1.358±0.431	1.024±0.443	1.0±0.451	1.245±0.465

TABLEAU 4.4 – SHD relatif de GSA, CAIRN.2, CAIRN.3 et PCSDC en fonction de p_{XY} et du réseau bayésien.

Taille	500			1000		
p_{XY}	GSA	CAIRN.1	PCSDC	GSA	CAIRN.1	PCSDC
0.5	0.856±0.034	0.851±0.025	0.851±0.025	0.880±0.038	0.889±0.022	0.889±0.022
0.6	0.887±0.031	0.851±0.025	0.861±0.030	0.918±0.029	0.889±0.022	0.902±0.028
0.7	0.887±0.031	0.851±0.025	0.861±0.030	0.917±0.029	0.889±0.022	0.902±0.028
0.8	0.886±0.030	0.851±0.025	0.861±0.030	0.917±0.030	0.889±0.022	0.902±0.028
0.9	0.886±0.031	0.851±0.025	0.861±0.030	0.915±0.030	0.889±0.022	0.902±0.028
1.0	0.844±0.031	0.861±0.030	0.861±0.030	0.863±0.029	0.902±0.028	0.902±0.028
Taille	5000			10000		
p_{XY}	GSA	CAIRN.1	PCSDC	GSA	CAIRN.1	PCSDC
0.5	0.887±0.034	0.931±0.019	0.931±0.019	0.890±0.041	0.950±0.016	0.950±0.016
0.6	0.930±0.030	0.931±0.019	0.934±0.028	0.938±0.032	0.950±0.016	0.939±0.024
0.7	0.930±0.030	0.931±0.019	0.934±0.028	0.938±0.032	0.950±0.016	0.939±0.024
0.8	0.929±0.029	0.931±0.019	0.934±0.028	0.938±0.032	0.950±0.016	0.939±0.024
0.9	0.929±0.029	0.931±0.019	0.934±0.028	0.937±0.032	0.950±0.016	0.939±0.024
1.0	0.866±0.032	0.934±0.028	0.934±0.028	0.864±0.031	0.939±0.024	0.939±0.024

TABLEAU 4.5 – F-score du squelette de GSA, CAIRN et PCSDC pour le réseau bayésien Alarm en fonction de p_{XY} et de la taille de la base de données.

connu : plus il y a de données, plus les tests d'indépendances seront précis et donc meilleur sera l'apprentissage. L'autre point à relever concerne la différence de F-score lorsque la certitude passe de 0.9 à 1. Pour des bases de données de 10 000 échantillons, on observe une légère baisse du F-score, contrairement aux résultats pour d'autres tailles de base de données. Cela signifie de l'apprentissage est moins bon lorsque les connaissances sont certaines, pour cette taille de base de données. Lorsque la base de données est plus petite, on observe un résultat au moins aussi bon sinon meilleur lorsque les données sont certaines. Ici, les connaissances sont correctes à 75%. Or avec une base de données suffisamment grande, le modèle appris sans les connaissances est déjà bon, donc le bénéfice des connaissances correctes va diminuer. Cependant, le déficit lié aux connaissances incorrectes reste le même. Ainsi, la certitude devient désavantageuse lorsque les données sont de bonne qualité et que les connaissances sont seulement partiellement correctes.

Une conséquence directe des conclusions tirées précédemment est l'influence de la taille de la base de données sur les résultats de CAIRN.2 et CAIRN.3, notamment par rapport à ceux de GSA et PCSDC. Sur le tableau 4.3, on voit que pour de petites bases de données de taille 500, le graphe issu de GSA présente un SHD comparable à ceux issus de CAIRN.2 et CAIRN.3, et donc que GSA apprend des modèles équivalents à ceux de CAIRN. Cependant, les choses changent lorsque la base de données devient plus grande. La qualité des modèles retournés par CAIRN augmente bien plus vite que celle des modèles de GSA en fonction de la taille de la base de données : une baisse du SHD de 53% pour CAIRN et 33% pour GSA lorsque la taille de la base de données augmente de 500 à 10000 échantillons (pour $p_{XY} = 0,9$). Puisque l'étape 3 de

Taille	500				1000			
Véracité (%)	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0	97.96±10.6	57.29±7.23	59.00±6.98	92.67±5.20	93.84±8.79	45.57±7.07	46.56±6.88	86.76±5.35
25	79.68±12.8	53.51±7.78	54.38±7.87	79.25±5.00	75.94±11.2	43.21±6.86	43.67±6.93	72.09±4.87
50	57.92±12.6	45.24±7.30	45.74±7.19	64.72±5.04	52.70±12.9	35.96±6.19	35.92±6.47	56.97±5.20
75	39.54±10.5	40.15±6.45	39.70±6.64	48.79±5.38	31.54±10.9	30.18±5.36	29.46±5.66	39.54±4.84
100	24.46±6.49	32.83±5.30	31.64±5.33	31.86±4.79	15.72±4.90	22.96±4.41	21.65±4.30	21.67±4.12
Taille	5000				10000			
Véracité (%)	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0	95.58±8.75	38.21±8.42	39.58±8.24	86.38±5.65	99.34±11.5	35.37±8.15	36.53±8.10	88.74±5.92
25	76.72±12.2	33.14±7.25	33.87±7.38	71.54±6.65	78.92±13.0	30.43±7.73	31.10±7.94	73.05±6.01
50	51.40±13.7	27.81±6.28	27.85±6.23	56.28±6.39	49.46±15.6	23.92±6.63	23.76±6.67	56.52±7.32
75	28.32±11.3	22.54±5.35	21.58±5.62	36.52±5.69	26.12±13.0	18.79±5.82	17.94±5.82	36.08±5.62
100	10.26±3.71	16.41±4.59	14.91±4.52	16.95±4.24	7.18±3.57	13.19±4.06	11.68±3.83	15.29±4.14

TABLEAU 4.6 – SHD de GSA, CAIRN.2, CAIRN.3 et PCSDC pour le réseau bayésien Alarm en fonction de la véracité et de la taille de la base de données.

RB	Alarm				Asia			
Véracité (%)	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0	2.415±0.221	0.965±0.213	1.0±0.208	2.182±0.143	2.100±0.420	0.972±0.401	1.0±0.411	1.788±0.283
25	2.265±0.361	0.978±0.214	1.0±0.218	2.112±0.196	1.776±0.815	0.968±0.375	1.0±0.345	1.692±0.474
50	1.846±0.493	0.999±0.226	1.0±0.224	2.021±0.229	1.351±0.837	0.989±0.418	1.0±0.453	1.595±0.481
75	1.312±0.524	1.044±0.248	1.0±0.260	1.692±0.264	1.032±1.04	1.103±0.499	1.0±0.569	1.838±0.647
100	0.688±0.249	1.101±0.308	1.0±0.303	1.137±0.284	0.502±0.939	1.433±0.615	1.0±0.675	1.712±0.779
RB	Insurance				Sachs			
Véracité (%)	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0	1.872±0.288	0.995±0.180	1.0±0.179	2.457±0.176	1.130±0.128	0.926±0.091	1.0±0.071	1.987±0.273
25	1.714±0.319	0.999±0.204	1.0±0.205	2.357±0.256	1.075±0.210	0.957±0.159	1.0±0.159	1.675±0.274
50	1.536±0.417	1.005±0.221	1.0±0.222	2.157±0.262	1.058±0.351	1.013±0.322	1.0±0.177	1.626±0.380
75	1.165±0.393	1.024±0.179	1.0±0.189	1.683±0.240	0.893±0.429	1.198±0.472	1.0±0.542	1.198±0.382
100	0.864±0.278	1.064±0.167	1.0±0.192	1.016±0.203	0.962±0.760	2.183±0.931	1.0±1.15)	1.458±0.694
RB	Child				Survey			
Véracité (%)	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0	2.633±0.569	1.011±0.279	1.0±0.239	5.056±0.727	1.334±0.288	0.949±0.180	1.0±0.179	1.273±0.176
25	2.312±0.587	1.005±0.345	1.0±0.341	4.578±0.712	1.557±0.319	0.980±0.204	1.0±0.205	1.365±0.256
50	2.203±0.720	1.051±0.538	1.0±0.537	4.250±0.767	1.963±0.417	1.013±0.221	1.0±0.222	1.697±0.262
75	2.227±1.17	1.089±0.829	1.0±0.829	5.099±0.996	1.720±0.393	1.014±0.179	1.0±0.189	1.423±0.240
100	3.017±2.84	2.155±2.01	1.0±1.76	6.776±2.72	2.241±0.278	1.019±0.167	1.0±0.192	1.120±0.203

TABLEAU 4.7 – SHD relatif de GSA, CAIRN.2, CAIRN.3 et PCSDC en fonction de la véracité et du réseau bayésien.

Taille	500				1000			
$\|K\|$	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0	69.54±13.1	67.14±5.35	61.92±5.95	75.20±4.82	59.58±13.5	55.26±5.22	50.47±5.23	57.77±4.97
25	50.64±12.7	53.95±6.28	52.23±6.47	60.69±5.59	43.58±11.9	43.39±5.48	41.01±6.00	48.79±5.83
50	42.20±10.0	42.99±6.41	42.23±6.70	51.40±5.71	35.00±10.1	32.74±4.99	31.60±5.08	41.09±5.18
75	33.50±9.34	33.03±6.61	32.84±6.67	42.47±5.16	26.44±9.02	23.33±5.82	22.94±5.97	35.05±4.99
100	29.64±7.53	24.55±6.10	24.58±6.12	37.71±4.33	23.16±8.83	14.23±5.13	14.23±5.13	31.30±3.74
Taille	5000				10000			
$\|K\|$	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0	59.68±13.0	36.80±3.34	33.16±3.42	38.35±4.21	62.74±10.7	30.60±2.27	27.58±3.10	28.67±2.93
25	39.04±14.8	30.78±4.21	28.71±4.70	35.96±5.86	37.34±15.1	24.80±4.53	22.89±4.75	30.31±6.29
50	30.36±11.7	23.69±5.29	22.57±5.36	36.43±5.53	28.98±12.9	19.95±5.30	18.93±5.51	34.28±5.51
75	22.42±10.1	16.41±4.62	16.04±4.71	35.93±5.90	19.78±10.2	14.56±4.82	14.10±4.86	36.17±6.36
100	17.50±8.74	10.64±4.98	10.64±4.98	35.86±4.04	15.42±9.80	10.51±5.18	10.51±5.18	37.94±4.32

TABLEAU 4.8 – SHD de GSA, CAIRN.2, CAIRN.3 et PCSDC pour le réseau bayésien Alarm en fonction de la taille de la base de connaissance $|K|$ et de la taille de la base de données.

RB	Alarm				Asia			
$\|K\|$	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0	1.800±0.305	1.110±0.350	1.0±0.333	1.157±0.426	0.584±0.645	1.188±0.100	1.0±0.175	1.351±0.106
25	1.360±0.403	1.072±0.343	1.0±0.348	1.253±0.364	0.396±0.619	1.217±0.337	1.0±0.419	1.532±0.386
50	1.345±0.427	1.050±0.363	1.0±0.370	1.614±0.297	1.113±1.03)	1.161±0.567	1.0±0.648	2.000±0.685
75	1.398±0.510	1.023±0.424	1.0±0.430	2.240±0.295	0.967±1.18)	1.093±0.868	1.0±0.951	2.440±0.860
100	1.645±0.689	1.000±0.523	1.0±0.524	3.370±0.326	1.833±1.21)	1.000±0.935	1.0±0.935	4.375±0.832
RB	Insurance				Sachs			
$\|K\|$	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0	1.315±0.243	1.052±0.044	1.0±0.065	1.147±0.040	1.782±0.400	1.697±0.094	1.0±0.538	1.842±0.226
25	1.159±0.322	1.040±0.131	1.0±0.138	1.270±0.140	1.172±0.409	1.413±0.279	1.0±0.499	1.479±0.363
50	1.067±0.326	1.018±0.196	1.0±0.202	1.498±0.190	1.016±0.406	1.245±0.420	1.0±0.522	1.332±0.386
75	1.227±0.419	1.014±0.256	1.0±0.258	2.136±0.303	0.905±0.375	1.111±0.476	1.0±0.530	1.255±0.372
100	0.990±0.398	1.000±0.404	1.0±0.403	2.846±0.307	1.006±0.355	1.000±0.426	1.0±0.426	1.580±0.290
RB	Child				Survey			
$\|K\|$	GSA	CAIRN.2	CAIRN.3	PCSDC	GSA	CAIRN.2	CAIRN.3	PCSDC
0	9.200±2.78	3.823±0.817	1.0±1.23	6.973±1.38	1.728±0.684	1.186±0.781	1.0±0.766	1.475±0.432
25	4.268±2.07	2.000±1.01	1.0±1.02	5.345±1.31	1.886±0.692	1.061±0.781	1.0±0.765	1.370±0.823
50	2.721±1.37	1.169±0.773	1.0±0.784	4.604±1.05	1.833±0.872	1.038±0.792	1.0±0.779	1.567±0.505
75	1.481±0.849	1.086±0.779	1.0±0.795	4.177±0.924	1.542±0.865	1.000±0.822	1.0±0.840	1.463±0.509
100	0.801±0.338	1.000±0.527	1.0±0.531	3.426±0.328	1.693±0.763	1.000±0.856	1.0±0.856	1.723±0.304

TABLEAU 4.9 – SHD relatif de GSA, CAIRN.2, CAIRN.3 et PCSDC en fonction de la taille de la base de connaissance $|K|$ et du réseau bayésien.

Taille	p_{XY}	CAIRN.1	CAIRN.2	CAIRN.3
500	0.8	0.147±0.066	0.942±0.038	0.985±0.021
	1.0	0.684±0.056	0.927±0.041	0.981±0.023
1000	0.8	0.218±0.085	0.953±0.032	0.999±0.005
	1.0	0.732±0.052	0.942±0.035	0.991±0.015
5000	0.8	0.307±0.084	0.953±0.028	0.998±0.007
	1.0	0.703±0.046	0.945±0.035	0.990±0.016
10000	0.8	0.311±0.083	0.954±0.027	0.999±0.005
	1.0	0.700±0.048	0.950±0.035	0.992±0.013

TABLEAU 4.10 – Pourcentage d'orientation de chaque étape de CAIRN pour le réseau bayésien Alarm en fonction de p_{XY} et de la taille de la base de données.

CAIRN n'utilise pas les données, et que l'étape 2 utilise la même approche qu GSA, cette différence de comportement repose avant tout sur l'utilisation d'une étape sous contraintes. Ces mêmes conclusions peuvent également être tirées pour les tableaux 4.6 (resp. 4.8), qui montrent le SHD en fonction des données et de la véracité des connaissances (resp. la taille de la base de connaissances).

Passons au tableau 4.10 et 4.11. Ces tableaux montrent le rapport entre le nombre d'arcs dans le graphe et le nombre d'arêtes. Cela permet d'avoir une idée de la part de graphe qui a été orientée. Cette grandeur est affichée en fonction de la taille de la base de données pour le tableau 4.10 et en fonction du réseau bayésien pour le tableau 4.11. Cette grandeur ne différencie pas les arcs correctement ou mal orientés, on ne va donc pas comparer notre algorithms à GSA et PCSDC à travers cette grandeur. Cependant, nous pouvons constater que la valeur obtenue à la fin de CAIRN.1 est bien plus basse que celle obtenue à la fin de CAIRN.2 ou CAIRN.3. Cela signifie que le graphe à la fin de l'étape sous contraintes est bien moins orienté que ceux à la fin de l'étape à base de score ou de l'étape interactive. Cela s'explique par la prudence dont nous faisons preuve durant l'étape sous contraintes. En effet, nous remettons en question les orientations obtenues avec les v-structures et durant la phase de propagation vis-à-vis des connaissances et des v-structures elles-mêmes. Par exemple, lorsqu'une arête a une double orientation, alors il existe un doute sur sa vraie orientation, et donc aucune n'est choisie jusqu'à l'étape à base de score.

Cette prudence dans l'orientation des arêtes s'explique par notre volonté d'obtenir des orientations causales. Ainsi, lorsqu'une arête est orientée, cette orientation doit être certaine. Sur les tableaux 4.12 et 4.13, on peut voir la précision sur les graphes obtenus à la fin de CAIRN.1, CAIRN.2, CAIRN.3, GSA et PCSDC. Cette précision, qui est donc le rapport entre le nombre d'arcs correctement orientés et le nombre d'arcs trouvés, est présenté en fonction de la taille de la base de données pour le tableau 4.12 et du réseau bayésien pour le tableau 4.13. On peut observer que les graphes obtenus à la fin de l'étape 1, 2 et 3 de CAIRN présentent une meilleure précision

RB	p_{XY}	CAIRN.1	CAIRN.2	CAIRN.3
alarm	0.8	0.307±0.084	0.953±0.028	0.998±0.007
	1.0	0.703±0.046	0.945±0.035	0.990±0.016
asia	0.8	0.439±0.152	0.908±0.091	0.999±0.012
	1.0	0.833±0.118	0.905±0.093	0.989±0.036
child	0.8	0.579±0.113	0.963±0.052	0.998±0.008
	1.0	0.769±0.074	0.940±0.056	0.977±0.035
insurance	0.8	0.455±0.080	0.976±0.037	1.000±0.000
	1.0	0.845±0.052	0.973±0.031	0.991±0.014
sachs	0.8	0.337±0.178	0.830±0.112	0.992±0.023
	1.0	0.822±0.092	0.901±0.079	0.989±0.025
survey	0.8	0.293±0.244	0.974±0.070	0.997±0.022
	1.0	0.760±0.125	0.948±0.102	0.990±0.042

TABLEAU 4.11 – Pourcentage d'orientation de chaque étape de CAIRN en fonction de p_{XY} et du réseau bayésien.

Taille	GSA	CAIRN.1	CAIRN.2	CAIRN.3	PCSDC
500	0.669±0.079	0.735±0.192	0.747±0.072	0.742±0.072	0.615±0.046
1000	0.723±0.093	0.790±0.168	0.855±0.058	0.848±0.059	0.692±0.046
5000	0.746±0.097	0.860±0.089	0.883±0.056	0.877±0.057	0.679±0.047
10000	0.764±0.113	0.877±0.078	0.895±0.063	0.888±0.062	0.673±0.048

TABLEAU 4.12 – Précision des graphes de GSA, CAIRN et PCSDC pour le réseau bayésien Alarm en fonction de la taille de la base de données.

RB	GSA	CAIRN.1	CAIRN.2	CAIRN.3	PCSDC
alarm	0.746±0.097	0.860±0.089	0.883±0.056	0.877±0.057	0.679±0.047
asia	0.831±0.168	0.973±0.147	0.915±0.101	0.899±0.105	0.728±0.112
child	0.828±0.085	0.979±0.054	0.927±0.067	0.919±0.068	0.633±0.063
insurance	0.787±0.081	0.945±0.057	0.884±0.050	0.882±0.049	0.686±0.048
sachs	0.799±0.095	0.754±0.304	0.786±0.130	0.780±0.122	0.751±0.082
survey	0.683±0.211	0.732±0.418	0.811±0.171	0.808±0.172	0.707±0.100

TABLEAU 4.13 – Précision des graphes de GSA, CAIRN et PCSDC en fonction du réseau bayésien.

que ceux des algorithmes GSA et PCSDC. Cela signifie que, parmi les arcs orientés de chaque graphe, la part d'arcs correctement orientés est plus grande dans les graphes de l'algorithme CAIRN que ceux des autres algorithmes. C'est exactement la conséquence de la "prudence" dont nous parlions précédemment. L'orientation donnée aux arcs est plus prudente, et donc plus précise.

Enfin, cette précision peut également être interprétée, vis-à-vis des connaissances, sous la forme de robustesse. Ici, la robustesse désigne la capacité de l'algorithme à avoir de bons résultats malgré la présence d'erreurs dans les connaissances qui lui sont délivrées. Dans les tableaux 4.6 et 4.7, on peut observer le SHD des modèles issus des algorithmes CAIRN.2, CAIRN.3, GSA et PCSDC en fonction de la véracité des connaissances. Dans le tableau 4.6, nous faisons également varier la taille de la base de données. Dans le tableau 4.7, nous faisons varier le réseau bayésien en second paramètre. On peut observer sur les deux tableaux qu'en présence de nombreuses erreurs dans les connaissances, l'algorithme CAIRN obtiendra des modèles ayant un SHD significativement inférieur aux modèles des autres algorithmes, ce qui signifie que les modèles de CAIRN sont de meilleures qualités. On constate notamment des valeurs 2,7 fois plus grandes lorsque les connaissances sont incorrectes et que la base de données est grande. On voit également grâce au tableau 4.7 que cette conclusion n'est pas aussi franche selon le réseau bayésien. La structure du réseau bayésien a donc son importance dans l'influence de la taille de la base de données sur l'apprentissage.

4.5 Conclusion

Dans ce chapitre, nous avons vu un algorithme hybride d'apprentissage de structure de réseau bayésien causal. Cet algorithme permet l'intégration de connaissances incertaines d'expert portant sur l'orientation des relations causales. L'algorithme s'articule autour de trois étapes. La première est une étape sous contraintes qui permet d'apprendre le squelette et d'orienter prudemment les premières v-structures. Le PDAG issu de cette première étape est affiné par une deuxième étape, à base de score, qui permet d'obtenir un PDAG représentant une classe d'équivalence de Markov. Ce graphe est complété durant une ultime étape durant laquelle l'expert intervient directement pour orienter les dernières arêtes.

Les connaissances d'expert sont intégrées durant tout le processus. Durant la première étape, des méthodes issues de PCDC permettent de définir le squelette et les premières orientations. Le score développé dans GSA permet une intégration de ces connaissances dans l'étape à base de score. Et durant l'étape interactive, nous proposons d'optimiser l'élicitation des connaissances via une heuristique visant à diminuer le nombre de questions.

Les expérimentations démontrent que cet algorithme surpasse les autres algorithmes étudiés dans une grande variété de contextes. De plus, l'algorithme reste robuste face aux erreurs dans les connaissances qui lui sont soumises, et permet un apprentissage de haute précision, notamment quand la base de données est grande.

Conclusion

En conclusion, Engie est une entreprise impliquée dans la transition énergétique. Cela implique une optimisation de la gestion des équipements énergétiques, telles que les éoliennes. Ces équipements sont des systèmes complexes, et l'intelligence artificielle a donc totalement sa place pour assister les opérateurs qui sont sur le terrain en charge des équipements. Il est important d'opter pour des modèles interprétables et explicables dans le but d'accompagner la prise de décision. Pour ces raisons, les réseaux bayésiens sont des outils adaptés, mais il est d'abord nécessaire d'en construire la structure. C'est ce qu'on appelle l'apprentissage de structure. Pour aller plus loin dans l'utilisation des réseaux bayésiens, il est également possible de considérer les relations causales dans les systèmes étudiés. Ainsi, on obtient un réseau bayésien causal permettant de mener des raisonnements plus complexes.

Engie possède une expertise sur les équipements qu'elle utilise. Pour apprendre les réseaux bayésiens et réseaux bayésiens causaux, il est donc intéressant d'utiliser non seulement des données, mais également des connaissances d'experts. Dans cette thèse, nous avons donc exploré l'apprentissage des réseaux bayésiens et réseaux bayésiens causaux assisté par des connaissances d'experts.

Dans le chapitre 1, nous avons tout d'abord abordé la définition des réseaux bayésiens. Nous avons vu que ce modèle est composé de deux composantes, une structure et une distribution. Ces deux composantes sont reliées principalement par les indépendances conditionnelles, qui définissent la structure et sont inhérentes à la distribution. Elles peuvent être lues dans la structure à l'aide du critère de d-séparation, et permettent de répartir ces structures dans des classes d'équivalences. Nous avons également traité la causalité, qui est un type de relation supplémentaire pouvant être utilisé avec les réseaux bayésiens pour obtenir des réseaux bayésiens causaux. La causalité permet de raisonner plus loin que les indépendances conditionnelles, notamment grâce aux interventions, mais pose également de nouveaux problèmes en terme d'apprentissage dus au problème d'identifiabilité.

Dans le chapitre 2, nous nous sommes intéressés à l'apprentissage des réseaux bayésiens. L'apprentissage de la structure de réseaux bayésiens à partir de données est un sujet de recherche actif. Dans cette thèse, nous avons abordé un échantillon de ces méthodes, que nous avons distinguées en trois approches : les approches sous contraintes, les approches à base de score et les approches hybrides.

L'utilisation de connaissances d'experts dans l'apprentissage de structure est une solution éprouvée pour améliorer la qualité du modèle appris. Mais l'intégration de ces connaissances est une question difficile, notamment à cause de la diversité des connaissances pouvant être élicitées auprès des experts. Il existe donc pour ce problème un large panel de solutions dans la littérature scientifique, dont deux ont été

présentées en particulier. D'abord PCDC, qui est un algorithme sous contraintes permettant d'utiliser des connaissances structurelles certaines. Et GSA qui est cette fois-ci un algorithme à base de score qui utilise également des connaissances structurelles, mais en prenant en compte l'incertitude des experts.

Enfin, l'apprentissage de réseaux bayésiens causaux a également été traité dans ce chapitre. La difficulté de l'apprentissage de ces modèles par rapport aux réseaux bayésiens classiques a été mise en évidence. En effet, il est plus difficile, voire parfois impossible, de trouver la totalité du graphe d'un modèle causal. Cependant, nous pouvons tout de même souligner que certaines approches permettent d'obtenir des graphes partiels, révélant par conséquent une partie des relations causales du système.

Les deux chapitres suivants sont dédiés aux contributions scientifiques de cette thèse. D'abord, dans le chapitre 3, nous avons développé l'algorithme PCSe. Il s'agit d'un algorithme sous contraintes d'apprentissage de structure de réseaux bayésiens permettant l'intégration de connaissances incertaines portant sur les indépendances et dépendances du modèle. Cet algorithme est innovant dans le sens où il permet d'intégrer intelligemment l'incertitude dans les tests d'indépendance, et donc au cœur même de l'approche sous contraintes. Pour cela, un nouveau test d'indépendance a été développé, conçu pour intégrer ces connaissances. Des expérimentations ont montré que cette intégration se fait avec succès.

Dans le chapitre 4, nous nous sommes intéressés à l'apprentissage de modèles causaux. L'algorithme présenté est cette fois-ci un algorithme hybride d'apprentissage de structure de réseau bayésien causal, encore une fois capable de prendre en compte des connaissances incertaines d'expert. En revanche, les connaissances portent ici directement sur les relations causales entre les variables. Cet algorithme hybride est divisé en trois étapes. La première étape sous contraintes a pour but de construire un premier graphe partiel et trouver les premières orientations causales. La seconde étape est à base de score, et vise à définir le modèle d'indépendance précis de la distribution, et donc celui de la structure. Ces deux premières étapes utilisent les mêmes connaissances, de manières différentes, mais toujours avec leurs incertitudes. La dernière étape se distingue des deux premières, et repose sur une interaction avec l'expert dans le but de compléter les orientations causales du modèle. Les connaissances utilisées à ce moment sont donc des connaissances nouvelles, provenant des réponses apportées par l'expert aux questions qui lui sont posées. Ces questions sont choisies méthodiquement, en tenant compte des informations manquantes dans le graphe. Des expérimentations ont ici aussi montré la pertinence de cette méthode, qui a été développée avec un objectif de précision de l'apprentissage.

En résumé, cette thèse présente deux algorithmes d'apprentissage de réseaux bayésiens et de réseaux bayésiens causaux utilisant des connaissances incertaines d'experts. Chacun de ces algorithmes a fait l'objet d'une publication scientifique, et ils ont été présentés dans des conférences internationales. Cette thèse participe ainsi à la recherche dans le domaine de l'apprentissage de structure de réseaux bayésiens, et plus précisément dans l'apprentissage en présence de connaissances d'experts. Comme pour toutes recherches scientifiques, les contributions de cette thèse ouvrent la porte à de nombreuses possibilités pour de futures recherches.

D'abord, bien que le sujet des éoliennes fut une motivation importante pour l'élaboration de cette thèse, aucune expérimentation n'a pu être menée sur un jeu de données concernant ces systèmes de production d'énergie. Comme nous l'avons vu, les expérimentations sont limitées à des jeux de données couramment utilisés dans la littérature scientifique liée aux réseaux bayésiens et réseaux bayésiens causaux. Or Engie possède, pour certains de ses parcs éoliens, des bases de données qui retranscrivent de nombreuses variables décrivant le fonctionnement des éoliennes, telles que les températures de différentes pièces, les vitesses de rotations, les quantités d'énergie générées et même les conditions météorologiques. Une perspective pour la recherche au sein de l'entreprise est donc d'appliquer ces algorithmes, avec les connaissances à dispositions, pour apprendre et perfectionner un modèle permettant de répondre aux exigences de l'entreprise en termes de diagnostic, de pronostic, de prise de décision et de prédiction. Ces travaux impliqueraient notamment d'évaluer l'impact des connaissances sur ces différentes tâches.

Cependant, une très grande partie de ces données sont sous la forme de variables continues. Or les algorithmes présentés ci-dessus ont tous été conçus pour des variables discrètes. Même s'il existe des techniques de discrétisation, celles-ci impliquent souvent des pertes d'informations importantes et peuvent poser des problèmes de performances. Il pourrait alors être intéressant d'adapter les algorithmes vus précédemment à des données continues. Plusieurs solutions existent pour utiliser des variables continues dans un réseau bayésien. La plus courante consiste à supposer que la distribution jointe est une distribution gaussienne multivariée. Mais cette hypothèse est très restrictive. De plus, l'utilisation de telles distributions nécessite la conception de nouvelles méthodes pour incorporer le savoir d'experts dans des tests d'indépendance fondés sur les matrices de variance-covariance.

D'autre part, le score BDeu n'est pas non plus adapté au données continues. Pour celles-ci, sous certaines conditions remplies par des distributions gaussiennes, il est possible d'utiliser le score BGe (Geiger et Heckerman 2002, Kuipers, Moffa et Heckerman 2014). Mais cela remet en question l'intégration des connaissances proposées par Amirkhani, Rahmati, Lucas et al. 2017 et utilisée dans l'algorithme CAIRN, et il est donc nécessaire de vérifier et, le cas échéant, démontrer que cette intégration est toujours possible.

Ensuite, rappelons que l'objectif de ces recherches étaient d'étendre le champ des possibles en matières d'intégrations des connaissances dans des algorithmes d'apprentissage de structure de réseaux bayésiens. Des méthodes ont donc été développées dans ce sens, mais seulement pour des connaissances spécifiques. Il peut être intéressant d'enrichir ces algorithmes pour permettre la prise en compte de connaissances plus diversifiées. Cette diversité s'exprime soit par le caractère certain ou incertain des connaissances, soit par leur sujet (ce sur quoi elles portent). Or l'utilisation de connaissances incertaines implique nécessairement la possibilité d'utiliser des connaissances certaines. En d'autres termes, les algorithmes que nous avons pu voir dans ce manuscrit laissaient la possibilité de prendre en compte des connaissances certaines, simplement en poussant le paramètre de certitude au maximum. La question ne se pose donc pas sur l'incertitude ou la certitude des connaissances.

C'est donc avant tout par rapport au sujet des connaissances que la diversification doit s'opérer. L'algorithme PCSe peut utiliser des connaissances sur les indépendances, mais il est envisageable d'utiliser en même temps des connaissances plus structurelles. Par connaissances structurelles, nous entendons ici des connaissances portant directement sur la présence ou l'absence d'arcs ou d'arêtes, à la manière de PCDC. Nous avons vu que deux problèmes ont été relevés pour l'utilisation des connaissances de l'algorithme PCSe

— le premier problème pose la question des actions à mener lorsque la base de données est trop petite pour effectuer des tests d'indépendance suffisamment précis pour être interprétables
— le second problème s'intéresse à la résolution des conflits entre la décision en fonction de la base de données et la décision en fonction de la base de données et des connaissances lors de l'interprétation du résultat du test d'indépendance.

Or les connaissances structurelles n'apportent pas directement d'informations sur la résolution des tests d'indépendance, ces deux problèmes peuvent donc être ignorés.

Ainsi, des connaissances structurelles certaines peuvent tout à fait accompagner les connaissances sur les indépendances. Cependant, prendre en compte l'incertitude des connaissances structurelles dans le cadre d'algorithmes sous contraintes est un défi en soit. Une solution possible consiste à utiliser des algorithmes à base de solveur (HYTTINEN, P. HOYER, EBERHARDT et al. 2013, HYTTINEN, EBERHARDT et JÄRVISALO 2014). Ces méthodes reposent sur une conversion des contraintes sur les indépendances trouvées dans les données en contraintes sur la présence et absence de chemin dans la structure. L'utilisation d'un solveur SAT, qui est une forme de solveur logique, doit ensuite permettre de résoudre le problème de satisfaisabilité ainsi soulevé. Une des principales difficultés de ces approches est la complexité. En effet, résoudre des problèmes grâce à un solveur prend du temps, ce qui implique un travail d'optimisation important pour pouvoir traiter des problèmes complexes avec beaucoup de variables.

En ce qui concerne CAIRN, l'algorithme utilise déjà des connaissances structurelles avec une dimension causale. Il est néanmoins possible de combiner dans un même algorithme les connaissances utilisées dans PCSe et celles de CAIRN. En effet, nous avons vu dans le chapitre 2 que les indépendances peuvent servir à apprendre un graphe causal partiel. Ainsi, une autre perspective est la fusion des algorithmes PCSe et CAIRN, pour obtenir un algorithme permettant d'utiliser les deux types de connaissances. Cette perspective a l'avantage de servir pleinement les objectifs d'Engie, puisque toutes les connaissances des experts de l'entreprise évoquées dans cette thèse pourraient être utilisables et donc valorisées au sein d'un même algorithme. Notons que les expériences ont montré que plus il y a de connaissances à disposition, plus le modèle appris est de bonne qualité. On peut légitimement penser que cela reste vrai, même si les connaissances sont hétérogènes. Ce résultat, s'il se vérifie, pourrait aboutir à des méthodes prometteuses.

Par ailleurs, l'utilisation de connaissances sur les indépendances dans l'algorithme CAIRN ne poserait a priori pas de grandes difficultés. En effet, les méthodes spécifiques aux intégrations des connaissances de PCSe et celles de CAIRN peuvent coexister

dans un même algorithme, puisque ces méthodes n'interviennent pas sur les mêmes éléments algorithmiques. Plus précisément, les méthodes de CAIRN interviennent notamment sur la restriction des tests d'indépendances. Au contraire, les méthodes de PCSe interviennent sur l'exécution des tests. Ainsi, on peut tout à fait retreindre l'ensemble des tests d'indépendance qui doivent être faits, tout en altérant leurs conclusions. Cependant, cela révèle une potentielle difficulté dans la cohérence des deux ensembles de connaissances. On peut en effet se demander si les connaissances de CAIRN ne pourraient pas empêcher les connaissances de PCSe de s'exprimer, et sous quelles conditions la cohabitation est possible.

Enfin, une dernière piste d'étude est la considération des connaissances d'Engie sous la forme d'une ontologie pour aider à l'apprentissage du réseau bayésien. En effet, Engie a également travaillé à la formalisation de ses connaissances sous la forme d'ontologies, des modèles de connaissances retranscrivant des relations entre des concepts sous la forme de triplets (GRUBER 1993). Une fois l'ensemble des connaissances formalisé, il serait possible d'exploiter l'ontologie de manière automatique dans un algorithme d'apprentissage de structure de réseau bayésien causal. C'est ce qu'ont proposé CHEN, LU, WITHERELL et al. 2021 dans leur article, dans lequel ils expliquent comment ils ont utilisé une ontologie pour obtenir un modèle causal utilisé dans le cadre d'un processus de fabrication additive. Dans le contexte des travaux d'Engie, l'entreprise a produit une ontologie sur le domaine des éoliennes. Celle-ci pourrait alors servir de point de départ pour des travaux dans cette direction.

Bibliographie

[Agr+96] Bruce AGRAMSON, John BROWN, Allan MURPHY et Robert L. WINKLER. « Hailfinder : A Bayesian system for forecasting severe weather ». In : *International Journal of Forecasting* 12 (1996), p. 57-71 (cf. p. 112).

[Aka70] Hirotugu AKAIKE. « Statistical predictor identification ». In : *Annals of the institute of Statistical Mathematics* 22.1 (1970), p. 203-217 (cf. p. 64).

[Ami+17] Hossein AMIRKHANI, Mohammad RAHMATI, Peter J. F. LUCAS et Arjen HOMMERSOM. « Exploiting Experts' Knowledge for Structure Learning of Bayesian Networks ». In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.11 (2017), p. 2154-2170 (cf. p. 78, 79, 152).

[BP63] Thomas BAYES et Richard PRICE. « An Essay towards Solving a Problem in the Doctrine of Chances ». In : *Philosophical Transactions* 53 (1763), p. 370-418 (cf. p. 24).

[Bei+89] Ingo A. BEINLIC, H. J. SUERMONDT, R. Martin CHAVEZ et Gregory F. COOPER. « The ALARM Monitoring System : A Case Study with two Probabilistic Inference Techniques for Belief Networks ». In : *Medical Informatics* 38 (1989) (cf. p. 112).

[Ber63] Claude BERGE. *Théorie des graphes et ses applications - 2e édition*. Dunod, 1963 (cf. p. 21).

[Bin+97] John BINDER, Daphne KOLLER, Stuart RUSSEL et Keyki KANAZAWA. « Adaptive Probabilistic Networks with Hidden Variables ». In : *Machine Learning* 29 (1997), p. 213-244 (cf. p. 112).

[Bou93] Remco R. BOUCKAERT. « Probabilistic network construction using the minimum description length principle ». In : *European conference on symbolic and quantitative approaches to reasoning and uncertainty*. Springer. 1993, p. 41-48 (cf. p. 66).

[Bou95] Remco R. BOUCKAERT. « Bayesian belief networks : from construction to inference ». Thèse de doct. 1995 (cf. p. 66, 69, 71).

[BV17] Kailash BUDHATHOKI et Jilles VREEKEN. « MDL for causal inference on discrete data ». In : *2017 IEEE International Conference on Data Mining (ICDM)*. 2017, p. 751-756 (cf. p. 90, 91).

[BV18] Kailash BUDHATHOKI et Jilles VREEKEN. « Accurate causal inference on discrete data ». In : *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE. 2018, p. 881-886 (cf. p. 90).

[Bun91] Wray BUNTINE. « Theory Refinement on Bayesian Networks ». In : *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 1991, p. 52-60 (cf. p. 65, 66, 69, 71).

[CZJ09] Cassio P. de CAMPOS, Zhi ZENG et Qiang JI. « Structure Learning of Bayesian Networks Using Constraints ». In : *Proceedings of the 26th Annual International Conference on Machine Learning*. Association for Computing Machinery, 2009, p. 113-120 (cf. p. 70, 71, 77).

[CP01] Luis De CAMPOS et Jose PUERTA. « Stochastic local algorithms for learning belief networks : Searching in the space of the orderings ». In : *Proceedings of the Sixth European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty*. Springer. 2001, p. 228-239 (cf. p. 67, 68, 70, 71).

[CC07] Luis M. de CAMPOS et Javier G. CASTELLANO. « Bayesian network learning algorithm using structural restrictions ». In : *International Journal of Approximate Reasoning* 45 (2007), p. 233-254 (cf. p. 77, 79, 81, 112).

[CGP02] Luis M. de CAMPOS, José A. GÁMEZ et José M. PUERTA. « Learning Bayesian networks by Ant Colony Optimisation : searching in two different spaces ». In : *Mathware and Soft Computing* 9 (2002), p. 251-268 (cf. p. 71).

[Che+21] Ruimin CHEN, Yan LU, Paul WITHERELL, Timothy SIMPSON, Soundar KUMARA et Hui YANG. « Ontology-Driven Learning of Bayesian Network for Causal Inference and Quality Assurance in Additive Manufacturing ». In : *IEEE Robotics and Automation Letters* 6.3 (2021), p. 6032-6038 (cf. p. 154).

[Che+02] Jie CHENG, Russell GREINER, Jonathan KELLY, David BELL et Weiru LIU. « Learning Bayesian networks from data : An information-theory based approach ». In : *Artificial Intelligence* 137.1 (2002), p. 43-90 (cf. p. 77).

[Chi96] David Maxwell CHICKERING. « Learning Bayesian Networks is NP-Complete ». In : *Learning from Data*. Sous la dir. de Lenz HJ. FISHER D. T. 112. Lecture Notes in Statistics. Springer, 1996, p. 121-130 (cf. p. 50).

[Chi02a] David Maxwell CHICKERING. « Learning equivalence classes of Bayesian-network structures ». In : *The Journal of Machine Learning Research* 2 (2002), p. 445-498 (cf. p. 67).

[Chi02b] David Maxwell CHICKERING. « Optimal structure identification with greedy search ». In : *Journal of machine learning research* 3.Nov (2002), p. 507-554 (cf. p. 67, 69, 71, 76).

[CH96] David Maxwell CHICKERING et David HECKERMAN. « Efficient approximations for the marginal likelihood of incomplete data given a Bayesian network ». In : *Proceedings of the Twelfth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 1996, p. 158-168 (cf. p. 65).

[CM14] Diego COLOMBO et Marloes H. MAATHUIS. « Order-Independent Constraint-Based Causal Structure Learning ». In : *Journal of Machine Learning Research* 15 (2014), p. 3921-3962 (cf. p. 57, 110, 112).

[Con+97] Cristina CONATI, Abigail S. GERTNER, Kurt VANLEHN et Marek J. DRUZDZEL. « On-Line Student Modeling for Coached Problem Solving Using Bayesian Networks ». In : *International Centre for Mechanical Sciences (Courses and Lectures)* 383 (1997) (cf. p. 112).

[CH92] Gregory F COOPER et Edward HERSKOVITS. « A Bayesian method for the induction of probabilistic networks from data ». In : *Machine learning* 9.4 (1992), p. 309-347 (cf. p. 65, 66, 71, 78).

[Daw79] A. Philip DAWID. « Conditional independence in statistical theory ». In : *Journal of the Royal Statistical Society : Series B (Methodological)* 41.1 (1979), p. 1-15 (cf. p. 27).

[DGW20] Gaspard DUCAMP, Christophe GONZALES et Pierre-Henri WUILLEMIN. « aGrUM/pyAgrum : a toolbox to build models and algorithms for Probabilistic Graphical Models in Python ». In : *Proceedings of the International Conference on Probabilistic Graphical Models*. 2020, p. 609-612 (cf. p. 114, 141).

[FNP99] Nir FRIEDMAN, Iftach NACHMAN et Dana PE'ER. « Learning Bayesian Network Structure from Massive Datasets : The Sparse Candidate Algorithm ». In : *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann. 1999 (cf. p. 72).

[GP95] David GALLES et Judea PEARL. « Testing Identifiability of Causal Effects ». In : *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 1995, p. 185-195 (cf. p. 43).

[GH02] Dan GEIGER et David HECKERMAN. « Parameter priors for directed acyclic graphical models and the characterization of several probability distributions ». In : *The Annals of Statistics* 30.5 (2002), p. 1412-1440 (cf. p. 152).

[Glo86] Fred GLOVER. « Future paths for integer programming and links to artificial intelligence ». In : *Computers & operations research* 13.5 (1986), p. 533-549 (cf. p. 69).

[GP92] Moisés GOLDSZMIDT et Judea PEARL. « Rank-Based Systems : A Simple Approach to Belief Revision, Belief Update, and Reasoning about Evidence and Actions ». In : *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers Inc., 1992, p. 661-672 (cf. p. 38).

[Gru93] Thomas R. GRUBER. « A translation approach to portable ontology specifications ». In : *Knowledge Acquisition* 5.2 (1993), p. 199-220 (cf. p. 154).

[Grü+08] Peter D GRÜNWALD et al. « Algorithmic information theory ». In : *Handbook of the Philosophy of Information* (2008), p. 281-320 (cf. p. 90).

[HC20] Aric HAGBERG et Drew CONWAY. *NetworkX : Network Analysis with Python*. 2020 (cf. p. 141).

[HGC95] David HECKERMAN, Dan GEIGER et David Maxwell CHICKERING. « Learning Bayesian Networks : The Combination of Knowledge and Statistical Data ». In : *Machine Learning* 20 (sept. 1995), p. 197-243 (cf. p. 65, 71, 78).

[Hu+12] Yong HU, Xizhu MO, Xiangzhou ZHANG, Yuran ZENG, Jianfeng DU et Kang XIE. « Intelligent Analysis Model for Outsourced Software Project Risk Using Constraint-based Bayesian Network ». In : *Journal of Software* 7.2 (fév. 2012), p. 440-449 (cf. p. 77).

[HEJ14] Antti HYTTINEN, Frederick EBERHARDT et Matti JÄRVISALO. « Constraint-based causal discovery : conflict resolution with answer set programming ». In : *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*. 2014, p. 340-349 (cf. p. 153).

[Hyt+13] Antti HYTTINEN, Patrik HOYER, Frederick EBERHARDT et Matti JÄRVISALO. « Discovering cyclic causal models with latent variables : a general SAT-based procedure ». In : *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*. 2013, p. 301-310 (cf. p. 153).

[JWL13] Junzhong JI, Hongkai WEI et Chunnian LIU. « An artificial bee colony algorithm for learning Bayesian networks ». In : *Soft Computing* 17.6 (2013), p. 983-994 (cf. p. 71).

[KR76] Ralph L KEENEY et Howard RAIFFA. *Decisions with Multiple Objectives - Preferences and Value Tradeoffs*. Cambridge university Press, 1976 (cf. p. 109).

[KSC84] Harri KIIVERI, Terry P SPEED et John B CARLIN. « Recursive causal models ». In : *Journal of the australian Mathematical Society* 36.1 (1984), p. 30-52 (cf. p. 28).

[KGV83] Scott KIRKPATRICK, Daniel GELATT JR et Mario VECCHI. « Optimization by Simulated Annealing ». In : *Science* 220.4598 (1983), p. 671-680 (cf. p. 70).

[KF09] Daphne KOLLER et Nir FRIEDMAN. *Probabilistic graphical models : principles and techniques*. MIT press, 2009 (cf. p. 63).

[Kol33] Andrei KOLMOGOROV. *Grundbegriffe der wahrscheinlichkeitsrechnung*. Springer-Verlag, 1933 (cf. p. 23).

[Kol65] Andrei KOLMOGOROV. « Three approaches to the quantitative definition of information' ». In : *Problemy Peredachi Informatsii* 1.1 (1965), p. 3-11 (cf. p. 90).

[KMH14] Jack KUIPERS, Giusi MOFFA et David HECKERMAN. « Addendum on the scoring of Gaussian directed acyclic graphical models ». In : *The Annals of Statistics* 42.4 (2014), p. 1689-1691 (cf. p. 152).

[Lar+96] Pedro LARRANAGA, Mikel POZA, Yosu YURRAMENDI, Roberto H. MURGA et Cindy M. H. KUIJPERS. « Structure learning of Bayesian networks by genetic algorithms : A performance analysis of control parameters ». In : *IEEE transactions on pattern analysis and machine intelligence* 18.9 (1996), p. 912-926 (cf. p. 70, 71).

[LS88] Steffen L. LAURITZEN et David J. SPIEGELHALTER. « Local Computation with Probabilities on Graphical Structures and their Application to Expert Systems (with discussion) ». In : *Journal of the Royal Statistical Society : Series B* 50 (1988), p. 157-224 (cf. p. 30, 112, 133).

[LB18] Andrew LI et Peter BEEK. « Bayesian Network Structure Learning with Side Constraints ». In : *Proceedings of the International Conference on Probabilistic Graphical Models*. T. 72. 2018, p. 225-236 (cf. p. 77).

[LG18] Hongru LI et Huiping GUO. « A Hybrid Structure Learning Algorithm for Bayesian Network Using Experts' Knowledge ». In : *Entropy* 20 (août 2018), p. 620 (cf. p. 78).

[LC16] Furui LIU et Laiwan CHAN. « Causal inference on discrete data via estimating distance correlations ». In : *Neural computation* 28.5 (2016), p. 801-814 (cf. p. 91).

[LSM10] Roger LUIS, L Enrique SUCAR et Eduardo F MORALES. « Inductive transfer for learning Bayesian networks ». In : *Machine learning* 79.1 (2010), p. 227-255 (cf. p. 78).

[MT99] Dimitris MARGARITIS et Sebastian THRUN. « Bayesian Network Induction via Local Neighborhoods ». In : *Advances in Neural Information Processing Systems*. Sous la dir. de S. SOLLA, T. LEEN et K. MÜLLER. T. 12. MIT Press, 1999 (cf. p. 59).

[Mee95] Christopher MEEK. « Causal Inference and Causal Explanation with Background Knowledge ». In : *Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence*. 1995, p. 403-410 (cf. p. 54, 128).

[NHM18] Preetam NANDY, Alain HAUSER et Marloes MAATHUIS. « High-dimensional consistency in score-based and hybrid structure learning ». In : *The Annals of Statistics* 46.6A (2018), p. 3151-3183 (cf. p. 74, 75).

[Pea86] Judea PEARL. *Markov and Bayes Networks : a Comparison of two graphical Representations of Probabilistic Knowledge*. Rapp. tech. UCLA, Computer Science Department, 1986 (cf. p. 32).

[Pea88] Judea PEARL. *Probabilistic reasoning in intelligent systems : networks of plausible inference*. Morgan kaufmann, 1988 (cf. p. 24, 30, 31).

[Pea93] Judea PEARL. « Comment : Graphical Models, Causality and Intervention ». In : *Statistical Science* 8.3 (1993), p. 266-269 (cf. p. 41).

[Pea95] Judea PEARL. « Causal Diagrams for Empirical Research ». In : *Biometrika* 82 (1995), p. 669-710 (cf. p. 39, 42).

[Pea09] Judea PEARL. *Causality*. 2nd. Cambridge University Press, 2009 (cf. p. 25, 35, 122).

[Pea10] Judea PEARL. « Causal inference ». In : *Causality : objectives and assessment* (2010), p. 39-58 (cf. p. 87).

[PV91] Judea PEARL et Thomas VERMA. « A theory of inferred causation ». In : *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*. 1991, p. 441-452 (cf. p. 88).

[PJS10] Jonas PETERS, Dominik JANZING et Bernhard SCHÖLKOPF. « Identifying cause and effect on discrete data using additive noise models ». In : *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010, p. 597-604 (cf. p. 89).

[Ram+17] Joseph RAMSEY, Madelyn GLYMOUR, Ruben SANCHEZ-ROMERO et Clark GLYMOUR. « A million variables and more : the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images ». In : *International journal of data science and analytics* 3.2 (2017), p. 121-129 (cf. p. 67, 71).

[RSZ06] Joseph RAMSEY, Peter SPIRTES et Jiji ZHANG. « Adjacency-Faithfulness and Conservative Causal Inference ». In : *Proceedings of the Twenty-Second Conference on Uncertainty in Artificial Intelligence*. 2006, p. 401-408 (cf. p. 57).

[Rei56] Hans REICHENBACH. *The direction of time*. University of California Press, 1956 (cf. p. 88).

[RD03] Matthew RICHARDSON et Pedro DOMINGOS. « Learning with Knowledge from Multiple Experts ». In : *Proceedings of the International Conference on Machine Learning* 2 (oct. 2003), p. 624-631 (cf. p. 78, 79).

[Ris78] Jorma RISSANEN. « Modeling by shortest data description ». In : *Automatica* 14.5 (1978), p. 465-471 (cf. p. 66).

[Rob94] Christian ROBERT. *The Bayesian Choice : A Decision-Theoretic Motivation*. Springer New York, NY, 1994 (cf. p. 50).

[Rob77] Robert W ROBINSON. « Counting unlabeled acyclic digraphs ». In : *Combinatorial mathematics V* (1977), p. 28-43 (cf. p. 67).

[Sac+05] Karen SACHS, Omar PEREZ, Dana PE'ER, Douglas A LAUFFENBURGER et Garry P NOLAN. « Causal protein-signaling networks derived from multiparameter single-cell data ». In : *Science* 308.5721 (2005), p. 523-529 (cf. p. 112).

[Sap90] Gilbert SAPORTA. *PRobabilités, analyse des données et statistique*. Editions Technip, 1990 (cf. p. 53).

[Sch78] Gideon SCHWARZ. « Estimating the dimension of a model ». In : *The annals of statistics* (1978), p. 461-464 (cf. p. 64).

[SD21] Marco SCUTARI et Jean-Baptiste DENIS. *Bayesian networks : with examples in R*. Chapman et Hall/CRC, 2021 (cf. p. 112).

[Sha48] Claude Elwood SHANNON. « A mathematical theory of communication ». In : *The Bell system technical journal* 27.3 (1948), p. 379-423 (cf. p. 90).

[Shi+06] Shohei SHIMIZU, Patrik O HOYER, Aapo HYVÄRINEN, Antti KERMINEN et Michael JORDAN. « A linear non-Gaussian acyclic model for causal discovery. » In : *Journal of Machine Learning Research* 7.10 (2006) (cf. p. 89).

[SP06] Ilya SHPITSER et Judea PEARL. « Identification of Joint Interventional Distributions in Recursive Semi-Markovian Causal Models ». In : *Proceedings of the Twenty-First National Conference on Artificial Intelligence*. AAAI Press, 2006, p. 1219-1226 (cf. p. 42, 43).

[SV93] Moninder SINGH et Marco VALTORTA. « An algorithm for the construction of Bayesian network structures from data ». In : *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence*. Elsevier. 1993, p. 259-265 (cf. p. 70).

[SC92] David J. SPIEGELHALTER et Robert G. COWELL. « Learning in probabilistic expert systems ». In : 1992 (cf. p. 112).

[SG91] Peter SPIRTES et Clark GLYMOUR. « An algorithm for Fast Recovery of Sparse Causal Graphs ». In : *Social Science Computer Review* 9.1 (1991), p. 62-72 (cf. p. 56).

[Ste+10] Oliver STEGLE, Dominik JANZING, Kun ZHANG, Joris M MOOIJ et Bernhard SCHÖLKOPF. « Probabilistic latent variable models for distinguishing between cause and effect ». In : *Advances in neural information processing systems* 23 (2010) (cf. p. 91).

[TK05] Marc TEYSSIER et Daphne KOLLER. « Ordering-Based Search : A Simple and Effective Algorithm for Learning Bayesian Networks ». In : *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*. 2005, p. 584-590 (cf. p. 67).

[TP02] Jin TIAN et Judea PEARL. « A General Identification Condition for Causal Effects ». In : *Eighteenth National Conference on Artificial Intelligence*. American Association for Artificial Intelligence, 2002, p. 567-573 (cf. p. 43).

[TAS03] Ioannis TSAMARDINOS, Constantin ALIFERIS et Alexander STATNIKOV. « Algorithms for Large Scale Markov Blanket Discovery ». In : jan. 2003, p. 376-381 (cf. p. 60, 61).

[TBA06] Ioannis TSAMARDINOS, Laura BROWN et Constantin ALIFERIS. « The max-min hill-climbing Bayesian network structure learning algorithm ». In : *Machine learning* 65.1 (2006), p. 31-78 (cf. p. 72, 74).

[TK92] Amos TVERSKY et Daniel KAHNEMAN. « Advances in Prospect Theory : Cumulative Representation of Uncertainty ». In : *Journal of Risk and Uncertainty* 5.4 (1992), p. 297-323 (cf. p. 98).

[Ver93] Thomas VERMA. *Graphical aspects of causal models*. Rapp. tech. UCLA, Computer Science Department, 1993 (cf. p. 37).

[VP90] Thomas VERMA et Judea PEARL. « Equivalence and Synthesis of Causal Models ». In : *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*. USA : Elsevier Science Inc., 1990, p. 255-270 (cf. p. 34, 55).

[VP92] Thomas VERMA et Judea PEARL. « An algorithm for deciding if a set of observed independencies has a causal explanation ». In : *Proceedings of the Eighth Conference on Uncertainty in Artificial Intelligence*. Morgan Kaufmann Publishers Inc., 1992, p. 323-330 (cf. p. 54).

[YM05] Sandeep YARAMAKALA et Dimitris MARGARITIS. « Speculative Markov Blanket Discovery for Optimal Feature Selection ». In : *Proceedings of the Fifth IEEE International Conference on Data Mining*. IEEE Computer Society, 2005, p. 809-812 (cf. p. 60, 61).

[YM13] Changhe YUAN et Brandon MALONE. « Learning optimal Bayesian networks : A shortest path perspective ». In : *Journal of Artificial Intelligence Research* 48 (2013), p. 23-65 (cf. p. 70, 71).

Index

A
Algorithme, 54
    A*, 70
    Algorithme génétique, 70
    ARGES, 75
    B, 69
    B&B, 70
    CAIRN, 121, 139
    Colonie d'abeilles, 70
    Colonie de fourmis, 70
    Constraint-Bayesian, 70
    CPC, 57
    Fast-IAMB, 61
    FGES, 67
    GES, 67, 69
    GSA, 81, 132
    GSMB, 59
    IAMB, 60
    IC, 55
    MMHC, 74
    MMPC, 74
    PC, 56
    PC stable, 57
    PCDC, 79, 126
    PCSDC, 112
    PCSe, 93, 110
    SLA, 77
    Sparse Candidate, 72
    VSN, 70
Ancêtre, 21
ANM, 89
Apprentissage, 44
    Paramètres, 45
    Structure, 50
Arc, 21
Arête, 21

B
Backdoor, 41
Base de données
    Interventionnelle, 40
    Observationnelle, 40

C
Causal
    Apprentissage, 87
    Découverte, 87
    Effet, 40
    Inférence, 89
Causalité, 35
Cause, 38
Chaîne, 21
Chaîne active, 29
Chemin, 21
Circuit, 21
Classe d'équivalence de Markov, 34
Collision, 29
Complexité de Kolmogorov, 90
Connaissance d'expert, 76, 95
    Certaine, 76
    Incertaine, 78, 95, 123
Connexion convergente, 29
Connexion divergente, 28
Couverture de markov, 31
CPDAG, 35
Cycle, 21

D
D-connexion, 30
D-map, 32
D-séparation, 29
DAG, 22
Descendant, 21
Distance Structurelle de Hamming, 139
Distribution de probabilité, 23
    Conditionnelle, 24
    Jointe, 23
    Marginale, 23
    Positive, 23
Do-calculus, 39

E
Effet, 38
Enfant, 21
Époux, 31
Équivalence de Markov, 34
Espace de recherche, 66
Espérance a posteriori, 50
Événement, 23

F
F-score, 113
Factorisation, 24
Faux négatif, 113
Faux positif, 113, 141
Fidélité, 123
    d'adjacence, 58
    d'orientation, 59
Fidélité d'adjacence, 123
Formule des probabilités
    Composées, 24
    Totales, 24
Frontdoor, 42

G
Graphe, 21
Graphoïde, 27

H
Heuristique, 69

I
I-map, 32
    Minimale, 33
Indépendance, 26
    Conditionnelle, 26
Information mutuelle, 60, 72
Information élémentaire, 25
Intervention, 38

L
Liste tabou, 69
Log-vraisemblance, 46

M
Maximum a posteriori, 49
Maximum de vraisemblance, 47

Modèle d'indépendance, 27
Modèle markovien, 37
Modèle semi-markovien, 37
Modèle structurel, 37

N
Nœud, 21

O
Opérateur *do*, 38

P
P-map, 32
P-value, 53
Parent, 21
PDAG, 34
Propriété locale de Markov, 28
Précision, 113, 141

R
Rappel, 113
Recherche gloutonne, 69
Recuit simulé, 70
Redémarrage aléatoire, 70
Relation d'ordre strict, 68
Requête
    de probabilités conditionnelles, 25
    MAP, 26
    MPE, 26
Règles de Meek, 54
Réseau bayésien, 24
    Causal, 35, 87

S
Score, 63
    AIC, 64
    BD, 65
    BDe, 65
    BDeu, 65
    BIC, 64
    K2, 65
    MDL, 66
Score décomposable, 63
Score équivalent, 63
Semi-graphoïde, 27
Squelette, 21

Suffisance causale, 123

T
Test d'indépendance, 52, 95
Théorème de Bayes, 24

U
Univers, 23

V
V-structure, 29
Variable aléatoire, 23
    Continue, 23
    Discrète, 23
Vrai négatif, 113
Vrai positif, 113, 141
Vraisemblance, 46, 64, 96, 99, 107