# Estimation et sélection de modèle pour le modèle des blocs latents

Vincent Brault

# Estimation et sélection de modèle pour le modèle des blocs latents

Vincent Brault

UNIVERSITÉ PARIS SUD

Comprendre le monde, construire l'avenir®

Département de mathématiques d'Orsay


UNIVERSITÉ PARIS-SUD

ÉCOLE DOCTORALE 142 : MATHÉMATIQUES DE LA RÉGION PARIS-SUD
LABORATOIRE DE MATHÉMATIQUES DE LA FACULTÉ DES SCIENCES D'ORSAY

DISCIPLINE : MATHÉMATIQUES


THÈSE DE DOCTORAT

Soutenue le mardi 30 septembre 2014 par


**Vincent Brault**


# Estimation et sélection de modèle pour le modèle des blocs latents

| | | |
|---|---|---|
| **Directeur de thèse :** | M. Gilles Celeux | Directeur de Recherche (INRIA) |
| **Co-directrice de thèse :** | Mme Christine Keribin | Maître de conférence (Université Paris-Sud) |
| **Composition du jury :** | | |
| Président du jury : | M. Pascal Massart | Professeur (Université Paris-Sud) |
| Rapporteurs : | M. Christophe Biernacki | Professeur (Université Lille 1) |
| | Mme Florence Forbes | Directrice de Recherche (INRIA) |
| Examinateurs : | M. Mohamed Nadif | Professeur (Université Paris Descartes) |

# Remerciement

*"De toute façon, ce sont surtout les remerciements qui sont lus."*
Patrick Bouvier

On a beau être prévenu, regarder les phd comics ou autres sites sur la vie d'un thésard, on se lance quand même dans l'aventure. Il serait chimérique de dire qu'une thèse est un long fleuve tranquille : elle est faite de moments de doute, de remises en question et de sentiments de solitudes. Mais ce sont eux qui nous donnent la rage de vaincre et l'envie de réussir et au final, aidé de nos collègues et amis, qui rendent nos moments de joie si intenses et qui nous permettent de vivre pleinement notre thèse. Voilà pourquoi, je voudrais remercier tous ceux qui ont croisé mon chemin durant ma thèse. Que nos relations soient maintenant amicales ou parfois plus tendues, sachez que mes remerciements sont sincères car ce travail et ces résultats sont également les vôtres et ma thèse n'aurait pas été la même sans vous. J'espère que mon manuscrit vous fait honneur.

J'ai essayé de regrouper mes remerciements par thématiques en essayant de ne pas être redondant ; ce choix ayant l'inconvénient que certains pourront être surpris de ne pas apparaître dans certains groupes (notamment ceux qui ont eu de nombreuses interactions avec moi).

Un grand merci à Gilles et à Christine, mes deux guides complémentaires dans cette traversée de la thèse. Merci pour leur écoute, leurs explications et surtout leur patience. Merci à Gilles de m'avoir proposé cette thèse, merci pour ses questions tatillonnes qui forcent à aller plus loin et ses connaissances englobant toutes les questions que je me suis posées. Merci à Christine pour son oeil à l'affût et pour son raisonnement cartésien qui m'ont obligé si souvent à consolider mes explications et m'ont forcé à être toujours plus rigoureux. C'est sur ces bases fortes que je peux construire sereinement ma vie de chercheur.

Merci à Christophe et Florence d'avoir eu la gentillesse d'accepter d'être mes rapporteurs. Merci pour leur relecture attentive et leurs encouragements. Merci à Florence pour cette discussion à la sortie d'une conférence qui m'a fait beaucoup réfléchir. Merci à Christophe pour son soutien, ses conseils et pour nos échanges autour du package blockcluster.

Merci à Pascal pour m'avoir dirigé vers Gilles lorsque je lui ai demandé conseil pour une thèse, pour nos discussions régulières tout au long de ces trois années et pour avoir accepté d'être dans mon jury. Cette continuité du début jusqu'à la fin a été importante pour moi.

Merci à Mohamed d'avoir accepté d'être dans mon jury et de m'avoir invité à discuter lors d'un workshop. Ses travaux et sa connaissance du co-clustering m'ont permis d'avancer à partir de bases solides.

Pour les mêmes raisons, merci à tous ceux et celles qui ont préparé le terrain des blocs latents. En particulier, merci à Gérard pour ses travaux, ses codes et pour m'avoir autorisé à utiliser ses programmes matlab. Merci, bien sûr, à Aurore, ma grande soeur du modèle des blocs latents pour avoir répondu à mes nombreuses questions et dont les travaux m'ont fortement aidé durant cette thèse. Merci à Mahendra

pour toutes ces discussions riches autour de la partie théorique du modèle des blocs latents qui m'ont permis d'avancer et m'ont ouvert de nouveaux horizons de recherches. Merci également à Catherine que j'ai si souvent citée sans jamais l'avoir rencontrée dont les travaux théoriques ont également fait avancer mes réflexions. Pour les mêmes raisons, merci à Sylvia dont les résultats sont arrivés au moment de ma réflexion sur une approche bayésienne du modèle et qui m'ont permis d'avancer certainement plus vite. Merci à ceux qui étudient le modèle des blocs latents en général, Julie, Tristan, Trung et bien d'autres qui font vivre nos réflexions.

Merci à nos confrères du modèle des graphes aléatoires avec qui nos interactions sont fortes. Merci en particulier à Antoine pour son amitié et ses nombreuses discussions autour de l'algorithme *LG*. Merci aussi à Alain, Jean-Jacques et Laurent pour leurs travaux théoriques qui nous permettent d'avancer sur les résultats du LBM. Et merci à Jean-Benoist pour ses discussions algorithmiques intéressantes.

Merci à ceux que j'ai croisés avant ma thèse et qui plus ou moins directement m'ont conduit à en faire une. En premier, mes nombreux profs de mathématiques qui m'ont donné le goût des maths, même si je ne peux pas tous vous citer, je garde le visage de chacun de vous en mémoire. Merci à nos professeurs et chargés de TD de probabilités de l'ENS Jean, Nicolas, Marie et Wendelin qui m'ont réconcilié avec les probabilités. Merci aussi à notre professeur de statistique Gérard qui m'a réellement donné le goût pour cette discipline et m'a encouragé notamment en me proposant le stage à Noomiz. Merci à mon tuteur Vincent qui m'a conduit vers Orsay où j'ai trouvé des cours très intéressants et riches. Et un merci particulier à Bertrand qui, durant le stage à Noomiz, m'a dit la phrase qui m'a convaincu de faire une thèse au moment où j'hésitais à plutôt chercher un poste dans le privé. Sans vous, ma vie professionnelle aurait pris une toute autre direction.

Merci à mes grands frères et grandes soeurs de thèse, en particulier Cathy, Jean-Patrick (que j'ai tous les deux beaucoup sollicité pour Capushe), Shuai, Rémy et Caroline pour avoir préparé un terrain fertile dans lequel je n'avais plus qu'à cultiver et laisser grandir tranquillement mes connaissances. Merci pour leurs travaux et leurs conseils qui m'ont aidé jour après jour. Merci également à mes petits frères et petites soeurs de thèse, notamment Mélina, Yann et Valérie qui ont eu la gentillesse de tester mes procédures et m'ont fourni de grands jeux de données et merci à Jana, plus discrète mais avec qui les échanges furent tout de même très intéressants.

Merci à mes collègues de bureau. Merci à ceux qui m'ont accueilli, en particulier Pierre, le grand frère que tout thésard devrait avoir, et Maud qui m'a appris à prendre soin de moi et à me ménager pour que je puisse tenir tout au long de cette thèse. Merci aussi à Jérôme, Nicolas (notamment pour nos discussions totalement hors du commun) et bien sûr l'autruche Bernadette. Merci à ceux qui m'ont accompagné durant cette thèse Huy-Cuong, Petra, Imène, Yi et en particulier Raphaël (qui a eu la malchance de devoir me supporter surtout au moment de la rédaction), Patrick (dont la bonne humeur du sud réchauffait le coeur) et Célia (qui a supporté mes sautes d'humeurs en restant calme). Merci à vous tous pour cette ambiance de travail que je trouve très stimulante.

Merci aux ingénieurs Jean-François et Lydia, Hector et Kaelig pour ces nombreuses soirées jeux vidéos et sorties sportives, Eric et Anne-Laure pour m'avoir supporté quand Jef avait l'idée de m'emmener à la muscu (oui Rémy, ça ne se voit toujours pas) et bien sûr Elodie et son fiancé Quentin, Laura (ainsi que Clara, Sofia et Jérémie), Eléna et Carole pour leur gentillesse, leur joie et leur bonne humeur.

Merci à Yves et Benjamin pour leur écoute et aux responsables du serveur cinaps, Sylvain et Loïc, pour m'avoir permis de lancer un grand nombre (peut-être trop grand parfois) de simulations.

Merci aux membres de l'équipe Select, Yves(s), Jean-Michel, Erwan, Patrick, Lucie, Clément, Jairo, Adrien, Tim, Nelo, Vincent et bien sûr Mohammed (avec qui j'ai beaucoup ri) qui ont contribué à une ambiance de travail prolifique. Merci également aux membres de l'agro-select, notamment Lilianne, Steven et Emilie pour les mêmes raisons.

Merci à Céline et Stéphane de me faire confiance pour la suite en me prenant en post-doc. J'essayerai de m'en montrer le plus digne possible. Merci également à l'équipe de l'Agro de m'accueillir en son sein.

Merci aux membres du groupe Sonata, en particulier à Marie-Laure pour sa bonne humeur communicative mais aussi Sylvie, Brigitte, Nicolas.

Merci aux membres de l'équipe Modal qui n'ont pas fait mentir la réputation chaleureuse des gens du nord. Merci en particulier à Matthieu et Clément pour nos nombreux échanges fort agréables, à Vincent et Parmeet que j'ai dérangés pour le package blockcluster ainsi que Julien et Serge pour nos discussions.

Merci à ceux qui m'ont aidé dans mes enseignements, d'abord durant mes vacations avec Manon, Nalini, Rachid et bien sûr Andrea (dont la bonne humeur est très stimulante) puis durant mon monitorat Michel, Caroline, Than-May, Pierre, Alice et Yves. Un merci particulier à Marie-Anne pour m'avoir guidé, conseillé et encouragé durant ces quatre années d'enseignement. Et merci, bien sûr, à mes étudiants (oui Christine, il m'a fallu trois ans pour ne plus dire "élèves") que je ne peux malheureusement pas tous citer ; merci pour leur sympathie et surtout leurs perles qui ont agrémenté mes corrections, j'espère leur avoir apporté quelque chose et qu'aujourd'hui ils font le métier qu'ils souhaitaient.

Merci à Didier, Xavier, Denis, Nicolas, Pierre, Florence et bien d'autres qui font vivre MATh en JEANS dans une gaieté et une joie qui donnent vraiment envie aux élèves (que je remercie également) de vivre les mathématiques ; j'aurais aimé avoir ce genre de club lorsque j'étais au lycée.

Merci à mes collègues du groupe Jeunes statisticiens de la SFdS avec en premier lieu, le co-organisateur des journées YSP Benjamin qui arrive à me supporter sans m'étrangler. Merci aussi à Séverine, Thomas, Rémi, Christophe, Cyrielle, Alexandre et Robin. Merci aussi à Servane pour sa patience face à nos nombreuses questions d'organisation des journées YSP. Merci aux autres membres de la SFdS avec qui j'ai la joie de collaborer, notamment la présidente Anne et bien sûr Nathalie que j'ai eu si souvent dérangée pour le site de la remontée des thèses.

Merci à tous les doctorants avec qui j'ai pu discuter en particulier à Laure (qui sera la première médaille Fields en stat), Aurélien, Elodie, Emmanuel, Thierry, Olivier, Nina, Viviane, Thomas, Corentin, Alba, Emilien, Paul, Anna, Christèle, Maxime, Pierre-Antoine, Matthias, Caroline, Liviana, Arthur de Paris-Sud pour leur bonne humeur et à ceux qui m'ont supporté en conférence comme Gaëlle (dont le calme est communicatif), Baptiste, Svetlana, Erwan, Mélisande, Cécile (également pour m'avoir invité à parler au séminaire des doctorants du LSTA), Quentin, Emilie(s), Rebecca, Maud, Marie-Liesse, Angelina et bien d'autres dont seuls les visages me reviennent au moment où j'écris ces remerciements (avec toutes mes excuses).

Merci également aux chercheurs de Paris Sud notamment Elisabeth, Edouard, David, Jean-François, Claire, Yohann, Kevin, Odile, Nathalie, Christophe, Raphaël, Camille, Maximie, Sophie, Arvind, Danielle, Konstantin, Christian, Thierry, Bernard et tous les autres pour ce climat convivial de travail. Merci aussi aux autres chercheurs que j'ai croisés et qui ont guidé ma vie dans la recherche, en particulier, Gilles, Sylvain, Francis, Jean-Michel et Judith.

Merci aux personnes qui nous aident au jour le jour. Merci aux trois secrétaires Valérie, Pascale et Katia que j'ai dérangées peut-être plus que nécessaire ; merci à vous de nous guider de façon compétente, patiente et gentille dans la jungle administrative. Merci à Pascal que j'ai si souvent dérangé pour des néons qui clignotaient ou autre problème matériel. Merci à Olivier, Laurent et le service informatique pour leur aide numérique. Et merci aux vigiles de ne pas m'avoir viré lorsque je travaillais tard le soir.

Un merci particulier aux créateurs du package TikZ qui m'a grandement facilité la rédaction de ma thèse.

Merci à mes amis et collègues de l'ENS pour m'avoir aidé à passer ces années et surtout pour ces longues discussions intéressantes sur absolument tous les problèmes : Manon et Matthieu, David et Julia, Olivier, Jérémy, Fathi, Max, Bastien et Yasmine, Daphné et Rémi, Hélène, Pierre, Silvain et Catherine, Julien, Guillaume, Vincent et Marie, Irène, Gabriel, Nicolas, Anisse, Rémi, Benoît, Xavier, Katharina, Jehanne, Mathieu, Matthieu, Jonas, Giancarlo, Titus, Aurélie...

Merci à mes amis du conseil municipal qui m'ont permis de continuer mon mandat par correspondance avec en premier lieu notre ancien maire et maintenant sénateur Jean-Jacques et sa femme Jeanine, au nouveau maire Vincent et sa femme Christine et leurs enfants ainsi que tous les autres membres Sophie, René, Martine, Patrick(s), Claude, Jeanine, Jeanne, Véronique, Laurent, Valéry, Nathalie(s), Pierre, Noëlle, Myriam, Philippe, Dominique, Céline(s), Yves, Alexandre, carole, Christine, Esther et leurs familles respectives ainsi qu'aux membres de notre section. La ville de Montlouis est à mon avis un endroit magnifique pour grandir et je suis heureux d'avoir pu apporter ma petite pierre à l'édifice.

Un grand merci à l'un des piliers cachés de cette thèse : mes amis qui m'ont aidé à ne jamais craquer et dont beaucoup ont déjà été cités dans les paragraphes précédents. Merci donc à Charlène et son ami Valentin pour m'avoir soutenu dans les périodes de doute même si je n'ai pas toujours été agréable, merci à Solène et son ami Alex pour son écoute et surtout son franc-parler qui m'a remis régulièrement sur la bonne route, merci à Sebastien, Charlotte, Mélanie et Hugues pour ces nombreux weekend et ces nombreuses heures de détente (promis, on ne jouera plus jamais au monopoly, je tiens trop à votre amitié), merci à mes amis de collège Maxime et sa femme Margriet, Simon et Eric avec qui nous sommes restés proches malgré l'éloignement ce qui réchauffe le coeur. Merci aussi à Aurélie, son ami Florentin, Emilie, Sylvia et leurs familles qui ont participé en grande partie à ce que je suis devenu. Merci à Coralie pour m'avoir aidé à passer un cap difficile et à reprendre confiance en moi. Merci également à Véronique, Marie-Jo, Annick, Claude, Alexandra et Marie-Aude pour votre amitié. Merci à mes amis plus récents mais pour qui mon amitié est très forte, en particulier Line et Justine pour ces nombreuses soirées à refaire le monde et à se parler franchement ; merci à Emilie, Rémi, Céline, Igor, Solenne, Pierre, Marion, Betula et Nathalie pour ces soirées et weekend de détente qui aident à mieux supporter la semaine ; merci aux témoins avec qui j'ai collaborés pour les mariages, en particulier Coralie, David, Damien, Cindy (et les familles de Sébastion, Charlotte, Hugues et Mélanie) et bien sûr, le groupe des témoins de Saint-Nazaire Elsa, Lou, Manu, Arthur et les familles des mariés (je me souviendrai longtemps du mariage de Max et Margriet grâce à vous). Et merci à ceux avec qui nos chemins se croisent un peu moins mais à qui je pense régulièrement Simon, Emeline, Morgane, Mathilde, Maïlys, Flora, Rima, Angelo, Monica, Christophe, Coralie, Anaïs, Mika, Bilal, Frédérique, Pierre et tant d'autres.

Enfin, merci à mes parents qui ont toujours cru en moi, même quand moi même, je doutais. Merci pour leur amour, leur éducation et de m'avoir permis de mettre toutes mes chances de mon côté sans jamais me mettre la pression. Merci bien sûr à ma môman pour faire semblant de s'intéresser à mon sujet de thèse et me demandant de lui expliquer ce que je faisais tandis que mon papounet souriait derrière en se demandant dans quoi elle s'aventurait. Merci à tous les membres de ma famille encore vivants ou disparus : Angèle,

Jean, Raymonde, André, Clémentine, Soeur Bernard, Jean-Yves, Geneviève, Joseph, Marie-Angèle, Bernard, Geneviève, Marie-Madeleine, Laëtitia, Stéphane, Maxime, Baptiste, Gabriel, Jeanne, Dominique, Amélie, Johan, Anne-Marie, Samuel, Elisa, Isaac, Eve, Yannick, Vanessa, Marie, Emmanuel, Noé, Adrien, Elisabeth, Antoine, Julius, Leïa, Denis, Caroline, Nicolas, Louis, Paul, Nicolas, Maud, Lilou, Anne, Guy, Julia, Pierre, Odile, Bruno, Maxime, Pauline, François, Maurice, Pierrette, Laëtitia, David, Théo, Arthus, Lucie, Emmanuel, Mila, Mahé, Julie, Pascal et tant d'autres. Des milliers de moments agréables me reviennent tandis que j'écris les noms de chacun de vous. Un dicton dit qu'on ne choisit pas sa famille mais, pour ma part, je n'en veux pas une autre, merci pour votre amour.

Et bien sûr, merci à mon écureuil rousse qui a su me soutenir durant cette fin de thèse même si j'ai conscience de ne pas avoir toujours été facile. Merci pour nos fous rire avec Milou, Guizmo, Gros Mickey, Caméléon, Tic et Tac. Merci pour ces soirées inoubliables et merci pour ta tendresse et ton affection. Gros cornet de glace.

Et enfin, merci à toi, lecteur, qui permet à mon travail de perdurer.

L'une de mes plus grandes craintes est d'avoir oublié quelqu'un. Si par hasard, vous êtes dans cette situation, je vous prie d'abord de m'excuser puis de m'envoyer le formulaire [1] suivant :

Je, sous signé, (nom, prénom)..... signale que [a] :

- ☐ mon prénom n'apparaît pas.
- ☐ mon prénom est mal orthographié.
- ☐ mon prénom est placé dans un groupe auquel je n'appartiens pas.
- ☐ autre : .....

Compléments éventuels à la demande : .....
.....
Je vous prie de corriger cette erreur assez rapidement.

*a.* Ne cocher qu'une seule case

Conformément au code de rédaction d'une thèse régissant les remerciements, je ferai mon maximum pour effectuer les corrections dans un délai relativement court.

1. Existant aussi au format numérique à l'adresse suivante :
`https://docs.google.com/forms/d/1liJX__BhnBqtDbNzf_Ah4lOr-3did9MCmOFjREo6c_4/viewform`

# Table des matières


**1 Classification croisée et modèle des blocs latents** **20**
1.1 Introduction . . . 20
1.2 La classification croisée . . . 21
1.2.1 Définition . . . 21
1.2.2 Blocs homogènes . . . 22
1.2.3 Classification croisée par blocs . . . 24
1.2.4 Classification imbriquée . . . 30
1.2.5 Classification avec chevauchement . . . 34
1.2.6 Problèmes ouverts . . . 39
1.2.7 Quand pouvons-nous utiliser la classification croisée ? . . . 42
1.3 Modèle des blocs latents . . . 43
1.3.1 Hypothèses, modèle et notations . . . 43
1.3.2 Résultats théoriques . . . 46
1.3.3 Plan de la thèse . . . 53
1.A Tableau récapitulatif . . . 55
1.B Borne de l'erreur . . . 57

**2 Algorithmes dérivés de l'algorithme *EM*** **62**
2.1 Introduction . . . 62
2.2 Algorithme *Expectation Maximisation* (EM) . . . 63
2.3 Algorithme *VEM* . . . 64
2.4 Algorithme *SEM-Gibbs* . . . 67
2.5 Algorithme *SEM+VEM* . . . 69
2.6 Dégénérescence des classes . . . 70
2.6.1 Simulations . . . 70
2.6.2 Cause de cette dégénérescence . . . 76
2.7 Conclusion . . . 79
2.A États absorbants de type II pour l'algorithme *SEM-Gibbs* . . . 80

**3 Approche bayésienne** **82**
3.1 Introduction . . . 82
3.2 Inférence bayésienne . . . 83
3.2.1 Lois a priori . . . 83
3.3 Algorithme *V-Bayes* . . . 85
3.4 Échantillonneur de *Gibbs* . . . 87
3.5 Choix empirique des paramètres $a$ et $b$ . . . 89
3.6 Critères d'arrêt . . . 89
3.6.1 Statistiques de Brooks-Gelman et améliorations . . . 92

3.6.2 Etude des amélioration apportées à la statistique de Brooks-Gelman . . . . . 95
3.7 Conclusion . . . . . 99

**4 Sélection de modèle** **100**
4.1 Introduction . . . . . 100
4.2 Sélection de modèle . . . . . 101
4.2.1 Critère *Integrated Completed Likelihood* . . . . . 101
4.2.2 Critère *Bayesian Information Criterion* . . . . . 104
4.2.3 Consistance . . . . . 106
4.2.4 Expérimentations numériques . . . . . 107
4.3 Quantifier une perte d'information . . . . . 113
4.3.1 Théorie . . . . . 113
4.3.2 Expérimentations numériques . . . . . 114
4.4 Conclusion . . . . . 116
4.A Critère $ICL_{(a,b)}$ . . . . . 117
4.A.1 Formule explicite . . . . . 117
4.A.2 Comportement théorique . . . . . 118
4.B Démonstration de la partie "quantifier une perte d'informations" . . . . . 119
4.B.1 Densité binaire contre densité ternaire . . . . . 119
4.B.2 Densité contrainte . . . . . 120

**5 Algorithme Largest Gaps** **122**
5.1 Introduction . . . . . 123
5.2 Algorithme *Largest Gaps* (*LG*) . . . . . 123
5.3 Consistance des estimateurs . . . . . 124
5.3.1 Consistance sous la condition que les paramètres soient fixes . . . . . 127
5.3.2 Étude de différentes asymptotiques . . . . . 128
5.4 Estimateurs empiriques de $\boldsymbol{\theta}$ . . . . . 129
5.5 Amélioration de l'estimation des classes . . . . . 130
5.6 Sélection de modèle . . . . . 131
5.7 Applications numériques . . . . . 134
5.7.1 Comportement de la borne 5.2 . . . . . 134
5.7.2 Étude de l'amélioration . . . . . 137
5.7.3 Étude de la sélection de modèles . . . . . 137
5.8 Conclusion . . . . . 138
5.A Indépendance des cases d'une ligne conditionnellement à l'appartenance à une classe de cette dernière . . . . . 138
5.B Démonstration de la consistance de $\widehat{z}^{LG}$ . . . . . 139
5.B.1 Évènement idéal . . . . . 139
5.B.2 Borne sur les probabilités . . . . . 140
5.B.3 Consistance . . . . . 142
5.C Calcul des conditions pour la section 5.3.2 . . . . . 142
5.C.1 Évolution de $\delta_\tau^{n,d}$ . . . . . 142
5.C.2 Evolution de $\pi_{\min}^{n,d}$ . . . . . 143
5.D Démonstration de la consistance des paramètres empiriques . . . . . 144
5.D.1 Paramètres fixes . . . . . 144
5.D.2 Conditions limites . . . . . 146
5.E Algorithme d'initialisation . . . . . 149
5.F Démonstrations sur la sélection de modèles . . . . . 149
5.F.1 Borne de l'estimation de $\widehat{g}^{LG}$ à $n$, $d$ et $S$ fixés . . . . . 149

5.F.2 Consistance des estimateurs du nombre de classes . . . 151
5.F.3 Consistance de l'algorithme $LG$ . . . 151
5.G Simulations . . . 152

**6 Comparaisons des différentes stratégies** **163**
6.1 Introduction . . . 163
6.2 Procédures possibles . . . 164
6.2.1 Algorithmes d'initialisation . . . 164
6.2.2 Algorithmes sensibles aux initialisations . . . 165
6.3 Comparaison . . . 166
6.3.1 Comparaison sur des données simulées . . . 167
6.3.2 Comparaison sur données réelles . . . 168
6.4 Comparaison deux à deux . . . 174
6.4.1 Comparaison entre l'échantillonneur de *Gibbs* et *SEM-Gibbs* . . . 174
6.4.2 Comparaison entre les algorithmes *CEM* et *V-Bayes* . . . 176
6.5 Discussion générale . . . 176

**7 Conclusions, méthodologie et perspectives** **179**
7.1 Conclusion . . . 179
7.2 Vers une méthodologie... . . . 180
7.3 Perspectives . . . 180

**A Fondements** **181**
A.1 Introduction . . . 181
A.2 Modèles de mélange . . . 181
A.2.1 Inférence statistique . . . 181
A.2.2 Modèle de mélange . . . 183
A.3 Algorithme Expectation Maximisation . . . 184
A.3.1 Vraisemblance d'un modèle de mélange . . . 184
A.3.2 Algorithme Expectation Maximisation . . . 184
A.3.3 Algorithme Stochastique Expectation Maximisation . . . 185
A.3.4 Différences de comportements entre les deux algorithmes . . . 186
A.3.5 Application . . . 186

**Bibliographie** **188**

**Index** **195**

# Présentation de la thèse

*"Si cela va sans dire, cela ira encore mieux en le disant."*
Talleyrand

**Estimation et sélection de modèles pour le modèle des blocs latents.** Le travail présenté dans cette thèse se place dans le cadre de la classification croisée. Étant donnée une matrice $\mathbf{x} = (x_{ij})_{n \times d}$ composée de $n$ lignes et $d$ colonnes, le principe est de rechercher des blocs dits homogènes, c'est-à-dire chacun composé d'une structure particulière différente de celles des autres blocs. La façon d'aborder ce problème dépend de l'objectif final : cela va de la recherche de blocs atypiques, ne faisant ressortir qu'un groupes de cases, à la classification des lignes et des colonnes formant les blocs à leurs intersections.

Le modèle des blocs latents, étudié dans ce manuscrit, se place dans cette dernière configuration : l'objectif est d'obtenir une classification des lignes en $g$ groupes (symbolisée par la matrice $\mathbf{z}$) et une classification des colonnes en $m$ groupes ($\mathbf{w}$) de telle sorte qu'une case $x_{ij}$ appartienne au bloc $(k, \ell)$ si et seulement si la ligne $i$ appartient à la classe $k$ et la colonne $j$ appartient à la classe $\ell$. Pour ce modèle, un bloc est dit "homogène" si les cases $x_{ij}$ le formant sont les réalisations de variables aléatoires $X_{ij}$ indépendantes et identiquement distribuées suivant une même loi inconnue conditionnellement à l'appartenance à un bloc ; ces lois appartenant toutes à une même famille paramétrique de densité $\varphi$ et dépendant d'un paramètre $\alpha_{k\ell}$ propre à chaque bloc $(k, \ell)$. Les cases des matrices étudiées prenant leurs valeurs dans l'ensemble $\{1, \ldots, r\}$, notre travail étudie les densités $\varphi$ de loi multinomiales en prenant $\alpha_{k\ell} = \left(\alpha^1_{k\ell}, \ldots, \alpha^r_{k\ell}\right) \in [0,1]^r$ tel que $\sum_{h=1}^r \alpha^h_{k\ell} = 1$. De plus, il est supposé que chaque ligne (resp. chaque colonne) est affectée à une classe $k$ indépendamment des autres lignes et colonnes selon une probabilité $\pi_k$ (resp. $\rho_\ell$). Dans cette thèse, nous avons cherché à estimer le nombre de classes en ligne $g$ et en colonne $m$, l'affectation des lignes et colonnes à chaque classe et l'estimation des paramètres du modèle.

L'étude de la classification pose certains problèmes de fond. Par exemple, il n'existe toujours pas de consensus sur l'unité statistique : devons-nous choisir la case du tableau, les lignes et les colonnes ou simplement le tableau lui-même ? À cette question s'ajoute le problème de l'asymptotique : est-ce lorsque les nombres de lignes et de colonnes tendent vers l'infini ? Le nombre de cases ? Ou le nombre de reproductions du tableau ? De plus, l'évaluation des méthodes diffère suivant les auteurs et les procédures sont parfois difficiles à reproduire. Enfin, il est important de rappeler que la classification croisée ne peut pas s'appliquer à tous les tableaux : de même que le regroupement des lignes doit avoir un sens (il serait étrange de comparer en même temps des sportifs courant un 100 mètres et la qualité de différents jus d'orange par exemple), celui des colonnes doit également être cohérent (une discussion de ce problème est proposée en section 1.2.7).

**Classification croisée et modèle des blocs latents.** Le modèle des blocs latents appartient à la famille des modèles de mélange ; à ce titre se posent les problèmes d'identifiabilité et de *label switching*. En s'inspirant des résultats obtenus par Celisse et al. (2012) dans le cadre du modèle des graphes aléatoires (ou *SBM* pour *Stochastic Block Model*) où les lignes et les colonnes représentent les mêmes entités et dont la même classification des lignes et des colonnes est demandée, nous avons montré l'existence de conditions suffisantes pour obtenir l'identifiabilité du modèle et pouvoir contourner le *label switching* (voir la section 1.3.2.1).

Le second problème rencontré est de ne pas pouvoir calculer, en un temps raisonnable, la vraisemblance du modèle. En effet, les hypothèses posées pour ce modèle (voir section 1.3.1) permettent d'obtenir la densité suivante :

$$p(\mathbf{x};\boldsymbol{\theta}) = \sum_{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W}} p(\mathbf{z};\boldsymbol{\theta})p(\mathbf{w};\boldsymbol{\theta})p(\mathbf{x}|\mathbf{z},\mathbf{w};\boldsymbol{\theta}) \ ,$$

où $\mathcal{Z}$ (resp. $\mathcal{W}$) représente l'ensemble de toutes les partitions possibles $\mathbf{z}$ des lignes (resp. $\mathbf{w}$ des colonnes) et $\boldsymbol{\theta}$ est la notation résumée de tous les paramètres du modèle (voir section 1.3.2.2) ; la somme ne pouvant pas être factorisée comme dans le cas de mélanges classiques. Ce problème empêche par là même d'évaluer la qualité des estimateurs obtenus.

Pour le comportement asymptotique des estimateurs, Mariadassou et Matias (2012) montrent, à partir de résultats obtenus par Celisse et al. (2012) dans le cadre du SBM, que si $\widehat{\boldsymbol{\theta}_{n,d}}$ est un estimateur de $\boldsymbol{\theta}^\star$ tel que $\widehat{\boldsymbol{\alpha}_{n,d}}$ converge en probabilité vers le vrai paramètre $\boldsymbol{\alpha}^\star$ alors nous avons :

$$\mathbb{P}\left(Z = \mathbf{z}^\star, W = \mathbf{w}^\star \,\middle|\, x; \widehat{\boldsymbol{\theta}_{n,d}}\right) \underset{n,d}{\longrightarrow} 1$$

où $\mathbf{z}^\star$ et $\mathbf{w}^\star$ représentent les vraies partitions. En revanche, il n'existe toujours pas de résultats sur le comportement de l'estimateur du maximum de vraisemblance.

**Algorithmes dérivés de l'algorithme *EM*.** Pour estimer les paramètres et les classes du modèle, la première intuition est d'utiliser l'algorithme *EM* (*Expectation Maximisation* de Dempster et al. (1977)) ; toutefois, la dépendance conditionnelle des variables latentes $\mathbf{z}$ et $\mathbf{w}$ connaissant les observations rend le calcul de l'étape d'estimation impossible dans un temps raisonnable. Deux algorithmes sont proposés pour contourner cette difficulté : l'algorithme *VEM* (*Variational Expectation Maximisation*) de Govaert et Nadif (2008) utilisant une approximation variationnelle maximisant une fonction $\mathcal{F}$ (appelée *énergie libre*) minorant la log-vraisemblance et l'algorithme *SEM-Gibbs* (*Stochastic Expectation Maximisation*) de Keribin et al. (2014) simulant une chaîne de Markov pour approximer stochastiquement les paramètres à estimer.

En étudiant l'algorithme *SEM-Gibbs*, nous avons mis en évidence un nouveau type d'états absorbants pour la chaîne de Markov engendrée par l'algorithme propre à la configuration en blocs du modèle ainsi que les configurations empêchant l'ergodicité de la chaîne obtenue (voir la section 2.4). Pour contourner ces états, il suffit d'empêcher d'avoir des estimations de $\alpha_{k\ell}^h$ valant 1 durant la simulation.

Nous avons ensuite étudié la combinaison de l'algorithme *SEM-Gibbs* et de l'algorithme *VEM*. En particulier, nous avons observé empiriquement que cette combinaison est beaucoup moins sensible aux initialisations que l'algorithme *VEM* et renvoie des estimations plus proches des valeurs ayant servi aux simulations que l'algorithme *SEM-Gibbs*. Toutefois, il apparaît que cette combinaison estime parfois un nombre de classes inférieur à celui demandé (nous parlons de *dégénérescence des classes*). En étudiant ce phénomène, nous avons vu qu'il se produisait soit durant la simulation de l'algorithme *VEM* où les estimations d'un des paramètres $\pi_k$ tendent vers 0, soit au moment de l'affectation des labels où aucune ligne n'est affectée à l'une des classes (de même pour les colonnes). Nous avons montré empiriquement que si l'algorithme renvoie moins de classes, cela ne signifie pas forcément qu'il y en ait réellement moins que demandé.

**Approche bayésienne.** Pour pallier ce problème de dégénérescence des classes, nous avons étudié une approche bayésienne du modèle. Pour cela, nous avons pris les lois a priori conjuguées suivantes pour les proportions :

$$\boldsymbol{\pi} \sim \mathcal{D}ir(a,\ldots,a) \quad \text{et} \quad \boldsymbol{\rho} \sim \mathcal{D}ir(a,\ldots,a)$$

et supposé que les lois a priori sur les $\alpha_{k\ell}$ étaient indépendantes :

$$\forall (k,\ell) \in \{1,\ldots,g\}\times\{1,\ldots,m\}, \quad \alpha_{k\ell} \sim \mathcal{D}ir(b,\ldots,b).$$

En reprenant les calculs de l'algorithme *VEM*, nous avons proposé une version bayésienne baptisée *V-Bayes* avec des estimations des $\pi_k$ et $\rho_\ell$ ne tendant pas vers 0 si le paramètre $a$ est plus grand que 1.

Nous avons également étudié l'échantillonneur de *Gibbs* et avons montré que les états qui sont absorbants pour l'algorithme *SEM-Gibbs* ne le sont pas pour lui. Nous avons proposé un critère d'arrêt basé sur la statistique de Brooks-Gelman (Brooks et Gelman, 1998) et l'amélioration proposée par Fu (2012) ; sur des simulations, notre amélioration permet généralement d'obtenir plus vite des résultats légèrement meilleurs que la statistique de Brooks-Gelman basique.

De la même manière que pour les algorithmes *SEM-Gibbs* et *VEM*, nous avons regardé la combinaison de l'échantillonneur de *Gibbs* et de l'algorithme *V-Bayes*. Sur des données simulées, nous avons remarqué que la combinaison bayésienne renvoie moins souvent de classes vides que la version fréquentiste lorsque $a$ est plus grand que 1. En revanche, prendre $b$ plus grand que 1 accentue le phénomène de dégénérescence. À partir de ces simulations, nous préconisons de prendre $a = 4$ et $b = 1$.

**Sélection de modèle.** Nous avons également regardé le problème du choix des nombres $g$ de classes en ligne et $m$ de classes en colonne. La log-vraisemblance du modèle étant impossible à calculer, les critères de sélection de modèles la pénalisant (comme *AIC* ou *BIC*) ne peuvent pas être utilisés. Nous avons donc choisi d'étudier le critère *ICL* (pour *Integrated Completed Likelihood*) initialement développé dans le cas de la classification des modèles de mélanges simples (Biernacki et al., 2000). Ce critère, d'inspiration bayésienne, nécessitant des lois a priori, nous avons repris celles utilisées précédemment. Il dépend donc des paramètres $a$ et $b$ et cherche le modèle maximisant :

$$ICL_{(a,b)}(g, m) = \log p\left(\mathbf{x}, \widehat{\mathbf{z}}, \widehat{\mathbf{w}}\right)$$

où le couple $(\widehat{\mathbf{z}}, \widehat{\mathbf{w}})$ dépend des nombres de classes $g$ et $m$ et maximisent l'énergie libre bayésienne $\mathcal{F}_B$.

En étudiant l'approximation asymptotique de ce critère, nous avons proposé un critère *BIC* (pour *Bayesian Information Criterion*) de la forme suivante :

$$BIC(g, m) = \log p\left(\mathbf{x}; \widehat{\theta}^{EMV}, g, m\right) - \frac{gm(r-1) + g - 1}{2} \log n - \frac{gm(r-1) + m - 1}{2} \log d.$$

où $\widehat{\theta}^{EMV}$ est l'estimateur du maximum de vraisemblance ; ce critère peut être vu comme une log-vraisemblance pénalisée par rapport aux lignes et aux colonnes. Comme le maximum de la vraisemblance ne peut pas être calculé, nous proposons de le remplacer par le maximum de l'énergie libre $\mathcal{F}$.

En reprenant les résultats de Mariadassou et Matias (2012) sur la convergence des labels, nous avons conjecturé que le critère *BIC* est asymptotiquement identique au critère $ICL_{(a,b)}$ ce qui impliquerait la consistance de l'estimation des nombres de classes $g$ et $m$ par ces deux critères. Ces conjectures ont pu être montrées empiriquement sur des simulations.

Nous avons également abordé le problème du choix de modèle lorsque nous avons un modèle sur une matrice de données qualitatives et un modèle pour la même matrice mais en ayant regroupé des occurrences ; nous proposons deux critères adaptés des précédents pour répondre à cette question permettant d'évaluer la perte d'information obtenue par cette simplification (voir section 4.3).

**Algorithme Largest Gaps.** Durant cette thèse, nous avons également adapté l'algorithme *Largest Gaps* (ou *LG*) proposé par Channarond et al. (2012) pour le cas du *SBM* binaire. Si la matrice vérifie les conditions suffisantes d'identifiabilité que nous avons obtenues et si nous faisons pour chaque ligne, la somme de toutes les cases, nous obtenons un mélange de lois binomiales de paramètres $d$ et $\tau_k$ (la moyenne des cases valant 1 d'une ligne de la classe $k$) dont les classes apparaissent de façon d'autant plus discriminante que $d$ est grand et que les écarts entre les $\tau_k$ sont importants.

La principale difficulté que nous avons rencontrée fut le problème de la double asymptotique. Toutefois, en adaptant les démonstrations, nous avons montré que l'algorithme *LG* renvoie des estimateurs des labels

et des paramètres consistants (à une permutation près) si nous avons les conditions suivantes :

$$\log n = o(d) \text{ et } \log d = o(n).$$

De plus, nous avons proposé une amélioration pour pouvoir également estimer les nombres de classes en ligne $g$ et en colonne $m$ de façon consistante. Toutefois, les simulations ne montrent de bons résultats que lorsque les nombres de lignes et de colonnes sont énormes.

Nous avons également proposé une généralisation de l'algorithme pour le cas où les données ne sont pas binaires.

**Comparaisons des différentes stratégies.** Enfin, nous avons fait une comparaison entre les différentes combinaisons possibles des algorithmes étudiés pour un temps d'exécution (elapsed) fixé. Ce choix a été fait pour ne pas pénaliser les algorithmes rapides sensibles aux initialisations qui peuvent être ainsi relancés plus souvent en comparaison des algorithmes plus lents à converger mais moins dépendant des valeurs initiales. Ces procédures ont été testées sur des données simulées et des données réelles.

Nous avons pu observer que l'échantillonneur de *Gibbs* donne de bons résultats lorsqu'il était utilisé comme initialisation pour des données simulées. En revanche, pour des données réelles, il semblerait préférable d'utiliser des initialisations aléatoires. Pour les deux cas, l'algorithme *V-Bayes* paraît être celui qui affine le mieux les résultats.

Nous préconisons la combinaison de l'échantillonneur de *Gibbs* avec l'algorithme *V-Bayes* avec un bon choix des paramètres.

# Notations

*"Une langue maternelle : cela n'existe pas. Nous naissons dans une langue inconnue. Le reste est une lente traduction."*

Ulises Drago, personnage de "La traduction" de Pablo De Santis (2004)

Les notations dans les articles sont comme une langue étrangère : courantes pour ceux qui les utilisent depuis longtemps mais incompréhensibles pour les non initiés. Nous proposons donc un traducteur notations/français pour permettre au lecteur de mieux trouver son chemin.
De manière générale, les variables sont notées en majuscule et leurs observations en minuscule ; si $a$ est une observation de la variable $A$, nous notons $p(a)$ pour $\mathbb{P}(A = a)$.

| **Variables, observations et paramètres** | | page |
|---|---|---|
| $\alpha_{k\ell}$ | paramètres de la loi du bloc $(k, \ell)$. Dans le cas de données binaires, $\alpha_{k\ell} \in [0, 1]$ et dans le cas de données catégorielles, $\alpha_{k\ell}$ représente l'ensemble des paramètres $\left(\alpha^1_{k\ell}, \ldots, \alpha^r_{k\ell}\right)$. | 45 |
| $\alpha^h_{k\ell}$ | sous-paramètre de $\alpha_{k\ell}$ dans le cas de données catégorielles. | 45 |
| $\boldsymbol{\alpha}$ | matrice regroupant les $g \times m$ groupes de paramètres $\alpha_{k\ell}$. | 45 |
| $\boldsymbol{\alpha}^\star$ | notation pour le vrai paramètre $\boldsymbol{\alpha}$ du modèle. | 50 |
| $\delta_\tau$ | distance minimale entre deux valeurs de coordonnées distinctes du $g$-uplet $\boldsymbol{\tau}$. | 127 |
| $\delta_\sigma$ | distance minimale entre deux valeurs de coordonnées distinctes du $m$-uplet $\boldsymbol{\sigma}$. | 127 |
| $\delta^\xi_{\mathfrak{t}}$ | différence entre les quantiles empiriques de niveau 97.5% et 2.5% de la chaîne $\xi_{\mathfrak{t}}$ dans le calcul de la statistique de Brooks-Gelman. | 92 |
| $\Delta$ | différence entre les quantiles empiriques de niveau 97.5% et 2.5% de l'agglomération des chaînes $\xi_{\mathfrak{t}}$ dans le calcul de la statistique de Brooks-Gelman. | 92 |
| $\mathcal{F}$ | fonction des distributions libres des variables latentes appelée énergie libre. | 64 |
| $\boldsymbol{\theta}$ | vecteur de tous les paramètres du modèle. | 29, 46 |
| $\boldsymbol{\theta}^\star$ | notation englobant tous les vrais paramètres du modèle. | 50 |
| $\xi$ | notation représentant une composante d'un paramètre ($\pi_k$, $\rho_\ell$ ou $\alpha_{k\ell}$). | 92 |
| $\pi_k$ | probabilité a priori pour une ligne d'appartenir à la classe $k$. | 46 |
| $\boldsymbol{\pi}$ | notation regroupant les $g$ probabilités $\pi_k$. | 46 |
| $\boldsymbol{\pi}^\star$ | notation pour le vrai paramètre $\boldsymbol{\pi}$ du modèle. | 50 |
| $\pi_{\min}$ | probabilité minimale d'appartenir à une classe en ligne. | 127 |
| $q_{\mathbf{zw}}$ | distribution libre du couple des variables latentes $(\mathbf{z}, \mathbf{w})$ . | 64 |
| $\rho_\ell$ | probabilité a priori pour une colonne d'appartenir à la classe $\ell$. | 46 |
| $\boldsymbol{\rho}$ | notation regroupant les $m$ probabilités $\rho_\ell$. | 46 |
| $\boldsymbol{\rho}^\star$ | notation pour le vrai paramètre $\boldsymbol{\rho}$ du modèle. | 50 |
| $\rho_{\min}$ | probabilité minimale d'appartenir à une classe en colonne. | 127 |

| | | page |
|---|---|---|
| $\sigma_\ell$ | probabilité d'une case d'une colonne de la classe $\ell$ de valoir 1 dans le cas binaire indépendamment des classes en ligne. | 48 |
| $\boldsymbol{\sigma}$ | notation regroupant les $m$ probabilités $\sigma_\ell$. | 47 |
| $\tau_k$ | probabilité d'une case d'une ligne de la classe $k$ de valoir 1 dans le cas binaire indépendamment des classes en colonne. | 48 |
| $\boldsymbol{\tau}$ | notation regroupant les $g$ probabilités $\tau_k$. | 47 |
| $v_{ijh}$ | matrice binaire représentant les cases de la matrice $x_{ij}$ prenant la valeur $h$. | 45 |
| $\mathbf{w}$ | matrice représentant les classes de l'ensemble $J$ (colonnes). Indifféremment, $w_j = \kappa$ et $w_{j\kappa} = 1$ signifient que la colonne $j$ appartient à la classe $\kappa$ (la première n'est possible que si nous avons une partition des colonnes comme pour le modèle des blocs latents). | 29 |
| $\mathbf{w}^\star$ | vraie partition de l'ensemble $J$ (colonnes) pour le modèle des blocs latents. | 51 |
| $w_{j\ell}$ | case de la matrice binaire $\mathbf{w}$ indiquant l'appartenance de la colonne $j$ à la classe $\ell$. | 43 |
| $\widehat{w}^{LG}$ | estimateur de la partition en colonne renvoyé par l'algorithme $LG$. | 124 |
| $\mathbf{x}$ | matrice des données. | 21 |
| $\mathbf{x}^B$ | matrice de données binarisées (par rapport à une matrice ternaire $\mathbf{x}^T$). | 113 |
| $\mathbf{x}^T$ | matrice de données ternaires. | 113 |
| $x_{ij}$ | observation de la matrice $\mathbf{x}$. Dans cette thèse, les valeurs appartiennent à l'ensemble $\{1, \ldots, r\}$ (sauf pour le cas binaire où nous prenons l'ensemble $\{0, 1\}$). | 21 |
| $X_{ij}$ | variable aléatoire dont la case $x_{ij}$ est une observation. | 22 |
| $\overline{X_{i\bullet}}$ | moyenne de la $i^{\text{ème}}$ ligne utilisée dans l'algorithme $LG$. | 124 |
| $\mathbf{z}$ | matrice représentant les classes de l'ensemble $I$ (lignes). Indifféremment, $z_i = \kappa$ et $z_{i\kappa} = 1$ signifient que la ligne $i$ appartient à la classe $\kappa$ (la première n'est possible que si nous avons une partition des lignes comme pour le modèle des blocs latents). | 29 |
| $\mathbf{z}^\star$ | vraie partition de l'ensemble $I$ (lignes) pour le modèle des blocs latents. | 51 |
| $z_{ik}$ | case de la matrice binaire $\mathbf{z}$ indiquant l'appartenance de la ligne $i$ à la classe $k$. | 43 |
| $\widehat{z}^{LG}$ | estimateur de la partition en ligne renvoyé par l'algorithme $LG$. | 124 |

**Indices**

| | | page |
|---|---|---|
| $c$ | indice des itérations des étapes des algorithmes (allant de 1 à $niter$ dans le cas de l'algorithme *SEM-Gibbs*). | 86 |
| $d$ | nombre de colonnes. | 21 |
| $g$ | nombre de classes en ligne. | 24, 43 |
| $h$ | indice allant de 1 à $r$ des valeurs prises par les variables $X_{ij}$ dans le cas de données catégorielles. | 45 |
| $i$ | indice des lignes allant de 1 à $n$, élément de l'ensemble $I$. | 21, 46 |
| $j$ | indice des colonnes allant de 1 à $d$, élément de l'ensemble $J$. | 21, 46 |
| $k$ | indice des classes en ligne allant de 1 à $g$. | 26, 46 |
| $\kappa$ | indice des blocs allant de 1 à $u$. | 21 |
| $\ell$ | indice des classes en colonne allant de 1 à $m$. | 26, 46 |
| $m$ | nombre de classes en colonne. | 24, 43 |
| $n$ | nombre de lignes. | 21 |
| $\mathfrak{N}$ | nombre d'itérations avant d'évaluer la statistique de Brooks-Gelman. | 92 |

| | | page |
|---|---|---|
| $N^h_{k\ell;\mathbf{z},\mathbf{w}}$ | nombre de cases valant $h$ dans le bloc $(k,\ell)$. | 88 |
| $niter$ | nombre d'itérations faites par l'algorithme *SEM-Gibbs*. | 67 |
| $niter_{int}$ | nombre d'itérations faites par l'algorithme *SEM-Gibbs* dans l'étape $SE$. | 67 |
| $r$ | cardinal de l'ensemble des valeurs pouvant être prises par les variables $X_{ij}$. Nous assimilons cet ensemble à $\{1,\ldots,r\}$ sauf dans le cas où $r=2$ où l'ensemble est $\{0,1\}$. | 45 |
| $t$ | indice des itérations à l'intérieur de l'étape $E$ pour certains algorithmes (allant de 1 à $niter_{int}$ dans le cas de l'algorithme *SEM-Gibbs*). | 65 |
| $\mathfrak{t}$ | indice des chaînes lancées en parallèle pour la statistique de Brooks-Gelman allant de 1 à $\mathcal{T}$. | 92 |
| $\mathcal{T}$ | nombre de chaînes lancées en parallèle pour utiliser la statistique de Brooks-Gelman. | 92 |
| $u$ | nombre de blocs (pour le modèle des blocs latents, $u = g \times m$). | 21 |

**Ensembles et partitions**

| | | page |
|---|---|---|
| $I$ | ensemble des individus, souvent assimilé à $\{1,\ldots,n\}$. | 21 |
| $J$ | ensemble des variables, souvent assimilé à $\{1,\ldots,d\}$. | 21 |
| $U$ | ensemble représentant $u$ sous-ensembles $U_\kappa$, appelés blocs, de l'ensemble $I \times J$. | 21 |
| $U_\kappa$ | sous-ensemble, appelé bloc, des indices des cases de la matrice $\mathbf{x}$ de la forme $U_\kappa = (Z_{\kappa_1}, W_{\kappa_2})$. Par abus, nous notons $x_{ij} \in U_\kappa$ pour $i \in Z_{\kappa_1}$ et $j \in W_{\kappa_2}$. | 21 |
| $W$ | ensemble représentant la partition en $m$ classes des colonnes $J$ (sauf pour le chapitre 1 où il représente $u$ sous-groupes). | 21, 43 |
| $W_\kappa$ | sous-ensemble de $J$. | 21 |
| $W_\ell$ | notation particulière de $W_\kappa$ pour le modèle des blocs latents. | 26, 43 |
| $\mathcal{W}$ | ensemble de toutes les partitions $\mathbf{w}$ possibles de $J$. | 29 |
| $Z$ | ensemble représentant la partition en $g$ classes des lignes $I$ (sauf pour le chapitre 1 où il représente $u$ sous-groupes). | 21, 43 |
| $Z_\kappa$ | sous-ensemble de $I$. | 21 |
| $Z_k$ | notation particulière de $Z_\kappa$ pour le modèle des blocs latents. | 26, 43 |
| $\mathcal{Z}$ | ensemble de toutes les partitions $\mathbf{z}$ possibles de $I$. | 28 |

**Notations générales**

| | | page |
|---|---|---|
| $\equiv_{\mathcal{Z}}$ | relation d'équivalence signifiant que deux matrices binaires appartenant à $\mathcal{Z}$ sont égales à une permutation près des indices des classes. | 52, 124 |
| $\equiv_{\mathcal{W}}$ | relation d'équivalence signifiant que deux matrices binaires appartenant à $\mathcal{W}$ sont égales à une permutation près des indices des classes. | 127 |
| $1^B$ | notation regroupant plusieurs entités (utilisée pour la binarisation de données ternaires). | 113 |
| $\mathrm{D_{KL}}$ | divergence de Kullback-Leiber. | 64 |
| $\boldsymbol{\theta}^\star$ | notation pour le vrai paramètre $\boldsymbol{\theta}$. | 50 |
| $\boldsymbol{\theta}^{(c)}$ | notation pour l'estimation du paramètre $\boldsymbol{\theta}$ à l'étape $c$ d'un algorithme. | 184 |

| | | page |
|---|---|---|
| $\widehat{\theta}^{C}$ | notation pour l'estimation du paramètre $\boldsymbol{\theta}$ par l'algorithme *CEM*. | 166 |
| $\widehat{\theta}^{CB}$ | notation pour l'estimation du paramètre $\boldsymbol{\theta}$ par l'algorithme *CEM-Bayes*. | 166 |
| $\widehat{\theta}^{G}$ | notation pour l'estimation du paramètre $\boldsymbol{\theta}$ par l'échantillonneur de *Gibbs*. | 88 |
| $\widehat{\theta}^{LG}$ | notation pour l'estimation du paramètre $\boldsymbol{\theta}$ par l'algorithme *LG*. | 129 |
| $\widehat{\theta}^{SEM}$ | notation pour l'estimation du paramètre $\boldsymbol{\theta}$ par l'algorithme *SEM-Gibbs*. | 67 |
| $\widehat{\theta}^{SV}$ | notation pour l'estimation du paramètre $\boldsymbol{\theta}$ par l'algorithme *SEM+VEM*. | 69 |
| $\widehat{\theta}^{VEM}$ | notation pour l'estimation du paramètre $\boldsymbol{\theta}$ par l'algorithme *VEM*. | 65 |
| $\underline{\lim}$ | limite inférieure. La limite inférieure d'une suite $(v_n)_{n\in\mathbb{N}}$ est définie par $\lim_{k\to+\infty} \inf\{u_n \vert n \geq k\}$ | 127 |
| $L$ | notation pour la fonction de log-vraisemblance du modèle en fonction de $\boldsymbol{\theta}$. | 63 |
| $(x_{ij})_{n\times d}$ | notation pour une matrice $\mathbf{x}$ ayant $i$ pour les indices en ligne allant de 1 à $n$ et $j$ pour les indices en colonne allant de 1 à $d$. | 21 |
| $\overline{\mathbf{x}}_{U_\kappa}$ | valeur moyenne du bloc $U_\kappa$. | 35 |
| $\overline{\mathbf{x}}_{k\ell}$ | valeur moyenne du bloc $(k,\ell)$ pour le modèle des blocs latents. | 27 |
| $\sum_i$ | abréviation de $\sum_{i=1}^{n}$. Pour simplifier les notations, nous enlevons les indices lorsqu'ils sont usuels. | 46 |
| $w_{j\bullet}$ | notation représentant la $j$ ème ligne de la matrice binaire $\mathbf{w}$. | 38 |
| $w_{+\ell}$ | notation représentant la somme des cases de la $\ell$ ème colonne de la matrice binaire $\mathbf{w}$. Cela représente le cardinal du $\ell$ ème groupe en colonne. | 43 |
| $\vert W_\kappa \vert$ | notation représentant le cardinal de $W_\kappa$. | 35 |
| $z_{i\bullet}$ | notation représentant la $i$ ème ligne de la matrice binaire $\mathbf{z}$. | 38 |
| $z_{+k}$ | notation représentant la somme des cases de la $k$ ème colonne de la matrice binaire $\mathbf{z}$. Cela représente le cardinal du $k$ ème groupe en ligne. | 43 |
| $\vert Z_\kappa \vert$ | notation représentant le cardinal de $Z_\kappa$. | 35 |
| $\varphi$ | densité des lois des composantes des modèles de mélange. | 45 |

**Acronymes**

| | | page |
|---|---|---|
| BIC | Bayesian Information Criterion (critère). | 100 |
| CEM | Algorithme *EM* ajoutant une étape de classification (ou Classification Expectation Maximisation). | 165 |
| EM | Expectation Maximisation (algorithme). | 63 |
| EMV | Estimateur du Maximum de Vraisemblance. | 63 |
| ICL | Integrated Classification Likelihood (critère). | 100 |
| LBM | Modèle des Blocs Latents (ou Latent Bloc Model). | 43 |
| LG | Largest Gaps (algorithme). | 123 |
| MCMC | Monte Carlo Markov Chain. | 41, 83 |
| SBM | Modèle de graphes aléatoires (ou Stochastic Bloc Model). | 46 |
| SEM | Algorithme *EM* stochastique (ou Stochastic Expectation Maximisation). | 62 |
| VEM | Algorithme *EM* variationnel (ou Variationnal Expectation Maximisation). | 65 |

# Chapitre 1

# Classification croisée et modèle des blocs latents

*"Il était une fois..."*
Conte de Perrault

**Sommaire**


**1.1 Introduction . . . . . 20**
**1.2 La classification croisée . . . . . 21**
1.2.1 Définition . . . . . 21
1.2.2 Blocs homogènes . . . . . 22
1.2.3 Classification croisée par blocs . . . . . 24
1.2.4 Classification imbriquée . . . . . 30
1.2.5 Classification avec chevauchement . . . . . 34
1.2.6 Problèmes ouverts . . . . . 39
1.2.7 Quand pouvons-nous utiliser la classification croisée ? . . . . . 42
**1.3 Modèle des blocs latents . . . . . 43**
1.3.1 Hypothèses, modèle et notations . . . . . 43
1.3.2 Résultats théoriques . . . . . 46
1.3.3 Plan de la thèse . . . . . 53
**1.A Tableau récapitulatif . . . . . 55**
**1.B Borne de l'erreur . . . . . 57**


## 1.1 Introduction

Poids, taille, forme, sexe, constitution, âge, conditions climatiques, catégorie sociale, composition moléculaire, valeur financière, mode de reproduction, opinion politique, localisation géographique... Il existe de nombreuses caractéristiques pouvant servir à la classification. Celles-ci sont utilisées, par exemple, en médecine où pour une même maladie, des médicaments différents sont prescrits à un homme adulte, un enfant, une personne âgée ou une femme enceinte (Papillon et al., 2001). Toutefois, les mêmes caractéristiques ne peuvent pas servir dans tous les cas ; si le sexe peut aider dans le diagnostic d'une douleur abdominale, il est moins important, voire inutile en comparaison du niveau d'étude pour estimer les capacités à exercer un emploi ou une fonction (Anker, 1997; d'Intignano et al., 1999). De plus, certaines

caractéristiques sont plus facilement exploitables que d'autres : par exemple, des critères biologiques permettent de faire une distinction entre homme et femme mais la distinction entre adulte et enfant est plus complexe (Deschavanne et Tavoillot, 2007).

Le principe de la classification est de former des groupes d'individus possédant des similitudes (mêmes habitudes de consommation, mêmes réactions à la prise d'un médicament...) différentes de celles des autres groupes. Présente dans de nombreux domaines, la classification est notamment utilisée en biologie en groupant, par exemple, des papillons suivant les tailles de leurs ailes et de leur corps (Celeux et Robert, 1993) ou bien encore en génomie en groupant des gènes réagissant de la même façon par rapport à différentes conditions de stress (Maugis et al., 2009).

Dès 1965 (Good, 1965), une extension de la classification a été proposée dans le but de réaliser une classification jointe des lignes et des colonnes d'un tableau. L'objectif de cette analyse est l'identification d'une structure sous-jacente existant entre ces deux ensembles.

Depuis, la classification croisée a été développée sous plusieurs variantes et son intérêt s'est considérablement accru ces dernières années avec l'arrivée de nombreuses applications comme l'analyse de données textuelles (Dhillon et al., 2003), la génomie (Hedenfalk et al., 2001; Jagalur et al., 2007), les systèmes de recommandation (Shan et Banerjee, 2008) ou bien encore en sociologie où des données politiques ont été étudiées par différents auteurs pour retrouver des groupes de politiciens et des enjeux politiques (Hartigan, 1975; Wyse et Friel, 2010). En 2006, elle a d'ailleurs été l'objet d'un challenge, Netflix, dont l'un des objectifs était de réussir à classer simultanément les utilisateurs de cette société de location de dvd et les films (Bennett et Lanning, 2007).

Dans cette introduction, nous exposons d'abord différents courants existants pour faire cette classification croisée ; cette partie reprend essentiellement l'article fait en collaboration avec Lomet (Brault et Lomet, 2014). Nous nous concentrons ensuite sur le modèle des blocs latents qui est celui étudié au cours de cette thèse. Nous introduisons les notations, les résultats déjà obtenus ainsi que les questions et problèmes encore en suspens ; cette partie reprend les idées de l'article fait en collaboration avec Mariadassou (Brault et Mariadassou, 2014). Nous terminons par le plan de la thèse.

## 1.2 La classification croisée

Dans cette partie, nous commençons par donner une définition générale de la classification croisée et ce que nous entendons par "blocs homogènes" puis nous étudions trois grands points de vue différant notamment par leurs hypothèses de classification allant des plus contraintes aux plus souples : la classification par blocs organisant l'ensemble des lignes et des colonnes en blocs homogènes, la classification imbriquée et celle avec chevauchement.

### 1.2.1 Définition

La classification croisée a pour objectif d'identifier dans un tableau de données une structure en blocs d'éléments homogènes et dissemblables des éléments des autres blocs.

La matrice de données $\mathbf{x} = (x_{ij})_{n\times d}$ est définie par l'ensemble $I$ des lignes des $n$ observations et l'ensemble $J$ des colonnes des $d$ variables. Chaque élément de la matrice $x_{ij}$ représente la relation binaire, discrète ou continue entre l'observation $i$ et la variable $j$. Avec ces notations, l'objectif de la classification croisée est de définir un ensemble $U$ de $u$ blocs $U_\kappa = (Z_{\kappa_1}, W_{\kappa_2})$ où $Z = \{Z_1, ..., Z_u\}$ est un ensemble de sous-groupes éventuellement répétés de lignes de $I$ et $W = \{W_1, ..., W_u\}$ un ensemble de sous-groupes éventuellement répétés de colonnes de $J$ pouvant être le produit cartésien d'une partition $Z$ de $I$ et d'une partition $W$ de $J$, le produit cartésien d'une hiérarchie $Z$ de $I$ et d'une hiérarchie $W$ de $J$, une partition de $I \times J$, une hiérarchie de $I \times J$ ou simplement un sous-groupe de $I \times J$ (*biclustering*). Chaque bloc

$U_\kappa$ est de forme rectangulaire si la matrice est réorganisée suivant les classes $Z_{\kappa_1}$ et $W_{\kappa_2}$. Par ailleurs, le nombre total $u$ de blocs recherchés peut être supposé connu ou non suivant les méthodes.

Les données peuvent être de tout type. Les cas les plus fréquemment étudiés sont : le cas binaire (absence ou présence de relation entre les lignes et les colonnes), le cas catégoriel (étudié dans cette thèse), continu et de contingence. Comme expliqué dans la suite, le type de donnée influence le choix du critère à optimiser.

Il existe un parallèle entre un tableau $\mathbf{x}$ et un graphe bipartie (voir la figure 1.1). Dans le cas binaire, une case $x_{ij}$ (en haut à gauche de la figure) représente la présence ou l'absence d'une relation entre la ligne $i$ et la colonne $j$ (en bas à gauche de la figure). Une réorganisation des lignes et des colonnes permet à notre oeil peu habitué de mieux voir la structure sous jacente (à droite sur la figure).

### 1.2.2 Blocs homogènes

Avant d'énumérer les différentes stratégies, nous rappelons les différentes structures se cachant derrière le terme "blocs homogènes". Un bloc est dit homogène s'il est composé d'une structure particulière différente de celles des autres blocs. Dans la plupart des modèles, en particulier pour ceux de la classification croisée par blocs (section 1.2.3) et de la classification imbriquée (section 1.2.4), la structure recherchée est une valeur constante dans tout le bloc. Toutefois, et notamment pour le cas de la classification avec chevauchement (section 1.2.5) et la recherche d'un bloc atypique, la structure peut être plus compliquée. D'un point de vue déterministe et en complétant le tableau proposé par Madeira et Oliveira (2004) (voir la figure 1.2), nous pouvons recenser sept types de blocs homogènes :

(a) un bloc composé de la même valeur pour toutes les cases :
si $x_{ij} \in U_\kappa$, alors $x_{ij} = \alpha^{(\kappa)}$ ;

(b) et (c) un bloc composé de la même valeur pour chaque ligne (resp. chaque colonne) :
si $x_{ij} \in U_\kappa$, alors $x_{ij} = \alpha_i^{(\kappa)}$ ou $x_{ij} = \beta_j^{(\kappa)}$ ;

(d) un bloc obtenu par une formule additive sur les lignes et les colonnes :
si $x_{ij} \in U_\kappa$, alors $x_{ij} = \alpha_i^{(\kappa)} + \beta_j^{(\kappa)}$ ;

(e) un bloc obtenu par une formule multiplicative sur les lignes et les colonnes :
si $x_{ij} \in U_\kappa$, alors $x_{ij} = \alpha_i^{(\kappa)} \times \beta_j^{(\kappa)}$ ;

(f) et (g) un bloc composé d'une évolution cohérente sur les lignes et/ou les colonnes :
si $x_{ij} \in U_\kappa$ et $x_{i'j} \in U_\kappa$, alors $i \leq i' \Rightarrow x_{ij} \leq x_{i'j}$ (cas sur les lignes).

En généralisant, nous supposons que pour chaque bloc $U_\kappa$, il existe une fonction $f_\kappa$ des lignes $i$ et des colonnes $j$ telle que :

$$\text{si } x_{ij} \in U_\kappa, \text{ alors } x_{ij} = f_\kappa(i, j).$$

Toutefois, supposer que, pour des données réelles, les blocs sont générés exactement sous l'une de ces formes déterministes crée des problèmes d'estimation (Ben-dor et al., 2002). Pour contourner cette difficulté, le point de vue probabiliste suppose que les cases $x_{ij}$ sont le résultat de variables aléatoires $X_{ij}$ dont la loi dépend du bloc $U_\kappa$ :

$$\text{si } X_{ij} \in U_\kappa, \text{ alors } X_{ij} \sim \mathcal{L}_\kappa$$

c'est-à-dire que ce sont des variables aléatoires de même loi pour tout un bloc $U_\kappa$ ; celle-ci appartenant généralement à une même famille pour tous les blocs. Les quatre types de lois les plus utilisés sont :

— les lois gaussiennes pour la modélisation de données continues (Lazzeroni et Owen, 2000; Govaert et Nadif, 2009),
— les lois de Bernoulli pour les données binaires (Tanay et al., 2002; Govaert et Nadif, 2008),
— les lois de Poisson pour les données de contingence (Govaert et Nadif, 2007; Aubert et al., 2014),
— les lois multinomiales pour les données catégorielles (Keribin et al., 2014).

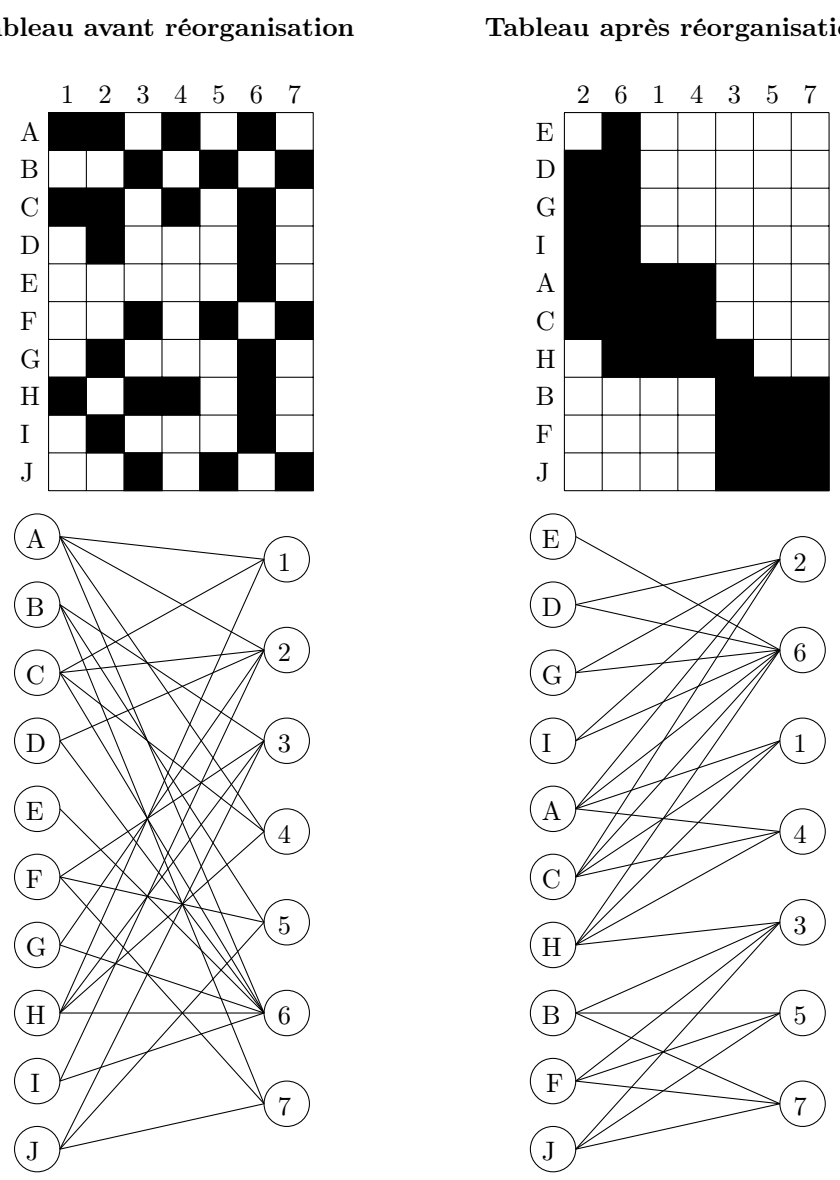



FIGURE 1.1 – En haut à gauche se trouve un tableau quelconque puis en haut à droite, une réorganisation mettant en évidence une structure sous-jacente. En bas se trouvent les graphes biparties associés à chaque organisation du tableau. Cet exemple est tiré de l'article de Govaert et Nadif (2008).

| | | | |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |
| 1 | 1 | 1 | 1 |

(a) Valeurs constantes pour le bloc

| | | | |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 2 |
| 4 | 4 | 4 | 4 |
| 3 | 3 | 3 | 3 |

(b) Valeurs constantes pour chaque ligne

| | | | |
|---|---|---|---|
| 1 | 2 | 4 | 3 |
| 1 | 2 | 4 | 3 |
| 1 | 2 | 4 | 3 |
| 1 | 2 | 4 | 3 |

(c) Valeurs constantes pour chaque colonne

| | | | |
|---|---|---|---|
| 1 | 2 | 4 | 3 |
| 3 | 4 | 6 | 5 |
| 2 | 3 | 5 | 4 |
| 0 | 1 | 3 | 2 |

(d) Evolution cohérente sur les lignes et les colonnes (modèle additif)

| | | | |
|---|---|---|---|
| 1 | 2 | 10 | 30 |
| 3 | 6 | 30 | 90 |
| 12 | 24 | 120 | 360 |
| 2 | 4 | 20 | 60 |

(e) Evolution cohérente sur les lignes et les colonnes (modèle multiplicatif)

| | | | |
|---|---|---|---|
| 1 | 2 | 4 | 6 |
| 2 | 3 | 4 | 5 |
| 1 | 6 | 6 | 7 |
| 2 | 3 | 7 | 9 |

(f) Evolution cohérente sur les colonnes (relation d'ordre)

| | | | |
|---|---|---|---|
| 1 | 0 | 2 | 3 |
| 2 | 0 | 4 | 5 |
| 4 | 5 | 6 | 7 |
| 5 | 6 | 7 | 9 |

(g) Evolution cohérente sur les lignes (relation d'ordre)

FIGURE 1.2 – Eventail non exhaustif de différentes structures de blocs considérés comme homogènes.

### 1.2.3 Classification croisée par blocs

La classification croisée par blocs considère une classification simultanée des lignes et des colonnes utilisant deux partitions (*coclustering*). En reprenant le tableau de la figure 1.1, nous pouvons partitionner les lignes et les colonnes pour faire ressortir une structure de blocs presque totalement noirs ou presque totalement blancs (voir la figure 1.3, à gauche). Le tableau peut ainsi être résumé (figure 1.3, au milieu) ou simplifié (figure 1.3, à droite). Pour chaque tableau, un graphe bipartie est associé (figure 1.3, en bas).

#### 1.2.3.1 Définition

L'objectif de la classification croisée par blocs est d'identifier la partition en $g$ sous-groupes homogènes en ligne et en $m$ sous-groupes homogènes en colonne comme représentée en figures 1.3 et 1.4. Ce type de classification croisée peut s'appliquer aux données binaires, catégorielles, continues ou de contingence se présentant sous la forme d'un tableau à double entrée comme les données de recommandation ou de génomie, par exemple.

En reprenant les notations précédentes, la détection de blocs homogènes de la matrice $\mathbf{x}$ peut être obtenue en partitionnant les lignes en $g$ classes et les colonnes en $m$ classes. L'objectif est alors de trouver une partition $Z = \{Z_1, ..., Z_g\}$ de $I$ et $W = \{W_1, ..., W_m\}$ de $J$ telles que l'ensemble $U = \{U_1, ..., U_u\}$

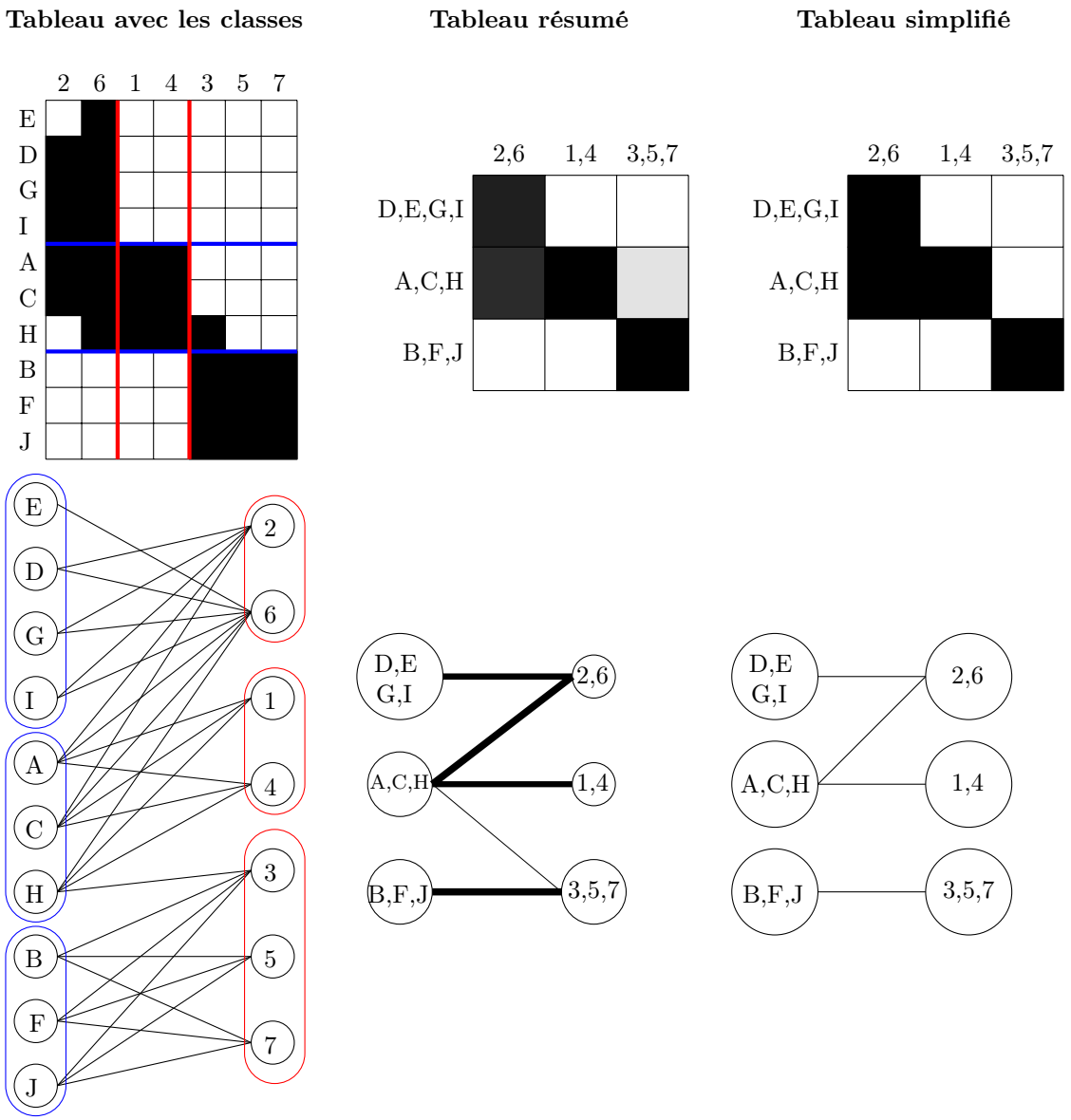



FIGURE 1.3 – En haut à gauche se trouve un tableau avec des classes en ligne et en colonne ; au milieu un tableau résumé où les cases sont grisées suivant la proportion de cases noires et à droite le tableau simplifié. En bas à gauche se trouve le graphe bipartie où les classes sont représentées par des ovales ; au milieu le graphe résumé avec des ronds plus ou moins grands suivant le cardinal de chaque groupe et des arêtes plus ou moins épaisses suivant s'il y a une forte proportion d'arêtes reliant les deux groupes ; à droite le graphe simplifié.

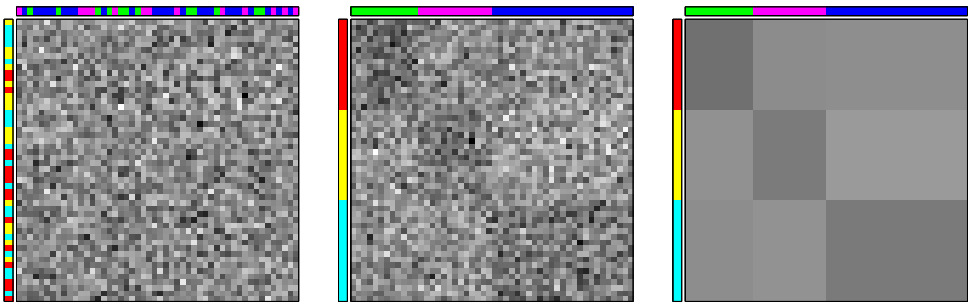


FIGURE 1.4 – Représentation d'un tableau de données continues en niveaux de gris : à gauche le tableau original, au centre le tableau organisé suivant les partitions en ligne et en colonne, à droite le tableau résumé des blocs.

obtenu à partir de leur produit cartésien (c'est-à-dire l'ensemble des $u = g \times m$ blocs $U_\kappa = (Z_k, W_\ell)$ couplant chaque élément de $Z$ et de $W$) soit alors une partition en forme de quadrillage de la matrice $\mathbf{x}$ avec des blocs contrastés entre eux.

Pour réaliser une telle classification croisée, certains auteurs (Lerman et Leredde, 1977) proposent une méthode dont l'algorithme est appliqué séparément et indépendamment sur les deux ensembles mais avec une analyse simultanée des résultats. D'autres (Tishby et al., 1999) réalisent une classification sur un ensemble puis utilisent cette classification pour effectuer celle sur le second. Enfin, plusieurs auteurs (Govaert, 1983; Kluger et al., 2003; Yoo et Choi, 2010; Seldin et Tishby, 2010) réalisent une classification simultanée des lignes et des colonnes via différentes méthodes exposées en sections 1.2.3.2 et 1.2.3.3.

La classification croisée par blocs est une méthode d'apprentissage non supervisée, nous distinguons deux grandes familles de méthodes mathématiques :

- la reconstruction de matrices (*reconstruction based*), où le problème est formalisé par une approximation de matrice et des mesures de dissimilarité, utilisant différents ensembles de contraintes (Govaert, 1977, 1995; Banerjee et al., 2007),
- des modèles probabilistes (*model-based*), dont les modèles de mélange et leurs variantes qui utilisent des variables latentes pour définir les classes en ligne et en colonne (Govaert et Nadif, 2003; Shan et Banerjee, 2008; Wyse et Friel, 2010).

#### 1.2.3.2 Méthodes par reconstruction de matrices

Les méthodes par reconstruction de matrices se basent sur le choix de mesures de distance ou plus généralement sur le choix de mesure de distorsion entre les données. Nous distinguons ainsi quatre courants :

- les méthodes « CRO »(Govaert, 1983),
- la factorisation de matrice non-négative (Seung et Lee, 2001),
- la décomposition en blocs à valeurs non-négatives (Long et al., 2005),
- la tri-factorisation orthogonale de matrice non-négative (Yoo et Choi, 2010).

##### 1.2.3.2.1 Méthodes « CRO »

Proposées par Govaert (1983), les méthodes « CRO » comprennent trois algorithmes *Croeuc, Crobin* et *Croki2* qui diffèrent par le type de données observées (quantitatives, binaires ou de contingence). L'objectif de ces algorithmes est de déterminer le couple de partitions de $I$ et $J$ qui optimise un critère mesurant la qualité d'un couple de partitions et dépendant du type de données.

Dans le cas des données quantitatives, le critère à minimiser par l'algorithme *Croeuc* est celui des moindres carrés. En reprenant les notations précédentes, ce critère s'écrit :

$$H(Z,W) = \sum_{i=1}^{n}\sum_{j=1}^{d}\sum_{k=1}^{g}\sum_{\ell=1}^{m} z_{ik}w_{j\ell}(x_{ij} - \overline{\mathbf{x}}_{k\ell})^2 \ ,$$

où $n, d, g, m$ sont respectivement le nombre de lignes, de colonnes, de classes en ligne et de classes en colonne et $\overline{\mathbf{x}}_{k\ell}$ la moyenne empirique du bloc $(k,\ell)$.

Dans le cas d'un tableau de données binaires, chaque bloc est associé à la valeur 0 ou 1. Le critère à minimiser par l'algorithme *Crobin* est alors la mesure entre le tableau initial et celui rempli par ces blocs de 0 et 1 (figure 1.3, à droite). Dans le cas des données de contingence, l'algorithme *Croki2* cherche à minimiser la perte d'information résultant du regroupement, ce qui est équivalent à déterminer les partitions maximisant le $\chi^2$ de contingence du tableau. Cela revient à maximiser la dépendance entre la partition des lignes et la partition des colonnes.

Ces algorithmes itératifs définissent une suite de couples de partitions à partir d'un couple initial. Pour ce faire, nous fixons une des partitions et nous cherchons la meilleure partition de l'autre ensemble. Puis, cette dernière étant fixée, nous cherchons alors la meilleure partition du premier ensemble. Ces étapes sont réalisées jusqu'à convergence des partitions vers un optimum local. Il est nécessaire de lancer ces algorithmes avec suffisamment de partitions initiales différentes afin d'éviter les optima locaux.

Se basant sur ces critères, Bock (1979) a proposé des algorithmes similaires pour des données quantitatives et Dhillon et al. (2003) pour des données de contingence. D'autre types d'algorithmes ont aussi été proposés comme l'algorithme génétique de Hansohm (2002) ou l'algorithme séquentiel de Podani et Feoli (1991) qui cherche à maximiser une différence de distances du $\chi^2$ et qui est appliqué à des données binaires en écologie.

#### 1.2.3.2.2 Factorisation de matrices non-négative (NMF)

La factorisation de matrices non-négatives (Seung et Lee, 2001) est utilisée dans les domaines de la reconnaissance de formes et du *text mining* basique. Les données sont le plus souvent des données de comptage (nombre de mots dans différents textes par exemple) et des données binaires (absence ou présence).

L'objectif de cette méthode est de factoriser la matrice des données $\mathbf{x}$, supposée non-négative (c'est-à-dire que pour tout couple $(i,j)$, $x_{ij} \geq 0$), en deux autres matrices non-négatives (arbitraires) et de minimiser l'erreur entre la matrice de départ et le produit des matrices obtenues :

$$\min_{\mathbf{A}\in\mathbb{R}_+^{n\times\eta},\, \mathbf{B}\in\mathbb{R}_+^{d\times\eta}} \left|\left|\mathbf{x} - \mathbf{A}\mathbf{B}^{\intercal}\right|\right|^2 \ ,$$

où $\mathbf{A} \in \mathbb{R}_+^{n\times\eta}$, $\mathbf{B} \in \mathbb{R}_+^{d\times\eta}$ sont deux matrices arbitraires non-négatives avec $\eta$ supposé plus petit que $n$ et $d$. Pour ce faire, nous optimisons la distance entre $\mathbf{x}$ et les matrices non-négatives. La convergence de cet algorithme est assurée, et l'unicité de la solution peut être acquise par normalisation d'un des vecteurs colonnes de l'une des matrices obtenues.

Une méthode similaire, la tri-factorisation non-négative (NTF), vise à factoriser la matrice initiale en trois matrices non-négatives dont une arbitraire $\mathbf{A} = (A_{k\ell})_{g\times m}$ qui peut être vue comme une matrice simplifiée et les deux autres représentant les appartenances aux classes en ligne $\mathbf{z} = (z_{ik})_{n\times g}$ et en colonne $\mathbf{w} = (w_{k\ell})_{d\times m}$. Le produit de ces trois matrices peut être vu comme un résumé de la matrice initiale $\mathbf{x}$. Comme précédemment, l'objectif de cette méthode est de minimiser l'écart entre la matrice initiale et le produit de ces trois matrices :

$$\min_{\mathbf{z}\in\mathbb{R}_+^{n\times g},\, \mathbf{A}\in\mathbb{R}_+^{g\times m},\, \mathbf{w}\in\mathbb{R}_+^{d\times m}} \left|\left|\mathbf{x} - \mathbf{z}\mathbf{A}\mathbf{w}^{\intercal}\right|\right|^2 \ ,$$

Cette minimisation est réalisée par un algorithme itératif alternant des procédures d'optimisation des moindres-carrés décrites ci-après.

#### 1.2.3.2.3 Décomposition en blocs à valeurs non-négatives (NBVD)

La décomposition en blocs à valeurs non-négatives (*non-negative block value decomposition*) a été proposée par Long et al. (2005). Elle repose sur la tri-factorisation non-négative. L'objectif est, comme précédemment, de minimiser l'erreur entre la matrice de départ et les trois matrices obtenues. Cet objectif correspond à une fonction convexe en chacune des trois matrices mais qui n'est pas convexe en les considérant ensemble. Cette fonction est optimisée en mettant à jour de manière itérative la décomposition à l'aide d'un ensemble de règles multiplicatives et atteint un optimum local.

La tri-factorisation de la matrice initiale n'est pas unique. Elle ne permet donc pas de fournir directement une classification croisée des classes : une normalisation du produit des matrices est notamment nécessaire afin d'obtenir les classes. Cette méthode, testée empiriquement sur quelques jeux de données, montre une amélioration de la précision de la classification par rapport aux méthodes classiques de NMF.

#### 1.2.3.2.4 Tri-factorisation orthogonale de matrice non-négative (ONMF)

Comme la méthode précédente, la tri-factorisation orthogonale de matrice non-négative factorise la matrice des données en trois matrices dont, cette fois, deux d'entre elles sont orthogonales. Dans leurs travaux, Yoo et Choi (2010) étudient les apports de cette contrainte d'orthogonalité pour obtenir une interprétation rigoureuse de la classification. Cette méthode n'est intéressante que lorsque la factorisation de matrice non-négative n'est pas réalisable. Par ailleurs, pour cette méthode comme pour les autres présentées dans cette section, la principale difficulté consiste à choisir a priori une mesure de distance appropriée : ce choix arbitraire influence les résultats.

### 1.2.3.3 Méthodes par modèle probabiliste

L'autre famille est basée sur l'utilisation de modèles probabilistes nécessitant des hypothèses portant sur l'origine des données. Ces hypothèses permettent la conception d'un modèle statistique où les paramètres sont alors estimés sur la base des données fournies.

En classification simple, une des approches les plus classiques et performantes est l'utilisation des modèles de mélange (voir McLachlan, 1982; Celeux et Govaert, 1992; Banfield et Raftery, 1993, et l'annexe A). Les données sont supposées être générées par un mélange de distributions de probabilité dont chaque composante $k$ représente une classe. Les variables aléatoires d'une classe suivent la même loi de densité $\varphi(\cdot;\alpha_k)$ : les composantes appartiennent à la même famille paramétrique dont le paramètre $\alpha_k$ dépend de la classe $k$. La vraisemblance des observations s'écrit alors :

$$p(\mathbf{x};\boldsymbol{\theta}) = \prod_{i=1}^{n}\left(\sum_{k=1}^{g}\pi_k\varphi(x_i;\alpha_k)\right)$$

où $\pi_k$ représente la probabilité pour une observation d'appartenir à la classe $k$ et $\boldsymbol{\theta}$ le vecteur des paramètres. En développant, nous obtenons l'écriture suivante :

$$p(\mathbf{x};\boldsymbol{\theta}) = \sum_{\mathbf{z}\in\mathcal{Z}} p(\mathbf{z};\boldsymbol{\theta})p(\mathbf{x}|\mathbf{z};\boldsymbol{\theta}) \tag{1.1}$$

où $\mathcal{Z}$ représente l'ensemble de toutes les partitions possibles $\mathbf{z}$ de $I$. La matrice $\mathbf{x} = (\mathbf{x}_1, ..., \mathbf{x}_n)$ des données est supposée composée de lignes $\mathbf{x}_i$ provenant d'un échantillon indépendant et identiquement distribué. L'estimation des paramètres et, par conséquent, celle des classes sont souvent réalisées par l'algorithme *EM* (*Expectation-Maximisation* (Dempster et al., 1977)).

Pour la classification croisée, plusieurs modèles probabilistes dont certains s'inspirant de la classification simple et des modèles de mélange ont été proposés pour représenter les structures en blocs homogènes des données.

#### 1.2.3.3.1 Modèle des blocs latents

Inspirés par les modèles de mélange pour la classification simple, Govaert et Nadif (2003) proposent le modèle des blocs latents qui est une extension du modèle de mélange classique utilisant deux variables latentes. Cette méthode considère que (i) les partitions en ligne $Z$ et colonne $W$ sont des variables latentes à estimer, (ii) chaque élément du tableau est issu d'une distribution de probabilité conditionnée par son appartenance à sa classe en ligne et en colonne et (iii) les appartenances aux classes sont supposées a priori indépendantes et conditionnellement à la connaissance de ces appartenances, les éléments du tableau sont supposés indépendants (Govaert et Nadif, 2013). La densité de ce modèle s'écrit :

$$p(\mathbf{x};\boldsymbol{\theta}) = \sum_{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W}} p(\mathbf{z};\boldsymbol{\theta})p(\mathbf{w};\boldsymbol{\theta})p(\mathbf{x}|\mathbf{z},\mathbf{w};\boldsymbol{\theta}) \ ,$$

où $\mathcal{Z}$ et $\mathcal{W}$ représentent les ensembles de toutes les partitions possibles $\mathbf{z}$ de $I$ et $\mathbf{w}$ de $J$ et $\boldsymbol{\theta}$ est la notation résumée de tous les paramètres du modèle (voir section 1.3.2.2). Nous retrouvons ainsi la forme de l'équation (1.1). Ce modèle peut se décliner selon les quatre types rappelées en section 1.2.2 en fonction du type de données. La probabilité conditionnelle $p(\mathbf{x}|\mathbf{z},\mathbf{w};\boldsymbol{\theta})$ est alors une gaussienne pour la modélisation de données continues (Govaert et Nadif, 2009), de Bernoulli pour les données binaires (Govaert et Nadif, 2008), de Poisson pour les données de contingence (Govaert et Nadif, 2007) et multinomiale pour les données catégorielles (Keribin et al., 2014). Une adaptation est en cours pour les données ordinales à partir du modèle génératif de Biernacki et Jacques (2012).

Le modèle des blocs latents étant celui étudié dans la suite, nous en expliquons à la fin de ce chapitre les difficultés du calcul de la vraisemblance (section 1.3.2.1) et dans le chapitre 2 celles de l'utilisation de l'algorithme $EM$ (section 2.2). Pour pallier ces problèmes, Govaert et Nadif (2008) proposent un algorithme d'estimation à partir de l'algorithme $EM$ en incluant une approximation variationnelle : les probabilités conditionnelles des partitions sont supposées indépendantes. Dans chaque cas du modèle des blocs latents (gaussien, de Bernoulli, de Poisson ou multinomial), un algorithme *VEM* (*Variational Expectation Maximization*) a été développé pour estimer les paramètres (cet algorithme est présenté en section 2.3).

Le modèle des blocs latents et l'algorithme VEM ont été, par ailleurs, utilisés par Lashkari et Golland (2009) comme modèle génératif et par Jagalur et al. (2007) pour étudier des données d'expression de gènes.

Plusieurs approches bayésiennes basées sur ce modèle ont été récemment développées. Shan et Banerjee (2008) et Van Dijk et al. (2009) proposent une approche bayésienne pour estimer les paramètres en supposant que le nombre de classes est une variable. Le premier utilise une approximation variationnelle alors que le second emploie un algorithme basé sur l'échantillonneur de Gibbs plus coûteux mais moins sensible aux initialisations. Meeds et Roweis (2007) montrent quant à eux que ces modèles bayésiens prennent facilement en compte les données manquantes et sont robustes pour des forts taux de données manquantes.

Enfin, Deodhar et Ghosh (2007) étendent le modèle pour une analyse croisée des clients et des produits en marketing en incluant la prise en compte d'attributs (localisation géographique, prix...) sur les lignes et les colonnes. Shafiei et Milios (2006) proposent quant à eux un modèle probabiliste autorisant le chevauchement des blocs pour les distributions régulières de la famille exponentielle.

#### 1.2.3.3.2 Autres modèles probabilistes

Rooth (1995) propose un modèle probabiliste pour la classification par blocs de données de contingence

avec une structure en diagonale et définit un algorithme utilisant des formules similaires aux formules d'estimation de Baum-Welch pour les chaînes de Markov cachées.

Hartigan (2000) étudie la classification croisée des votes aux congrès des États-Unis. Les sénateurs sont partitionnés en blocs et les mesures législatives sont partitionnées en types. Pour ce faire, il propose un modèle logistique pour réaliser une classification croisée de données binaires : la probabilité de chaque bloc et type de la partition finale est estimée par une méthode de Monte-Carlo par chaînes de Markov.

Nowicki et Snijders (2001) proposent un autre modèle différant par son application et par le fait que $I = J$. Ce modèle de graphes aléatoires (*Stochastic Block Model*) construit un modèle de mélange pour les relations entre les objets et identifie les classes latentes via une inférence a posteriori. Ce modèle pour graphe suppose que les sommets sont répartis entre plusieurs classes (latentes) et que la distribution de probabilité de la relation en deux sommets ne dépend que de la classe à laquelle ils appartiennent. Ces auteurs illustrent ce modèle par un exemple tiré des réseaux sociaux (*Kapferer's tailor shop*). Une revue bibliographique est proposée par Matias et Robin (2014).

Kemp et al. (2006) présentent un modèle infini relationnel qui découvre des structures stochastiques de données relationnelles sous la forme d'observations binaires. Ce modèle non paramétrique bayésien est appliqué à quatre types de problèmes : classer les objets et leurs caractéristiques, découvrir des systèmes de parenté, découvrir la structure de données politiques et pour étudier des ontologies.

### 1.2.4 Classification imbriquée

La condition que les blocs de $U$ soient le résultat du produit cartésien d'une partition de $I$ et d'une partition de $J$ peut être trop contraignante. Sur la figure 1.5 est représentée une matrice réorganisée sous les conditions des modèles précédents. Les blocs de la cinquième classe en colonne sont distingués par les classes en ligne alors qu'ils pourraient être regroupés en une seule colonne commune à toutes les lignes. Le modèle serait ainsi plus parcimonieux. Dans le cas de Netflix, cela signifie que les films de cette classe plaisent de la même façon à tout le monde tandis que les appréciations d'autres films vont dépendre des utilisateurs.

Par ailleurs, il est parfois intéressant de ne chercher que les blocs les plus contrastés en ne classant pas les autres cases. Par exemple, Ben-dor et al. (2002) ont fait ressortir un groupe de gènes ayant des comportements similaires face à une mutation de la tumeur du cancer du sein au milieu d'autres gènes n'ayant pas de comportements particuliers.

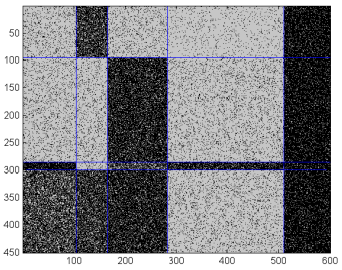



FIGURE 1.5 – Représentation d'une matrice réorganisée sous la condition de la classification croisée (section 1.2.3).

Dans cette perspective, la classification imbriquée (appelée aussi hiérarchique) autorise pour deux blocs $U_\kappa = (Z_\kappa, W_\kappa)$ et $U_{\kappa'} = (Z_{\kappa'}, W_{\kappa'})$ une hiérarchie possible dans les groupes de classification :

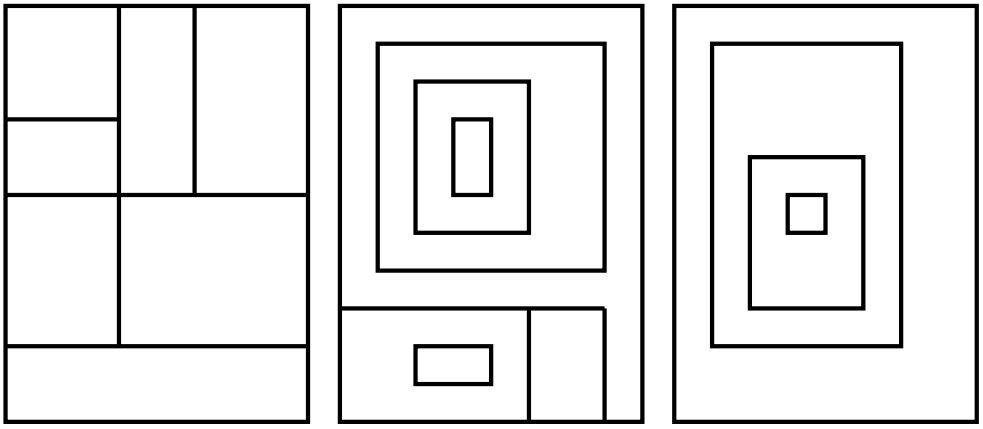
— soit sur les lignes et/ou les colonnes (figure 1.6 (a)) :

$$\begin{aligned} Z_\kappa \cap Z_{\kappa'} \neq \emptyset \quad &\Rightarrow \quad Z_\kappa \subseteq Z_{\kappa'} \text{ ou } Z_{\kappa'} \subseteq Z_\kappa \\ \text{et } W_\kappa \cap W_{\kappa'} \neq \emptyset \quad &\Rightarrow \quad W_\kappa \subseteq W_{\kappa'} \text{ ou } W_{\kappa'} \subseteq W_\kappa, \end{aligned}$$

— soit sur les blocs entièrement (figure 1.6 (b) et (c)) :

$$U_\kappa \cap U_{\kappa'} \neq \emptyset \quad \Rightarrow \quad U_\kappa \subseteq U_{\kappa'} \text{ ou } U_{\kappa'} \subseteq U_\kappa.$$

La différence avec la section 1.2.3 étant que l'inclusion stricte est désormais autorisée. Dans cette partie, nous abordons le cas de la classification imbriquée sur les lignes et/ou les colonnes (figure 1.6 (a)) puis la classification imbriquée sur les blocs (figures 1.6 (b) et (c)).



(a) (b) (c)

FIGURE 1.6 – Représentations schématiques de trois classifications imbriquées : (a) est une classification imbriquée sur les lignes et les colonnes, (b) est une classification imbriquée non restreinte sur les blocs et (c) une classification restreinte sur les blocs.

En reprenant la matrice réordonnée de la figure 1.1, nous pouvons faire une succession de coupures (voir la partie supérieure de la figure 1.7) : les deux premières séparent le bloc inférieur droit de la partie supérieure gauche (sur la figure 1.7, en haut à gauche), une troisième coupure verticale permet de regrouper les cases noires situées à gauche du bloc supérieur (sur la figure 1.7, en haut au milieu) et enfin, une dernière coupure permet au bloc du milieu de la partie supérieure de séparer les cases blanches des cases noires (sur la figure 1.7, en haut à droite). Suivant les critères, il est possible de continuer les subdivisions pour obtenir des sous-blocs.

Une fois la classification obtenue, nous pouvons étudier les sous-graphes biparties correspondant (partie basse de la figure 1.7). Nous pouvons voir que certains sont composés de peu d'arêtes alors que d'autres sont très denses.

#### 1.2.4.1 Classification imbriquée sur les lignes et les colonnes

La première méthode consiste à faire des découpes successives horizontalement ou verticalement pour séparer certains blocs en deux parties et est appelée *Direct Splitting* (voir Hartigan, 1975, Chapitre 14).

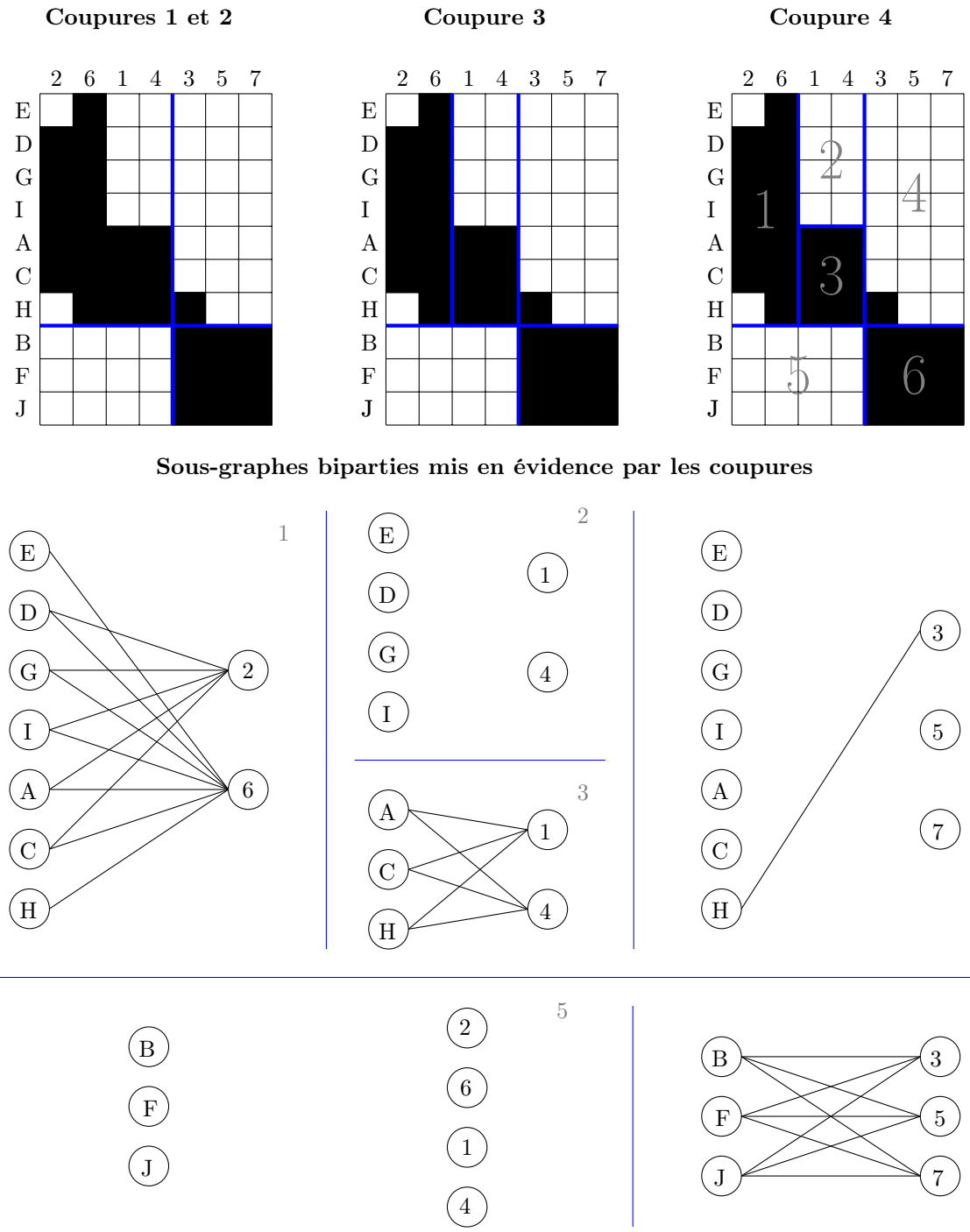



FIGURE 1.7 – En haut est schématisée l'évolution des coupures successives faites par la classification imbriquée. Une fois les six blocs obtenus par les quatre coupures (sur le tableau en haut à droite), nous pouvons étudier chaque sous-graphe associé (figure du bas).

Toutes les données du tableau sont ainsi classées.

Pour trouver une telle classification, des algorithmes déterministes sont proposés. Hartigan (1975) introduit ceux de *fractionnement direct* (*Splitting Algorithm*) cherchant à séparer les blocs tout en minimisant la variance de chacun des deux nouveaux blocs obtenus. L'algorithme de *fractionnement à un sens* (*One-Way Splitting Algorithm*) ne regarde que l'erreur suivant les lignes ou les colonnes et l'algorithme de *fractionnement à deux sens* (*Two-Way Splitting Algorithm*) minimise l'erreur sur les deux. Ces algorithmes ont été testés sur le pourcentage de votes pour le candidat républicain aux présidentielles dans 6 états du sud des Etats-Unis pour 6 années différentes (tableau 1.2). Nous voyons que trois états (KY, MD et MO) ont été regroupés puis que les trois autres se différencient sauf sur certaines années. Duffy et Quiroz (1991) proposent une amélioration de ces algorithmes en autorisant les permutations des lignes et des colonnes.

| | 32 | 36 | 40 | 60 | 64 | 68 |
|---|---|---|---|---|---|---|
| MS | 4 | 3 | 4 | 25 | 87 | 14 |
| SC | 2 | 1 | 4 | 49 | 59 | 14 |
| LA | 7 | 11 | 14 | 29 | 57 | 23 |
| KY | 40 | 40 | 42 | 54 | 36 | 44 |
| MD | 36 | 37 | 41 | 46 | 35 | 42 |
| MO | 35 | 38 | 48 | 50 | 36 | 45 |

TABLE 1.2 – Classification du pourcentage de votants pour le candidat républicain aux présidentielles pour 6 années dans 6 états du sud des États-Unis : Mississippi (MS), Caroline du Sud (SC), Louisiane (LA), Kentucky (KY), Maryland (MD) et Missouri (MO).

D'un point de vue probabiliste, Roy et Teh (2008) proposent un modèle appelé *processus Mondrian* basé sur une généralisation du processus de Dirichlet (Robert, 2006) utilisé dans le cadre des mélanges simples. Il cherche alors à estimer les découpes par des algorithmes MCMC et plus particulière de Metropolis-Hasting. Wang et al. (2011) expliquent que ce modèle permet de diminuer le nombre de coupures en comparaison de celui obtenu pour le modèle des blocs latents. Roy et Teh (2008) ont testé leurs algorithmes sur les échanges commerciaux et diplomatiques entre 24 pays. Ils ont pu retrouver que les pays échangent essentiellement avec ceux qui sont séparés par une frontière ou un océan.

#### 1.2.4.2 Classification imbriquée par blocs

Le regroupement de données imbriquées par blocs ne prend plus en compte ni les lignes, ni les colonnes dans leur globalité. Sur la figure 1.8 est représenté un tableau avec deux regroupements différents. La figure 1.8 (a) représente le tableau initial, la figure 1.8 (b) un regroupement par une méthode de fractionnement sans permutation et la figure 1.8 (c) un regroupement imbriqué par blocs. Pour isoler le groupe de 2, la méthode par fractionnement sépare l'ensemble des 1 contrairement à la méthode imbriquée par bloc. Dans ce type de classification, certaines données peuvent être mises à l'écart.

Ce type de regroupement est utilisé notamment en génétique pour séparer le bruit des données intéressantes. Dans le cas de la classification imbriquée restreinte (figure 1.6 (c)), les blocs sont contraints à être imbriqués c'est-à-dire que pour tout $\kappa \geq 1$, $U_\kappa \subseteq U_{\kappa-1}$ où $U_0$ représente la partition initiale alors que dans le cas non restreint, nous pouvons avoir différents groupes de blocs (figure 1.6 (b)).

Pour la classification non restreinte, Hartigan (Hartigan, 1975, Chapitre 15) propose l'algorithme de *fractionnement joint à deux sens* (*Two-Way Joining Algorithm*) reprenant le principe de fractionnement

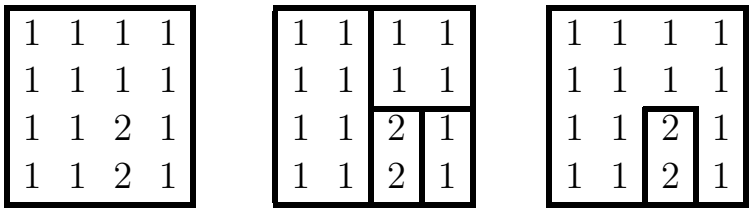



FIGURE 1.8 – Différence entre un regroupement par fractionnement et imbriqué par blocs : à gauche le tableau initial, au milieu une partition par fractionnement et à droite un regroupement imbriqué.

direct mais adapté aux blocs plutôt qu'aux lignes.

La classification restreinte peut être traitée comme un cas particulier du regroupement simple de la partie 1.2.5 où à chaque étape, le bloc $U_{\kappa(c)}$ obtenu à la $c^{\text{ème}}$ itération est considéré comme étant la matrice à étudier.

### 1.2.5 Classification avec chevauchement

La dernière classification présentée est celle avec chevauchement. Le but est la recherche d'un ou plusieurs blocs atypiques qui peuvent éventuellement se chevaucher. Sur la figure 1.9 sont représentées deux parties de matrices avec deux blocs se chevauchant soit parce qu'ils ont un lien (comme sur la (a)), soit car ils ont une partie différente (comme sur la (b)). Les classifications vues jusqu'à présent empêchent ce chevauchement. Comme mentionné dans la section 1.2.2, les cases composant ces blocs atypiques ne sont pas forcément égales à une même valeur. Notamment, en section 1.2.5.4 est présenté un algorithme recherchant simplement une structure ordonnée sur les lignes.

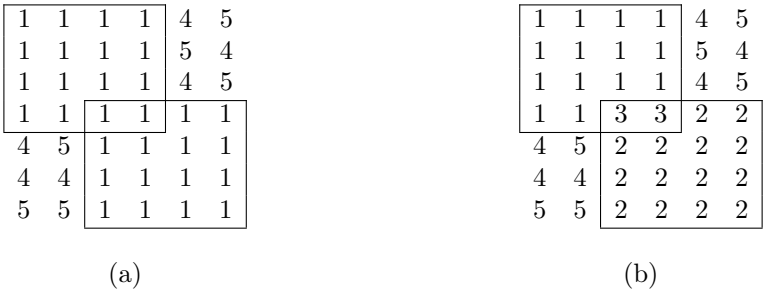



FIGURE 1.9 – Deux classifications avec chevauchement : celle de gauche est composée de blocs avec l'intersection faisant le lien entre les deux et celle de droite comporte des blocs avec l'intersection considérée comme un bloc à part.

En reprenant la matrice réordonnée de la figure 1.1, nous pouvons mettre en évidence trois blocs (voir la partie supérieure de la figure 1.10). Le premier, en haut à gauche de la matrice, est à la fois grand et possédant une grande proportion de cases noires (voir la figure 1.10, en haut à gauche). Le second, en bas à droite de la matrice, est plus petit mais possède une grande proportion de cases noires également

(voir la figure 1.10, en haut au milieu). Enfin, celui au milieu de la matrice reprend des cases des deux précédents pour obtenir un bon rapport taille/proportion de cases noires (voir la figure 1.10, en haut à droite). Le choix du critère est important car, suivant celui-ci, les blocs peuvent être plus petits mais avec une plus grande proportion de cases noires.
Une fois les blocs obtenus, nous pouvons étudier les sous-graphes associés qui sont majoritairement denses (voir la partie inférieure de la figure 1.10).

La classification par chevauchement (figures 1.10 et 1.11) est la moins restreinte. La seule condition pour deux blocs $U_\kappa$ et $U_{\kappa'}$ est qu'ils ne soient pas identiques :

$$U_\kappa \cap U_{\kappa'} \neq \emptyset \Rightarrow U_\kappa \neq U_{\kappa'}.$$

Toutefois, la représentation schématique proposée en figure 1.10 ne peut être possible que si la réorganisation des lignes et des colonnes s'y prête. Sur la figure 1.11, nous voyons quatre blocs mis en évidence et se chevauchant (le jaune, le bleu, le rouge et le bordeaux). A gauche, le bloc bordeaux est scindé en deux alors qu'à droite, c'est le jaune. Nous voyons qu'aucune permutation ne permet d'avoir une représentation de chacun des quatre blocs d'un seul tenant en même temps.

De nombreux algorithmes sont proposés pour cette question, Prelić et al. (2006) font une comparaison de cinq d'entre eux (*SAMBA*, *ISA*, *OPSM*, *xMotif* et les algorithmes de Cheng et Church (1999)) et Hanczar et Nadif (2010) les utilisent pour proposer une méthode basée sur du bootstrap.

Dans cette section, nous nous limitons à certains d'entre eux et mettons en annexe 1.A un tableau non exhaustif des algorithmes proposés pour résoudre ce problème. D'abord, la $\delta$-classification est traitée pour mieux comprendre les problèmes algorithmiques sous-jacents provenant de ce relâchement des hypothèses. Dans un second temps, une vision probabiliste est abordée et enfin, des approches spécifiques pour le cas binaire et le cas de données ordinales sont présentées.

#### 1.2.5.1 $\delta$-classification

Pour répondre au problème de classification par chevauchement, Cheng et Church (1999) introduisent une fonction de coût $H$ des résidus quadratiques d'un bloc $U_\kappa = (Z_\kappa, W_\kappa)$ :

$$H(U_\kappa) = \frac{1}{|Z_\kappa||W_\kappa|} \sum_{i \in Z_\kappa, j \in W_\kappa} (x_{ij} - \overline{x}_{iW_\kappa} - \overline{x}_{Z_\kappa j} + \overline{\mathbf{x}}_{U_\kappa})^2$$

avec $|Z_\kappa|$ représentant le cardinal de l'ensemble $Z_\kappa$, $\overline{\mathbf{x}}_{U_\kappa}$ la moyenne du bloc $U_\kappa$ et

$$\overline{x}_{iW_\kappa} = \frac{1}{|W_\kappa|} \sum_{j \in W_\kappa} x_{ij} \qquad \overline{x}_{Z_\kappa j} = \frac{1}{|Z_\kappa|} \sum_{i \in Z_\kappa} x_{ij}.$$

Ce critère étant nul pour des blocs contenant des progressions arithmétiques en ligne et en colonne, le minimiser revient souvent à sélectionner des blocs ne contenant qu'une seule case. Cheng et Church (1999) proposent plusieurs algorithmes cherchant le plus grand bloc $U_\kappa$ tel que $H(U_\kappa) \leq \delta$ où $\delta > 0$ est un paramètre donné. Nous parlons de *$\delta$-classification*.

L'algorithme *Brute-Force Deletion and Addition* calcule pour chaque bloc $U_\kappa$ possible sa valeur $H(U_\kappa)$ en commençant le calcul sur la matrice initiale, puis sur tous les blocs avec une ligne ou une colonne en moins, puis avec deux, et ainsi de suite jusqu'à obtenir le plus grand bloc de coût inférieur à $\delta$. Cet algorithme est optimal mais la complexité étant de l'ordre $\mathcal{O}(2^{nd})$, il ne peut s'appliquer qu'à des petites matrices (avec moins de 20 lignes et colonnes).

Deux versions plus rapides basées respectivement sur des algorithmes de type $forward$ et *backward* consistent pour l'une, de partir de la matrice initiale et de supprimer une ligne ou une colonne en faisant

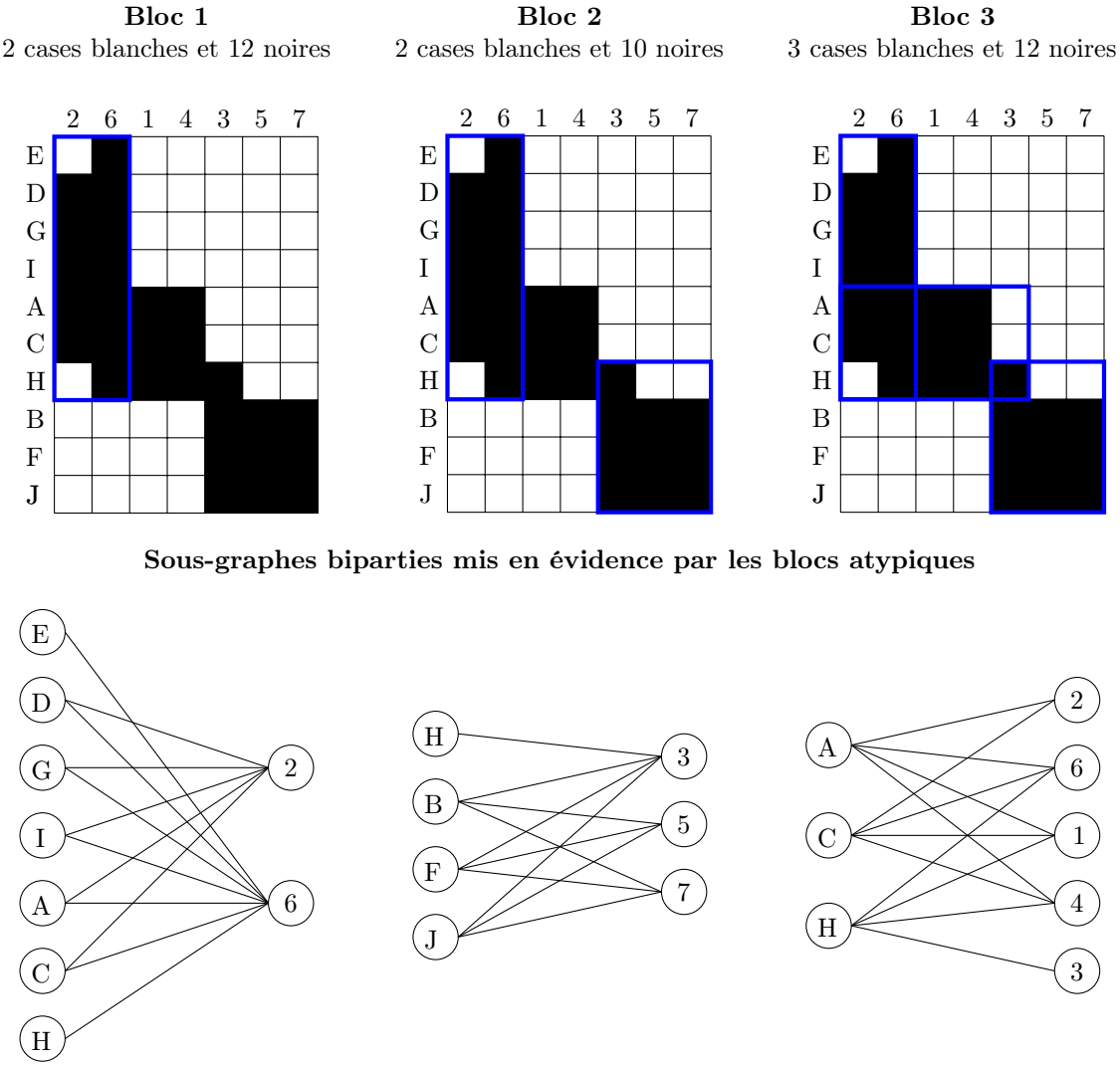



FIGURE 1.10 – Sur la partie supérieure se trouve l'évolution de la sélection des différents blocs : à gauche se trouve l'un des plus grands blocs ayant une proportion supérieure à 5/6 ème de cases noires (dans la partie supérieure gauche de la matrice) ; au milieu, un bloc contenant une proportion de 4/5 ème de cases noires n'étant pas en intersection avec le premier (dans la partie inférieure droite de la matrice) et enfin, un bloc de cases noires d'une proportion de cases noires de 3/4 rt (dans le milieu de la matrice). Sur la partie inférieure se trouvent les sous-graphes associés aux blocs atypiques mis en évidence.

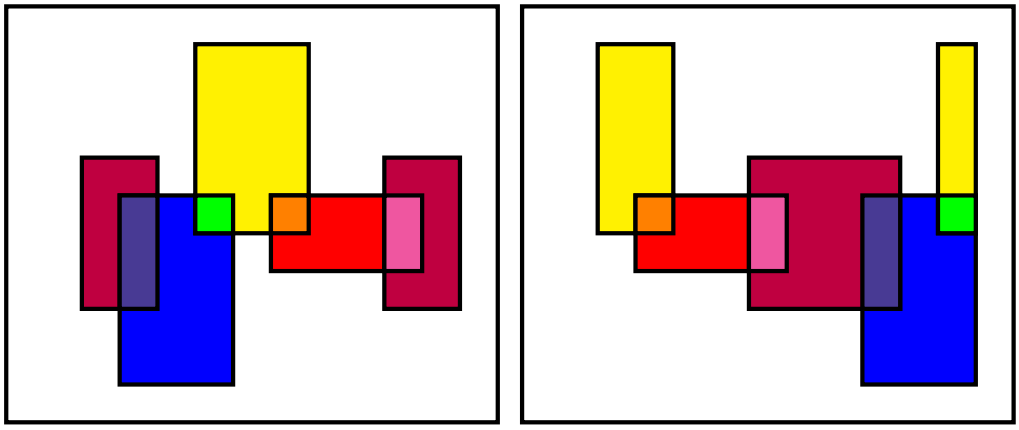


FIGURE 1.11 – Représentations schématiques d'une classification avec chevauchement suivant deux réorganisations des colonnes. Aucune permutation ne permet d'avoir une représentation de chacun des quatres blocs d'un seul tenant en même temps.

diminuer la fonction de coût le plus vite possible (algorithme *Multiple Node Deletion*), pour l'autre, de partir de l'ensemble vide et d'ajouter à chaque étape une ligne ou une colonne en faisant augmenter la fonction de coût le moins possible (algorithme *Multiple Node Addition*). Ces algorithmes ont une complexité de l'ordre $\mathcal{O}\left((nd)^2\right)$ et convergent vers des maxima locaux.

Enfin, Cheng et Church (1999) proposent un algorithme type $forward/backward$ alternant les deux précédents. Une fois un premier bloc $U_1$ obtenu, l'algorithme remplace les cases de $U_1$ par un tirage aléatoire et recommence jusqu'à obtenir le nombre $u$ de blocs désiré. Cet algorithme a été testé sur l'expression du cycle cellulaire de la levure Saccharomyces cerevisiae obtenant 100 blocs de telle sorte que toutes les conditions et $97,12\%$ des gènes soient classés dans au moins un bloc.

Sur la même hypothèse, Yang et al. (2002) proposent l'algorithme *FLOC* (pour *FLexible Overlapped Clustering*) utilisant la même procédure mais directement sur les $u$ blocs. Cet algorithme possède une complexité en $\mathcal{O}\left((nd)^2 u\right)$ et est moins dépendant du hasard que celui de Cheng et Church (1999). Toutefois, les algorithmes de $\delta$-classifications sont peu robustes face au bruit, préférant des groupes plus grands que ceux ayant servi pour les simulations (voir Prelić et al., 2006).

#### 1.2.5.2 Vision probabiliste

Dans leur article, Lazzeroni et Owen (2000) proposent une vision probabiliste pour trouver les $u$ blocs (avec $u$ inconnu). Ce modèle suppose que les blocs $U_\kappa = (Z_\kappa, W_\kappa)$ peuvent éventuellement se chevaucher et que chaque bloc possède une structure propre, c'est-à-dire qu'il existe une fonction $f_\kappa(i,j)$ rattachée à ce bloc. Ce modèle traite ainsi tous les cas évoqués en section 1.2.2. Une case $x_{ij}$ va appartenir à un, plusieurs ou aucun bloc et sera le résultat du tirage aléatoire d'une variable $X_{ij}$ dont la loi va dépendre des effets cumulés des blocs auxquels elle appartient :

$$X_{ij} = \sum_{\kappa=1}^{u} f_\kappa(i,j) z_{i\kappa} w_{j\kappa} + \varepsilon_{ij}$$

avec $z_{i\kappa}$ (resp $w_{j\kappa}$) valant 1 si la $i^{\text{ème}}$ ligne appartient à $Z_\kappa$ (resp la $j^{\text{ème}}$ colonne appartient à $W_\kappa$) et $\varepsilon_{ij}$ des variables aléatoires indépendantes de même loi. Ce modèle prend en compte le cas de la figure 1.9 (b) en considérant les $f_\kappa(i,j)$ constants par blocs. Contrairement à la section 1.2.3, il n'est fixé aucune

contrainte sur les $z_{i\kappa}$ et les $w_{j\kappa}$ qui pourront, par exemple, valoir 1 pour toute une ligne $z_{i\cdot}$ ou $w_{j\cdot}$. Dans le cas d'un bruit symétrique, l'idée est alors de minimiser les moindres carrés :

$$\sum_{i=1}^{n}\sum_{j=1}^{d}\left(x_{ij}-\sum_{\kappa=1}^{u}f_{\kappa}(i,j)z_{i\kappa}w_{j\kappa}\right)^2.$$

Ceci n'est pas réalisable en un temps raisonnable sachant de plus que $u$ est inconnu. Lazzeroni et Owen (2000) proposent un algorithme itératif cherchant le $\kappa$ ème bloc à partir des $\kappa-1$ précédents en minimisant :

$$\sum_{i=1}^{n}\sum_{j=1}^{d}\left(x_{ij}^{\kappa-1}-f_{\kappa}(i,j)z_{i\kappa}w_{j\kappa}\right)^2$$

où

$$x_{ij}^{\kappa-1}=x_{ij}-\sum_{\kappa'=1}^{\kappa-1}\widehat{f_{\kappa'}}(i,j)z_{i\kappa'}w_{j\kappa'}$$

représente la matrice où les valeurs estimées $\widehat{f_{\kappa'}}$ des blocs précédents ont été retranchées (si une case $x_{ij}$ dépend uniquement des blocs déjà trouvés, $x_{ij}^{\kappa-1}$ est alors le résultat du tirage de $\varepsilon_{ij}$). L'algorithme s'arrête lorsque les cases $x_{ij}^{\kappa-1}$ du dernier bloc obtenu sont considérées comme les résultats des tirages de variables aléatoires $\varepsilon_{ij}$.

Les auteurs ont testé leurs algorithmes sur des données nutritionnelles de 961 aliments, ce qui a permis de séparer certains groupes d'aliments suivant différentes variables nutritionnelles (par exemple, certains parmesans, parties du boeuf et parties du poulet ont été séparés car possédant plus de protéines et moins de cholestérol que d'autres aliments) ; ils ont aussi étudié les valeurs mensuelles de taux de change entre différents pays séparant, par exemple, 71 mois où le dollar l'emportait sur 11 autres devises ; ou encore sur des données d'expression de gènes de levures suivant différentes conditions expérimentales mettant en évidence 34 blocs se chevauchant avec certains gènes se retrouvant dans 18 groupes.

#### 1.2.5.3 Samba

Dans le cas de matrices binaires, Tanay et al. (2002) définissent un algorithme probabiliste appelé *SAMBA*. En prenant le point de vue des graphes biparties, les cases $x_{ij}$ valant 1 si les sommets $i$ et $j$ sont reliés et 0 sinon, le but est de chercher un sous-graphe à la fois grand et dense.

L'idée de base est de supposer que chaque case est le résultat d'un tirage aléatoire d'une loi de Bernoulli de paramètre $p$ la proportion de cases valant 1 dans la matrice. L'objectif est de sélectionner un bloc $U_\kappa$ contenant une grande proportion de 1 tout en étant le plus grand possible. En première approche, Tanay et al. (2002) proposent d'utiliser une fonction de coût $H_p$ calculant, pour chaque bloc $U_\kappa=(Z_\kappa,W_\kappa)$, la valeur de la queue d'une loi binomiale de paramètres $|Z_\kappa||W_\kappa|$ et $p$ à partir de la somme des cases du bloc (c'est-à-dire le nombre de cases valant 1) :

$$H_p(U_\kappa)=\sum_{\ell=\sum_{x_{ij}\in U_\kappa}x_{ij}}^{|Z_\kappa||W_\kappa|}\binom{|Z_\kappa||W_\kappa|}{\ell}p^{\ell}(1-p)^{1-\ell}. \tag{1.2}$$

Pour sélectionner des blocs ayant un grand nombre de 1, il faut que $p$ soit plus petit que 1/2. Sous cette condition, la fonction de coût ne favorise pas les blocs monocases (contrairement à celle de la section 1.2.5.1) car, par exemple, elle sélectionne un bloc de 4 cases contenant 3 cases avec des 1 plutôt qu'un bloc d'une seule case valant 1. En revanche, la structure globale de la matrice n'influence pas le choix du bloc car pour tout $p$ plus petit que 1/2, les fonctions $H_p$ sélectionnent les mêmes blocs.

Pour contourner ce problème, Tanay et al. (2002) supposent que les cases $x_{ij}$ sont le résultat de variables aléatoires de lois de Bernoulli de paramètre $p_{ij}$ représentant la fraction de sous-graphes biparties contenant l'arête $(i, j)$ et ayant le même degré que le graphe associé à la matrice totale. L'estimation de ces paramètres est faite par simulation de Monte-Carlo et est d'autant plus proche de 1 que les sommets de l'arête sont reliés avec d'autres sommets. En adaptant le critère (1.2) et une fois les $p_{ij}$ obtenus, l'algorithme *SAMBA* recherche le sous-graphe désiré avec une complexité de $\mathcal{O}\left(d2^{\max_j x_{+j}}\right)$. L'algorithme est d'autant plus rapide que le nombre de colonnes est faible et qu'il n'y a pas trop de 1 dans celles ci.
L'algorithme a été testé sur les données de levure mettant en évidence des liens entre des gènes bien connus et d'autres qui n'étaient pas encore classifiés.

#### 1.2.5.4 Algorithme *OPSM*

Dans le cas de données où une relation d'ordre est possible (comme des données ordinales), Bendor et al. (2002) proposent un algorithme recherchant un bloc avec une progression ordonnée sur les colonnes. Dans ce modèle, le bloc $U_\kappa = (Z_\kappa, W_\kappa)$ est supposé être généré en deux temps : soit $s$ un entier positif inférieur à $d$ et $W_\kappa = \{j_1, \ldots, j_s\}$ un groupe ordonné parmi tous les groupes possibles de tailles $s$ de $\mathcal{P}(\{1, \ldots, d\})$ l'ensemble de toutes les parties de $\{1, \ldots, d\}$. Chaque ligne $i$ a la même probabilité $\pi$ d'appartenir à $Z_\kappa$ et, dans ce cas, les cases de la ligne $i$ correspondant aux colonnes de $W_\kappa$ vérifient $x_{ij_1} < x_{ij_2} < \ldots < x_{ij_s}$, les autres cases étant considérées comme du bruit.
Connaître $W_\kappa$ permet l'obtention rapide d'un ensemble contenant $Z_\kappa$, il suffit de conserver les lignes respectant l'ordre demandé. En pratique, la variable $W_\kappa$ est inconnue et il est impossible de faire le même raisonnement pour chaque groupe de taille $s$.

L'algorithme *OPSM* (pour *order-preserving submatrices*) cherche les indices de $W_\kappa$ en commençant par les plus extrêmes ; c'est-à-dire les indices $W_\kappa^{(1)} = \{j_1, j_s\}$ puis en alternant les petits et grands indices ($W_\kappa^{(2)} = \{j_1, j_2, j_s\}$, $W_\kappa^{(3)} = \{j_1, j_2, j_{s-1}, j_s\}$, $W_\kappa^{(4)} = \{j_1, j_2, j_3, j_{s-1}, j_s\}$...).
L'idée est que les "grands sauts" entre les valeurs sur les colonnes sont d'autant plus visibles que les indices sont séparés dans $W_\kappa$ et la suite des groupes $Z_\kappa^{(c)}$ associés à chaque $W_\kappa^{(c)}$ est décroissante. L'inconvénient est que cet algorithme est sensible à l'estimation du premier groupe $W_\kappa^{(1)}$.

De plus, les auteurs soulignent que l'obligation d'ordre est trop rigide (ceci le rend très sensible au bruit (voir Prelić et al., 2006)) et suggèrent un allègement de la condition en incluant une probabilité que certaines valeurs puissent rompre la relation.
Cet algorithme a été testé sur des données du cancer du sein (Hedenfalk et al., 2001). L'algorithme a extrait de la matrice de taille $3226 \times 22$ une sous matrice de taille $347 \times 4$.

### 1.2.6 Problèmes ouverts

La comparaison de toutes ces méthodes est difficile à cause du manque de consensus sur l'objectif poursuivi et les critères d'évaluation des méthodes. Nous présentons dans les deux sections suivantes quelques éléments de réponse à ces questions. À la fin, nous discutons du problème du choix du nombre de blocs, déjà abordé avec certains critères.

#### 1.2.6.1 Unité statistique

En classification simple, les données sont généralement sous la forme d'un tableau où chaque ligne représente un objet ou individu et où chaque colonne représente une variable enregistrée sur un ensemble d'objets. L'objectif de la classification est de regrouper les individus, c'est-à-dire les lignes du tableau de données, qui sont alors les unités d'intérêt.

En classification croisée, le but est de trouver la structure latente d'un tableau de données en blocs homogènes de lignes et de colonnes. Se pose alors la question de l'unité statistique et ce choix reste encore

à ce jour une question ouverte : est-ce le nombre de vecteurs en ligne, en colonne, le nombre d'éléments du tableau ou le tableau lui-même ?

Cette question a une importance notamment lors de l'étude asymptotique des propriétés des modèles probabilistes employés, lors du développement et de l'étude de la consistance des critères de sélection (Keribin et al., 2012; Lomet et al., 2012b; Keribin et al., 2014).

#### 1.2.6.2 Évaluation des méthodes

L'évaluation objective des techniques d'apprentissage non supervisé sur des données réelles est difficile. Plusieurs points de vue sont légitimes et mesurer la pertinence d'une classification par sa capacité à retrouver un étiquetage particulier n'est donc pas nécessairement approprié. Ainsi, l'utilisation de données simulées est une technique traditionnelle pour l'évaluation expérimentale des algorithmes de classification car elle apporte une connaissance a priori sur les données. Un ensemble de jeux de données contrôlés permet de tester les limites d'un algorithme quant à sa capacité à retrouver une structure donnée dans des cas plus ou moins favorables.

L'absence de jeux de données de référence et de consensus sur l'évaluation des algorithmes amène chaque auteur (Good, 1965; Banerjee et al., 2007; Mariadassou et al., 2010) à proposer son propre protocole dont la reproductibilité est souvent limitée, du fait de la brièveté de la description et l'absence de mise à disposition des données. De plus, la qualité des résultats est difficilement évaluable car la difficulté intrinsèque du problème de classification est inconnue du lecteur. En effet, si en classification simple, une analyse en composantes principales permet de visualiser le degré de mélange des classes, en classification croisée, cette approche est plus difficile du fait de la dualité des lignes et des colonnes du tableau de données. Un exemple typique de ce phénomène est le jeu de données « Classic3 » utilisé en *text mining* qui, selon la métrique utilisée pour sa représentation, peut amener à une visualisation des classes très séparées, mettant ainsi en évidence un jeu de données facile à classer. Lomet et al. (2012c) ont proposé un protocole de simulation dont la mesure de la difficulté est exprimée par le risque conditionnel de Bayes : le risque de Bayes est conditionné au tableau observé. Ces auteurs ont mis à disposition [1] des jeux de données simulées binaires, continues et de contingence de différentes difficultés et tailles pour le cas de la classification croisée par blocs.

#### 1.2.6.3 Choix du nombre de blocs

Si certains modèles incluent dans leur définition le choix du nombre de blocs (Lazzeroni et Owen, 2000; Wyse et Friel, 2010), l'ensemble des méthodes de classification croisée présentées dans les sections précédentes pose le problème majeur du choix de modèle. L'objectif de cette sélection est d'obtenir une classification pertinente, à un niveau de granularité approprié.

Ce problème est bien connu et abondamment traité en classification simple (Fraley et Raftery, 1998; Biernacki et al., 2000; Burnham et Anderson, 2004). Les solutions existantes en classification simple, basées sur des indices, des heuristiques ou des critères d'information, ont été ou peuvent être adaptées à la classification croisée.

##### 1.2.6.3.1 Indices de sélection adaptés de la classification simple

Charrad et al. (2010) étendent à la classification croisée sept indices utilisés en classification simple : l'indice de Dunn, l'indice de Baker et Hubert, l'indice de Davies et Bouldin, l'indice de Calinsky et Harabsz, l'indice Silhouette de Rousseeuw, l'indice de Hubert et Levin et l'indice de Krzanowski et Lai. Leur proposition consiste à appliquer ces différents indices de classification simple indépendamment sur les lignes et sur les colonnes du tableau de donnés, puis de calculer un indice global par une combinaison linéaire de ces deux derniers. Charrad et al. (2010) proposent également un huitième indice appelé la

1. Les données sont disponibles à l'adresse `https://www.hds.utc.fr/coclustering/doku.php`.

méthode de la différentielle : utilisant l'algorithme `Croki2` (Govaert, 1983) qui maximise le critère du $\chi^2$, cette méthode a pour objectif de sélectionner le nombre de classes pour lequel ce critère ne croît plus ou croît plus lentement. Ces huit indices ont été testés sur six tables de contingence simulées de taille $200 \times 100$ et dont le nombre de classes diffère. Dans ces expériences, tous les indices sélectionnent majoritairement les bons nombres de classes, avec un léger avantage pour la méthode différentielle et l'indice de Baker et Hubert. Comme toutes les approches basées sur l'évaluation des deux classifications du tableau, en ligne et en colonne, ces indices ne tiennent pas compte de la structure croisée du problème. Le choix de la pondération de la combinaison linéaire est alors critique, et Charrad et al. (2010) n'apportent pas de solution à ce nouveau problème.

Rocci et Vichi (2008) proposent une extension de l'indice de Calinski-Harabasz (CH) qui est le rapport entre la dispersion inter-blocs et la dispersion intra-blocs. Cet indice est le plus performant d'une étude comparative de 30 indices en classification simple (Milligan et Cooper, 1985). Dans les expériences sur données simulées de Rocci et Vichi (2008), cet indice obtient de bons résultats pour des classes très séparées.

Schepers et al. (2008) proposent une extension de l'indice Silhouette qui est appliqué pour des classifications simples basées sur la distance euclidienne comme les $k$-moyennes (*k-means*). Cette extension est une moyenne pondérée par le nombre de lignes et de colonnes des indices Silhouette calculés sur l'ensemble des lignes et des colonnes. Ces auteurs font la remarque que ces deux derniers indices sont de simples extensions d'indices utilisés pour les $k$-moyennes et par conséquent ne prennent pas pleinement en compte la nature spécifique des données et des modèles en classification croisée.

#### 1.2.6.3.2 Sélection de modèle en classification croisée utilisant un modèle probabiliste

L'utilisation de modèles statistiques permet de considérer le problème du choix du nombre de classes comme un problème de sélection de modèle. Des critères de sélection employés généralement en statistique peuvent donc être dérivés et appliqués au cas de la classification croisée.

Utilisant le modèle des blocs latents présentés précédemment, Van Dijk et al. (2009) emploient deux critères empiriques de sélection de modèles pour choisir le nombre de classes : un dérivé des critères empiriques associés au facteur de Bayes et le critère AIC3. Le facteur de Bayes sélectionne le modèle dont la vraisemblance marginale est maximale. Ce facteur se base sur l'estimation des distributions marginales pour plusieurs nombres de classes en ligne et en colonne qui sont très coûteuses. Van Dijk et al. (2009) appliquent aussi le critère AIC3 en se basant sur les résultats des simulations de Andrews et Currim (2003). La pénalité de ce critère tient compte du nombre de paramètres du modèle, du nombre de classes et du nombre d'étiquettes des appartenances aux classes estimées. Les auteurs n'étudient pas les performances des critères sur des données simulées, mais ils argumentent la pertinence des classes obtenues sur des données réelles.

Wyse et Friel (2010) ont développé une approche bayésienne du modèle des blocs latents dont le choix du nombre de classes est un paramètre intégré dans le modèle. Ils emploient une approche bayésienne utilisant un algorithme MCMC (*Markov Chain Monte Carlo*) incluant une distribution a priori sur le nombre de classes. Ils ont notamment testé leur algorithme sur des données simulées et réelles, comme celles du parlement américain étudiées dans le chapitre 4.

Récemment, Keribin et al. (2012) et Lomet et al. (2012b,a) ont montré que le critère exact ICL (*Integrated Classification Likelihood*), initialement développé dans le cas de la classification simple par modèle de mélange (Biernacki et al., 2000), peut être calculé exactement au cas du modèle des blocs latents binaires et qualitatifs afin de sélectionner le nombre de classes. Le critère ICL évalue le logarithme de la vraisemblance intégrée des données complétées du modèle des blocs latents. Supposant l'indépendance a priori des paramètres du modèle des blocs latents binaires et des distributions a priori sur ces derniers, ces auteurs développent un critère de sélection exact.Ces critères sont notamment étudiés dans le chapitre 4.

### 1.2.7 Quand pouvons-nous utiliser la classification croisée ?

En statistique, les observations $x_i$ sont supposées être de même nature : il serait étrange de comparer en même temps des sportifs courant un 100 mètres et la qualité de différents jus d'orange par exemple.

En classification croisée, comme nous cherchons à la fois à classer les lignes et les colonnes de la matrice, il faut que cette condition soit vraie également pour les éléments représentés en colonnes. Par exemple, si l'une des colonnes représente le sexe de l'individu $i$ et une autre la possession d'animaux de compagnies ou non ; une façon de binariser les données est de mettre 1 si c'est une femme, 0 si c'est un homme, 1 s'il a un animal et 0 sinon. En classification simple, l'inversion du codage H/F a peu d'importance. En revanche, comme nous pouvons le voir sur la figure 1.12, cette inversion peut influencer directement la classification des colonnes mettant ainsi la question sur le "sexe" et sur les "animaux" dans la même classe ou non.

De plus, si certaines binarisations peuvent paraître intuitives (comme l'absence ou présence d'un état), l'angle de la question peut donner deux configurations opposées : par exemple, pour connaître la situation familiale, il est possible de demander "êtes-vous en couple ?" ou "êtes-vous célibataire ?" ; bien que dans les deux cas, il est naturel de mettre zéro pour "non" (absence d'un état), les deux résultats renvoient des configurations opposées.

Toutefois, si toutes les questions attendent pour réponse "homme" ou "femme", le codage n'a alors aucune influence, puisque le choix de la convention inverse modifiera de façon cohérente l'ensemble des cases du tableau.

En particulier, ce problème est moins présent lorsque $I = J$ (comme le cas du SBM) et dans le cas de tableaux de contingence où, généralement, nous avons des observations à la fois sur les lignes et les colonnes.

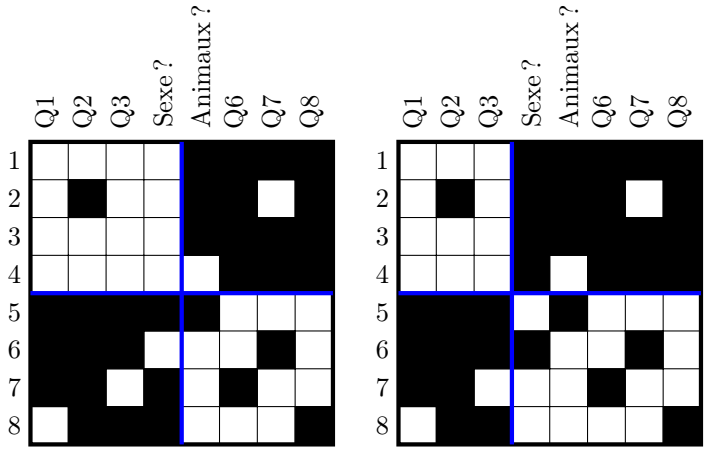



FIGURE 1.12 – Représentation schématique de l'influence de la binarisation : sur le tableau de gauche, la colonne "sexe" est mise dans la classe 1 alors que dans celui de droite, où la binarisation est inversée, la colonne appartient à la classe 2.

Les unités de mesure sont également importantes, en particulier pour les données continues. Si dans un tableau, une variable en colonne prend des valeurs en mètres alors qu'une autre est en kilomètres, cela n'influence pas la classification si nous ne regardons que les individus mais cela a une véritable importance pour la classification croisée. De plus, la même unité de mesure n'a pas le même sens pour deux

variables : deux villes situées à une dizaine de kilomètres l'une de l'autre sont considérées généralement comme proches tandis qu'une montagne dont la cime est à sept kilomètres est considérée beaucoup plus haute qu'une montagne où le sommet est à deux kilomètres.

En conclusion, avant d'utiliser la classification croisée, il est important de réfléchir au codage des variables et au sens que pourrait ou ne pourrait pas avoir tel ou tel regroupement.

## 1.3 Modèle des blocs latents

Comme mentionné précédemment, le modèle étudié dans cette thèse est celui des blocs latents (ou LBM) qui se place dans le cadre de la classification croisée par blocs. Nous cherchons donc une partition en ligne et une partition en colonne de telle sorte que les blocs obtenus par leur produit cartésien contiennent des données homogènes différentes de celles des autres blocs. Pour ce modèle, nous supposons que les données sont homogènes si elles sont le résultat de variables aléatoires de même loi pour un bloc.

### 1.3.1 Hypothèses, modèle et notations

Dans la suite du manuscrit, et sauf mention contraire, nous supposons que le nombre de classes en ligne et en colonne sont des paramètres fixes notés respectivement $g$ et $m$. Comme expliqué en section 1.2.3.3.1, nous faisons trois hypothèses :

$(H_1)$ il existe des partitions en ligne $\mathbf{z}$ et en colonne $\mathbf{w}$ telles que les cases $x_{ij}$ sont le résultat de variables aléatoires $X_{ij}$ indépendantes conditionnellement au couple $(\mathbf{z}, \mathbf{w})$ :

$$p(\mathbf{x}|\mathbf{z}, \mathbf{w}) = \prod_{i=1}^{n} \prod_{j=1}^{d} p(x_{ij}|\mathbf{z}, \mathbf{w})$$

et leurs lois appartiennent à une même famille de lois paramétriques dont le paramètre ne dépend que du bloc,

$(H_2)$ les variables $Z$ et $W$ sont indépendantes :

$$\forall (\mathbf{z}, \mathbf{w}) \in \mathcal{Z} \times \mathcal{W}, \quad p(\mathbf{z}, \mathbf{w}) = p(\mathbf{z})p(\mathbf{w}),$$

$(H_3)$ pour chaque ligne $i$, la loi d'affectation des labels est une loi multinomiale dont les paramètres sont les mêmes pour toutes les lignes ; de même pour les colonnes.

Ces hypothèses impliquent un certain nombre de notations que nous détaillons dans la suite de cette section.

$(H_1)$ Comme les partitions en ligne et en colonne sont supposés être des variables aléatoires, nous notons $Z$ la représentation des classes en ligne et $W$ celle des classes en colonne. Suivant les auteurs, $Z$ représente un vecteur colonne de taille $n$ où la ligne $i$ appartient à la classe $k$ si et seulement si $z_i = k$ ou une matrice binaire de taille $n \times g$ où la ligne $i$ appartient à la classe $k$ si et seulement si $z_{ik} = 1$. Comme nous pouvons le voir sur l'exemple 1.3.1, la matrice binaire est composée d'exactement une seule case valant 1 par ligne (voir tableau 1.3) ; c'est la notation privilégiée par la suite.

Dans le cas des matrices binaires, nous obtenons les effectifs de chaque classe en sommant les cases pour chaque colonne. Nous notons donc $z_{+k}$ et $w_{+\ell}$ les cardinaux respectifs de la $k$ ème classe en ligne et la $\ell$ ème classe en colonne (voir le tableau 1.4). Le nombre de cases dans le bloc $(k, \ell)$ est alors le produit

de $z_{+k}$ par $w_{+\ell}$.

*Exemple* 1.3.1. En reprenant l'exemple de la figure 1.3 et en assimilant les lettres à leurs positions dans l'alphabet, nous obtenons les matrices d'appartenance aux classes dans le tableau 1.3.

| | Matrice colonne | Matrice binaire |
|---|---|---|
| 10 lignes classées dans 3 classes | $\mathbf{z} = \begin{pmatrix} 2 \\ 3 \\ 2 \\ 1 \\ 1 \\ 3 \\ 1 \\ 2 \\ 1 \\ 3 \end{pmatrix}$ | $\mathbf{z} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ |
| 7 colonnes classées dans 3 classes | $\mathbf{w} = \begin{pmatrix} 2 \\ 1 \\ 3 \\ 2 \\ 3 \\ 1 \\ 3 \end{pmatrix}$ | $\mathbf{w} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ |

TABLE 1.3: Exemples de matrices d'appartenance aux classes à partir de l'exemple de la figure 1.3.

En sommant les colonnes des matrices binaires, nous obtenons les effectifs pour chaque classe en ligne et en colonne. Ensuite, en multipliant $z_{+k}$ par $w_{+\ell}$, nous obtenons le nombre de cases dans le bloc $(k, \ell)$. Toutes ces informations sont résumées pour l'exemple du tableau 1.3 dans le tableau 1.4.

| $\mathbf{z}$ \ $\mathbf{w}$ | | $w_{+1}$ | $w_{+2}$ | $w_{+3}$ |
|---|---|---|---|---|
| | | 2 | 2 | 3 |
| $z_{+1}$ | 4 | 8 | 8 | 12 |
| $z_{+2}$ | 3 | 6 | 6 | 9 |
| $z_{+3}$ | 3 | 6 | 6 | 9 |

TABLE 1.4 – À partir des tableaux binaires 1.3, nous obtenons les effectifs de chaque classe et de chaque bloc.

En plus de l'indépendance des variables $X_{ij}$ conditionnellement à la connaissance des blocs, nous supposons que toutes les variables $X_{ij}$ suivent une loi appartenant à la même famille paramétrique dont la densité est notée $\varphi$ et dont les paramètres $\alpha_{k\ell}$ dépendent du bloc $(k,\ell)$. Ainsi, toutes les cases d'un même bloc $(k,\ell)$ sont la réalisation de variables aléatoires de même loi de densité $\varphi(x_{ij};\alpha_{k\ell})$ :

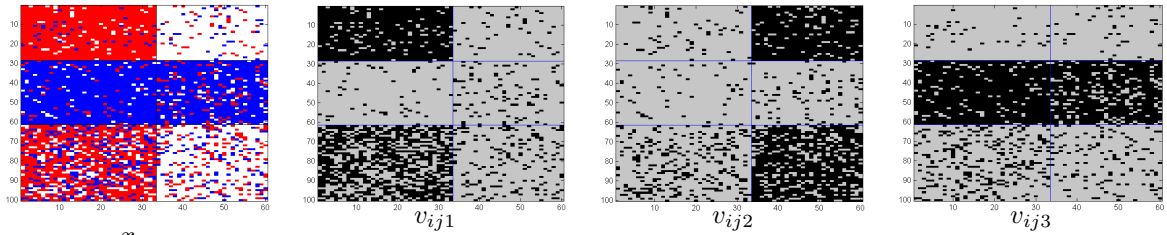
$$p(\mathbf{x}|\mathbf{z},\mathbf{w}) = \prod_{i=1}^{n}\prod_{j=1}^{d}\prod_{k=1}^{g}\prod_{\ell=1}^{m}\varphi(x_{ij};\alpha_{k\ell})^{z_{ik}w_{j\ell}}$$

où $z_{ik}w_{j\ell}$ vaut 1 si et seulement si la case $x_{ij}$ appartient au bloc $(k,\ell)$. Nous résumons par $\boldsymbol{\alpha} = (\alpha_{k\ell})_{g\times m}$ la matrice regroupant les paramètres $\alpha_{k\ell}$.
Dans cette thèse, nous étudions des données catégorielles et binaires : nous supposons que les variables aléatoires prennent des valeurs dans l'ensemble non ordonné $\{1,\dots,r\}$.

La loi étudiée est donc une loi multinomiale de paramètres 1 et $\alpha_{k\ell} = \left(\alpha^1_{k\ell},\dots,\alpha^r_{k\ell}\right)$. En conséquence, nous avons :

$$\forall (k,\ell)\in\{1,\dots,g\}\times\{1,\dots,m\},\quad \sum_{h=1}^{r}\alpha^h_{k\ell} = 1 \text{ et } \forall h\in\{1,\dots,r\},\ \alpha^h_{k\ell}\in[0,1].$$

Pour simplifier la formule de la densité, nous introduisons le tableau binaire $(v_{ijh})_{n\times d\times r}$ tel que $v_{ijh} = 1$ si et seulement si $x_{ij} = h$. Ce tableau peut être vu comme une reproduction de la matrice $x_{ij}$ où n'est conservée que l'information sur la valeur $h$ (voir la figure 1.13).



$x_{ij}$ $v_{ij1}$ (rouge) $v_{ij2}$ (blanc) $v_{ij3}$ (bleu)

FIGURE 1.13 – Représentation d'une matrice avec trois valeurs (à gauche) puis ses représentations par la matrice $(v_{ijh})_{n\times d\times r}$ : $(v_{ij1})$ pour les rouges, $(v_{ij2})$ pour les blanches et $(v_{ij3})$ pour les bleues ; le noir représentant la présence et le gris l'absence des couleurs.

Avec ces notations, nous obtenons la formule suivante de la densité :

$$\varphi(x_{ij};\alpha_{k\ell}) = \prod_{h=1}^{r}\left(\alpha^h_{k\ell}\right)^{v_{ijh}}.$$

*Remarque* 1.3.2. *Cas de données binaires*
Pour le cas particulier de données binaires, nous prenons l'ensemble usuel $\{0,1\}$ et la loi de Bernoulli de paramètre $\alpha_{k\ell}$ avec $\alpha_{k\ell}\in[0,1]$. Dans ce cas, il n'est pas nécessaire d'utiliser la matrice $v_{ijh}$, la densité étant :

$$\varphi(x_{ij};\alpha_{k\ell}) = \alpha_{k\ell}^{x_{ij}}(1-\alpha_{k\ell})^{1-x_{ij}}.$$

$(H_2)$ L'indépendance des variables $Z$ et $W$ a pour conséquence que l'affectation de chaque ligne et colonne à une classe se fait également de façon indépendante.

$(H_3)$ Nous notons $\pi_k$ la probabilité d'appartenir à la $k$ ème classe en ligne :

$$\forall i \in \{1, \dots, n\}, \quad \mathbb{P}(Z_{ik} = 1) = \mathbb{P}(Z_i = k) = \pi_k$$

ou encore, comme les valeurs de $z_{ik}$ sont uniquement 0 ou 1, nous avons :

$$\forall i \in \{1, \dots, n\}, \quad p(z_{i\bullet}) = \prod_{k=1}^{g} \pi_k^{z_{ik}}$$

où $z_{i\bullet}$ représente la ligne de la matrice $\mathbf{z}$.

En notant $\boldsymbol{\pi} = (\pi_1, \dots, \pi_g)$ l'ensemble des paramètres $\pi_k$, l'hypothèse $(H_3)$ suppose que la loi de la variable $Z_i$ est une multinomiale $\mathcal{M}(1; \boldsymbol{\pi})$, ce qui est équivalent à une loi discrète sur $\{1, \dots, g\}$ de poids $\boldsymbol{\pi}$.

De même, nous notons $\rho_\ell$ la probabilité d'appartenir à la $\ell$ ème classe en colonne. Nous notons $\boldsymbol{\rho} = (\rho_1, \dots, \rho_m)$ l'ensemble des probabilités $\rho_\ell$.

Avec ces notations, nous obtenons finalement que :

$$p(\mathbf{z}, \mathbf{w}) = \left( \prod_{i=1}^{n} \prod_{k=1}^{g} \pi_k^{z_{ik}} \right) \left( \prod_{j=1}^{d} \prod_{\ell=1}^{m} \rho_\ell^{w_{j\ell}} \right).$$

**Notations 1.3.3.** Afin de ne pas alourdir les formules, nous omettons généralement les bornes pour les sommes et les produits. Ainsi, nous notons $\sum_i$ pour $\sum_{i=1}^{n}$. Nous renvoyons aux pages 16 et suivantes pour avoir des tableaux récapitulatifs et mettons ici l'ensemble des indices utilisés et les valeurs pouvant être prises (voir la figure 1.14, à gauche) :

— $i$ indice des lignes allant de 1 à $n$.
— $j$ indice des colonnes allant de 1 à $d$.
— $k$ indice des classes en ligne allant de 1 à $g$.
— $\ell$ indice des classes en colonne allant de 1 à $m$.
— $h$ indice des valeurs prises par les variables catégorielles allant de 1 à $r$.

Enfin, nous notons $\boldsymbol{\theta}$ l'ensemble de tous les paramètres à estimer (c'est-à-dire que $\boldsymbol{\theta} = (\boldsymbol{\pi}, \boldsymbol{\rho}, \boldsymbol{\alpha})$).

*Remarque* 1.3.4.
Le modèle de graphes aléatoires (SBM) est un modèle proche de celui des blocs latents, il y est régulièrement fait référence dans notre manuscrit. Leur différence fondamentale vient de l'hypothèse que $I = J$. Ainsi, la même partition sur l'ensemble $I$ est recherchée sur les lignes et les colonnes, occasionnant de nouvelles dépendances. Les paramètres à estimer sont uniquement $\boldsymbol{\pi}$ et $\boldsymbol{\alpha}$ (ce dernier étant supposé être représenté par une matrice symétrique lorsque les graphes ne sont pas orientés).

### 1.3.2 Résultats théoriques

Avant et durant notre thèse, certains résultats théoriques ont été démontrés. Nous les recensons dans cette partie ainsi que certaines des questions ouvertes.

#### 1.3.2.1 Identifiabilité

L'identifiabilité est nécessaire à la bonne estimation d'un modèle paramétrique. Rappelons qu'un modèle paramétrique est *identifiable* si la fonction $\theta \mapsto \mathbb{P}_\theta$ est injective sur l'ensemble $\Theta$ des paramètres. Un modèle est dit *génériquement identifiable* si la fonction est injective sur $\Theta$ privé d'un ensemble de mesure nulle.

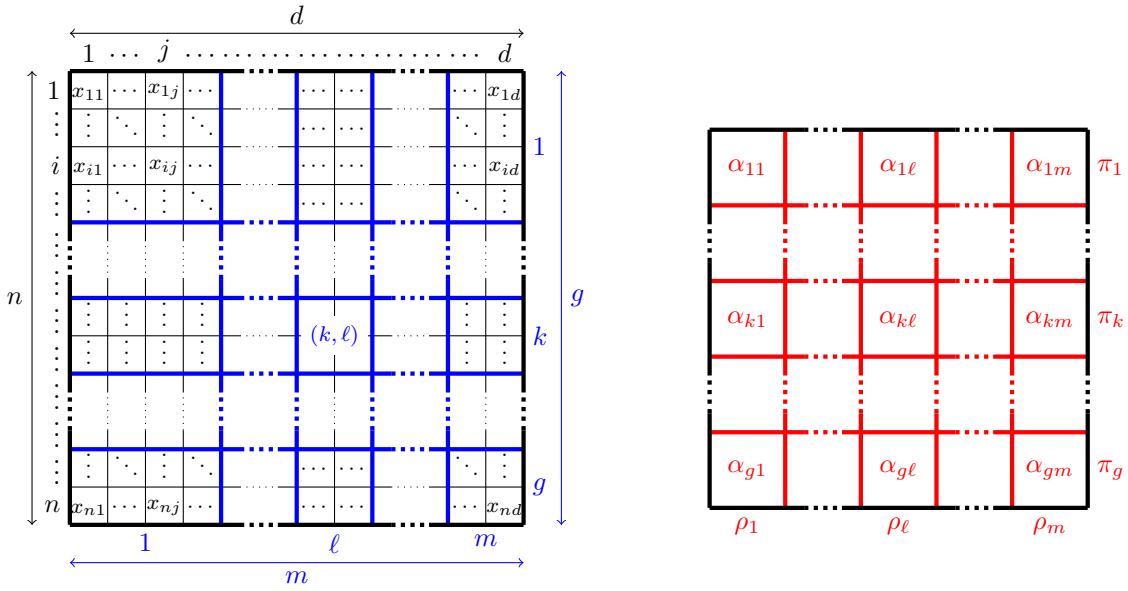



FIGURE 1.14 – Représentation schématique de tous les indices utilisés dans la suite. À gauche : en noir ceux concernant les cases et en bleu ceux concernant les blocs. À droite, les notations correspondant aux probabilités.

*Exemple* 1.3.5. Le modèle gaussien $\mathcal{N}(0,\theta)$ défini sur $\mathbb{R}$ par la fonction de densité $f_\theta(x) = \frac{1}{\sqrt{2\pi\theta^2}} e^{-\frac{1}{2\theta^2}x^2}$ n'est pas identifiable pour l'ensemble des paramètres $\Theta = \mathbb{R}^\star$ car deux paramètres $\theta$ et $-\theta$ donnent la même densité. En revanche, il l'est pour l'ensemble des paramètres strictement positifs $\Theta = \mathbb{R}_+^\star$ et celui des paramètres strictement négatifs $\Theta = \mathbb{R}_-^\star$.

De plus, le modèle est génériquement identifiable sur l'ensemble des paramètres $\Theta = \mathbb{R}_+^\star \cup \{-1\}$ ; en effet, l'ensemble $\{-1\}$ est de mesure de Lebesgue nulle.

Comme nous l'expliquons en section 1.2.3.3.1, le modèle des blocs latents est un modèle de mélange. Rappelons déjà que les modèles de mélange ne sont pas identifiables car il est possible d'intervertir les labels des classes sans changer la densité (cet effet est appelé *label switching*).

Toutefois, Gyllenberg et al. (1994) montrent que les modèles simples de mélange de Bernoulli multivariées ne sont pas identifiables indépendamment de ce problème de label switching. En revanche, Allman et al. (2009) donnent une condition suffisante pour assurer l'identifiabilité dans certains cas mais cette condition ne s'étend pas facilement au cas du modèle des blocs latents.

Dans le cadre du modèle des blocs latents, Keribin et al. (2014) étendent les résultats obtenus par Celisse et al. (2012) pour le modèle de graphes aléatoires afin d'avoir une condition suffisante d'identifiabilité dans le cas de données binaires et de contingence.

*Hypothèse* 1.3.6. **Conditions suffisantes d'identifiabilité**
Plaçons nous dans le cas de données binaires et rappelons que $\boldsymbol{\alpha} = (\alpha_{k\ell})_{g\times m}$ est la matrice $g \times m$ des paramètres des lois de Bernoulli. Les trois conditions supposent que :

$C_1$ : pour tout $1 \leq k \leq g$, $\pi_k > 0$ et les coordonnées du vecteur $\boldsymbol{\tau} = \boldsymbol{\alpha\rho}$ sont distinctes,
$C_2$ : pour tout $1 \leq \ell \leq m$, $\rho_\ell > 0$ et les coordonnées du vecteur $\boldsymbol{\sigma} = \boldsymbol{\pi}^\mathsf{T}\boldsymbol{\alpha}$ sont distinctes,
$C_3$ : $n \geq 2m-1$ et $d \geq 2g-1$.

La probabilité $\tau_k$, $k$ ème composante du vecteur $\boldsymbol{\tau}$, représente la probabilité d'une case d'une ligne de la classe $k$ de valoir 1 dans le cas binaire indépendamment des classes en colonne (voir la section 5.2). De même pour le vecteur $\boldsymbol{\sigma}$ sur les colonnes.

**Théorème 1.3.7.** ***Keribin et al. 2014***
*Sous les conditions $C_1$, $C_2$ et $C_3$, le modèle des blocs latents binaire est identifiable.*
*Sous la condition $C_3$, l'ensemble des paramètres ne vérifiant pas les conditions $C_1$ et $C_2$ étant de mesure de Lebesgue nulle, le modèle des blocs latents binaire est génériquement identifiable.*

*Remarque* 1.3.8.
Les conditions $C_1$ et $C_2$ sont la base de l'algorithme Largest Gaps étudié dans le chapitre 5.
La condition $C_3$ est naturelle : pour pouvoir différencier les classes en colonne, il est normal d'avoir un nombre suffisant d'observations (donc de lignes). En prenant le cas extrême où il n'y aurait qu'une seule ligne, il n'y aurait que deux types de colonnes : celles dont les cases valent 1 et celles dont les cases valent 0.

Dans le cas de deux classes en ligne et en colonne, Keribin et al. remarquent que les conditions $C_1$ et $C_2$ ne sont pas nécessaires pour obtenir l'identifiabilité (Keribin et al., 2014) ; il suffit simplement que les $\alpha_{k\ell}$ ne soient pas égaux. Empiriquement, ce résultat est rassurant car nous voyons sur la figure 1.15 un échiquier (à gauche) et une réorganisation en ayant d'abord mis les colonnes et lignes impaires puis les paires (à droite) ; dans les simulations, il apparaît que cette réorganisation en deux classes se fait naturellement alors qu'elle ne respecte ni la condition $C_1$, ni la condition $C_2$.

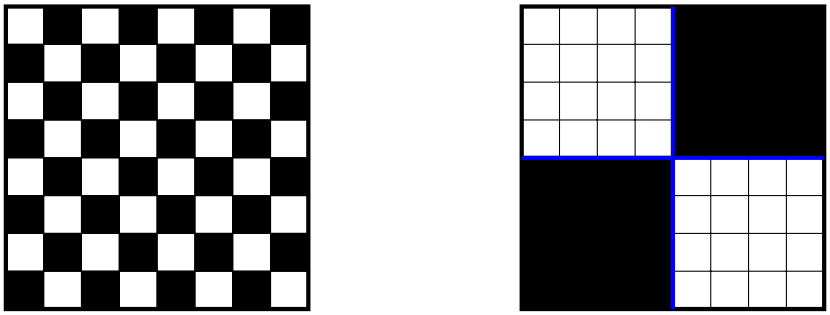


FIGURE 1.15 – L'exemple de l'échiquier : à gauche se trouve un échiquier et à droite le même en ayant d'abord mis les colonnes et lignes impaires puis les paires pour obtenir deux classes.

Les conditions $C_1$ et $C_2$ permettent de contourner le problème du label switching sur les paramètres. En effet, les composantes du vecteur $\boldsymbol{\tau} = (\tau_1, \ldots, \tau_g)$ appartenant chacune à $[0, 1]$ et étant toutes différentes par hypothèse, il est possible de trouver une permutation $\mathfrak{s}$ de l'ensemble $\{1, \ldots, g\}$ de telle sorte que :

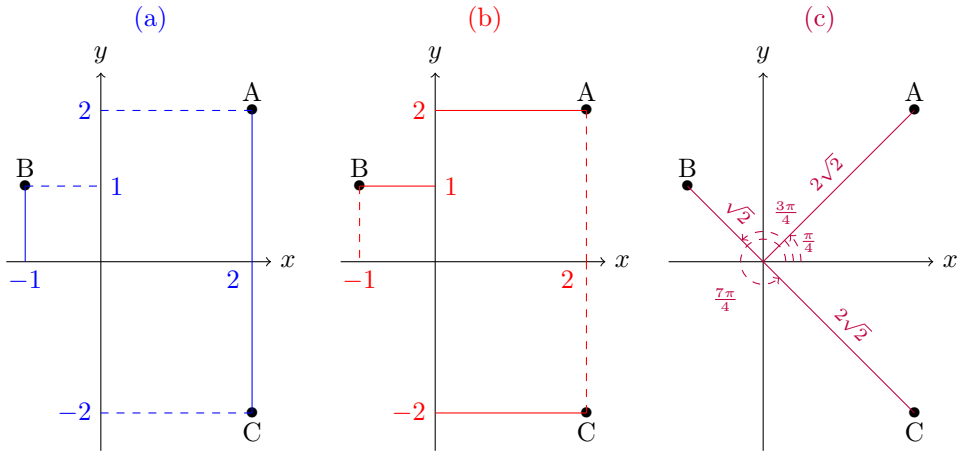
$$\tau_{\mathfrak{s}^{-1}(1)} < \tau_{\mathfrak{s}^{-1}(2)} < \ldots < \tau_{\mathfrak{s}^{-1}(g)}.$$

Ainsi, pour chaque classe $k$, nous affectons le label $\mathfrak{s}(k)$. Par symétrie, nous pouvons faire la même chose sur les colonnes avec le vecteur $\boldsymbol{\sigma} = (\sigma_1, \ldots, \sigma_m)$).
Comme nous sommes dans $\mathbb{R}$, la relation d'ordre obtenue est totale et la seule autre possibilité est de prendre l'ordre inverse du classement.
Toutefois, si les valeurs sont trop proches, de petites fluctuations peuvent intervertir l'estimation de deux probabilités ; ce problème peut influencer la qualité des résultats comme pour l'échantillonneur de *Gibbs* étudié dans le chapitre 3.

*Remarque* 1.3.9. **Cas de données catégorielles**
Le cas binaire est étendu au cas catégoriel en regardant les matrices $\boldsymbol{\alpha}^{(h)} = \left(\alpha_{k\ell}^{h}\right)_{g\times m}$ pour chaque valeur $h \in \{1,\dots,r\}$ (Keribin et al., 2014). Pour chaque classe $k$ en ligne, nous obtenons un vecteur $\tau_k = \left(\tau_k^1,\dots,\tau_k^r\right)$ appartenant à $[0,1]^r \subset \mathbb{R}^r$ (la dernière coordonnée étant totalement décrite par les $r-1$ premières). La relation d'ordre sur les vecteurs $\tau_k$ est plus compliquée car dans $\mathbb{R}^p$ avec $p \geq 2$, les relations d'ordre total ne sont pas équivalentes (voir l'exemple 1.3.10).
Ce problème peut être important pour l'échantillonneur de *Gibbs* étudié dans le chapitre 3 car, suivant les relations choisies, nous pouvons avoir des résultats très stables ou légèrement instables. Or, il n'existe pas de relations universelles et il faut procéder au cas par cas. Le choix retenu dans cette thèse est d'ordonner par rapport à la première coordonnée, puis, en cas d'égalité, par rapport à la seconde et ainsi de suite (voir la figure 1.16 (a)).

*Exemple* 1.3.10. Sur la figure 1.16 sont représentées schématiquement trois relations d'ordre total dans $\mathbb{R}^2$ :
- (a) l'ordre utilisé dans cette thèse regarde la première coordonnée, puis en cas d'égalité la seconde coordonnée,
- (b) même relation d'ordre mais en commençant par la seconde puis la première coordonnée,
- (c) relation d'ordre basée sur les coordonnées polaires : nous regardons d'abord la distance à l'origine puis, en cas d'égalité, l'angle formé entre la partie positive de l'axe des abscisses et la demi-droite partant de l'origine et passant par le point évalué.

Nous voyons qu'avec trois relations différentes, nous obtenons trois classements totalement différents dont aucun n'est le classement inverse d'un autre.
Dans le cas des classements $(a)$ et $(c)$, nous voyons qu'une légère fluctuation sur le point $A$ ou $C$ peut inverser l'ordre de ces points.




| | | | |
|---|---|---|---|
| Traits pleins | 1 ère coordonnée | 2 ème coordonnée | distance à l'origine |
| Traits pointillés | 2 ème coordonnée | 1 ère coordonnée | angle entre $[Ox)$ et $[OM]$ |
| Ordre obtenu | B, C puis A | C, B puis A | B, A puis C |

FIGURE 1.16 – Comparaison de trois relations d'ordre totales dans $\mathbb{R}^2$. Le trait plein est l'ordre prioritaire puis, en cas d'égalité, le trait en pointillé représente l'ordre secondaire.

#### 1.3.2.2 Vraisemblance

La vraisemblance du modèle des blocs latents s'écrit :

$$\begin{aligned} p(\mathbf{x};\boldsymbol{\theta}) &= \sum_{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W}} p(\mathbf{z},\mathbf{w};\boldsymbol{\theta})p(\mathbf{x}|\mathbf{z},\mathbf{w};\boldsymbol{\theta}) \\ &= \sum_{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W}} p(\mathbf{z};\boldsymbol{\theta})p(\mathbf{w};\boldsymbol{\theta})p(\mathbf{x}|\mathbf{z},\mathbf{w};\boldsymbol{\theta}) \qquad (H_2) \\ &= \sum_{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W}} p(\mathbf{z};\boldsymbol{\theta})p(\mathbf{w};\boldsymbol{\theta})\prod_{i,j} p(x_{ij}|\mathbf{z},\mathbf{w}) \qquad (H_1) \\ &= \sum_{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W}} \left(\prod_{i,k}\pi_k^{z_{ik}}\right)\left(\prod_{j,\ell}\rho_\ell^{w_{j\ell}}\right)\left(\prod_{i,j,k,\ell}\varphi(x_{ij};\alpha_{k\ell})^{z_{ik}w_{j\ell}}\right). \qquad (H_3) \end{aligned}$$

Nous retrouvons ainsi la forme de la vraisemblance d'un modèle de mélange ; voir l'équation (1.1). Toutefois et contrairement au cas des mélanges classiques, il n'est pas possible de factoriser les termes à cause de la dépendance simultanée en ligne et colonne. Or, le cardinal de $\mathcal{Z}\times\mathcal{W}$ est exactement $g^n \times m^d$ car pour chaque ligne (resp. colonne), nous avons $g$ classes (resp. $m$ classes) possibles. À titre d'exemple, en prenant $n = d = 20$ lignes et colonnes et $g = m = 2$ classes en ligne et en colonne, le cardinal est d'environ $10^{12}$ couples, soit en traitant 100 000 couples à la seconde, un calcul qui durerait un peu plus de 33 ans.

Ce problème est handicapant à deux égards : il rend impossible la recherche du maximum de vraisemblance et l'utilisation de critères de sélection de modèle pénalisant la log-vraisemblance (ce problème est abordé dans le chapitre 4).

#### 1.3.2.3 Régime asymptotique du modèle des blocs latents

Avant d'étudier le comportement asymptotique du modèle des blocs latents, il faut définir quel est le régime asymptotique. Se pose alors à nouveau la question de l'unité statistique ; pour la suite, nous supposons que c'est la case d'un tableau. Toutefois et si, par exemple, le nombre de colonnes est fixe alors que le nombre de lignes tend vers l'infini, nous ne sommes plus dans un problème de classification croisée mais de classification simple sur les lignes. À l'opposé, si le nombre de lignes et de colonnes croissent de la même façon vers l'infini (comme pour le cas du $SBM$), nous pouvons considérer être dans un régime asymptotique du modèle des blocs latents. Entre les deux se trouve toute une palette de possibilités.

Par ailleurs, certains auteurs (par exemple Channarond et al., 2012) incluent dans le comportement asymptotique le comportement des probabilités $\pi_k$ ou $\rho_\ell$ qui tendent vers 0 (par exemple, si le nombre de classes tend vers l'infini) ou encore lorsque les coordonnées des vecteurs $\boldsymbol{\tau}$ et $\boldsymbol{\sigma}$ tendent à être égales. Tout ceci est donc à prendre en compte lors de l'étude du comportement asymptotique des estimateurs et des critères de sélection de modèle.

*Remarque* 1.3.11. **Cas du SBM**
Dans le cadre du $SBM$, il n'y a pas ce problème de double asymptotique car les ensembles $I$ et $J$ sont égaux, seul le cardinal $n$ vers l'infini.

#### 1.3.2.4 Consistance des estimateurs asymptotiques

Pour l'instant, la consistance de l'estimateur du maximum de vraisemblance n'a pas été démontrée dans le cas du modèle des blocs latents. Dans le cadre du $SBM$, Celisse et al. (2012) montrent que si $\widehat{\boldsymbol{\theta}_n} = (\widehat{\boldsymbol{\pi}_n}, \widehat{\boldsymbol{\alpha}_n})$ est l'estimateur du maximum de vraisemblance et $\boldsymbol{\theta}^\star = (\boldsymbol{\pi}^\star, \boldsymbol{\alpha}^\star)$ les vrais paramètres alors $\widehat{\boldsymbol{\alpha}_n}$ est consistant sous les hypothèses suivantes :

(1) il n'y a pas deux lignes ou deux colonnes identiques dans la matrice $\boldsymbol{\alpha}^\star$,
(2) les probabilités de la matrice $\boldsymbol{\alpha}^\star$ appartiennent à un intervalle $[\zeta; 1-\zeta]$ avec $\zeta > 0$ indépendant des $\alpha_{k\ell}$,
(3) les probabilités du vecteur $\boldsymbol{\pi}^\star$ appartiennent également à un intervalle $[\gamma; 1-\gamma]$ avec $\gamma > 0$ indépendant des $\pi_k$.

Pour la consistance de $\widehat{\boldsymbol{\pi}_n}$, il faut rajouter l'hypothèse suivante :

(4) l'estimateur du maximum de vraisemblance $\widehat{\boldsymbol{\alpha}_n}$ de $\boldsymbol{\alpha}^\star$ vérifie :

$$||\widehat{\boldsymbol{\alpha}_n} - \boldsymbol{\alpha}^\star||_\infty = o_{\mathbb{P}}\left(\sqrt{\log n}/n\right).$$

Sous ces quatre conditions, l'estimateur du maximum de vraisemblance $\widehat{\boldsymbol{\theta}_n}$ est consistant.

*Remarque* 1.3.12. Les trois premières conditions sont relativement classiques, la dernière est plus contraignante. Sur des simulations, Celisse et al. (2012) illustrent empiriquement qu'elle n'est pas infirmée mais ne l'ont pas démontrée.

La consistance des estimateurs du maximum de vraisemblance pourrait être étendue au cas du modèle des blocs latents en adaptant la dernière condition au problème de la double asymptotique.

Pour l'estimation des classes, Mariadassou et Matias (2012) montrent dans le cadre du modèle des blocs latents que si $\widehat{\boldsymbol{\theta}_{n,d}}$ est un estimateur de $\boldsymbol{\theta}^\star$ tel que $\widehat{\boldsymbol{\alpha}_{n,d}}$ converge en probabilité vers le vrai paramètre $\boldsymbol{\alpha}^\star$ alors nous avons :

$$\mathbb{P}\left(Z = \mathbf{z}^\star, W = \mathbf{w}^\star \,\middle|\, x; \widehat{\boldsymbol{\theta}_{n,d}}\right) \underset{n,d}{\longrightarrow} 1 \tag{1.3}$$

où $\mathbf{z}^\star$ et $\mathbf{w}^\star$ sont les vraies partitions du modèle. Ceci a pour conséquence que si nous prenons :

$$\left(\widehat{\mathbf{z}_{n,d}}, \widehat{\mathbf{w}_{n,d}}\right) \in \operatorname{argmax}\ p\left(\mathbf{z}, \mathbf{w} \,\middle|\, x; \widehat{\boldsymbol{\theta}_{n,d}}\right)$$

alors, pour $n$ et $d$ suffisamment grands, la partition estimée est la bonne avec forte probabilité.

*Remarque* 1.3.13.
Ce résultat est utilisé dans le chapitre 4 pour étudier la consistance du critère $ICL$ et dans le chapitre 6 pour justifier l'étude de l'algorithme $CEM$.
Il est intéressant de remarquer que dans les deux résultats de consistance, la condition est mise sur les paramètres $\boldsymbol{\alpha}$ et non sur les paramètres $\boldsymbol{\pi}$ et $\boldsymbol{\rho}$. Or, comme nous le verrons dans le chapitre 5, l'estimation des $\alpha_{k\ell}$ dépend à la fois du nombre de lignes et du nombre de colonnes ce qui rend difficile l'étude asymptotique.

#### 1.3.2.5 Évaluation des résultats

Comme nous l'expliquons dans la section 1.2.6.1 et 1.2.6.2, il n'existe pas de consensus ni pour l'unité statistique, ni pour l'évaluation des résultats. Dans notre manuscrit, nous choisissons de prendre l'erreur de classification étudiée dans la thèse de Lomet (Lomet, 2012) regardant le taux de mauvais classement des cases dans les blocs. Pour cela, nous introduisons la *distance de classification* sur les lignes entre deux matrices de partition $\mathbf{z}$ et $\mathbf{z}'$ par :

$$\operatorname{dist}_{n,g}(\mathbf{z}; \mathbf{z}') = 1 - \max_{\mathfrak{s} \in \mathfrak{S}(\{1,\ldots,g\})} \frac{1}{n} \sum_{i,k} z_{ik} z'_{i\mathfrak{s}(k)}$$

où $\mathfrak{S}(\{1,\ldots,g\})$ représente l'ensemble des permutations possibles de l'ensemble $\{1,\ldots,g\}$ ; ces permutations permettent d'éviter le label switching. Dans le cas où les deux matrices à comparer n'ont pas

le même nombre de classes, nous prenons simplement $g$ le nombre maximal et nous supposons que les classes supplémentaires sont vides.

*Remarque numérique* 1.3.14.
Le cardinal de l'ensemble $\mathfrak{S}(\{1,\dots,g\})$ étant $g!$, le calcul numérique de cette distance devient difficile lorsque $g$ est plus grand que 8.
Dans ces cas, il est possible d'utiliser les conditions d'identifiabilité de Keribin pour réorganiser les classes et comparer les écarts sur les matrices obtenues. Pour cela, il suffit de prendre $\boldsymbol{\pi}^{(\mathbf{z})}$ et $\boldsymbol{\rho}^{(\mathbf{w})}$ comme les fréquences empiriques de chaque classe des matrices $\mathbf{z}$ et $\mathbf{w}$ et $\boldsymbol{\alpha}^{(\mathbf{z},\mathbf{w})}$ comme les fréquences empiriques des cases noires de chaque bloc pour le cas binaire (voir section 5.4).

*Remarque* 1.3.15.
Le terme de "distance" peut-être considéré comme abusif car ce n'est pas une distance sur l'ensemble $\mathcal{Z}$. En revanche, en prenant la relation d'équivalence $\equiv_{\mathcal{Z}}$ où deux matrices de partitions binaires $z^1$ et $z^2$ sont équivalentes s'il existe une permutation $\mathfrak{s}$ des éléments de $\{1,\dots,g\}$ telle que pour tout $i$ et $k$, $z^1_{ik} = z^2_{i\mathfrak{s}(k)}$ alors, la fonction $\mathrm{dist}_{n,g}$ est une distance sur l'ensemble quotient $\mathcal{Z}/_{\equiv_{\mathcal{Z}}}$, l'ensemble des matrices de partitions égales au label switching près.

De manière symétrique, nous définissons la distance de classification sur les colonnes notée $\mathrm{dist}_{d,m}$. Enfin, la distance de classification entre deux couples $(\mathbf{z},\mathbf{w})$ et $(\mathbf{z}',\mathbf{w}')$ est définie par :

$$\mathrm{dist}_{(n,g)\times(d,m)}\left((\mathbf{z},\mathbf{w});(\mathbf{z}',\mathbf{w}')\right) = \mathrm{dist}_{n,g}\left(\mathbf{z};\mathbf{z}'\right) + \mathrm{dist}_{d,m}\left(\mathbf{w};\mathbf{w}'\right) - \mathrm{dist}_{n,g}\left(\mathbf{z};\mathbf{z}'\right) \times \mathrm{dist}_{d,m}\left(\mathbf{w};\mathbf{w}'\right).$$

Nous parlons d'*erreur de classification* d'un couple $(\mathbf{z},\mathbf{w})$ lorsque nous calculons la distance de classification de celui-ci avec les vraies partitions $(\mathbf{z}^\star,\mathbf{w}^\star)$ :

$$e_{(n,g)\times(d,m)}\left((\mathbf{z},\mathbf{w})\right) = \mathrm{dist}_{(n,g)\times(d,m)}\left((\mathbf{z},\mathbf{w});(\mathbf{z}^\star,\mathbf{w}^\star)\right). \tag{1.4}$$

*Remarque* 1.3.16.
L'erreur est comprise entre 0 et 1 et peut être vue comme le nombre moyen de cases n'appartenant pas au bon bloc. En particulier, la case $x_{ij}$ n'est pas dans le bloc $(k,\ell)$ si la ligne $i$ n'est pas dans la classe $k$ ou si la colonne $j$ n'est pas dans la classe $\ell$. Se tromper dans la classification d'une ligne revient donc à pénaliser toutes les cases de cette dernière (et d'augmenter potentiellement l'erreur de $\frac{1}{d}$).
Dans le cas de matrices fortement dissymétriques, l'erreur ne donne donc pas le même poids aux classifications en ligne et en colonne (voir la figure 1.17).

**Propriété 1.3.17.** ***Borne maximale sur l'erreur***
*À cause des permutations, si le nombre de lignes et de colonnes est fixé et s'il n'y a aucune classe vide, la borne maximale de l'erreur sur les lignes (resp. sur les colonnes) dépend du nombre de classes demandé.*

En particulier, nous indiquons dans le tableau 1.5 les bornes maximales $e^{\max}_{n,g}$ de l'erreur sur les lignes en fonction de $g$ et de $n$ dont nous avons mis les démonstrations en annexe 1.B. Nous pouvons notamment voir que la fonction $e^{\max}_{n,g}$ en fonction de $g$ est d'abord croissante puis décroissante.
Cette borne maximale a un impact sur des protocoles comme celui proposé par Lomet et al. (2012c).

Parfois, nous utilisons l'erreur en 0-1 notée $e^{0-1}_{(n,g)\times(d,m)}$ qui associe 0 si et seulement si la partition obtenue est correcte au label switching près et 1 dans tous les autres cas (Lomet et al., 2012c). Autrement dit, nous notons pour le classement des blocs :

$$e^{0-1}_{(n,g)\times(d,m)}\left((\mathbf{z},\mathbf{w})\right) = \mathbb{1}_{e_{(n,g)\times(d,m)}((\mathbf{z},\mathbf{w}))>0}.$$

Nous avons le même type de formule pour les déclinaisons sur le classement des lignes et celui des colonnes. Cette erreur est beaucoup plus contraignante que l'autre puisqu'une seule ligne ou colonne mal classée renvoie la valeur maximale.

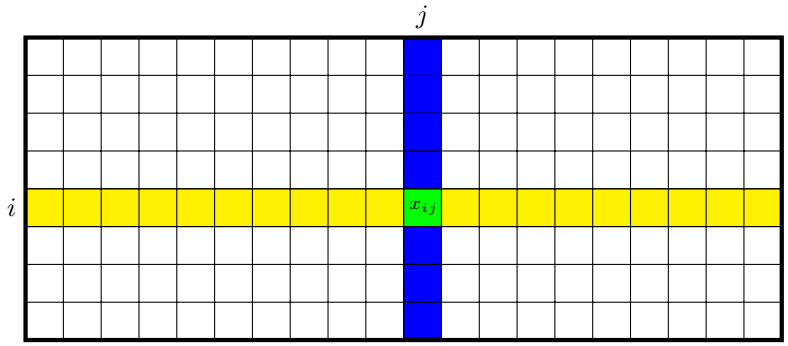



FIGURE 1.17 – Représentation schématique de l'erreur $e_{(n,g)\times(d,m)}$ définie par l'équation (1.4). Si la ligne $i$ (en jaune) est mal classée, cela représente 20 cases soit une proportion de $\frac{1}{8}$ de la matrice et si c'est la colonne $j$ (en bleu), cela représente 8 cases soit une proportion de $\frac{1}{20}$. Dans le cas où la ligne et la colonne sont mal classées, la case $x_{ij}$ (en vert) est comptée deux fois, d'où l'importance d'enlever le produit des deux erreurs.

| $g$ | 1 | 2 | 3 | ... | $g$ ($g^2 \leq n$) | ... | $n - g_{bis}$ ($2g_{bis} \leq n-2$) | ... | $n-2$ | $n-1$ | $n$ |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $e^{\max}_{n,g}$ | 0 | $\frac{1}{2}$ | $\frac{2}{3}$ | ... | $\frac{g-1}{g}$ | ... | $\frac{2(g_{bis}-1)}{n}$ | ... | $\frac{4}{n}$ | $\frac{2}{n}$ | 0 |

TABLE 1.5 – Borne maximale de l'erreur sur les lignes suivant le nombre de classes demandé.

### 1.3.3 Plan de la thèse

Dans le chapitre 2, nous expliquons pourquoi l'algorithme *Expectation Maximisation* ne peut pas être utilisé tel quel pour le modèle des blocs latents. Nous étudions ensuite les algorithmes *VEM* (proposé par Govaert et Nadif (2008)) et *SEM-Gibbs* (proposé par Keribin et al. (2014)) et plus particulièrement la combinaison de ces deux algorithmes. Nous expliquons les difficultés rencontrées, notamment le problème de dégénérescence de classes.

Dans le chapitre 3, nous proposons une adaptation bayésienne du modèle pour répondre à cette question de dégénérescence. Nous montrons en particulier qu'avec des lois a priori correctement choisies, l'algorithme *V-Bayes*, version bayésienne de l'algorithme *VEM*, régularise l'estimation des classes. Nous étudions également l'échantillonneur de *Gibbs* et proposons un critère d'arrêt basé sur la statistique de Brooks-Gelman (Brooks et Gelman, 1998) et l'adaptation proposée par Fu (2012).

Dans le chapitre 4, nous étudions le critère de sélection de modèle *ICL* (pour *Integrated Completed Likelihood*) et montrons qu'une formule exacte est possible. Nous utilisons alors une approximation de ce critère pour déduire un critère *BIC* (pour *Bayesian Information Criterion*). Nous étudions ensuite leurs comportements. Dans un deuxième temps, nous adaptons les critères pour quantifier la perte d'information d'une binarisation des données.

Dans le chapitre 5, nous étudions une adaptation de l'algorithme *Largest Gaps* proposé par Channa-

rond et al. (2012) dans le cadre des modèles de graphes aléatoires. Nous démontrons que les estimateurs fournis par cet algorithme sont consistants mais qu'il faut un grand nombre d'observation pour obtenir de bonnes estimations. Ce chapitre permet, entre autres, de mieux appréhender le problème de la double asymptotique.

Dans le chapitre 6, nous comparons tous les algorithmes décrits dans les chapitres précédents sur des données simulées et réelles. En particulier, nous regardons les procédures à partir d'un temps d'exécution (*elapsed*) fixé.

Enfin, nous concluons en rappelant les résultats obtenus, proposons une méthodologie et énonçons les problèmes encore ouverts que nous jugeons importants d'étudier.

## 1.A Tableau récapitulatif

Pour une vue plus globale, nous proposons un tableau récapitulatif mais non exhaustif de différents algorithmes existant :

| Nom | Section | Données | Critère | Classification | Applications | Référence |
|---|---|---|---|---|---|---|
| Collapsed sampler | 1.2.3 | Tout type | Probabiliste | Simultanée | Cadre général, génétique, sociologie | Wyse et Friel (2010) |
| Crobin | 1.2.3 | Binaires | Déterministe | Simultanée | Cadre général, sociologie | Govaert (1983) |
| Croki2 | 1.2.3 | Contingence | Déterministe | Simultanée | Géographie, données textuelles | Govaert (1983) |
| Croeuc | 1.2.3 | Continues | Déterministe | Simultanée | Cadre général, données de recommandation, écologie | Govaert (1983) |
| NBVD | 1.2.3 | Tout type | Déterministe | Simultanée | Données textuelles | Long et al. (2005) |
| NMF | 1.2.3 | Tout type | Déterministe | Simultanée | Cadre général | Seung et Lee (2001) |
| ONMF | 1.2.3 | Tout type | Déterministe | Simultanée | Données textuelles | Yoo et Choi (2010) |
| SEM | 1.2.3 | Tout type | Probabiliste | Simultanée | Cadre général, génétique, sociologie, données textuelles, données de recommandation | Keribin et al. (2010) |
| VEM, CEM | 1.2.3 | Tout type | Probabiliste | Simultanée | Cadre général, génétique, sociologie, données textuelles, données de recommandation | Govaert et Nadif (2007, 2008, 2009) |
| V-Bayes | 1.2.3 | Tout type | Probabiliste | Simultanée | Cadre général, génétique, sociologie | Keribin et al. (2012, 2014) |

| Nom | Section | Données | Critère | Classification | Applications | Référence |
|---|---|---|---|---|---|---|
| Brute-Force Deletion and Addition | 1.2.5 | Tout type | Déterministe | Simultanée | Génétique | Cheng et Church (1999) |
| Finding a Given Number of Biclusters | 1.2.5 | Tout type | Déterministe | Simultanée | Génétique | Cheng et Church (1999) |
| FLOC | 1.2.5 | Tout type | Déterministe | Simultanée | Génétique | Yang et al. (2002) |
| ISA (Iterative Signature Algorithm) | 1.2.5 | Binaire | Déterministe | Alternée | Génétique | Ihmels et al. (2004) |
| Lazzeroni and Owen | 1.2.5 | Tout type | Probabiliste | Simultanée | Génétique, biologie, économie | Lazzeroni et Owen (2000) |
| Multiple Node Deletion | 1.2.5 | Tout type | Déterministe | Simultanée | Génétique | Cheng et Church (1999) |
| Node Addition | 1.2.5 | Tout type | Déterministe | Simultanée | Génétique | Cheng et Church (1999) |
| One-Way splitting | 1.2.4 | Tout type | Déterministe | Alternée | Sociologie | Hartigan (1975) |
| OPSM | 1.2.5 | Ordonnées | Probabiliste | Simultanée | Génétique | Ben-dor et al. (2002) |
| Permutation-Based Block Clustering | 1.2.4 | Tout type | Déterministe | Alternée | Sociologie | Duffy et Quiroz (1991) |
| Ping-Pong | 1.2.5 | Binaires | Déterministe | Simultanée | Marketing | Oyanagi et al. (2001) |
| Processus Mondrain | 1.2.4 | Multinomiale | Probabiliste | Simultanée | Génétique, Sociologie | Roy et Teh (2008) |
| Samba | 1.2.5 | Binaires | Probabiliste | Simultanée | Génétique | Tanay et al. (2002) |
| Single Node Deletion | 1.2.5 | Tout type | Déterministe | Simultanée | Génétique | Cheng et Church (1999) |
| Two-Way splitting | 1.2.4 | Tout type | Déterministe | Alternée | Sociologie | Hartigan (1975) |
| xMotif | 1.2.5 | Binaire | Probabiliste | Simultanée | Génétique | Murali et Kasif (2003) |

## 1.B Borne de l'erreur

Comme expliqué dans la section 1.3.2.5, à cause des permutations, l'erreur $e_{n,g}$ sur les lignes ne peut pas dépasser une certaine valeur qui est strictement inférieure à 1. Prenons l'exemple de trois lignes avec deux classes non vides et prenons toutes les combinaisons possibles entre la vraie partition $\mathbf{z}^\star$ et celle que nous pouvons obtenir $\mathbf{z}$ (voir le tableau 1.6). Sans les permutations, l'erreur va de 0 à 1 mais avec les permutations, elle ne dépasse pas 1/3.

**Erreur sans les permutations**

| $\mathbf{z}^\star$ \ $\mathbf{z}$ | $\begin{pmatrix}1\\2\\2\end{pmatrix}$ | $\begin{pmatrix}1\\1\\2\end{pmatrix}$ | $\begin{pmatrix}1\\2\\1\end{pmatrix}$ | $\begin{pmatrix}2\\1\\2\end{pmatrix}$ | $\begin{pmatrix}2\\1\\1\end{pmatrix}$ | $\begin{pmatrix}2\\2\\1\end{pmatrix}$ |
|---|---|---|---|---|---|---|
| $(1\ 2\ 2)^\intercal$ | 0 | 1/3 | 1/3 | 2/3 | 1 | 2/3 |
| $(1\ 1\ 2)^\intercal$ | 1/3 | 0 | 2/3 | 1/3 | 2/3 | 1 |
| $(1\ 2\ 1)^\intercal$ | 1/3 | 2/3 | 0 | 1 | 2/3 | 1/3 |
| $(2\ 1\ 2)^\intercal$ | 2/3 | 1/3 | 1 | 0 | 1/3 | 2/3 |
| $(2\ 1\ 1)^\intercal$ | 1 | 2/3 | 2/3 | 1/3 | 0 | 1/3 |
| $(2\ 2\ 1)^\intercal$ | 2/3 | 1 | 1/3 | 2/3 | 1/3 | 0 |

**Erreur avec les permutations**

| $\mathbf{z}^\star$ \ $\mathbf{z}$ | $\begin{pmatrix}1\\2\\2\end{pmatrix}$ | $\begin{pmatrix}1\\1\\2\end{pmatrix}$ | $\begin{pmatrix}1\\2\\1\end{pmatrix}$ | $\begin{pmatrix}2\\1\\2\end{pmatrix}$ | $\begin{pmatrix}2\\1\\1\end{pmatrix}$ | $\begin{pmatrix}2\\2\\1\end{pmatrix}$ |
|---|---|---|---|---|---|---|
| $(1\ 2\ 2)^\intercal$ | 0 | 1/3 | 1/3 | 1/3 | 0 | 1/3 |
| $(1\ 1\ 2)^\intercal$ | 1/3 | 0 | 1/3 | 1/3 | 1/3 | 0 |
| $(1\ 2\ 1)^\intercal$ | 1/3 | 1/3 | 0 | 0 | 1/3 | 1/3 |
| $(2\ 1\ 2)^\intercal$ | 1/3 | 1/3 | 0 | 0 | 1/3 | 1/3 |
| $(2\ 1\ 1)^\intercal$ | 0 | 1/3 | 1/3 | 1/3 | 0 | 1/3 |
| $(2\ 2\ 1)^\intercal$ | 1/3 | 0 | 1/3 | 1/3 | 1/3 | 0 |

TABLE 1.6 – Borne maximale de l'erreur sur les lignes pour trois lignes et deux classes non vides : en haut sans les permutations et en bas avec les permutations, les chiffres en rouge signifie qu'il faut inverser les labels pour minimiser l'erreur.

Durant notre thèse, nous avons démontré les bornes maximales pour les cas du tableau 1.5. Nous mettons dans cette annexe les démonstrations.

Pour les démonstrations des bornes du tableau 1.5, nous commençons par les plus grands nombres de classes et finissons par les plus petits. Les premières démonstrations se font en regardant les constructions possibles des partitions ; les secondes en analysant les effectifs possibles en commun entre les différentes classes. Ces démonstrations présupposent qu'il y ait un nombre de lignes suffisant et qu'aucune classe ne soit vide ; nous notons $\mathbf{z}$ et $\mathbf{z}^\star$ les deux partitions.

$g = n$ **:** pour le cas où il y a autant de classes en ligne que de lignes, il y a exactement une ligne par

classe et il suffit de prendre la permutation qui associe les classes telle que les cases soient dans la même classe. L'erreur est alors de 0.

$g = n - 1$ **:** pour le cas où il y a une classe de moins que de lignes, cela signifie qu'il y a exactement une classe de deux lignes (car il n'y a aucune classes vides) :

— soit les deux classements possèdent les mêmes lignes dans leurs classes respectives de cardinal 2, et dans ce cas, en associant les deux classes par une permutation, nous avons une erreur de 0,

— soit elles contiennent des lignes différentes et, sans perte de généralité, nous supposons que ce sont les lignes 1 et 2 pour $\mathbf{z}$ et les lignes 3 et 4 pour $\mathbf{z}^\star$. Alors, nous associons le grand groupe de $\mathbf{z}$ au groupe contenant uniquement la ligne 1 de $\mathbf{z}^\star$ ; le groupe ne contenant que la ligne 3 de $\mathbf{z}$ au grand groupe de $\mathbf{z}^\star$ et le groupe ne contenant que la ligne 4 de $\mathbf{z}$ avec le groupe de $\mathbf{z}^\star$ ne contenant que la ligne 2. Les autres groupes étant des singletons, nous faisons la même association que précédemment. En mettant le même label lorsque les groupes sont égaux par la permutation, nous obtenons finalement :

$$\mathbf{z} = \begin{pmatrix} 1 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ \vdots \\ n-1 \end{pmatrix} \quad \text{et} \quad \mathbf{z}^\star = \begin{pmatrix} 1 \\ 3 \\ 2 \\ 2 \\ 4 \\ 5 \\ \vdots \\ n-1 \end{pmatrix}$$

soit deux lignes de mal classées.

— soit elles ont une ligne en commun et en les associant, deux lignes sont mal classées,

D'où une erreur maximale de $2/n$.

$g = n - g_{bis}$ **:** en reprenant la démonstration précédente et si, pour chaque partition, il y a un groupe contenant $g_{bis} + 1$ lignes et que ces groupes sont totalement différents, alors il y a au moins $g_{bis}$ lignes de mal classées pour chaque cas soit $2g_{bis}$ lignes mal classées. Si leur intersection est non vide, nous associons les deux groupes et l'erreur est au pire la même. Ceci n'est possible que si $2(g_{bis} + 1) \leq n$.

Montrons maintenant que la borne ne peut pas dépasser cette valeur (voir la figure 1.18) : pour former les groupes, nous prenons $n - g_{bis}$ lignes formant des singletons (groupes 1 et 2 pour la matrice $\mathbf{z}$ et groupes 2 et 4 pour la matrice $\mathbf{z}^\star$ sur la figure 1.18) puis nous rajoutons dans un second temps à ces groupes les $g_{bis}$ autres lignes (groupe 3 et 4 pour la matrice $\mathbf{z}$ et groupes 1 et 3 pour la matrice $\mathbf{z}^\star$ sur la figure 1.18). Nous divisons ainsi le nombre de lignes en 4 groupes (voir la figure 1.18) :

1. les lignes sélectionnées initialement pour former les groupes pour la partition $\mathbf{z}$ mais pas pour la partition $\mathbf{z}^\star$,
2. les lignes sélectionnées initialement pour former les groupes pour les deux partitions,
3. les lignes qui ne sont sélectionnées initialement par aucune partition,
4. les lignes sélectionnées initialement pour former les groupes pour la partition $\mathbf{z}^\star$ mais pas pour la partition $\mathbf{z}$.

En notant $n_i$ l'effectif du groupe $i$, nous pouvons faire plusieurs constatations : comme le nombre de points mis à l'écart est de $g_{bis}$, nous avons pour la partition $\mathbf{z}$, la formule $n_3 + n_4 = g_{bis}$ et $n_1 + n_4 = g_{bis}$ pour la partition $\mathbf{z}^\star$. Nous en déduisons que $n_1 = n_4$.

Pour le classement, nous associons les singletons des deux classement du groupe 2 de façon à ce que chaque ligne de ce groupe soit dans la même classe pour les deux partitions. Pour les autres, nous associons les singletons de la partition $\mathbf{z}$ correspondant au groupe 1 avec ceux de la partition $\mathbf{z}^\star$ du groupe 3. Ainsi,

nous avons $n_1 + n_4$ lignes mal classées, $n_3$ potentiellement mal classées et $n_2$ bien classées. À partir des résultats précédents, nous avons :

$$n_2 = n - (n_1 + n_3 + n_4) = n - [n_1 + (g_{bis} - n_1) + n_1] = n - g_{bis} - n_1$$

qui est minimale pour le cas où $n_1 = g_{bis}$.
Ce raisonnement n'est possible que si le nombre $g_{bis}$ est inférieur ou égal à la moitié de $n$.

Nous venons de montrer que nous ne pouvons pas faire pire qu'une erreur de $\frac{2g_{bis}}{n}$ et nous avons un cas où cette erreur est atteinte si $2g_{bis} \leq n - 2$ d'où le résultat.

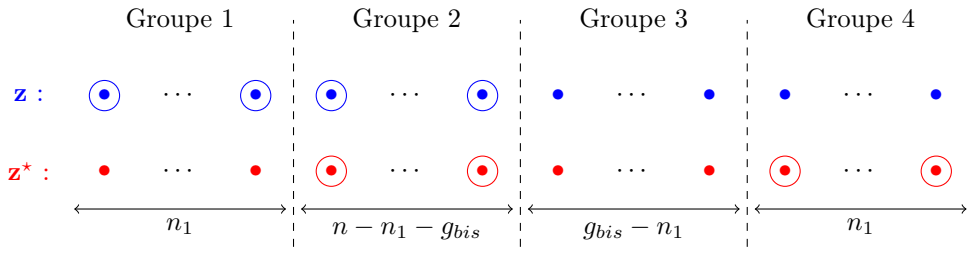



FIGURE 1.18 – Représentation schématique de la construction des classes pour le cas où $g = n - g_{bis}$. Les matrices sont représentées horizontalement, les points de chaque colonne représentent le même indice de ligne (celui du haut pour le classement de $\mathbf{z}$ et celui du bas pour le classement de $\mathbf{z}^\star$). Les points entourés d'un cercle sont ceux sélectionnés initialement pour créer les singletons. Le groupe 1 (resp. 4) englobe les lignes sélectionnées pour créer les singletons initiaux pour faire la partition $\mathbf{z}$ (resp. $\mathbf{z}^\star$) mais qui ne sont pas sélectionnées pour celle de $\mathbf{z}^\star$ (resp. $\mathbf{z}$) ; le groupe 2, les lignes initiales en commun et le groupe 3, les lignes qui ne sont sélectionnées par aucune partition au départ.

A l'opposé, si nous avons peu de groupes :

$g = 1$ **:** c'est forcément le même groupe de chaque côté.

$g = 2$ **:** en prenant les deux groupes de la partition $\mathbf{z}$, nous avons :
— dans le premier groupe, les lignes qui sont dans le groupe 1 de $\mathbf{z}^\star$ soit un effectif de $n_{11}$ et les lignes qui sont dans le deuxième groupe de $\mathbf{z}^\star$ soit un effectif de $n_{12}$,
— de même pour le deuxième groupe avec des effectifs de $n_{21}$ et $n_{22}$.
Sans perte de généralité, supposons que la permutation optimale est l'identité et regardons ce que cela implique sur les effectifs des classements. Nous voyons que le nombre de lignes bien classées est alors de $n_{11} + n_{22}$ et que celui de lignes mal classées est de $n_{12} + n_{21}$.
Si nous supposons que $n_{12} + n_{21} > n_{11} + n_{22}$ alors, la permutation associant les classes 1 et 2 de chaque partition est meilleure et nous contredisons l'hypothèse que l'identité est la permutation optimale. Ceci implique que $n_{12} + n_{21} \leq n_{11} + n_{22}$ et pour avoir le plus grand nombre de lignes mal classées, il faut avoir l'égalité.
Comme la somme est censée faire $n$, nous obtenons une borne maximale de l'erreur de $1/2$ (voir la figure 1.19 en haut).

$g = 3$ **:** Avec les mêmes notations et toujours sans perte de généralité, nous supposons que l'identité est la permutation optimale. Par le même raisonnement que pour $g = 2$, si pour deux classes $k$ et $k'$ nous avons $n_{kk'} + n_{k'k} > n_{kk} + n_{k'k'}$ alors la permutation qui inverse ces deux classes ne touche pas aux autres

effectifs et classe mieux plus de lignes. Par conséquent, le pire cas est quand nous avons pour tout couple de classes $(k, k')$, la relation $n_{kk'} + n_{k'k} = n_{kk} + n_{k'k'}$. Ce qui implique que :

$$\begin{cases} n_{11} + n_{22} &= n_{12} + n_{21} \\ n_{11} + n_{33} &= n_{13} + n_{31} \\ n_{22} + n_{33} &= n_{23} + n_{32} \end{cases}$$

Or, le nombre de lignes mal classées, que nous notons $n_{MC}$, est la somme des termes de droites. Donc, en notant $n_{BC}$ le nombre de lignes bien classées, nous avons :

$$n_{MC} = n_{12} + n_{21} + n_{13} + n_{31} + n_{23} + n_{32} = 2\left(n_{11} + n_{22} + n_{33}\right) = 2n_{BC}$$

et comme la somme de $n_{BC}$ et de $n_{MC}$ vaut $n$, nous en déduisons que le nombre maximal de lignes mal classées est de $2/3n$ soit une erreur maximale de $2/3$ (voir la figure 1.19 en bas).

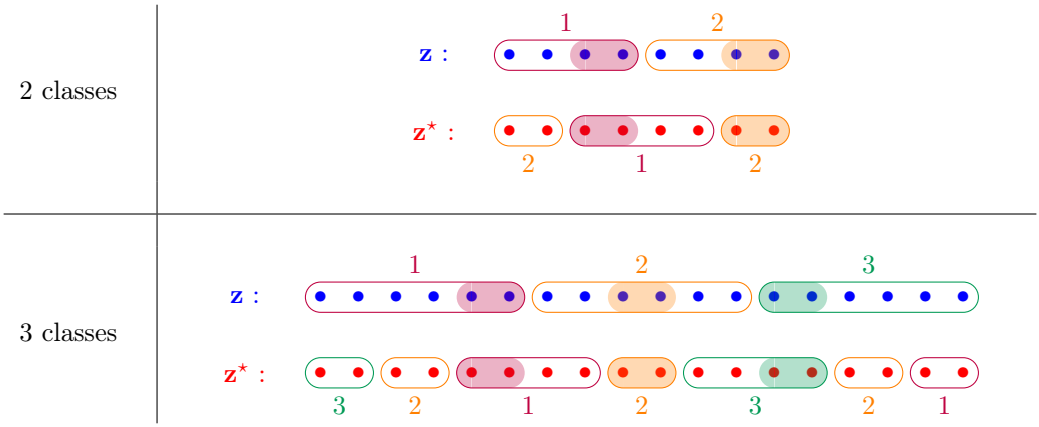



FIGURE 1.19 – Représentation schématique de la construction des classes pour le cas où $g^2 \leq n$ pour 2 et 3 classes ayant à chaque fois 2 points en commun pour chaque classe. Les matrices sont représentées horizontalement, les points de chaque colonne représentent le même indice de ligne et les groupes entourant les points correspondent aux classes (bordeaux pour la 1, orange pour la 2 et vert pour la 3). Les points surlignés sont ceux dans la même classe pour chacune des partitions.

$g^2 \leq n$ **:** en reprenant le raisonnement précédent, nous obtenons les relations suivantes :

$$\forall k \in \{1, \ldots, g-1\}, \quad \forall k' \in \{k+1, \ldots, g\}, \quad n_{kk} + n_{k'k'} = n_{kk'} + n_{k'k}. \tag{1.5}$$

De même, nous avons alors :

$$\begin{aligned}
n_{MC} &= \sum_{k=1}^{g-1}\sum_{k'=k+1}^{g}(n_{kk'}+n_{k'k}) \\
&= \sum_{k=1}^{g-1}\sum_{k'=k+1}^{g} n_{kk} + \sum_{k=1}^{g-1}\sum_{k'=k+1}^{g} n_{k'k'} \\
&= \sum_{k=1}^{g-1}(g-k)\,n_{kk} + \sum_{k'=2}^{g}\sum_{k=1}^{k'-1} n_{k'k'} \\
&= \sum_{k=1}^{g-1}(g-k)\,n_{kk} + \sum_{k'=2}^{g}(k'-1)\,n_{k'k'} \\
&= (g-1)\,n_{11} + \sum_{k=2}^{g-1}[(g-k)\,n_{kk} + (k-1)\,n_{kk}] + (g-1)\,n_{gg} \\
&= (g-1)\,n_{11} + \sum_{k=2}^{g-1}[(g-1)\,n_{kk}] + (g-1)\,n_{gg} \\
&= (g-1)\sum_{k=1}^{g} n_{kk} \\
&= (g-1)\,n_{BC}.
\end{aligned}$$

Et comme nous avons la somme qui vaut $n$, nous obtenons :

$$n_{MC} + n_{BC} = n \Leftrightarrow (g-1)\,n_{BC} + n_{BC} = n \Leftrightarrow n_{BC} = \frac{n}{g}$$

et la borne maximale est de $\frac{g-1}{g}$.
Pour cela, il faut au moins que chaque effectif $n_{kk'}$ soit plus grand que 1. Nous avons donc besoin que $g^2 \leq n$. De plus, cette borne n'est qu'un majorant dans le cas où nous n'avons pas les égalités de (1.5) mais seulement des inégalités.

Entre les deux états limites, il est possible de prendre la borne obtenue dans le cas où $g^2 \leq n$. Cette borne est toutefois trop grande car il n'y a plus la possibilité d'avoir toutes les égalités de 1.5 mais un certain nombre d'inégalités strictes.

# Chapitre 2

# Algorithmes dérivés de l'algorithme *EM*

*"Rien ne semble vrai qui ne puisse sembler faux."*
Montaigne

**Sommaire**


**2.1 Introduction . . . . . 62**
**2.2 Algorithme *Expectation Maximisation* (EM) . . . . . 63**
**2.3 Algorithme *VEM* . . . . . 64**
**2.4 Algorithme *SEM-Gibbs* . . . . . 67**
**2.5 Algorithme *SEM+VEM* . . . . . 69**
**2.6 Dégénérescence des classes . . . . . 70**
2.6.1 Simulations . . . . . 70
2.6.2 Cause de cette dégénérescence . . . . . 76
**2.7 Conclusion . . . . . 79**
**2.A États absorbants de type II pour l'algorithme *SEM-Gibbs* . . . . . 80**


## 2.1 Introduction

Comme nous l'expliquons dans le chapitre 1, le modèle des blocs latents peut être vu comme un modèle de mélange. Dans ce chapitre, nous expliquons pourquoi il est impossible, dans un temps raisonnable, d'utiliser l'algorithme *EM* (Dempster et al., 1977) puis nous étudions deux algorithmes dérivés de ce premier pour contourner cette difficulté.

Pour cela, nous présentons d'abord l'approximation variationnelle qui est à la base de l'algorithme *VEM* de Govaert et Nadif (2008), nous expliquons le principe de l'algorithme et présentons quelques limites de celui-ci. Notamment, nous parlons du problème de dégénérescence des classes, terme utilisé dans le cas où un algorithme renvoie un nombre de classes inférieur à celui demandé.

Ensuite, nous présentons l'algorithme *SEM-Gibbs* introduit par Keribin et al. (2014) qui est une adaptation au modèle des blocs latents de l'algorithme $SEM$ (pour *Stochastic Expectation Maximisation*) proposé par Celeux et al. (1995) dans le cas des mélanges classiques. Nous rappelons quelques résultats sur l'algorithme $SEM$ dans le cas de mélanges simples et en étudions les intérêts et inconvénients dans le cas du modèle des blocs latents.

Enfin, nous parlons de la combinaison de ces deux algorithmes pour tirer profit des avantages de chacun tout en diminuant les inconvénients et abordons le problème de la dégénérescence des classes.

## 2.2 Algorithme *Expectation Maximisation* (EM)

Le modèle des blocs latents étant un modèle de mélange, le calcul de l'estimateur du maximum de vraisemblance (EMV) ne peut pas être obtenu en cherchant à annuler la dérivée de la log-vraisemblance comme pour des cas d'inférences statistiques simples (voir la section A.2.1). L'idée proposée par Dempster et al. (1977) est de maximiser la log-vraisemblance en tirant parti de la structure à données manquantes du problème. Pour cela, ils décomposent la log-vraisemblance, notée $L$, comme la différence de l'espérance conditionnelle connaissant un paramètre $\boldsymbol{\theta}^{(c)}$ et les observations de la log-vraisemblance complétée et d'une fonction en $\boldsymbol{\theta}$ :

$$L(\boldsymbol{\theta}) = \log p(X;\boldsymbol{\theta}) = \underbrace{\mathbb{E}_{\mathbf{Z}|X;\boldsymbol{\theta}^{(c)}}\left[\log\left(p(X,\mathbf{Z};\boldsymbol{\theta})\right)\right]}_{=:Q\left(\boldsymbol{\theta}\middle|\boldsymbol{\theta}^{(c)}\right)} - \underbrace{\mathbb{E}_{\mathbf{Z}|X;\boldsymbol{\theta}^{(c)}}\left[\log\left(p(\mathbf{Z}|X;\boldsymbol{\theta})\right)\right]}_{=:H\left(\boldsymbol{\theta}\middle|\boldsymbol{\theta}^{(c)}\right)}$$

où $\mathbf{Z}$ représente dans ce cas l'ensemble des variables latentes (voir la section A.3.2).

Pour tout paramètre $\boldsymbol{\theta}$, la valeur $H\left(\boldsymbol{\theta}\middle|\boldsymbol{\theta}^{(c)}\right)$ est inférieure à $H\left(\boldsymbol{\theta}^{(c)}\middle|\boldsymbol{\theta}^{(c)}\right)$ et pour un paramètre $\boldsymbol{\theta}^{(c+1)}$ dont la valeur $Q\left(\boldsymbol{\theta}^{(c+1)}\middle|\boldsymbol{\theta}^{(c)}\right)$ est supérieure à $Q\left(\boldsymbol{\theta}^{(c)}\middle|\boldsymbol{\theta}^{(c)}\right)$, la log-vraisemblance associée est plus grande (voir l'annexe A.3.2).

L'algorithme *Expectation Maximisation* (ou *EM*) part d'une première valeur $\boldsymbol{\theta}^{(0)}$ et fait des itérations successives des deux étapes suivantes :

---

**Algorithme *Expectation Maximisation*** :

---

1. Initialisation de $\boldsymbol{\theta}^{(0)}$.
2. Itération :
   (a) Étape $E$ : calcul pour tout $\boldsymbol{\theta}$ de

$$Q\left(\boldsymbol{\theta}\middle|\boldsymbol{\theta}^{(c)}\right) = \mathbb{E}_{\mathbf{Z}|X;\boldsymbol{\theta}^{(c)}}\left[\log\left(p(X,\mathbf{Z};\boldsymbol{\theta})\right)\right].$$

   (b) Étape $M$ : calcul de $\boldsymbol{\theta}^{(c+1)}$ tel que

$$\boldsymbol{\theta}^{(c+1)} \in \operatorname{argmax} Q\left(\boldsymbol{\theta}\middle|\boldsymbol{\theta}^{(c)}\right).$$

---

Pour stopper l'algorithme, il existe de nombreux critères d'arrêt : par exemple, après un nombre fini d'itérations, lorsque la valeur $L\left(\theta^{(c)}\right)$ n'évolue plus ou encore lorsque les paramètres $\boldsymbol{\theta}^{(c)}$ successifs sont suffisamment proches les uns des autres.

*Remarque numérique* 2.2.1.
Dans certains cas, et en particulier lorsque le calcul de $Q$ est compliqué, il peut être préférable de ne chercher qu'une simple valeur augmentant $Q$, plutôt que la valeur la rendant maximum.

L'algorithme *EM* converge vers un maximum local de la log-vraisemblance. Il est conseillé de relancer plusieurs fois l'algorithme à partir d'initialisations différentes pour espérer détecter un maximum global. Il existe un certain nombre de stratégies d'initialisation (voir par exemple Biernacki et al., 2006), certaines spécifiques aux lois des mélanges (comme pour la section 5.5).

Dans le cas du modèle des blocs latents, le calcul de l'espérance conditionnelle aux observations, et

calculée avec la valeur $\boldsymbol{\theta}^{(c)}$ courante du paramètre, de la log-vraisemblance complétée est le suivant :

$$Q\left(\boldsymbol{\theta}\left|\boldsymbol{\theta}^{(c)}\right.\right) = \sum_{i,k} s_{ik}^{(c)} \log \pi_k + \sum_{j,\ell} t_{j\ell}^{(c)} \log \rho_\ell + \sum_{i,j,k,\ell} e_{ijk\ell}^{(c)} \log \varphi(x_{ij}; \alpha_{k\ell})$$

avec

$$s_{ik}^{(c)} = \mathbb{P}\left(z_{ik} = 1 \left| \mathbf{x}; \boldsymbol{\theta}^{(c)}\right.\right), \quad t_{j\ell}^{(c)} = \mathbb{P}\left(w_{j\ell} = 1 \left| \mathbf{x}; \boldsymbol{\theta}^{(c)}\right.\right),$$
$$\text{et } e_{ijk\ell}^{(c)} = \mathbb{P}\left(z_{ik} = 1, w_{j\ell} = 1 \left| \mathbf{x}; \boldsymbol{\theta}^{(c)}\right.\right).$$

Or, si nous avons supposé, par l'hypothèse $H_2$ de la section (1.3.1), l'indépendance des variables latentes, cette indépendance ne se retrouve pas dans la loi conditionnelle et le calcul de $e_{ijk\ell}^{(c)}$ ne se factorise pas comme dans le cas d'un mélange simple :

$$e_{ijk\ell} = \sum_{\substack{(\mathbf{z},\mathbf{w})\in\mathcal{Z}\times\mathcal{W} \\ z_{ik}w_{j\ell}=1}} p\left(\mathbf{z}, \mathbf{w} \left| \mathbf{x}; \boldsymbol{\theta}^{(c)}\right.\right).$$

Cette somme est alors obligatoire et le cardinal de l'ensemble sur laquelle elle est faite est presque du même ordre de grandeur que celui de $\mathcal{Z} \times \mathcal{W}$. Il est utopique d'espérer pouvoir calculer une somme sur un ensemble de cardinal $|\mathcal{Z} \times \mathcal{W}|$ en un temps raisonnable (voir section 1.3.2.2) donc a fortiori pour le calcul de $Q$ à chaque itération.
Cette dépendance intervient également, mais à moindre mesure, dans le calcul des $s_{ik}^{(c)}$ et $t_{j\ell}^{(c)}$.

## 2.3 Algorithme *VEM*

Pour contourner cette difficulté, Govaert et Nadif (2008) proposent d'utiliser une approximation variationnelle. Le principe est de considérer l'étape $E$ de l'algorithme *EM* comme une optimisation fonctionnelle de distributions libres des variables latentes $q_{\mathbf{zw}}(\mathbf{z}, \mathbf{w})$ quelconques. Ainsi, ils décomposent la log-vraisemblance comme la somme de deux fonctions (voir par exemple Keribin, 2010) :

$$L(\boldsymbol{\theta}) = \underbrace{\mathbb{E}_{(\mathbf{Z},\mathbf{W})\sim q_{\mathbf{zw}}}\left[\log\left(\frac{p\left(\mathbf{x}, \mathbf{Z}, \mathbf{W}; \boldsymbol{\theta}\right)}{q_{\mathbf{zw}}(\mathbf{Z}, \mathbf{W})}\right)\right]}_{=:\mathcal{F}(q_{\mathbf{zw}};\boldsymbol{\theta})} + \mathrm{D_{KL}}\left(q_{\mathbf{zw}} || p(\cdot, \cdot | \mathbf{x}; \boldsymbol{\theta})\right)$$

où $\mathcal{F}$ est une fonction de la distribution libre $q_{\mathbf{zw}}$ et $\mathrm{D_{KL}}$ est la divergence de Kullback-Leiber (Kullback et Leibler, 1951). Comme la divergence de Kullback-Leiber est positive, l'*énergie libre* $\mathcal{F}$ est une minorante de la log-vraisemblance égale à cette dernière si et seulement si la distribution libre $q_{\mathbf{zw}}$ est égale à la densité $p\left(\cdot, \cdot \left| \mathbf{x}; \boldsymbol{\theta}^{(c)}\right.\right)$. Comme pour l'algorithme *EM*, l'estimation de l'estimateur du maximum de vraisemblance se fait par une maximisation alternée de l'énergie libre :

— Étape $E$ remplacée par : maximisation de l'énergie libre $\mathcal{F}\left(q_{\mathbf{zw}}; \boldsymbol{\theta}^{(c)}\right)$ en $q_{\mathbf{zw}}$ à $\boldsymbol{\theta}^{(c)}$ fixé,

— Étape $M$ inchangée : maximisation de l'énergie libre $\mathcal{F}\left(q_{\mathbf{zw}}^{(c+1)}; \boldsymbol{\theta}\right)$ en $\boldsymbol{\theta}$ à $q_{\mathbf{zw}}^{(c+1)}$ fixée.

Le principe de l'approximation variationnelle est de maximiser l'énergie libre à $\boldsymbol{\theta}^{(c)}$ fixé en prenant les distributions libres dans une famille permettant de faire les calculs sans trop dégrader l'estimation.

Nous avons vu dans la section 2.2 que le problème d'utilisation l'algorithme *EM* est dû aux variables latentes non indépendantes conditionnellement aux observations ; Govaert et Nadif (2008) utilisent une

approximation en champ moyen, c'est-à-dire supposent que les distributions libres $q_{\mathbf{zw}}$ se factorisent en $\mathbf{z}$ et $\mathbf{w}$ :

$$q_{\mathbf{zw}}(\mathbf{z}, \mathbf{w}) = q_{\mathbf{z}}(\mathbf{z}) q_{\mathbf{w}}(\mathbf{w}),$$

qui, comme pour l'hypothèse $(H_2)$ de la section 1.3.1, a pour conséquence l'indépendance conditionnelle des lois des labels d'appartenance aux classes pour chaque ligne et chaque colonne. Ainsi, à chaque étape, ils maximisent la fonction :

$$\begin{aligned}\mathcal{F}(q_{\mathbf{zw}}; \boldsymbol{\theta}) &= \sum_{i,k} s_{ik}^{(c)} \log \pi_k + \sum_{j,\ell} t_{j\ell}^{(c)} \log \rho_\ell + \sum_{i,j,k,\ell} s_{ik}^{(c)} t_{j\ell}^{(c)} \log \varphi(x_{ij}; \alpha_{k\ell}) \\ &\quad - \sum_{i,k} s_{ik}^{(c)} \log s_{ik}^{(c)} - \sum_{j,\ell} t_{j\ell}^{(c)} \log t_{j\ell}^{(c)}\end{aligned}$$

en alternant par rapport aux $\left(s_{ik}^{(c)}, t_{j\ell}^{(c)}\right)$ et aux $\boldsymbol{\theta}$.

L'algorithme *VEM* (*Variationnal Expectation Maximisation*) proposé par Govaert et Nadif (2008) est le suivant :

**Algorithme *Variationnal Expectation Maximisation*** :

1. Initialisation de $\boldsymbol{\theta}^{(0)}$ et de $t_{j\ell}^{(0)}$.
2. Itération jusqu'à ce que $\mathcal{F}\left(q_{\mathbf{zw}}; \boldsymbol{\theta}^{(c)}\right)$ n'évolue plus :
   (a) Étape $VE$ : maximisation alternée de l'énergie libre à $\boldsymbol{\theta}^{(c)}$ fixé en prenant $t_{j\ell}^{(t=0)} = t_{j\ell}^{(c)}$ :
      i. pour tout $i$ et $k$, calcul de $s_{ik}^{(t+1)}$ à $t_{j\ell}^{(t)}$ et à $\boldsymbol{\theta}^{(c)}$ fixés :

$$s_{ik}^{(t+1)} = \frac{\pi_k^{(c)} \prod_{\ell=1}^{m} \prod_{h=1}^{r} \left( \left( \alpha_{k\ell}^{h\ (c)} \right)^{\sum_{j=1}^{d} t_{j\ell}^{(t)} v_{ijh}} \right)}{\sum_{k'=1}^{g} \pi_{k'}^{(c)} \prod_{\ell=1}^{m} \prod_{h=1}^{r} \left( \left( \alpha_{k'\ell}^{h\ (c)} \right)^{\sum_{j=1}^{d} t_{j\ell}^{(t)} v_{ijh}} \right)},$$

      ii. pour tout $j$ et $\ell$, calcul de $t_{j\ell}^{(t+1)}$ à $s_{ik}^{(t+1)}$ et à $\boldsymbol{\theta}^{(c)}$ fixés de la même manière.

      Obtention des probabilités $s_{ik}^{(c+1)}$ et $t_{j\ell}^{(c+1)}$, les dernières calculées.

   (b) Étape $M$ : calcul du paramètre $\boldsymbol{\theta}^{(c+1)}$ :

$$\pi_k^{(c+1)} = \frac{\sum_{i=1}^{n} s_{ik}^{(c+1)}}{n}, \quad t_{j\ell}^{(c+1)} = \frac{\sum_{j=1}^{d} t_{j\ell}^{(c+1)}}{d}$$

$$\text{et } \alpha_{k\ell}^{h\ (c+1)} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{d} s_{ik}^{(c+1)} t_{j\ell}^{(c+1)} v_{ijh}}{\sum_{i=1}^{n} \sum_{j=1}^{d} s_{ik}^{(c+1)} t_{j\ell}^{(c+1)}}.$$

3. Obtention d'un estimateur $\widehat{\theta}^{VEM} = \boldsymbol{\theta}^{(\infty)}$.
4. Calcul des estimateurs des classes en ligne $\widehat{z}^{VEM}$ et en colonne $\widehat{w}^{VEM}$ tels que :

$$\widehat{z}_i^{VEM} \in \underset{k=1,\ldots,g}{\operatorname{argmax}}\, s_{ik}^{(\infty)} \quad \text{et} \quad \widehat{w}_j^{VEM} \in \underset{\ell=1,\ldots,m}{\operatorname{argmax}}\, t_{j\ell}^{(\infty)}$$

*Remarque numérique* 2.3.1.
En pratique, Govaert et Nadif (2008) montrent qu'il est suffisant de faire une seule itération de l'étape 2.(a). L'algorithme est à la fois plus rapide et ses estimations sont meilleures.

**Avantages et inconvénients de l'algorithme *VEM***

+ L'algorithme *VEM* converge assez rapidement vers un maximum local de l'énergie libre.
− L'algorithme *VEM* est notablement plus dépendant des conditions initiales que l'algorithme *EM* (voir par exemple Bishop, 2006, chapitre 10). Le maximum global de l'énergie libre est un minorant de celui de la log-vraisemblance.

**Comportement de l'énergie libre**
L'énergie libre est un minorant de la log-vraisemblance dont la qualité dépend de la divergence de Kullback-Leiber qui ne peut être estimée. Les deux problèmes qui se posent sont de savoir si

— le paramètre $\theta$ qui maximise l'énergie libre est proche de celui qui maximise la log-vraisemblance,
— la valeur maximale de l'énergie libre est proche de celle de la log-vraisemblance.

Dans le cas des modèles de mélange classiques, Keribin (2010, section 3.2) recense dans son article un certain nombre de résultats :

— Wang et Titterington (2004a) montrent que, dans le cas de mélanges simples de lois de la famille exponentielle, l'estimateur obtenu avec l'approximation en champ moyen converge vers la vraie valeur des paramètres et que la loi de l'estimateur est asymptotiquement normale à la vitesse $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$.
— De plus, Wang et Titterington (2005) et Wang et al. (2006) montrent que dans le cas gaussien, l'estimateur obtenu avec une approximation variationnelle converge en vitesse $\mathcal{O}\left(\frac{1}{n}\right)$ vers l'estimateur du maximum de vraisemblance mais la matrice de covariance de la loi limite est inférieure à celle de l'estimateur du maximum de vraisemblance, cela a notamment une influence dans le calcul des intervalles de confiance (la démonstration de cette dernière affirmation peut être étendue au cas des mélanges de lois appartenant à la famille exponentielle ; voir Wang et Titterington (2004b)).
— Toutefois, Humphreys et Titterington (2000) montrent que l'approximation est plutôt mauvaise lorsque le nombre d'observations n'est pas assez important. Ils illustrent leur propos sur des mélanges de lois bêta.

Ces observations, associées à la convergence des labels démontrée par Mariadassou et Matias (2012) montrant que la loi conditionnelle tend vers une dirac et se factorise donc, laissent penser que l'approximation variationnelle en champ moyen donne de bons résultats lorsque le nombre d'observations est assez grand.
En revanche, nous ignorons toujours le comportement à distance finie.

*Remarque numérique* 2.3.2. **Estimation de la log-vraisemblance par simulations**
Dans un des travaux de cette thèse, nous avons essayé de comprendre le comportement de l'énergie libre par rapport à celui de la log-vraisemblance. Pour cela, nous avons tenté d'estimer cette dernière en utilisant des méthodes de simulations types MCMC (*Monte Carlo Markov Chain*) à partir d'un paramètre $\boldsymbol{\theta}$ fixé pour des matrices de tailles inférieures ou égales à 20 lignes et 20 colonnes. Comme l'énergie libre était parfois plus grande que notre estimation de la log-vraisemblance lorsque nous avions entre 10 et 20 lignes ou colonnes, nous en avons déduit que nos estimations n'étaient pas correctes. Nous voyons deux raisons à cela :

— la première approximation est due au nombre d'itérations. Vu le cardinal de l'ensemble $\mathcal{Z} \times \mathcal{W}$, il est fort probable que les 10 000 000 d'itérations faites n'ont pas suffit pour nos estimations.
— La deuxième approximation est numérique. Bien que nous ayons essayé un important nombre d'astuces informatiques pour limiter cette approximation, il est difficile de prendre en compte le cumul des arrondis.

## 2.4 Algorithme *SEM-Gibbs*

Plutôt que d'approximer numériquement la loi jointe conditionnelle, Keribin et al. (2014) proposent de la simuler à l'aide d'un échantillonneur de *Gibbs* (Gelfand et Smith, 1990), c'est-à-dire de simuler un grand nombre de couples $(\mathbf{z}, \mathbf{w})$ de telle sorte que la loi stationnaire de la chaîne de Markov obtenue soit la loi jointe conditionnelle.

L'algorithme *SEM-Gibbs* ainsi défini ne fait plus une approximation numérique de la loi conditionnelle mais une approximation stochastique.

---

**Algorithme *Stochastic Expectation Maximisation*** :

---

1. Initialisation de $\boldsymbol{\theta}^{(0)}$ et de $\mathbf{w}^{(0)}$.
2. Itération *niter* fois :
   (a) Étape $SE$ : simulation des couples $\left(\mathbf{z}^{(t)}, \mathbf{w}^{(t)}\right)$ en prenant $\mathbf{w}^{(t=0)} = \mathbf{w}^{(c)}$ et en itérant $niter_{int}$ fois :
      i. estimation de la loi $p\left(\mathbf{z} \left| \mathbf{x}, \mathbf{w}^{(t)}; \boldsymbol{\theta}^{(c)} \right.\right)$ puis tirage de $\mathbf{z}^{(t+1)}$ :

$$p\left(z_{ik} = 1 \left| x_{i\bullet}, \mathbf{w}^{(t)}; \boldsymbol{\theta}^{(c)} \right.\right) = \frac{\pi_k^{(c)} \prod_{\ell=1}^{m} \prod_{h=1}^{r} \left( \left( \alpha_{k\ell}^{h\ (c)} \right)^{\sum_{j=1}^{d} w_{j\ell}^{(t)} v_{ijh}} \right)}{\sum_{k'=1}^{g} \pi_{k'}^{(c)} \prod_{\ell=1}^{m} \prod_{h=1}^{r} \left( \left( \alpha_{k'\ell}^{h\ (c)} \right)^{\sum_{j=1}^{d} w_{j\ell}^{(t)} v_{ijh}} \right)},$$

      ii. de la même manière, estimation de la loi $p\left(\mathbf{w} \left| \mathbf{x}, \mathbf{z}^{(t+1)}; \boldsymbol{\theta}^{(c)} \right.\right)$ puis tirage de $\mathbf{w}^{(t+1)}$.

      Conservation du dernier couple $\left(\mathbf{z}^{(c+1)}, \mathbf{w}^{(c+1)}\right)$ simulé.

   (b) Étape $M$ : mise à jour du paramètre $\boldsymbol{\theta}^{(c+1)}$ :

$$\pi_k^{(c+1)} = \frac{\sum_{i=1}^{n} z_{ik}^{(c+1)}}{n}, \quad \rho_\ell^{(c+1)} = \frac{\sum_{j=1}^{d} w_{j\ell}^{(c+1)}}{d}$$

$$\text{et } \alpha_{k\ell}^{h\ (c+1)} = \frac{\sum_{i=1}^{n} \sum_{j=1}^{d} z_{ik}^{(c+1)} w_{j\ell}^{(c+1)} v_{ijh}}{\sum_{i=1}^{n} \sum_{j=1}^{d} z_{ik}^{(c+1)} w_{j\ell}^{(c+1)}}.$$

3. Obtention d'un estimateur $\widehat{\theta}^{SEM} = \frac{1}{niter} \sum_{c=1}^{niter} \boldsymbol{\theta}^{(c)}$ en moyennant les paramètres obtenus.
4. Calcul des estimateurs des classes en ligne $\widehat{z}^{SEM}$ et en colonne $\widehat{w}^{SEM}$ par affectation à la classe majoritairement affectée durant les simulations :

$$\widehat{z}_i^{SEM} \in \underset{k=1,\ldots,g}{\operatorname{argmax}} \sum_{c=1}^{niter} z_{ik}^{(c)} \quad \text{et} \quad \widehat{w}_j^{SEM} \in \underset{\ell=1,\ldots,m}{\operatorname{argmax}} \sum_{c=1}^{niter} w_{j\ell}^{(c)}$$

---

L'algorithme *SEM-Gibbs* génère une chaîne de Markov sur $\boldsymbol{\theta}^{(c)}$ dont la moyenne peut être vue comme une approximation de l'estimateur du maximum de vraisemblance. Il est préférable de ne pas prendre en compte les premières estimations pour ne pas biaiser le résultat, c'est-à-dire ne pas prendre les sommes à partir de 1 pour les étapes 3 et 4 mais un nombre plus grand ; nous parlons de *temps de chauffe*.

**Avantages et inconvénients de l'algorithme *SEM-Gibbs***

+ L'algorithme *SEM-Gibbs* est peu sensible aux initialisations et renvoie une estimation $\widehat{\theta}^{SEM}$ proche de la vraie valeur si *niter* est suffisamment grand.
− Les estimations renvoyées fluctuent autour des valeurs recherchées suivant les chaînes générées.

*Remarque numérique* 2.4.1.
En pratique, Keribin et al. (2010) montrent qu'une seule itération de l'étape $SE$ suffit. Pour que l'algorithme soit plus rapide, il est donc conseillé de prendre $niter_{int} = 1$.

Il ressort des simulations que le problème de label switching n'apparaît presque jamais. Il faut que *niter* soit très grand pour que l'algorithme rencontre cette difficulté. Toutefois, il est conseillé d'utiliser les conditions suffisantes $C_1$ et $C_2$ du critère d'identifiabilité de Keribin (section 1.3.2.1) pour réordonner les résultats avant de moyenner sur les $\boldsymbol{\theta}^{(c)}$ et d'affecter les classes en ligne et en colonne. Par contre, il est déconseillé de réorganiser les labels durant la simulation, la chaîne de Markov associée serait alors biaisée (Celeux et al., 2000).

*Remarque* 2.4.2. **États absorbants de l'algorithme *SEM-Gibbs***
Un *état absorbant* est une configuration des matrices $\left(\mathbf{z}^{(c+1)}, \mathbf{w}^{(c+1)}\right)$ telles qu'à l'étape suivante, la simulation de l'étape $SE$ génère presque sûrement le même couple. L'algorithme *SEM-Gibbs* en possède potentiellement deux types :

**Type I** : si l'algorithme renvoie une partition avec une classe vide, cette dernière le reste jusqu'à la fin de la simulation. Or, la probabilité d'avoir des partitions avec une seule classe est strictement positive (sauf si nous sommes dans un état absorbant de type II) et l'algorithme renvoie donc une seule classe avec une probabilité tendant vers 1 avec le nombre d'itérations *niter*.

Pour éviter ce problème, l'étape $SE$ est répétée tant que les matrices $\left(\mathbf{z}^{(c+1)}, \mathbf{w}^{(c+1)}\right)$ contiennent des classes vides. L'obtention de matrices sans classes vides est une variable aléatoire suivant une loi géométrique de paramètre plus grand que $\frac{\binom{n}{g}\binom{d}{m}}{(n^2d)^g(d^2n)^m}$ ; l'espérance pouvant être très grande, il est recommandé de relancer l'algorithme à partir d'une nouvelle valeur initiale si le nombre d'essais est trop important.

**Type II** : le deuxième état absorbant que nous avons identifié est dû à la structure en blocs du modèle. Si pour une itération $c$, les partitions simulées $\left(\mathbf{z}^{(c)}, \mathbf{w}^{(c)}\right)$ sont telles que pour chaque classe en ligne et en colonne, il y ait au moins un bloc composé de cases avec la même valeur (comme sur la figure 2.1, à gauche), l'algorithme réaffecte presque sûrement les mêmes labels pour chaque ligne à l'itération suivante. Pour être précis, il faut rajouter des conditions sur les autres cases dont une formalisation mathématique ainsi qu'une démonstration est proposée en annexe 2.A ; si ces conditions ne sont pas respectées, il est possible que seulement certaines lignes ou colonnes soient bloquées (comme sur la figure 2.1, à droite) voir aucunes.

Pour palier ce problème, et en même temps limiter celui de l'état absorbant de type I, nous proposons de redonner arbitrairement la valeur 0.1 (resp. 0.9) pour les estimations $\alpha_{k\ell}^{h\ (c+1)}$ trop proches de 0 (resp. de 1).

En pratique, cet état absorbant peut se rencontrer lorsque le nombre de classes demandé est élevé ou lorsque la matrice étudiée est composée d'une valeur dominante.
Il peut se retrouver pour d'autres lois :

— pour les lois de Poisson en ayant des blocs composés uniquement de la valeur 0,
— pour des données gaussiennes, dans le cas où des blocs sont composés de la même valeur. En théorie, la probabilité d'avoir ce type de matrice avec des blocs contenant plus de 2 cases est nulle ; mais en pratique, surtout si les données sont discrétisées, il est possible de rencontrer ce type de

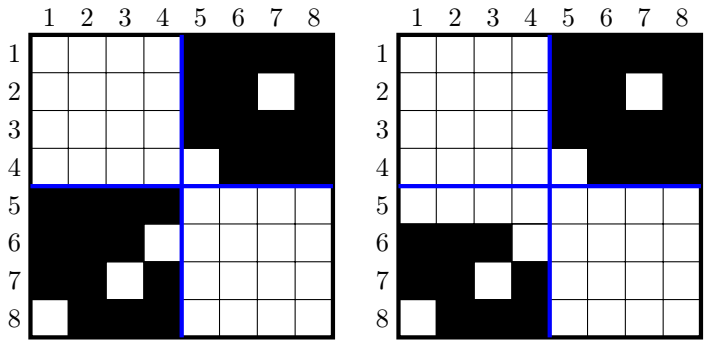



FIGURE 2.1 – À gauche, représentation d'une matrice étant dans un état absorbant de type II pour le cas binaire. L'algorithme *SEM-Gibbs* reste bloqué dans cette configuration à cause des deux blocs blancs (en haut à gauche et en bas à droite). À droite, la même matrice avec une ligne totalement blanche ; seule cette ligne peut changer d'affectation de classe.

configuration.

Avec ces améliorations, la chaîne $\left(\mathbf{z}^{(c)}, \mathbf{w}^{(c)}\right)$ obtenue est ergodique sur l'espace des configurations privé des états absorbants ; et avec elle la chaîne $\left(\boldsymbol{\theta}^{(c)}\right)$ sur l'espace des paramètres tel qu'aucun paramètre $\alpha_{k\ell}^{h}$ ne puisse valoir 1, ni aucune proportion $\pi_k$ ou $\rho_\ell$ ne puisse valoir 0. Nous espérons que la loi stationnaire de $\left(\boldsymbol{\theta}^{(c)}\right)$ se concentre alors autour de l'estimateur du maximum de vraisemblance.

## 2.5 Algorithme *SEM*+*VEM*

L'algorithme *VEM* donne de bons résultats lorsque l'initialisation est proche de l'optimum, l'algorithme *SEM-Gibbs* est peu sensible aux initialisations et propose des estimations proches des valeurs recherchées. Nous proposons donc d'utiliser les avantages de chacun en les couplant en une procédure notée *SEM*+*VEM* définie par :

---

**Algorithme *SEM*+*VEM*** :

---

1. Initialisation de $\boldsymbol{\theta}^{(0)}$ et $\mathbf{w}^{(0)}$.
2. Avec ces initialisations, utilisation de l'algorithme *SEM-Gibbs* pour obtenir des estimateurs $\widehat{\theta}^{SEM}$ et $\widehat{w}^{SEM}$.
3. En prenant ces estimateurs comme initialisation, utilisation de l'algorithme *VEM* pour obtenir des estimateurs $\widehat{\theta}^{VEM}$, $\widehat{z}^{VEM}$ et $\widehat{w}^{VEM}$.

---

Nous notons $\widehat{\theta}^{SV}$, $\widehat{z}^{SV}$ et $\widehat{w}^{SV}$ les estimateurs obtenus.

*Remarque* 2.5.1.
Les remarques faites précédemment sur l'énergie libre et sur le nombre d'itération *niter* pour l'algorithme *SEM-Gibbs* sont encore valables.

Pour montrer empiriquement les améliorations obtenues par la combinaison *SEM*+*VEM*, nous avons effectué un plan de simulation avec $g = 3$ classes en ligne, $m = 2$ classes en colonne, $n = 100$ lignes et

$d = 60$ colonnes, des proportions équilibrées et des paramètres $\boldsymbol{\alpha}$ de la forme :

$$\boldsymbol{\alpha} = \begin{pmatrix} \varepsilon & 1-\varepsilon \\ 1-\varepsilon & 2\varepsilon - 0.49 \\ 1-\varepsilon & 1-\varepsilon \end{pmatrix}$$

en faisant varier $\varepsilon \in \{0.25, 0.3, 0.35\}$. Nous avons simulé 100 matrices et utilisé les algorithmes *SEM-Gibbs* et *VEM* avec la même initialisation aléatoire. Ensuite, nous avons utilisé le résultat de l'algorithme *SEM-Gibbs* comme initialisation d'un algorithme *VEM* ; ceci permet de voir l'amélioration de la combinaison des deux par rapport aux algorithme *VEM* et *SEM-Gibbs* seuls. Il est à noter que l'algorithme *VEM* est désavantagé par le fait qu'une seule initialisation soit proposée ; une comparaison à temps d'exécution (*elapsed*) fixé est proposée dans le chapitre 6.

Sur la figure 2.2, sont représentées les boxplots des estimations des composantes du paramètre $\boldsymbol{\theta}$ pour chacun des algorithmes après avoir utilisé a posteriori le critère d'identifiabilité pour contourner l'éventuel label switching. L'algorithme *VEM* est plus souvent bloqué dans un maximum local que les autres algorithmes. De plus, coupler l'algorithme *SEM-Gibbs* avec l'algorithme *VEM* permet de diminuer légèrement la variance des estimateurs et surtout d'améliorer l'estimation de certains paramètres ; notamment pour l'estimation des $\boldsymbol{\alpha}$ où nous pouvons voir des valeurs proches de 0.5 pour l'algorithme *SEM-Gibbs* qui n'apparaissent pas pour l'algorithme *SEM+VEM*.

## 2.6 Dégénérescence des classes

En augmentant le nombre de classes, l'algorithme *SEM+VEM* renvoie parfois un nombre de classes inférieur à celui demandé ; nous parlons de *dégénérescence des classes*. Deux possibilités peuvent être à l'origine de ce phénomène :

1. soit des probabilités $\pi_k^{(c)}$, estimées par l'algorithme *VEM*, d'appartenir à la classe $k$ tendent vers 0 (nous parlons de classes *qui se vident*),
2. soit au moment de l'affectation des classes en fin d'algorithme, il existe une classe en ligne $k$ dont aucune probabilité $s_{ik}^{(\infty)}$ n'est majoritaire pour chaque $i$.

### 2.6.1 Simulations

Pour mettre en évidence ce comportement, nous reprenons les résultats des simulations obtenus dans notre article (Keribin et al., 2014).
Le premier plan d'expériences consiste à simuler des matrices binaires avec $g = 5$ et $m = 4$ classes en ligne et colonne, des paramètres pour les blocs de la forme :

$$\boldsymbol{\alpha} = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & 1-\varepsilon \end{pmatrix}$$

où $\varepsilon$ varie entre 0 et 0.5 pour obtenir des matrices plus ou moins simples à classifier (allant de + à +++) suivant le protocole proposé par Lomet et al. (2012c). De plus, nous faisons varier :

— les proportions :

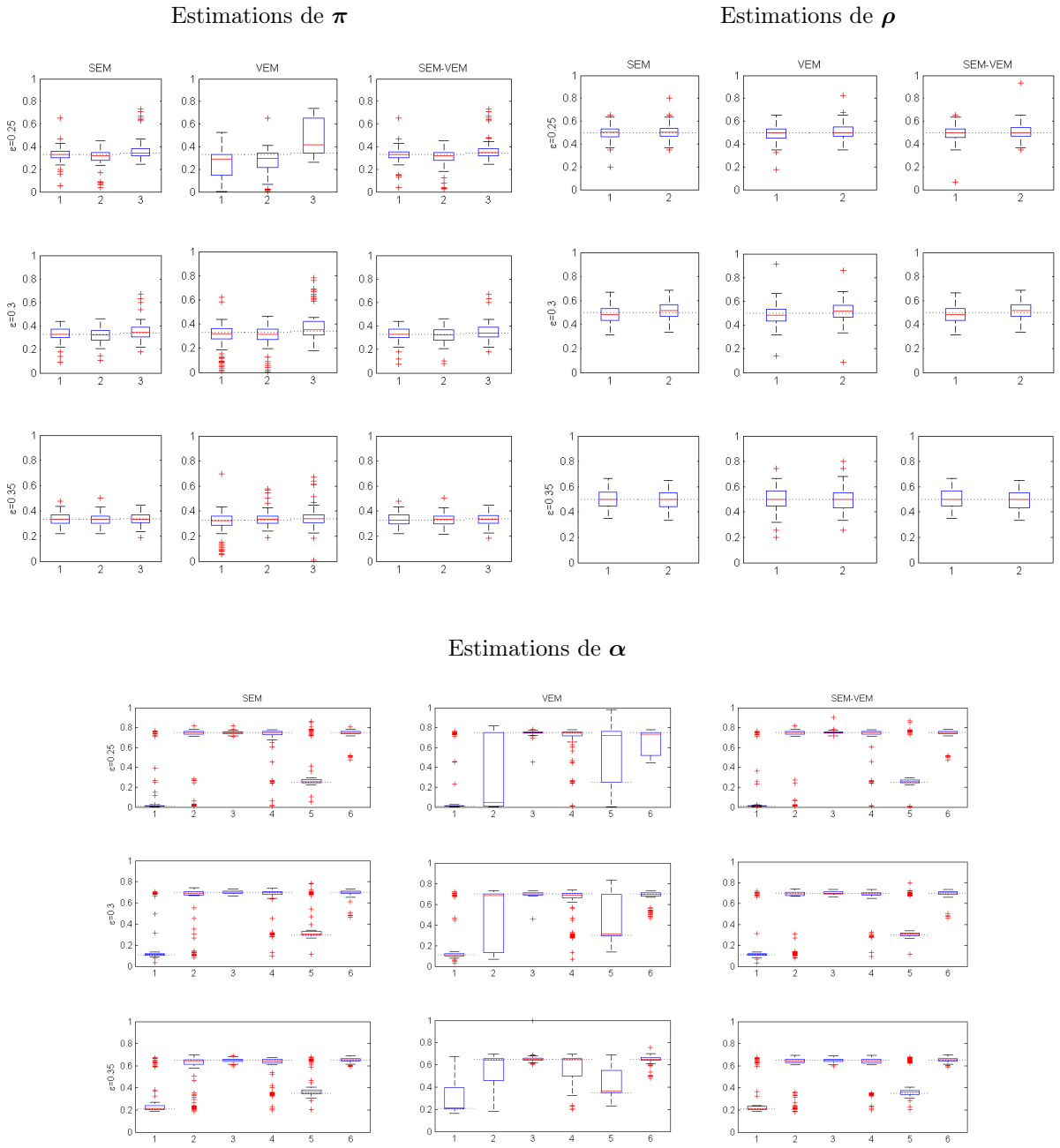



FIGURE 2.2 – Représentations par boxplot des estimations des composantes de $\boldsymbol{\pi}$ et de $\boldsymbol{\rho}$ (en haut) et de $\boldsymbol{\alpha}$ (en bas) en fonction des algorithmes (en colonne) et de la difficulté des données (en ligne). Le trait en pointillé symbolise la valeur ayant servi à la simulation.

- proportion équilibrée :

$$\boldsymbol{\pi} = \begin{pmatrix} 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \end{pmatrix} \quad \text{et} \quad \boldsymbol{\rho} = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix},$$

- proportion avec une progression arithmétique :

$$\boldsymbol{\pi} = \begin{pmatrix} 0.1 \\ 0.15 \\ 0.2 \\ 0.25 \\ 0.3 \end{pmatrix} \quad \text{et} \quad \boldsymbol{\rho} = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.3 \\ 0.4 \end{pmatrix},$$

— le nombre $(n, d)$ de lignes et de colonnes parmi les couples suivant :

$$(100, 200), \quad (150, 150), \quad \text{et} \quad (200, 100).$$

Nous utilisons l'algorithme *SEM+VEM* avec 5 000 itérations pour *SEM-Gibbs* et le bon nombre de classes en ligne et en colonne ou un nombre de classes en ligne et en colonne plus grand $(g, m) = (8, 8)$. Dans le tableau 2.1, sont représentés les pourcentages de fois où l'algorithme a renvoyé au moins une classe vide sur 500 matrices simulées. Nous pouvons faire plusieurs observations :

1. l'algorithme *SEM+VEM* renvoie moins de classes vides lorsque le bon nombre de classes est demandé (en comparaison au cas où il est demandé trop de classes).
2. Lorsque le bon nombre de classes est demandé, l'algorithme *SEM+VEM* renvoie moins de classes vides lorsque la matrice est considérée comme facile que lorsqu'elle est considérée comme difficile. Lorsque la matrice est considérée comme difficile, il semblerait qu'il y ait plus de maxima locaux avec des classes vides.
3. À l'opposé, lorsque le nombre de classes demandé est supérieur au bon nombre, l'algorithme renvoie des classes vides plutôt pour les cas simples, ne sachant pas comment subdiviser la partition originale.

Au vu de ces remarques, notamment de la première, nous pouvons penser que l'algorithme renvoie une ou plusieurs classes vides car le nombre de classes ayant servi à simuler les données est inférieur à celui demandé et qu'il a finalement sélectionné le bon nombre de classes. Il est vrai que dans le cas de certaines matrices comme l'échiquier (voir la figure 1.15), demander plus de deux classes en ligne ou en colonne conduit souvent à l'obtention de classes vides.
Toutefois, nous pouvons objecter deux remarques à cette intuition :

— l'algorithme renvoie tout de même des classes vides pour le bon nombre de classes (jusqu'à 18% dans certains cas),
— même si le bon nombre de classes est $(8, 8)$, il y a beaucoup plus de maxima locaux que pour $(5, 4)$ et donc plus de chances d'obtenir des classes vides.

Pour vérifier ces remarques, les tableaux 2.2 et 2.3 représentent le nombre de classes obtenu au final lorsque l'algorithme *SEM+VEM* a été utilisé en demandant un nombre de classes $(g, m) = (8, 8)$ dans le cadre de l'expérience du tableau 2.1. Bien que l'algorithme ait renvoyé un nombre de classes inférieur, ce nombre est moins de 1% de fois le bon et il est même parfois inférieur à celui ayant servi aux simulations. Ceci laisse penser que ce serait plutôt le deuxième argument qui serait à l'origine de la dégénérescence

| | | $(g,m)=(5,4)$ | | | $(g,m)=(8,8)$ | | |
|---|---|---|---|---|---|---|---|
| | | + | ++ | +++ | + | ++ | +++ |
| Proportions équilibrées | $(100,200)$ | 5.6 | 7.2 | 13.8 | 57.6 | 41.8 | 43.2 |
| | $(150,150)$ | 5.4 | 5.6 | 7.4 | 61.2 | 48.4 | 44 |
| | $(200,100)$ | 3.2 | 5.6 | 10.4 | 74.8 | 72 | 68.2 |
| Proportions inégales | $(100,200)$ | 5.4 | 7 | 16.8 | 65 | 54.6 | 47.4 |
| | $(150,150)$ | 7.6 | 13 | 17.6 | 69.2 | 54.2 | 57.2 |
| | $(200,100)$ | 13.2 | 11.8 | 18.2 | 81.4 | 67.8 | 68.2 |

TABLE 2.1 – Sur 500 matrices, pourcentage d'estimations obtenues par l'algorithme *SEM+VEM* dont au moins l'une des classes est vide en fonction du nombre de lignes et de colonnes des matrices (ligne), de la difficulté (colonne), des proportions (ligne globale) et du nombre de classes demandé (colonne globale). Une couleur est mise pour mieux voir les différences entre les valeurs : plus le nombre est rouge, plus celui-ci est proche de 100.

des classes lorsque le nombre de classes demandé est plus grand.

Nous faisons un dernier plan d'expérience pour vérifier la deuxième remarque. Pour cela, nous utilisons les données[1] fournies par le protocole proposé par Lomet et al. (2012c) ; plus particulièrement, nous étudions les matrices avec $g = m = 5$ et $g = m = 10$ classes en ligne et en colonne pour des nombres de lignes et de colonnes égaux valant 50, 100, 200 et 500. Pour chaque configuration, sont proposées 20 matrices. Pour chacune d'elle, nous utilisons l'algorithme *SEM+VEM* avec le bon nombre de classes et un nombre égal à $(8,8)$ pour 25 initialisations aléatoires. Nous supposons donc avoir trop de classes lorsque le vrai nombre est $g = m = 5$ et pas assez lorsqu'il est de $g = m = 10$.
Le tableau 2.4 résume les pourcentages de fois où l'algorithme a renvoyé une classe vide.

Indépendamment du nombre de classes ayant servi à la simulation, l'algorithme *SEM+VEM* renvoie plus souvent des classes vides lorsque le nombre demandé est plus grand.
De plus, si la dégénérescence de l'algorithme est une information sur le nombre de classes, nous nous attendons à ce que celui-ci renvoie plus souvent des classes vides lorsque nous avons beaucoup de lignes et de colonnes (et donc, beaucoup d'informations) ; en particulier dans le cas de matrices faciles (+). Nous voyons que ce n'est pas le cas.

En conclusion, si l'algorithme *SEM+VEM* renvoie des classes vides, cela ne signifie pas forcément que le nombre de classes réel est inférieur à celui demandé. Il est nécessaire de mieux comprendre cette dégénérescence pour l'éviter.

1. Les données sont disponibles à l'adresse `https://www.hds.utc.fr/coclustering/doku.php`.

**Proportions équilibrées**

$+$

| | $g \backslash m$ | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|
| $(100, 200)$ | 4 | 0 | 0 | 0 | 0 | 0 |
| | 5 | [0] | 0 | 1 | 1 | 1 |
| | 6 | 0 | 16 | 19 | 15 | 19 |
| | 7 | 0 | 16 | 19 | 15 | 19 |
| | 8 | 0 | 5 | 25 | 69 | 212 |
| $(150, 150)$ | 5 | [0] | 0 | 0 | 0 | 0 |
| | 6 | 0 | 12 | 6 | 8 | 11 |
| | 7 | 0 | 7 | 30 | 31 | 37 |
| | 8 | 0 | 19 | 42 | 103 | 194 |
| $(200, 100)$ | 4 | 0 | 0 | 0 | 0 | 0 |
| | 5 | [1] | 1 | 1 | 0 | 0 |
| | 6 | 2 | 13 | 10 | 11 | 5 |
| | 7 | 0 | 24 | 31 | 28 | 12 |
| | 8 | 0 | 32 | 85 | 118 | 126 |

$++$

| | $g \backslash m$ | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| $(100, 200)$ | 4 | 0 | 0 | 0 | 0 | 1 | 0 |
| | 5 | 0 | [0] | 0 | 0 | 0 | 1 |
| | 6 | 0 | 0 | 8 | 9 | 4 | 12 |
| | 7 | 0 | 0 | 3 | 19 | 24 | 34 |
| | 8 | 0 | 0 | 3 | 17 | 75 | 291 |
| $(150, 150)$ | 5 | 0 | [0] | 0 | 0 | 0 | 0 |
| | 6 | 0 | 0 | 14 | 9 | 9 | 4 |
| | 7 | 0 | 0 | 14 | 16 | 24 | 28 |
| | 8 | 0 | 0 | 5 | 36 | 93 | 258 |
| $(200, 100)$ | 4 | 1 | 0 | 0 | 0 | 0 | 0 |
| | 5 | 0 | [1] | 2 | 0 | 0 | 0 |
| | 6 | 0 | 0 | 19 | 9 | 4 | 3 |
| | 7 | 0 | 0 | 15 | 34 | 24 | 8 |
| | 8 | 0 | 0 | 34 | 79 | 127 | 140 |

$+++$

| | $g \backslash m$ | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|
| $(100, 200)$ | 4 | 0 | 0 | 1 | 0 | 0 |
| | 5 | [0] | 4 | 1 | 0 | 0 |
| | 6 | 1 | 7 | 10 | 8 | 6 |
| | 7 | 0 | 8 | 15 | 18 | 34 |
| | 8 | 0 | 4 | 26 | 73 | 284 |
| $(150, 150)$ | 5 | [1] | 0 | 0 | 0 | 1 |
| | 6 | 1 | 8 | 8 | 4 | 3 |
| | 7 | 0 | 7 | 21 | 17 | 18 |
| | 8 | 1 | 6 | 30 | 94 | 280 |
| $(200, 100)$ | 4 | 0 | 0 | 0 | 0 | 0 |
| | 5 | [3] | 2 | 0 | 0 | 0 |
| | 6 | 2 | 16 | 13 | 4 | 1 |
| | 7 | 1 | 22 | 37 | 23 | 6 |
| | 8 | 1 | 30 | 68 | 112 | 159 |

TABLE 2.2 – Sur 500 matrices, répartition du nombre de classes obtenu par l'algorithme *SEM+VEM* utilisé avec un nombre de classes initiales de $(g, m) = (8, 8)$ pour le cas des proportions équilibrées. Le nombre de classes ayant servi à la simulation est encadré.

**Proportions inégales**

$(100, 200)$

| | + | | | | | ++ | | | | | | +++ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $g \backslash m$ | 4 | 5 | 6 | 7 | 8 | 3 | 4 | 5 | 6 | 7 | 8 | 3 | 4 | 5 | 6 | 7 | 8 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 |
| 5 | [0] | 3 | 2 | 1 | 0 | 0 | [5] | 1 | 1 | 2 | 1 | 0 | [3] | 2 | 1 | 3 | 4 |
| 6 | 2 | 16 | 15 | 13 | 16 | 0 | 2 | 13 | 13 | 6 | 12 | 0 | 2 | 9 | 18 | 15 | 9 |
| 7 | 1 | 21 | 27 | 24 | 46 | 0 | 1 | 15 | 26 | 35 | 41 | 0 | 2 | 5 | 13 | 25 | 37 |
| 8 | 0 | 18 | 47 | 73 | 175 | 0 | 0 | 3 | 24 | 71 | 227 | 0 | 0 | 6 | 19 | 62 | 263 |

$(150, 150)$

| | + | | | | | ++ | | | | | | +++ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $g \backslash m$ | 4 | 5 | 6 | 7 | 8 | 3 | 4 | 5 | 6 | 7 | 8 | 3 | 4 | 5 | 6 | 7 | 8 |
| 4 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 |
| 5 | [1] | 1 | 1 | 2 | 0 | 0 | [4] | 5 | 1 | 0 | 1 | 1 | [4] | 4 | 3 | 2 | 0 |
| 6 | 0 | 15 | 20 | 9 | 5 | 1 | 1 | 11 | 9 | 6 | 1 | 0 | 7 | 13 | 15 | 6 | 8 |
| 7 | 4 | 18 | 26 | 23 | 21 | 0 | 2 | 14 | 29 | 31 | 22 | 0 | 3 | 12 | 26 | 23 | 21 |
| 8 | 0 | 28 | 63 | 108 | 154 | 0 | 1 | 17 | 39 | 76 | 229 | 0 | 3 | 13 | 35 | 85 | 214 |

$(200, 100)$

| | + | | | | | ++ | | | | | | +++ | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $g \backslash m$ | 4 | 5 | 6 | 7 | 8 | 3 | 4 | 5 | 6 | 7 | 8 | 3 | 4 | 5 | 6 | 7 | 8 |
| 4 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 5 | [3] | 4 | 0 | 1 | 0 | 0 | [8] | 4 | 2 | 1 | 1 | 3 | [8] | 5 | 3 | 3 | 0 |
| 6 | 5 | 21 | 9 | 7 | 0 | 0 | 9 | 13 | 15 | 8 | 6 | 1 | 13 | 19 | 11 | 4 | 0 |
| 7 | 6 | 25 | 35 | 26 | 11 | 0 | 3 | 27 | 29 | 32 | 24 | 2 | 9 | 19 | 28 | 18 | 16 |
| 8 | 7 | 60 | 82 | 105 | 93 | 0 | 2 | 20 | 57 | 77 | 161 | 0 | 6 | 18 | 64 | 90 | 159 |

TABLE 2.3 – Sur 500 matrices, répartition du nombre de classes obtenu par l'algorithme *SEM+VEM* utilisé avec un nombre de classes initiales de $(g, m) = (8, 8)$ pour le cas des proportions inégales. Le nombre de classes ayant servi à la simulation est encadré.

| erreur / $(n, d)$ | + | | ++ | | +++ | |
|---|---|---|---|---|---|---|
| | Matrices / Initialisation | $(5,5)$ $(10,10)$ | Matrices / Initialisation | $(5,5)$ $(10,10)$ | Matrices / Initialisation | $(5,5)$ $(10,10)$ |
| (50,50) | Bon nombre | 0 18.6 | Bon nombre | 0 17.4 | Bon nombre | 0 17.6 |
| | $(8,8)$ | 2.8 2.4 | $(8,8)$ | 2.8 1.2 | $(8,8)$ | 1 0.8 |
| (100,100) | Bon nombre | 1 0 | Bon nombre | 0 0 | Bon nombre | 0 0 |
| | $(8,8)$ | 0 0 | $(8,8)$ | 0.4 0 | $(8,8)$ | 0 0 |
| (200,200) | Bon nombre | 0 0 | Bon nombre | 0 0 | Bon nombre | 0 0 |
| | $(8,8)$ | 0 0 | $(8,8)$ | 0 0 | $(8,8)$ | 0 0 |
| (500,500) | Bon nombre | 0 0 | Bon nombre | 0 0 | Bon nombre | 0 0 |
| | $(8,8)$ | 0 0 | $(8,8)$ | 0.4 0 | $(8,8)$ | 0.2 0 |

TABLE 2.4 – Résultats de l'expérience sur le nombre de classes vides pour les données de Lomet et al. (2012c) en fonction du nombre de lignes et colonnes des matrices (lignes globales) et de la difficulté des données (colonnes globales). Chaque case des petits tableaux représente le pourcentage de fois où l'algorithme *SEM+VEM* a obtenu une partition avec des classes vides suivant les initialisations (en ligne) pour chaque nombre de classes ayant servi à la simulation (en colonne).

### 2.6.2 Cause de cette dégénérescence

Lorsque le nombre de classes est faible, la fonction du maximum de vraisemblance possède peu de maxima locaux mais leur nombre augmente avec celui des classes ; parmi ces maxima locaux, un nombre de plus en plus important sont des cas dégénérés.

#### Algorithme *VEM*

Sur la figure 2.3, nous voyons les trajectoires des estimations $\pi_k^{(c)}$ pour l'algorithme *VEM* sur le cas d'une matrice [2] considérée comme facile par le protocole proposé par Lomet et al. (2012c). Nous avons pris des proportions initiales équilibrées pour les $\boldsymbol{\pi}$ et les $\boldsymbol{\rho}$ et avons simulé des valeurs de $\boldsymbol{\alpha}$ et une matrice initiale $\mathbf{w}^{(0)}$ de façon aléatoire. Au début, deux probabilités augmentent tandis que la troisième se rapproche de 0. Une fois que les deux plus fortes sont à peu près stables, la plus faible est trop proche de 0 pour espérer remonter ultérieurement.

#### Algorithme *SEM-Gibbs*

Sur la figrue 2.4 sont représentées deux trajectoires allant jusqu'à 100 000 itérations provenant des simulations de l'algorithme *SEM-Gibbs* à partir des matrices de la première expérience de la section 2.6.1. Ces deux trajectoires ont été choisies car elles se décomposent chacune en deux phases :

1. la première phase est assez instable, si nous prenons une fenêtre glissante, la variance pour chaque $\pi_k^{(c)}$ est assez grande. L'algorithme *SEM-Gibbs* est certainement dans un maximum local.

2. Les données sont disponibles à l'adresse `https://www.hds.utc.fr/coclustering/doku.php?id=en:downloads`.

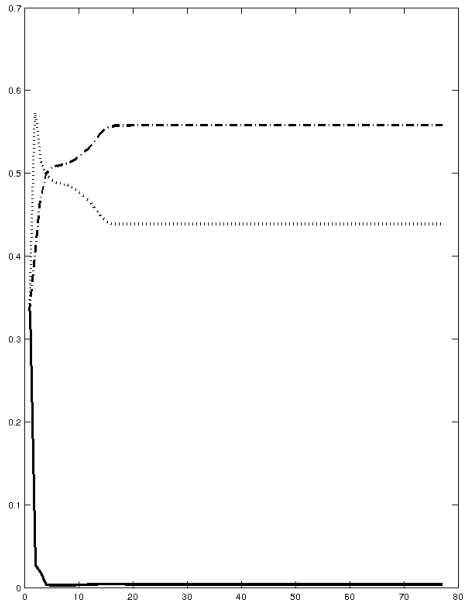



FIGURE 2.3 – Evolution des estimations des $\pi_k^{(c)}$ pour l'algorithme *VEM*. L'une des probabilités se rapproche progressivement vers 0.

2. La deuxième phase est beaucoup plus stable, les estimations n'oscillent pas au delà d'un intervalle d'amplitude $\frac{3}{d}$. L'algorithme *SEM-Gibbs* est dans un maximum global ou dans un maximum local dont la vraisemblance est plus grande que précédemment.

Plus la première phase est longue et plus elle a d'impact dans les estimations finales.

Nous présentons dans le chapitre 3 un point de vue bayésien du modèle des blocs latents et montrons qu'en choisissant correctement les lois a priori, ces difficultés sont limitées.

*Remarque* 2.6.1.
Dans la pratique, la première phase des trajectoires de la figure 2.4 est souvent très courte lorsque l'initialisation aléatoire n'est pas trop mauvaise. Une autre possibilité est de lancer plusieurs fois l'algorithme *SEM-Gibbs* pour différentes initialisations et un nombre plus petit d'itérations puis de conserver l'estimation maximisant l'énergie libre. Dans son livre, Robert (2006, section 6.3.1) résume les avantages de chacune des stratégies de la façon suivante :

— une seule chaîne a l'avantage d'éviter de gaspiller des temps de chauffe répétés dû aux multiples chaînes,
— de multiples chaînes ont l'avantage de permettre un contrôle de la convergence en loi.

Dans le cadre des simulations par méthode $MCMC$, Geyer (1992); Raftery et Lewis (1992) étudient une seule longue chaîne alors que Gelman et Rubin (1992) regardent plusieurs initialisations. À notre connaissance, il n'existe toujours pas de consensus quant au meilleur choix.

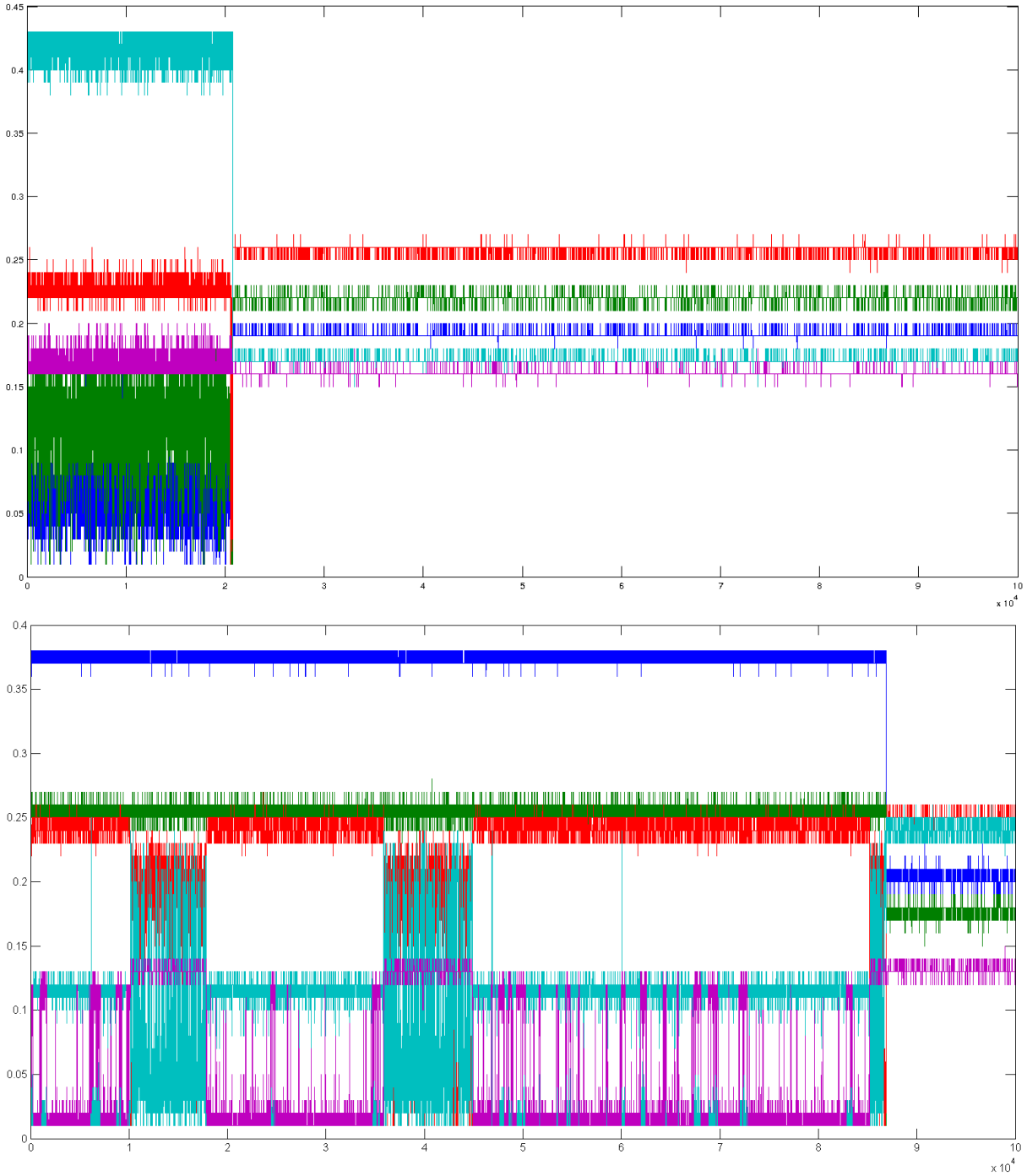



FIGURE 2.4 – Deux évolutions des estimations des $\pi_k^{(c)}$ pour l'algorithme *SEM-Gibbs*. Pour chacune, deux phases sont visibles : la première est associée à un maximum local, peu stable et la seconde au maximum global.

## 2.7 Conclusion

Les algorithmes *VEM* et *SEM-Gibbs* permettent de définir des algorithmes de type *EM* dans le cadre du modèle des blocs latents, pour lequel l'espérance conditionnelle aux observations n'est pas calculable. Le premier, faisant une approximation numérique de la log-vraisemblance, renvoie de bons résultats lorsque l'initialisation n'est pas trop éloignée des valeurs cibles mais est très sensible aux initialisations. Le second, utilisant un schéma stochastique, est peu sensible aux initialisations mais renvoie un résultat variable, fluctuant autour de l'EMV.

La combinaison *SEM+VEM* que nous proposons comble les lacunes de chacun des algorithmes produisant ainsi une procédure peu sensible aux initialisations et obtenant de bons résultats sur des données simulées. Il reste à résoudre le choix du nombre d'itérations *niter* de l'algorithme *SEM-Gibbs* et l'étude de la qualité de l'approximation variationnelle de l'énergie libre.

Toutefois, notamment lorsque le nombre de classes augmente, cette procédure renvoie parfois un nombre inférieur de classes à celui demandé. Nous abordons dans le prochain chapitre une façon de limiter ce problème.

# 2.A États absorbants de type II pour l'algorithme *SEM-Gibbs*

Dans cet annexe, nous formalisons les états absorbants de type II pour l'algorithme *SEM-Gibbs*.

**Théorème 2.A.1.** ***États absorbants de type II pour l'algorithme SEM-Gibbs***
*Si à une itération c, l'algorithme SEM-Gibbs simule un couple* $\left(\mathbf{z}^{(c)}, \mathbf{w}^{(c)}\right)$ *tel que :*

- *pour toute classe en ligne* $k$ *et pour toute ligne* $i$ *n'appartenant pas à la classe* $k$*, nous ayons les propriétés suivantes :*
  - *il existe une classe* $\ell(i)$ *telle que le bloc* $(k, \ell(i))$ *soit composé uniquement de la valeur* $h$*,*
  - *il existe une case* $x_{ij(i)}$ *ne valant pas* $h$*.*
- *nous ayons la version symétrique pour toutes les classes en colonne.*

*Sous ces conditions, le couple* $\left(\mathbf{z}^{(c+1)}, \mathbf{w}^{(c+1)}\right)$ *est égal à* $\left(\mathbf{z}^{(c)}, \mathbf{w}^{(c)}\right)$ *presque sûrement.*

**Démonstration :**

Si le premier paramètre $\boldsymbol{\theta}^{(0)}$ est tel que toutes les valeurs $\alpha_{k\ell}^{h\ (0)}$ soient toutes strictement positives alors, à chaque itération, les lignes et les colonnes ont une affectation presque sûrement. Il ne reste plus qu'à montrer que si le couple $\left(\mathbf{z}^{(c)}, \mathbf{w}^{(c)}\right)$ respecte les conditions du théorème 2.A.1, les affectations sont les mêmes à l'itération suivante.

Soit une classe $k$ et une ligne $i$ n'appartenant pas à la classe $k$. Par hypothèse, il existe une classe $\ell(i)$ en colonne telle que toutes les cases du bloc $(k, \ell(i))$ valent la même valeur $h$. Donc, par l'étape de mise à jour, nous avons :

$$\begin{aligned}
\alpha_{k\ell(i)}^{h} &= \frac{\sum_{i=1}^{n}\sum_{j=1}^{d} z_{ik}^{(c)} w_{j\ell(i)}^{(c)} v_{ijh}}{\sum_{i=1}^{n}\sum_{j=1}^{d} z_{ik}^{(c)} w_{j\ell(i)}^{(c)}} \\
&= \frac{\sum_{i|z_{ik}^{(c)}=1}\sum_{j|w_{j\ell(i)}^{(c)}} z_{ik}^{(c)} w_{j\ell(i)}^{(c)} v_{ijh}}{\sum_{i|z_{ik}^{(c)}=1}\sum_{j|w_{j\ell(i)}^{(c)}} z_{ik}^{(c)} w_{j\ell(i)}^{(c)}} \\
&= \frac{\sum_{i|z_{ik}^{(c)}=1}\sum_{j|w_{j\ell(i)}^{(c)}} z_{ik}^{(c)} w_{j\ell(i)}^{(c)} \times 1}{\sum_{i|z_{ik}^{(c)}=1}\sum_{j|w_{j\ell(i)}^{(c)}} z_{ik}^{(c)} w_{j\ell(i)}^{(c)}} \\
&= 1
\end{aligned}$$

et la fonction du bloc $(k, \ell(i))$ est une dirac centrée sur $h$.

Or, nous savons qu'il existe une colonne $j(i)$ telle que $x_{ij(i)}$ soit différente de $h$ ou encore que $v_{ij(i)h} = 0$

donc nous avons :

$$\begin{aligned}
\pi_k^{(c)} \prod_{\ell=1}^{m} \prod_{h'=1}^{r} \left( \left( {\alpha_{k\ell(i)}^{h'}}^{(c)} \right)^{\sum_{j=1}^{d} w_{j\ell}^{(0)} v_{ijh'}} \right) &= \pi_k^{(c)} \prod_{h'=1}^{r} \left( {\alpha_{k\ell(i)}^{h'}}^{(c)} \right)^{w_{j(i)\ell(i)}^{(0)} v_{ij(i)h'}} \\
&\quad \times \prod_{h'=1}^{r} \prod_{(\ell,j)\neq(\ell(i),j(i))} \left( \left( {\alpha_{k\ell}^{h'}}^{(c)} \right)^{w_{j\ell}^{(0)} v_{ijh'}} \right) \\
&= \pi_k^{(c)} \left( {\alpha_{k\ell(i)}^{h}}^{(c)} \right)^{w_{j(i)\ell(i)}^{(0)} v_{ij(i)h}} \prod_{h'\neq h} \left( {\alpha_{k\ell(i)}^{h'}}^{(c)} \right)^{w_{j(i)\ell(i)}^{(0)} v_{ij(i)h'}} \\
&\quad \times \prod_{h'=1}^{r} \prod_{(\ell,j)\neq(\ell(i),j(i))} \left( \left( {\alpha_{k\ell}^{h}}^{(c)} \right)^{w_{j\ell}^{(0)} v_{ijh'}} \right) \\
&= \pi_k^{(c)} 1^{w_{j(i)\ell(i)}^{(0)} \underbrace{v_{ij(i)h}}_{=0}} \underbrace{\prod_{h'\neq h} 0^{w_{j(i)\ell(i)}^{(0)} v_{ij(i)h'}}}_{=0} \\
&\quad \times \prod_{h'=1}^{r} \prod_{(\ell,j)\neq(\ell(i),j(i))} \left( \left( {\alpha_{k\ell}^{h}}^{(c)} \right)^{w_{j\ell}^{(0)} v_{ijh'}} \right) \\
&= 0
\end{aligned}$$

et la probabilité pour la ligne $i$ d'appartenir à la classe $k$ est nulle.

Comme ceci est vrai pour toute classe $k$ et pour toute ligne n'appartenant pas à la classe $k$, alors aucune ligne ne peut changer de classes.

Par symétrie et comme la matrice de partition des lignes est la même presque sûrement, nous avons le même résultat sur les colonnes.

□

*Exemple* 2.A.2.
Nous voyons sur la figure 2.1, à droite, une matrice ayant des blocs contenant qu'une seule couleur mais ne respectant pas les conditions du théorème 2.A.1, la ligne blanche peut alors passer d'une classe à l'autre.

Un autre contre-exemple dans le cas binaire est d'obtenir à un moment la matrice de paramètres $\alpha = \begin{pmatrix} 1 & 1 \\ 1 & 0.5 \end{pmatrix}$. Toutes les lignes (resp. colonnes) ont alors la possibilité d'appartenir à la deuxième classe des lignes (resp. des colonnes).

# Chapitre 3

# Approche bayésienne

*Et si vous étiez un bayésien qui s'ignore ?*
Article de Lecoutre (2005)

**Sommaire**


**3.1 Introduction** . . . . . **82**
**3.2 Inférence bayésienne** . . . . . **83**
3.2.1 Lois a priori . . . . . 83
**3.3 Algorithme *V-Bayes*** . . . . . **85**
**3.4 Échantillonneur de *Gibbs*** . . . . . **87**
**3.5 Choix empirique des paramètres $a$ et $b$** . . . . . **89**
**3.6 Critères d'arrêt** . . . . . **89**
3.6.1 Statistiques de Brooks-Gelman et améliorations . . . . . 92
3.6.2 Etude des amélioration apportées à la statistique de Brooks-Gelman . . . . . 95
**3.7 Conclusion** . . . . . **99**


## 3.1 Introduction

Comme nous l'esquissons dans la fin du chapitre précédent, l'une des approches choisies pour contourner le problème de dégénérescence des classes est d'adopter le point de vue bayésien. Nous commençons par un rappel sur l'inférence bayésienne : en quoi elle se différencie du point de vue fréquentiste, quels en sont les objectifs et quelles sont les méthodes couramment utilisées. L'inférence bayésienne est utilisée dans le cas des blocs latents comme une méthode de régularisation ; l'estimateur du mode de la distribution a posteriori servant de substitut à l'estimateur de maximum de vraisemblance.

En particulier, nous supposons que $\boldsymbol{\theta}$ n'est plus un paramètre mais une variable, tout en continuant de supposer que les nombres de classes en ligne $g$ et en colonne $m$ sont fixes ; alors que d'autres auteurs bayésiens les supposent également aléatoires (voir par exemple Van Dijk et al., 2009; Wyse et Friel, 2010). Nous explicitons le modèle bayésien, en particulier les lois a priori choisies et discutons le choix des paramètres, point à la fois fondamental et critique de l'inférence bayésienne.

Ensuite, nous étudions l'influence des paramètres des lois a priori sur l'estimation de la loi a posteriori de $\boldsymbol{\theta}$ et proposons une version bayésienne de l'algorithme *VEM*, appelée *V-Bayes*. Nous montrons qu'avec un choix judicieux de loi a priori, cette version a tendance à limiter la dégénérescence des classes.

Dans un troisième temps, puisque $\boldsymbol{\theta}$ est désormais considéré comme une variable aléatoire, nous étudions l'échantillonneur de *Gibbs* comme prolongement de l'algorithme *SEM-Gibbs*. Nous comparons leurs comportements et proposons un critère d'arrêt basé sur la statistique de Brooks-Gelman (Brooks et Gelman, 1998). Notre optique est d'estimer le mode de la loi a posteriori et non sa forme ; ce qui est suffisant pour résoudre le problème de dégénérescence.

Enfin, nous reprenons une partie des plans de simulation du chapitre 2 pour illustrer la prise en compte de la dégénérescence et proposer empiriquement la calibration des paramètres.

## 3.2 Inférence bayésienne

En statistique, deux points de vue se différencient sur leur prise en compte du statut de $\boldsymbol{\theta}$ : les fréquentistes supposent que c'est un paramètre à estimer avec une certaine incertitude et les bayésiens une variable aléatoire dont il faut estimer la loi. L'intérêt de l'inférence bayésienne est de pouvoir prendre en compte une information a priori, dans une une loi a priori $p(\boldsymbol{\theta})$, provenant d'expériences passées et d'étudier la loi a posteriori de $\boldsymbol{\theta}$ connaissant les observations $\mathbf{x}$. La règle de *Bayes* établit que :

$$p\left(\boldsymbol{\theta}|\mathbf{x}\right) = \frac{p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{x})}.$$

Comme $p(\mathbf{x})$ ne dépend pas de $\boldsymbol{\theta}$, cette relation s'écrit généralement sous forme d'une proportionnalité :

$$p\left(\boldsymbol{\theta}|\mathbf{x}\right) \propto p(\mathbf{x}|\boldsymbol{\theta})p(\boldsymbol{\theta}).$$

La calibration de la loi a priori est importante et en même temps difficile.

La forme de la loi a posteriori est souvent compliquée à estimer nécessitant l'utilisation d'approximations stochastiques par des algorithmes de type $MCMC$ (Monte Carlo Markov Chain) ; le principe étant la simulation de chaînes de Markov ayant la loi recherchée pour loi stationnaire. En particulier, nous citons deux d'entre eux :

- l'algorithme de Metropolis-Hastings (Metropolis et al., 1953) utilisé lorsque la loi *cible* est connue à une normalisation près ; cet algorithme a l'avantage de pouvoir être utilisé dans presque tous les cas sans trop de contraintes mais converge souvent lentement (voir par exemple Robert, 2006, section 6.3.2) ;
- l'échantillonneur de *Gibbs* décrit par Geman et Geman (1984) lorsqu'il est possible de simuler une partie de $\boldsymbol{\theta}$ connaissant l'autre partie. Il a déjà été utilisé dans la section 2.4 pour estimer la loi de $(\mathbf{z}, \mathbf{w})$ condititionnellement aux observations dans l'étape $SE$ de l'algorithme *SEM-Gibbs*.

L'échantillonneur de *Gibbs* est étudié pour le cas du modèle des blocs latents en section 3.4.

Dans la suite, nous nous intéressons plus particulièrement au mode de la loi a posteriori.

### 3.2.1 Lois a priori

Lorsque les lois des observations appartiennent à la famille exponentielle :

$$p(\mathbf{x}|\boldsymbol{\theta}) = b(\mathbf{x}) \exp\left[\langle \boldsymbol{\theta}, R(\mathbf{x})\rangle - \Psi(\boldsymbol{\theta})\right]$$

où $b$, $R$ et $\Psi$ sont des fonctions et $\langle \cdot, \cdot \rangle$ représente le produit scalaire ; un choix naturel possible de lois a priori est celui des lois conjuguées (Raiffa et Schlaifer, 1961) s'écrivant sous la forme :

$$p(\boldsymbol{\theta}) \propto \exp\left[\langle \boldsymbol{\theta}, \xi\rangle - \lambda\Psi(\boldsymbol{\theta})\right]$$

où $\xi$ et $\lambda$ sont des paramètres permettant d'exprimer l'information a priori.

En particulier, les lois conjuguées des lois multinomiales sont les lois de Dirichlet et celles des lois de Bernoulli, les lois bêta. Une nouvelle fois, nous n'avons pas de raisons de privilégier une classe par rapport à une autre et nous choisissons de prendre le même paramètre $a$ pour toutes les lois a priori sur les proportions :

$$\boldsymbol{\pi} \sim \mathcal{D}ir(a, \ldots, a) \quad \text{et} \quad \boldsymbol{\rho} \sim \mathcal{D}ir(a, \ldots, a).$$

Pour les variables $\alpha_{k\ell}$, nous supposons qu'elles sont indépendantes et de même loi :

$$\forall (k, \ell) \in \{1, \ldots, g\} \times \{1, \ldots, m\}, \quad \alpha_{k\ell} \sim \mathcal{D}ir(b, \ldots, b).$$

La représentation graphique de la figure 3.1 résume la modélisation bayésienne.

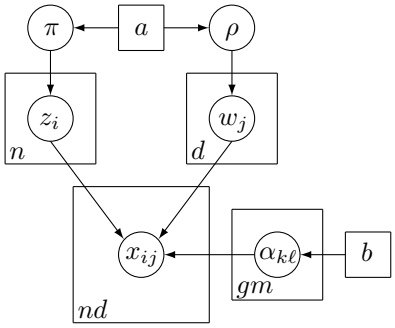



FIGURE 3.1 – Représentation graphique du modèle bayésien associé au modèle des blocs latents.

Le choix des paramètres $a$ et $b$ est important (Kass et Wasserman, 1996), sur la figure 3.2 sont représentées différentes densités de lois bêta $\mathcal{B}e(b, b)$ suivant les valeurs de $b$ :

(a) si $b < 1$, la densité met du poids sur les valeurs extrêmes. En particulier, c'est le cas pour la loi a priori non informative de Jeffreys (voir Jeffreys, 1946) avec $b = \frac{1}{2}$,

(b) si $b = 1$, loi a priori non informative de Laplace (1825), la densité est uniforme sur tous les paramètres,

(c) si $b > 1$, la densité met plus de poids sur les valeurs centrales. Des valeurs équilibrées sont alors plus probables qu'une forte disproportion.

Ces constatations sont également vraies pour le paramètre $a$, la loi bêta étant un cas particulier avec deux paramètres de la loi de Dirichlet.

Dans le cas de mélanges gaussiens simples, Frühwirth-Schnatter (2011) propose de prendre pour les paramètres des lois a priori des proportions des valeurs de $a$ valant 4 ou 16.5 suivant s'il y a peu ou beaucoup d'observations. Ce choix, testé empiriquement, permet aux algorithmes $MCMC$ de produire moins de classes vides.

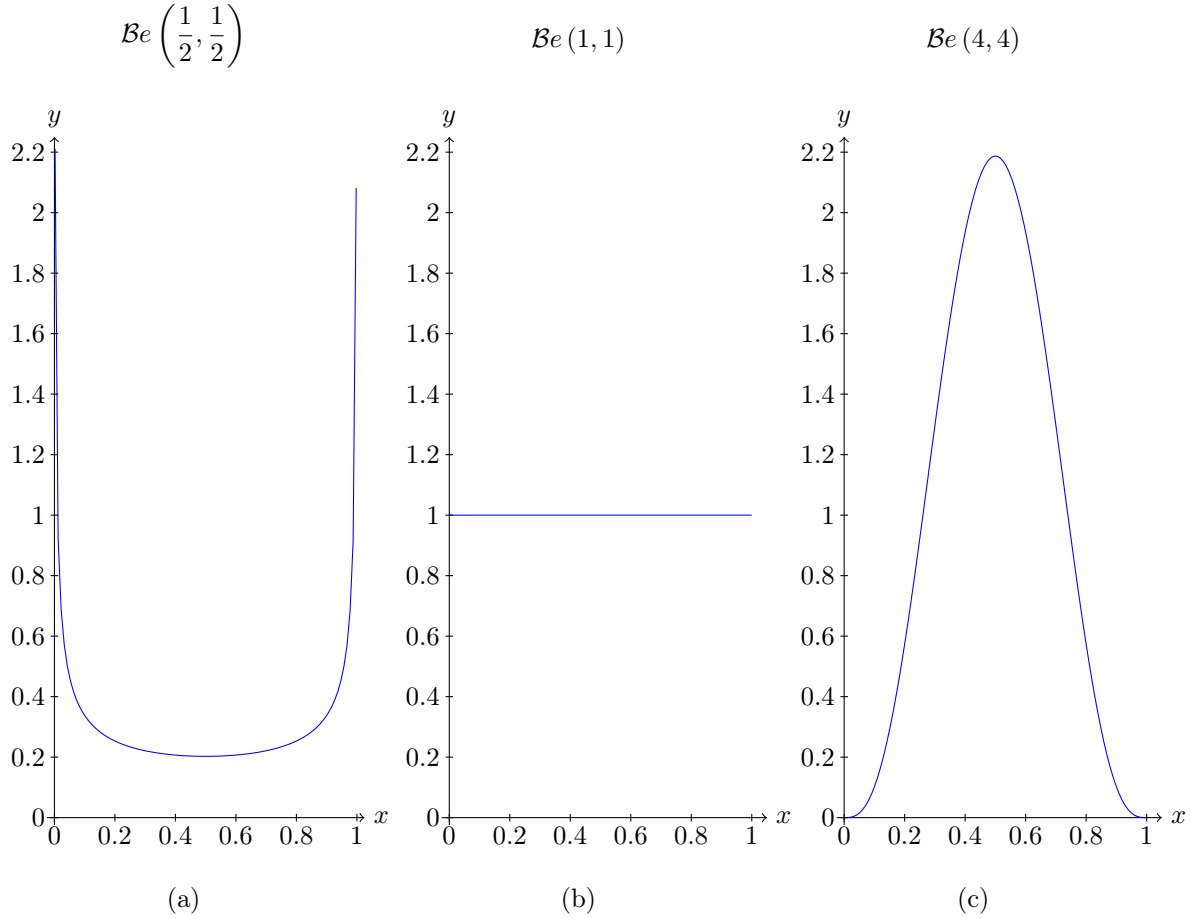



FIGURE 3.2 – Densité de lois bêta $\mathcal{B}e(b,b)$ pour différentes valeurs de $b$ : 1/2 pour la figure (a), 1 pour la figure (b) et 4 pour la figure (c).

## 3.3 Algorithme *V-Bayes*

Pour estimer le mode de la loi a posteriori de $\boldsymbol{\theta}$, la même démarche que pour l'algorithme *EM* peut être faite (McLachlan et Krishnan, 2008) :

$$\begin{aligned}\log p(\mathbf{x},\boldsymbol{\theta}) &= Q\left(\boldsymbol{\theta}\left|\boldsymbol{\theta}^{(c)}\right.\right) - H\left(\boldsymbol{\theta}\left|\boldsymbol{\theta}^{(c)}\right.\right) + \log p(\boldsymbol{\theta}) \\ &= Q_B\left(\boldsymbol{\theta}\left|\boldsymbol{\theta}^{(c)}\right.\right) - H\left(\boldsymbol{\theta}\left|\boldsymbol{\theta}^{(c)}\right.\right).\end{aligned}$$

où $Q$ et $H$ sont définies en section 2.2 et $Q_B\left(\boldsymbol{\theta}\left|\boldsymbol{\theta}^{(c)}\right.\right) = Q\left(\boldsymbol{\theta}\left|\boldsymbol{\theta}^{(c)}\right.\right) + \log p(\boldsymbol{\theta})$.

Or, comme pour l'algorithme *EM* de la section 2.2, le calcul de $Q_B\left(\boldsymbol{\theta}\left|\boldsymbol{\theta}^{(c)}\right.\right)$ est impossible et nous faisons la même approximation variationnelle que pour l'algorithme *VEM*. Nous obtenons ainsi l'algorithme

*V-Bayes* cherchant à maximiser l'énergie libre $\mathcal{F}_B$ définie par :

$$\mathcal{F}_B(\boldsymbol{\theta}) = \mathcal{F}(\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}).$$

---

**Algorithme *V-Bayes*** :

---

1. Initialisation de $\boldsymbol{\theta}^{(0)}$ et $t_{j\ell}^{(0)}$.
2. Itération jusqu'à ce que $\mathcal{F}_B\left(q_{\mathbf{zw}}^{(c)}; \boldsymbol{\theta}^{(c)}\right)$ n'évolue plus :
   (a) Étape $VE$ : de la même façon que pour l'algorithme *VEM* de la section 2.3, maximisation alternée de l'énergie libre à $\boldsymbol{\theta}^{(c)}$ fixé en prenant $t_{j\ell}^{(t=0)} = t_{j\ell}^{(c)}$ :
      i. calcul de $s_{ik}^{(t+1)}$ à $t_{j\ell}^{(t)}$ et à $\boldsymbol{\theta}^{(c)}$ fixé.
      ii. calcul de $t_{j\ell}^{(t+1)}$ à $s_{ik}^{(t+1)}$ et à $\boldsymbol{\theta}^{(c)}$ fixé.

      Obtention des probabilités $s_{ik}^{(c+1)}$ et $t_{j\ell}^{(c+1)}$, les dernières calculées.
   (b) Étape $M$ : calcul de $\boldsymbol{\theta}^{(c+1)}$ :

$$\pi_k^{(c+1)} = \frac{a - 1 + \sum_{i=1}^{n} s_{ik}^{(c+1)}}{g(a-1) + n}, \quad \rho_\ell^{(c+1)} = \frac{a - 1 + \sum_{j=1}^{d} t_{j\ell}^{(c+1)}}{m(a-1) + d}$$

$$\text{et } \alpha_{k\ell}^{h\ (c+1)} = \frac{b - 1 + \sum_{i=1}^{n}\sum_{j=1}^{d} s_{ik}^{(c+1)} t_{j\ell}^{(c+1)} v_{ijh}}{r\,(b-1) + \sum_{i=1}^{n}\sum_{j=1}^{d} s_{ik}^{(c+1)} t_{j\ell}^{(c+1)}}.$$

3. Obtention d'un estimateur $\widehat{\theta}^{VB} = \boldsymbol{\theta}^{(\infty)}$.
4. Calcul des estimateurs des classes en ligne $\widehat{z}^{VB}$ et en colonne $\widehat{w}^{VB}$ tels que :

$$\widehat{z}_i^{VB} \in \underset{k=1,\dots,g}{\operatorname{argmax}}\, s_{ik}^{(\infty)} \quad \text{et} \quad \widehat{w}_j^{VB} \in \underset{\ell=1,\dots,m}{\operatorname{argmax}}\, t_{j\ell}^{(\infty)}$$

---

*Remarque* 3.3.1.
Nous mettons en rouge dans le formulaire l'apport de l'approche bayésienne :

— si $a = b = 1$, c'est-à-dire sous des lois a priori uniformes, les algorithmes *V-Bayes* et *VEM* sont identiques,

— si $a > 1$, les estimations des paramètres $\pi_k^{(c)}$ sont minorées par $\frac{a-1}{g(a-1)}$. Ainsi, les cas où les estimations tendent vers 0 sont évités (comme sur les figures 2.3 et 3.3, gauche).

En reprenant les conditions d'expérience de la figure 2.3 mais en utilisant l'algorithme *V-Bayes* avec $a = 4$ et $b = 1$ pour les mêmes initialisations, la figure 3.3, à droite, montre que la trajectoire de la probabilité représentée par un trait plein a également commencé par tendre vers 0 mais s'est stabilisée à une certaine valeur avant de remonter et de devenir la plus forte des probabilités.

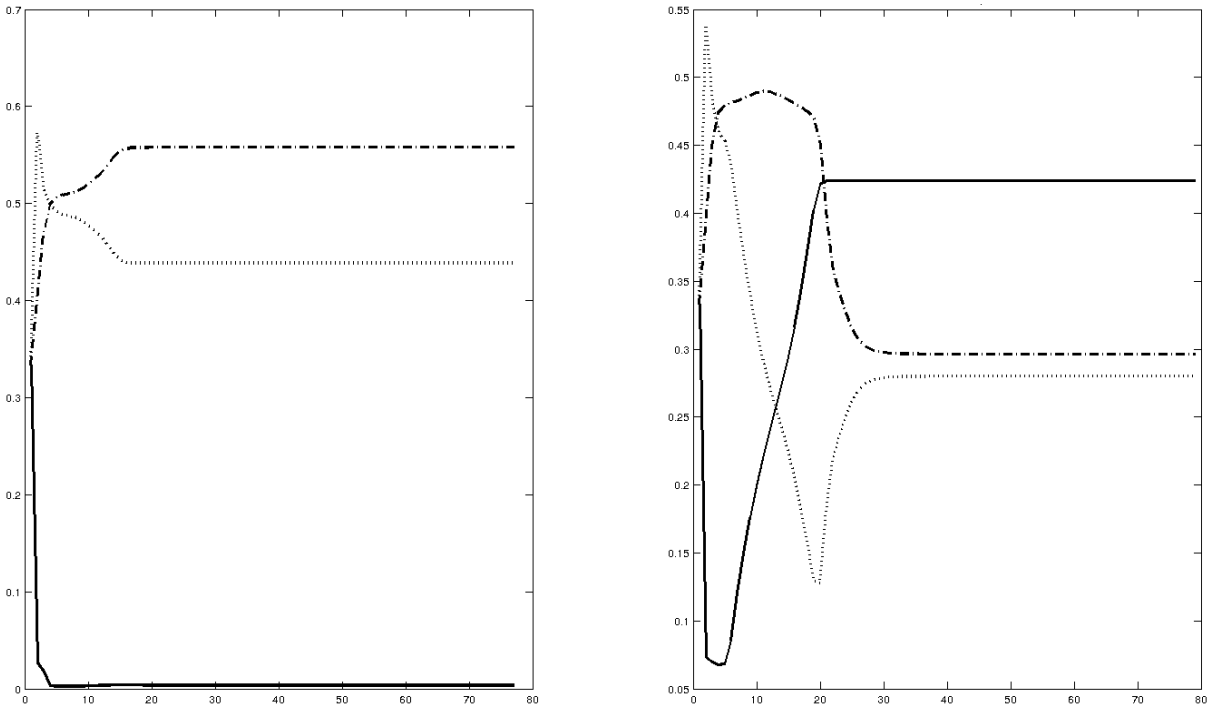



FIGURE 3.3 – Evolution des estimations des $\pi_k^{(c)}$ pour l'algorithme *VEM* (à gauche) et pour l'algorithme *V-Bayes* avec $a = 4$ et $b = 1$ (à droite) en partant de la même initialisation. Les estimations de la probabilité symbolisée par un trait plein se rapprochent progressivement vers 0 pour l'algorithme *VEM* tandis qu'elles ne descendent pas en dessous d'un certain seuil pour l'algorithme *V-Bayes*.

## 3.4 Échantillonneur de *Gibbs*

Plutôt que de faire une approximation variationnelle, un algorithme type $MCMC$ peut être utilisé pour estimer la loi a posteriori. Pour cela, nous étudions l'échantillonneur de *Gibbs* défini ainsi :

**Échantillonneur de *Gibbs* :**

1. Initialisation de $\boldsymbol{\theta}^{(0)}$ et de $\mathbf{w}^{(0)}$.
2. Itération *niter* fois :
   (a) Simulation de $\mathbf{z}^{(c+1)}$ suivant la loi $p\left(\mathbf{z}\left|\mathbf{x}, \mathbf{w}^{(c)}; \theta^{(c)}\right.\right)$.
   (b) Simulation de $\mathbf{w}^{(c+1)}$ suivant la loi $p\left(\mathbf{w}\left|\mathbf{x}, \mathbf{z}^{(c+1)}; \theta^{(c)}\right.\right)$.
   (c) Simulation de $\boldsymbol{\pi}^{(c+1)}$ suivant la loi $\boldsymbol{\pi}\left|\mathbf{z}^{(c+1)}\right. \sim \mathcal{D}\left(z_{+1}^{(c+1)}+a, \ldots, z_{+g}^{(c+1)}+a\right)$ avec $z_{+k}^{(c+1)} = \sum_i z_{ik}^{(c+1)}$.
   (d) Simulation de $\boldsymbol{\rho}^{(c+1)}$ suivant la loi $\boldsymbol{\rho}\left|\mathbf{w}^{(c+1)}\right. \sim \mathcal{D}\left(w_{+1}^{(c+1)}+a, \ldots, w_{+m}^{(c+1)}+a\right)$ avec $w_{+\ell}^{(c+1)} = \sum_j w_{j\ell}^{(c+1)}$.
   (e) Simulation de $\boldsymbol{\alpha}^{(c+1)}$ suivant la loi

$$\alpha_{k\ell}\left|\mathbf{x}, \mathbf{z}^{(c+1)}, \mathbf{w}^{(c+1)}\right. \sim \mathcal{D}\left(N^1_{k\ell;\mathbf{z},\mathbf{w}}{}^{(c+1)}+b, \ldots, N^r_{k\ell;\mathbf{z},\mathbf{w}}{}^{(c+1)}+b\right)$$

   avec $N^h_{k\ell;\mathbf{z},\mathbf{w}}{}^{(c+1)} = \sum_{i,j} z_{ik}^{(c+1)} w_{j\ell}^{(c+1)} v_{ijh}$ le nombre de cases valant $h$ dans le bloc $(k,\ell)$.
3. Obtention d'un estimateur $\widehat{\theta}^G = \frac{1}{niter}\sum_{c=1}^{niter} \boldsymbol{\theta}^{(c)}$ en moyennant les $\boldsymbol{\theta}^{(c)}$ simulés.
4. Calcul des estimateurs des classes en ligne $\widehat{z}^G$ et en colonne $\widehat{w}^G$ par affectation à la classe majoritairement affectée durant les simulations :

$$\widehat{z}_i^G \in \underset{k=1,\ldots,g}{\operatorname{argmax}} \sum_{c=1}^{niter} z_{ik}^{(c)} \quad \text{et} \quad \widehat{w}_j^G \in \underset{\ell=1,\ldots,m}{\operatorname{argmax}} \sum_{c=1}^{niter} w_{j\ell}^{(c)}$$

Deux chaînes de Markov $\left(\boldsymbol{\theta}^{(c)}\right)$ et $\left(\mathbf{z}^{(c)}, \mathbf{w}^{(c)}\right)$ ergodiques de lois invariantes $p(\boldsymbol{\theta}|\mathbf{x})$ et $p(\mathbf{z}, \mathbf{w}|\mathbf{x})$ respectivement sont ainsi obtenues. De plus, comme l'espace des valeurs du couple $(\mathbf{z}, \mathbf{w})$ est fini, la convergence vers la loi stationnaire est *uniformément géométrique* (Billingsley, 1986), c'est-à-dire qu'il existe $C > 0$ et $\mathfrak{r} \in ]0,1[$ tels que :

$$d_{VT}\left(p^{(c)}(\cdot|\mathbf{x}), p(\cdot|\mathbf{x})\right) \leq C\mathfrak{r}^c$$

où $d_{VT}$ représente la distance en variation totale. Par le *principe de dualité* (voir par exemple Robert, 2006, exercice 6.28), la convergence vers la loi stationnaire, qui est la loi a posteriori de $\boldsymbol{\theta}$, est également uniformément géométrique pour la chaîne $\left(\boldsymbol{\theta}^{(c)}\right)$.

L'échantillonneur de *Gibbs* est encore moins sensible aux initialisations que l'algorithme *SEM-Gibbs* ; en contrepartie, il est beaucoup plus fluctuant et sensible au label switching : une fois l'ensemble des valeurs obtenues, il est conseillé d'utiliser les conditions suffisantes des critères d'identifiabilité de Keribin et al. pour réordonner les simulations avant d'estimer les paramètres et les partitions ; en revanche, il est déconseillé de le faire durant la simulation, la chaîne de Markov associée serait alors biaisée (Celeux et al., 2000). De même, un temps de chauffe est recommandé pour limiter l'influence des premières simulations.

La moyenne peut être vue comme une approximation de l'estimateur du maximum de la loi a posteriori.

*Remarque* 3.4.1.
Les $\theta^{(c)}$ simulés sont corrélés entre eux. Pour diminuer cette influence, nous pourrions prendre une valeur tous les pas d'itérations fixé.

Nous obtenons ainsi la correspondance suivante :

$$\begin{array}{ccc} \textit{VEM} & \rightarrow & \textit{V-Bayes} \\ \textit{SEM-Gibbs} & \rightarrow & \textit{Gibbs} \end{array}$$

et pouvons proposer la même combinaison que pour le chapitre 2.

**Comparaison avec l'algorithme *SEM-Gibbs***
D'un point de vue algorithmique, deux remarques sont à faire :

- les lois étant continues, la probabilité d'obtenir des paramètres $\alpha_{k\ell}^h$ valant 0 ou 1 nécessaire pour être dans un état absorbant de type II pour l'algorithme *SEM-Gibbs* (voir la remarque 2.4.2) est nulle,
- si une classe en ligne ou en colonne se vide, elle peut de nouveau se remplir. L'état absorbant de type I pour l'algorithme *SEM-Gibbs* n'en est donc pas un pour l'échantillonneur de *Gibbs*.
  En particulier, si une classe en ligne $k$ ou en colonne $\ell$ est vide, les $r$-uplet $\alpha_{k\ell}$ correspondants sont simulés suivant des lois de Dirichlet $\mathcal{D}(b, \ldots, b)$ ; si nous prenons par exemple $b = 1$, nous obtenons une loi uniforme sur tous les $r$-uplets et nous pouvons ainsi obtenir une configuration totalement différente.

## 3.5 Choix empirique des paramètres $a$ et $b$

Dans cette partie, nous expérimentons le choix des valeurs des paramètres $a$ et $b$. Pour ce faire, reprenant les matrices simulées dans la partie 2.6.1, nous utilisons l'échantillonneur de *Gibbs* en prenant 5 000 itérations combiné avec l'algorithme *V-Bayes* pour des paramètres $a$ et $b$ valant 1, 4 ou 16 (en enlevant le couple $(1, 1)$) à partir des mêmes initialisations.

Les résultats pour les cas de proportions identiques (tableau 3.1) et différentes (tableau 3.2) sont sensiblement les mêmes.

Prendre $a$ valant 4 ou 16 a un effet bénéfique contre la dégénérescence des classes ; plus particulièrement pour le cas où le nombre de classes demandé est supérieur à celui ayant servi pour la simulation. Ceci est en accord avec les remarques de la section 3.3 car l'algorithme *V-Bayes* ne va pas laisser de classes se vider durant la simulation.

En revanche, prendre $b > 1$ accentue le phénomène de dégénérescence des classes. En effet, ces valeurs de paramètre favorisent des $\alpha_{k\ell}$ proches de 0.5, et par la même occasion la ressemblance de configurations et un nombre moindre de classes distinctes.

Au vu de ces résultats, nous proposons de prendre pour les paramètres $a = 4$ et $b = 1$.

## 3.6 Critères d'arrêt

Jusqu'à présent, nous avons fixé à 5 000 le nombre d'itérations pour l'échantillonneur de *Gibbs* ; ce choix arbitraire peut parfois être trop grand, parfois trop petit suivant le type des matrices. Dans cette partie, nous proposons un critère d'arrêt pour stopper l'échantillonneur de *Gibbs*.

**(g,m)=(5,4)**

| | + | | | ++ | | | +++ | | |
|---|---|---|---|---|---|---|---|---|---|
| $a \backslash b$ | 1 | 4 | 16 | 1 | 4 | 16 | 1 | 4 | 16 |
| (100, 200) | | | | | | | | | |
| 1 | 5.6 | 38 | 65.6 | 7.2 | 20.6 | 56.8 | 13.8 | 11.8 | 48.6 |
| 4 | 0.4 | 1 | 14.8 | 0.6 | 0.4 | 0.4 | 0.8 | 0.4 | 0.2 |
| 16 | 0.2 | 0.2 | 1.8 | 0 | 0.4 | 0.2 | 0.2 | 0.2 | 0 |
| (150, 150) | | | | | | | | | |
| 1 | 5.4 | 16.2 | 47.4 | 5.6 | 8.4 | 36 | 7.4 | 5.2 | 39.2 |
| 4 | 0 | 0 | 2 | 0.2 | 0 | 0 | 0.2 | 0 | 0.2 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0.2 |
| (200, 100) | | | | | | | | | |
| 1 | 3.2 | 12.2 | 49.2 | 5.6 | 4.2 | 34.8 | 10.4 | 6.6 | 44.4 |
| 4 | 0.2 | 0 | 4.2 | 0 | 0.2 | 0.2 | 0 | 0 | 0 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

**(g,m)=(8,8)**

| | + | | | ++ | | | +++ | | |
|---|---|---|---|---|---|---|---|---|---|
| $a \backslash b$ | 1 | 4 | 16 | 1 | 4 | 16 | 1 | 4 | 16 |
| (100, 200) | | | | | | | | | |
| 1 | 57.6 | 88.4 | 100 | 41.8 | 57.2 | 100 | 43.2 | 59 | 100 |
| 4 | 0 | 0.8 | 100 | 0 | 0.8 | 98.2 | 1.8 | 3.6 | 93.2 |
| 16 | 0 | 1.6 | 93.8 | 0.2 | 1 | 15 | 1.6 | 3 | 24.2 |
| (150, 150) | | | | | | | | | |
| 1 | 61.2 | 68.4 | 100 | 48.4 | 35.2 | 100 | 44 | 46 | 100 |
| 4 | 0 | 0.4 | 100 | 0 | 0 | 84 | 0.8 | 1.4 | 77.2 |
| 16 | 0 | 0 | 29.4 | 0 | 0 | 2 | 0.2 | 1.8 | 14.8 |
| (200, 100) | | | | | | | | | |
| 1 | 74.8 | 90 | 100 | 72 | 58.8 | 100 | 68.2 | 59.8 | 100 |
| 4 | 0 | 0.6 | 100 | 0 | 0.2 | 99.2 | 0.8 | 1.8 | 95.6 |
| 16 | 0 | 0.2 | 99.8 | 0 | 0 | 11.6 | 0.4 | 2 | 19.8 |

TABLE 3.1 – Pourcentages de classes vides, pour des données simulées avec des proportions égales, obtenues par l'algorithme *SEM+VEM* (cases $(a, b) = (1, 1)$) et par l'échantillonneur de *Gibbs* combiné avec l'algorithme *V-Bayes* initialisés avec $(g, m) = (5, 4)$ classes (en haut) et $(g, m) = (8, 8)$ classes (en bas), pour 500 matrices dans des cas faciles (+), moyens (++) et difficiles (+++) (en colonne) pour différentes tailles d'échantillons (en ligne).

**(g,m)=(5,4)**

| | + | | | ++ | | | +++ | | |
|---|---|---|---|---|---|---|---|---|---|
| $(100, 200)$ | a \ b | 1 | 4 | 16 | 1 | 7 | 23.2 | 73.4 | 16.8 | 18.8 | 74.8 |
| | 4 | 0.8 | 2.6 | 30.4 | 4 | 1.6 | 1.8 | 0.8 | 2.2 | 2.2 | 0.8 |
| | 16 | 1 | 0.8 | 9.4 | 16 | 0.6 | 1 | 1.2 | 1 | 1.2 | 2.6 |
| $(150, 150)$ | 1 | 7.6 | 29.2 | 71.8 | 1 | 13 | 11.4 | 62.6 | 17.6 | 8.4 | 65 |
| | 4 | 1 | 0 | 6.8 | 4 | 0.6 | 0.6 | 0.6 | 0.6 | 1.2 | 0.6 |
| | 16 | 0.2 | 0.6 | 0.2 | 16 | 0.2 | 1.4 | 0.8 | 0.4 | 0.8 | 0 |
| $(200, 100)$ | 1 | 13.2 | 20 | 71.8 | 1 | 11.8 | 18.4 | 67.6 | 18.2 | 7.8 | 69.6 |
| | 4 | 1 | 0.2 | 12 | 4 | 0.4 | 0.8 | 0.4 | 0.8 | 1 | 0.6 |
| | 16 | 0.6 | 0.4 | 0.6 | 16 | 0.4 | 0.8 | 0.2 | 0 | 0.4 | 0 |

**(g,m)=(8,8)**

| | + | | | ++ | | | +++ | | |
|---|---|---|---|---|---|---|---|---|---|
| $(100, 200)$ | a \ b | 1 | 4 | 16 | 1 | 54.6 | 82.6 | 100 | 47.4 | 84.2 | 100 |
| | 4 | 1.2 | 5.8 | 100 | 4 | 1.8 | 3.6 | 98.2 | 3 | 10 | 97.6 |
| | 16 | 2.2 | 3.8 | 99 | 16 | 2.4 | 4.4 | 25.2 | 5.2 | 13.2 | 31 |
| $(150, 150)$ | 1 | 69.2 | 89 | 100 | 1 | 54.2 | 64.6 | 100 | 57.2 | 69.4 | 100 |
| | 4 | 0.4 | 1.2 | 100 | 4 | 0.2 | 1.4 | 91.6 | 2.2 | 5.8 | 86.8 |
| | 16 | 2.6 | 2.2 | 34 | 16 | 1.4 | 2.2 | 6 | 3 | 7.4 | 31.8 |
| $(200, 100)$ | 1 | 81.4 | 96 | 100 | 1 | 67.8 | 76.8 | 100 | 68.2 | 73 | 100 |
| | 4 | 1.4 | 2.6 | 100 | 4 | 1.2 | 2.8 | 99 | 2.2 | 5.8 | 94.2 |
| | 16 | 2.2 | 4.6 | 99.2 | 16 | 2.4 | 3 | 23 | 3.8 | 9 | 27.8 |

TABLE 3.2 – Pourcentages de classes vides, pour des données simulées avec des proportions inégales, obtenues par l'algorithme *SEM+VEM* (cases $(a, b) = (1, 1)$) et par l'échantillonneur de *Gibbs* combiné avec l'algorithme *V-Bayes* initialisés avec $(g, m) = (5, 4)$ classes (en haut) et $(g, m) = (8, 8)$ classes (en bas), pour 500 matrices dans des cas faciles (+), moyens (++) et difficiles (+++) (en colonne) pour différentes tailles d'échantillons (en ligne).

### 3.6.1 Statistiques de Brooks-Gelman et améliorations

Afin de déterminer le nombre d'itérations nécessaires, nous proposons l'utilisation de la statistique de Brooks-Gelman (Brooks et Gelman, 1998) qui permet d'estimer le moment où la chaîne simulée a exploré la loi stationnaire. Le principe est de simuler $\mathcal{T}$ chaînes, chacune indicée $\mathfrak{t}$, en parallèle et de regarder pour chaque composante de chaque paramètre de $\boldsymbol{\theta}$, notée $\xi$ par la suite, le moment où les chaînes associées ont les mêmes lois empiriques. Plus formellement, la procédure est ainsi définie :

---

**Critère d'arrêt pour l'échantillonneur de *Gibbs*** :

---

1. Simulation de $\mathcal{T}$ chaînes en parallèle par la procédure de l'échantillonneur de *Gibbs*.
2. Toutes les $\mathfrak{N}$ itérations, pour chaque paramètre $\xi$ :
   (a) Pour chaque trajectoire $\xi_{\mathfrak{t}} = \left\{\xi_{\mathfrak{t}}^1, \ldots, \xi_{\mathfrak{t}}^{\mathfrak{N}}\right\}$, calcul de la différence $\delta_{\mathfrak{t}}^{\xi}$ entre les quantiles empiriques de niveau 97.5% et 2.5%.
   (b) Calcul de la différence $\Delta$ entre les quantiles empiriques de niveau 97.5% et 2.5% pour l'échantillon complet $\{\xi_1, \ldots, \xi_{\mathcal{T}}\}$.
   (c) Estimation de la statistique de Brooks-Gelman :
   $$\widehat{R}_{BG;\xi} = \frac{\Delta}{\overline{\delta}^{\xi}}$$
   où $\overline{\delta}^{\xi} = \frac{1}{\mathcal{T}} \sum_{\mathfrak{t}=1}^{\mathcal{T}} \delta_{\mathfrak{t}}^{\xi}$ est la moyenne empirique des $\delta_{\mathfrak{t}}^{\xi}$.
3. Si toutes les statistiques $\widehat{R}_{BG;\xi}$ sont plus petites que 1.2, arrêt de la chaîne, sinon prolongement des chaînes en reprenant à l'étape 1.

---

*Remarque* 3.6.1.
Le principe de ce critère vient du fait que si toutes les chaînes ont à peu près la même loi empirique alors pour tout $\mathfrak{t}$, $\Delta \approx \delta_{\mathfrak{t}}^{\xi}$ et ainsi $\widehat{R}_{BG;\xi} \approx 1$ ; ce qui stoppe l'échantillonneur.

*Remarque numérique* 3.6.2.
La figure 3.4 représente des trajectoires de $\mathcal{T} = 10$ chaînes pour une même composante $\xi$ de paramètre sur des données simulées. Nous savons que le maximum global est situé aux alentours de 0.3. En dehors des chaînes 1, 3, 6, 7, 8 et 9 qui oscillent dès les premières itérations autour de la valeur attendue, deux comportements peuvent ralentir la convergence :

(a) les chaînes 4, 5 et 10 oscillent d'abord autour d'une situation stationnaire puis autour de la valeur attendue. Pour chaque chaîne $\mathfrak{t}$, tant que sa longueur n'est pas plus grande que 20 fois le nombre d'itérations correspondant à la première situation stationnaire, la valeur $\delta_{\mathfrak{t}}^{\xi}$ sera plus grande que la valeur moyenne.

(b) la chaîne 2 oscille durant toute sa trajectoire autour d'une valeur qui n'est pas proche de la valeur attendue. Lorsque cette chaîne changera de régime à l'itération $c$ et en admettant que toutes les autres chaînes soient autour de la valeur attendue dès le début, il faudra attendre d'avoir simulé pour chaque chaîne $20c/\mathfrak{N}$ itérations supplémentaires pour que la valeur $\Delta$ ne soit plus influencée par cette partie de la chaîne.

Pour pallier ces ralentissements, nous proposons trois améliorations ; la première étant celle proposée par Fu (2012) qui suggère de comparer les statistiques modifiées à la valeur de 1.05 :

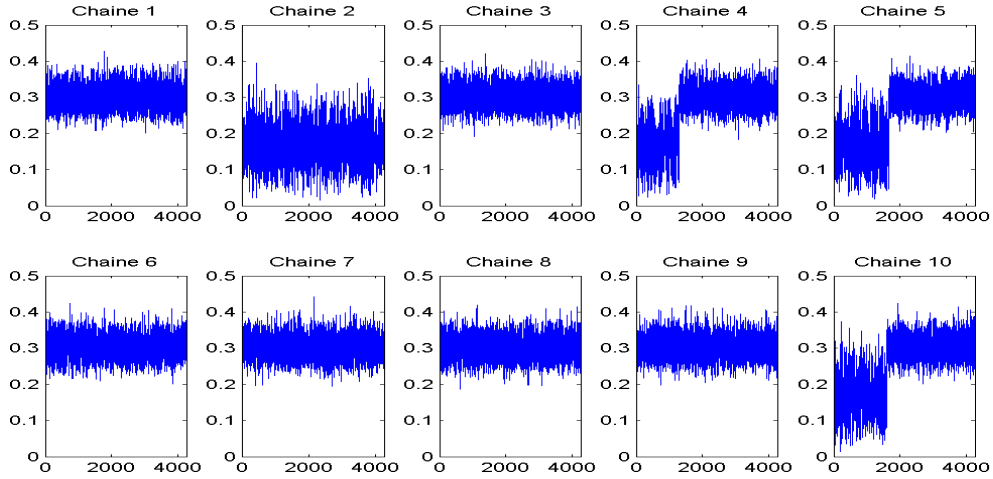



FIGURE 3.4 – Représentation de 10 chaînes pour un paramètre $x_i$.

---

**Amélioration du critère d'arrêt pour l'échantillonneur de *Gibbs* :**

---

1. Simulation de $\mathcal{T}$ chaînes, chacune indicée $\mathfrak{t}$, en parallèle par la procédure de l'échantillonneur de *Gibbs* comme précédemment.
2. Toutes les $\mathfrak{N}$ itérations, calcul des $\widehat{R}_{BG;\xi}$ de la même façon que précédemment.
3. Si toutes les statistiques $\widehat{R}_{BG;\xi}$ sont plus petites que 1.05, arrêt de la chaîne, sinon :
   (a) calcul, pour chaque $\mathfrak{t}$, de la statistique $\widehat{R}^{-\mathfrak{t}}_{BG;\xi}$ sur le même principe mais en enlevant la chaîne $\mathfrak{t}$,
   (b) calcul de la statistique $\widehat{R}^{-\zeta\%}_{BG;\xi}$ en enlevant pour chaque chaîne les $\zeta\%$ premières itérations pour une valeur fixée de $\zeta$,
   (c) calcul, pour chaque $\mathfrak{t}$, de la statistique $\widehat{R}^{-\mathfrak{t},\zeta\%}_{BG;\xi}$ en couplant les deux points précédents.
4. Si pour l'une de ces procédures, toutes les statistiques sont plus petites que 1.05, arrêt de la chaîne en conservant les estimations correspondantes, sinon prolongement des chaînes en reprenant à l'étape 1.

---

Le point $(a)$ permet de se débarrasser d'une chaîne bloquée dans un maximum local. Fu (2012) montre que cette amélioration diminue la statistique avec moins d'itérations. Le principe du point $(b)$ et, a fortiori, du point $(c)$ est de supposer que le temps de chauffe n'est pas assez long ; ils permettent dans ce cas de diminuer le temps de convergence.

*Exemple* 3.6.3. La figure 3.5 représente l'application de ces améliorations à partir de l'exemple de la figure 3.4. La chaîne écartée est la 2, les chaînes restantes se ressemblent (figure (a)).
L'augmentation du temps de chauffe (trait rouge des figures (b) et (c)) enlève surtout des simulations provenant de l'état stationnaire ralentissant la convergence de l'algorithme.

(a)

(b)

(c)

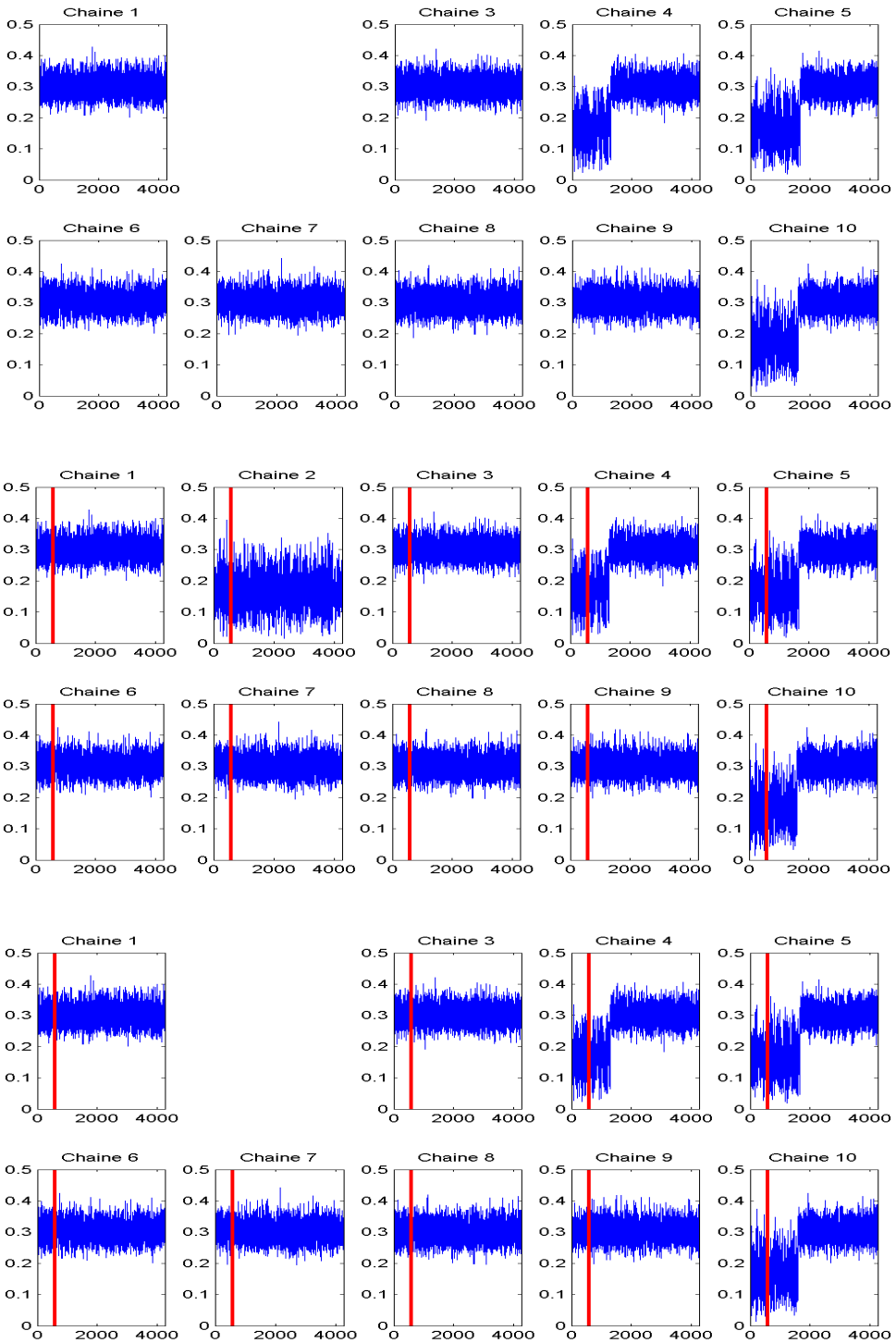



FIGURE 3.5 – Représentation des différentes procédures pour le calcul de la statistique modifiée de Brooks-Gelman : en enlevant la chaîne 2 (a), en augmentant le temps de chauffe (trait rouge de (b)) et en couplant les deux (c).

### 3.6.2 Etude des amélioration apportées à la statistique de Brooks-Gelman

La question est alors de vérifier si les améliorations proposées précédemment diminuent réellement le temps de convergence par rapport à la forme première de l'utilisation de la statistique de Brooks-Gelman sans dégrader la qualité des estimations. Pour cela, le plan de simulations reprend les mêmes paramètres qu'en section 2.6.1 : $g = 5$ classes en ligne et $m = 4$ en colonne avec des proportions équilibrées et la matrice des paramètres $\boldsymbol{\alpha}$ suivante :

$$\boldsymbol{\alpha} = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & 1-\varepsilon \end{pmatrix}$$

en faisant varier $\varepsilon$ pour obtenir des matrices de difficultés différentes suivant le protocole de Lomet et al. (2012c).
Dans le cas de matrices de tailles $(100, 200)$ et $(200, 100)$ et pour chaque niveau de difficulté, 500 matrices sont générées et utilisées pour comparer l'échantillonneur de *Gibbs* en arrêtant après 5 000 itérations (base), avec la statistique de Brooks-Gelman (BG) et avec la statistique améliorée (Amelior).

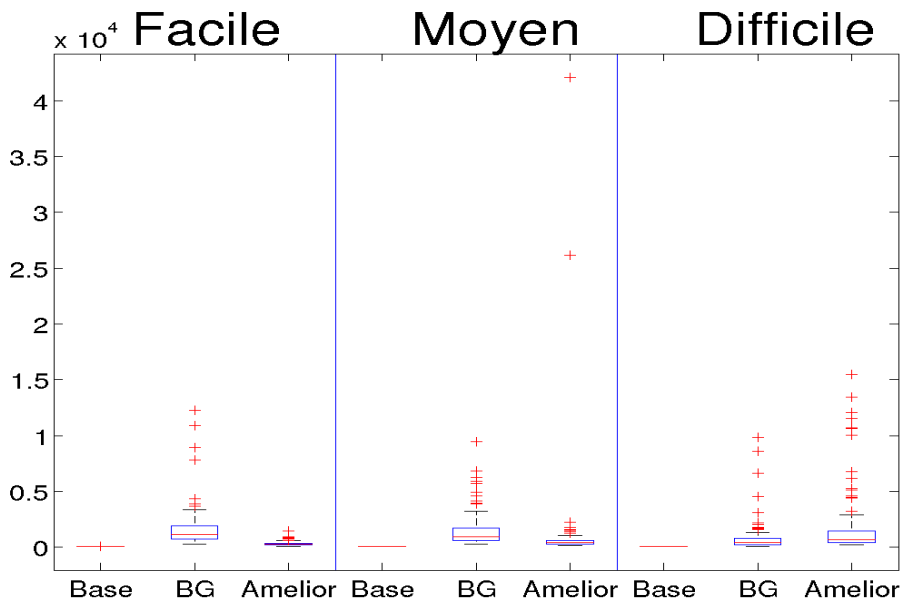
La figure 3.6 représente les boxplots des temps *elapsed* respectifs de chaque algorithme ; en particulier, l'échantillonneur utilisant l'amélioration a parfois été beaucoup plus long à converger. Un agrandissement sur les valeurs plus faibles (figure 3.7) montre qu'en règle générale l'échantillonneur avec l'amélioration converge plus vite que celui utilisant la statistique de Brooks-Gelman de base.

Enfin, comme nous avons imposé à tous les algorithmes de faire au moins 5 000 itérations, les échantillonneurs utilisant un critère d'arrêt mettent plus de temps à s'arrêter que l'algorithme utilisant seulement 5 000 itérations.

Sur la figure 3.8 sont représentées les boxplots des erreurs de classification pour chacun des échantillonneurs. En règle générale, les trois méthodes renvoient la même qualité avec une légère amélioration pour le critère d'arrêt amélioré.

Les statistiques de Brooks-Gelman et leurs améliorations permettent d'obtenir un critère pour arrêter l'échantillonneur de *Gibbs* sans dégrader la qualité des estimations en comparaison des résultats du cas avec un nombre fixé à 5 000 itérations. En revanche, ils sont plus coûteux en temps et n'ont pas permis d'améliorer significativement l'estimation des classements.

(100, 200)




(200, 100)

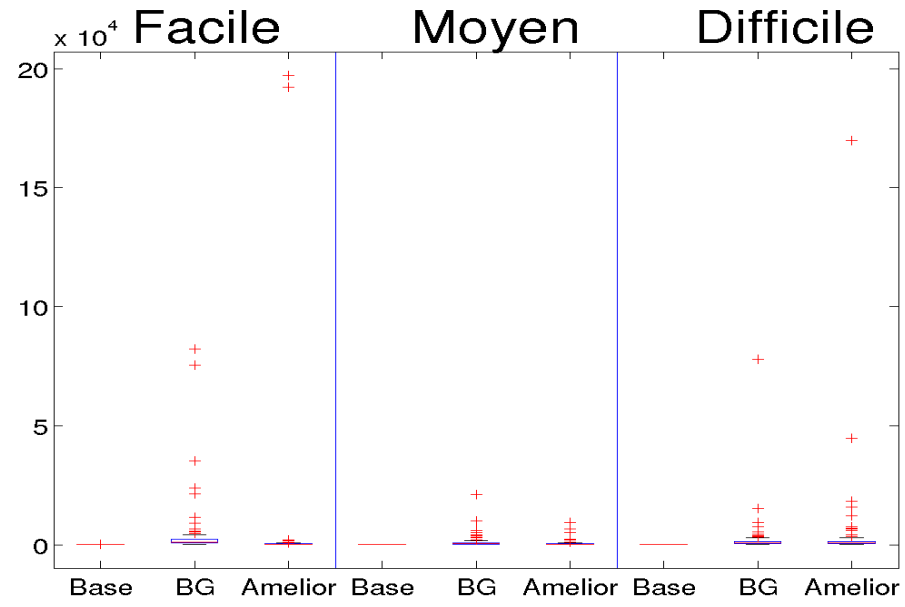



FIGURE 3.6 – Boxplot des temps de calcul (en secondes) pour l'échantillonneur de *Gibbs* en arrêtant après 5 000 itérations (base), avec la statistique de Brooks-Gelman (BG) et avec la statistique améliorée (Amelior) pour différentes difficultés (facile, moyen, difficile) et différentes tailles de matrices : $(100, 200)$ en haut et $(200, 100)$ en bas.

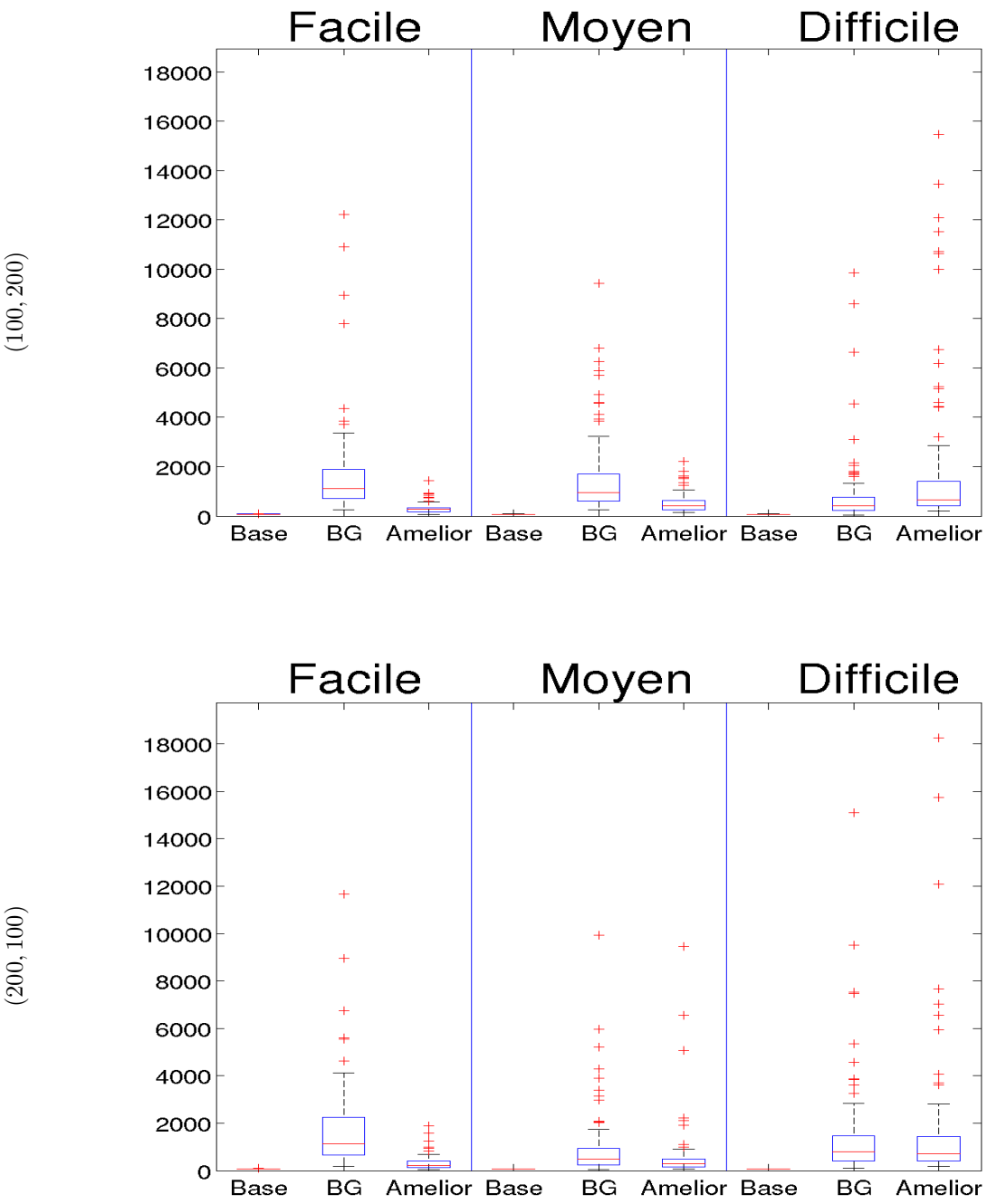



FIGURE 3.7 – Agrandissement sur les valeurs plus petites des boxplot des temps de calcul (en secondes) pour l'échantillonneur de *Gibbs* en arrêtant après 5 000 itérations (base), avec la statistique de Brooks-Gelman (BG) et avec la statistique améliorée (Amelior) pour différentes difficultés (facile, moyen, difficile) et différentes tailles de matrices : $(100, 200)$ en haut et $(200, 100)$ en bas.

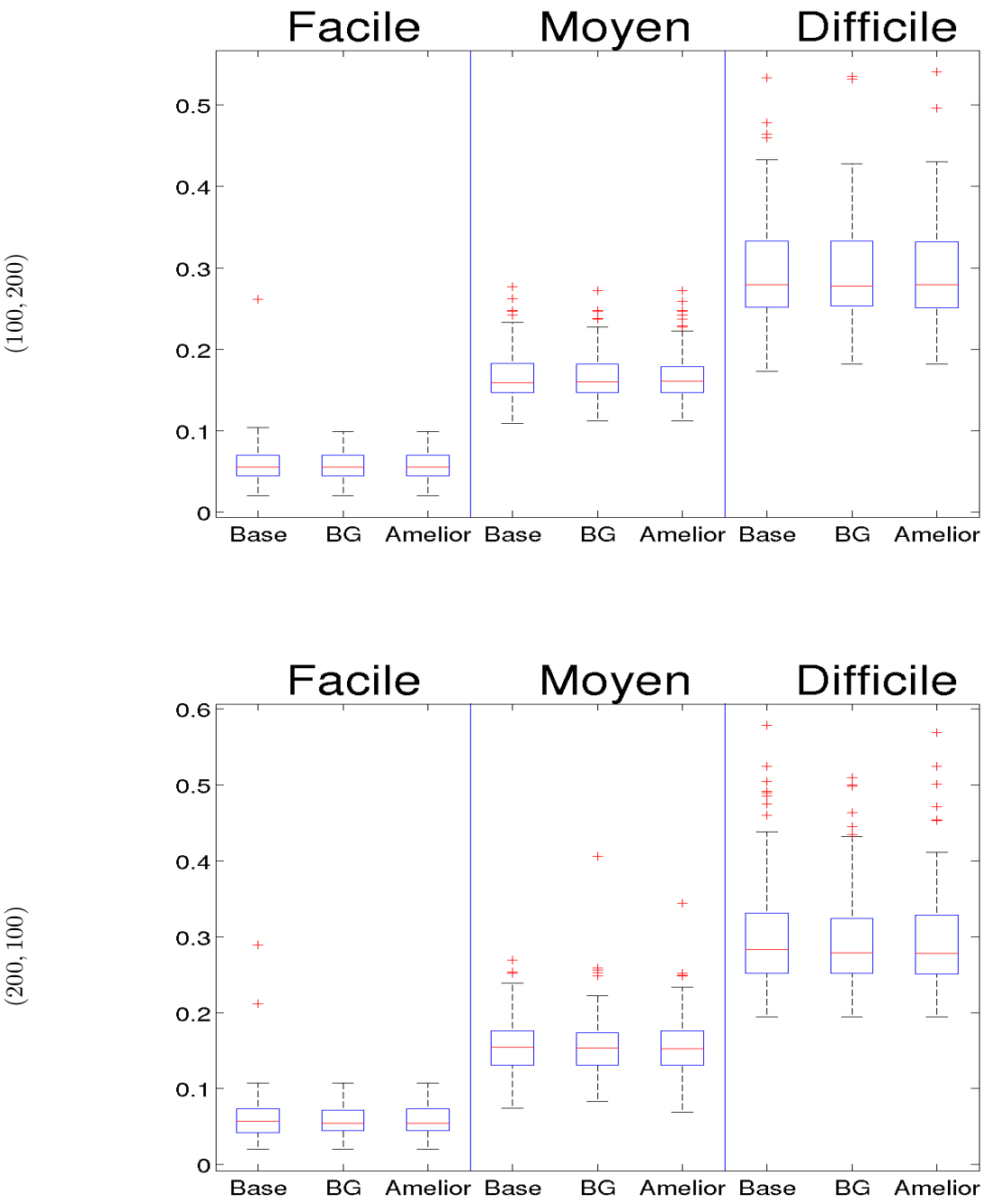



FIGURE 3.8 – Boxplot des erreurs de partitions $e_{(n,g)\times(d,m)}$ pour l'échantillonneur de *Gibbs* en arrêtant après 5 000 itérations (base), avec la statistique de Brooks-Gelman (BG) et avec la statistique améliorée (Amelior) pour différentes difficultés (facile, moyen, difficile) et différentes tailles de matrices : $(100, 200)$ en haut et $(200, 100)$ en bas.

## 3.7 Conclusion

L'approche bayésienne abordée dans ce chapitre a permis la création de deux algorithmes et d'une procédure répondant en partie aux problèmes de dégénérescence de l'algorithme *SEM+VEM*.

Tirant partie de la régularisation bayésienne, et pour des paramètres bien choisis, l'algorithme *V-Bayes* apporte une réponse au problème de dégénérescence des classes rencontré par l'algorithme *VEM*.

L'échantillonneur de *Gibbs*, pour sa part, ne souffre pas du problème d'états absorbants, en théorie et en pratique, rencontré par l'algorithme *SEM-Gibbs*. En revanche, la définition de son critère d'arrêt reste délicate malgré les éléments de réponse apportés avec la proposition de notre statistique de Brooks-Gelman améliorée. Il serait intéressant de trouver un critère d'arrêt basé sur la recherche du mode permettant peut-être d'accélérer la convergence et la qualité des résultats.

# Chapitre 4

# Sélection de modèle

*"Tous les modèles sont faux, mais certains sont utiles"*
George Box

## Sommaire


**4.1 Introduction . . . . . 100**
**4.2 Sélection de modèle . . . . . 101**
4.2.1 Critère *Integrated Completed Likelihood* . . . . . 101
4.2.2 Critère *Bayesian Information Criterion* . . . . . 104
4.2.3 Consistance . . . . . 106
4.2.4 Expérimentations numériques . . . . . 107
**4.3 Quantifier une perte d'information . . . . . 113**
4.3.1 Théorie . . . . . 113
4.3.2 Expérimentations numériques . . . . . 114
**4.4 Conclusion . . . . . 116**
**4.A Critère $ICL_{(a,b)}$ . . . . . 117**
4.A.1 Formule explicite . . . . . 117
4.A.2 Comportement théorique . . . . . 118
**4.B Démonstration de la partie "quantifier une perte d'informations" . . . . . 119**
4.B.1 Densité binaire contre densité ternaire . . . . . 119
4.B.2 Densité contrainte . . . . . 120


## 4.1 Introduction

Jusqu'à présent, les méthodes utilisées présupposent la connaissance du nombre de classes en ligne $g$ et en colonne $m$. En règle générale, cette information est inconnue et nous nous intéressons dans ce chapitre à l'évaluation de $g$ et $m$.

Après un rappel sur la sélection de modèle, nous nous intéressons à l'adaptation du critère *ICL* (pour *Integrated Completed Likelihood*), initialement développé dans le cas de la classification des modèles de mélanges simples (Biernacki et al., 2000) ; pour ce critère d'inspiration bayésienne, nous discutons du choix des lois a priori et calculons une formule explicite pour le cas du modèle des blocs latents. Puis, nous montrons que le calcul direct du critère *BIC* (pour *Bayesian Information Criterion*) n'est pas possible pour le modèle des blocs latents et conjecturons une forme basée sur l'approximation asymptotique

du critère *ICL*.

Ensuite, nous conjecturons que les critères *ICL* et *BIC* ont les mêmes comportements asymptotiques et que si l'estimateur du couple $(g^\star, m^\star)$ renvoyé par le critère *BIC* est consistant alors celui du critère *ICL* l'est aussi. Nous les étudions alors sur des données simulées et réelles.

Enfin, nous adaptons ces critères pour quantifier la perte d'information lorsque des données ternaires sont binarisées.

## 4.2 Sélection de modèle

Suivant les objectifs, le but de la sélection de modèles est de choisir parmi une collection :
— un modèle permettant de faire les meilleures prédictions,
— ou un modèle le plus proche possible de celui ayant servi à la génération des données,
— ou un modèle permettant de faire de la classification.
Dans le cadre des modèles de mélanges, nous supposons que tous les modèles sont de la forme :

$$p(x_i; \boldsymbol{\theta}) = \sum_{k=1}^{g} \pi_k \varphi(x_i; \alpha_k)$$

où le paramètre $g$ caractérise le modèle $\mathfrak{M}_g$. La sélection se fait alors sur le nombre de classes (en plus de la complexité des paramètres $\alpha_k$).

Pour répondre au problème de sélection de modèle, il existe de nombreux critères pour le cas des mélanges simples dont un certain nombre sont étendus à la classification croisée (voir la section 1.2.6.3.2). Dans ce chapitre, nous nous intéressons à l'adaptation des critères *ICL* et *BIC*.

### 4.2.1 Critère *Integrated Completed Likelihood*

Dans le cas des modèles de mélanges simples, le critère *ICL* (pour *Integrated Completed Likelihood*) proposé par Biernacki et al. (2000) se situe dans un contexte de classification ; le principe est de se placer dans un cadre bayésien et de trouver le modèle $\mathfrak{M}_g$ maximisant la probabilité des données complètes $(\mathbf{x}, \mathbf{z}^\star)$ connaissant ce modèle :

$$\widehat{g} \in \underset{g}{\operatorname{argmax}}\, p\left(\mathbf{x}, \mathbf{z}^\star \middle| \mathfrak{M}_g\right).$$

En pratique, la partition $\mathbf{z}^\star$ est inconnue et le calcul de $p\left(\mathbf{x}, \mathbf{z}^\star \middle| \mathfrak{M}_g\right)$ est parfois compliqué : l'idée proposée par les auteurs est de calculer pour chaque modèle $\mathfrak{M}_g$ l'estimateur du maximum de vraisemblance $\widehat{\theta}_{\mathfrak{M}_g}$

$$\widehat{\theta}_{\mathfrak{M}_g} \in \underset{\theta \in \Theta}{\operatorname{argmax}}\, p\left(\mathbf{x} | \boldsymbol{\theta}; g\right) \tag{4.1}$$

puis de prendre la matrice de partition $\widehat{\mathbf{z}}_{\mathfrak{M}_g}$ maximisant la probabilité connaissant les observations et $\widehat{\theta}_{\mathfrak{M}_g}$ :

$$\widehat{\mathbf{z}}_{\mathfrak{M}_g} \in \underset{\mathbf{z} \in \mathcal{Z}}{\operatorname{argmax}}\, p\left(\mathbf{z} \middle| \mathbf{x}, \widehat{\theta}_{\mathfrak{M}_g}\right). \tag{4.2}$$

et enfin de sélectionner le modèle maximisant la log-vraisemblance complète intégrée :

$$ICL(g) = p(\mathbf{x}, \widehat{\mathbf{z}}_{\mathfrak{M}_g})$$

qui peut être explicite pour certains choix de lois a priori.

L'estimateur du paramètre $g$ est alors celui maximisant le critère $ICL(g)$.

Pour le cadre du modèle des blocs latents, nous reprenons le choix des lois a priori du chapitre 3 ; le critère dépend donc des paramètres $a$ et $b$ :

$$ICL_{(a,b)}(g,m) = \log p\left(\mathbf{x}, \widehat{\mathbf{z}}, \widehat{\mathbf{w}}\right).$$

En utilisant l'indépendance conditionnelle de $\mathbf{z}$ et $\mathbf{w}$ conditionnellement à la connaissance de $\boldsymbol{\alpha}$, $\boldsymbol{\pi}$ et $\boldsymbol{\rho}$, la vraisemblance complète intégrée peut s'écrire :

$$\begin{aligned}
p(\mathbf{x},\mathbf{z},\mathbf{w}) &= \int p(\mathbf{x},\mathbf{z},\mathbf{w},\boldsymbol{\alpha},\boldsymbol{\pi},\boldsymbol{\rho})p(\boldsymbol{\alpha})p(\boldsymbol{\pi})p(\boldsymbol{\rho})d\boldsymbol{\alpha}d\boldsymbol{\pi}d\boldsymbol{\rho} \\
&= \int p(\mathbf{x}|\mathbf{z},\mathbf{w},\boldsymbol{\alpha},\boldsymbol{\pi},\boldsymbol{\rho})p(\mathbf{z},\mathbf{w}|\boldsymbol{\alpha},\boldsymbol{\pi},\boldsymbol{\rho})p(\boldsymbol{\alpha})p(\boldsymbol{\pi})p(\boldsymbol{\rho})d\boldsymbol{\alpha}d\boldsymbol{\pi}d\boldsymbol{\rho} \\
&= \int p(\mathbf{x}|\mathbf{z},\mathbf{w},\boldsymbol{\alpha})p(\boldsymbol{\alpha})d\boldsymbol{\alpha} \int p(\mathbf{z}|\boldsymbol{\pi})p(\boldsymbol{\pi})d\boldsymbol{\pi} \int p(\mathbf{w}|\boldsymbol{\rho})p(\boldsymbol{\rho})d\boldsymbol{\rho} \\
&= p(\mathbf{z})p(\mathbf{w})p(\mathbf{x}|\mathbf{z},\mathbf{w}).
\end{aligned}$$

Ce qui donne la décomposition suivante :

$$ICL_{(a,b)}(g,m) = \log p(\mathbf{z}) + \log p(\mathbf{w}) + \log p(\mathbf{x}|\mathbf{z},\mathbf{w}). \tag{4.3}$$

Il ne reste plus qu'à calculer chaque terme (en annexe 4.A.1) pour obtenir la formule explicite :

$$\begin{aligned}
ICL_{(a,b)}(g,m) &= \log p\left(\mathbf{x}, \widehat{\mathbf{z}}, \widehat{\mathbf{w}}\right) \\
&= \log\Gamma(ga) + \log\Gamma(ma) - (m+g)\log\Gamma(a) + mg(\log\Gamma(rb) - r\log\Gamma(b)) \\
&\quad - \log\Gamma(n+ga) - \log\Gamma(d+ma) \\
&\quad + \sum_{k=1}^{g} \log\Gamma(\widehat{z}_{+k}+a) + \sum_{\ell=1}^{m} \log\Gamma(\widehat{w}_{+\ell}+a) \\
&\quad + \sum_{k,\ell}\left[\left(\sum_{h=1}^{r} \log\Gamma\left(N^{h}_{k\ell;\widehat{\mathbf{z}},\widehat{\mathbf{w}}}+b\right)\right) - \log\Gamma(\widehat{z}_{+k}\widehat{w}_{+\ell}+rb)\right],
\end{aligned}$$

où le couple $(\widehat{\mathbf{z}}, \widehat{\mathbf{w}})$ dépend des nombres de classes $g$ et $m$.

**Obtention de $(\widehat{\mathbf{z}}, \widehat{\mathbf{w}})$**

Pour l'obtention de $(\widehat{\mathbf{z}}, \widehat{\mathbf{w}})$, il n'est possible de calculer ni l'estimateur du maximum de vraisemblance (comme pour l'équation (4.1)), ni le couple maximisant la probabilité connaissant les observations et $\boldsymbol{\theta}$ (comme pour l'équation (4.2)).
Nous proposons de prendre les estimateurs des partitions fournis par la combinaison de l'échantillonneur de *Gibbs* et de l'algorithme *V-Bayes* proposée dans le chapitre 3 :

$$\left(\widehat{\mathbf{z}}, \widehat{\mathbf{w}}, \widehat{\theta}^{GB}\right) \in \underset{\mathbf{z},\mathbf{w},\boldsymbol{\theta}}{\operatorname{argmax}}\, \mathcal{F}_B\left(q_{\mathbf{zw}};\boldsymbol{\theta}\right)$$

où $\mathcal{F}_B$ représente l'énergie libre définie en section 3.3.

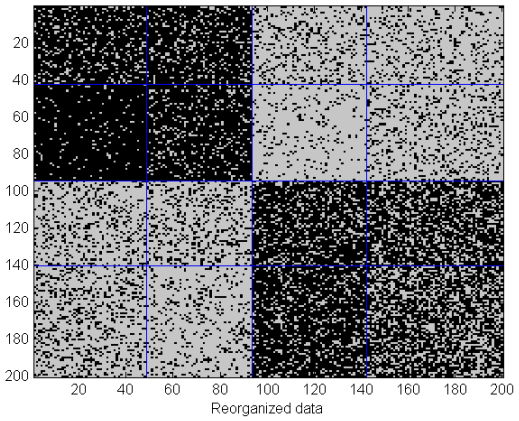



FIGURE 4.1 – Représentation de la matrice utilisée pour mesurer l'influence des paramètres de $a$ et $b$. Cette matrice est réorganisée par les partitions ayant servi à la simulation.

**Choix des paramètres** $(a, b)$

Pour tester empiriquement l'influence des paramètres $(a, b)$, nous avons simulé une matrice avec $g = m = 4$ classes en ligne et en colonne et $n = d = 200$ lignes et colonnes dont une représentation suivant la partition originelle est proposée sur la figure 4.1. Les quatres classes sont formées par deux groupes distincts eux mêmes séparés en deux groupes plus semblables.

Nous utilisons l'échantillonneur de *Gibbs* combiné avec l'algorithme *V-Bayes* pour des classes en ligne et en colonne allant de 2 à 8 puis nous regardons le couple des nombres de classes sélectionné par le critère $ICL_{(a,b)}$ suivant les valeurs des paramètres en faisant varier ces derniers de 4 à 200 par pas de 4 (nous ajoutons aussi les cas où ils valent 1/2). Les résultats sont représentés en trois dimensions sur la figure 4.2.

Plus $b$ est grand, moins le critère sélectionne de classes. En revanche, l'influence de $a$ semble moins importante.

Cette expérience semble indiquer que nous pouvons choisir $1 \leq a \leq 200$ et $1 \leq b \leq 10$. Dans notre cas, et par continuité avec le chapitre 3, nous proposons de prendre $a = 4$ et $b = 1$.

**Comportement théorique**

Le critère $ICL_{(a,b)}$ recherche le couple $(g, m)$ maximisant la log-vraisemblance complète $\log p(\mathbf{x}, \widehat{\mathbf{z}}, \widehat{\mathbf{w}})$. Dans cette partie, nous étudions le type de configurations que le critère va préférer.

Comme nous le voyons en début de section sur l'équation (4.3), le critère $ICL_{(a,b)}$ s'écrit comme la somme de deux termes : la probabilité du couple $(\widehat{\mathbf{z}}, \widehat{\mathbf{w}})$ et celle de $\mathbf{x}$ connaissant le couple $(\widehat{\mathbf{z}}, \widehat{\mathbf{w}})$ :

$$ICL_{(a,b)}(g, m) = \underbrace{\log p(\widehat{\mathbf{z}}) + \log p(\widehat{\mathbf{w}})}_{A} + \underbrace{\log p(\mathbf{x} | \widehat{\mathbf{z}}, \widehat{\mathbf{w}})}_{B}. \tag{4.4}$$

D'une part, si nous fixons la disposition des blocs $\big($c'est-à-dire si nous fixons le couple $(\widehat{\mathbf{z}}, \widehat{\mathbf{w}})\big)$, le critère $ICL_{(a,b)}$ est minimum pour des blocs avec un équilibre entre les cases blanches et les cases noires (pour le cas binaire). À l'opposé, il est maximum pour des blocs ne contenant que des cases monochromes

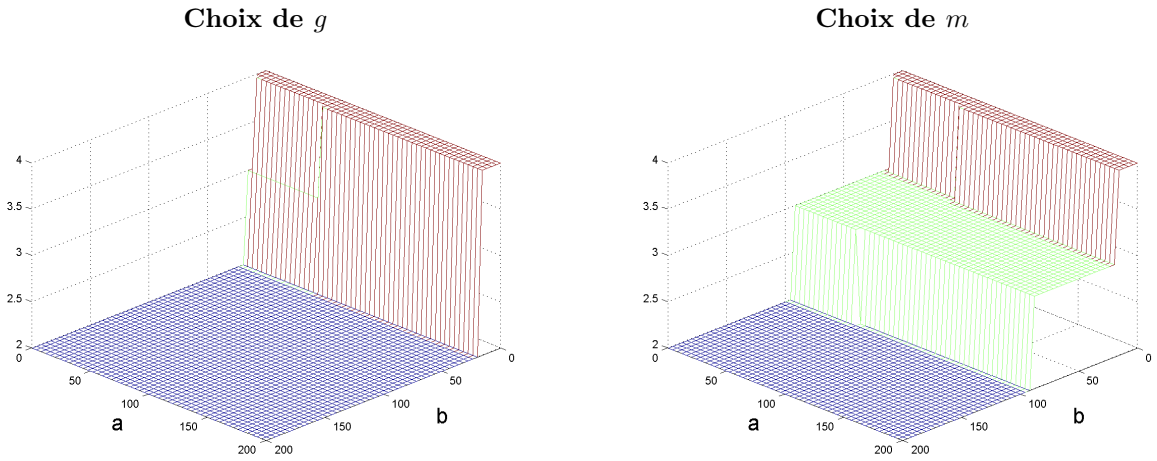



FIGURE 4.2 – Représentations en trois dimensions des nombres de classes en ligne (à gauche) et en colonne (à droite) sélectionnés par le critère $ICL_{(a,b)}$ suivant les valeurs de $a$ et de $b$.

(voir l'annexe 4.A.2). Pour faire augmenter le terme $B$, il est donc préférable d'avoir des blocs contenant des cases avec la même valeur, quitte à subdiviser certaines classes.

D'autre part, plus nous subdivisons les partitions en ligne et en colonne, plus le terme $A$ est petit.

Le critère renvoie donc un modèle faisant l'équilibre entre des blocs monochromes et des partitions des lignes et colonnes pas trop fines.

*Exemple* 4.2.1. La figure 4.3 représente trois classements possibles pour l'exemple de l'échiquier de la figure 1.15 et nous avons calculé les critères $ICL_{(4,1)}$ associés.

Pour une seule classe, la valeur du critère est maximale pour le terme $A$ mais très faible pour la partie $B$ à cause des mêmes effectifs des cases noires et blanches (figure de gauche).
En revanche, sur la figure du milieu, les blocs sont constitués chacun d'une seule valeur et les lignes et les colonnes ne sont divisées qu'en deux groupes de mêmes effectifs ; le critère renvoie une valeur assez forte pour cette configuration.
Lorsque nous subdivisons des blocs ne contenant qu'une seule couleur (entre la figure du milieu et la figure de droite), nous diminuons les deux termes $A$ et $B$.

### 4.2.2 Critère *Bayesian Information Criterion*

Le critère *BIC* (pour *Bayesian Information Criterion*) proposé par Schwarz et al. (1978) se place dans un cadre d'estimation et cherche le modèle le plus probable connaissant les observations c'est-à-dire maximisant :

$$\mathbb{P}\left(\mathfrak{M}_g|\mathbf{x}\right) \propto \mathbb{P}\left(\mathbf{x}|\mathfrak{M}_g\right)\mathbb{P}\left(\mathfrak{M}_g\right).$$

En supposant que tous les modèles sont équiprobables, cela revient à choisir :

$$\widehat{g} \in \underset{g}{\operatorname{argmax}}\, \mathbb{P}\left(\mathbf{x}|\mathfrak{M}_g\right).$$

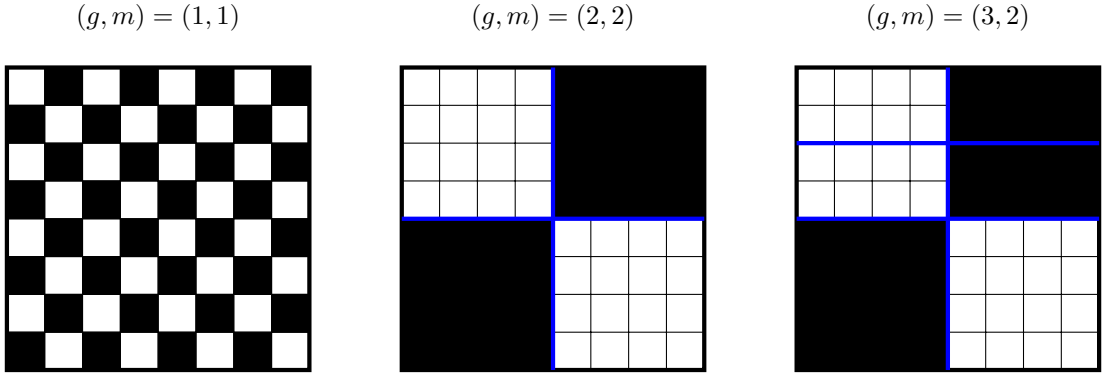



FIGURE 4.3 – Trois classifications possibles d'un échiquier : à gauche avec une seule classe en ligne et en colonne, au milieu avec deux classes en ligne et en colonne et à droite avec trois classes en ligne et deux en colonne.

Ce calcul n'étant pas possible, certains auteurs (voir par exemple Raftery, 1995) proposent une approximation asymptotique. Pour ce faire, ils définissent une loi a priori sur les paramètres $\boldsymbol{\theta}_g$ du modèle $\mathfrak{M}_g$ (voir par exemple Lebarbier et Mary-Huard, 2004) :

$$\begin{aligned}\mathbb{P}\left(\mathbf{x}|\mathfrak{M}_g\right) &= \int_{\Theta_g} \mathbb{P}\left(\mathbf{x}|\boldsymbol{\theta}_g,\mathfrak{M}_g\right)\mathbb{P}\left(\boldsymbol{\theta}_g|\mathfrak{M}_g\right)d\boldsymbol{\theta}_g \\ &= \int_{\Theta_g} e^{\log\left[\mathbb{P}\left(\mathbf{x}|\boldsymbol{\theta}_g,\mathfrak{M}_g\right)\mathbb{P}\left(\boldsymbol{\theta}_g|\mathfrak{M}_g\right)\right]}d\boldsymbol{\theta}_g \\ &= \int_{\Theta_g} e^{G(\boldsymbol{\theta}_g)}d\boldsymbol{\theta}_g\end{aligned}$$

où $G(\boldsymbol{\theta}_g)=\log\left[\mathbb{P}\left(\mathbf{x}|\boldsymbol{\theta}_g,\mathfrak{M}_g\right)\mathbb{P}\left(\boldsymbol{\theta}_g|\mathfrak{M}_g\right)\right]$.
Or, si une fonction $L$ de $\mathbb{R}^d$ vers $\mathbb{R}$ est deux fois différentiables et atteint un unique maximum en $u^\star$, l'*approximation de Laplace* donne alors

$$\int_{\mathbb{R}^d} e^{nL(u)}du = e^{nL(u^\star)}\left(\frac{2\pi}{n}\right)^{d/2}\left|-L''\left(u^\star\right)\right|^{-\frac{1}{2}}+\mathcal{O}\left(n^{-1}\right).$$

Sous certaines conditions, cette approximation reste vraie pour une fonction dépendant de $n$. En particulier, en prenant $L=\frac{1}{n}G$, ils obtiennent que :

$$\mathbb{P}\left(\mathbf{x}|\mathfrak{M}_g\right)=e^{G(\boldsymbol{\theta}_g^\star)}\left(\frac{2\pi}{n}\right)^{\nu_\theta/2}\left|H_{\boldsymbol{\theta}_g^\star}\right|^{-\frac{1}{2}}+\mathcal{O}\left(n^{-1}\right) \tag{4.5}$$

où $\nu_\theta$ représente le nombre de paramètres libres de $\boldsymbol{\theta}_g$ et $H_\theta$ l'opposée de la matrice hessienne des dérivées secondes partielles de $L$.

En passant au log et sous certaines conditions de régularité de l'EMV et de la matrice $H$, l'approximation suivante est obtenue :

$$\mathbb{P}\left(\mathbf{x}|\mathfrak{M}_g\right)\approx\max_{\boldsymbol{\theta}_g\in\Theta_g}\log p(\mathbf{x};\boldsymbol{\theta}_g,\mathfrak{M}_g)-\frac{\nu_\theta}{2}\log n \tag{4.6}$$

L'approximation asymptotique du critère *BIC* cherche alors le modèle maximisant l'équation (4.6). Keribin (2010) rappelle que l'approximation utilisée peut être mauvaise pour de petits jeux de données.

**Adaptation au modèle des blocs latents**

Dans le cas du modèle des blocs latents, les variables aléatoires étant dépendantes conditionnellement aux observations, les conditions pour établir l'approximation de l'équation (4.5) ne sont pas remplies.

Pour contourner cette difficulté, nous commençons par donner une approximation asymptotique du critère $ICL_{(a,b)}$. Pour ce faire, nous reprenons l'équation (4.4) et appliquons à chaque terme l'approximation de Laplace pour obtenir l'appoximation asymptotique, notée $ICL_{BIC}$, suivante :

$$ICL(g,m) \approx \max_{\boldsymbol{\theta}} \log p\left(\mathbf{x}, \widehat{\mathbf{z}}, \widehat{\mathbf{w}}; \boldsymbol{\theta}\right) - \frac{g-1}{2}\log n - \frac{m-1}{2}\log d - \frac{gm(r-1)}{2}\log(nd) = ICL_{BIC}(g,m). \quad (4.7)$$

Dans leur article, Biernacki et al. (2000) remplacent le paramètre maximisant la vraisemblance complète intégrée par l'estimateur du maximum de vraisemblance notée $\widehat{\theta}^{EMV}$ :

$$\begin{aligned} \max_{\boldsymbol{\theta}} \log p\left(\mathbf{x}, \widehat{\mathbf{z}}, \widehat{\mathbf{w}}; \boldsymbol{\theta}\right) &\approx \log p\left(\mathbf{x}, \widehat{\mathbf{z}}, \widehat{\mathbf{w}}; \widehat{\theta}^{EMV}\right) \\ &\approx \log p\left(\widehat{\mathbf{z}}, \widehat{\mathbf{w}} \,\middle|\, \mathbf{x}; \widehat{\theta}^{EMV}\right) + \log p\left(\mathbf{x}; \widehat{\theta}^{EMV}\right). \end{aligned}$$

Ainsi, le critère $ICL_{BIC}$ peut s'écrire :

$$ICL_{BIC}(g,m) = \log p\left(\widehat{\mathbf{z}}, \widehat{\mathbf{w}} \,\middle|\, \mathbf{x}; \widehat{\theta}^{EMV}\right) + BIC(g,m). \quad (4.8)$$

Par analogie et en tenant compte des équations (4.7) et (4.8), nous proposons un critère *BIC* de la forme suivante :

$$BIC(g,m) = \log p\left(\mathbf{x}; \widehat{\theta}^{EMV}\right) - \frac{gm(r-1)+g-1}{2}\log n - \frac{gm(r-1)+m-1}{2}\log d. \quad (4.9)$$

Nous obtenons une log-vraisemblance pénalisée par deux pénalités, l'une sur les lignes et l'autre sur les colonnes : les unités statistiques peuvent être alors vues respectivement comme le nombre de lignes et de colonnes. Les paramètres libres en ligne (resp. en colonne) sont alors $\boldsymbol{\pi}$ (resp. $\boldsymbol{\rho}$) et les paramètres $\boldsymbol{\alpha}$ des densités conditionnelles.

Comme la log-vraisemblance n'est pas calculable (voir section 2.2), nous proposons de la remplacer par l'énergie libre dans le calcul du critère *BIC*. Toutefois, comme leurs comportements semblent différents, surtout à distance finie, il est possible que le modèle sélectionné ne soit alors pas le même que si nous avions pu utiliser la log-vraisemblance.

### 4.2.3 Consistance

Le critère *BIC* est connu pour renvoyer des estimateurs consistants du nombre de classes (dans le sens où l'estimateur $\widehat{g}$ du nombre de classes est proche du vrai paramètre $g^\star$ lorsque le nombre d'observations est assez grand) dans le cas de mélanges simples où la log-vraisemblance est bornée (Keribin, 2000) contrairement au critère *ICL* qui ne l'est pas (Baudry, 2009) avec ce contraste.

En revanche, dans le cas du modèle des blocs latents, nous faisons la conjecture suivante :

**Conjecture 4.2.2.** ***Comportement asymptotique des critères***
*Pour tout couple $(a, b)$ et pour $n$ et $d$ assez grands, le critère BIC a le même comportement que le critère $ICL_{(a,b)}$.*

En effet, nous pouvons remarquer que pour deux couples $(a, b)$ et $(a', b')$ fixés, l'influence de la loi a priori disparaît lorsque le nombre d'observations est suffisamment grand. De plus, lorsque le nombre d'observation est suffisamment grand, le critère $ICL_{BIC}$ possède le même comportement que le critère $ICL_{(a,b)}$ pour $a$ et $b$ fixés ; il reste donc à montrer que c'est également le même comportement que celui du critère *BIC*.

Or, comme nous l'avons indiqué en section 1.3.2.4 et plus particulièrement par l'équation (1.3), Mariadassou et Matias (2012) ont démontré que, pour le cas du modèle des blocs latents, la distribution des labels $p(\mathbf{z}, \mathbf{w} | \mathbf{x}; \boldsymbol{\theta})$ conditionnellement aux observations se concentre autour de la partition réelle avec une forte probabilité dès que $\boldsymbol{\theta}$ est telle que $\boldsymbol{\alpha}$ n'est pas trop loin de la vraie valeur. Par conséquent, par la règle du maximum a posteriori et en prenant un estimateur $\widehat{\theta}$ tel que $\widehat{\boldsymbol{\alpha}}$ converge vers le vrai paramètre, la valeur $\log p\left(\widehat{\mathbf{z}}, \widehat{\mathbf{w}} \,\middle|\, \mathbf{x}; \widehat{\theta}\right)$ tend vers 0 pour les vrais paramètres $g^\star$ et $m^\star$.

En outre, ce terme négatif peut s'ajouter à la valeur du critère *BIC* comme une pénalité supplémentaire pour un nombre de classes $g$ et $m$ incorrects ; entraînant que le critère $ICL_{BIC}$ renverrait également des estimateurs consistants du nombre de classes.

Notons que cette conjecture a besoin de supposer la consistance des estimateurs du maximum de vraisemblance et du maximum variationnel pour le cas du modèle des blocs latents.

Cette conjecture n'est pas démontrée pour l'instant et reste un défi intéressant ; toutefois les simulations de la section 4.2.4 semblent la corroborer.

Enfin, si cette conjecture est vraie et si les estimateurs du couple $(g, m)$ du critère *BIC* sont consistants (comme pour le cas des modèles de mélange classiques), cela impliquerait que ceux des critères $ICL_{(a,b)}$ le sont également.

### 4.2.4 Expérimentations numériques

Dans cette section, nous regardons sur des données simulées et des données réelles le comportement des critères définis précédemment.

#### Données simulées

Pour les données simulées, nous reprenons les mêmes paramètres que précédemment (comme pour les sections 2.6.1 et 3.5), à savoir 5 classes en ligne et 4 en colonne avec des proportions équilibrées ou déséquilibrées et la matrice des paramètres $\boldsymbol{\alpha}$ suivante :

$$\boldsymbol{\alpha} = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & 1-\varepsilon \end{pmatrix}$$

en faisant varier $\varepsilon$ pour obtenir des matrices de difficultés différentes (+, ++ et +++) suivant le protocole de Lomet et al. (2012c).
Pour chaque cas, nous prenons des matrices de tailles $(100, 200)$, $(150, 150)$ et $(200, 100)$ ; pour les proportions équilibrées, nous utilisons également des matrices de tailles $(500, 1000)$, $(750, 750)$ et $(1000, 500)$.

Pour chaque matrice, nous utilisons l'échantillonneur de *Gibbs* combiné avec l'algorithme *V-Bayes* avec $a = 4$ et $b = 1$ pour des nombres de classes en ligne et en colonne allant de 2 à 8 et regardons à chaque fois les valeurs des critères.

Le nombre de fois où chaque couple $(g, m)$ a été sélectionné par chaque critère est résumé dans le tableau 4.1 pour le critère $ICL_{(4,1)}$ avec des proportions équilibrées, le tableau 4.2 pour le critère *BIC* avec des proportions équilibrées et le tableau 4.3 pour les deux critères avec des proportions inégales.

Les deux critères ont tendance à sélectionner moins de classes lorsque qu'il y a peu de lignes et de colonnes, en particulier pour les matrices difficiles (+++). Mais lorsqu'il y a suffisamment d'observations, les deux critères semblent avoir des comportements similaires et sélectionnent généralement les bons nombres de classes.

Dans le cas des mélanges simples, le critère *ICL* donne toujours un nombre de classes inférieur ou égal à celui du critère *BIC* ; toutefois, dans le cas du modèle des blocs latents, et lorsqu'il y a peu d'observations, le critère *BIC* sélectionne plus souvent un nombre inférieur de classes que le critère $ICL_{(4,1)}$. Cela peut s'expliquer par le fait que nous remplaçons la log-vraisemblance par l'énergie libre : cette dernière est une minoration certainement plus mauvaise quand les nombres de classes demandés sont petits. Le critère *BIC* utilisé avec l'énergie libre est doublement altéré pour les modèles avec un grand nombre de classes : une fois par l'utilisation de l'énergie libre et une deuxième fois par l'approximation asymptotique. Lorsque le nombre d'observations augmente, l'approximation variationnelle est meilleure et cet effet de double altération est donc atténué.

**Données réelles : votes au parlement américain**

Nous avons testé la procédure sur les données du parlement américain (*UCI Congressionnal Voting Records*[1]). Le jeu de données représente le vote de 435 parlementaires américains, membres de la 98 ème *Chambre des représentants* (*United States House of Representatives*), répartis en 168 républicains et 267 démocrates et s'exprimant sur 16 lois (éducation, sécurité, politique internationale...) suivant 3 niveaux de réponse ("pour", "contre" et "abstention").

Nous utilisons l'échantillonneur de *Gibbs* combiné avec l'algorithme *V-Bayes* pour des nombres de classes en ligne et en colonne allant de 2 à 14 et regardons pour chaque critère la partition sélectionnée. Le critère $ICL_{(4,1)}$ retient le couple $(g, m) = (5, 7)$ et le critère *BIC* celui de $(g, m) = (4, 6)$. À nouveau, le critère *BIC* sélectionne moins de classes que le critère $ICL_{(4,1)}$.

Sur la figure 4.4 sont représentées les matrices réorganisées suivant les partitions sélectionnées respectivement par les critères $ICL_{(4,1)}$ (à gauche) et *BIC* (à droite) en prenant "pour" en noir, "contre" en blanc et "abstention" en gris ; et sur le tableau 4.4 la répartition des démocrates et des républicains pour chacune des classes.

Les groupes sur les lignes se séparent en deux parties : la partie des votants composée d'un groupe majoritairement démocrate, un groupe majoritairement républicain et un groupe plutôt mixte et la partie des abstentionnistes où le critère $ICL_{(4,1)}$ sépare les trois grands abstentionnistes de la chambre (ils se sont abstenus, 14, 15 et 16 fois) des autres (qui se sont abstenus au maximum 12 fois) ; distinction qui n'est pas faite par le critère *BIC*.

Pour les colonnes, les groupes 1 et 2 sont des lois où les républicains se sont majoritairement exprimés "pour" à l'opposé des démocrates qui se sont exprimés "contre" et inversement pour les groupes 3 et 4 ;

1. Le jeu de données est disponible à l'adresse suivante :
`http://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records`.

**Critère $ICL_{(4,1)}$ et proportions équilibrées**

| | + | | | ++ | | | | +++ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $(100, 200)$ | $g \backslash m$ | 4 | 5 | 6 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 3 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 3 | 7 | 0 | 0 | 0 |
| | 4 | 0 | 0 | 0 | 4 | 0 | 12 | 0 | 0 | 4 | 3 | 33 | 0 | 0 |
| | 5 | 34 | 4 | 0 | 5 | 0 | 8 | 23 | 2 | 5 | 0 | 7 | 0 | 0 |
| | 6 | 11 | 0 | 0 | 6 | 0 | 0 | 4 | 0 | 6 | 0 | 0 | 0 | 0 |
| | 7 | 1 | 0 | 0 | 7 | 0 | 0 | 1 | 0 | 7 | 0 | 0 | 0 | 0 |
| $(150, 150)$ | $g \backslash m$ | 4 | 5 | 6 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 3 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 3 | 11 | 0 | 0 | 0 |
| | 4 | 0 | 0 | 0 | 4 | 1 | 18 | 0 | 0 | 4 | 0 | 32 | 1 | 0 |
| | 5 | 35 | 3 | 1 | 5 | 0 | 2 | 29 | 0 | 5 | 0 | 1 | 5 | 0 |
| | 6 | 11 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 1 | 0 |
| $(200, 100)$ | $g \backslash m$ | 4 | 5 | 6 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 3 | 0 | 0 | 0 | 3 | 1 | 1 | 0 | 0 | 3 | 5 | 2 | 0 | 0 |
| | 4 | 0 | 0 | 0 | 4 | 0 | 16 | 5 | 0 | 4 | 0 | 36 | 4 | 0 |
| | 5 | 41 | 4 | 1 | 5 | 0 | 1 | 25 | 1 | 5 | 0 | 0 | 2 | 1 |
| | 6 | 4 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 |
| $(500, 1000)$ | $g \backslash m$ | 4 | 5 | 6 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 4 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 4 | 0 | 10 | 0 | 0 |
| | 5 | 38 | 0 | 0 | 5 | 0 | 0 | 44 | 0 | 5 | 0 | 2 | 36 | 0 |
| | 6 | 11 | 0 | 0 | 6 | 0 | 0 | 5 | 0 | 6 | 0 | 0 | 1 | 0 |
| | 7 | 1 | 0 | 0 | 7 | 0 | 0 | 1 | 0 | 7 | 0 | 0 | 1 | 0 |
| $(750, 750)$ | $g \backslash m$ | 4 | 5 | 6 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 4 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 4 | 0 | 0 | 8 | 0 |
| | 5 | 45 | 0 | 0 | 5 | 0 | 0 | 44 | 3 | 5 | 0 | 0 | 40 | 0 |
| | 6 | 5 | 0 | 0 | 6 | 0 | 0 | 3 | 0 | 6 | 0 | 0 | 2 | 0 |
| $(1000, 500)$ | $g \backslash m$ | 4 | 5 | 6 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 4 | 0 | 0 | 0 | 4 | 0 | 1 | 0 | 0 | 4 | 0 | 5 | 0 | 0 |
| | 5 | 45 | 4 | 0 | 5 | 0 | 0 | 43 | 6 | 5 | 0 | 0 | 43 | 2 |
| | 6 | 1 | 0 | 0 | 6 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 |

TABLE 4.1 – Sur 50 matrices, répartition du nombre de classes obtenu par la procédure avec le critère $ICL_{(4,1)}$ et pour le cas des proportions équilibrées suivant la taille (lignes globales) et la difficulté (colonnes globales) des matrices. L'emplacement du nombre de classes ayant servi à la simulation est encadré.

**Critère *BIC* et proportions équilibrées**

| | + | | | ++ | | | | +++ | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | $g \backslash m$ | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 3 | 0 | 0 | 0 | 3 | 0 | 0 | 0 | 0 | 3 | 12 | 0 | 0 | 0 |
| $(100, 200)$ | 4 | 0 | 0 | 0 | 4 | 1 | 30 | 1 | 0 | 4 | 4 | 34 | 0 | 0 |
| | 5 | 2 | **34** | 5 | 5 | 0 | 12 | **6** | 0 | 5 | 0 | 0 | **0** | 0 |
| | 6 | 0 | 9 | 0 | 6 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 |
| | $g \backslash m$ | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 3 | 0 | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 3 | 22 | 2 | 0 | 0 |
| $(150, 150)$ | 4 | 1 | 0 | 0 | 4 | 2 | 42 | 0 | 0 | 4 | 0 | 25 | 0 | 0 |
| | 5 | 0 | **35** | 2 | 5 | 0 | 1 | **4** | 0 | 5 | 0 | 0 | **1** | 0 |
| | 6 | 0 | 12 | 0 | 6 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 |
| | $g \backslash m$ | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 3 | 0 | 0 | 0 | 3 | 3 | 1 | 0 | 0 | 3 | 13 | 5 | 0 | 0 |
| $(200, 100)$ | 4 | 4 | 0 | 0 | 4 | 0 | 36 | 2 | 0 | 4 | 0 | 32 | 0 | 0 |
| | 5 | 0 | **41** | 0 | 5 | 0 | 0 | **8** | 0 | 5 | 0 | 0 | **0** | 0 |
| | 6 | 0 | 5 | 0 | 6 | 0 | 0 | 0 | 0 | 6 | 0 | 0 | 0 | 0 |
| | $g \backslash m$ | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 4 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 4 | 0 | 1 | 0 | 0 |
| $(500, 1000)$ | 5 | 0 | **38** | 1 | 5 | 0 | 0 | **45** | 0 | 5 | 0 | 1 | **45** | 0 |
| | 6 | 0 | 11 | 0 | 6 | 0 | 0 | 4 | 0 | 6 | 0 | 0 | 2 | 0 |
| | 7 | 0 | 0 | 0 | 7 | 0 | 0 | 1 | 0 | 7 | 0 | 0 | 1 | 0 |
| | $g \backslash m$ | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 4 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 4 | 0 | 8 | 0 | 0 |
| $(750, 750)$ | 5 | 0 | **45** | 0 | 5 | 0 | 0 | **44** | 0 | 5 | 0 | 0 | **39** | 0 |
| | 6 | 0 | 5 | 0 | 6 | 0 | 0 | 6 | 0 | 6 | 0 | 0 | 3 | 0 |
| | $g \backslash m$ | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 | $g \backslash m$ | 2 | 3 | 4 | 5 |
| | 4 | 0 | 0 | 0 | 4 | 0 | 0 | 0 | 0 | 4 | 0 | 4 | 0 | 0 |
| $(1000, 500)$ | 5 | 0 | **45** | 2 | 5 | 0 | 0 | **43** | 1 | 5 | 0 | 0 | **43** | 1 |
| | 6 | 0 | 3 | 0 | 6 | 0 | 0 | 6 | 0 | 6 | 0 | 0 | 2 | 0 |

TABLE 4.2 – Sur 50 matrices, répartition du nombre de classes obtenu par la procédure avec le critère *BIC* et pour le cas des proportions équilibrées suivant la taille (lignes globales) et la difficulté (colonnes globales) des matrices. L'emplacement du nombre de classes ayant servi à la simulation est encadré.

**Critère $ICL_{(4,1)}$ et proportions inégales**

$(100, 200)$

| $g$ \ $m$ | 3 | 4 | 5 | 6 |
|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 |
| 4 | 18 | 8 | 0 | 0 |
| 5 | 2 | **17** | 2 | 0 |
| 6 | 0 | 2 | 0 | 0 |
| 7 | 0 | 1 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 3 | 1 | 0 | 0 | 0 |
| 4 | 1 | 46 | 1 | 0 |
| 5 | 0 | 1 | **0** | 0 |
| 6 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 |
|---|---|---|---|
| 3 | 14 | 2 | 0 |
| 4 | 4 | 30 | 0 |
| 5 | 0 | 0 | **0** |
| 6 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 |

$(150, 150)$

| $g$ \ $m$ | 3 | 4 | 5 | 6 |
|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 |
| 4 | 20 | 14 | 0 | 0 |
| 5 | 1 | **6** | 2 | 0 |
| 6 | 0 | 7 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 3 | 0 | 1 | 0 | 0 |
| 4 | 1 | 41 | 5 | 0 |
| 5 | 0 | 0 | **2** | 0 |
| 6 | 0 | 0 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 |
|---|---|---|---|
| 3 | 4 | 10 | 0 |
| 4 | 0 | 33 | 2 |
| 5 | 0 | 1 | **0** |
| 6 | 0 | 0 | 0 |

$(200, 100)$

| $g$ \ $m$ | 3 | 4 | 5 | 6 |
|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 |
| 4 | 15 | 13 | 5 | 0 |
| 5 | 2 | **9** | 0 | 3 |
| 6 | 0 | 3 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 3 | 0 | 3 | 0 | 0 |
| 4 | 0 | 32 | 13 | 1 |
| 5 | 0 | 0 | **2** | 0 |
| 6 | 0 | 0 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 |
|---|---|---|---|
| 3 | 5 | 17 | 0 |
| 4 | 0 | 27 | 1 |
| 5 | 0 | 0 | **0** |
| 6 | 0 | 0 | 0 |

**Critère $BIC$ et proportions inégales**

$(100, 200)$

| $g$ \ $m$ | 3 | 4 | 5 | 6 |
|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 |
| 4 | 29 | 10 | 0 | 0 |
| 5 | 0 | **10** | 0 | 0 |
| 6 | 0 | 1 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 3 | 0 | 1 | 0 | 0 |
| 4 | 1 | 47 | 0 | 0 |
| 5 | 0 | 1 | **0** | 0 |
| 6 | 0 | 0 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 |
|---|---|---|---|
| 3 | 17 | 6 | 0 |
| 4 | 3 | 24 | 0 |
| 5 | 0 | 0 | **0** |
| 6 | 0 | 0 | 0 |

$(150, 150)$

| $g$ \ $m$ | 3 | 4 | 5 | 6 |
|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 |
| 4 | 29 | 13 | 0 | 0 |
| 5 | 2 | **2** | 1 | 0 |
| 6 | 0 | 3 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 3 | 0 | 4 | 0 | 0 |
| 4 | 0 | 43 | 2 | 0 |
| 5 | 0 | 0 | **1** | 0 |
| 6 | 0 | 0 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 |
|---|---|---|---|
| 3 | 10 | 17 | 0 |
| 4 | 0 | 23 | 0 |
| 5 | 0 | 0 | **0** |
| 6 | 0 | 0 | 0 |

$(200, 100)$

| $g$ \ $m$ | 3 | 4 | 5 | 6 |
|---|---|---|---|---|
| 3 | 0 | 0 | 0 | 0 |
| 4 | 30 | 8 | 1 | 0 |
| 5 | 2 | **7** | 0 | 0 |
| 6 | 0 | 1 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 3 | 0 | 6 | 0 | 0 |
| 4 | 0 | 40 | 3 | 0 |
| 5 | 0 | 2 | **0** | 0 |
| 6 | 0 | 0 | 0 | 0 |

| $g$ \ $m$ | 2 | 3 | 4 |
|---|---|---|---|
| 3 | 10 | 18 | 0 |
| 4 | 0 | 22 | 0 |
| 5 | 0 | 0 | **0** |
| 6 | 0 | 0 | 0 |

TABLE 4.3 – Sur 50 matrices, répartition du nombre de classes obtenu par la procédure avec les critères $ICL_{(4,1)}$ (en haut) et $BIC$ (en bas) et pour le cas des proportions inégales suivant la taille (lignes globales) et la difficulté (colonnes globales) des matrices. L'emplacement du nombre de classes ayant servi à la simulation est encadré.

le groupe "mixte" des lignes donnant les subdivisions plus fines. De plus, les deux critères séparent la dernière loi comprenant le plus grand nombre d'abstentions. Enfin, la différence du nombre de classes se fait sur les lois du groupe 5 du critère *BIC* où le critère $ICL_{(4,1)}$ préfère séparer celle où plus de parlementaires se sont abstenus ; ce choix est cohérent avec la distinction en ligne des abstentionnistes.

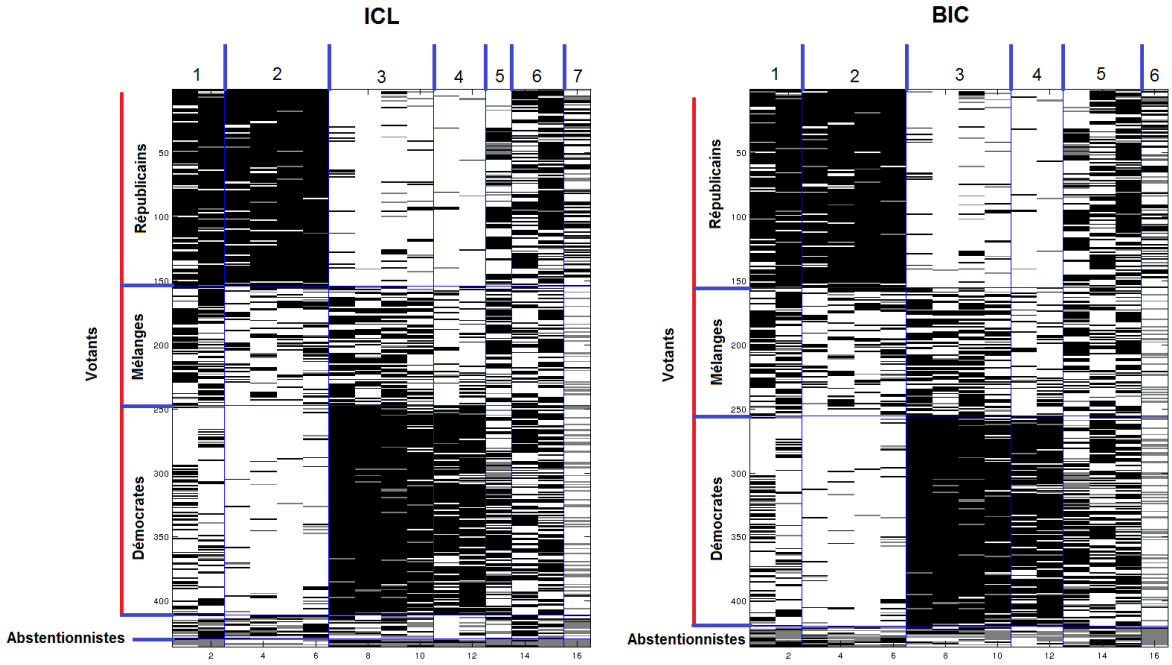



FIGURE 4.4 – Réorganisation de la matrice suivant les partitions sélectionnées respectivement par les critères $ICL_{(4,1)}$ (à gauche) et *BIC* (à droite). Les votes "pour" sont en noirs, les "contre" en blanc et les abstentions en gris.

| ICL | Rép | Dém | Total |
|---|---|---|---|
| 1 | 132 | 22 | 154 |
| 2 | 25 | 68 | 93 |
| 3 | 1 | 162 | 163 |
| 4 | 2 | 4 | 6 |
| 5 | 8 | 11 | 19 |

| BIC | Rép | Dém | Total |
|---|---|---|---|
| 1 | 131 | 24 | 155 |
| 2 | 27 | 73 | 100 |
| 3 | 1 | 163 | 164 |
| 4 | 9 | 7 | 15 |

TABLE 4.4 – Répartitions des démocrates et des répulicains sélectionnées respectivement par les critères $ICL_{(4,1)}$ (à gauche) et *BIC* (à droite).

Nous avons comparé les résultats obtenus avec ceux proposés par Wyse et Friel (2010) sur une version binarisée où les abstentions et les votes "contre" sont agrégés. Supposant que les nombres de classes $g$ et $m$ sont des variables aléatoires, les auteurs proposent un *échantillonneur intégré*[2] intégrant les paramètres

2. Traduction libre de *collapsed sampler*

en prenant une loi uniforme sur ces derniers (c'est-à-dire $(a, b) = (1, 1)$ dans la figure 3.1) et cherchant à estimer la loi jointe des modèles et des partitions $(\mathbf{z}, \mathbf{w})$ associées. Par la relation suivante :

$$\begin{aligned}
\log p(\mathbf{z}, \mathbf{w}, g, m|\mathbf{x}) &= \log p(\mathbf{x}|\mathbf{z}, \mathbf{w}, g, m) + \log p(\mathbf{z}, \mathbf{w}, g, m) - \log p(\mathbf{x}) \\
&= \log p(\mathbf{x}|\mathbf{z}, \mathbf{w}, g, m) + \log p(\mathbf{z}, \mathbf{w}|g, m) + \log p(g, m) - \log p(\mathbf{x}) \\
&= \log p(\mathbf{x}, \mathbf{z}, \mathbf{w}|g, m) + \log p(g, m) - \log p(\mathbf{x}) \\
&= ICL_{(1,1)}(\mathbf{z}, \mathbf{w}, g, m) + \log p(g, m) - \log p(\mathbf{x})
\end{aligned}$$

nous voyons que si nous prenons des lois a priori uniformes, maximiser cette loi revient à maximiser le critère $ICL_{(1,1)}$. Dans leur article, Wyse et Friel (2010) utilisent une loi de Poisson tronquée de paramètre 1, favorisant ainsi les petites valeurs pour les nombres de classes.

Nous avons utilisé leur algorithme [3] dans les mêmes conditions que leur article, à savoir 110 000 itérations dont 10 000 utilisées pour le temps de chauffe. L'algorithme a renvoyé le couple $(g, m) = (6, 11)$ (couple n'appartenant pas à la région de confiance à 60% de leur article) pour un critère $ICL_{(1,1)}$ correspondant de $-3565$. Nous avons reproduit l'expérience 30 fois et obtenu systématiquement 6 classes en ligne pour des classes en colonne allant de 10 à 14 suivant les expériences ; le meilleur critère $ICL_{(1,1)}$ étant de $-3554$ pour le couple $(g, m) = (6, 13)$. Nous avons utilisé notre procédure sur les données binarisées en prenant les mêmes paramètres et avons obtenu le couple $(g, m) = (5, 13)$, pour un critère $ICL_{(1,1)}$ de $-3553$ ; notons toutefois que le critère $ICL_{(1,1)}$ est très "plat" aux alentours de cette valeur.

En conclusion, nous obtenons des résultats très proches même si l'échantillonneur de Wyse et Friel (2010) n'a jamais sélectionné 5 classes en ligne (ce qui peut paraître paradoxal dans la mesure où leurs lois a priori sur les nombres de classes favorisent les faibles valeurs) et pour une procédure du même ordre de grandeur en terme d'itérations.

## 4.3 Quantifier une perte d'information

Dans certains cas, comme pour l'article de Wyse et Friel (2010), des auteurs sont obligés de regrouper des niveaux de réponse pour appliquer leurs algorithmes ; le plus souvent pour les binariser. L'inconvénient est alors de savoir comment comparer les résultats avec ceux obtenus sur les matrices non transformées.

Dans cette partie, nous proposons des adaptations des critères précédents permettant de quantifier la perte d'information du passage de données ternaires à des données binarisées (où deux niveaux de réponse ne sont alors plus distingués).

### 4.3.1 Théorie

Pour comparer des résultats entre des données ternaires et les mêmes binarisées, la première intuition est de calculer les critères associés à chaque résultat (par exemple $ICL_{(4,1)}$) pour le type de données correspondant (binaire ou ternaire) puis de prendre le modèle avec la plus grande valeur. Or, cette démarche sélectionne systématiquement le modèle binarisé (voir l'annexe 4.B.1). Il est donc nécessaire d'adapter les critères précédents pour pénaliser la perte d'information.

Supposons que nous avons des données ternaires, notées $\mathbf{x}^T$, appartenant à l'ensemble $\{0, 1, 2\}$ et que la matrice binarisée, notée $\mathbf{x}^B$, regroupe les valeurs 1 et 2 sous une même valeur notée $1^B$. Une fois ce regroupement effectué, il devient donc impossible de savoir si une case $x_{ij}^B$ valant $1^B$ signifie que la case $x_{ij}^T$ vaut 1 ou 2.

3. Les programmes sont disponibles sur la page `http://www.ucd.ie/statdept/jwyse`.

Nous proposons donc de voir les données binaires comme des données ternaires avec la contrainte que $\alpha^1_{k\ell} = \alpha^2_{k\ell}$ pour tout couple $(k, \ell)$. Or, une case $x^B_{ij}$ vaut 0 dans la matrice binarisée si et seulement si la case associée $x^T_{ij}$ de la matrice ternaire vaut également 0. En notant $\alpha^B$ le paramètre de la loi des données binaires de densité $\varphi_B$, ceci implique que :

$$\begin{aligned}\alpha^0_{k\ell} &= \mathbb{P}\left(x^T_{ij} = 0\right) \\ &= \mathbb{P}\left(x^B_{ij} = 0\right) \\ &= \left(1 - \alpha^B_{k\ell}\right).\end{aligned}$$

Or, nous savons que :

$$\begin{aligned}\alpha^0_{k\ell} + \alpha^1_{k\ell} + \alpha^2_{k\ell} = 1 &\Leftrightarrow \left(1 - \alpha^B_{k\ell}\right) + 2\alpha^1_{k\ell} = \left(1 - \alpha^B_{k\ell}\right) + \alpha^B_{k\ell} \\ &\Leftrightarrow 2\alpha^1_{k\ell} = \alpha^B_{k\ell} \\ &\Leftrightarrow \alpha^1_{k\ell} = \frac{1}{2}\alpha^B_{k\ell}.\end{aligned}$$

En notant $\varphi_{TC}$ la *densité ternaire contrainte*, il ne reste plus qu'à exprimer la relation existant entre les deux densités :

$$\begin{aligned}\varphi_{TC}(x^B_{ij}; \alpha^B) &= \left(\alpha^0_{k\ell}\right)^{\mathbb{1}\left\{x^T_{ij}=0\right\}} \left(\alpha^1_{k\ell}\right)^{\mathbb{1}\left\{x^T_{ij}=1\right\}} \left(\alpha^2_{k\ell}\right)^{\mathbb{1}\left\{x^T_{ij}=2\right\}} \\ &= \left(1 - \alpha^B_{k\ell}\right)^{\mathbb{1}\left\{x^T_{ij}=0\right\}} \left(\frac{1}{2}\alpha^B_{k\ell}\right)^{\mathbb{1}\left\{x^T_{ij}=1\right\}} \left(\frac{1}{2}\alpha^B_{k\ell}\right)^{\mathbb{1}\left\{x^T_{ij}=2\right\}} \\ &= \left(\frac{1}{2}\right)^{\mathbb{1}\left\{x^T_{ij}=1\right\}+\mathbb{1}\left\{x^T_{ij}=2\right\}} \left(1 - \alpha^B_{k\ell}\right)^{\mathbb{1}\left\{x^T_{ij}=0\right\}} \left(\alpha^B_{k\ell}\right)^{\mathbb{1}\left\{x^T_{ij}=1\right\}+\mathbb{1}\left\{x^T_{ij}=2\right\}} \\ &= \left(\frac{1}{2}\right)^{\mathbb{1}\left\{x^B_{ij}=1^B\right\}} \left(1 - \alpha^B_{k\ell}\right)^{\mathbb{1}\left\{x^B_{ij}=0\right\}} \left(\alpha^B_{k\ell}\right)^{\mathbb{1}\left\{x^B_{ij}=1^B\right\}} \\ &= \frac{1}{2^{x^B_{ij}}}\varphi_B(x^B_{ij}; \alpha^B).\end{aligned}$$

Conclusion, nous avons la relation suivante :

$$\varphi_{TC}(x^B_{ij}; \alpha^B) = \frac{1}{2^{x^B_{ij}}}\varphi_B(x^B_{ij}; \alpha^B). \tag{4.10}$$

À partir de l'équation (4.10), nous obtenons les critères pénalisés suivants :

$$\begin{aligned}BIC^{TC}(g, m) &= BIC^B(g, m) - N^B_1 \log 2 \\ ICL^{TC}_{(a,b)}(g, m) &= ICL^B_{(a,b)}(g, m) - N^B_1 \log 2\end{aligned}$$

où $N^B_1$ représente le nombre de $1^B$ dans la matrice $\mathbf{x}^B$ (les démonstrations reposent sur le lien entre les deux log-vraisemblances et sont mises en annexe 4.B.2).

Plus il y a de données binarisées et plus la pénalité est forte mais plus les blocs sont potentiellement homogènes dans la matrice binaire.

### 4.3.2 Expérimentations numériques

Dans cette partie, nous analysons empiriquement le comportement de ces critères pénalisés. Pour ce faire, nous regardons d'abord des données simulées puis des données réelles.

### Données simulées

Pour cette partie, nous simulons $n = 200$ lignes et $d = 100$ colonnes, des proportions équilibrées en ligne et en colonne et des paramètres de la forme suivante :

$$\alpha^0 = \begin{pmatrix} 1-\varepsilon & 1-\varepsilon & 1-\varepsilon \\ 1-2\varepsilon & 1-\varepsilon & 1-\varepsilon \\ 1-2\varepsilon & 1-2\varepsilon & 1-\varepsilon \\ 1-2\varepsilon & 1-2\varepsilon & 1-2\varepsilon \end{pmatrix} \quad \alpha^1 = \begin{pmatrix} \varepsilon\eta & \varepsilon\eta & \varepsilon\eta \\ 2\varepsilon\eta & \varepsilon\eta & \varepsilon\eta \\ 2\varepsilon\eta & 2\varepsilon\eta & \varepsilon\eta \\ 2\varepsilon\eta & 2\varepsilon\eta & 2\varepsilon\eta \end{pmatrix} \quad \alpha^2 = \begin{pmatrix} \varepsilon(1-\eta) & \varepsilon(1-\eta) & \varepsilon(1-\eta) \\ 2\varepsilon(1-\eta) & \varepsilon(1-\eta) & \varepsilon(1-\eta) \\ 2\varepsilon(1-\eta) & 2\varepsilon(1-\eta) & \varepsilon(1-\eta) \\ 2\varepsilon(1-\eta) & 2\varepsilon(1-\eta) & 2\varepsilon(1-\eta) \end{pmatrix}$$

avec $\varepsilon \in \{0.001, 0.01, 0.1, 0.4, 0.45, 0.4999\}$ paramètre de proportions des valeurs $\{1, 2\}$ et $\eta \in \{0.001, 0.01, 0.1, 0.4, 0.45, 0.4999\}$ paramètre de proportion entre les valeurs 1 et 2. En particulier, si $\eta \approx 0.5$ alors $\alpha^1 \approx \alpha^2$ et si $\varepsilon \approx 0$, alors $\alpha^0_{k\ell} \approx 1$ pour tout $(k, \ell)$.

Pour chaque combinaison, nous simulons 50 matrices et utilisons pour chacune l'échantillonneur de *Gibbs* combiné avec l'algorithme *V-Bayes* et regardons les critères $ICL_{(4,1)}$ et *BIC* pour des classes en ligne et en colonne allant de 2 à 6. Dans un second temps, nous regroupons les valeurs 1 et 2 pour obtenir des données binarisées et recommençons l'expérience avec les critères ternaires contraints.

Le tableau 4.5 récapitule le nombre de fois où un modèle binaire a été sélectionné. Lorsque la proportion de cases valant $1^B$ dans la matrice binaire est élevée (ligne $\varepsilon = 0.4999$), les blocs binaires obtenus sont alors plus homogènes et la pénalisation n'est pas suffisante pour sélectionner un modèle ternaire. Dans ces cas, les nombres de classes sélectionnés sont souvent plus petits que ceux ayant servi aux simulations.

D'un autre côté, lorsque l'une des valeurs binarisées est sous-représentée par rapport à l'autre (colonne $\eta = 0.001$), les valeurs étant presque binaires, les modèles ternaires ne sont alors pas toujours choisis.

En revanche, lorsque les configurations ne sont pas proches de ces cas, le modèle ternaire est choisi plus souvent.

**Critère $ICL_{(4,1)}$**

| $\varepsilon$ \ $\eta$ | 0.001 | 0.01 | 0.1 | 0.4 | 0.45 | 0.4999 |
|---|---|---|---|---|---|---|
| 0.001 | 11 | 0 | 0 | 0 | 0 | 0 |
| 0.01 | 20 | 0 | 0 | 0 | 0 | 0 |
| 0.1 | 47 | 0 | 0 | 0 | 0 | 0 |
| 0.4 | 45 | 36 | 0 | 0 | 0 | 9 |
| 0.45 | 45 | 40 | 25 | 1 | 0 | 9 |
| 0.4999 | 43 | 38 | 50 | 50 | 50 | 50 |

**Critère *BIC***

| $\varepsilon$ \ $\eta$ | 0.001 | 0.01 | 0.1 | 0.4 | 0.45 | 0.4999 |
|---|---|---|---|---|---|---|
| 0.001 | 4 | 0 | 0 | 0 | 0 | 0 |
| 0.01 | 12 | 1 | 0 | 0 | 0 | 0 |
| 0.1 | 29 | 7 | 0 | 0 | 0 | 0 |
| 0.4 | 24 | 25 | 0 | 0 | 0 | 9 |
| 0.45 | 24 | 22 | 0 | 2 | 0 | 10 |
| 0.4999 | 27 | 12 | 1 | 50 | 50 | 50 |

TABLE 4.5 – Nombre de fois où un modèle binaire a été sélectionné suivant les critères ($ICL_{(4,1)}$ à gauche et *BIC* à droite) : en colonne la disproportion des valeurs 1 et 2 et en ligne la proportion de cases valant $1^B$ dans la matrice binaire.

### Données réelles

Nous avons ensuite testé la même procédure sur les données du parlement américain en étudiant les différentes façons de regrouper les votes. Nous voyons dans le tableau 4.6 que les meilleurs critères ternaires contraints sont obtenus pour le regroupement des "pour" et des "contre" (*exprimés* dans le tableau). Malgré la forte pénalisation, les effectifs sont assez proches et la matrice comporte une grande majorité de valeurs binarisées impliquant des blocs avec des valeurs fortement dominantes ; les valeurs des critères

avant pénalisation sont donc très élevées. D'ailleurs, le nombre de classes sélectionné pour ce regroupement est très faible en comparaison des autres. Toutefois, les données ternaires restent sélectionnées par les deux critères.

De plus, les critères *BIC* sélectionnent à nouveau moins de classes que les critères $ICL_{(4,1)}$.

**Valeurs**

| | $ICL_{(4,1)}$ | *BIC* |
|---|---|---|
| Contr & Abs | -6000.3 | -6048.4 |
| Pour & Abs | -6065.3 | -6088.0 |
| Exprimés | -5836.0 | -5825.0 |
| Ternaire | -4492.8 | -4684.2 |

**Couples sélecionnés**

| | $ICL_{(4,1)}$ | *BIC* |
|---|---|---|
| Contr & Abs | (5,12) | (3,8) |
| Pour & Abs | (4,10) | (3,7) |
| Exprimés | (3,3) | (3,3) |
| Ternaire | (5,7) | (4,6) |

TABLE 4.6 – Valeurs des critères suivant les regroupements (à gauche) et couples sélectionnés pour chacun (à droite).

## 4.4 Conclusion

Dans ce chapitre, nous obtenons le critère de sélection de modèle $ICL_{(4,1)}$ possédant une formule explicite non asymptotique et donnant de bons résultats sur des données simulées. De plus, nous pouvons espérer que ce critère soit consistant pour l'estimation du nombre de classes.

En revanche, le critère *BIC* utilisant l'énergie libre semble sous-évaluer le nombre de classes en ligne et en colonne lorsque le nombre d'observation est faible. Ceci est certainement dû aux deux approximations faites, l'une par l'énergie libre et l'autre asymptotique.

Toutefois, nous avons conjecturé que, pour un nombre suffisant d'observations, le comportement du critère *BIC* est similaire à celui du critère $ICL_{(4,1)}$, confortés par les résultats numériques ; il reste toutefois à le démontrer.

Enfin, les critères pénalisés proposés pour quantifier la perte d'information du passage du ternaire au binaire renvoient des résultats cohérents empiriquement : ils sélectionnent des modèles binaires lorsque les données binarisées sont fortement majoritaires (et qu'un nombre de blocs bien inférieur peut être choisi pour la matrice binaire), lorsque les données binarisées sont composées d'effectifs égaux de chaque ou encore lorsque l'une des données est fortement sous-représentées signifiant que le tableau en lui-même est quasiment binaire.

## 4.A Critère $ICL_{(a,b)}$

Dans cette annexe, nous regroupons les démonstrations relatives au critère $ICL_{(a,b)}$.

### 4.A.1 Formule explicite

Dans cette annexe, nous explicitons les calculs du critère exact $ICL_{(a,b)}(g,m)$ pour les données qualitatives. Comme nous l'avons expliqué, nous avons la formule :

$$ICL_{(a,b)}(g,m) = \log p(\mathbf{z}) + \log p(\mathbf{w}) + \log p(\mathbf{x}|\mathbf{z},\mathbf{w}).$$

Maintenant, conformément à la définition du modèle des blocs latents, nous avons :

$$p(\mathbf{z}|\boldsymbol{\pi}) = \prod_{i,k} \pi_k^{z_{ik}}.$$

Comme la loi a priori de $\boldsymbol{\pi}$ est une loi de Dirichlet $\mathcal{D}(a,\ldots,a)$ :

$$p(\boldsymbol{\pi}) = \frac{\Gamma(ga)}{\Gamma(a)^g} \prod_k \pi_k^{a-1},$$

et nous obtenons :

$$p(\boldsymbol{\pi}|\mathbf{z}) = \frac{p(\mathbf{z}|\boldsymbol{\pi})p(\boldsymbol{\pi})}{\int p(\mathbf{z}|\boldsymbol{\pi})p(\boldsymbol{\pi})} \propto \prod_k \pi_k^{z_{+k}+a-1}.$$

Nous reconnaissons une densité non normalisée de Dirichlet $\mathcal{D}(z_{+1}+a,\ldots,z_{+g}+a)$ dont le facteur de normalisation vaut :

$$\frac{\Gamma(n+ga)}{\prod_k \Gamma(z_{+k}+a)}.$$

Finalement, l'expression de $p(\mathbf{z})$ découle directement de la formule de Bayes :

$$p(\mathbf{z}) = \frac{\Gamma(ag)}{\Gamma(a)^g} \frac{\prod_k \Gamma(z_{+k}+a)}{\Gamma(n+ag)}. \tag{4.11}$$

De la même manière, nous avons :

$$p(\mathbf{w}) = \frac{\Gamma(am)}{\Gamma(a)^m} \frac{\prod_\ell \Gamma(w_{+\ell}+a)}{\Gamma(d+am)}. \tag{4.12}$$

Il reste à calculer la forme de $p(\mathbf{x}|\mathbf{z},\mathbf{w})$. Comme les variables $\boldsymbol{\alpha}$ et $(\mathbf{z},\mathbf{w})$ sont indépendantes, nous avons

$$p(\boldsymbol{\alpha}|\mathbf{x},\mathbf{z},\mathbf{w}) = \frac{p(\mathbf{x}|\mathbf{z},\mathbf{w},\boldsymbol{\alpha})p(\boldsymbol{\alpha})}{\int p(\mathbf{x}|\mathbf{z},\mathbf{w},\boldsymbol{\alpha})p(\boldsymbol{\alpha})d\boldsymbol{\alpha}},$$

et, en utilisant l'indépendance conditionnelle des cases $x_{ij}$ connaissant $\mathbf{z}$, $\mathbf{w}$ et $\boldsymbol{\alpha}$,

$$\begin{aligned} p(\mathbf{x}|\mathbf{z},\mathbf{w},\boldsymbol{\alpha}) &= \prod_{i,j,k,\ell} \left( \prod_h (\alpha_{k\ell}^h)^{v_{ijh}} \right)^{z_{ik}w_{j\ell}} \\ &= \prod_{k,\ell} \left( \prod_h (\alpha_{k\ell}^h)^{\sum_{i,j} z_{ik}w_{j\ell}v_{ijh}} \right) \\ &= \prod_{k,\ell} \left( \prod_h (\alpha_{k\ell}^h)^{N_{k\ell;\mathbf{z},\mathbf{w}}^h} \right). \end{aligned}$$

Ce qui entraîne

$$\begin{aligned} p(\boldsymbol{\alpha}|\mathbf{x},\mathbf{z},\mathbf{w}) &\propto \prod_{k,\ell}\left(p(\alpha_{k\ell})\prod_h (\alpha_{k\ell}^h)^{N^h_{k\ell;\mathbf{z},\mathbf{w}}}\right) \\ &\propto \prod_{k,\ell}\left[\left(\prod_h (\alpha_{k\ell}^h)^{b-1}\right)\left(\prod_h (\alpha_{k\ell}^h)^{N^h_{k\ell;\mathbf{z},\mathbf{w}}}\right)\right] \\ &\propto \prod_{k,\ell}\left(\prod_h (\alpha_{k\ell}^h)^{b+N^h_{k\ell;\mathbf{z},\mathbf{w}}-1}\right), \end{aligned}$$

car $p(\alpha_{k\ell})$ est la densité d'une loi de Dirichlet $\mathcal{D}(b,\ldots,b)$. Chaque terme $(k,\ell)$ est la densité non-normalisée d'une loi de Dirichlet $\mathcal{D}(b+N^1_{k\ell;\mathbf{z},\mathbf{w}},\ldots,b+N^r_{k\ell;\mathbf{z},\mathbf{w}})$ avec comme facteur de normalisation

$$\frac{\Gamma(z_{+k}w_{+\ell}+rb)}{\prod_h \Gamma(N^h_{k\ell;\mathbf{z},\mathbf{w}}+b)}.$$

Ce qui donne la formule suivante :

$$p(\mathbf{x}|\mathbf{z},\mathbf{w}) = \prod_{k,\ell}\frac{\Gamma(rb)}{\Gamma(b)^r}\frac{\prod_h \Gamma(N^h_{k\ell;\mathbf{z},\mathbf{w}}+b)}{\Gamma(z_{+k}w_{+\ell}+rb)}. \tag{4.13}$$

Au final, le critère $ICL_{(a,b)}(g,m)$, présenté dans la section 4.2.1, est directement dérivé des équations (4.3), (4.11), (4.12) et (4.13).

### 4.A.2 Comportement théorique

Pour étudier le comportement théorique du critère $ICL_{(a,b)}$ pour un couple $(\mathbf{z},\mathbf{w})$ fixé, nous commençons par reprendre l'équation (4.3) :

$$\log p(\mathbf{x},\mathbf{z},\mathbf{w}) = \log p(\mathbf{z}) + \log p(\mathbf{w}) + \log p(\mathbf{x}|\mathbf{z},\mathbf{w}).$$

Comme le couple $(\mathbf{z},\mathbf{w})$ est fixé, il suffit de regarder $\log p(\mathbf{x}|\mathbf{z},\mathbf{w})$ dont nous rappelons la forme :

$$\log p(\mathbf{x}|\mathbf{z},\mathbf{w}) = \sum_{k,\ell}\left[\log\Gamma(rb) - r\log\Gamma(b) - \log\Gamma\left(z_{+k}w_{+\ell}+rb\right) + \sum_{h=1}^{r}\log\Gamma(N^h_{k\ell;\mathbf{z},\mathbf{w}}+b)\right].$$

En notant $B$ la fonction bêta multinomiale définie pour tout $\beta\in\mathbb{R}^r_{+\star}$ par :

$$B(\beta) = \frac{\prod_{h=1}^r \Gamma(\beta_h)}{\Gamma\left(\sum_{h=1}^r \beta_h\right)},$$

nous avons :

$$\log p(\mathbf{x}|\mathbf{z},\mathbf{w}) = \sum_{k,\ell}\left[-\log B(b,\ldots,b) + \log B\left(N^1_{k\ell;\mathbf{z},\mathbf{w}}+b,\ldots,N^r_{k\ell;\mathbf{z},\mathbf{w}}+b\right)\right].$$

Or, nous savons que si nous imposons une contrainte sur la somme des paramètres, la fonction $B$ est minimale pour des valeurs égales et maximale si toutes les coordonnées sont nulles sauf une.

En conclusion, pour une configuration $(\mathbf{z},\mathbf{w})$, la log-vraisemblance complète $\log p(\mathbf{x},\mathbf{z},\mathbf{w})$ est plus élevée pour des blocs homogènes que pour des blocs avec des proportions équilibrées de chaque niveau de réponse.

## 4.B Démonstration de la partie "quantifier une perte d'informations"

Dans cette partie, nous mettons les démonstrations relatives à la partie 4.3.

### 4.B.1 Densité binaire contre densité ternaire

Pour la suite nous supposons que si $x^T \in \{0,1,2\}$ alors nous notons $x^B$ telle que :

$$\begin{cases} x^B = 0 & \text{si } x^T = 0 \\ x^B = 1^B & \text{si } x^T = 1 \text{ ou } x^T = 2 \end{cases}.$$

Ainsi, $x^B$ est la version binarisée de $x^T$ où les valeurs 1 et 2 sont regroupées derrière la valeur $1^B$.

Montrons maintenant que si nous supposons que $x^T$ est le résultat d'une variable ternaire $X^T$ et $x^B$ celle d'une variable binaire $X^B$ alors la densité associée à la variable binaire est toujours plus grande en $x^B$ que celle de la ternaire en $x^T$.

Pour cela, notons $\alpha_i$ la probabilité que la variable $X^T$ vale $i$ et $\alpha_B$ la probabilité que la variable $X^B$ vale $1^B$. Nous obtenons ainsi :

$$\begin{aligned} (1-\alpha_B) = \mathbb{P}(X^B = 0) &= \mathbb{P}(X^T = 0) = \alpha_0 \\ \alpha_B = \mathbb{P}\left(X^B = 1^B\right) &= \mathbb{P}(X^T = 1 \cup X^T = 2) = \alpha_1 + \alpha_2 \end{aligned}$$

et, comme les probabilités sont positives, cela signifie qu'il existe une constante $t \in [0,1]$ telle que :

$$\begin{cases} \alpha_1 &= & t\alpha_B \\ \alpha_2 &= & (1-t)\alpha_B \end{cases}.$$

Au final, en notant $\varphi_B$ et $\varphi_T$ les densités respectivement binaires et ternaires, nous avons :

$$\begin{aligned} \varphi_T(x^T) &= \alpha_0^{\mathbb{1}_{\{x^T=0\}}} \alpha_1^{\mathbb{1}_{\{x^T=1\}}} \alpha_2^{\mathbb{1}_{\{x^T=2\}}} \\ &= (1-\alpha_B)^{\mathbb{1}_{\{x^T=0\}}} (t\alpha_B)^{\mathbb{1}_{\{x^T=1\}}} ((1-t)\alpha_B)^{\mathbb{1}_{\{x^T=2\}}} \\ &= (1-\alpha_B)^{\mathbb{1}_{\{x^T=0\}}} \alpha_B^{\mathbb{1}_{\{x^T=1\}}+\mathbb{1}_{\{x^T=2\}}} \underbrace{t^{\mathbb{1}_{\{x^T=1\}}}(1-t)^{\mathbb{1}_{\{x^T=2\}}}}_{\leq 1} \\ &\leq (1-\alpha_B)^{\mathbb{1}_{\{x^T=0\}}} \alpha_B^{\underbrace{\mathbb{1}_{\{x^T=1\}}+\mathbb{1}_{\{x^T=2\}}}_{=\mathbb{1}_{\{x^B=1^B\}}}} \\ &\leq (1-\alpha_B)^{\mathbb{1}_{\{x^B=0\}}} \alpha_B^{\mathbb{1}_{\{x^B=1^B\}}} \\ &\leq \varphi_B(x^B). \end{aligned}$$

Comme la densité est plus forte, la vraisemblance l'est aussi.

#### Critère *BIC*

En particulier, pour le critère *BIC*, la partie ternaire a forcément une vraisemblance plus faible et est plus pénalisée donc n'est jamais sélectionnée.

**Critère $ICL_{(a,b)}$**

Dans cette partie, nous montrons que le critère $ICL_{(a,b)}$ choisit toujours le modèle binaire lorsque $b$ est entier. Pour cela, nous supposons avoir le même couple $(\mathbf{z}, \mathbf{w})$ puis nous comparons les log-vraisemblances complètes intégrées du cas ternaire et du cas binaire :

$$\begin{aligned}
\log p_T(\mathbf{x},\mathbf{z},\mathbf{w}) - \log p_B(\mathbf{x},\mathbf{z},\mathbf{w}) &= \sum_{k,\ell} \Big[ \log\Gamma(3b) - \log\Gamma(2b) - \log\Gamma(b) - \log\Gamma\left(b + N^2_{k\ell;\mathbf{z},\mathbf{w}} + N^3_{k\ell;\mathbf{z},\mathbf{w}}\right) \\
&\quad + \log\Gamma\left(b + N^2_{k\ell;\mathbf{z},\mathbf{w}}\right) + \log\Gamma\left(b + N^3_{k\ell;\mathbf{z},\mathbf{w}}\right) \\
&\quad - \log\Gamma\left(3b + z_{+k}w_{+\ell}\right) + \log\Gamma\left(2b + z_{+k}w_{+\ell}\right) \Big] \\
&= \sum_{k,\ell} \log \left[ \frac{(3b-1)!}{(2b-1)!\,(b-1)!} \frac{\left(b + N^2_{k\ell;\mathbf{z},\mathbf{w}} - 1\right)!\left(b + N^3_{k\ell;\mathbf{z},\mathbf{w}} - 1\right)!}{\left(b + N^2_{k\ell;\mathbf{z},\mathbf{w}} + N^3_{k\ell;\mathbf{z},\mathbf{w}} - 1\right)!} \right. \\
&\quad \left. \times \frac{(2b + z_{+k}w_{+\ell} - 1)!}{(3b + z_{+k}w_{+\ell} - 1)!} \right] \\
&= \sum_{k,\ell} \log \left[ \frac{(2b)\dots(3b-1)}{(b-1)!} \times \frac{1}{(2b + z_{+k}w_{+\ell})\dots(3b + z_{+k}w_{+\ell} - 1)!} \right. \\
&\quad \left. \times \frac{\left(b + N^3_{k\ell;\mathbf{z},\mathbf{w}} - 1\right)!}{\left(b + N^2_{k\ell;\mathbf{z},\mathbf{w}} - 1\right)\dots\left(b + N^2_{k\ell;\mathbf{z},\mathbf{w}} + N^3_{k\ell;\mathbf{z},\mathbf{w}} - 1\right)} \right] \\
&= \sum_{k,\ell} \log \left[ \underbrace{\frac{(b-1)!}{(b-1)!}}_{=1} \underbrace{\frac{(b)\dots\left(b + N^3_{k\ell;\mathbf{z},\mathbf{w}} - 1\right)}{\left(b + N^2_{k\ell;\mathbf{z},\mathbf{w}} - 1\right)\dots\left(b + N^2_{k\ell;\mathbf{z},\mathbf{w}} + N^3_{k\ell;\mathbf{z},\mathbf{w}} - 1\right)}}_{\leq 1} \right. \\
&\quad \left. \times \underbrace{\frac{(2b)\dots(3b-1)}{(2b + z_{+k}w_{+\ell})\dots(3b + z_{+k}w_{+\ell} - 1)!}}_{\leq 1} \right] \\
&< 0.
\end{aligned}$$

Donc, pour toutes les partitions, le critère binaire associé est meilleur que le ternaire.

## 4.B.2 Densité contrainte

À partir de l'équation (4.10) sur les densités :

$$\varphi_{TC}(x^B_{ij}; \alpha^B) = \frac{1}{2^{x^B_{ij}}} \varphi_B(x^B_{ij}; \alpha^B),$$

nous montrons dans cette partie comment obtenir les deux critères pénalisés.

Pour cela, nous commençons par exprimer la vraisemblance du ternaire contraint en fonction de celle

du binaire :

$$\begin{aligned}
p_{TC}\left(\mathbf{x}^B;\boldsymbol{\theta}^B\right) &= \sum_{(z,\mathbf{w})\in\mathcal{Z}\times\mathcal{W}}\prod_{i,k}\pi_k^{z_{ik}}\prod_{j,\ell}\rho_\ell^{w_{j\ell}}\prod_{i,j,k,\ell}\varphi_{TC}\left(\mathbf{x}^B;\alpha_{k\ell}^B\right)^{z_{ik}w_{j\ell}}\\
&= \sum_{(z,\mathbf{w})\in\mathcal{Z}\times\mathcal{W}}\prod_{i,k}\pi_k^{z_{ik}}\prod_{j,\ell}\rho_\ell^{w_{j\ell}}\prod_{i,j,k,\ell}\frac{1}{2^{x_{ij}^B z_{ik}w_{j\ell}}}\varphi_B\left(\mathbf{x}^B;\alpha_{k\ell}^B\right)^{z_{ik}w_{j\ell}}\\
&= \sum_{(z,\mathbf{w})\in\mathcal{Z}\times\mathcal{W}}\frac{1}{2^{\sum_{i,j,k,\ell}x_{ij}^B z_{ik}w_{j\ell}}}\prod_{i,k}\pi_k^{z_{ik}}\prod_{j,\ell}\rho_\ell^{w_{j\ell}}\prod_{i,j,k,\ell}\varphi_B\left(\mathbf{x}^B;\alpha_{k\ell}^B\right)^{z_{ik}w_{j\ell}}\\
&= \sum_{(z,\mathbf{w})\in\mathcal{Z}\times\mathcal{W}}\frac{1}{2^{N_1^B}}\prod_{i,k}\pi_k^{z_{ik}}\prod_{j,\ell}\rho_\ell^{w_{j\ell}}\prod_{i,j,k,\ell}\varphi_B\left(\mathbf{x}^B;\alpha_{k\ell}^B\right)^{z_{ik}w_{j\ell}}\\
&= \frac{1}{2^{N_1^B}}p_B\left(\mathbf{x}^B;\boldsymbol{\theta}^B\right).
\end{aligned}$$

La version pénalisée du critère *BIC* découle de cette constatation.

Pour le critère $ICL_{(a,b)}$, nous voyons sur la formule (4.4) que l'influence de la densité se fait dans la partie $p(\mathbf{x}|\mathbf{z},\mathbf{w})$. En reprenant la démonstration de l'annexe 4.A.1, nous avons :

$$\begin{aligned}
p_{TC}\left(\mathbf{x}^B\middle|\,\mathbf{z},\mathbf{w}\right) &= \int p_{TC}\left(\mathbf{x}^B\middle|\,\mathbf{z},\mathbf{w},\boldsymbol{\alpha}^B\right)p(\boldsymbol{\alpha}^B)d\boldsymbol{\alpha}^B\\
&= \int\frac{1}{2^{N_1^B}}p_B\left(\mathbf{x}^B\middle|\,\mathbf{z},\mathbf{w},\boldsymbol{\alpha}^B\right)p(\boldsymbol{\alpha}^B)d\boldsymbol{\alpha}^B\\
&= \frac{1}{2^{N_1^B}}p_B\left(\mathbf{x}^B\middle|\,\mathbf{z},\mathbf{w}\right)
\end{aligned}$$

ce qui donne le résultat.

# Chapitre 5

# Algorithme Largest Gaps

*"Tout est difficile avant d'être simple"*
Thomas Fuller

## Sommaire


**5.1 Introduction . . . . . 123**
**5.2 Algorithme *Largest Gaps* (*LG*) . . . . . 123**
**5.3 Consistance des estimateurs . . . . . 124**
5.3.1 Consistance sous la condition que les paramètres soient fixes . . . . . 127
5.3.2 Étude de différentes asymptotiques . . . . . 128
**5.4 Estimateurs empiriques de $\boldsymbol{\theta}$ . . . . . 129**
**5.5 Amélioration de l'estimation des classes . . . . . 130**
**5.6 Sélection de modèle . . . . . 131**
**5.7 Applications numériques . . . . . 134**
5.7.1 Comportement de la borne 5.2 . . . . . 134
5.7.2 Étude de l'amélioration . . . . . 137
5.7.3 Étude de la sélection de modèles . . . . . 137
**5.8 Conclusion . . . . . 138**
**5.A Indépendance des cases d'une ligne conditionnellement à l'appartenance à une classe de cette dernière . . . . . 138**
**5.B Démonstration de la consistance de $\widehat{z}^{LG}$ . . . . . 139**
5.B.1 Évènement idéal . . . . . 139
5.B.2 Borne sur les probabilités . . . . . 140
5.B.3 Consistance . . . . . 142
**5.C Calcul des conditions pour la section 5.3.2 . . . . . 142**
5.C.1 Évolution de $\delta_\tau^{n,d}$ . . . . . 142
5.C.2 Evolution de $\pi_{\min}^{n,d}$ . . . . . 143
**5.D Démonstration de la consistance des paramètres empiriques . . . . . 144**
5.D.1 Paramètres fixes . . . . . 144
5.D.2 Conditions limites . . . . . 146
**5.E Algorithme d'initialisation . . . . . 149**
**5.F Démonstrations sur la sélection de modèles . . . . . 149**
5.F.1 Borne de l'estimation de $\widehat{g}^{LG}$ à $n$, $d$ et $S$ fixés . . . . . 149
5.F.2 Consistance des estimateurs du nombre de classes . . . . . 151
5.F.3 Consistance de l'algorithme $LG$ . . . . . 151

## 5.1 Introduction

Dans ce chapitre, nous adaptons l'algorithme *Largest Gaps* (ou *LG*) proposé par Channarond et al. (2012) au modèle des blocs latents. Le principe de cet algorithme est d'estimer les labels à partir des sommes des cases de chaque ligne et colonne.

Dans leur article, Channarond et al. (2012) étudient cette procédure sur le modèle SBM où les nombres de lignes et de colonnes sont identiques ; dans le cas du modèle des blocs latents, la principale contrainte est que les lignes et les colonnes peuvent augmenter de façon disproportionnée.

En adaptant les démonstrations de leur article, nous démontrons que les estimateurs des labels et des paramètres renvoyés par l'algorithme sont consistants sous la condition que le rapport entre les lignes et les colonnes ne tende pas trop vite vers 0 ou l'infini.

Contrairement au cas du SBM, les sommes des cases des lignes sont indépendantes conditionnellement à la connaissance des labels, nous proposons d'améliorer les estimations de l'algorithme *LG* en utilisant un algorithme *EM*.

De plus, nous proposons une adaptation différente pour que l'algorithme *LG* propose une estimation du nombre de classes. Nous montrons que cette procédure permet d'obtenir des estimateurs consistants également.

Enfin, nous testons les résultats théoriques sur des simulations. Nous montrons plus particulièrement que, malgré sa simplicité, l'algorithme *LG* peut servir pour estimer le nombre de classes, les partitions et les paramètres mais qu'il faut un très grand nombre d'observations.

## 5.2 Algorithme *Largest Gaps* (*LG*)

L'algorithme *Largest Gaps* repose sur une simplification du modèle : le principe est de regarder la loi d'une case $X_{ij}$ de la matrice sachant que sa ligne appartient à la classe $k$ indépendamment du classement des colonnes :

$$\begin{aligned}\mathbb{P}(X_{ij}=1|z_{ik}=1) &= \sum_{\ell=1}^{m}\mathbb{P}(X_{ij}=1|z_{ik}=1,w_{j\ell}=1)\mathbb{P}(w_{j\ell}=1|z_{ik}=1)\\ &= \sum_{\ell=1}^{m}\mathbb{P}(X_{ij}=1|z_{ik}=1,w_{j\ell}=1)\mathbb{P}(w_{j\ell}=1)\\ &= \sum_{\ell=1}^{m}\alpha_{k\ell}\rho_{\ell}.\end{aligned}$$

En reprenant les notations $\tau_k = \sum_{\ell=1}^{m}\alpha_{k\ell}\rho_\ell$ du critère d'identifiabilité de Keribin et al. (section 1.3.2.1) et conditionnellement au fait que la ligne $i$ appartienne à la classe $k$, les cases $X_{ij}$ sont indépendantes (voir section 5.A) et la somme $X_{i+}$ des cases de la ligne $i$ suit une loi binomiale de paramètres $d$ et $\tau_k$ :

$$X_{i+}|z_{ik}=1 \sim \mathcal{B}(d,\tau_k).$$

L'idée est alors que si les composantes du vecteur $\boldsymbol{\tau}$ sont suffisamment distinctes par rapport au nombre d'observations, les sommes de chaque ligne doivent s'agglutiner autour de $g$ groupes provenant chacun d'une même loi binomiale. Comme nous pouvons le voir sur la figure 5.1, il est possible d'obtenir la partition ayant servi à simuler les données lorsque les groupes sont suffisamment séparés.

Pour automatiser cette procédure, nous étudions la variable $\overline{X_{i\bullet}}$ représentant la moyenne de la $i$ ème ligne et ayant les propriétés suivantes :

**Propriété 5.2.1.** *Les résultats théoriques de l'algorithme Largest Gaps reposent sur les cinq propriétés suivantes de* $\overline{X_{i\bullet}}$ *:*

- $\overline{X_{i\bullet}} \in [0,1]$.
- $\mathbb{E}\left[\overline{X_{i\bullet}} \middle| z_{ik}=1\right] = \frac{\mathbb{E}[X_{i+}|z_{ik}=1]}{d} = \tau_k$.
- $\mathbb{V}\left[\overline{X_{i\bullet}} \middle| z_{ik}=1\right] = \frac{\mathbb{V}[X_{i+}|z_{ik}=1]}{d^2} = \frac{\tau_k(1-\tau_k)}{d}$.
- *Par la loi des grands nombres,* $\overline{X_{i\bullet}} \middle| z_{ik}=1 \underset{d\to+\infty}{\longrightarrow} \tau_k$ *p.s.*
- *D'après le théorème de limite centrale, la loi limite de* $\overline{X_{i\bullet}} \middle| z_{ik}=1$ *est* $\mathcal{N}\left(\tau_k, \frac{\tau_k(1-\tau_k)}{d}\right)$.

De ces constatations, l'algorithme $LG$ est ainsi défini :

---

**Algorithme Largest Gaps** :

---

1. Calcul pour tout $i \in \{1,\dots,n\}$, la moyenne $\overline{X_{i\bullet}}$ de la ligne $i$.
2. Ordonnancement dans l'ordre croissant pour former le groupe $\left(\overline{X_{(1)\bullet}},\dots,\overline{X_{(n)\bullet}}\right)$.
3. Calcul pour tout $i \in \{2,\dots,n\}$, les sauts $G_i = \overline{X_{(i)\bullet}} - \overline{X_{(i-1)\bullet}}$.
4. Sélection de $i_1,\dots,i_{g-1}$ (ordonnés) tels que $(G_{i_1},\dots,G_{i_{g-1}})$ soient les $g-1$ plus grands sauts.
5. Création les groupes tels que $z_{ik}=1$ si $(i_{k-1}) < (i) \leq (i_k)$ avec $i_0=0$ et $i_g=n$.
6. Répétition avec les colonnes pour obtenir $m$ groupes.

---

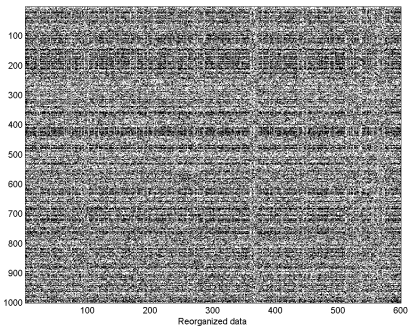
Nous obtenons ainsi deux estimateurs des partitions, l'un en ligne $\widehat{z}^{LG}$ et l'autre en colonne $\widehat{w}^{LG}$. La figure 5.2 représente les différentes étapes illustrées à partir de l'exemple de la figure 5.1.

*Remarque* 5.2.2. **Cas de données qualitatives**
Dans le cas de données qualitatives, nous pouvons faire la somme sur les lignes et les colonnes des matrices $v_{ijh}$. Nous obtenons ainsi pour chaque ligne un vecteur $(v_{i+1},\dots,v_{i+r})$ ayant pour loi une multinomiale de paramètres $d$ et $\tau_k = \left(\tau_k^1,\dots,\tau_k^r\right)$ conditionnellement au fait que la ligne $i$ appartienne à la classe $k$. L'affectation des classes peut alors se faire suivant l'une des coordonnées de la même façon que pour le cas binaire. Cette proposition a l'inconvénient de devoir connaître la coordonnée où les $\tau_k$ sont différents. Une alternative est proposée en section 5.5.

## 5.3 Consistance des estimateurs

Dans cette partie, nous montrons que l'estimateur $\widehat{z}^{LG}$ des labels est consistant lorsque $n$ et $d$ tendent vers l'infini sous la condition de connaître le bon nombre de classes et que $n$ ne croît pas trop vite devant $d$. Pour contourner le label switching, nous utilisons la relation d'équivalence $\equiv_{\mathcal{Z}}$ définie dans la partie 1.3.2.5 supposant que deux matrices de partitions binaires $z^1$ et $z^2$ sont équivalentes s'il existe une permutation $\mathfrak{s}$ des éléments de $\{1,\dots,g\}$ telle que pour tout $i$ et $k$, $z^1_{ik} = z^2_{i\mathfrak{s}(k)}$.

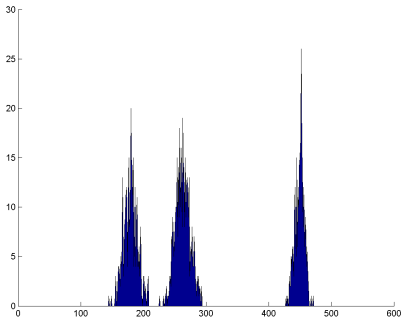
Matrice initiale

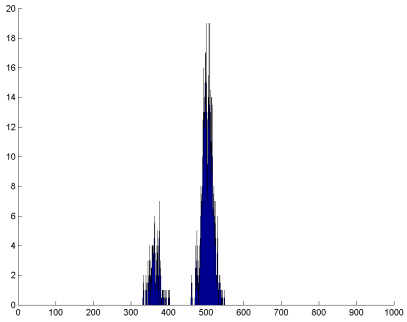



Lignes

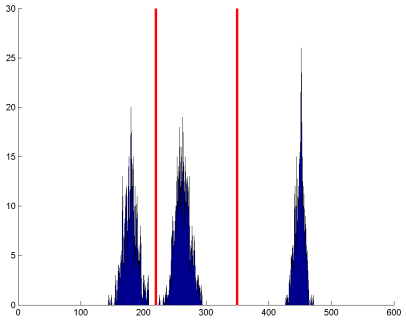
Colonnes

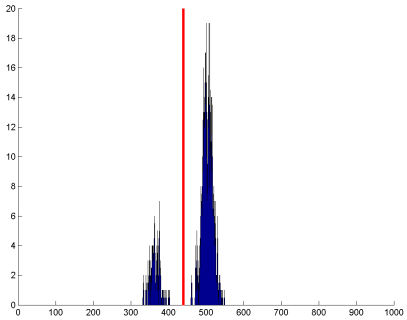
Histogrammes des sommes





Séparations





Matrice réorganisée

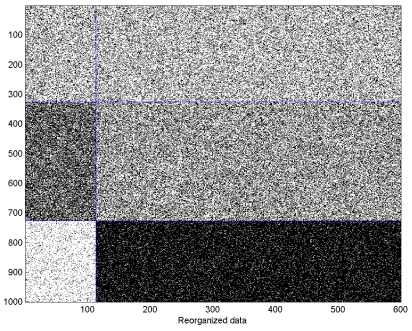



FIGURE 5.1 – Illustrations des constatations de la partie 5.2. La matrice initiale se trouve en haut, puis les histogrammes des sommes sont représentés sur la deuxième ligne ; nous voyons respectivement 3 et 2 classes mises en évidence sur la troisième ligne pour obtenir la matrice représentée en bas.

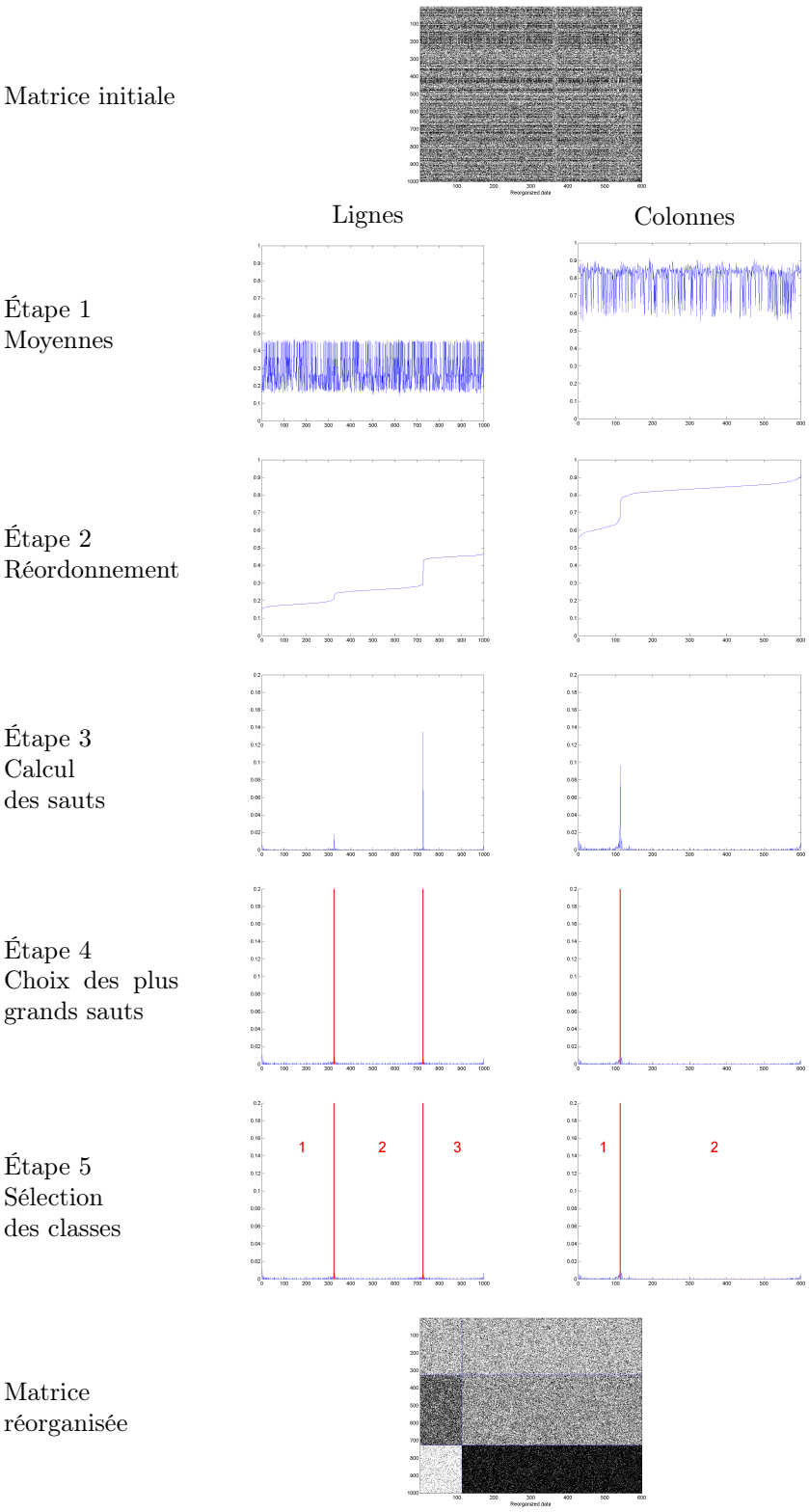



FIGURE 5.2 – Illustration de chaque étape de l'algorithme $LG$ à partir de l'exemple de la figure 5.1 : colonne de gauche pour les lignes et celle de droite pour les colonnes.

### 5.3.1 Consistance sous la condition que les paramètres soient fixes

Pour montrer la consistance de l'estimation de la classification en ligne, nous majorons la probabilité que la classification estimée ne soit pas la bonne en fonction de $n$ et de $d$ ; nous notons cet évènement $E_z = \left\{\widehat{z}^{LG} \not\equiv_{\mathcal{Z}} \mathbf{z}^\star\right\}$ où $\mathbf{z}^\star$ est la vraie partition. Pour cela, nous avons besoin d'un certain nombre de notations.

**Notations 5.3.1.**
La probabilité minimale d'appartenir à une classe en ligne est notée $\pi_{\min}$ :

$$\pi_{\min} = \min_{1\leq k\leq g} \pi_k.$$

La distance minimale entre deux valeurs de $\boldsymbol{\tau}$ de coordonnées distinctes est notée $\delta_\tau$ :

$$\delta_\tau = \min_{k\neq k'} \left|\tau_k - \tau_{k'}\right|.$$

De même, $\rho_{\min}$ représente la probabilité minimale d'appartenir à une classe en colonne et $\delta_\sigma$ la distance minimale entre deux valeurs de $\boldsymbol{\sigma}$ de coordonnées distinctes.

Si la condition $C_1$ du critère d'identifiabilité de Keribin et al. (section 1.3.2.1) n'est pas vérifiée alors $\delta_\tau = 0$.

À l'aide de ces notations, nous obtenons la borne suivante pour la probabilité de mauvais classement en ligne :

**Théorème 5.3.2.** ***Borne de l'évènement*** $E_z$
*Soit l'évènement* $E_z = \left\{\widehat{z}^{LG} \not\equiv_{\mathcal{Z}} \mathbf{z}^\star\right\}$, *alors :*

$$\mathbb{P}(E_z) \leq 2ne^{-\frac{1}{8}d\delta_\tau^2} + g(1-\pi_{\min})^n. \tag{5.1}$$

La démonstration est adaptée de celle de Channarond et al. (2012) et est mise en annexe 5.B. Elle repose essentiellement sur la recherche d'un évènement appartenant au complémentaire de $E_z$ et de l'utilisation de l'inégalité de Hoeffding.

*Remarque* 5.3.3.
Pour obtenir la consistance de l'estimateur $\widehat{z}^{LG}$, il suffit d'avoir la relation suivante :

$$\underline{\lim}_{n,d} \frac{d}{\log n} > \frac{8}{\delta_\tau^2}$$

où $\underline{\lim}$ représente la limite inférieure. La démonstration est mise en annexe 5.B.3.

En particulier, cette condition est vérifiée si :

$$\log n = o(d).$$

Par symétrie, une borne similaire est obtenue pour l'estimateur de l'affectation des labels en colonne en notant $\equiv_{\mathcal{W}}$ l'égalité de deux matrices de l'ensemble $\mathcal{W}$ à une permutation des labels des classes près.

À partir de ces deux bornes, nous montrons la consistance des estimateurs des labels en explicitant une borne tendant vers 0 de l'évènement $E_{z,w} = \left\{\widehat{z}^{LG} \not\equiv_{\mathcal{Z}} \mathbf{z}^\star\right\} \bigcup \left\{\widehat{w}^{LG} \not\equiv_{\mathcal{W}} \mathbf{w}^\star\right\}$ lorsque $n$ et $d$ tendent vers l'infini.

**Théorème 5.3.4.** ***Consistance des estimateurs des classes***
*Soit l'évènement* $E_{z,w} = \left\{\widehat{z}^{LG} \not\equiv_{\mathcal{Z}} \mathbf{z}^\star\right\} \bigcup \left\{\widehat{w}^{LG} \not\equiv_{\mathcal{W}} \mathbf{w}^\star\right\}$*, alors :*

$$\mathbb{P}(E_{z,w}) \leq 2ne^{-\frac{1}{8}d\delta_\tau^2} + g(1-\pi_{\min})^n + 2de^{-\frac{1}{8}n\delta_\sigma^2} + m(1-\rho_{\min})^d. \tag{5.2}$$

$[C_{n,d}]$ *: Si* $n$ *et* $d$ *tendent vers l'infini tels que* $\log n$ *soit négligeable devant* $d$ *et* $\log d$ *soit négligeable devant* $n$*,*
*alors les estimateurs des labels en ligne et en colonne renvoyés par l'algorithme LG sont consistants.*

La démonstration consiste à reprendre la borne (5.1) pour majorer l'ensemble sur les lignes et faire de même pour les colonnes.

*Remarque* 5.3.5.
La borne (5.2) permet d'avoir une première vision du déséquilibre entre les lignes et les colonnes puisqu'en reprenant la démonstration de l'annexe 5.B.3, nous conservons la consistance des estimateurs s'il existe $\varepsilon > 0$ et si $n$ et $d$ tendent vers l'infini tels que :

$$\frac{8}{\delta_\tau^2}\log n(1+\varepsilon) < d < e^{\frac{\delta_\sigma^2 n}{8}(1-\varepsilon)}.$$

### 5.3.2 Étude de différentes asymptotiques

Comme nous le mentionnons dans la section 1.3.2.3, il est intéressant d'étudier l'influence asymptotique de trois quantités (pour la classification des lignes) lorsque $n$ et $d$ tendent vers l'infini :

1. le nombre de classes $g^{n,d}$ tend vers l'infini,
2. la probabilité minimale $\pi_{\min}^{n,d}$ tend vers 0,
3. l'écart minimal $\delta_\tau^{n,d}$ entre deux valeurs de $\boldsymbol{\tau}$ de coordonnées distinctes tend vers 0.

*Remarque* 5.3.6.
Notons que le nombre $g^{n,d}$ de classes en ligne est borné à la fois par le comportement de $\pi_{\min}^{n,d}$ et celui de $\delta_\tau^{n,d}$ par :

$$g^{n,d} \leq \frac{1}{\pi_{\min}^{n,d}} \quad \text{et} \quad g^{n,d} \leq \frac{1}{\delta_\tau^{n,d}} + 1.$$

En particulier, il est possible que le nombre de classes en ligne $g^{n,d}$ soit constant alors que l'une des deux valeurs entre $\pi_{\min}^{n,d}$ et $\delta_\tau^{n,d}$ tende vers 0. Toutefois, ces bornes n'étant pas optimales, nous conservons la possibilité que le nombre de classes tende vers l'infini.

**Théorème 5.3.7.** ***Consistance des estimateurs des labels***
*Sous les conditions sur les lignes (resp. les conditions symétriques sur les colonnes) :*

*[C1.ligne] Condition sur* $\delta_\tau^{n,d}$ *:*

$$\underline{\lim}_{n,d}\, \delta_\tau^{n,d}\sqrt{\frac{d}{\log n}} > 2\sqrt{2}.$$

*[C2.ligne] Condition sur* $\pi_{\min}^{n,d}$ *:*
*(a)* $g^{n,d} \underset{n,d\to+\infty}{\longrightarrow} +\infty$ *et*

$$\underline{\lim}_{n,d}\, -\frac{n\log\left(1-\pi_{\min}^{n,d}\right)}{\log g^{n,d}} > 1,$$

*(b) ou*

$$\pi_{\min}^{n,d} n \underset{n,d\to+\infty}{\longrightarrow} +\infty.$$

*Et sous la condition* $[C_{n,d}]$ *que* $n$ *et* $d$ *tendent vers l'infini tels que* $\log n$ *soit négligeable devant* $d$ *et* $\log d$ *soit négligeable devant* $n$*,*
*alors l'estimateur* $\left(\widehat{z}^{LG}, \widehat{w}^{LG}\right)$ *est consistant.*

La démonstration repose sur l'étude de la borne (5.2) (mise en annexe 5.C).

*Remarque* 5.3.8.
Dans leur article, Channarond et al. (2012) ont donné une condition sur le nombre de classes que nous pouvons retrouver à l'aide de la remarque 5.3.6 :

$$g^{n,d} = \mathcal{O}\left(\sqrt{\frac{d}{\log n}}\right).$$

En particulier, la condition $[C2.ligne]$ est vérifiée si $\pi_{\min}^{n,d}$ est de la forme $\frac{1}{n^\gamma}$ pour tout $\gamma \in ]0;1[$.
Les démonstrations sont mises en annexe 5.C.

## 5.4 Estimateurs empiriques de $\boldsymbol{\theta}$

Une fois les estimateurs $\widehat{z}^{LG}$ et $\widehat{w}^{LG}$ obtenus, nous pouvons estimer le paramètre $\boldsymbol{\theta}$ par les moyennes empiriques suivantes :

$$\widehat{\pi}_k^{LG} = \frac{\widehat{z}_{+k}^{LG}}{n}, \quad \widehat{\rho}_\ell^{LG} = \frac{\widehat{w}_{+\ell}^{LG}}{d} \quad \text{et} \quad \widehat{\alpha}_{k\ell}^{LG} = \frac{\sum_{i,j} \widehat{z}_{ik}^{LG} x_{ij} \widehat{w}_{j\ell}^{LG}}{\widehat{z}_{+k}^{LG} \widehat{w}_{+\ell}^{LG}}.$$

Pour montrer la consistance de $\widehat{\theta}^{LG}$, nous majorons le fait que l'estimation soit loin de la vraie valeur $\boldsymbol{\theta}^\star$.

**Théorème 5.4.1.** ***Convergence des estimateurs empiriques***
*Pour tout* $t > 0$*, notons :*

$$\mathcal{C}_t^{z,w} = \left\{ \left\| \widehat{\theta}^{LG} - \boldsymbol{\theta}^\star \right\|_{+\infty} > t \right\}.$$

*Alors, pour tout* $C \in ]0,1[$ *:*

$$\begin{aligned} \mathbb{P}\left(\mathcal{C}_t^{z,w}\right) \quad \leq \quad & gm\left[2e^{-2\pi_{\min}\rho_{\min}ndt^2(1-C)} + 4e^{-(C\pi_{\min}\rho_{\min})^2 n} + 4e^{-(C\pi_{\min}\rho_{\min})^2 d}\right] \\ & + 2ge^{-2nt^2} + 2me^{-2dt^2} \\ & + 2ne^{-d\frac{\delta_\pi^2}{8}} + g(1-\pi_{\min})^n + 2de^{-n\frac{\delta_\rho^2}{8}} + m(1-\rho_{\min})^d. \end{aligned} \tag{5.3}$$

*Sous la condition* $[C_{n,d}]$*, les estimateurs empiriques de* $\boldsymbol{\theta}$ *renvoyés par l'algorithme LG sont consistants.*

*Remarque* 5.4.2. **Constante optimale**
La valeur optimale de la constante $C$ de cette borne dépend de $n$, $d$, $\pi_{\min}$ et $\rho_{\min}$ ; son calcul exact est difficile à obtenir. Toutefois, nous pouvons remarquer qu'à une constante $C$ fixée, la première partie de la borne est négligeable par rapport à chacune des deux autres parties (voir annexe 5.D.2). Par conséquent, nous pouvons conjecturer que la constante optimale tend vers 1 lorsque $n$ et $d$ tendent vers l'infini.

En supposant que les paramètres évoluent, nous obtenons de nouvelles conditions suffisantes à partir de la borne (5.3).

**Théorème 5.4.3.** *Sous la condition* $[C1]$ *du théorème 5.3.7 pour les lignes et les colonnes auxquelles nous ajoutons les conditions :*

*[C3.ligne] :*

*(a)* $g^{n,d}m^{n,d} \underset{n,d\to+\infty}{\longrightarrow} +\infty$ *et*

$$\underline{\lim}_{n,d} \frac{(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n}{\log\left(g^{n,d}m^{n,d}\right)} > 1,$$

*(b) ou*

$$(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n \underset{n,d\to+\infty}{\longrightarrow} +\infty.$$

*[C3.colonne] :*

*(a)* $g^{n,d}m^{n,d} \underset{n,d\to+\infty}{\longrightarrow} +\infty$ *et*

$$\underline{\lim}_{n,d} \frac{(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 d}{\log\left(g^{n,d}m^{n,d}\right)} > 1,$$

*(b) ou*

$$(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 d \underset{n,d\to+\infty}{\longrightarrow} +\infty.$$

*Et sous la condition* $[C_{n,d}]$ *sur le nombre de lignes et de colonnes, alors les estimateurs empiriques sont consistants.*

La démonstration est une simple étude de l'évolution de la borne suivant celles des paramètres (voir annexe 5.D.2).

*Remarque* 5.4.4.
Les deux nouvelles conditions sont dues à l'estimation empirique de $\boldsymbol{\alpha}$. Le fait que les lignes et les colonnes doivent croître très vite peut surprendre mais la variable $\pi_{\min}^{n,d}\rho_{\min}^{n,d}$ représente la proportion du plus petit bloc dans la matrice. Donc si cette proportion diminue, il faut un nombre de cases qui augmente en conséquence. Comme nous n'avons pas écarté l'hypothèse que la proportion minimale $\rho_{\min}^{n,d}$ d'appartenir à une classe en colonne tend vers 0 avec $n$, il faut que le nombre de lignes compense également.

## 5.5 Amélioration de l'estimation des classes

Dans la section 5.2, nous avons vu que les variables $X_{i+}$ suivent des lois de mélanges binomiaux que nous pouvons également approximer par des mélanges gaussiens. Dans sa version première, l'algorithme $LG$ ne prend pas en compte le fait qu'en sommant les lignes et les colonnes, nous obtenons des mélanges.

Une amélioration consiste à prendre en compte cette information, notamment lorsque le plus petit saut sélectionné est de taille proche de $1/n$ ou $1/d$ (ce qui est le cas si deux "bosses" sont imbriquées), en couplant avec un algorithme $EM$ (voir annexe A.3 ou Dempster et al. (1977)). Cette amélioration est notamment intéressante dans le cas de données qualitatives où distinguer les sauts est plus compliqué alors que nous obtenons assez facilement des mélanges de lois multinomiales.

Toutefois, l'un des avantages de l'algorithme $LG$ est sa rapidité, et nous savons que l'algorithme $EM$ est sensible aux initialisations. En règle générale, il est recommandé de lancer plusieurs fois l'algorithme avec différentes initialisations pour obtenir un bon résultat. Or, faire plusieurs initialisations ralentirait fortement la vitesse d'obtention d'un résultat pour l'algorithme. Deux types d'initialisations sont proposés :

— comme les lois des composantes sont assez distinctes, nous pouvons utiliser en initialisation un algorithme de type $K$-means (MacQueen, 1967) ou un algorithme $CEM$ (Celeux et Govaert, 1992) avec des lois binomiales relancés chacun avec plusieurs initialisations : ces algorithmes convergent

plus vite que l'algorithme $EM$ et donnent des résultats assez proches lorsque les composantes sont bien séparées. Ce deuxième choix est celui fait pour le cas de données qualitatives.
— Comme nous le voyons sur la figure 5.1, lorsque le nombre d'observations est suffisamment grand, nous pouvons presque voir à l'oeil nu l'emplacement des composantes de $\boldsymbol{\tau}$ aux pieds des sommets des composantes dans le cas de données binaires. Partant de ce constat, nous proposons une initialisation pouvant être vue comme une dérivée de l'algorithme *watershed* (Aaron, 2004) et illustrée sur la figure 5.3. Une formalisation algorithmique est proposée en annexe 5.E

À partir de ces remarques, l'algorithme $LG$ amélioré est le suivant :

---

**Algorithme Largest Gaps adapté** :

---

1. Calcul pour tout $i \in \{1, \dots, n\}$, $\overline{X_{i\bullet}}$.
2. Ordonnancement dans l'ordre croissant pour former le groupe $\left(\overline{X_{(1)\bullet}}, \dots, \overline{X_{(n)\bullet}}\right)$.
3. Initialisation en prenant l'un des algorithmes suivants :
   — $K$-means,
   — algorithme $CEM$,
   — initialisation proposée précédemment.
4. Utilisation l'algorithme $EM$ :
   — soit en supposant que les lois sont des binomiales,
   — soit en supposant que les lois sont des gaussiennes.

---

## 5.6 Sélection de modèle

Enfin, l'algorithme $LG$ peut servir pour sélectionner le nombre de classes : en regardant les graphiques de la figure 5.1 et en analysant la démonstration 5.B, nous voyons qu'au lieu de sélectionner les $g-1$ plus grands sauts, nous pouvons choisir tous les sauts plus grands qu'une valeur seuil notée $S$. En particulier, nous pouvons faire les constations suivantes :

— la valeur $S$ peut être différente pour le classement des lignes et des colonnes ; en particulier s'il y a une disproportion entre les lignes et les colonnes,
— si la valeur $S$ est trop grande, l'algorithme ne sélectionne pas les plus petits sauts,
— si la valeur $S$ est trop petite, l'algorithme sélectionne trop de sauts.

En notant $\widehat{g}^{LG}$ l'estimateur du vrai nombre de classes $g^\star$ fourni par cette modification de l'algorithme $LG$, nous avons le théorème suivant :

**Théorème 5.6.1.** ***Sélection du nombre de classes en ligne***
*Si la valeur $S$ est strictement positive et plus petite que $\delta_\tau$, alors :*

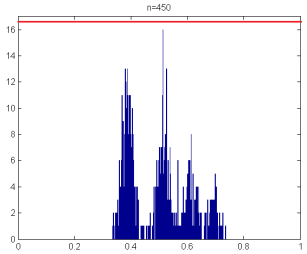
$$\mathbb{P}\left(\widehat{g}^{LG} \neq g^\star\right) \leq 2n\left[e^{-d\frac{S^2}{2}} + e^{-d\frac{(\delta_\tau - S)^2}{2}}\right] + g^\star\left(1 - \pi_{\min}\right)^n. \tag{5.4}$$

*En particulier, si $n$ et $d$ tendent vers l'infini tels que $\log n$ soit négligeable devant $d$,*
*alors l'estimateur du nombre de classes en ligne renvoyé par l'algorithme $LG$ est consistant pour toute valeur $S \in ]0, \delta_\tau[$.*

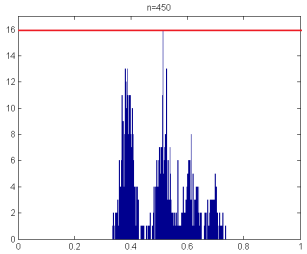
La démonstration est mise en annexe 5.F.1.

*Remarque* 5.6.2.
La valeur $S$ optimale est $\delta_\tau/2$.

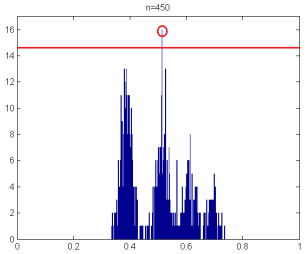
**Illustration pour un cas avec $n = 450$ lignes et $4$ classes**



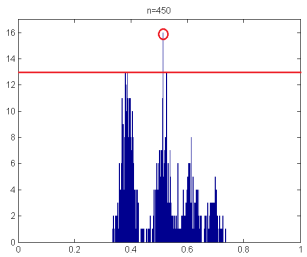
Étape 1 :
Histogramme des valeurs $\overline{X_{i\bullet}}$.



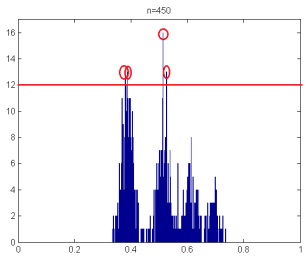
Étape 2 : $N_{\max} = 16$
Un seul pic d'une hauteur égale à la valeur maximale.



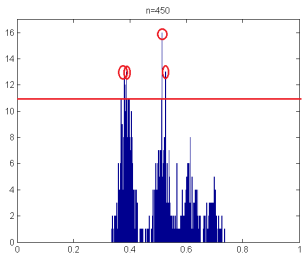
Étape 3 : $N = 15$
Aucun indice $j$ dont la valeur $N_j$ vaut 15.



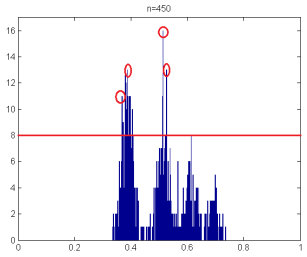
Étape 3 : $N = 13$
5 nouveaux indices apparaissent dont deux groupes de paires collées. L'algorithme ne conserve que les 3 plus éloignés.



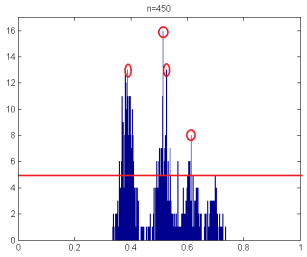
Étape 3 : $N = 12$
Le nouvel indice apparu est voisin de ceux du groupe $\mathcal{D}$.



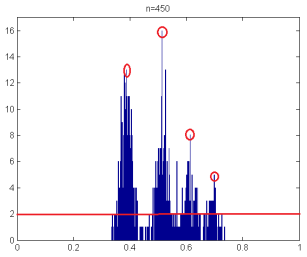
Étape 3 : $N = 11$
5 nouveaux indices dont un est voisin de ceux du groupe $\mathcal{D}$. L'algorithme conserve celui qui est le plus à gauche en éliminant un sélectionné précédemment.



Étape 3 : $N = 8$
Une nouvelle fois, un indice précédemment sélectionné est remplacé par le nouveau pique éloigné.



Étape 3 : $N = 5$
Le dernier pique est sélectionné.



Étape 3 : $N = 2$
L'algorithme ne sélectionnera plus d'autres piques. Nous avons nos 4 pics de bosses.

FIGURE 5.3 – Illustrations de l'algorithme d'initialisation pour un cas où les composantes sont imbriquées et où l'algorithme $LG$ aurait considéré une seule classe. À la fin, nous obtenons bien les 4 pics désirés. Dans cette version, une amélioration a été effectuée pour les pics des bosses extérieures.

**Corollaire 5.6.3.** ***Paramètres fixes***
*Si nous prenons la valeur $S^{n,d}$ telle que :*

1. *$S^{n,d}$ tend vers 0 lorsque $n$ et $d$ tendent vers l'infini,*
2. *condition sur $n$ et $d$ :*

$$\lim_{n,d} S^{n,d}\sqrt{\frac{d}{\log n}} > \sqrt{2},$$

*alors l'estimateur du nombre de classes en ligne renvoyé par l'algorithme LG est consistant.*

La démonstration est une adaptation de celle de l'équation (5.1) (mise en annexe 5.F.2).

*Remarque* 5.6.4.
Les deux conditions imposent que $\frac{d}{\log n}$ tende vers l'infini.
Une valeur $S^{n,d}$ possible est :

$$\left(\frac{\log n}{d}\right)^{1/4}.$$

Comme la fonction tend vers 0 lorsque $n$ et $d$ tendent vers l'infini, nous avons la condition 1 et tout en vérifiant la condition 2.

**Corollaire 5.6.5.** ***Paramètres évolutifs***
*Si nous supposons que les paramètres varient tels que :*

*[C4] : condition sur $\delta_\tau^{n,d}$ :*

$$\lim_{n,d} \frac{\delta_\tau^{n,d}}{S^{n,d}} > 1,$$

*[C5] : condition sur $n$ et $d$ : sous la condition $C_{n,d}$ et :*

$$\lim_{n,d} S^{n,d}\sqrt{\frac{d}{\log n}} > \sqrt{2},$$

*[C6] : condition sur $\pi_{\min}^{n,d}$ : condition [C2.ligne] du théorème 5.3.7.*

*Alors l'estimateur du nombre de classes en ligne renvoyé par l'algorithme LG est consistant.*

La démonstration est identique aux précédentes. Avec l'hypothèse $[C4]$, nous sommes dans le cadre du théorème 5.6.1 lorsque $n$ et $d$ sont suffisamment grands et nous reprenons la démonstration du théorème 5.3.7 et du corollaire 5.6.3 pour faire tendre la borne vers 0.

Le dernier théorème nous permet de dire que tous les estimateurs sont consistants si nous possédons assez d'observations.

**Corollaire 5.6.6.** ***Consistance de l'algorithme LG***
*Si nous utilisons l'algorithme LG en prenant les sauts plus grands qu'un certain seuil $S^{n,d}$ (pouvant être différent pour le choix du nombre de lignes et de colonnes) tel que soient vérifiées les conditions suivantes :*

- *[C4bis] : condition sur $\delta_\tau^{n,d}$ :*

$$\lim_{n,d} \frac{\delta_\tau^{n,d}}{S^{n,d}} > 2,$$

- *la condition [C3] du théorème 5.4.3 et la condition [C5] du corollaire 5.6.5,*
- *les conditions symétriques pour les colonnes.*

*Alors les estimateurs du nombre de classes en ligne et en colonne, des partitions en ligne et en colonne et de $\boldsymbol{\theta}$ renvoyés par l'algorithme LG sont consistants.*

La démonstration est mise en annexe 5.F.3.
Toutefois, avant d'obtenir le bon nombre de classes, il faut un grand nombre d'observations (voir la section 5.7.3) ; ce qui rend l'algorithme inutilisable en pratique.

## 5.7 Applications numériques

Dans cette partie, nous présentons les résultats empiriques obtenus sur des simulations pour vérifier les résultats théoriques. Comme les deux classements se font de manière indépendante, nous ne regardons que l'erreur de classification sur les lignes que nous évaluons par les erreurs de classement de la section 1.3.2.5 par rapport à la vraie partition $\mathbf{z}^\star$ simulée. Pour évaluer la difficulté, nous regardons essentiellement le paramètre $\delta_\tau$, un plan de simulation étant jugé difficile si cette valeur est très petite.

La première sous-partie porte sur l'étude de la borne (5.2) en fonction du nombre de lignes $n$ et de colonnes $d$, la probabilité minimale $\pi_{\min}$ et la distance minimale entre deux bosses $\delta_\tau$. Pour cela, nous fixons deux paramètres et faisons varier les deux autres. Nous évaluons la qualité du classement avec l'erreur $e_{n,g}^{0-1}$ qui est associée à l'évènement $E_{\mathbf{z}}$ et avec l'erreur $e_{n,g}$ pour le cas où l'algorithme serait utilisé en pratique.

Dans la seconde partie, nous évaluons la qualité des améliorations proposées dans la section 5.7.2 en reprenant le premier plan de simulations de la section 5.7.1 et en regardant l'erreur $e_{n,g}$.

Enfin, nous regardons la qualité de la sélection du nombre de classes en reprenant les $\boldsymbol{\theta}$ proposés pour faire les simulations de la section 5.7.1 mais en prenant beaucoup plus de lignes et de colonnes. Pour cela, nous étudions le seuil optimal et celui proposé en remarque 5.6.4.

### 5.7.1 Comportement de la borne 5.2

Pour la première simulation, nous regardons la décroissance de la borne (5.2) lorsque $n$ et $d$ évoluent. Nous prenons les paramètres des lois conditionnelles de la forme :

$$\boldsymbol{\alpha} = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & 1-\varepsilon \end{pmatrix}$$

où nous faisons varier $\varepsilon$ entre 0 et 0.5 pour obtenir des matrices plus ou moins simples à classifier pour l'algorithme *LG*. Pour les poids du mélanges, nous prenons deux cas :

— équilibrée :

$$\boldsymbol{\pi} = \begin{pmatrix} 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \end{pmatrix} \quad \text{et} \quad \boldsymbol{\rho} = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix},$$

ce qui donne les vecteurs suivants :

$$\boldsymbol{\tau} = \begin{pmatrix} \varepsilon \\ 0.25 + 0.5\varepsilon \\ 0.5 \\ 0.75 - 0.5\varepsilon \\ 1 - \varepsilon \end{pmatrix} \quad \text{et} \quad \boldsymbol{\rho} = \begin{pmatrix} 0.8 - 0.6\varepsilon \\ 0.6 - 0.2\varepsilon \\ 0.4 + 0.2\varepsilon \\ 0.2 + 0.6\varepsilon \end{pmatrix},$$

et les paramètres :

$$\begin{aligned} \delta_\tau = 0.25 - 0.5\varepsilon \quad &\text{et} \quad \delta_\sigma = 0.2 - 0.4\varepsilon, \\ \pi_{\min} = 0.2 \quad &\text{et} \quad \rho_{\min} = 0.25. \end{aligned}$$

Pour ces proportions, les bosses sont espacées de la même distance et sont théoriquement de même volume. En particulier, le classement est impossible si $\varepsilon = 0.5$.

— Avec une progression arithmétique :

$$\boldsymbol{\pi} = \begin{pmatrix} 0.1 \\ 0.15 \\ 0.2 \\ 0.25 \\ 0.3 \end{pmatrix} \quad \text{et} \quad \boldsymbol{\rho} = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.3 \\ 0.4 \end{pmatrix},$$

ce qui donne les vecteurs suivants :

$$\boldsymbol{\tau} = \begin{pmatrix} \varepsilon \\ 0.1 + 0.8\varepsilon \\ 0.3 + 0.4\varepsilon \\ 0.6 - 0.2\varepsilon \\ 1 - \varepsilon \end{pmatrix} \quad \text{et} \quad \boldsymbol{\rho} = \begin{pmatrix} 0.9 - 0.8\varepsilon \\ 0.75 - 0.5\varepsilon \\ 0.55 - 0.1\varepsilon \\ 0.3 + 0.4\varepsilon \end{pmatrix},$$

et les paramètres :

$$\begin{aligned} \delta_\tau = 0.1 - 0.2\varepsilon \quad &\text{et} \quad \delta_\sigma = 0.15 - 0.3\varepsilon, \\ \pi_{\min} = 0.1 \quad &\text{et} \quad \rho_{\min} = 0.1. \end{aligned}$$

Pour ces proportions, les bosses sont espacées de distances multiples des distance minimales et sont théoriquement de volumes proportionnels. En particulier, le classement est impossible si $\varepsilon = 0.5$.

Pour le plan d'expérience, nous prenons $\varepsilon \in \{0.05, 0.25, 0.35, 0.45\}$ et nous faisons varier $n$ et $d$ de 20 en 20 en partant de 20 et en allant jusqu'à 800. Pour chaque combinaison, nous simulons 50 matrices sur lesquelles nous appliquons l'algorithme $LG$ et nous calculons l'erreur moyenne de classification en ligne. L'évolution de l'erreur moyenne en fonction du nombre de lignes et colonnes pour chaque combinaison est représentée sur la figure 5.6 pour l'erreur $e_{n,g}$ et la figure 5.7 pour l'erreur $e^{0-1}_{n,g}$ en annexe 5.G.

Nous voyons que les différences de classification dépendent surtout de la variable $\varepsilon$. Comme dans le cas des proportions inégales, les deux groupes les plus proches sont également les plus petits, l'erreur faite reste très faible car les grandes classes sont correctement estimées.

Nous voyons que, pour les cas où $\varepsilon$ est égal à 0.45, l'erreur $e_{n,g}$ augmente d'abord avec le nombre de lignes puis redescend. Ceci est peut-être dû à la remarque précédente ou à la borne maximale de l'erreur car 20 lignes est en dessous de $g^2 = 25$ trouvé dans le tableau 1.5.

En agrandissant le cas des proportions égales avec $\varepsilon = 0.25$ (voir la figure 5.4), nous pouvons voir cette double influence :

$d$ : plus le nombre de colonnes augmente, plus l'erreur diminue. La forme exponentielle se voit surtout pour un grand nombre de colonnes.

$n$ : Plus le nombre de lignes augmente, plus nous risquons de commettre une erreur de classement. Le facteur linéaire n'est toutefois pas forcément visible.

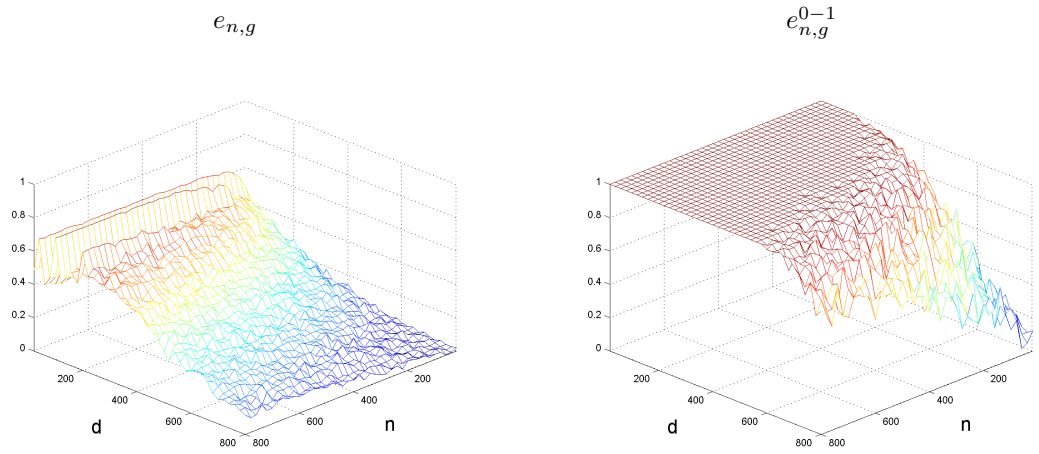



FIGURE 5.4 – Influence du nombre de lignes et de colonnes sur les deux erreurs de classification (à gauche $e_{n,g}$ et à droite $e^{0-1}_{n,g}$) pour les lignes dans le cas de proportions égales et pour $\varepsilon = 0.25$.

Une autre influence intéressante est celle des valeurs $\delta_\tau$ et $\pi_{\min}$ lorsque le nombre de lignes et de colonnes sont fixes sur le classement en ligne. Pour cela, nous choisissons les paramètres de la façon suivante :

$$\boldsymbol{\pi} = \begin{pmatrix} \pi_{\min} \\ 0.25 - 0.25\pi_{\min} \\ 0.25 - 0.25\pi_{\min} \\ 0.25 - 0.25\pi_{\min} \\ 0.25 - 0.25\pi_{\min} \end{pmatrix}, \quad \boldsymbol{\rho} = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix},$$

$$\text{et } \boldsymbol{\alpha} = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & 1-\varepsilon \end{pmatrix}$$

Nous conservons ainsi le fait que $\delta_\tau = 0.25 - 0.5\varepsilon$ puisque celui ci ne dépend que de la classification sur les colonnes. Nous fixons le nombre de lignes et de colonnes à 500 ou 1000 et faisons varier :

— $\pi_{\min}$ de 0.005 en 0.005 en partant de 0.005 jusqu'à 0.2 (cas d'égalité des variables),

— $\delta_\tau$ de 0.01 en 0.01 en partant de 0.01 jusqu'à 0.4.

Comme précédemment, nous simulons pour chaque cas 50 matrices et faisons la moyenne des erreurs.

Nous voyons sur les figures 5.8 et 5.9 les résultats en fonction du nombre de lignes et de colonnes pour les erreurs $e_{n,g}$ et $e^{0-1}_{n,g}$. Nous retrouvons les constations précédentes où plus il y a de colonnes et mieux l'algorithme classifie les lignes. Nous voyons également que plus les bosses sont éloignées et plus la classification est correcte. En revanche, nous pouvons être surpris par l'influence de $\pi_{\min}$ sur les résultats car plus il augmente, et plus l'erreur moyenne a tendance à être grande (en particulier lorsque le paramètre $\delta_\tau$ est proche de 0.2 et que le nombre de colonnes est très faible). Comme toutes les classes dépendent de la probabilité minimale, plus la probabilité minimale augmente et plus la première bosse a de chances de s'étaler (à $\delta_\tau$ fixe) et donc qu'une ligne soit classée dans la bosse d'à côté.

Au vu du nombre de paramètres qui entrent en jeu dans la borne, il existe un grand nombre de plans d'expériences possibles (ne serait-ce qu'en regardant différentes configurations pour $\boldsymbol{\theta}$ déjà). L'intérêt de cet algorithme étant essentiellement théorique, nous nous contentons de ces deux types d'expérience.

### 5.7.2 Étude de l'amélioration

Dans cette partie, nous regardons la qualité des améliorations proposées dans la section 5.5 en commençant par celle de l'initialisation. Pour cela, nous reprenons les matrices simulées dans la section 5.7.1 et regardons la moyenne des normes infinies entre l'estimation des variables $\boldsymbol{\tau}$ renvoyée par l'algorithme et les vraies valeurs. Nous voyons sur la figure 5.10 que les estimations sont très bonnes dès que les pics commencent à apparaître c'est-à-dire lorsqu'il y a suffisamment de lignes pour que chaque groupe apparaisse et suffisamment de colonnes pour que les pics se démarquent les uns des autres.

Dans un second temps, nous regardons la qualité des estimations des classifications renvoyées par les algorithmes $EM$ en supposant que les lois de mélange sont des lois binomiales ou gaussiennes. Les résultats sont représentés sur les figures 5.11 pour le cas des proportions équilibrées et 5.12 pour le cas des proportions déséquilibrées en prenant l'erreur $e_{n,g}$ (nous ne prenons que ce choix car à partir du moment où l'algorithme $LG$ classe correctement les lignes, les améliorations aussi). Nous voyons que les améliorations avec l'algorithme $EM$ diminuent fortement l'erreur de classification et plus particulièrement pour l'algorithme présupposant que les lois sont des binomiales : nous observons une décroissance lisse. A l'opposé, la décroissance pour l'algorithme $EM$ avec des lois gaussiennes se fait en deux étapes (voir la figure 5.5) : pour des valeurs de $d$ faibles, la décroissance est lente puis nous avons une chute à partir du moment où le nombre d'observations est suffisamment important pour que l'approximation asymptotique soit correcte. Cette chute n'apparaît pas pour $\varepsilon = 0.45$ car le nombre d'observations est trop faible par rapport à la difficulté des matrices.

### 5.7.3 Étude de la sélection de modèles

Pour le dernier plan d'expérience, nous utilisons la même configuration pour les paramètres $\boldsymbol{\theta}$ servant à simuler les matrices pour des valeurs de $\varepsilon$ appartenant à l'ensemble $\{0.05, 0.1, 0.15, 0.2\}$ et pour des lignes et colonnes allant de 250 en 250 en partant de 250 et en allant jusqu'à 10 000. Pour chaque combinaison, nous simulons 50 matrices et regardons le nombre de classes en ligne sélectionnés par le seuil valant $\delta_\tau/2$ (figure 5.13) et celui valant $\left(\frac{\log n}{d}\right)^{1/4}$ (figures 5.14 et 5.14).

L'algorithme sélectionne plus vite le bon nombre de classes lorsque le seuil est optimal (figure 5.13) ; alors qu'il faut un certain nombre de colonnes par rapport au nombre de lignes pour que le seuil de la remarque 5.6.4 soit suffisamment petit. Dans les deux cas, plus le nombre de colonnes est élevé et plus la probabilité d'avoir le bon nombre de classes est élevé alors que le nombre de lignes diminue cette

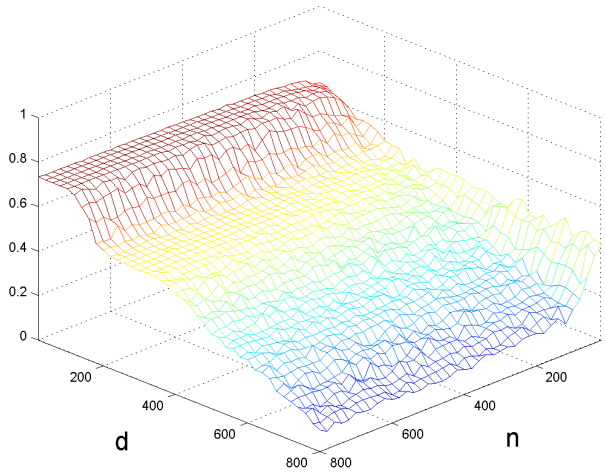



FIGURE 5.5 – Influence du nombre de lignes et de colonnes sur l'erreur de classification pour les lignes pour la version améliorée en ajoutant une estimation par l'algorithme $EM$ supposant que les lois du mélange sont des gaussiennes dans le cas de proportions égales et pour $\varepsilon = 0.35$.

probabilité. De plus, l'influence de $\delta_\tau$ est plus grande que dans le cas des estimations des classements.

Toutefois, il est à remarquer que le nombre de lignes et de colonnes est beaucoup plus grand que pour les plans précédents. Si nous utilisons l'algorithme $LG$ pour estimer tous les paramètres, les estimations dépendront surtout de celles du nombre de classes.

## 5.8 Conclusion

L'intérêt de l'algorithme $LG$ est avant tout théorique. Le fait d'utiliser cette simplification du modèle permet d'avoir une vision du problème de la double asymptotique et du rapport des vitesses de croissance possible pour le nombre de lignes et de colonnes. Il permet également de comprendre les paramètres qui entrent en jeu dans la difficulté de faire la classification d'une matrice et il aide à mieux comprendre les deux conditions $C_1$ et $C_2$ du critère d'identifiabilité de Keribin et al. (théorème 1.3.7).

Toutefois, il peut quand même être utilisé comme initialisation d'algorithmes plus performants. En effet, nous voyons en section 5.7.2 sur des simulations que combiné à un algorithme type $EM$ avec des lois binomiales, il renvoie, en peu de temps, une bonne estimation des classifications en ligne et en colonne lorsque les données ne sont pas trop compliquées ; par exemple si le nombre de classes est très petit.

## 5.A Indépendance des cases d'une ligne conditionnellement à l'appartenance à une classe de cette dernière

Dans cette partie, nous montrons l'indépendance des cases d'une même ligne sachant que cette dernière appartient à la classe $k$.

Nous savons que les cases $X_{ij}$ sont indépendantes conditionnellement à l'appartenance des lignes et des colonnes. Soient deux cases $X_{ij}$ et $X_{ij'}$ d'une même ligne $i$ appartenant à une classe $k$, alors nous avons :

$$\begin{aligned}
\mathbb{P}(X_{ij}, X_{ij'}|z_{ik}=1) &= \sum_{\ell,\ell'} \mathbb{P}(X_{ij}, X_{ij'}|z_{ik}=1, w_{j\ell}=1, w_{j'\ell'}=1)\mathbb{P}(w_{j\ell}=1, w_{j'\ell'}=1) \\
&= \sum_{\ell,\ell'} \mathbb{P}(X_{ij}|z_{ik}=1, w_{j\ell}=1, w_{j'\ell'}=1)\mathbb{P}(X_{ij'}|z_{ik}=1, w_{j\ell}=1, w_{j'\ell'}=1) \\
&\quad \times \mathbb{P}(w_{j\ell}=1)\mathbb{P}(w_{j'\ell'}=1) \\
&= \sum_{\ell,\ell'} \mathbb{P}(X_{ij}|z_{ik}=1, w_{j\ell}=1)\mathbb{P}(X_{ij'}|z_{ik}=1, w_{j'\ell'}=1) \\
&\quad \times \mathbb{P}(w_{j\ell}=1)\mathbb{P}(w_{j'\ell'}=1) \\
&= \left(\sum_{\ell} \mathbb{P}(X_{ij}|z_{ik}=1, w_{j\ell}=1)\mathbb{P}(w_{j\ell}=1)\right) \\
&\quad \times \left(\sum_{\ell'} \mathbb{P}(X_{ij'}|z_{ik}=1, w_{j'\ell'}=1)\mathbb{P}(w_{j'\ell'}=1)\right) \\
&= \mathbb{P}(X_{ij}|z_{ik}=1)\mathbb{P}(X_{ij'}|z_{ik}=1).
\end{aligned}$$

D'où l'indépendance des variables $X_{ij}$ et $X_{ij'}$ d'une même ligne $i$ conditionnellement au fait que celle-ci appartienne à une classe $k$.

## 5.B Démonstration de la consistance de $\widehat{z}^{LG}$

Cette démonstration se fait en trois étapes, nous regardons d'abord un évènement particulier appartenant à $\overline{E_z}$ dont nous majorons la probabilité de son complémentaire dans un second temps et enfin, nous regardons les conditions pour obtenir la consistance de notre estimateur.

### 5.B.1 Évènement idéal

Pour que l'algorithme $LG$ renvoie le bon résultat, il faut les deux conditions suivantes :

— toutes les classes doivent avoir au moins un élément, nous notons donc :

$$A_g = \bigcap_{k=1}^{g} \{z_{+k} \neq 0\}.$$

— Les probabilités $\tau_k$ doivent être suffisamment distinctes. Pour cela, nous introduisons la distance maximale entre chaque variable $\overline{X_{i\bullet}}$ et son espérance :

$$d_n = \max_{k\in\{1\dots,g\}} \sup_{i|z^\star_{ik}=1} \left|\overline{X_{i\bullet}} - \tau_k\right|.$$

En prenant $\varepsilon > 0$, nous regardons alors l'évènement vérifiant que cette distance n'est pas trop grande devant l'éloignement de deux groupes :

$$\left\{ d_n \leq \frac{\delta_\tau}{4+\varepsilon} \right\}.$$

La suite de cette sous-section consiste à montrer que $B_\varepsilon = A_g \bigcap \left\{ d_n \leq \frac{\delta_\tau}{4+\varepsilon} \right\}$ est inclus dans $\overline{E_z}$ pour tout $\varepsilon > 0$ ou autrement dit que, sous l'évènement $B_\varepsilon$, l'algorithme $LG$ renvoie les bonnes partitions.
Pour cela, nous fixons $\varepsilon > 0$ et nous supposons être sous l'évènement $B_\varepsilon$. Nous voyons déjà que toutes les classes possèdent au moins un représentant et nous montrons maintenant que toutes les variables $\overline{X_{i\bullet}}$ d'une même classe sont plus proches les unes des autres que des variables de classes différentes :

— si les lignes $i$ et $i'$ appartiennent à la même classe $k$, nous avons :

$$\begin{aligned} \left|\overline{X_{i\bullet}} - \overline{X_{i'\bullet}}\right| &\leq \left|\overline{X_{i\bullet}} - \tau_k\right| + \left|\overline{X_{i'\bullet}} - \tau_k\right| \\ &\leq 2d_n \\ &\leq \frac{2\delta_\tau}{4+\varepsilon} \end{aligned}$$

par définition de $d_n$.

— Au contraire, si la ligne $i$ appartient à la classe $k$ et la ligne $i'$ appartient à une classe différente $k'$, nous avons :

$$\begin{aligned} \left|\overline{X_{i\bullet}} - \overline{X_{i'\bullet}}\right| &\geq \left|\overline{X_{i\bullet}} - \tau_{k'}\right| - \left|\overline{X_{i'\bullet}} - \tau_{k'}\right| \\ &\geq \left|\overline{X_{i\bullet}} - \tau_{k'}\right| - d_n \\ &\geq |\tau_{k'} - \tau_k| - \left|\overline{X_{i\bullet}} - \tau_k\right| - \frac{\delta_\tau}{4+\varepsilon} \\ &\geq \delta_\tau - \frac{\delta_\tau}{4+\varepsilon} - \frac{\delta_\tau}{4+\varepsilon} \\ &\geq \frac{\delta_\tau(4+\varepsilon-2)}{4+\varepsilon} \\ &> \frac{2\delta_\tau}{4+\varepsilon}. \end{aligned}$$

Ainsi, si nous sommes sous l'évènement $B_\varepsilon$, les lignes $i$ et $i'$ sont dans la même classe si et seulement si $\left|\overline{X_{i\bullet}} - \overline{X_{i'\bullet}}\right| \leq \frac{2\delta_\tau}{4+\varepsilon}$ et nous obtenons alors exactement $g-1$ éléments de $(G_i)_{1\leq i\leq n-1}$ qui seront strictement plus grands que $\frac{2\delta_\tau}{4+\varepsilon}$.
Sous ces conditions, l'algorithme va correctement classer les lignes ce qui revient à dire que pour tout $\varepsilon > 0$, $B_\varepsilon$ est inclus dans $\overline{E_z}$.

### 5.B.2 Borne sur les probabilités

Nous majorons ensuite la probabilité de l'évènement complémentaire.

Nous avons pour tout $\varepsilon > 0$ :

$$\begin{aligned} \mathbb{P}(E_z) &\leq \mathbb{P}\left(\overline{B_\varepsilon}\right) \\ &\leq \mathbb{P}\left(\overline{A_g} \bigcup \overline{\left\{ d_n \leq \frac{\delta_\tau}{4+\varepsilon} \right\}}\right) \\ &\leq \mathbb{P}\left(\overline{A_g}\right) + \mathbb{P}\left(\left\{ d_n > \frac{\delta_\tau}{4+\varepsilon} \right\}\right). \end{aligned}$$

Il ne reste plus qu'à majorer ces deux probabilités.
La deuxième utilise l'inégalité de Hoeffding appliquée aux variables $X_{i+}$ qui, conditionnellement à l'appartenance de la $i$ ème ligne à la classe $k$, suivent des lois binomiales de paramètres $d$ et $\tau_k$ et nous avons, en introduisant la variable $z$, pour tout $t > 0$ :

$$\begin{aligned}
\mathbb{P}\left(\left|\overline{X_{i\bullet}} - \tau_k\right| > t \middle| z_{ik} = 1\right) &= \mathbb{P}\left(|X_{i+} - d\tau_k| > dt \middle| z_{ik} = 1\right) \\
&\leq 2e^{-\frac{2(dt)^2}{\sum_{i=1}^{d}(1-0)^2}} \\
&\leq 2e^{-2dt^2}.
\end{aligned}$$

Ainsi, nous obtenons :

$$\begin{aligned}
\mathbb{P}(d_n > t) &= \mathbb{E}\left[d_n > t\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[d_n > t \middle| z\right]\right] \\
&= \mathbb{E}\left[\mathbb{P}\left(d_n > t \middle| z\right)\right] \\
&= \mathbb{E}\left[\mathbb{P}\left(\bigcup_{k=1}^{g}\bigcup_{i|z_{ik}=1}\left\{\left|\overline{X_{i\bullet}} - \tau_k\right| > dt\right\} \middle| z\right)\right] \\
&\leq \mathbb{E}\left[\sum_{k=1}^{g}\sum_{i|z_{ik}=1}\mathbb{P}\left(\left\{\left|\overline{X_{i\bullet}} - \tau_k\right| > dt\right\} \middle| z\right)\right] \\
&\leq \mathbb{E}\left[\sum_{k=1}^{g}\sum_{i|z_{ik}=1} 2e^{-2dt^2}\right] \\
&\leq 2ne^{-2dt^2}.
\end{aligned}$$

Au final, nous avons :

$$\mathbb{P}\left(\left\{d_n > \frac{\delta_\tau}{4+\varepsilon}\right\}\right) \leq 2ne^{-2d\left(\frac{\delta_\tau}{4+\varepsilon}\right)^2}.$$

*Remarque* 5.B.1. Nous pouvons déjà remarquer la double influence de $n$ et de $d$. Plus nous avons de lignes, plus il est difficile de toutes les classer correctement alors que, plus nous avons de colonnes et plus nous avons d'informations pour faire un bon classement.

Pour la deuxième majoration, l'évènement $\overline{A_g}$ signifie qu'il existe au moins une classe ne contenant aucune ligne ou autrement dit qu'il existe $k$ avec $z_{+k} = 0$. De plus, nous savons que $Z_{+k}$ suit une loi binomiale de paramètres $n$ et $\pi_k$ donc nous avons :

$$\begin{aligned}
\mathbb{P}\left(\overline{A_g}\right) &= \mathbb{P}\left(\bigcup_{k=1}^{g}\{z_{+k} = 0\}\right) \\
&\leq \sum_{k=1}^{g}\mathbb{P}\left(\{z_{+k} = 0\}\right) \\
&\leq \sum_{k=1}^{g}\left(1 - \pi_k\right)^n \\
&\leq g\left(1 - \pi_{\min}\right)^n.
\end{aligned}$$

*Remarque* 5.B.2. Plus le nombre de classes est grand et plus la probabilité minimale est petite, plus le nombre d'observations devra être grand pour qu'il n'y ait aucune classe vide.

Nous obtenons finalement pour tout $\varepsilon > 0$ :

$$\mathbb{P}\left(E_z\right) \leq 2ne^{-2d\left(\frac{\delta_\tau}{4+\varepsilon}\right)^2} + g\left(1-\pi_{\min}\right)^n.$$

Nous faisons ensuite tendre $\varepsilon$ vers 0 et nous obtenons la formule (5.1) :

$$\mathbb{P}(E_z) \leq 2ne^{-\frac{1}{8}d\delta_\tau^2} + g(1-\pi_{\min})^n.$$

### 5.B.3 Consistance

Pour montrer la consistance, il suffit de montrer que la borne 5.2 obtenue précédemment tend vers 0.

Supposons que nous vérifions l'inégalité :

$$\underline{\lim}_{n,d} \frac{d}{\log n} > \frac{8}{\delta_\tau^2}.$$

Dans ce cas, il existe une constante $C > \frac{8}{\delta_\tau^2}$ telle que $\frac{d}{\log n} \geq C$ pour $n$ et $d$ suffisamment grands et nous avons :

$$\begin{aligned}
n\exp\left(-\frac{1}{8}d\delta_\tau^2\right) &= \exp\left(\log n - \frac{1}{8}d\delta_\tau^2\right) \\
&= \exp\left[-\frac{\delta_\tau^2}{8}\log n\left(\frac{d}{\log n} - \frac{8}{\delta_\tau^2}\right)\right] \\
&\leq \exp\left[-\frac{\delta_\tau^2}{8}\log n \underbrace{\left(C - \frac{8}{\delta_\tau^2}\right)}_{>0}\right] \\
&\underset{n,d\to+\infty}{\longrightarrow} 0.
\end{aligned}$$

D'où le résultat.

## 5.C Calcul des conditions pour la section 5.3.2

Dans cette section, nous mettons les démonstrations des conditions suffisantes pour obtenir la consistance de l'estimateur $\widehat{z}^{LG}$ lorsque les paramètres peuvent varier.

### 5.C.1 Évolution de $\delta_\tau^{n,d}$

La démonstration est sensiblement la même que pour la section 5.B.3.
Si nous supposons que $\underline{\lim}_{n,d} \delta_\tau^{n,d}\sqrt{\frac{d}{\log n}} > 2\sqrt{2}$ alors il existe $C > 2\sqrt{2}$ telle que $\delta_\tau^{n,d}\sqrt{\frac{d}{\log n}} \geq C$ pour $n$ et

$d$ suffisamment grands, c'est-à-dire que $\left(\delta_\tau^{n,d}\right)^2 \frac{d}{\log n} - 8 \geq C^2 - 8 > 0$ et nous avons :

$$\begin{aligned} n \exp\left(-\frac{1}{8} d \left(\delta_\tau^{n,d}\right)^2\right) &= \exp\left[-\frac{1}{8}\log n \left(\left(\delta_\tau^{n,d}\right)^2 \frac{d}{\log n} - 8\right)\right] \\ &\leq \exp\left[-\frac{1}{8}\log n \left(C^2 - 8\right)\right] \\ &\underset{n,d\to+\infty}{\longrightarrow} 0 \end{aligned}$$

### 5.C.2 Evolution de $\pi_{\min}^{n,d}$

La condition sur $\pi_{\min}^{n,d}$ vient de la deuxième partie de la formule (5.1).
Si $g^{n,d}$ tend vers l'infini alors, pour les mêmes raisons que précédemment, il existe $C' > 1$ tel que pour $n$ et $d$ assez grands, nous ayons

$$-\frac{n \log\left(1 - \pi_{\min}^{n,d}\right)}{\log g^{n,d}} > C'$$

et nous obtenons :

$$\begin{aligned} g^{n,d}\left(1 - \pi_{\min}^{n,d}\right)^n &= \exp\left[\log g^{n,d} + n \log\left(1 - \pi_{\min}^{n,d}\right)\right] \\ &= \exp\left[-\log g^{n,d}\left(-\frac{n \log\left(1 - \pi_{\min}^{n,d}\right)}{\log g^{n,d}} - 1\right)\right] \\ &\leq \exp\left[-\log g^{n,d}\left(C' - 1\right)\right] \\ &\underset{n,d\to+\infty}{\longrightarrow} 0 \end{aligned}$$

Sinon, nous avons :

$$\left(1 - \pi_{\min}^{n,d}\right)^n = \exp\left[n \log\left(1 - \pi_{\min}^{n,d}\right)\right]$$

Or, nous savons que $\log(1-t) \underset{0}{\sim} -t$ alors :

$$n \log\left(1 - \pi_{\min}^{n,d}\right) \underset{+\infty}{\sim} -n\pi_{\min}^{n,d}.$$

Donc, si $\pi_{\min}^{n,d} n \underset{n\to+\infty}{\longrightarrow} +\infty$, $\left(1 - \pi_{\min}^{n,d}\right)^n \to 0$.
Pour la condition plus faible de la remarque 5.3.8, nous reprenons la borne de la remarque 5.3.6 :

$$g^{n,d} \leq \frac{1}{\pi_{\min}^{n,d}} \quad \Leftrightarrow \quad g^{n,d}\left(1 - \pi_{\min}^{n,d}\right)^n \leq \frac{1}{\pi_{\min}^{n,d}}\left(1 - \pi_{\min}^{n,d}\right)^n$$

et la suite reste la même qu'au début de la section.
Pour la borne sur $g^{n,d}$, nous prenons la deuxième borne :

$$g^{n,d} \leq \frac{1}{\delta_\tau^{n,d}} + 1.$$

Or, nous avons vu dans la section 5.C.1 qu'il existe $C > 2\sqrt{2}$ telle que $\delta_\tau^{n,d}\sqrt{\frac{d}{\log n}} \geq C$ pour $n$ et $d$ suffisamment grands ce qui donne :

$$g^{n,d} \leq \frac{1}{C}\sqrt{\frac{d}{\log n}} + 1$$

et nous obtenons le résultat.

## 5.D Démonstration de la consistance des paramètres empiriques

Dans cette partie, nous démontrons la consistance des paramètres empiriques de $\widetilde{\theta}$ : d'abord lorsque les paramètres sont fixes puis en les faisant tendre vers des conditions limites.

### 5.D.1 Paramètres fixes

Dans cette partie, nous démontrons la borne obtenue dans la section 5.4. Pour cela, nous cherchons à majorer pour tout $t > 0$, l'évènement $\mathcal{C}_t^{z,w} = \left\{ \left\| \widehat{\theta}^{LG} - \boldsymbol{\theta}^\star \right\|_{+\infty} > t \right\}$. Nous allons diviser en deux cas, celui où les partitions sont correctes et son complémentaire :

$$\begin{aligned}
\mathbb{P}(\mathcal{C}_t^{z,w}) &= \mathbb{P}(\mathcal{C}_t^{z,w} \cap \overline{E}_{z,w}) + \mathbb{P}(\mathcal{C}_t^{z,w} \cap E_{z,w}) \\
&\leq \mathbb{P}(\mathcal{C}_t^{z,w} \cap \overline{E}_{z,w}) + \mathbb{P}(E_{z,w}) \\
&\leq \mathbb{P}\left( \left\| \widetilde{\boldsymbol{\theta}} - \boldsymbol{\theta}^\star \right\|_{+\infty} > t \middle| \overline{E}_{z,w} \right) \mathbb{P}(\overline{E}_{z,w}) + \mathbb{P}(E_{z,w}) \\
\text{avec} \qquad & \widetilde{\pi}_k = \frac{z^\star_{+k}}{n},\ \widetilde{\rho}_\ell = \frac{w^\star_{+\ell}}{d} \text{ et } \widetilde{\alpha}_{k\ell} = \frac{\sum_{i,j} z^\star_{ik} x_{ij} w^\star_{j\ell}}{z^\star_{+k} w^\star_{+\ell}}
\end{aligned}$$

où $\mathbf{z}^\star$ et $\mathbf{w}^\star$ sont les vraies matrices de répartition.
Nous possédons déjà une borne pour la deuxième probabilité, il reste donc à calculer la probabilité que $\widetilde{\boldsymbol{\theta}}$ soit différent de $\boldsymbol{\theta}^\star$ puisque $\mathbb{P}(\overline{E}_{z,w}) \leq 1$.

Nous supposons d'abord que les classes trouvées par l'algorithme sont correctes ou, autrement dit, que $\widehat{\theta}^{LG} = \widetilde{\boldsymbol{\theta}}$. Nous regardons donc l'éloignement des approximations des paramètres en prenant les estimateurs empiriques. En particulier, si tous les paramètres sont éloignés, la norme sup l'est également :

$$\begin{aligned}
\mathbb{P}\left( \left\| \widetilde{\boldsymbol{\theta}} - \boldsymbol{\theta}^\star \right\|_{+\infty} > t \right) &\leq \mathbb{P}\left( \|\widetilde{\boldsymbol{\pi}} - \boldsymbol{\pi}^\star\|_{+\infty} > t \right) + \mathbb{P}\left( \|\widetilde{\boldsymbol{\rho}} - \boldsymbol{\rho}^\star\|_{+\infty} > t \right) + \mathbb{P}\left( \|\widetilde{\boldsymbol{\alpha}} - \boldsymbol{\alpha}^\star\|_{+\infty} > t \right) \\
&\leq \sum_{k=1}^{g} \mathbb{P}\left( |\widetilde{\pi}_k - \pi_k^\star| > t \right) + \sum_{\ell=1}^{m} \mathbb{P}\left( |\widetilde{\rho}_\ell - \rho_\ell^\star| > t \right) + \sum_{k,\ell} \mathbb{P}\left( |\widetilde{\alpha}_{k\ell} - \alpha_{k\ell}^\star| > t \right).
\end{aligned}$$

Or, d'après l'inégalité de Hoeffding, pour tout $k \in \{1, \ldots, g\}$ et $t > 0$ :

$$\mathbb{P}\left( |\widetilde{\pi}_k - \pi_k^\star| > t \right) \leq 2e^{-2nt^2}.$$

Un résultat similaire est obtenu pour les paramètres de $\rho$.
Pour les paramètres de $\widetilde{\boldsymbol{\alpha}}$, nous commençons par inclure la taille des blocs :

$$\widetilde{\alpha}_{k\ell} = \frac{1}{z^\star_{+k} w^\star_{+\ell}} \sum_{(i,j)|z^\star_{ik} w^\star_{j\ell}=1} X_{ij}$$

et, si nous connaissons les matrices $\mathbf{z}^\star$ et $\mathbf{w}^\star$, nous pouvons utiliser l'inégalité de Hoeffding :

$$\mathbb{P}\left(|\widetilde{\alpha}_{k\ell} - \alpha^\star_{k\ell}| > t\right) = \mathbb{E}\left[\mathbb{P}\left(|\widetilde{\alpha}_{k\ell} - \alpha^\star_{k\ell}| > t\right) | \mathbf{z}^\star, \mathbf{w}^\star\right] \leq \mathbb{E}\left[2e^{-2z^\star_{+k}w^\star_{+\ell}t^2}\right].$$

Nous regardons maintenant la divergence entre la taille des blocs $(k,\ell)$ obtenus avec les matrices $(\mathbf{z}^\star, \mathbf{w}^\star)$ soit $z^\star_{+k}w^\star_{+\ell}$ et la taille théorique $\pi^\star_k\rho^\star_\ell nd$. Nous introduisons une suite $r_{n,d}$ tendant vers l'infini dont nous définissons les propriétés un peu plus loin et nous introduisons l'évènement $\{|z^\star_{+k}w^\star_{+\ell} - \pi^\star_k\rho^\star_\ell nd| > r_{n,d}\}$ :

$$\begin{aligned}
\mathbb{E}\left[e^{-2z^\star_{+k}w^\star_{+\ell}t^2}\right] &= \mathbb{E}\Big[e^{-2z^\star_{+k}w^\star_{+\ell}t^2}\mathbb{1}_{\left\{|z^\star_{+k}w^\star_{+\ell} - \pi^\star_k\rho^\star_\ell nd| \leq r_{n,d}\right\}} + \underbrace{e^{-2z^\star_{+k}w^\star_{+\ell}t^2}}_{\leq 1}\mathbb{1}_{\left\{|z^\star_{+k}w^\star_{+\ell} - \pi^\star_k\rho^\star_\ell nd| > r_{n,d}\right\}}\Big] \\
&\leq \mathbb{E}\left[e^{-2z^\star_{+k}w^\star_{+\ell}t^2}\mathbb{1}_{\left\{-r_{n,d} \leq z^\star_{+k}w^\star_{+\ell} - \pi^\star_k\rho^\star_\ell nd \leq r_{n,d}\right\}}\right] + \mathbb{P}\left(|z^\star_{+k}w^\star_{+\ell} - \pi^\star_k\rho^\star_\ell nd| > r_{n,d}\right) \\
&\leq \mathbb{E}\left[e^{-2t^2(\pi^\star_k\rho^\star_\ell nd - r_{n,d})}\right] + \mathbb{P}\left(\left|\frac{z^\star_{+k}w^\star_{+\ell}}{nd} - \pi^\star_k\rho^\star_\ell\right| > \frac{r_{n,d}}{nd}\right) \\
&\leq e^{-2r_{n,d}t^2\left(\frac{\pi^\star_k\rho^\star_\ell nd}{r_{n,d}} - 1\right)} + \mathbb{P}\left(\left|\frac{z^\star_{+k}w^\star_{+\ell}}{nd} - \pi^\star_k\rho^\star_\ell\right| > \frac{r_{n,d}}{nd}\right) \\
&\leq e^{-2r_{n,d}t^2\left(\frac{\pi_{\min}\rho_{\min}nd}{r_{n,d}} - 1\right)} + \mathbb{P}\left(\left|\frac{z^\star_{+k}w^\star_{+\ell}}{nd} - \pi^\star_k\rho^\star_\ell\right| > \frac{r_{n,d}}{nd}\right)
\end{aligned}$$

Pour la deuxième probabilité, nous avons :

$$\begin{aligned}
\mathbb{P}\left(\left|\frac{z^\star_{+k}w^\star_{+\ell}}{nd} - \pi^\star_k\rho^\star_\ell\right| > \frac{r_{n,d}}{nd}\right) &= \mathbb{P}\left(\left|\left(\frac{z^\star_{+k}}{n} - \pi^\star_k\right)\frac{w^\star_{+\ell}}{d} + \left(\frac{w^\star_{+\ell}}{d} - \rho^\star_\ell\right)\pi^\star_k\right| > \frac{r_{n,d}}{nd}\right) \\
&\leq \mathbb{P}\left(\left|\frac{z^\star_{+k}}{n} - \pi^\star_k\right|\frac{w^\star_{+\ell}}{d} > \frac{r_{n,d}}{2nd}\right) + \mathbb{P}\left(\left|\frac{w^\star_{+\ell}}{d} - \rho^\star_\ell\right|\pi^\star_k > \frac{r_{n,d}}{2nd}\right) \\
&\leq \mathbb{P}\left(\left|\frac{z^\star_{+k}}{n} - \pi^\star_k\right| > \frac{r_{n,d}}{2nd}\right) + \mathbb{P}\left(\left|\frac{w^\star_{+\ell}}{d} - \rho^\star_\ell\right| > \frac{r_{n,d}}{2nd}\right) \quad \text{car} \quad \frac{w^\star_{+\ell}}{d} < 1 \\
&\leq 2\exp\left[-2n\left(\frac{r_{n,d}}{2nd}\right)^2\right] + 2\exp\left[-2d\left(\frac{r_{n,d}}{2nd}\right)^2\right] \\
&\leq 2\exp\left[-\frac{r^2_{n,d}}{2nd^2}\right] + 2\exp\left[-\frac{r^2_{n,d}}{2n^2d}\right]
\end{aligned}$$

Au final, nous avons :

$$\mathbb{P}\left(|\widetilde{\alpha}_{k\ell} - \alpha^\star_{k\ell}| > t\right) \leq 2e^{-2r_{n,d}t^2\left(\frac{\pi_{\min}\rho_{\min}nd}{r_{n,d}} - 1\right)} + 4e^{-\frac{r^2_{n,d}}{2nd^2}} + 4e^{-\frac{r^2_{n,d}}{2n^2d}}.$$

Il ne reste plus qu'à choisir $r_{n,d}$.

*Remarque* 5.D.1. Pour que la borne tende bien vers 0, il faut que $r_{n,d}$ vérifie les conditions suivantes :

$$\underline{\lim}_{n,d\to+\infty} \frac{\pi_{\min}\rho_{\min}nd}{r_{n,d}} > 1, \quad \frac{r^2_{n,d}}{n^2d} \to +\infty \text{ et } \frac{r^2_{n,d}}{nd^2} \to +\infty.$$

D'après ces conditions, $r_{n,d}$ ne peut pas aller plus vite que $nd$ pour que la première partie de la borne tende vers 0 ni moins vite que $\sqrt{nd}$ (pour les deux autres).
Pour que la borne ait la décroissance la plus rapide, il faut prendre $r_{n,d}$ de la forme $C\pi_{\min}\rho_{\min}nd$ avec $0 < C < 1$.
La constante $C$ optimale dépend à la fois de $n$, $d$, $\pi_{\min}$ et $\rho_{\min}$.

Pour l'autre partie, nous reprenons simplement l'équation 5.2 et nous obtenons la formule 5.3.

### 5.D.2 Conditions limites

Avant de montrer la consistance du début de la borne, nous montrons que la condition 3 sur les lignes et les colonnes impliquent la condition 2 du théorème 5.3.7. Pour cela, nous regardons le cas des lignes :

— si le paramètre $g^{n,d}$ tend vers l'infini alors, par l'hypothèse 3.*ligne*, nous savons qu'il existe une constante $C > 1$ telle que, pour $n$ et $d$ assez grands, nous ayons :

$$\frac{(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n}{\log\left(g^{n,d}m^{n,d}\right)} > C.$$

Pour la condition 2 sur les lignes du théorème 5.3.7, nous regardons la formule :

$$-\frac{n\log\left(1-\pi_{\min}^{n,d}\right)}{\log g^{n,d}}.$$

Or, comme $g^{n,d}$ tend vers l'infini, la probabilité $\pi_{\min}^{n,d}$ tend vers 0, la formule précédente équivaut à :

$$\frac{n\pi_{\min}^{n,d}}{\log g^{n,d}}$$

qui est positive. Ainsi, pour $n$ et $d$ assez grands, nous avons :

$$\frac{(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n}{\log\left(g^{n,d}m^{n,d}\right)} > C \quad \Leftrightarrow \quad \frac{n\pi_{\min}^{n,d}}{\log g^{n,d}} > C\frac{\log m^{n,d}}{\pi_{\min}^{n,d}{\rho_{\min}^{n,d}}^2}.$$

Or, par la première partie de la remarque 5.3.6, nous savons que :

$$\pi_{\min}^{n,d} \leq \frac{1}{g^{n,d}} \quad \text{et} \quad \rho_{\min}^{n,d} \leq \frac{1}{m^{n,d}}$$

et nous obtenons finalement que :

$$\begin{aligned}\frac{n\pi_{\min}^{n,d}}{\log g^{n,d}} &> C\frac{\log m^{n,d}}{\pi_{\min}^{n,d}{\rho_{\min}^{n,d}}^2} \\ &> g^{n,d}Cm^{n,d^2}\log m^{n,d} \\ &\underset{n,d}{\longrightarrow} +\infty\end{aligned}$$

par hypothèse sur $g^{n,d}$.
Comme la fraction tend vers l'infini, la limite inférieure est plus grande que 1.

— Sinon, nous savons que :

$$(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n \underset{n,d}{\longrightarrow} +\infty$$

donc, en particulier, pour tout $M > 0$, il existe $N$ tel que pour tout $n$ plus grand que $N$, nous ayons :

$$(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n > M$$

et dans ce cas, nous avons :

$$\begin{aligned}(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n > M \quad &\Leftrightarrow \quad n\pi_{\min}^{n,d} > M \times \underbrace{\frac{1}{\pi_{\min}^{n,d}(\rho_{\min}^{n,d})^2}}_{\geq 1} \\ &\Leftrightarrow \quad n\pi_{\min}^{n,d} > M.\end{aligned}$$

Comme ceci est vrai pour tout $M > 0$, $n\pi_{\min}^{n,d}$ tend vers $+\infty$.

A l'aide des conditions 1 et 2, la dernière partie de la borne tend vers 0 comme expliqué en annexe 5.C. Pour la borne sur les estimateurs $\widetilde{\boldsymbol{\pi}}$, nous reprenons la borne sur le nombre de classes en ligne et, par une conséquence de l'hypothèse 1 sur les lignes, il existe une constance $C'$ telle que pour $n$ et $d$ assez grands, nous ayons :

$$g^{n,d} \leq C' \sqrt{\frac{d}{\log n}}.$$

Ainsi, nous avons :

$$\begin{aligned}
g^{n,d} e^{-2nt^2} &\leq C' \sqrt{\frac{d}{\log n}} e^{-2nt^2} \\
&\leq C' \exp\left[\frac{1}{2}\left(\log d - \log\log n\right) - 2nt^2\right] \\
&\leq C' \exp\Bigg[ \underbrace{-2n\Big(-\underbrace{\frac{\log d}{4n}}_{=o(1)} + \frac{\log\log n}{4n} + t^2\Big)}_{\underset{n,d\to+\infty}{\longrightarrow} t^2} \Bigg] \\
&\underset{n,d\to+\infty}{\longrightarrow} 0.
\end{aligned}$$

Nous voyons une nouvelle fois le problème de la double asymptotique.
Par les mêmes arguments, nous démontrons la convergence vers 0 de la borne sur les estimateurs $\widetilde{\boldsymbol{\rho}}$.

Pour la borne sur les estimateurs $\widetilde{\boldsymbol{\alpha}}$, dans le cas où $g^{n,d}m^{n,d}$ tend vers l'infini, les condtions 3 et 4 nous permettent de dire qu'il existe une constante $C' > 1$ telle que pour $n$ et $d$ assez grands, nous ayons :

$$\begin{aligned}
\frac{(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n}{\log\left(g^{n,d}m^{n,d}\right)} &\geq C' \\
\frac{(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 d}{\log\left(g^{n,d}m^{n,d}\right)} &\geq C'
\end{aligned}$$

Or, pour tout $0 < C < 1$, la borne sur les estimateurs $\widetilde{\boldsymbol{\alpha}}$ se divise en trois parties :

$$g^{n,d}m^{n,d}\left[2e^{-2C\pi_{\min}^{n,d}\rho_{\min}^{n,d}ndt^2\left(\frac{1}{C}-1\right)} + 4e^{-(C\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n} + 4e^{-(C\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 d}\right].$$

En particulier, si nous prenons $\frac{1}{\sqrt{C'}} < C < 1$, alors $C\sqrt{C'} > 1$ et nous avons :

$$\begin{aligned}
g^{n,d}m^{n,d}e^{-(C\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n} &= \exp\left[\log\left(g^{n,d}m^{n,d}\right) - (C\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n\right] \\
&= \exp\left[\log\left(g^{n,d}m^{n,d}\right)\left(1 - C^2\frac{(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n}{\log\left(g^{n,d}m^{n,d}\right)}\right)\right] \\
&\leq \exp\Big[\log\left(g^{n,d}m^{n,d}\right)\underbrace{\left(1 - C^2C'\right)}_{<0}\Big] \\
&\underset{n,d\to+\infty}{\longrightarrow} 0.
\end{aligned}$$

Sinon, la condition que $(\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n$ tende vers l'infini donne directement le résultat.
Nous procédons de la même façon pour la dernière partie de la borne.

Pour la première partie, nous montrons qu'elle est négligeable par rapport à $g^{n,d}m^{n,d}e^{-(C\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n}$. Pour cela, nous montrons d'abord que $\pi_{\min}^{n,d}\rho_{\min}^{n,d}n$ tend vers $+\infty$. Or, la condtion 3 en ligne implique dans les deux cas que $\left(\pi_{\min}^{n,d}\rho_{\min}^{n,d}\right)^2 n$ tend vers $+\infty$. En particulier, pour tout $M > 0$, il existe $N > 0$ tel que pour tout $n \geq N$, nous ayons :

$$\begin{aligned}\left(\pi_{\min}^{n,d}\rho_{\min}^{n,d}\right)^2 n > M \quad &\Leftrightarrow \quad \pi_{\min}^{n,d}\rho_{\min}^{n,d}n > M\,\underbrace{\frac{1}{\pi_{\min}^{n,d}\rho_{\min}^{n,d}}}_{\leq 1} \\ &\Leftrightarrow \quad \pi_{\min}^{n,d}\rho_{\min}^{n,d}n > M.\end{aligned}$$

Et nous avons le résultat souhaité. Il ne reste plus qu'à montrer que le rapport tend bien vers 0 :

$$\begin{aligned}\frac{e^{-2C\pi_{\min}^{n,d}\rho_{\min}^{n,d}ndt^2\left(\frac{1}{C}-1\right)}}{e^{-(C\pi_{\min}^{n,d}\rho_{\min}^{n,d})^2 n}} \quad &= \quad \exp\left[-C\pi_{\min}^{n,d}\rho_{\min}^{n,d}n\underbrace{\left(2dt^2\underbrace{\left(\frac{1}{C}-1\right)}_{>0} - C\pi_{\min}^{n,d}\rho_{\min}^{n,d}\right)}_{>0 \text{ pour } d \text{ grand}}\right] \\ &\underset{n,d\to+\infty}{\longrightarrow} \quad 0.\end{aligned}$$

Et nous avons directement le résultat.

## 5.E Algorithme d'initialisation

La formalisation de l'algorithme d'initialisation est la suivante :

**Initialisation de l'amélioration de l'algorithme** $LG$ :

1. Calculer pour chaque indice $j \in \{0, \dots, d\}$ le nombre $N_j$ de valeurs $\overline{X_{i\bullet}}$ valant $\frac{j}{d}$.
2. Créer l'ensemble $\mathcal{D} = \left\{ j \in \{0, \dots, d\} \middle| N_j = N_{\max} \right\}$ des indices $j$ dont le nombre $N_j$ est égal à la valeur maximale $N_{\max}$.
   (a) Si $|\mathcal{D}| \leq g$, prendre $\mathcal{R} = \mathcal{D}$.
   (b) Sinon, créer $\mathcal{R}$ en prenant le sous ensemble de $\mathcal{D}$ de taille $g$ dont la distance minimale entre deux valeurs est la plus élevée.
3. Pour $N$ allant de $N_{\max}$ à 1 :
   (a) Créer l'ensemble $\mathcal{D}' = \left\{ j \in \{0, \dots, d\} \middle| N_j = N \right\}$ des indices $j$ dont le nombre $N_j$ est égal à la valeur $N$.
   (b) Créer l'ensemble $\mathcal{R}' = \left\{ j \in \mathcal{D}' \middle| \forall j' \in \mathcal{D}, |j - j'| > 1 \right\}$ des indices non voisins avec ceux de $\mathcal{D}$.
   (c) Pour chaque indice $j \in \mathcal{R}'$ :
      i. Si $|\mathcal{R}| < g$, ajouter $j$ à $\mathcal{R}$ (c'est-à-dire prendre $\mathcal{R} = \mathcal{R} \cup \{j\}$).
      ii. Sinon, conserver comme étant $\mathcal{R}$ le sous ensemble de $\mathcal{R} \cup \{j\}$ de taille $g$ dont la distance minimale entre deux valeurs est la plus élevée.
   (d) Fusionner les ensembles $\mathcal{D}$ et $\mathcal{D}'$ en un seul ensemble $\mathcal{D}$ (c'est-à-dire prendre $\mathcal{D} = \mathcal{D} \cup \mathcal{D}'$).
4. Renvoyer l'ensemble $\left\{ \frac{j}{d} \middle| j \in \mathcal{R} \right\}$.

*Remarque numérique* 5.E.1.
Si l'étape 3 est correctement implémentée, chaque itération peut être faite en ne regardant qu'une seule fois chaque indice $j$. La complexité de l'algorithme est alors de $\mathcal{O}(N_{\max} d)$.
Pour accélérer la convergence, nous pouvons insérer des critères d'arrêt dans la boucle. Par exemple, nous pouvons arrêter l'algorithme lorsque le plus grand intervalle du complémentaire de $\mathcal{D}$ est de longueur inférieure à la distance minimale entre deux éléments de $\mathcal{R}$ puisque nous rejetons automatiquement tous les indices restant.

## 5.F Démonstrations sur la sélection de modèles

Dans cette partie, nous démontrons les résultats de la partie 5.6. Une nouvelle fois, nous regardons uniquement les démonstrations sur les lignes.

### 5.F.1 Borne de l'estimation de $\widehat{g}^{LG}$ à $n$, $d$ et $S$ fixés

La démonstration reprend celle de la consistance de l'estimateur des classes de la section 5.B. Pour que le bon nombre de classes soit sélectionné, il faut à la fois qu'il y ait un représentant de chaque classe et que les bosses soient à la fois suffisamment espacées et pas trop dispersées par rapport à la valeur $S$. En particulier, l'évènement idéal que nous étudions est $B_S = A_{g^\star} \bigcap \{2d_n < S\} \bigcap \{S \leq \delta_\tau - 2d_n\}$

$$\text{où } d_n = \max_{k \in \{1 \dots, g\}} \sup_{i | z_{ik} = 1} \left| \overline{X_{i\bullet}} - \tau_k \right| \text{ et } A_{g^\star} = \bigcap_{k=1}^{g^\star} \{ z^\star_{+k} \neq 0 \}.$$

En reprenant les démonstrations de la section 5.B, nous avons :

— si les lignes $i$ et $i'$ appartiennent à la même classe $k$, nous avons :

$$\begin{aligned} \left|\overline{X_{i\bullet}} - \overline{X_{i'\bullet}}\right| &\leq 2d_n \\ &< S \end{aligned}$$

si nous sommes sous l'évènement $\{2d_n < S\}$.

— Au contraire, si la ligne $i$ appartient à la classe $k$ et la ligne $i'$ appartient à une classe différente $k'$, nous avons :

$$\begin{aligned} \left|\overline{X_{i\bullet}} - \overline{X_{i'\bullet}}\right| &\geq \delta_\tau - 2d_n \\ &\geq S. \end{aligned}$$

si nous sommes sous l'évènement $\{S \leq \delta_\tau - 2d_n\}$.

Conclusion, sous la condition qu'il y ait un représentant de chaque classe (c'est-à-dire sous l'hypothèse que nous soyons sous l'évènement $A_{g^\star}$), il y a exactement $g^\star - 1$ sauts qui sont sélectionnés.

À partir de là, nous calculons la probabilité de l'évènement complémentaire :

$$\begin{aligned} \mathbb{P}\left(\widehat{g}^{LG} \neq g^\star\right) &\leq \mathbb{P}\left(\overline{B_S}\right) \\ &\leq \mathbb{P}\left(\overline{A_{g^\star}}\right) + \mathbb{P}\left(\overline{\{2d_n < S\}}\right) + \mathbb{P}\left(\overline{\{S \leq \delta_\tau - 2d_n\}}\right). \\ &\leq \mathbb{P}\left(\overline{A_{g^\star}}\right) + \mathbb{P}\left(\{2d_n \geq S\}\right) + \mathbb{P}\left(\{S > \delta_\tau - 2d_n\}\right). \end{aligned}$$

Or, nous savons par la section 5.B, d'une part que :

$$\mathbb{P}\left(\overline{A_{g^\star}}\right) \leq g^\star \left(1 - \pi_{\min}\right)^n,$$

et d'autre part que pour tout $t > 0$ :

$$\mathbb{P}(d_n > t) \leq 2ne^{-2dt^2}.$$

En particulier, comme nous avons supposé que $S \in ]0, \delta_\tau[$, nous avons, d'une part que pour tout $\varepsilon > 0$ :

$$\begin{aligned} \mathbb{P}\left(2d_n \geq S\right) &\leq \mathbb{P}\left(2d_n > S + \varepsilon\right) \\ &\leq \mathbb{P}\left(d_n > \frac{S+\varepsilon}{2}\right) \\ &\leq 2ne^{-2d\left(\frac{S+\varepsilon}{2}\right)^2} \\ &\leq 2ne^{-d\frac{(S+\varepsilon)^2}{2}} \end{aligned}$$

et comme ceci est vrai pour tout $\varepsilon > 0$, nous obtenons :

$$\mathbb{P}\left(2d_n \geq S\right) \leq 2ne^{-d\frac{S^2}{2}}.$$

D'autre part, nous avons :

$$\begin{aligned}\mathbb{P}\left(S > \delta_\tau - 2d_n\right) &= \mathbb{P}\left(d_n > \frac{\delta_\tau - S}{2}\right)\\ &\leq 2ne^{-2d\left(\frac{\delta_\tau - S}{2}\right)^2}\\ &\leq 2ne^{-d\frac{(\delta_\tau - S)^2}{2}}.\end{aligned}$$

Et nous obtenons la borne 5.4.

### 5.F.2 Consistance des estimateurs du nombre de classes

Tout d'abord, nous utilisons la première condition : comme $S^{n,d}$ tend vers 0 lorsque $n$ et $d$ tendent vers l'infini alors il existe $(N, D) \in \mathbb{N}^2$ tel que pour tout $n \geq N$ et pour tout $d \geq D$, $S^{n,d} < \delta_\tau$ et nous sommes alors sous les conditions du théorème 5.6.1. Nous supposons donc que $n$ et $d$ vérifient ces conditions.

Comme les paramètres sont fixes, la dernière partie de la borne tend vers 0 lorsque $n$ tend vers l'infini. De plus, comme nous avons que $S^{n,d} < \delta_\tau$ et que $\frac{d}{\log n}$ tend vers l'infini lorsque $n$ et $d$ tendent vers l'infini par la condition 2 (voir la remarque 5.6.4), la deuxième partie de la borne tend vers 0 également.

Enfin, pour la première partie de la borne, nous utilisons à nouveau la condition 2 : il existe une constante $C > \sqrt{2}$ telle que pour $n$ et $d$ assez grand, nous ayons :

$$S^{n,d^2}\frac{d}{\log n} \geq C^2$$

et par le même raisonnement que dans les sections précédentes, nous avons :

$$\begin{aligned}2ne^{-d\frac{S^{n,d^2}}{2}} &= 2\exp\left[\log n - d\frac{S^{n,d^2}}{2}\right]\\ &= 2\exp\left[-\frac{\log n}{2}\left(\frac{d}{\log n}S^{n,d^2} - 2\right)\right]\\ &\leq 2\exp\left[-\frac{\log n}{2}\underbrace{\left(C^2 - 2\right)}_{>0}\right]\\ &\underset{n,d\to+\infty}{\longrightarrow} 0\end{aligned}$$

### 5.F.3 Consistance de l'algorithme *LG*

Pour démontrer le théorème 5.6.6, nous divisons l'espace en deux :

1. avoir les bonnes estimations sachant que nous avons le bon nombre de classes en ligne et en colonnes et nous réutilisons la borne du théorème 5.4.3,
2. ne pas avoir le bon nombre de classes et nous réutilisons la démonstration précédente.

Pour pouvoir utiliser le théorème 5.4.3, il faut vérifier la première condition (les autres étant supposées dans le théorème 5.6.6). Pour cela, nous utilisons la première hypothèse du théorème : il existe une constante $C > 2$ telle que pour $n$ et $d$ assez grands, nous ayons :

$$\frac{\delta_\tau^{n,d}}{S^{n,d}} \geq C$$

et avec la seconde hypothèse, il existe une constante $C' > \sqrt{2}$ telle que pour $n$ et $d$ assez grands, nous ayons :

$$S^{n,d}\sqrt{\frac{d}{\log n}} \geq C'$$

et donc, pour $n$ et $d$ assez grands, nous avons :

$$\delta_\tau^{n,d}\sqrt{\frac{d}{\log n}} \geq CS^{n,d}\sqrt{\frac{d}{\log n}} \geq CC' > 2\sqrt{2}$$

et nous retrouvons la condition du théorème.

## 5.G Simulations

Dans cette partie, nous mettons les graphiques associés aux résultats des simulations. Pour alléger la visualisation, nous choisissons de représenter les résultats sous forme de graphiques en trois dimensions où les couleurs rouges symbolisent de fortes valeurs alors que les couleurs bleues représentent de faibles valeurs.

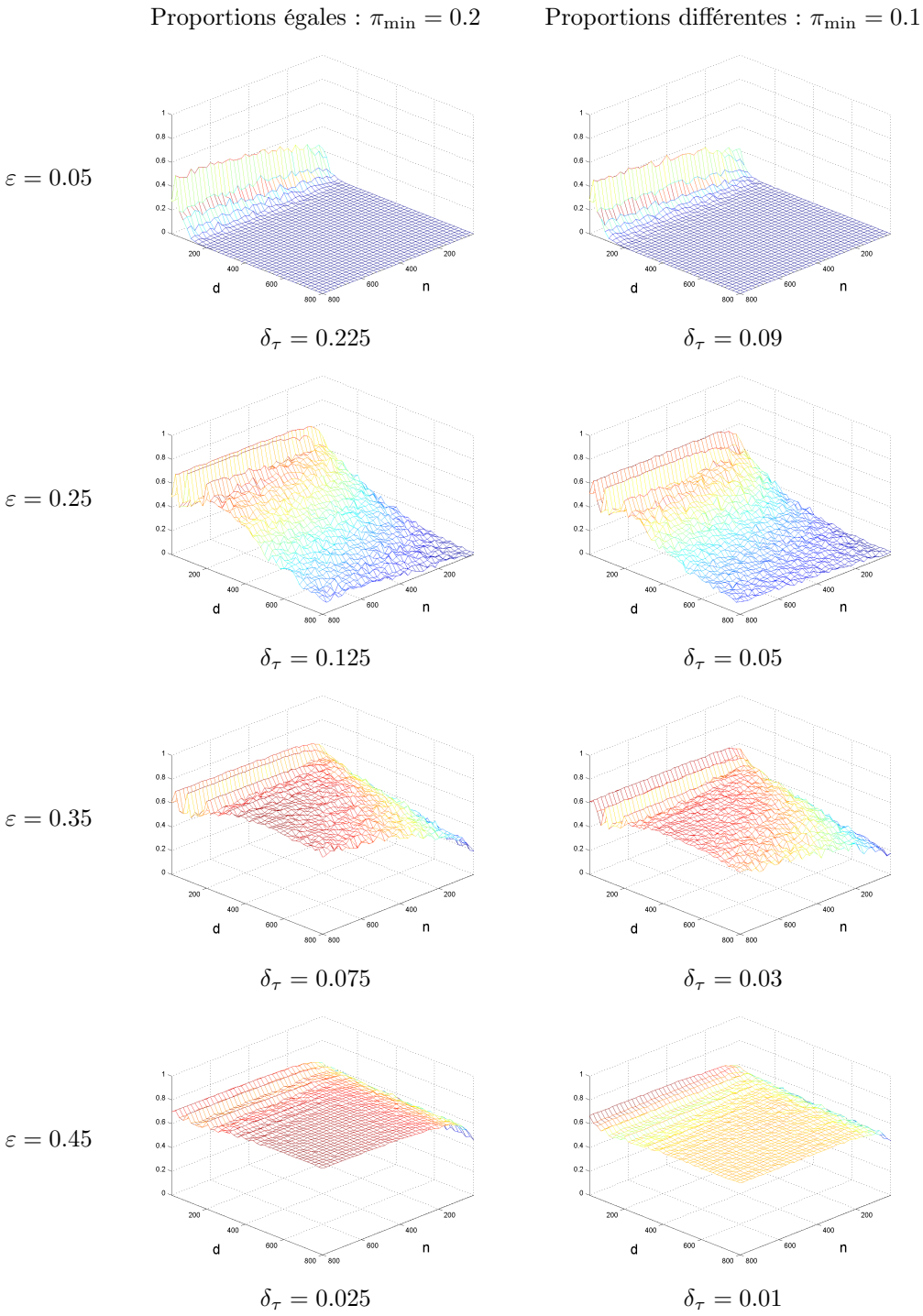



FIGURE 5.6 – Influence du nombre de lignes et de colonnes sur l'erreur de classification pour les lignes dans des cas de proportions égales (colonne de gauche) et inégales (colonnes de droite) et pour des valeurs différentes de $\varepsilon$ (lignes) pour l'erreur $e_{(n,g)\times(d,m)}$.

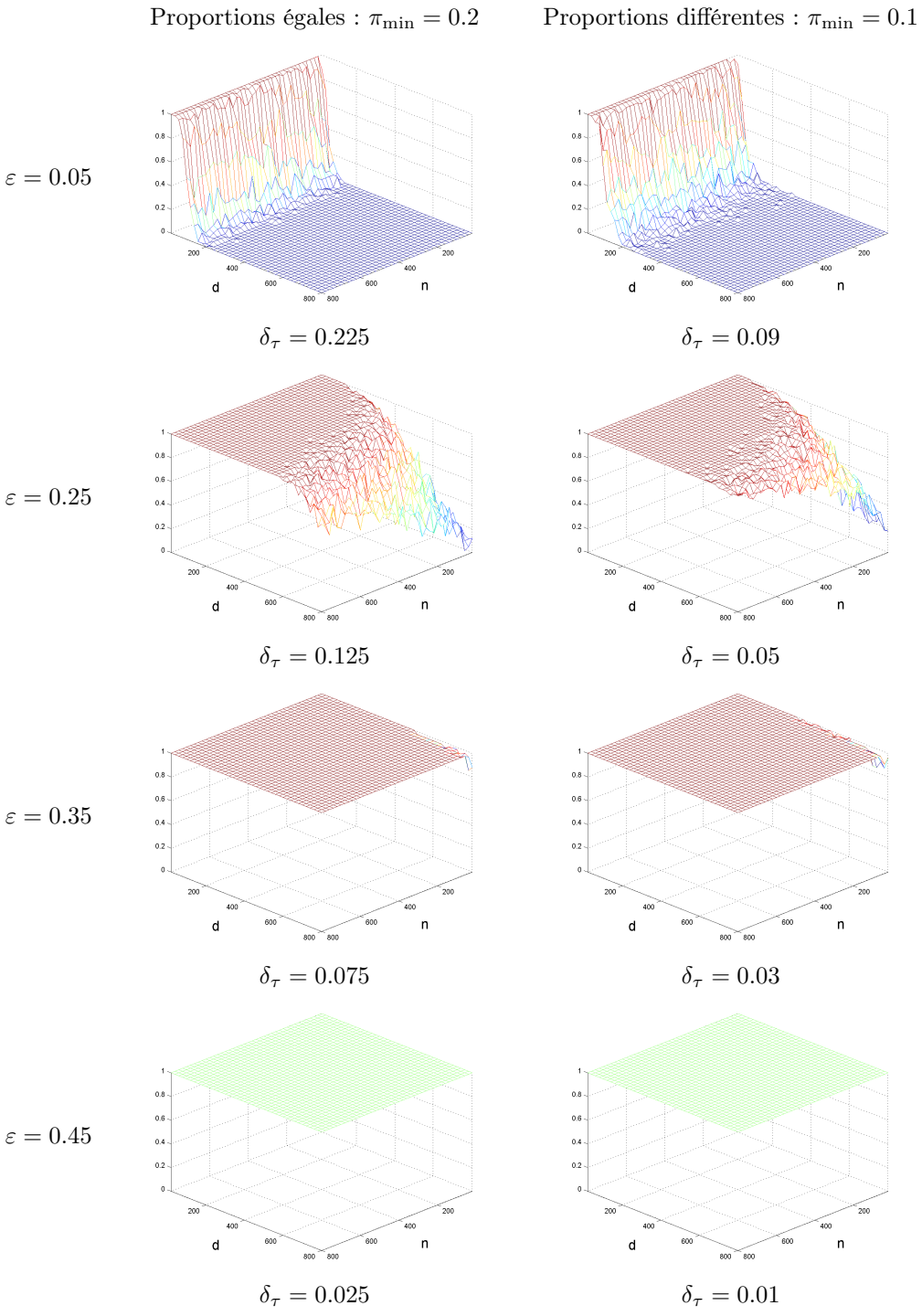



FIGURE 5.7 – Influence du nombre de lignes et de colonnes sur l'erreur de classification pour les lignes dans des cas de proportions égales (colonne de gauche) et inégales (colonnes de droite) et pour des valeurs différentes de $\varepsilon$ (lignes) pour l'erreur $e^{0-1}_{(n,g)\times(d,m)}$.

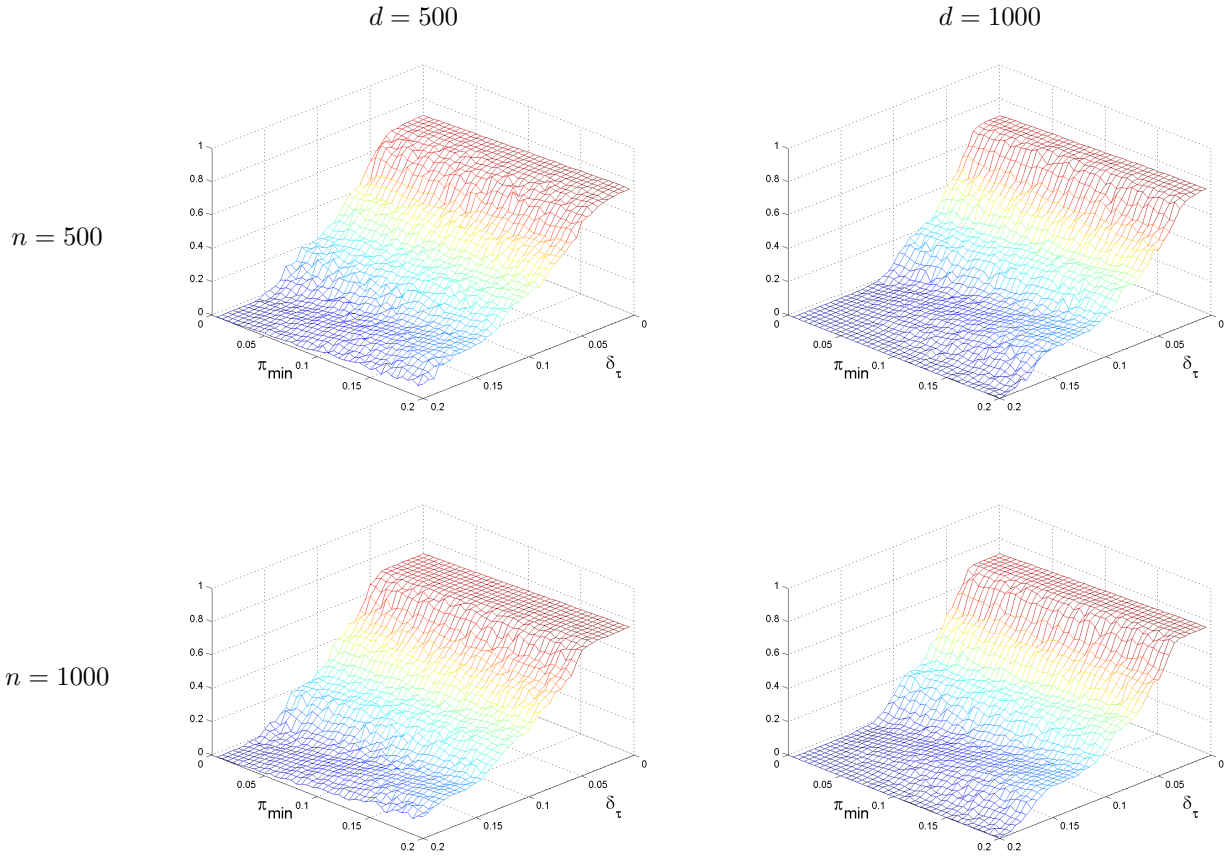



FIGURE 5.8 – Influence des paramètres $\delta_\tau$ et $\pi_{\min}$ sur l'erreur de classification pour les lignes pour des matrices avec $d = 500$ colonnes (colonne de gauche) et $d = 1000$ colonnes (colonnes de droite) et pour un nombre de lignes $n$ de 500 et 1000 (lignes) pour l'erreur $e_{(n,g)\times(d,m)}$.

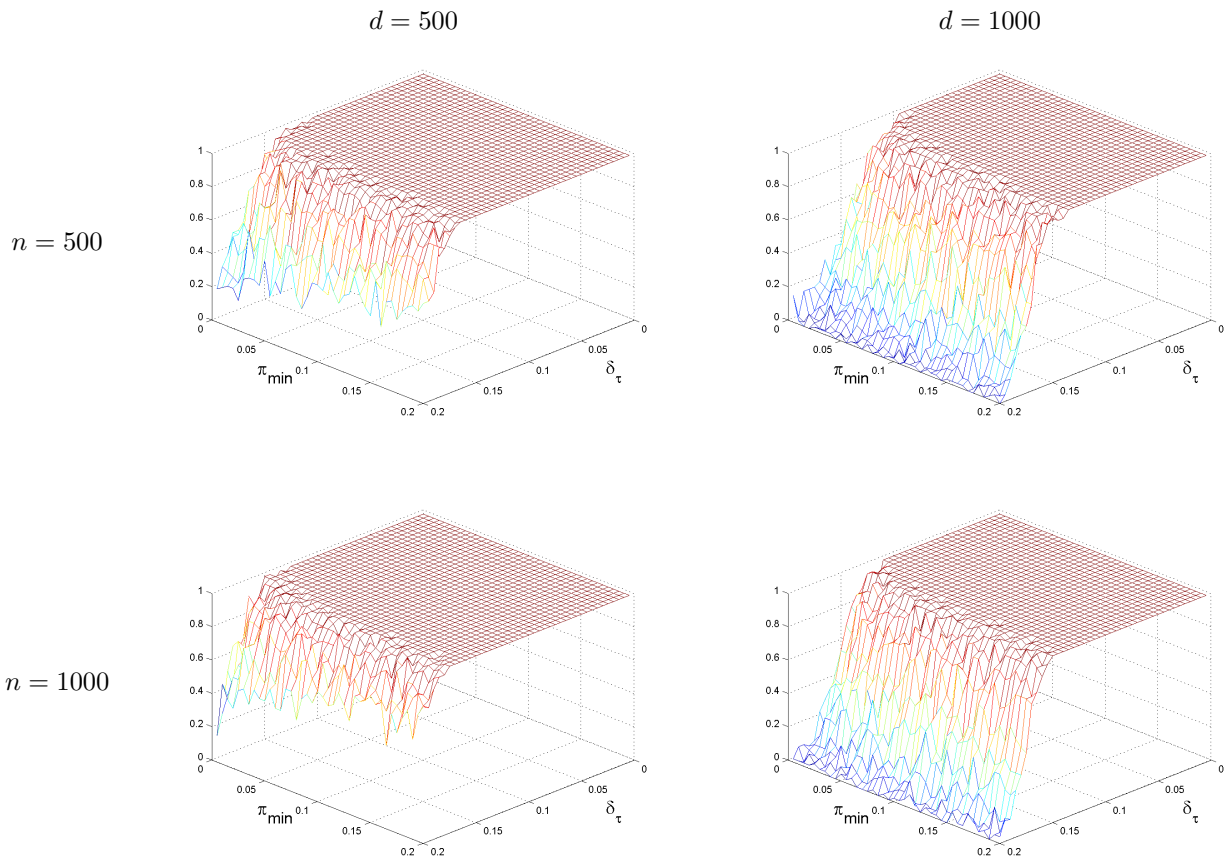



FIGURE 5.9 – Influence des paramètres $\delta_\tau$ et $\pi_{\min}$ sur l'erreur de classification pour les lignes pour des matrices avec $d = 500$ colonnes (colonne de gauche) et $d = 1000$ colonnes (colonnes de droite) et pour un nombre de lignes $n$ de 500 et 1000 (lignes) pour l'erreur $e^{0-1}_{(n,g)\times(d,m)}$.

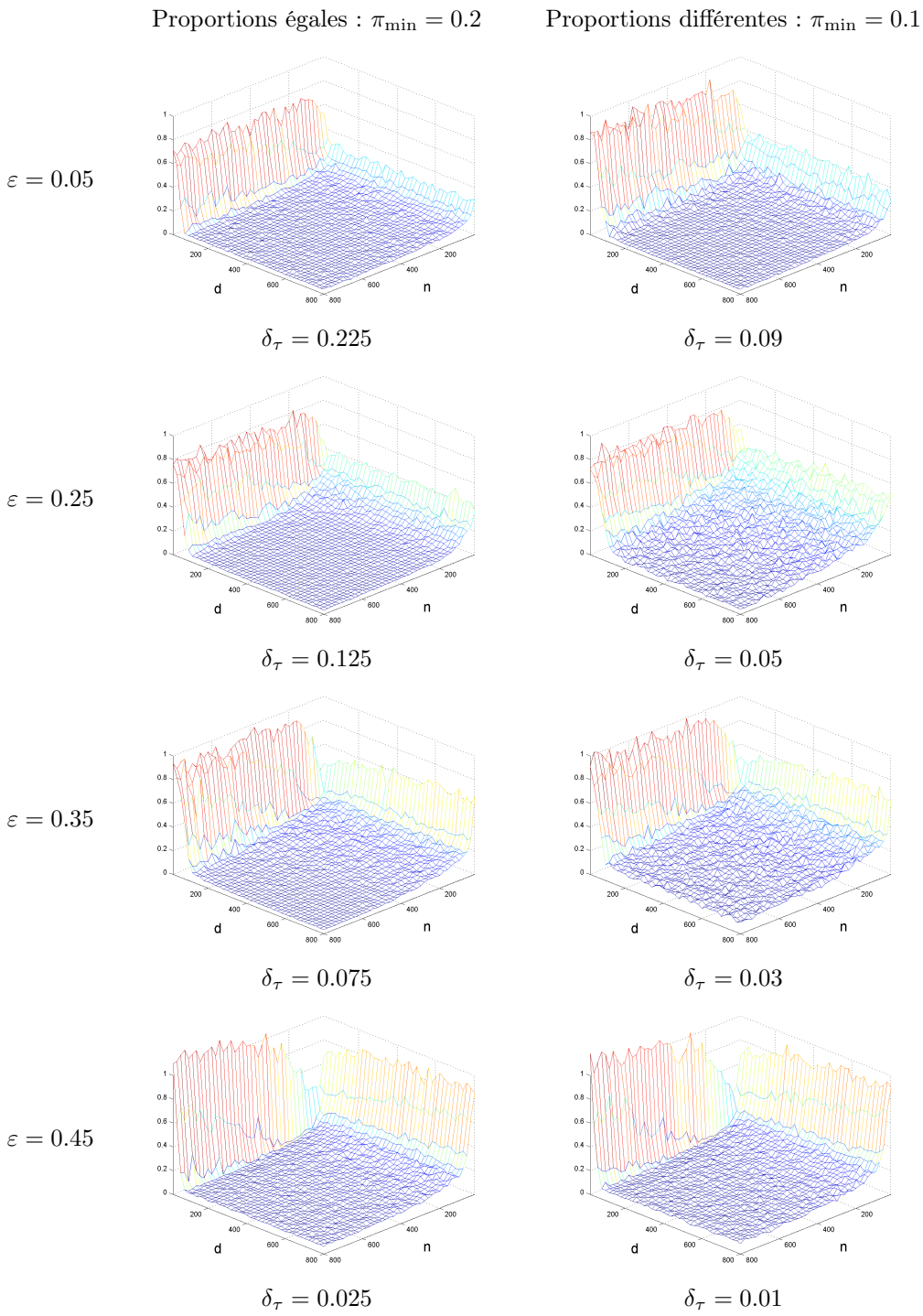



FIGURE 5.10 – Influence du nombre de lignes et de colonnes sur l'erreur des estimations des $\boldsymbol{\tau}$ renvoyés par l'initialisation dans des cas de proportions égales (colonne de gauche) et inégales (colonnes de droite) et pour des valeurs différentes de $\varepsilon$ (lignes).

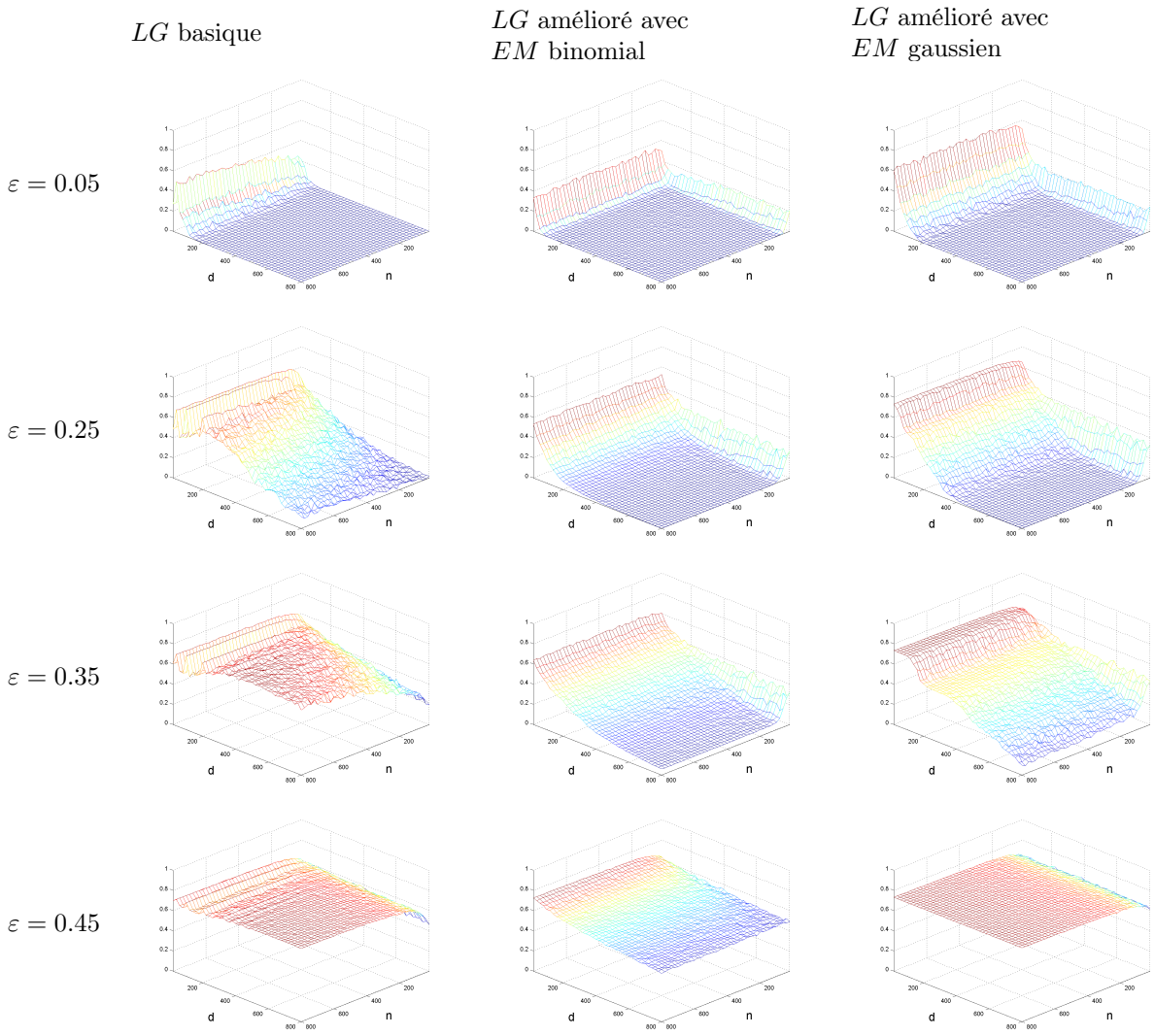



FIGURE 5.11 – Comparaison du taux moyen d'erreur de classification $e_{(n,g)\times(d,m)}$ entre l'algorithme $LG$ basique (colonne de gauche), son amélioration avec un algorithme $EM$ présupposant que les lois des mélanges sont des binomiales (colonne centrale) et avec un algorithme $EM$ présupposant que les lois des mélanges sont des gaussiennes (colonne de droite) dans le cas des proportions équilibrées (donc $\pi_{\min} = 0.2$).

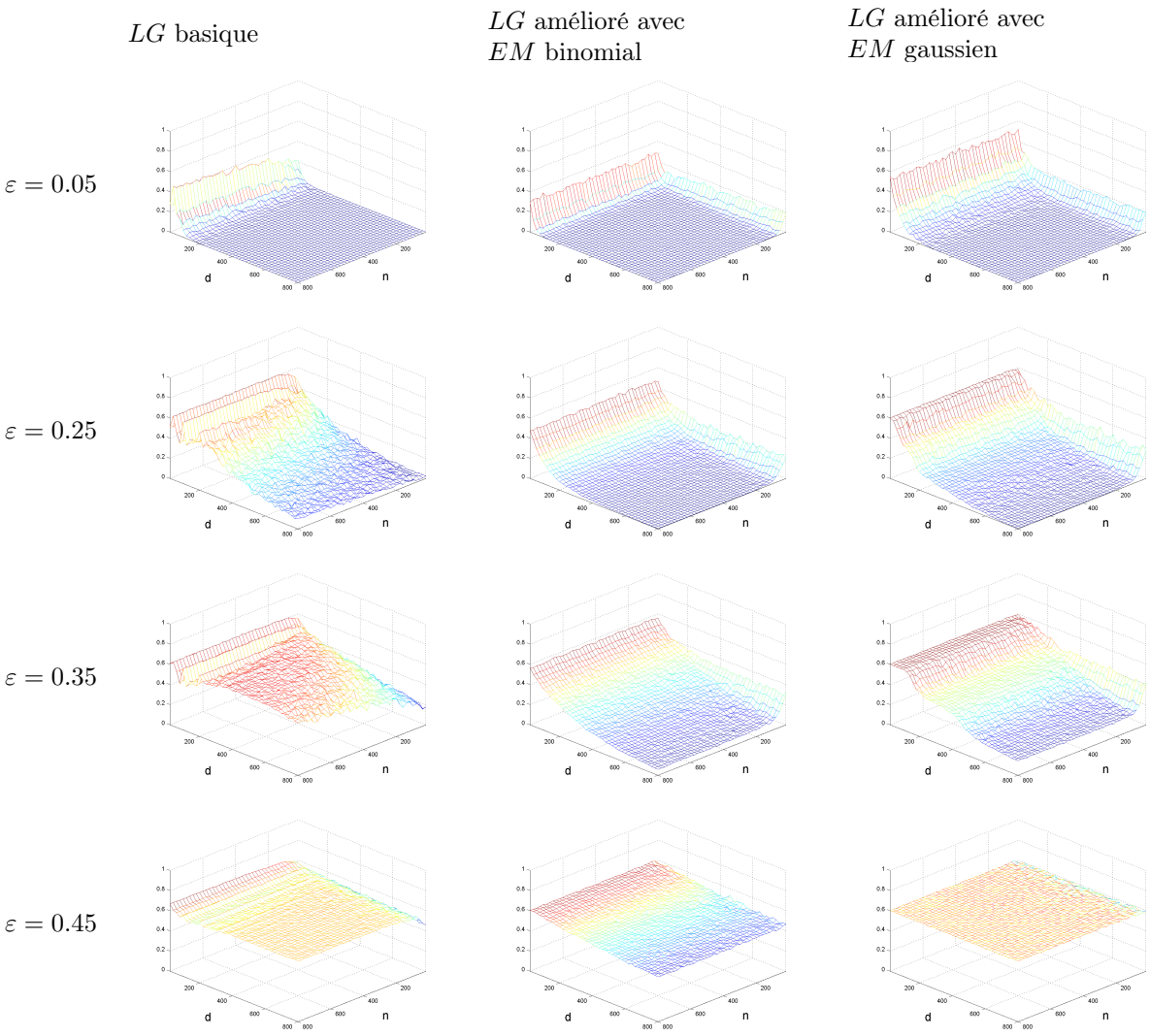



FIGURE 5.12 – Comparaison du taux moyen d'erreur de classification $e_{(n,g)\times(d,m)}$ entre l'algorithme $LG$ basique (colonne de gauche), son amélioration avec un algorithme $EM$ présupposant que les lois des mélanges sont des binomiales (colonne centrale) et avec un algorithme $EM$ présupposant que les lois des mélanges sont des gaussiennes (colonne de droite) dans le cas des proportions inégales (donc $\pi_{\min} = 0.1$).

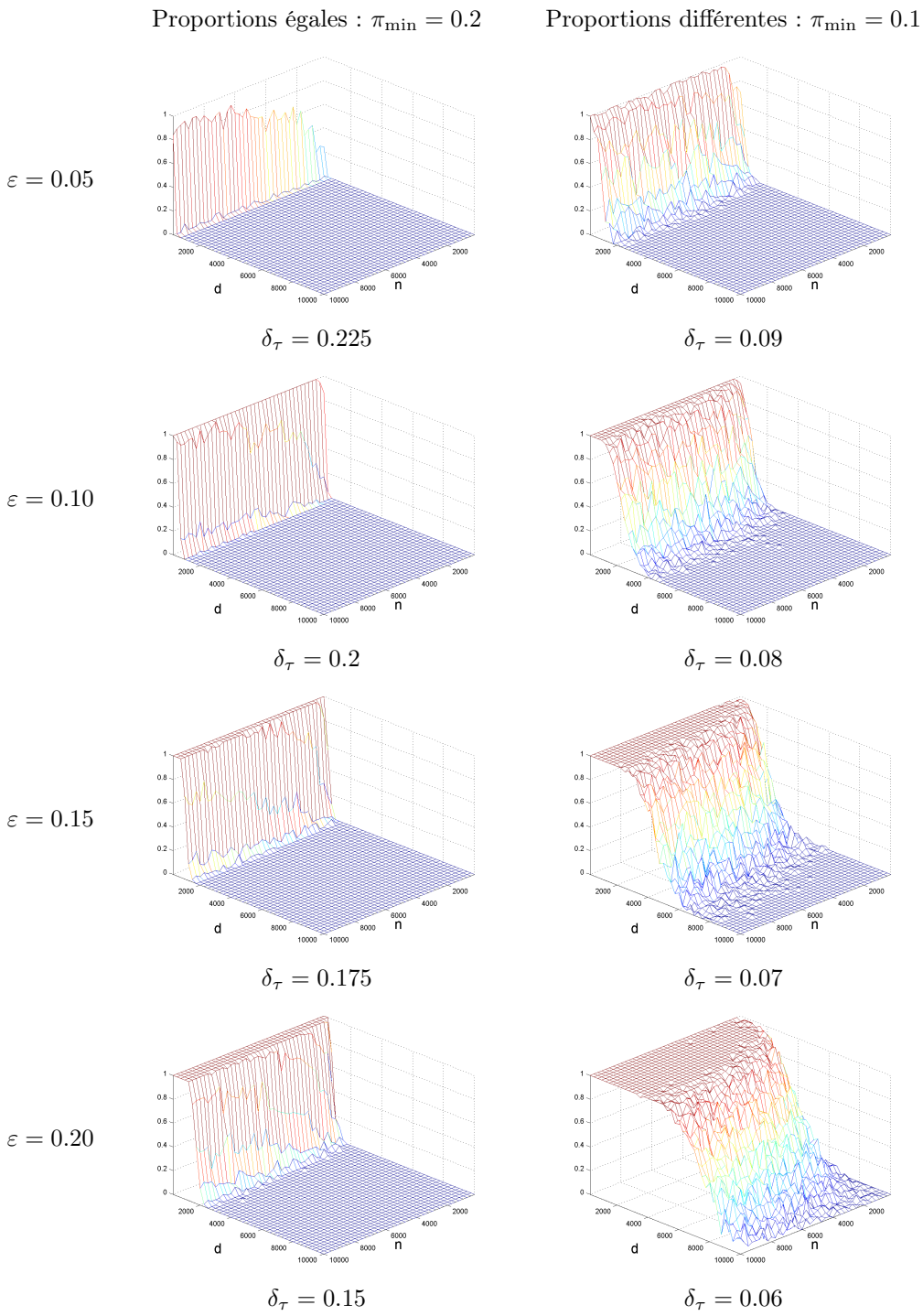



FIGURE 5.13 – Pourcentage du nombre de fois où le mauvais nombre de classes en ligne a été sélectionné par l'algorithme $LG$ dans des cas de proportions égales (colonne de gauche) et inégales (colonnes de droite) et pour des valeurs différentes de $\varepsilon$ (lignes) pour le seuil $S = \delta_\tau/2$.

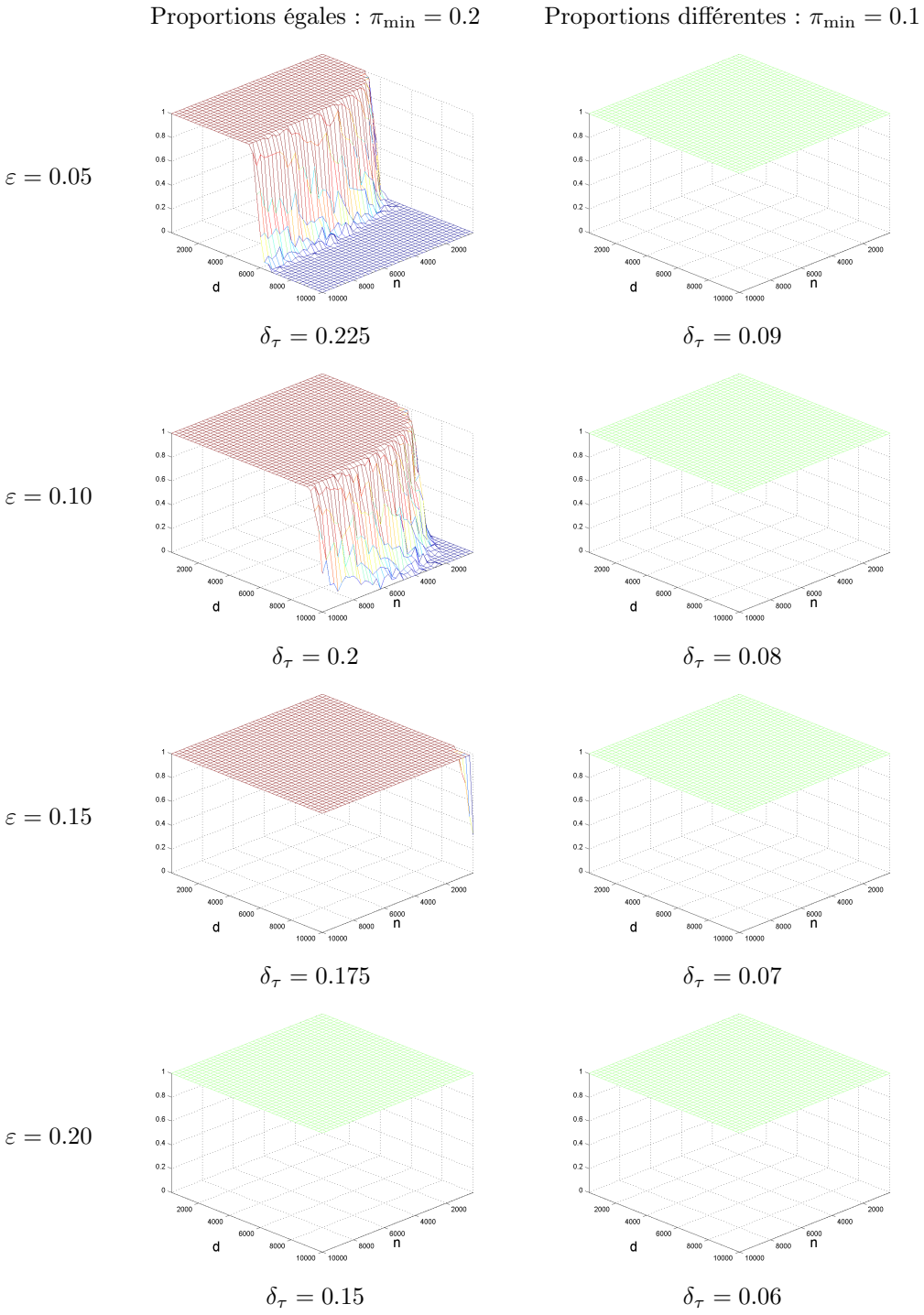



FIGURE 5.14 – Pourcentage du nombre de fois où le mauvais nombre de classes en ligne a été sélectionné par l'algorithme $LG$ dans des cas de proportions égales (colonne de gauche) et inégales (colonnes de droite) et pour des valeurs différentes de $\varepsilon$ (lignes) pour le seuil $S^{n,d} = \left(\frac{\log n}{d}\right)^{1/4}$.

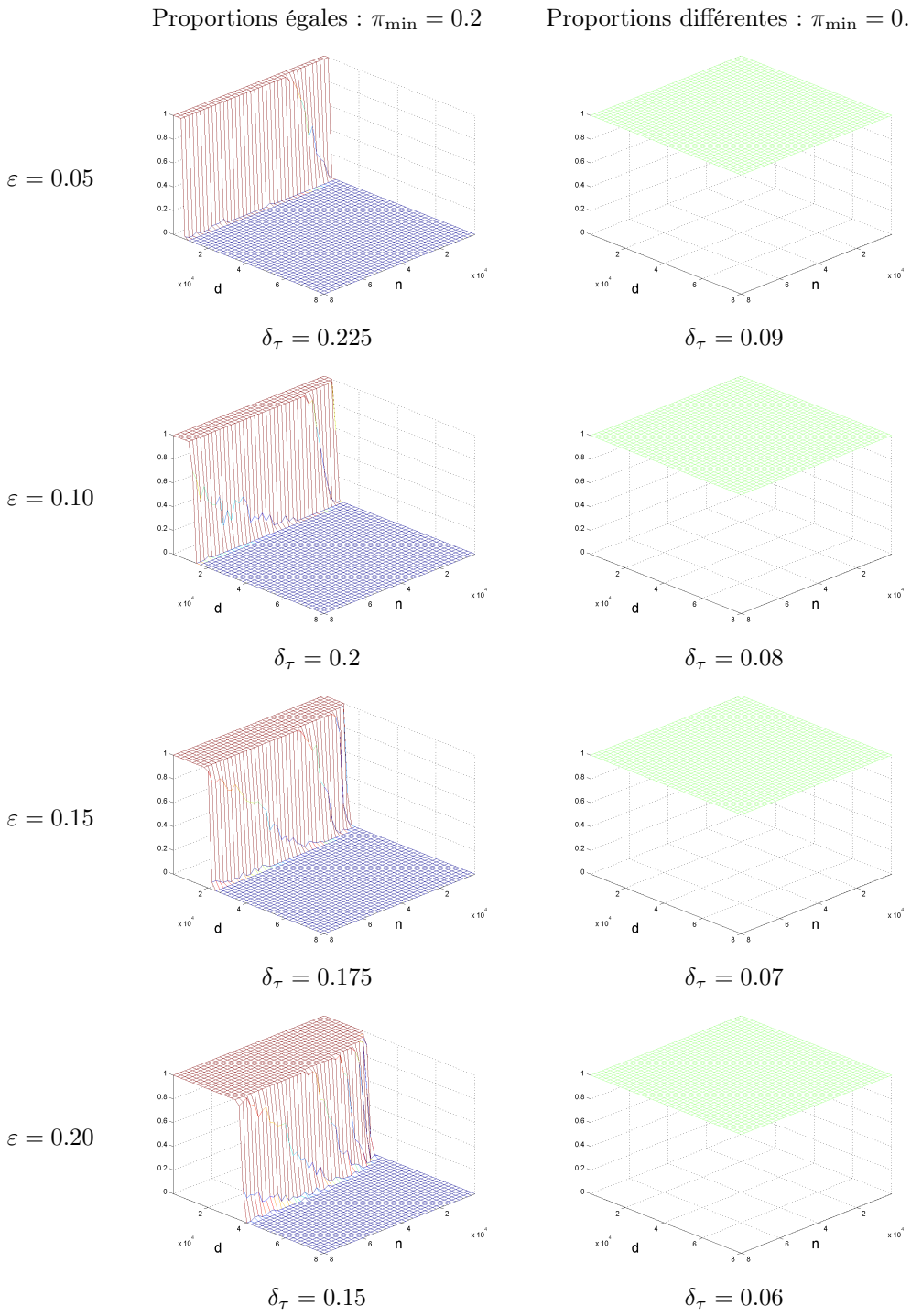



FIGURE 5.15 – Pourcentage du nombre de fois où le mauvais nombre de classes en ligne a été sélectionné par l'algorithme $LG$ dans des cas de proportions égales (colonne de gauche) et inégales (colonnes de droite) et pour des valeurs différentes de $\varepsilon$ (lignes) pour le seuil $S^{n,d} = \left(\frac{\log n}{d}\right)^{1/4}$ pour un nombre de lignes et de colonnes allant de 2 000 à 80 000 par pas de 2 000.

# Chapitre 6

# Comparaisons des différentes stratégies

*"Simulations are believed by no one except those who conducted them.*
*Experimental results are believed by everyone except those who conducted them."*
Anonyme

**Sommaire**


**6.1 Introduction . . . . . 163**
**6.2 Procédures possibles . . . . . 164**
6.2.1 Algorithmes d'initialisation . . . . . 164
6.2.2 Algorithmes sensibles aux initialisations . . . . . 165
**6.3 Comparaison . . . . . 166**
6.3.1 Comparaison sur des données simulées . . . . . 167
6.3.2 Comparaison sur données réelles . . . . . 168
**6.4 Comparaison deux à deux . . . . . 174**
6.4.1 Comparaison entre l'échantillonneur de *Gibbs* et *SEM-Gibbs* . . . . . 174
6.4.2 Comparaison entre les algorithmes *CEM* et *V-Bayes* . . . . . 176
**6.5 Discussion générale . . . . . 176**


## 6.1 Introduction

Dans les chapitres précédents, nous avons étudié différents algorithmes et différentes stratégies pour estimer les partitions et paramètres du modèle des blocs latents. La question de la comparaison est délicate : un nombre fini d'initialisations à partir de valeurs aléatoires favorise les algorithmes peu sensibles à celles-ci, à l'opposé, un nombre d'itérations fixé trop petit favorise les algorithmes rapides...

Dans ce chapitre, nous proposons de comparer les différentes procédures en tenant compte d'un temps d'exécution imparti. Les algorithmes rapides à converger sont ainsi relancés plus souvent tandis que les autres peuvent perdre en qualité s'ils sont trop lents.

Pour cela, nous commençons par un rappel des différentes procédures en listant les algorithmes pouvant servir pour l'initialisation et ceux servant généralement pour "affiner" les résultats. À ces derniers, nous ajoutons l'algorithme *CEM* déjà étudié par Nadif et Govaert (2007) (voir aussi Govaert et Nadif, 2008) et son adaptation bayésienne dont nous rappelons la forme.

Ensuite, nous comparons les procédures pour un temps d'exécution (*elapsed*) fixé sur des données simulées et réelles. Enfin, nous comparons plus particulièrement les algorithmes aux comportements proches pour voir dans quel cadre il est préférable d'utiliser chacun.

## 6.2 Procédures possibles

Nous avons vu jusqu'à présent différents algorithmes pouvant servir pour l'initialisation ou pour "affiner" les résultats. Dans cette partie, nous recensons chacun d'entre eux en rappelant leurs intérêts et inconvénients.

### 6.2.1 Algorithmes d'initialisation

Nous englobons sous le terme "d'algorithmes d'initialisation" les algorithmes et procédures servant à proprement parler pour initialiser et ceux qui sont peu sensibles aux initialisations.

#### Initialisation aléatoire

La façon la plus courante d'initialiser un algorithme est de prendre des premières valeurs au hasard dans l'espace des possibilités (en favorisant ou non certaines). Cette procédure possède le grand avantage d'être quasi-immédiate ce qui permet de pouvoir l'utiliser plusieurs fois en peu de temps. En revanche, les choix du nombre d'initialisations nécessaire et de la répartition des initialisations dans l'espace des paramètres sont des problèmes assez complexes ayant une grande importance sur les résultats ultérieurs.

#### Algorithme *SEM-Gibbs*

Étant peu sensible aux initialisations et renvoyant généralement des résultats proches de ceux souhaités, l'algorithme *SEM-Gibbs* peut servir comme initialisation d'autres algorithmes. Pour éviter les états absorbants mentionnés dans la remarque 2.4.2, nous itérons l'étape $SE$ tant que l'algorithme renvoie des couples $\left(\mathbf{z}^{(c+1)}, \mathbf{w}^{(c+1)}\right)$ de matrices avec des classes vides (afin d'éviter les états absorbants de type I) et donnons arbitrairement la valeur 0.1 (resp. 0.9) à un paramètre $\alpha_{k\ell}^{(c+1)}$ trop proche de 0 (resp. de 1) afin d'éviter les états absorbants de type II.

Ces choix ont l'avantage de limiter les cas dégénérés mais peuvent ralentir l'algorithme si celui-ci se rapproche d'états absorbants.

#### Échantillonneur de *Gibbs*

Comme l'algorithme *SEM-Gibbs*, l'échantillonneur de *Gibbs* peut servir pour l'initialisation. Pour les procédures, nous ne simulons qu'une seule chaîne de 5 000 itérations. Comme sur les données simulées de la section 3.6.2, la procédure avec 5 000 itérations donne des résultats proches de ceux renvoyés lorsqu'un critère d'arrêt est utilisé, ce choix d'arrêt nous permet de prendre le temps utilisé par l'échantillonneur de *Gibbs* comme référence (qui est différent suivant la taille des matrices, le nombre de classes, la difficultés ou encore la nature des données mais à peu près identique pour des matrices de mêmes configurations).

#### Algorithme *LG*

Pour les simulations, nous utilisons la version améliorée avec un algorithme *EM* utilisant des lois binomiales et l'initialisation déterministe détaillée sur la figure 5.3 (sauf pour le cas de données multinomiales où nous utilisons comme initialisation l'algorithme *CEM*). L'algorithme *Largest Gaps* est ainsi déterministe pour les données binaires.

### 6.2.2 Algorithmes sensibles aux initialisations

Différents algorithmes peuvent être utilisés après la phase d'initialisation.

#### Algorithmes *VEM* et *V-Bayes*

Les algorithmes *VEM* et *V-Bayes* donnent des maxima locaux de l'énergie libre, approximation de la log-vraisemblance calculée en supposant indépendantes les probabilités conditionnellement aux observations.

Le comportement de l'énergie libre n'est pas connu mais nous pensons, avec l'appui de simulations, que cette approximation est meilleure lorsque le nombre d'observations est grand.

#### Algorithme *CEM* et *CEM-Bayes*

Le dernier algorithme étudié est l'algorithme *CEM* (*Classification Expectation Maximisation*) introduit par Celeux et Govaert (1992) dans le cas des modèles de mélanges classiques et adapté une première fois par Govaert et Nadif (2008) pour le modèle des blocs latents. Le principe est d'ajouter une étape d'affectation à la classe majoritaire après l'étape $E$. L'algorithme cherche alors à maximiser la log-vraisemblance complète $\log p(\mathbf{x}, \mathbf{z}, \mathbf{w}; \theta)$.

En plus de sa rapidité, l'algorithme possède l'intérêt de ne pas nécessiter d'approximation du critère à maximiser. De plus, comme nous le rappelons en section 1.3.2.4 et plus particulièrement par l'équation (1.3), Mariadassou et Matias (2012) ont démontré que, pour le cas du modèle des blocs latents et sous le vrai nombre de classes en ligne et en colonne, la densité $p\left(\mathbf{z}, \mathbf{w} \middle| \mathbf{x}; \widehat{\theta}\right)$ se concentre autour des vraies partitions sous la condition que l'estimateur $\widehat{\theta}$ soit tel que $\widehat{\boldsymbol{\alpha}}$ converge en probabilité vers les vrais paramètres $\boldsymbol{\alpha}^\star$. Or, nous avons la décomposition suivante :

$$\log p(\mathbf{x}, \mathbf{z}, \mathbf{w}; \theta) = \log p(\mathbf{z}, \mathbf{w} | \mathbf{x}; \theta) + \log p(\mathbf{x}; \theta),$$

et si le paramètre $\theta$ est tel que $\boldsymbol{\alpha}$ soit proche du vrai paramètre $\boldsymbol{\alpha}^\star$, nous obtenons que $\log p(\mathbf{z}, \mathbf{w}; \theta)$ est proche de 0 avec une probabilité tendant vers 1 avec le nombre d'observations. L'algorithme permet donc d'estimer la log-vraisemblance lorsque le nombre d'observations est suffisamment important.

Le formulaire pour les algorithmes *CEM* et *CEM-Bayes* est le suivant :

**Algorithme *CEM-Bayes* :**

1. Initialisation de $\boldsymbol{\theta}^{(0)}$ et $\mathbf{w}^{(0)}$.
2. Itération jusqu'à ce que $(\mathbf{z}^{(c+1)}, \mathbf{w}^{(c+1)}) = (\mathbf{z}^{(c)}, \mathbf{w}^{(c)})$ :
   (a) Étape $CE$ : de la même façon que pour l'algorithme *VEM* de la section 2.3, maximisation alternée de l'énergie libre à $\boldsymbol{\theta}^{(c)}$ fixé en prenant $t_{j\ell}^{(t=0)} = w_{j\ell}^{(c)}$ :
      i. calcul de $s_{ik}^{(t+1)}$ à $t_{j\ell}^{(t)}$ et à $\boldsymbol{\theta}^{(c)}$ fixé.
      ii. calcul de $t_{j\ell}^{(t+1)}$ à $s_{ik}^{(t+1)}$ et à $\boldsymbol{\theta}^{(c)}$ fixé.

      Obtention des probabilités $s_{ik}^{(c+1)}$ et $t_{j\ell}^{(c+1)}$, les dernières calculées.
   (b) Sélection du couple $(\mathbf{z}^{(c+1)}, \mathbf{w}^{(c+1)})$ tel que :

$$\begin{aligned} z_i^{(c+1)} &\in \underset{k=1,\ldots,g}{\operatorname{argmax}}\, s_{ik}^{(c+1)} \\ w_j^{(c+1)} &\in \underset{\ell=1,\ldots,m}{\operatorname{argmax}}\, t_{j\ell}^{(c+1)} \end{aligned}$$

   (c) Étape $M$ : calcul de $\boldsymbol{\theta}^{(c+1)}$ :

$$\pi_k^{(c+1)} = \frac{a - 1 + \sum_{i=1}^{n} z_{ik}^{(c+1)}}{g(a-1) + n}, \quad \rho_\ell^{(c+1)} = \frac{a - 1 + \sum_{j=1}^{d} w_{j\ell}^{(c+1)}}{m(a-1) + d}$$

$$\text{et } \alpha_{k\ell}^{h\ (c+1)} = \frac{b - 1 + \sum_{i=1}^{n}\sum_{j=1}^{d} z_{ik}^{(c+1)} w_{j\ell}^{(c+1)} v_{ijh}}{r\,(b-1) + \sum_{i=1}^{n}\sum_{j=1}^{d} z_{ik}^{(c+1)} w_{j\ell}^{(c+1)}}.$$

3. Obtention d'un estimateur $\widehat{\theta}^C = \boldsymbol{\theta}^{(\infty)}$.
4. Calcul des estimateurs des classes en ligne $\widehat{z}^C$ et en colonne $\widehat{w}^C$, le dernier couple $(\mathbf{z}^{(\infty)}, \mathbf{w}^{(\infty)})$ obtenu.

Une nouvelle fois, nous mettons en rouge la distinction avec la version bayésienne.

Comme pour l'algorithme *VEM*, il est préférable de ne faire qu'une seule itération de l'étape $CE$.

Pour un nombre d'initialisations fixé, Govaert et Nadif (2008) ont montré que l'algorithme *CEM* est plus rapide mais ses estimations sont plus éloignées des partitions et paramètres ayant servi aux simulations que l'algorithme *VEM* ; ceci est dû au fait qu'il discrétise l'espace des paramètres : les valeurs estimées appartiennent à une grille avec un pas régulier (par exemple de $1/n$ pour les proportions en ligne) et les estimations différentes sont au moins écartées par ce pas. En contrepartie, il se bloque plus facilement dans un maximum local et si la discrétisation est trop grossière, la valeur cible peut être assez loin d'un point de cette grille.

## 6.3 Comparaison

Pour comparer les différentes procédures, nous établissons le plan d'expérience suivant :

1. Initialisation aléatoire puis utilisation de l'échantillonneur de *Gibbs*. En utilisant les résultats obtenus :
   (a) Utilisation de l'algorithme *V-Bayes* ; le temps *elapsed* $T$ mis par cette procédure sert alors de référence.
   (b) Utilisation des autres algorithmes de la section 6.2.2.
2. Avec l'initialisation aléatoire précédente, utilisation de l'algorithme *SEM-Gibbs* en stoppant lorsque le temps $T$ est dépassé. Utilisation des algorithmes de la section 6.2.2 en prenant comme initialisation le résultat obtenu.
3. Pour chaque algorithme de la section 6.2.2, utilisation avec des initialisations au hasard relancées tant que le temps $T$ n'ait pas été dépassé.
4. Utilisation de l'algorithme $LG$ comme initialisation des algorithmes de la section 6.2.2.

Nous obtenons ainsi 16 résultats provenant des combinaisons des 4 algorithmes utilisés pour l'initialisation et des 4 algorithmes de la section 6.2.2.

### 6.3.1 Comparaison sur des données simulées

Pour la comparaison sur les données simulées, nous reprenons la configuration des sections 2.6.1, 3.5 et 4.2.4 c'est-à-dire que nous simulons des matrices avec $g = 5$ et $m = 4$ classes en ligne et colonne, des proportions des blocs de la forme :

$$\boldsymbol{\alpha} = \begin{pmatrix} \varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & \varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & \varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & \varepsilon \\ 1-\varepsilon & 1-\varepsilon & 1-\varepsilon & 1-\varepsilon \end{pmatrix}$$

où nous faisons varier $\varepsilon$ entre 0 et 0.5 pour obtenir des matrices plus ou moins simples à classifier (allant de $+$ à $+++$) suivant le protocole proposé par Lomet et al. (2012c). De plus, nous faisons varier :

— les proportions :
- proportions équilibrées :

$$\boldsymbol{\pi} = \begin{pmatrix} 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \\ 0.2 \end{pmatrix} \quad \text{et} \quad \boldsymbol{\rho} = \begin{pmatrix} 0.25 \\ 0.25 \\ 0.25 \\ 0.25 \end{pmatrix},$$

- proportions avec une progression arithmétique :

$$\boldsymbol{\pi} = \begin{pmatrix} 0.1 \\ 0.15 \\ 0.2 \\ 0.25 \\ 0.3 \end{pmatrix} \quad \text{et} \quad \boldsymbol{\rho} = \begin{pmatrix} 0.1 \\ 0.2 \\ 0.3 \\ 0.4 \end{pmatrix},$$

— le nombre $(n, d)$ de lignes et de colonnes parmi les couples suivants :

$$(100, 200) \text{ et } (200, 100).$$

Pour chaque configuration, nous simulons 500 matrices et utilisons la procédure décrite en début de section 6.3.

Les résultats sont illustrés sous forme de boxplot sur la figure 6.1 pour les proportions équilibrées et sur la figure 6.2 pour les proportions non équilibrées. En règle générale, les procédures initialisées par l'échantillonneur de *Gibbs* donnent des meilleurs résultats que celles initialisées par l'algorithme *SEM-Gibbs*. Ceci est dû d'abord aux états absorbants empêchant parfois l'algorithme *SEM-Gibbs* de faire autant d'itérations que l'échantillonneur de *Gibbs* ; mais surtout à la discrétisation de l'espace des paramètres qui demande à l'algorithme *SEM-Gibbs* de faire plus d'itérations que l'échantillonneur pour visiter un nombre suffisant de maxima locaux. Une étude plus approfondie est proposée en section 6.4.1.

Pour les mêmes raisons, l'algorithme *CEM* renvoie de moins bons résultats lorsque les matrices sont considérées comme difficiles. En revanche, dans les cas où les blocs sont bien séparés, les résultats sont assez proches de ceux des algorithmes *VEM* et *V-Bayes*.

Il y a peu de différences entre l'échantillonneur de *Gibbs* combiné avec l'algorithme *V-Bayes* et l'échantillonneur de *Gibbs* avec l'algorithme *VEM* ou l'algorithme *V-Bayes* avec des initialisations au hasard ; le premier étant à préférer dans le cas de proportions équilibrées et les seconds dans les cas de proportions déséquilibrées.

### 6.3.2 Comparaison sur données réelles

Dans cette partie, nous analysons comment se comportent les procédures sur des données réelles. Pour cela, nous avons considéré :

- Des données de blasons mérovingiens (étudiées par Leredde et Perin, 1980), matrice binaire de taille $59 \times 26$ représentant le lien entre des blasons mérovingiens (en ligne) et des caractéristiques (en colonne).
- Des données du parlement américain déjà utilisées dans le chapitre 4, matrice ternaire de taille $435 \times 16$.
- Des données de génomique, matrice binaire de taille $1937 \times 1937$ représentant les interactions inter-gènes.

Pour chacun de ces jeux de données et pour chacune des stratégies, nous faisons un plan d'expérience regardant tous les couples $(g, m)$ de classes sur une grille allant de 2 à $g_{\max}$ pour les lignes et de 2 à $m_{\max}$ pour les colonnes ($g_{\max}$ et $m_{\max}$ dépendant des données). Comme précédemment, c'est le temps de l'échantillonneur de *Gibbs* couplé avec l'algorithme *V-Bayes* qui sert de référence pour chaque couple $(g, m)$. Enfin, nous utilisons le critère $ICL_{(4,1)}$ pour la sélection de modèle.

#### Blasons mérovingiens

La matrice des données sur les blasons mérovingiens comprend 59 lignes représentant les blasons et 26 colonnes représentant leurs caractéristiques (voir Leredde et Perin, 1980, pour une explication détaillée). Cette matrice est très petite et possède, à peu près, le même nombre de 1 sur chaque ligne.

Ici, nous avons choisi $g_{\max} = m_{\max} = 26$ ; ces chiffres dépassent les conditions suffisantes d'identifiabilité de Keribin et al. Nous affichons sur la figure 6.3 le nombre de classes et la représentation obtenue pour chaque stratégie.

Les stratégies obtenant les meilleurs critères $ICL_{(4,1)}$ sont celles couplant les initialisations aléatoires avec l'algorithme *V-Bayes*, *CEM* ou *CEM-Bayes*. Ensuite, viennent les initialisations par *SEM-Gibbs* puis les initialisations par l'échantillonneur de *Gibbs*. Enfin, l'algorithme *LG* sous-estime le nombre

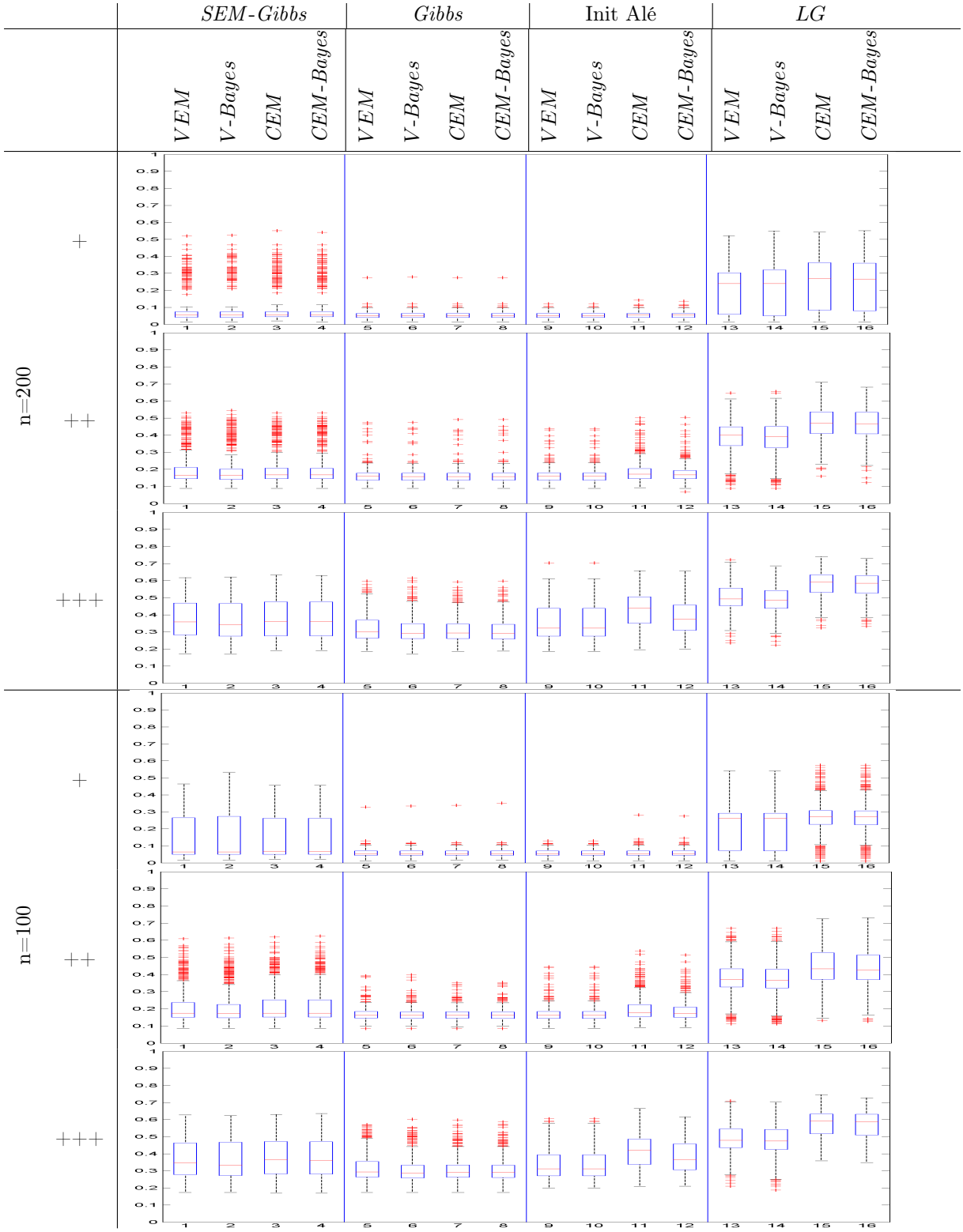



FIGURE 6.1 – Boxplot des erreurs des combinaisons (initialisations pour les colonnes globales combinées avec les algorithmes de la section 6.2.2 pour les subdivisions) pour les proportions équilibrées suivant la taille des matrices (lignes globales) et la difficulté (subdivision).

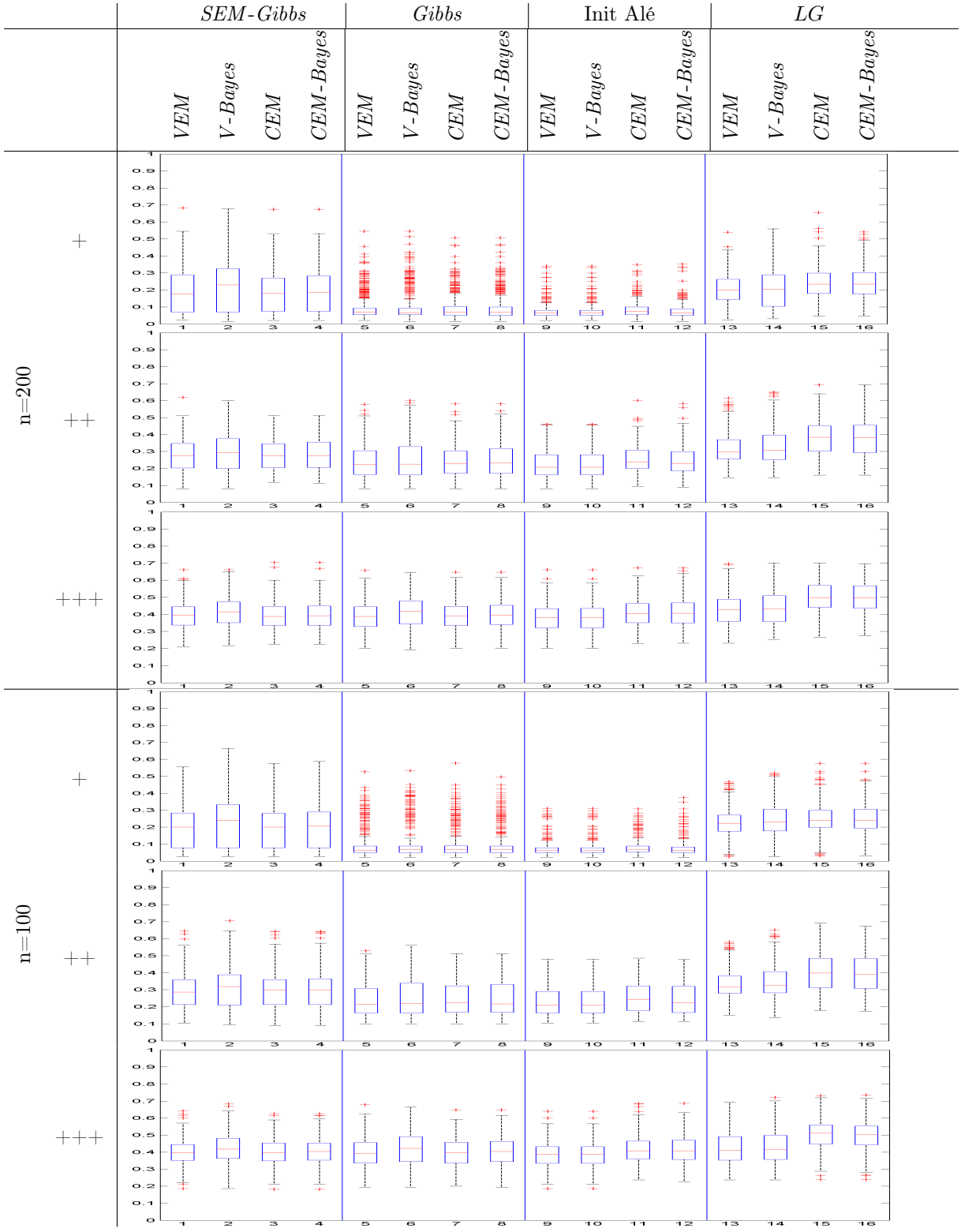



FIGURE 6.2 – Boxplot des erreurs des combinaisons (initialisations pour les colonnes globales combinées avec les algorithmes de la section 6.2.2 pour les subdivisions) pour les proportions non équilibrées suivant la taille des matrices (lignes globales) et la difficulté (subdivision).

de classes de façon prévisible ; en partie à cause du nombre de cases par ligne qui est quasiment le même pour toutes.

Les nombres de lignes et de colonnes étant très petits, les valeurs des effectifs de chaque classe en ligne et en colonne sont faibles et les lois utilisées lors des simulations de l'échantillonneur de *Gibbs* ont une grande variance ; les trajectoires de l'échantillonneur de *Gibbs* sont donc certainement très variables. En revanche, comme l'algorithme *SEM-Gibbs* discrétise l'espace des paramètres, il est plus stable.

Les différences entre la partition sélectionnée par l'échantillonneur de *Gibbs* avec l'algorithme *V-Bayes* et celle obtenue par l'initialisation aléatoire avec l'algorithme *V-Bayes* (optimale pour le critère $ICL_{(4,1)}$) se situent dans les classes 2 et 3 pour les lignes et les classes centrales pour les colonnes. La partition de l'initialisation aléatoire a créé plus de groupes (donc plus petits) pour les colonnes ce qui a pour conséquence d'avoir des blocs plus contrastés. L'échantillonneur de *Gibbs* avec l'algorithme *VEM* renvoie la même partition en ligne que la partition retenue avec le plus fort critère $ICL_{(4,1)}$, par contre, il crée une colonne à une classe différente, ce occasionne un critère plus faible.

L'algorithme *SEM-Gibbs* combiné avec l'algorithme *V-Bayes* renvoie presque la même partition que celle de l'initialisation aléatoire et de l'algorithme *V-Bayes* (à une colonne près) sauf qu'il subdivise l'une des classes. Les blocs sont ainsi plus contrastés mais le critère $ICL_{(4,1)}$ pénalise cette augmentation du nombres de blocs.
Les partitions obtenues par les initialisations de *LG* sont des sur-partitions de toutes les autres ; c'est-à-dire avec les mêmes classes mais en en concaténant plus.

D'un point de vue interprétation des données, nous retrouvons l'évolution technique au cours du temps. Pour les lignes, les classes sont les mêmes que celles obtenues par Leredde et Perin (1980). En revanche, pour les colonnes, les classes de l'article ont été subdivisée par nos procédures.

**Parlement américain**

Pour comparer les algorithmes sur des données ternaires, nous avons réutilisé les données du parlement américain décrites dans la section 4.2.4. Nous rappelons que le jeu de données représente le vote de 435 parlementaires américains, membres de la 98 ème *Chambre des représentants* (*United States House of Representatives*), répartis en 168 républicains et 267 démocrates et s'exprimant sur 16 lois (éducation, sécurité, politique internationale...) suivant 3 niveaux de réponse ("pour", "contre" et "abstention"). Pour les analyser, nous avons choisi $g_{\max} = m_{\max} = 16$. Pour la représentation des résultats, les votes "pour" sont en bleu, les "contre" en rouge et les abstentions en blanc.

La figure 6.4 montre que la meilleure stratégie est à nouveau l'initialisation aléatoire couplée avec l'algorithme *V-Bayes*. Juste après, vient la combinaisons *Gibbs*+*V-Bayes*. En règle générale, les initialisations aléatoires et l'échantillonneur de *Gibbs* donnent des résultats proches. Notons que les critères sont moins bons que ceux de la section 4.2.4 où nous utilisions pour chaque couple $(g, m)$ trois fois la combinaison *Gibbs*+*V-Bayes* sur des initialisations différentes.
En revanche, l'algorithme *SEM-Gibbs* semble ne pas avoir souvent convergé.
Enfin, l'algorithme *LG* donne des résultats plus concordants que pour les blasons mérovingiens. Les données ternaires donnent probablement plus d'informations pour cet algorithme.

D'un point de vue interprétation des données, la loi comportant le plus grand taux d'abstentions a été isolée dans chacun des modèles sélectionnés. De même, les trois abstentionnistes ont systématiquement été mis ensemble. Enfin, en règle générale, ce sont les gros blocs rouges et bleus qui font la différence pour les classifications croisées (ce qui manque à l'algorithme *LG*).

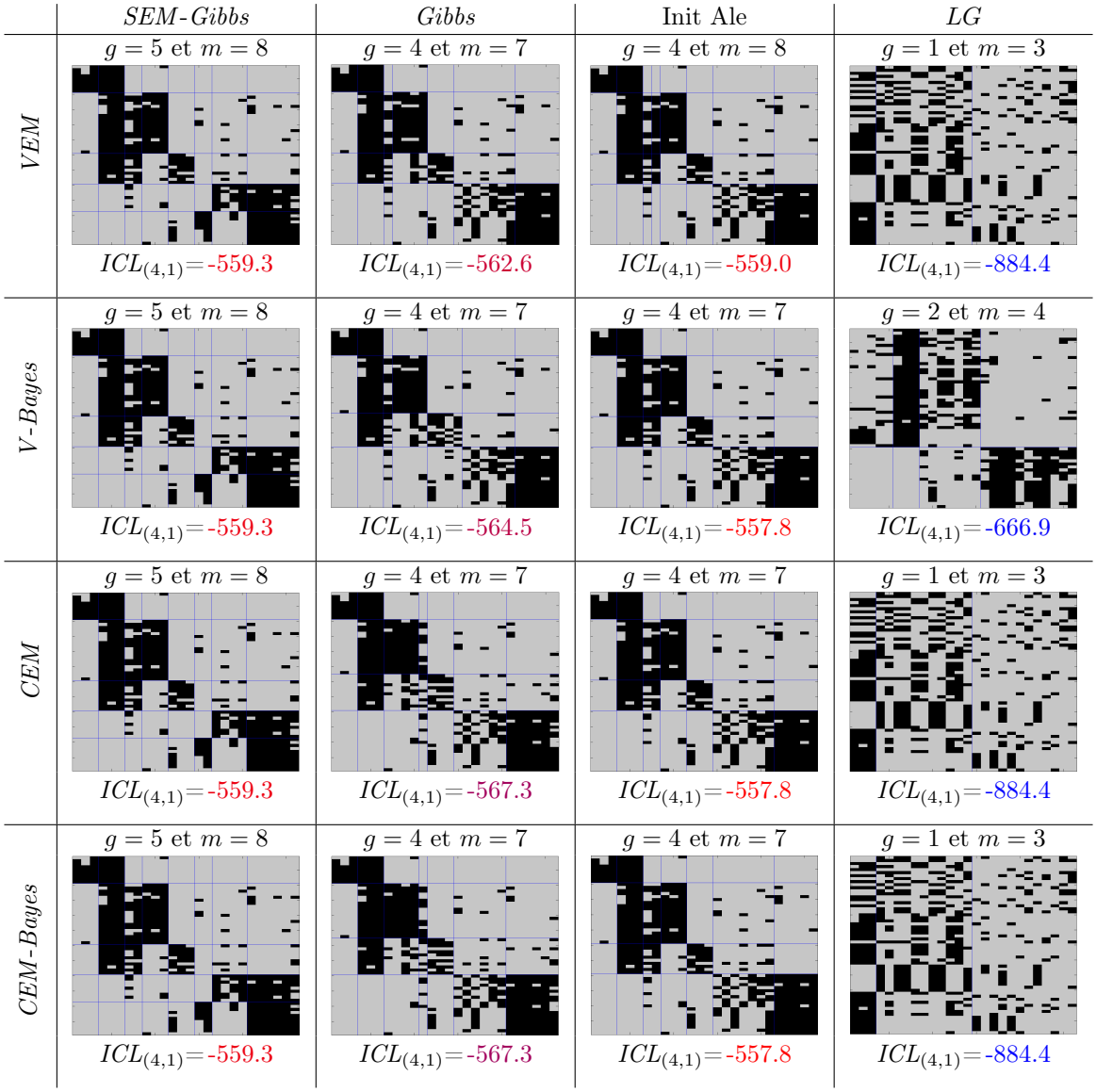



FIGURE 6.3 – Résultat des différentes stratégies pour les données sur les blasons mérovingiens suivant les initialisations (en colonne) et les algorithmes de la section 6.2.2 (en ligne).

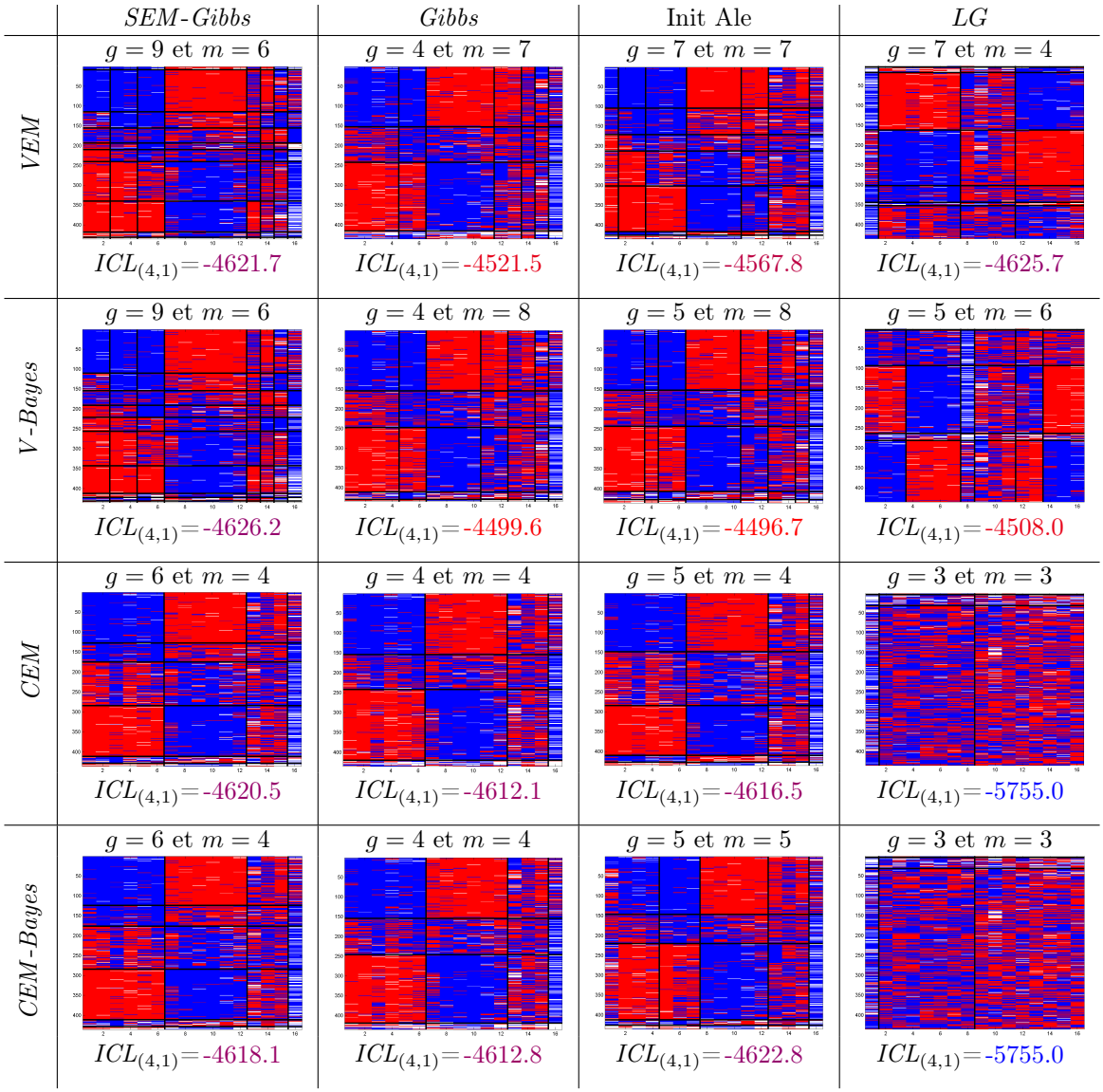



FIGURE 6.4 – Résultat des différentes stratégies pour les données sur les votes du parlement américain suivant les initialisations (en colonne) et les algorithmes de la section 6.2.2 (en ligne). Le code couleur est : "pour" en bleu, "contre" en rouge et "abstention" en blanc.

**Interactions entre les gènes de l'*Arabidopsis thaliana***

Les dernières données étudiées proviennent du mémoire de master de Vasseur (2013) et portent sur les interactions entre les 1937 facteurs de transciptions de la plante *Arabidopsis thaliana* ; les ensembles des observations et des variables représentent donc les mêmes individus et la matrice ne respecte pas toutes les conditions de notre modèle car toutes les cases de la diagonale ont la même valeur. Toutefois, la recherche de groupes différents pour les lignes et les colonnes est pertinente car elle permet de trouver des groupes de gènes influençant fortement ou non et des groupes de gènes fortement influencés ou non. Ici, nous avons choisi $g_{\max} = m_{\max} = 30$.

Pour une meilleure lisibilité, nous présentons sur la figure 6.5 les résultats résumés où les blocs $(k, \ell)$ sont plus ou moins noirs si la valeur $\alpha_{k\ell}$ estimée est plus ou moins proche de 1. Le meilleur critère $ICL_{(4,1)}$ est obtenu pour l'initialisation aléatoire avec l'algorithme *V-Bayes* suivi de près par l'échantillonneur de *Gibbs* couplé avec les algorithmes *VEM*, *CEM* et *CEM-Bayes*.

En revanche, les combinaisons utilisant une autre initialisation que l'échantillonneur de *Gibbs* et finissant par l'algorithme *CEM* ou *CEM-Bayes* renvoient des modèles avec des critères plus faibles en partie à cause de classes en colonne avec des effectifs très petits.

Enfin, les combinaisons utilisant l'algorithme *LG* minimisent à nouveau le nombre de classes.

D'un point de vue interprétation des données, nous pouvons voir en haut des lignes et à gauche des colonnes des groupes de gènes interférant essentiellement entre eux. Des groupes de gènes très influents (à droite des colonnes) et des groupes de gènes très influencés (en bas des lignes) ont également été mis en évidence.

## 6.4 Comparaison deux à deux

Dans cette partie, nous comparons les algorithmes aux comportements similaires, à savoir :
— l'algorithme *SEM-Gibbs* et l'échantillonneur de *Gibbs*,
— les algorithme *V-Bayes* et *CEM*.

### 6.4.1 Comparaison entre l'échantillonneur de *Gibbs* et *SEM-Gibbs*

Pour mieux comprendre la différence entre l'échantillonneur de *Gibbs* et l'algorithme *SEM-Gibbs*, nous avons regardé de près leurs résultats pour les expériences de la section 6.3.1. À temps d'exécution donné, l'algorithme *SEM-Gibbs* disposait de plus d'itérations que l'échantillonneur de *Gibbs* en règle générale (cela reste de l'ordre de 1% en plus en moyenne et peut être dû au fait que le temps utilisé comme référence était celui de l'échantillonneur de *Gibbs* combiné avec l'algorithme *V-Bayes* et pas uniquement le premier). L'algorithme *SEM-Gibbs* ne s'est donc pas bloqué dans une situation où il devait réitérer un grand nombre de fois l'étape $SE$ pour obtenir des partitions sans classes vides.

Nous avons simulé de nouvelles trajectoires des estimations des $\theta^{(c)}$ de l'échantillonneur de *Gibbs* et de l'algorithme *SEM-Gibbs* à partir d'une même initialisation pour un cas de difficulté moyenne (++) avec des proportions équilibrées et un nombre de lignes $n = 100$. Comme l'algorithme *SEM-Gibbs* n'estimait pas correctement les proportions, nous l'avons laissé exécuter 7 500 itérations contre les 5 000 pour l'échantillonneur de *Gibbs*. Les trajectoires sont représentées sur la figure 6.6.

Dans les deux cas, l'initialisation force les algorithmes à donner des valeurs fortes pour la classe violette au début. En revanche, à un moment, l'échantillonneur de *Gibbs* va diminuer cette valeur alors que

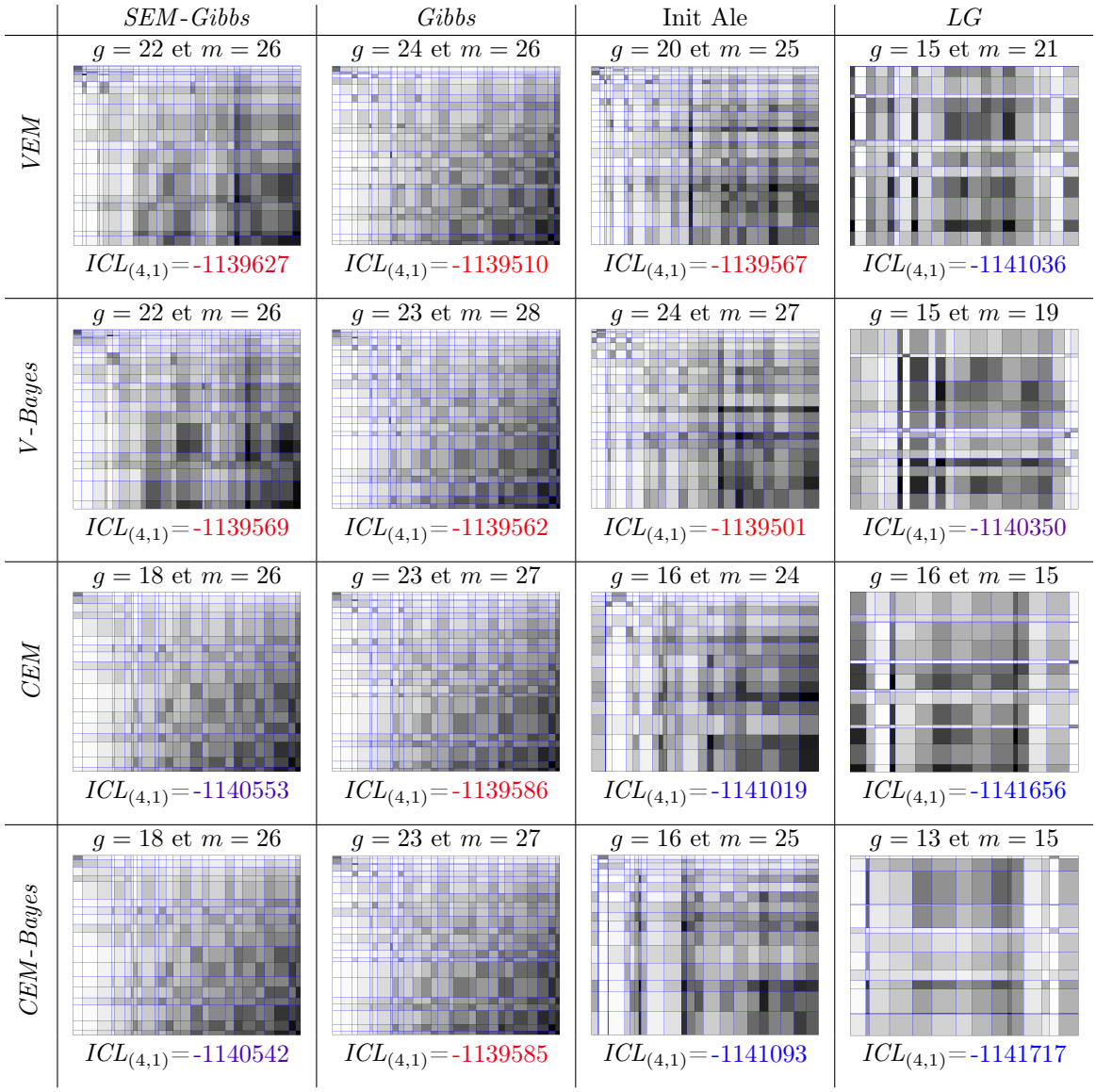



FIGURE 6.5 – Résultat des différentes stratégies pour les données d'interactions entre les gènes de l'*Arabidopsis thaliana* suivant les initialisations (en colonne) et les algorithmes de la section 6.2.2 (en ligne).

l'algorithme *SEM-Gibbs* va toujours la garder très haute ; au lieu d'obtenir des proportions équilibrées. La discrétisation de l'espace des paramètres semble gêner l'algorithme *SEM-Gibbs* pour sortir de ce maximum local.

### 6.4.2 Comparaison entre les algorithmes *CEM* et *V-Bayes*

La deuxième comparaison porte sur les algorithmes *CEM* et *V-Bayes* qui font tous les deux une approximation de la log-vraisemblance. Comme nous l'avons remarqué sur les données réelles, l'algorithme *CEM* renvoie, en règle générale, des modèles avec des valeurs pour les critères moins bonnes que celles de l'algorithme *V-Bayes* ; mais est-ce parce que toutes les estimations sont moins bonnes, ou simplement autour des vraies valeurs de $g$ et $m$ ?

Pour répondre à cette question, nous avons repris la matrice des interactions entre les gènes de l'*Arabidopsis thaliana* de la section 6.3.2 et avons utilisé pour chaque couple de classes $(g, m)$ les algorithmes *CEM* et *V-Bayes* pour des initialisations aléatoires relancées pendant 30 minutes (temps *elapsed*). Nous avons ensuite regardé les critères $ICL_{(4,1)}$ associés à chaque partition renvoyée.

Le tableau 6.1 représente pour chaque couple l'algorithme ayant eu le meilleur critère $ICL_{(4,1)}$. L'algorithme *CEM* possède des valeurs plus élevées pour le critère $ICL_{(4,1)}$ pour des petits nombres $g$ ou $m$. Pourtant, l'algorithme n'est relancé en moyenne que 2 à 3 fois plus que l'algorithme *V-Bayes* ; pour comparaison, l'algorithme a été relancé plus de 20 fois plus pour le couple $(28, 29)$. Le nombre de maxima locaux semble donc beaucoup plus importants pour l'algorithme *CEM* que pour l'algorithme *V-Bayes* lorsque le nombre de classes augmente.

## 6.5 Discussion générale

Sur les données simulées, ce sont souvent les initialisations avec l'échantillonneur de *Gibbs* qui ont renvoyé plus souvent les partitions ayant servi à la simulation suivies par les initialisations aléatoires. En revanche, sur les données réelles, l'échantillonneur de *Gibbs* semble ne pas avoir correctement estimé la zone où se trouve les vrais paramètres ; l'algorithme *V-Bayes* a parfois compensé ce problème.

Les algorithmes bayésiens, en règle générale, favorisent les proportions équilibrées. Si nous savons qu'elles ne le sont pas, il serait préférable de prendre des valeurs de $a$ plus proches de 1. Mais la question peut aussi se poser de la pertinence d'une classe trop petite.

L'algorithme *SEM-Gibbs* met souvent plus de temps à converger que l'échantillonneur de *Gibbs*. Toutefois, la discrétisation des estimations des paramètres donne une plus grande stabilité, ceci a été important pour la matrice des blasons mérovingiens. Enfin, les résultats sont relativement similaires sur les grosses matrices de données ; la discrétisation semble moins influencer.

Les algorithmes *CEM* et *CEM-Bayes* favorisent souvent des modèles plus parcimonieux contenant des blocs très discriminants et s'accommodant de ceux qui ne sont pas homogènes. Nous pouvons remarquer sur les données des interactions entre les gènes de l'*Arabidopsis thaliana* que ces algorithmes sont proches des autres ce qui est en accord avec le besoin d'avoir beaucoup d'observations pour estimer correctement les paramètres et partitions. De plus, ils donnent de meilleurs résultats que les algorithmes variationnels lorsque le nombre de classes demandé est assez faible.

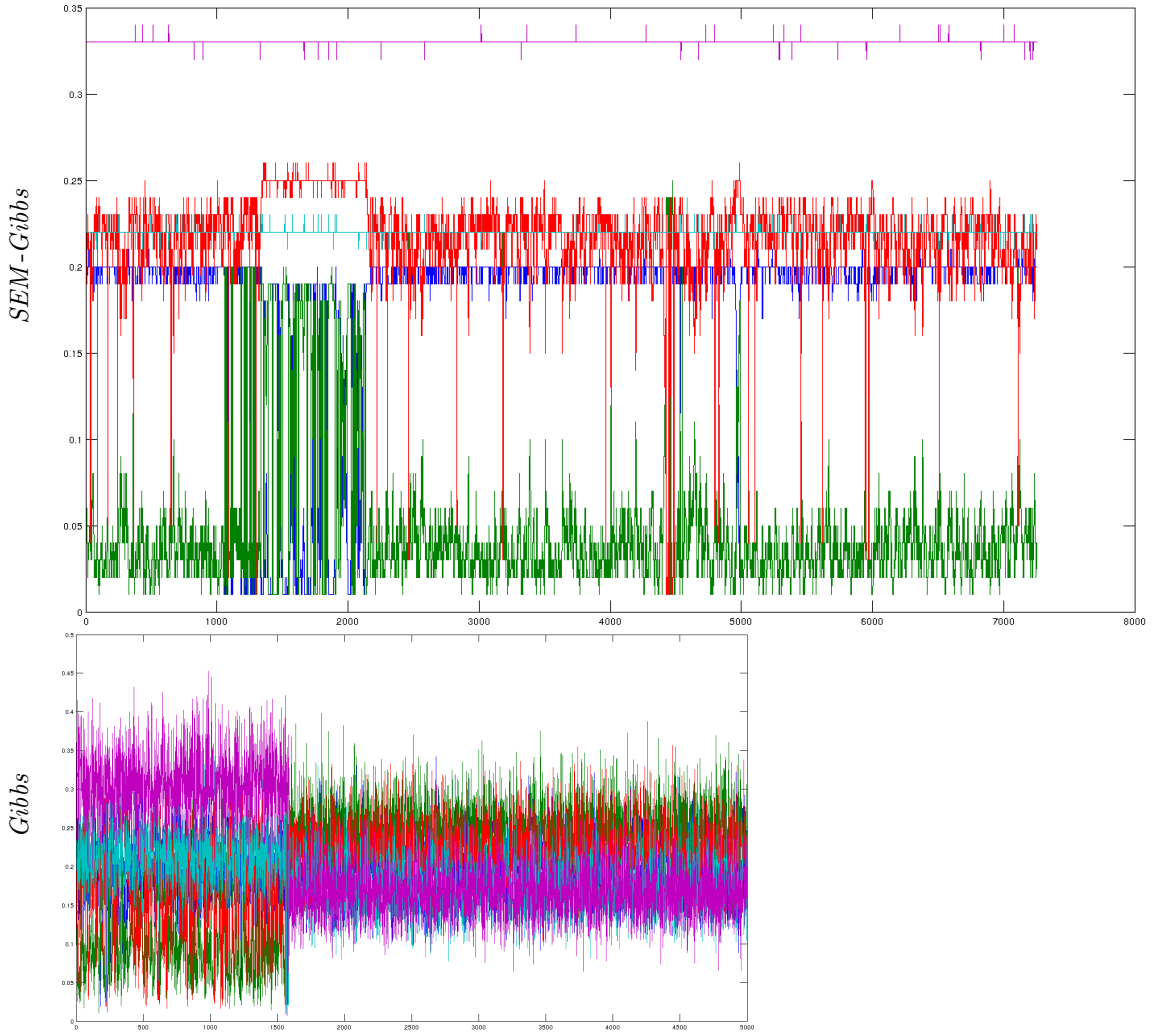



FIGURE 6.6 – Evolution des trajectoires de $\theta^{(c)}$ pour l'algorithme *SEM-Gibbs* (en haut) et l'échantillonneur de *Gibbs* (en bas) pour une matrice de difficulté moyenne avec des initialisations équilibrées et $n = 100$.

| g \ m | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | C | C | C | C | C | C | C | C | C | C | V | V | V | V |
| 3 | C | C | C | C | C | C | C | C | C | V | V | V | V | V |
| 4 | C | C | C | C | C | C | V | C | C | V | V | V | V | V |
| 5 | C | C | V | C | C | C | C | C | C | V | V | V | V | V |
| 6 | C | C | C | V | C | V | C | V | V | V | V | V | V | V |
| 7 | C | C | C | C | C | V | V | V | V | V | V | V | V | V |
| 8 | C | C | V | C | C | V | C | V | V | V | V | V | V | V |
| 9 | C | C | C | C | V | V | V | C | V | V | V | V | V | V |
| 10 | C | C | C | C | C | V | V | V | V | V | V | V | V | V |
| 11 | V | C | C | V | V | V | V | V | V | V | V | V | V | V |
| 12 | C | V | V | V | V | V | V | V | V | V | V | V | V | V |
| 13 | C | V | V | V | V | V | V | V | V | V | V | V | V | V |
| 14 | V | C | V | C | V | V | V | V | V | V | V | V | V | V |
| 15 | C | C | V | V | V | V | V | V | V | V | V | V | V | V |
| 16 | C | C | V | V | V | V | V | V | V | V | V | V | V | V |
| 17 | C | C | V | V | V | V | V | V | V | V | V | V | V | V |
| 18 | C | C | V | V | V | V | V | V | V | V | V | V | V | V |
| 19 | C | V | V | V | V | V | V | V | V | V | V | V | V | V |
| 20 | C | V | V | V | V | V | V | V | V | V | V | V | V | V |
| 21 | C | C | V | C | V | V | V | V | V | V | V | V | V | V |
| 22 | C | C | C | C | V | V | V | V | V | V | V | V | V | V |
| 23 | C | V | V | V | V | V | V | V | V | V | V | V | V | V |
| 24 | C | C | V | V | V | V | V | V | V | V | V | V | V | V |
| 25 | C | C | C | V | V | V | V | V | V | V | V | V | V | V |
| 26 | C | C | V | C | V | V | V | V | V | V | V | V | V | V |
| 27 | C | C | V | C | V | V | V | V | V | V | V | V | V | V |
| 28 | C | C | C | C | V | V | V | V | V | V | V | V | V | V |
| 29 | C | C | C | V | V | V | V | V | V | V | V | V | V | V |
| 30 | C | C | C | C | V | V | V | V | V | V | V | V | V | V |
| 31 | C | C | C | C | V | V | V | V | V | V | V | V | V | V |

TABLE 6.1 – Représentation de l'algorithme ayant renvoyé le meilleur critère $ICL_{(4,1)}$ pour chaque couple $(g, m)$ : C pour *CEM* et V pour *V-Bayes*.

# Chapitre 7

# Conclusions, méthodologie et perspectives

*"L'urgent est fait. L'impossible est en cours. Pour les miracles, prévoir un délai."*
Anonyme

**Sommaire**


**7.1 Conclusion** . . . . . **179**
**7.2 Vers une méthodologie...** . . . . . **180**
**7.3 Perspectives** . . . . . **180**


## 7.1 Conclusion

Durant cette thèse, nous avons effectué un certain nombre d'avancées dans l'étude du modèle des blocs latents.

**Identifiabilité :** nous avons obtenu des conditions suffisantes d'identifiabilité englobant presque tout l'espace des paramètres sauf un ensemble de mesure de Lebesgue nulle (sous les conditions que $n \geq 2m-1$ et $d \geq 2g-1$) et proposé une façon de contourner le problème du *label switching*.

**Dégénérescence des classes :** nous avons montré empiriquement que l'échantillonneur de *Gibbs* associé à l'algorithme *V-Bayes* avec $(a, b) = (4, 1)$ résout en grande partie le problème de dégénérescence des classes rencontré par l'algorithme *SEM-Gibbs* combiné avec l'algorithme *VEM*.

**Estimation :** pour l'estimation des labels et des paramètres, nous préconisons l'utilisation de l'échantillonneur de *Gibbs* associé à l'algorithme *V-Bayes* avec $(a, b) = (4, 1)$ car renvoyant les meilleurs résultats de classification sur des données simulées parmi les procédures étudiées. De plus, le critère d'arrêt que nous proposons permet d'obtenir plus vite des résultats légèrement meilleurs que celui utilisant la statistique de Brooks-Gelman dans sa version basique.
Toutefois, l'algorithme *SEM-Gibbs* peut être préféré dans le cas de petites matrices car plus stable que l'échantillonneur de *Gibbs*.

**Sélection de modèle :** nous avons montré que le critère $ICL_{(a,b)}$ fournit de bonnes estimations quel que soit le nombre d'observations. Nous avons également conjecturé que les estimations du nombre de

classes obtenues à partir de ce critère sont consistantes.

## 7.2 Vers une méthodologie...

Au vu des résultats obtenus, nous préconisons l'utilisation du critère $ICL_{(a,b)}$ pour sélectionner le nombre de classes et les matrices $(\mathbf{z}, \mathbf{w})$.

**Quelles partitions $(\widehat{\mathbf{z}}, \widehat{\mathbf{w}})$ choisir ?** Pour le choix des partitions associées au couple $(g, m)$, nous préconisons de prendre le couple fourni par la combinaison de l'échantillonneur de *Gibbs* et de l'algorithme *V-Bayes* (voir la section 4.2.1). Ce choix a été utilisé pour les simulations de la section 4.2.4 renvoyant de bons résultats.

**Quels couples $(g, m)$ tester ?** Dans l'idéal, il serait préférable de tester tous les couples possibles mais en pratique et dans le cas de grandes matrices, ceci est très coûteux en temps.

Pour atténuer ce problème, nous préconisons d'utiliser d'abord l'algorithme $LG$ pour chaque couple $(g, m)$ et de regarder quel couple $\left(\widehat{g}^{LG}, \widehat{m}^{LG}\right)$ est sélectionné : les résultats des chapitres 5 et 6 montrent que cette procédure fournit des valeurs plus petites que celles choisies finalement. Dans un second temps, il ne reste plus qu'à tester tous les couples $(g, m)$ tels que $g \geq \widehat{g}^{LG}$ et $m \geq \widehat{m}^{LG}$ en utilisant la procédure avec l'échantillonneur de *Gibbs* et l'algorithme *V-Bayes*.

## 7.3 Perspectives

Malgré ses cinquante ans d'histoire, beaucoup de questions restent encore ouvertes.

**Comportement asymptotique des estimateurs :** notre résultat sur l'identifiabilité ouvre la voie à l'étude du comportement de l'estimateur du maximum de vraisemblance ; les résultats de Celisse et al. (2012) dans le cadre du $SBM$ pouvant servir de point de départ.

**Protocole de simulations des données :** il serait intéressant de continuer le travail effectué par Lomet (2012), notamment d'essayer de comprendre l'influence de la borne maximale de l'erreur $e_{(n,g)\times(d,m)}$ obtenue en section 1.3.2.5 sur son protocole et de continuer la réflexion sur la façon de caractériser une bonne estimation. De plus, il serait utile d'augmenter le nombre de jeux de données simulés mis à disposition notamment en incluant des matrices dissymétriques dans le nombre de lignes et de colonnes ou de classes en ligne et en colonne.

**Échantillonneur de *Gibbs* :** pour améliorer les procédures que nous proposons, il serait utile de réfléchir à un critère d'arrêt plus rapide pour stopper l'échantillonneur de *Gibbs* ; soit en continuant l'étude basée sur la statistique de Brooks-Gelman, soit en réfléchissant à un nouveau critère peut-être basé plutôt sur les matrices $(\mathbf{z}, \mathbf{w})$ que sur les paramètres $\boldsymbol{\theta}$.

**Méthodologie :** enfin, il serait intéressant de trouver un moyen de diminuer le nombre de couples $(g, m)$ à étudier pour l'utilisation du critère $ICL_{(a,b)}$.

# Annexe A

# Fondements

*"On ne peut bâtir sur du sable."*
Moses Isegawa

**Sommaire**


**A.1 Introduction . . . . . 181**
**A.2 Modèles de mélange . . . . . 181**
A.2.1 Inférence statistique . . . . . 181
A.2.2 Modèle de mélange . . . . . 183
**A.3 Algorithme Expectation Maximisation . . . . . 184**
A.3.1 Vraisemblance d'un modèle de mélange . . . . . 184
A.3.2 Algorithme Expectation Maximisation . . . . . 184
A.3.3 Algorithme Stochastique Expectation Maximisation . . . . . 185
A.3.4 Différences de comportements entre les deux algorithmes . . . . . 186
A.3.5 Application . . . . . 186


## A.1 Introduction

Les modèles de mélange et l'algorithme $EM$ étant les bases de cette thèse, nous faisons un rappel de ces deux notions.

## A.2 Modèles de mélange

Pour expliquer les modèles de mélange, nous nous appuyons sur un exemple de balistique.

### A.2.1 Inférence statistique

Nous supposons sur la figure A.1 qu'un tireur a choisi l'une des deux cibles et a tiré jusqu'à ce que son chargeur soit vide. Le but est de trouver la cible visée.

Nous pouvons supposer qu'un tireur moyen vise un point et que les impacts se trouvent plus ou moins éloignés suivant la qualité du tireur sans favoriser de direction. Nous supposons que la loi des impacts est une loi gaussienne en deux dimensions centrée sur l'endroit visé et de variance $\sigma^2 I_2$ :

$$X \sim \mathcal{N}\left(\mu = \begin{pmatrix}\mu_1 \\ \mu_2\end{pmatrix}, \sigma^2 I_2\right).$$

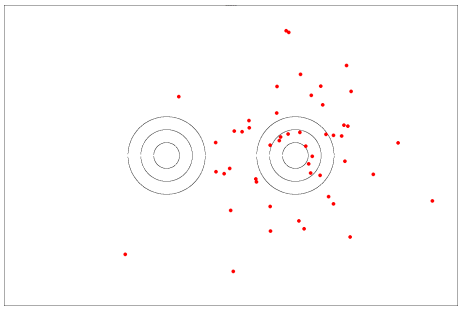


FIGURE A.1 – Impact de balles produit par les tirs d'un unique tireur visant toujours la même cible.

En supposant les impacts indépendants, nous calculons la logvraisemblance :

$$\begin{aligned}
\log p(X_1, \ldots, X_n; \mu, \sigma^2) &= \log \left( \prod_{i=1}^{n} \frac{1}{2\pi\sigma^2} e^{-\frac{1}{2\sigma^2} \|X_i - \mu\|_2^2} \right) \\
&= \sum_{i=1}^{n} \left( -\log\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2} \|X_i - \mu\|_2^2 \right) \\
&= -n \log 2\pi - n \log \sigma^2 - \frac{1}{2\sigma^2} \sum_{i=1}^{n} \|X_i - \mu\|_2^2
\end{aligned}$$

L'idée est alors de maximiser cette fonction. Comme les observations sont fixées, nous cherchons à maximiser en $\mu$ et $\sigma^2$. En dérivant, nous obtenons :

$$\begin{aligned}
\widehat{\mu} &= \frac{1}{n} \sum_{i=1}^{n} X_i \\
\widehat{\sigma^2} &= \frac{1}{2n} \sum_{i=1}^{n} \|X_i - \widehat{\mu}\|_2^2
\end{aligned}$$

Sur la figure A.2 sont représentées la moyenne et une région de confiance à 95% centrée sur cette moyenne.

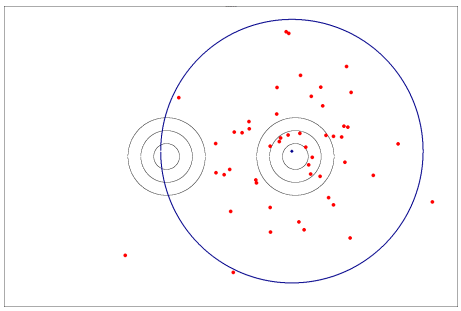


FIGURE A.2 – Le point bleu montre la moyenne et le cercle une région de confiance à 95%.

La moyenne se situe dans la cible de droite qui est totalement dans la région de confiance en comparaison à la cible de gauche. Le tireur a très certainement visé la cible de droite.

### A.2.2 Modèle de mélange

Pour expliquer le principe du modèle de mélange, nous avons besoin de 3 cibles et des impacts de la figure A.3. Si nous supposons qu'il n'y a qu'un seul tireur, la moyenne se trouve sur la cible du milieu mais la région de confiance à 50% contient peu d'impacts situés essentiellement sur les bords plutôt qu'au centre ; il semble également que l'axe des abscisses soit privilégié. Ceci est en contradiction avec une loi gaussienne sphérique car les impacts devraient plutôt se situer autour de la moyenne sans être dans une direction privilégiée.

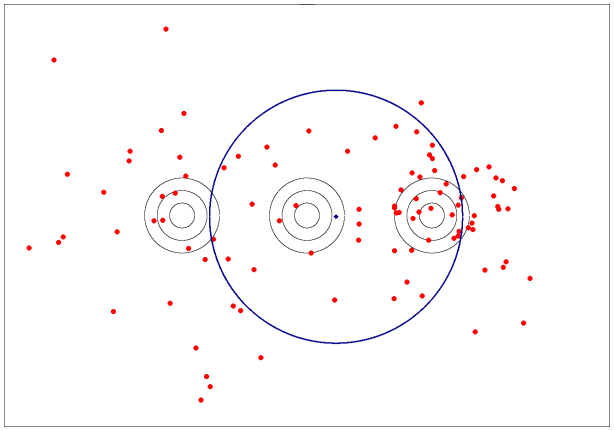


FIGURE A.3 – Trois cibles en noir avec les impacts en rouge. Si on suppose qu'il y a un seul tireur, le point bleu correspond à l'endroit visé et le cercle bleu à une région de confiance à 50%.

En regardant les impacts, nous pensons alors à deux tireurs visant chacun une cible. Si nous savons à quels tireurs appartiennent quels impacts, la question se divise en deux sous problèmes se résolvant comme dans la section A.2.1. Mais nous ignorons cette information ou même si chaque tireur a tiré autant de balles que l'autre.

Le principe du modèle de mélange est de supposer que les variables $X_i$ appartiennent à $K$ groupes possédant chacun une densité $\varphi_k(x;\boldsymbol{\theta}_k)$. En notant $Z$ la matrice aléatoire de taille $n \times K$ telle que $Z_{ik} = 1$ si et seulement si la variable $X_i$ appartient au groupe $k$, nous avons :

$$\mathbb{P}(X_i = x_i | Z_{ik} = 1; \boldsymbol{\theta}) = \varphi_k(x_i; \boldsymbol{\theta}_k).$$

De plus, nous supposons que la probabilité d'appartenir à un groupe $k$ est la même pour toutes les variables aléatoires, souvent notée $\pi_k$. Finalement, à l'aide d'une formule des probabilités totales, nous obtenons :

$$\mathbb{P}(X_i = x_i; \boldsymbol{\theta}) = \sum_{k=1}^{K} \mathbb{P}(X_i = x_i | Z_{ik} = 1; \boldsymbol{\theta}) \mathbb{P}(Z_{ik} = 1; \boldsymbol{\theta}) = \sum_{k=1}^{K} \pi_k \varphi_k(x_i; \boldsymbol{\theta}_k).$$

*Exemple* A.2.1. Dans notre exemple, nous supposons qu'il y a deux tireurs (donc $K = 2$) visant deux points différents ($\mu^1 \neq \mu^2$) et nous n'avons aucune raison de penser qu'ils ont la même habilité (soit $\sigma_1 \neq \sigma_2$).

## A.3 Algorithme Expectation Maximisation

Dans cette partie, nous rappelons que maximiser la vraisemblance est compliqué et comment l'algorithme Expectation Maximisation proposé par Dempster et al. (1977) permet de contourner ce problème.

### A.3.1 Vraisemblance d'un modèle de mélange

À l'aide du paragraphe précédent, nous pouvons calculer la logvraisemblance d'un modèle de mélange en supposant l'indépendance des $X_i$ :

$$\begin{aligned}\log p(X_1,\dots,X_n;\boldsymbol{\theta}) &= \log\left(\prod_{i=1}^{n}\mathbb{P}(X_i;\boldsymbol{\theta})\right)\\ &= \sum_{i=1}^{n}\log\left(\sum_{k=1}^{K}\pi_k\varphi_k(X_i;\boldsymbol{\theta}_k)\right)\end{aligned}$$

Maximiser cette formule est difficile dès que $K \geq 2$ puisque même en dérivant, nous obtenons des sommes de fractions.

### A.3.2 Algorithme Expectation Maximisation

L'idée proposée par Dempster et al. (1977) est de maximiser la log-vraisemblance en tirant parti de la structure à données manquantes du problème. Pour cela, ils décomposent la log-vraisemblance comme la différence de l'espérance conditionnelle connaissant un paramètre $\boldsymbol{\theta}^{(c)}$ et les observations de la log-vraisemblance complétée et d'une fonction en $\boldsymbol{\theta}$ :

$$\begin{aligned}L(\boldsymbol{\theta}) &= \log p(X;\boldsymbol{\theta})\\ &= \log p(X;\boldsymbol{\theta})\sum_{z\in\mathcal{Z}}p(z|X;\boldsymbol{\theta}^{(c)})\\ &= \sum_{z\in\mathcal{Z}}\log p(X;\boldsymbol{\theta})p(z|X;\boldsymbol{\theta}^{(c)})\\ &= \sum_{z\in\mathcal{Z}}\left[\log\left(\frac{p(X,z;\boldsymbol{\theta})}{p(z|X;\boldsymbol{\theta})}\right)p(z|X;\boldsymbol{\theta}^{(c)})\right]\\ &= \underbrace{\sum_{z\in\mathcal{Z}}\left[\log\left(p(X,z;\boldsymbol{\theta})\right)p(z|X;\boldsymbol{\theta}^{(c)})\right]}_{=Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(c)})} - \underbrace{\sum_{z\in\mathcal{Z}}\left[\log\left(p(z|X;\boldsymbol{\theta})\right)p(z|X;\boldsymbol{\theta}^{(c)})\right]}_{=H(\boldsymbol{\theta}|\boldsymbol{\theta}^{(c)})}\end{aligned}$$

Or, nous avons la proposition suivante :

**Proposition A.3.1.** *Pour tout* $\boldsymbol{\theta}\in\Theta$, $H(\boldsymbol{\theta}|\boldsymbol{\theta}^{(c)}) - H(\boldsymbol{\theta}^{(c)}|\boldsymbol{\theta}^{(c)}) \geq 0$.

*Démonstration* C'est une application de l'inégalité de Jensen :

$$\begin{aligned}
H(\boldsymbol{\theta}|\boldsymbol{\theta}^{(c)}) - H(\boldsymbol{\theta}^{(c)}|\boldsymbol{\theta}^{(c)}) &= -\sum_{z\in\mathcal{Z}}\left[\log\left(p(z|X;\boldsymbol{\theta})\right)p(z|X;\boldsymbol{\theta}^{(c)})\right] \\
&\quad +\sum_{z\in\mathcal{Z}}\left[\log\left(p(z|X;\boldsymbol{\theta}^{(c)})\right)p(z|X;\boldsymbol{\theta}^{(c)})\right] \\
&= -\sum_{z\in\mathcal{Z}}\left[\log\left(\frac{p(z|X;\boldsymbol{\theta})}{p(z|X;\boldsymbol{\theta}^{(c)})}\right)p(z|X;\boldsymbol{\theta}^{(c)})\right] \\
&\leq -\log\left(\sum_{z\in\mathcal{Z}}\left[\frac{p(z|X;\boldsymbol{\theta})}{p(z|X;\boldsymbol{\theta}^{(c)})}p(z|X;\boldsymbol{\theta}^{(c)})\right]\right) \\
&\leq -\log\left(\sum_{z\in\mathcal{Z}}p(z|X;\boldsymbol{\theta})\right) \\
&\leq 0
\end{aligned}$$

□

Ainsi, si nous prenons $\boldsymbol{\theta}^{(c+1)} \in \operatorname{argmax} Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(c)})$, nous avons :

$$L(\boldsymbol{\theta}^{(c+1)}) \geq L(\boldsymbol{\theta}^{(c)}).$$

L'algorithme *E*xpectation *M*aximisation de Dempster et al. (1977) est donc le suivant :

---

**Algorithme Expectation Maximisation** :

---

1. Choisir un premier $\boldsymbol{\theta}^{(0)} \in \Theta$.
2. Itérer jusqu'à ce que $L(\boldsymbol{\theta}^{(c)})$ n'évolue plus :
   (a) Étape $E$ : calcul pour tout $\boldsymbol{\theta}$ de

$$Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(c)}) = \mathbb{E}_{Z|X;\boldsymbol{\theta}^{(c)}}\left[\log\left(p(X,z;\boldsymbol{\theta})\right)\right].$$

   (b) Étape $M$ : prendre $\boldsymbol{\theta}^{(c+1)}$ tel que

$$\boldsymbol{\theta}^{(c+1)} \in \underset{\boldsymbol{\theta}\in\Theta}{\operatorname{argmax}}\, Q(\boldsymbol{\theta}|\boldsymbol{\theta}^{(c)}).$$

---

*Remarque* A.3.2. Cet algorithme converge vers un maximum local de la logvraisemblance.
L'étape $M$ peut être remplacée par une étape d'augmentation de $Q$ plutôt que de maximisation lorsque les calculs sont trop difficiles.

### A.3.3 Algorithme Stochastique Expectation Maximisation

L'algorithme *Stochastique Expectation Maximisation* ou *SEM* proposé par Celeux et al. (1995) ajoute une étape de simulation des données manquantes après l'étape $E$. Nous obtenons ainsi une loi stationnaire (Feodor Nielsen, 2000). Cet algorithme est peu sensible aux initialisations.

**Algorithme *Stochastic Expectation Maximisation* :**

1. Initialisation de $\boldsymbol{\theta}^{(0)}$.
2. Itérer *niter* fois :
   (a) Étape $SE$ : estimation de la loi $p\left(\mathbf{z}\left|\mathbf{x};\boldsymbol{\theta}^{(c)}\right.\right)$ puis tirage de $\mathbf{z}^{(c+1)}$ :

$$p\left(z_{ik}=1\left|x_i;\boldsymbol{\theta}^{(c)}\right.\right)=\frac{\pi_k^{(c)}\varphi\left(x_i;\alpha_k^{(c)}\right)}{\sum_{k'=1}^{g}\pi_{k'}^{(c)}\varphi\left(x_i;\alpha_{k'}^{(c)}\right)},$$

   (b) Étape $M$ : mise à jour du paramètre $\boldsymbol{\theta}^{(c+1)}$ :

$$\pi_k^{(c+1)}=\frac{\sum_{i=1}^{n}z_{ik}^{(c+1)}}{n}$$

   et la mise à jour de $\alpha_k^{(c+1)}$ dépend de la densité.
3. Obtention d'un estimateur $\widehat{\theta}^{SEM}=\frac{1}{niter}\sum_{c=1}^{niter}\boldsymbol{\theta}^{(c)}$ en moyennant les paramètres obtenus.
4. Calcul des estimateurs des classes $\widehat{z}^{SEM}$ par affectation à la classe majoritairement affectée durant les simulations :

$$\widehat{z}_i^{SEM}\quad\in\quad\underset{k=1,\ldots,g}{\operatorname{argmax}}\sum_{c=1}^{niter}z_{ik}^{(c)}.$$

### A.3.4 Différences de comportements entre les deux algorithmes

La fonction du maximum de vraisemblance peut être vue comme un massif montagneux (comme sur la figure A.4) dont le but est de trouver la plus haute montagne. Lorsque le nombre de classes est faible, ce massif comporte relativement peu de pics mais le nombre de sommets augmente avec le nombre de classes.

Pour retrouver la plus haute montagne, l'algorithme *VEM* peut être vu comme un randonneur partant du principe qu'il faille toujours monter pour atteindre le plus haut sommet. S'il commence à la place du photographe qui a pris la photo de la figure A.4, il finira sur le premier pic que nous voyons.

L'algorithme *SEM-Gibbs* peut être vu comme un randonneur qui, à chaque pas, choisit au hasard la prochaine direction en favorisant celles qui le font le plus monter. Ainsi, si nous le laissons marcher assez longtemps, il change régulièrement de montagnes en restant plus ou moins longtemps suivant si la pente est raide et si la montagne est haute ou non.

### A.3.5 Application

Nous avons appliqué l'algorithme $EM$ sur les données de la section précédente en relançant plusieurs fois avec des initialisations aléatoires et nous obtenons sur la figure A.5 la répartition des impacts sélectionnée par celui-ci. Même si ce n'est pas l'exacte répartition, le choix semble cohérent.

FIGURE A.4 – Photo d'un massif montagneux. Nous voyons en premier plan un sommet assez proche où un randonneur ayant la même stratégie que l'algorithme *VEM* risquerait d'aller tandis qu'un randonneur ayant la même stratégie que l'algorithme *SEM-Gibbs* le visiterait d'abord avant d'en descendre pour explorer les autres pics. Crédit photo : Hugues Habert.

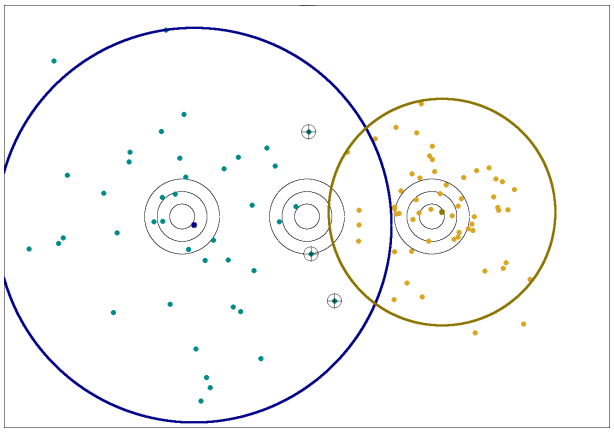


FIGURE A.5 – En bleu, les impacts du tireur de gauche et en or, celui de droite (des croix ont été mises lorsque les mauvais labels ont été affectés). Les couleurs plus foncées représentent les deux moyennes et des cercles symbolisent les deux régions de confiance à 95%.

# Bibliographie


C. Aaron. Algorithme em et classification non supervisée, 2004.

S. E. Allman, C. Matias, et A. J. Rhodes. Identifiability of parameters in latent structure models with many observed variables. *The Annals of Statistics*, 37(6A) :3099–3132, 2009. doi : 10.1214/09-AOS689. URL `http://hal.archives-ouvertes.fr/hal-00591202/en/`.

R. L. Andrews et I. S. Currim. A comparison of segment retention criteria for finite mixture logit models. *Journal of Marketing Research*, pages 235–243, 2003.

R. Anker. Ségrégation professionnelle hommes-femmes : les théories en présence. *Revue internationale du travail*, 136(3), 1997.

J. Aubert, T. Ha, et T. MaryHuard. Modele à blocs latents pour l'analyse de données métagénomiques. Dans $46^{ème}$ *journées de Statistiques de la SFdS*, 2014.

A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, et D. S. Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. *J. Mach. Learn. Res.*, 8 :1919–1986, Dec. 2007. ISSN 1532-4435. URL `http://dl.acm.org/citation.cfm?id=1314498.1314563`.

J. D. Banfield et A. E. Raftery. Model-based Gaussian and non-Gaussian clustering. *Biometrics*, pages 803–821, 1993.

J.-P. Baudry. *Sélection de modèle pour la classification non supervisée. Choix du nombre de classes.* Thèse, Université Paris Sud - Paris XI, Dec. 2009. URL `http://tel.archives-ouvertes.fr/tel-00461550/en/`.

A. Ben-dor, B. Chor, R. Karp, et Z. Yakhini. Discovering local structure in gene expression data : The order-preserving submatrix problem. pages 49–57, 2002.

J. Bennett et S. Lanning. The netflix prize. Dans *Proceedings of KDD cup and workshop*, volume 2007, page 35, 2007.

C. Biernacki et J. Jacques. Modele génératif pour données ordinales. Dans *44e Journées de Statistique, SFdS*, Bruxelles, Belgique, May 2012. URL `http://jds2012.ulb.ac.be/myreview/files/default/submission/submission_123.pdf`.

C. Biernacki, G. Celeux, et G. Govaert. Assessing a mixture model for clustering with the integrated completed likelihood. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(7) :719–725, 2000.

C. Biernacki, G. Celeux, A. Echenim, G. Govaert, et F. Langrognet. Le logiciel MIXMOD d'analyse de mélange pour la classification et l'analyse discriminante. *La revue de Modulad*, (35) :25–44, Dec. 2006. URL `http://hal.archives-ouvertes.fr/hal-00469501`.

P. Billingsley. *Measure and Probability*. John Willey, 1986.

C. M. Bishop. *Pattern recognition and machine learning*, volume 1. springer New York, 2006.

H. Bock. Simultaneous clustering of objects and variables. *Analyse des données et Informatique*, pages 187–203, 1979.

V. Brault et A. Lomet. Revue bibliographique pour la classification croisée. 2014.

V. Brault et M. Mariadassou. Latent Bloc Model : a Review. 2014.

S. P. Brooks et A. Gelman. General methods for monitoring convergence of iterative simulations. *Journal of computational and graphical statistics*, 7(4) :434–455, 1998.

K. P. Burnham et D. R. Anderson. Multimodel inference : understanding AIC and BIC in model selection. *Sociological Methods & Research*, 33(2) :261, 2004.

G. Celeux et G. Govaert. A classification EM algorithm for clustering and two stochastic versions. *Computational Statistics & Data Analysis*, 14 :315–332, October 1992. ISSN 0167-9473. doi : 10.1016/0167-9473(92)90042-E. URL `http://dl.acm.org/citation.cfm?id=146597.146608`.

G. Celeux et C. Robert. Une histoire de discrétisation. *La Revue de Modulad*, 11 :7–44, 1993.

G. Celeux, D. Chauveau, et J. Diebolt. On Stochastic Versions of the EM Algorithm. Rapport de recherche RR-2514, INRIA, 1995. URL `http://hal.inria.fr/inria-00074164`.

G. Celeux, M. Hurn, et C. P. Robert. Computational and inferential difficulties with mixture posterior distributions. *Journal of the American Statistical Association*, 95(451) :957–970, 2000.

A. Celisse, J.-J. Daudin, et L. Pierre. Consistency of maximum-likelihood and variational estimators in the stochastic block model. *Electronic Journal of Statistics*, 6 :1847–1899, 2012.

A. Channarond, J.-J. Daudin, S. Robin, et al. Classification and estimation in the stochastic blockmodel based on the empirical degrees. *Electronic Journal of Statistics*, 6 :2574–2601, 2012.

M. Charrad, Y. Lechevallier, G. Saporta, et M. Ben Ahmed. Détermination du nombre de classes dans les méthodes de bipartitionnement. Dans *17ème Rencontres de la Société Francophone de Classification*, pages 119–122, Saint-Denis de la Réunion, June 2010.

Y. Cheng et G. Church. Biclustering of expression data. Dans *International Conference on Intelligent Systems for Molecular Biology ; ISMB. International Conference on Intelligent Systems for Molecular Biology*, volume 8, pages 93–103, 1999.

P. De Santis. *La traduction*. Métailié edition, 2004.

A. P. Dempster, N. M. Laird, D. B. Rubin, et al. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal statistical Society*, 39(1) :1–38, 1977.

M. Deodhar et J. Ghosh. Simultaneous co-clustering and modeling of market data. Dans *Proceedings of the Workshop for Data Mining in Marketing (DMM 2007)(Leipzig, Germany). IEEE Computer Society Press, Los Alamitos, CA*, 2007.

E. Deschavanne et P.-H. Tavoillot. *Philosophie des âges de la vie*. Grasset, 2007.

I. S. Dhillon, S. Mallela, et D. S. Modha. Information-theoretic co-clustering. Dans *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98. ACM, 2003.

B. M. d'Intignano, M. Aglietta, G. Cette, M. Glaude, et C. Sofer. *Egalité entre femmes et hommes : aspects économiques*. La Documentation française, 1999.

D. E. Duffy et J. Quiroz. A permutation-based algorithm for block clustering. *Journal of Classification*, 8(1) :65–91, 1991. ISSN 0176-4268.

S. Feodor Nielsen. The stochastic em algorithm : estimation and asymptotic results. *Bernoulli*, 6(3) : 457–489, 2000.

C. Fraley et A. E. Raftery. How many clusters ? which clustering method ? answers via model-based cluster analysis. *The Computer Journal*, 41(8) :578–588, 1998.

S. Frühwirth-Schnatter. *Mixtures : estimation and applications*, chapter Dealing with label switching under model uncertainty, pages 193–218. Wiley, 2011. ISBN 9781119993896 111999389. Editors Mergensen, K., Robert, P. and Titterington, M.

S. Fu. *Inversion probabiliste bayésienne en analyse d'incertitude*. Thèse, Université Paris-Sud 11, 2012.

A. E. Gelfand et A. F. Smith. Sampling-based approaches to calculating marginal densities. *Journal of the American statistical association*, 85(410) :398–409, 1990.

A. Gelman et D. B. Rubin. Inference from iterative simulation using multiple sequences. *Statistical science*, pages 457–472, 1992.

S. Geman et D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6) :721–741, 1984.

C. J. Geyer. Practical markov chain monte carlo. *Statistical Science*, pages 473–483, 1992.

I. J. Good. *Categorization of classification*. Mathematics and Computer Science in Biology and Medicine. Her Majesty's Stationery Office, 1965.

G. Govaert. Algorithme de classification d'un tableau de contingence. Dans *First international symposium on data analysis and informatics*, pages 487–500, Versailles, 1977. INRIA.

G. Govaert. *Classification croisée*. Thèse d'état, Université Pierre et Marie Curie, 1983.

G. Govaert. Simultaneous clustering of rows and columns. *Control and Cybernetics*, 24(4) :437–458, 1995.

G. Govaert et M. Nadif. Clustering with block mixture models. *Pattern Recognition*, 36 :463–473, 2003.

G. Govaert et M. Nadif. Clustering of contingency table and mixture model. *European Journal of Operational Research*, 183 :1055–1066, 2007.

G. Govaert et M. Nadif. Block clustering with Bernoulli mixture models : Comparison of different approaches. *Computational Statistics and Data Analysis*, 52 :3233–3245, 2008.

G. Govaert et M. Nadif. Un modèle de mélange pour la classification croisée d'un tableau de données continue. Dans *CAP'09, 11e conférence sur l'apprentissage artificiel*, pages 287–302, Hammamet, Tunisie, 2009. URL `http://hal.archives-ouvertes.fr/hal-00447804/en/`.

G. Govaert et M. Nadif. *Co-Clustering*. ISTE Ltd and John Wiley & Sons, Inc, 2013.

M. Gyllenberg, T. Koski, E. Reilink, et M. Verlann. Non-Uniqueness in Probabilistic Numerical Identification of Bacteria. *Journal of Applied Probability*, 31(2) :542–548, 1994.

B. Hanczar et M. Nadif. Dans J. Balcázar, F. Bonchi, A. Gionis, et M. Sebag, editors, *Machine Learning and Knowledge Discovery in Databases*, volume 6321 of *Lecture Notes in Computer Science*. 2010. ISBN 978-3-642-15879-7.

J. Hansohm. Two-mode clustering with genetic algorithms. Dans *Classification, automation, and new media*, pages 87–93. Springer, 2002.

J. A. Hartigan. *Clustering Algorithms*. John Wiley & Sons, Inc., New York, NY, USA, 99th edition, 1975. ISBN 047135645X.

J. A. Hartigan. Bloc voting in the United States senate. *Journal of Classification*, 17(1) :29–49, 2000.

I. Hedenfalk, D. Duggan, Y. Chen, M. Radmacher, M. Bitter, R. Simon, P. Meltzer, B. Gusterson, M. Esteller, et M. Raffeld. Gene-expression profiles in hereditary breast cancer. *New Eng. J. Med.*, 344 :539–548, 2001.

K. Humphreys et D. Titterington. Approximate bayesian inference for simple mixtures. Dans *COMPSTAT*, pages 331–336. Springer, 2000.

J. Ihmels, S. Bergmann, et N. Barkai. Defining transcription modules using large-scale gene expression data. *Bioinformatics*, 20(13) :1993–2003, 2004.

M. Jagalur, C. Pal, E. Learned-Miller, R. T. Zoeller, et D. Kulp. Analyzing in situ gene expression in the mouse brain with image registration, feature extraction and block clustering. *BMC Bioinformatics*, 8 (Suppl 10) :S5, 2007. ISSN 1471–2105.

H. Jeffreys. An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 186(1007) :453–461, 1946.

R. E. Kass et L. Wasserman. The selection of prior distributions by formal rules. *Journal of the American Statistical Association*, 91(435) :1343–1370, 1996.

C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, et N. Ueda. Learning systems of concepts with an infinite relational model. Dans *Proceedings of The Twenty-First National Conference on Artificial Intelligence*, pages 381–388. AAAI Press, 2006.

C. Keribin. Consistent estimation of the order of mixture models. *Sankhya Series A*, 62 :49–66, 2000.

C. Keribin. Méthodes bayésiennes variationnelles : concepts et applications en neuroimagerie. *Journal de la Société Française de Statistique*, 151(2) :107–131, 2010.

C. Keribin, G. Govaert, et G. Celeux. Estimation d'un modèle à blocs latent par l'algorithme SEM. Dans *42e Journées de Statistique, SFdS*, Marseille, France, May 2010. URL `http://hal.archives-ouvertes.fr/hal-00554409/en/`.

C. Keribin, V. Brault, G. Celeux, et G. Govaert. Model selection for the binary latent block model. Dans *20th International Conference on Computational Statistics*, Limassol, Chypre, 2012. URL `http://hal.inria.fr/hal-00778145`.

C. Keribin, V. Brault, G. Celeux, et G. Govaert. Estimation and selection for the latent block model on categorical data. *Statistics and Computing*, pages 1–16, 2014. ISSN 0960-3174. doi : 10.1007/s11222-014-9472-2. URL `http://dx.doi.org/10.1007/s11222-014-9472-2`.

Y. Kluger, R. Basri, J. T. Chang, et M. Gerstein. Spectral biclustering of microarray data : coclustering genes and conditions. *Genome Research*, 13(4) :703–716, 2003.

S. Kullback et R. A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, pages 79–86, 1951.

P. S. Laplace. *Essai philosophique sur les probabilités*. 1825.

D. Lashkari et P. Golland. Co-clustering with generative models. Rapport technique, MITCSAIL-TR-2009-054, 2009.

L. Lazzeroni et A. Owen. Plaid models for gene expression data. *Statistica Sinica*, 12 :61–86, 2000.

E. Lebarbier et T. Mary-Huard. Le critère BIC : fondements théoriques et interprétation. Rapport de recherche RR-5315, INRIA, 2004. URL `http://hal.inria.fr/inria-00070685/en/`.

B. Lecoutre. Et si vous étiez un bayésien qui s'ignore ? *Revue Modulad*, 32 :92–105, 2005.

H. Leredde et P. Perin. Les plaques-boucles mérovingiennes. *Dossiers de l'archéologie*, 42 :83–87, 1980.

I. Lerman et H. Leredde. La méthode des pôles d'attraction. *Journées Analyse des Données et Informatique*, 1977.

A. Lomet. *Sélection de modèle pour la classification croisée de données continues.* Thèse, Université de Technologie de Compiègne, Décembre 2012.

A. Lomet, G. Govaert, et Y. Grandvalet. An approximation of the integrated classification likelihood for the latent block model. Dans *Data Mining Workshops (ICDMW), 2012 IEEE 12th International Conference on*, pages 147–153. IEEE, 2012a.

A. Lomet, G. Govaert, et Y. Grandvalet. Model selection in block clustering by the integrated classification likelihood. Dans *Compstat 2012*, pages 519–530, 2012b.

A. Lomet, G. Govaert, et Y. Grandvalet. Un protocole de simulation de données pour la classification croisée. Dans *44$^e$Journées de Statistique de la SFdS*, 2012c.

B. Long, Z. M. Zhang, et P. S. Yu. Co-clustering by block value decomposition. Dans *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 635–640. ACM, 2005.

J. MacQueen. Some methods for classification and analysis of multivariate observations. Dans *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, volume 1, page 14. California, USA, 1967.

S. C. Madeira et A. L. Oliveira. Biclustering algorithms for biological data analysis : a survey. *Computational Biology and Bioinformatics, IEEE/ACM Transactions on*, 1(1) :24–45, 2004.

M. Mariadassou et C. Matias. Convergence of the groups posterior distribution in latent or stochastic block models. *arXiv preprint arXiv :1206.7101*, 2012.

M. Mariadassou, S. Robin, et C. Vacher. Uncovering latent structure in valued graphs : a variational approach. *The Annals of Applied Statistics*, 4(2) :715–742, 2010.

C. Matias et S. Robin. Modeling heterogeneity in random graphs : a selective review. *arXiv preprint arXiv :1402.4296*, 2014.

C. Maugis, M.-L. Martin-Magniette, J.-P. Tamby, J.-P. Renou, A. Lecharny, S. Aubourg, et G. Celeux. Sélection de variables pour la classification par mélanges gaussiens pour prédire la fonction des gènes orphelins. *La Revue de Modulad*, 40 :69–80, 2009.

G. McLachlan et T. Krishnan. *The EM algorithm and extensions*, volume 382. John Wiley & Sons, 2008.

G. J. McLachlan. The classification and mixture maximum likelihood approaches to cluster analysis. *Handbook of statistics*, 2 :199–208, 1982.

E. Meeds et S. Roweis. Nonparametric Bayesian biclustering. Technical report, University of Toronto, 2007.

N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, et E. Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6) :1087–1092, 1953.

G. W. Milligan et M. C. Cooper. An examination of procedures for determining the number of clusters in a data set. *Psychometrika*, 50(2) :159–179, 1985.

T. Murali et S. Kasif. Extracting conserved gene expression motifs from gene expression data. Dans *Pacific Symposium on Biocomputing*, volume 8, pages 77–88, 2003.

M. Nadif et G. Govaert. Block clustering and statistical modelling. Dans *Symposium on mixture modeling*, Leuven, Belgium, November 20-21 2007.

K. Nowicki et T. A. B. Snijders. Estimation and prediction for stochastic blockstructures. *Journal of the American Statistical Association*, 96(455) :1077–1087, 2001.

S. Oyanagi, K. Kubota, et A. Nakase. Application of matrix clustering to web log analysis and access prediction. Dans *WEBKDD 2001-Mining Web Log Data Across All Customers Touch Points, Third International Workshop*, pages 13–21, 2001.

M.-J. Papillon, C. Laurier, L. Barnard, et J. Baril. Consommation de médicaments. *la santé et le bien-être*, page 445, 2001.

C. Perrault. *Les contes de Perrault.* J. Hetzel, 1867.

J. Podani et E. Feoli. A general strategy for the simultaneous classification of variables and objects in ecological data tables. *Journal of Vegetation Science*, 2(4) :435–444, 1991. ISSN 1100–9233.

A. Prelić, S. Bleuler, P. Zimmermann, A. Wille, P. Bühlmann, W. Gruissem, L. Hennig, L. Thiele, et E. Zitzler. A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, 22(9) :1122, 2006.

A. E. Raftery. Bayesian model selection in social research. *Sociological methodology*, 25 :111–164, 1995.

A. E. Raftery et S. M. Lewis. [practical markov chain monte carlo] : Comment : One long run with diagnostics : Implementation strategies for markov chain monte carlo. *Statistical Science*, pages 493–497, 1992.

H. Raiffa et R. Schlaifer. Applied statistical decision theory. *Division of Research, Harvard Business School, Boston, MA*, 1961.

C. Robert. *Le choix bayésien : Principes et pratique.* Springer Science & Business, 2006. URL `http://opac.inria.fr/record=b1119586`.

R. Rocci et M. Vichi. Two-mode multi-partitioning. *Computational Statistics and Data Analysis*, 52(4) : 1984–2003, 2008.

M. Rooth. Two-dimensional clusters in grammatical relations. Dans *AAAI Symposium on Representation and Acquisition of Lexical Knowledge*, 1995.

D. M. Roy et Y. W. Teh. The mondrian process. Dans *NIPS*, pages 1377–1384, 2008.

G. Saporta. *Probabilités, analyse des données et statistique*. Editions Technip, 2011.

J. Schepers, E. Ceulemans, et I. Van Mechelen. Selecting among multi-mode partitioning models of different complexities : A comparison of four model selection criteria. *Journal of Classification*, 25(1) : 67–85, 2008.

G. Schwarz et al. Estimating the dimension of a model. *The annals of statistics*, 6(2) :461–464, 1978.

Y. Seldin et N. Tishby. Pac-Bayesian analysis of co-clustering and beyond. *The Journal of Machine Learning Research*, 11 :3595–3646, 2010.

D. Seung et L. Lee. Algorithms for non-negative matrix factorization. *Advances in Neural Information Processing Systems 13*, pages 556–562, 2001.

M. Shafiei et E. Milios. Model-based overlapping co-clustering. Dans *Proceeding of SIAM Conference on Data Mining*, 2006.

H. Shan et A. Banerjee. Bayesian co-clustering. Dans *Eighth IEEE International Conference on Data Mining, 2008. ICDM'08*, pages 530–539, 2008.

A. Tanay, R. Sharan, et R. Shamir. Discovering statistically significant biclusters in gene expression data. Dans *Proceedings of ISMB 2002*, pages 136–144, 2002.

N. Tishby, F. Pereira, et W. Bialek. The information bottleneck method. Dans *Invited paper to The 37th annual Allerton Conference on Communication, Control, and Computing*, 1999.

B. Van Dijk, J. Van Rosmalen, et R. Paap. A Bayesian approach to two-mode clustering. Technical Report 2009-06, Econometric Institute, 2009. URL `http://hdl.handle.net/1765/15112`.

Y. Vasseur. Modèles de régression linéaire en grande dimension pour l'étude des facteurs de transcription d'arabidopsis thaliana. Mémoire de master, Université Paris Sud, 2013.

B. Wang et D. Titterington. Convergence and asymptotic normality of variational bayesian approximations for exponential family models with missing values. Dans *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 577–584. AUAI Press, 2004a.

B. Wang et D. Titterington. Variational bayes estimation of mixing coefficients. Dans *Deterministic and statistical methods in machine learning*, pages 281–295. Springer, 2005.

B. Wang et M. Titterington. Inadequacy of interval estimates corresponding to variational bayesian approximations. 2004b.

B. Wang, D. Titterington, et al. Convergence properties of a general algorithm for calculating variational bayesian estimates for a normal mixture model. *Bayesian Analysis*, 1(3) :625–650, 2006.

P. Wang, K. B. Laskey, C. Domeniconi, et M. Jordan. Nonparametric bayesian co-clustering ensembles. Dans *The 2011 SIAM International Conference on Data Mining*, 2011.

J. Wyse et N. Friel. Block clustering with collapsed latent block models. *Statistics and Computing*, pages 1–14, 2010.

J. Yang, W. Wang, H. Wang, et P. Yu. $\delta$-clusters : Capturing subspace correlation in a large data set, 2002.

J. Yoo et S. Choi. Orthogonal nonnegative matrix tri-factorization for co-clustering : Multiplicative updates on stiefel manifolds. *Information processing & management*, 46(5) :559–570, 2010.

# Index


Algorithme
  $K$-means, 130
  $\delta$-classification, 35
  Échantillonneur de Gibbs, 88, 92, 93
  Brute-Force Deletion and Addition, 35
  CC, 35
  CEM, 130, 166
  CEM-Bayes, 166
  Crobin, 26
  Croeuc, 26
  Croki2, 26
  EM, 28, 63, 184, 185
  EM variationnel, 65
  Expectation Maximisation, 63, 184
  FLexible Overlapped Clustering, 37
  initialisation de l'amélioration de l'algorithme LG, 149
  ISA, 35
  Largest Gaps, 123, 124
  Largest Gaps adapté, 131
  MCMC, 33
  Metropolis-Hasting, 33
  Metropolis-Hastings, 83
  Multiple Node Addition, 37
  Multiple Node Deletion, 37
  One-Way Splitting algorithm, 33
  Order-Preserving Submatrices, 35, 39
  SAMBA, 35, 38
  SEM, 185
  SEM+VEM, 69
  SEM-Gibbs, 67
  Splitting algorithm, 33
  Stochastic Expectation Maximisation, 67
  Stochastique Expectation Maximisation, 185
  Two-Way Joining Algorithm, 33
  Two-Way Splitting algorithm, 33
  V-Bayes, 86
  Variational Expectation Maximization, 29
  VEM, 29, 65
  watershed, 131
  xMotif, 35
Approximation
  de Laplace, 105
  en champ moyen, 65
  variationnelle, 64

Bayes
  Inférence bayésienne, 83
  règle, 83
Biclustering, 21
Bloc, 18
  homogène, 22
Brooks-Gelman
  statistique, 92
  statistique modifiée, 93

Classe
  qui se vide, 70
Classification
  $\delta$-classification, 35
  avec chevauchement, 34
  croisée, 21
  hiérarchique, 30
  imbriquée, 30
  imbriquée non restreinte, 33
  imbriquée par blocs, 33
  imbriquée restreinte, 33
  imbriquée sur les lignes et les colonnes, 31
  jointe, 21
  principe, 21
Coclustering, 24
Convergence
  géométrique, 88
Critère
  Bayesian Information Criterion, 100
  BIC, 100
  d'arrêt pour l'échantillonneur de Gibbs, 92
  d'arrêt pour l'échantillonneur de Gibbs (amélioration), 93

ICL, 100
ICLab, 102
Integrated Completed Likelihood, 100

Décomposition
en blocs à valeurs non-négatives, 26, 28
Dégénérescence
des classes, 62, 70
Distance
de classification, 51
Distribution
libre, 64
Donnée
homogène, 43

Échantillonneur
de Gibbs, 88
intégré, 112
Énergie
libre, 64
Erreur
de classification, 52
Estimateur
du maximum de vraisemblance, 63
État
absorbant, 68

Factorisation
de matrice non-négative, 26, 27
Formule
des probabilités totales, 183
Fractionnement
direct, 33

Gibbs
Échantillonneur, 88
SEM-Gibbs, 67
Graphe
aléatoire, 30, 46
bipartie, 22

Hoeffding
Inégalité, 141

Identifiabilité, 46, 48
conditions suffisantes, 47
générique, 46
Inégalité
de Hoeffding, 141
de Jensen, 185
Inférence
bayésienne, 83

Label
switching, 47
Logvraisemblance, 182
Loi
a priori, 83
a priori conjuguée, 83
bêta, 84
de Dirichlet, 84
multinomiale, 45

Matrice
non-négative, 27
Méthode
CRO, 26
Model
based, 26
Modèle
de mélange, 183
des blocs latents, 43
génériquement identifiable, 46
identifiable, 46
infini relationnel, 30

Non-negative block value decomposition, 28

Principe
de dualité, 88
Processus
de Dirichlet, 33
Mondrian, 33

Reconstruction
based, 26
de matrices, 26
Règle
de Bayes, 83
Relation
d'équivalence, 52, 124
d'ordre totale, 48

Statistiques
de Brooks-Gelman, 92
de Brooks-Gelman modifiée, 93
Stochastic
bloc model, 30, 46

Temps
de chauffe, 67
Tri-factorisation
orthogonale de matrice non-négative, 26, 28