Nº 12-001-X au catalogue
ISSN 1712-5685

Techniques d'enquête

Algorithme génétique de regroupement pour la stratification et la répartition simultanée de l'échantillon dans les plans de sondage

par Mervyn O'Luing, Steven Prestwich et S. Armagan Tarim

Date de diffusion : le 17 décembre 2019

Statistique Canada Statistics Canada

Canada

Comment obtenir d'autres renseignements

Pour toute demande de renseignements au sujet de ce produit ou sur l'ensemble des données et des services de Statistique Canada, visiter notre site Web à www.statcan.gc.ca.

Vous pouvez également communiquer avec nous par :

Courriel à STATCAN.infostats-infostats.STATCAN@canada.ca

Téléphone entre 8 h 30 et 16 h 30 du lundi au vendredi aux numéros suivants :

- Service de renseignements statistiques 1-800-263-1136
- Service national d'appareils de télécommunications pour les malentendants 1-800-363-7629
- Télécopieur 1-514-283-9350

Programme des services de dépôt

- Service de renseignements 1-800-635-7943
- Télécopieur 1-800-565-7757

Normes de service à la clientèle

Statistique Canada s'engage à fournir à ses clients des services rapides, fiables et courtois. À cet égard, notre organisme s'est doté de normes de service à la clientèle que les employés observent. Pour obtenir une copie de ces normes de service, veuillez communiquer avec Statistique Canada au numéro sans frais 1-800-263-1136. Les normes de service sont aussi publiées sur le site www.statcan.gc.ca sous « Contactez-nous » > « Normes de service à la clientèle ».

Note de reconnaissance

Le succès du système statistique du Canada repose sur un partenariat bien établi entre Statistique Canada et la population du Canada, les entreprises, les administrations et les autres organismes. Sans cette collaboration et cette bonne volonté, il serait impossible de produire des statistiques exactes et actuelles.

Techniques d'enquête, décembre 2019
Vol. 45, nº 3, pp. 549-568
Statistique Canada, nº 12-001-X au catalogue

Algorithme génétique de regroupement pour la stratification et la répartition simultanée de l'échantillon dans les plans de sondage

Mervyn O'Luing, Steven Prestwich et S. Armagan Tarim[1]

Résumé

Lorsque la base de sondage est importante, il est difficile d'optimiser la stratification et la taille de l'échantillon dans un plan de sondage univarié ou multivarié. Il existe diverses façons de modéliser et de résoudre ce problème. Un des moyens les plus naturels est l'algorithme génétique (AG) combiné à l'algorithme d'évaluation de Bethel-Chromy. Un AG recherche itérativement la taille minimale d'échantillon permettant de respecter les contraintes de précision lorsqu'il s'agit de partitionner les strates atomiques formées par le produit cartésien de variables auxiliaires. Nous signalons un inconvénient avec les AG classiques appliqués à un problème de regroupement et proposons un nouvel algorithme génétique de « regroupement » avec des opérateurs génétiques au lieu des opérateurs classiques. Des expériences indiquent qu'on se trouve ainsi à améliorer nettement la qualité de solution pour un même effort de calcul.

Mots-clés : Algorithme génétique de regroupement; stratification optimale; répartition de l'échantillon; logiciel R.

1 Introduction

Nous allons traiter du problème d'optimisation que pose la détermination simultanée de la stratification et de la répartition de l'échantillon dans des scénarios univariés ou multivariés. Nous nous reportons à cette fin à Ballin et Barcaroli (2013). En principe, nous pouvons parvenir à une stratification optimale (propre à donner la plus petite taille d'échantillon) en essayant toutes les partitions possibles de *strates atomiques*, mais le nombre de ces partitions croît exponentiellement avec le nombre de ces strates.

Nous avons besoin d'un algorithme de recherche efficace pour ne pas avoir à évaluer chaque partition possible. Souvent, les AG convergent rapidement vers des solutions optimales ou quasi optimales et réussissent particulièrement bien à « négocier » les espaces accidentés de recherche où abondent les minima locaux. L'algorithme de Bethel-Chromy se trouve à combiner les algorithmes apparentés de Bethel (1985, 1989) et de Chromy (1987) et convient bien à des scénarios univariés ou multivariés. Il se sert de multiplicateurs de Lagrange pour dégager la taille minimale d'échantillon respectant les contraintes de précision pour une stratification. Ballin et Barcaroli (2013) combinent un AG et cet algorithme dans une recherche de taille minimale d'échantillon. Ils évaluent ainsi chaque partition créée par l'AG. On trouvera une description entière de la méthodologie et de l'énoncé du problème dans Ballin et Barcaroli (2013). Toutefois, ces auteurs emploient un AG classique dont on sait qu'il convient mal au traitement de problèmes de partition.

Nous proposerons d'appliquer à l'AG des opérateurs génétiques qui se prêtent mieux à un tel emploi. C'est là un exemple de la catégorie d'algorithmes évolutionnistes appelés *algorithmes génétiques de regroupement* (AGR). Nous avons renouvelé notre AG en nous inspirant de ce travail (Barcaroli, 2019). À

1. Mervyn O'Luing et Steven Prestwich, Insight Centre for Data Analytics, Département d'informatique, University College Cork, Irlande. Courriel : mervyn.oluing@insight-centre.org et steven.prestwich@insight-centre.org; S. Armagan Tarim, Cork University Business School, University College Cork, Irlande. Courriel : armagan.tarim@ucc.ie.

la section 2, nous exposons la raison d'être de notre recherche et présentons les AGR. À la section 2.3, nous décrivons l'AGR que nous appliquons au problème à résoudre. À la section 3, nous comparons l'AG de départ à notre AGR pour des données d'essai du domaine public. À la section 4, nous décrivons une version de notre AGR au rendement amélioré dans une mise en œuvre rapide en langage C++ de la fonction *bethel.r*, laquelle est intégrée en R par le paquet Rcpp. Nous concluons enfin à la section 5.

2 Algorithmes génétiques classiques et de regroupement

Dans cette section, il sera question d'AG « classiques » et d'AG de « regroupement ». Nous expliquerons en quoi les derniers conviennent mieux à notre problème.

2.1 Algorithmes génétiques classiques

Les AG forment une catégorie d'algorithmes d'optimisation inspirée par la nature et formée d'après la capacité des organismes à résoudre le problème complexe de l'adaptation à la vie sur Terre. Les variables d'un problème d'optimisation sont appelées *gènes* et leurs valeurs, *allèles*. Une solution possible est une liste d'allèles appelée *chromosome*. Un ensemble de chromosomes est ce qu'on appelle habituellement une *population*. Pour ne pas confondre cette population avec la population visée, nous parlerons de *population de chromosomes* à propos des AG. La fonction objectif (qui est à maximiser par convention) est celle de l'adaptation d'un chromosome. Dans la recherche de chromosomes adaptés (c'est-à-dire de solutions à haute valeur d'objectif), nous employons deux *opérateurs génétiques*, ceux de la *mutation* pour les légères variations aléatoires équivalant à de petits pas locaux dans un algorithme d'escalade, d'une part, et de la *permutation* pour de grandes variations avec *recombinaison* des gènes de deux *chromosomes parents*. Un opérateur de recombinaison bien connu est l'opérateur de *permutation ponctuelle.* Dans ce cas, nous choisissons deux chromosomes *parents* aux allèles

$$a_1, \ldots, a_N \qquad b_1, \ldots, b_N,$$

prenons un entier aléatoire i (*point de permutation*) avec $1 \leq i < N$ et produisons deux nouveaux chromosomes de *filiation*

$$a_1, \ldots, a_i, b_{i+1}, \ldots, b_N \qquad b_1, \ldots, b_i, a_{i+1}, \ldots, a_N.$$

Les chromosomes fils pourraient alors être soumis à une *mutation* aléatoire où quelques allèles sont changés avant d'être reversés dans la population de chromosomes. Il existe diverses méthodes de sélection des parents et de remplacement des chromosomes. Dans les AG *générationnels*, toute la population de chromosomes se trouve remplacée par sa descendance et les parents sont souvent choisis au hasard, mais avec un biais en faveur des chromosomes mieux adaptés. Dans les AG *stationnaires*, une seule descendance naît à chaque itération de l'algorithme et le chromosome le moins adapté est habituellement remplacé dans la population de chromosomes. Les AG donnent dans bien des cas des résultats plus robustes que ceux des algorithmes de recherche à escalade grâce à leur recours à la recombinaison. Ils ont trouvé de nombreux usages depuis leur introduction en 1975 par John Holland.

L'AG représenté dans le paquet *SamplingStrata* en *R* (R Core Team, 2015) (Barcaroli, 2014) est un AG générationnel élitiste où les strates atomiques L sont considérées comme les éléments d'un ensemble (ou gènes) pour une stratégie type de permutation. À chaque itération, les meilleures solutions (*élite*) sont reportées à la génération qui suit. Chaque gène représente une variable du problème. C'est ce que nous appelons l'AG classique parce que l'on fait intervenir une représentation et des opérateurs génétiques classiques, comme nous allons le décrire.

Diviser les strates atomiques en groupes disjoints est un exemple de problème de *regroupement* qui ressemble aux problèmes de *découpe*, de *garnissage* et de *partition*. Ce qui a motivé la présente recherche, c'est que nous savons que les AG classiques donnent de piètres résultats avec les problèmes de regroupement. La raison en est que la représentation chromosomique d'un regroupement comporte beaucoup de *symétrie* (ou de *redondance*) : permuter entre groupes donne un regroupement équivalent, de sorte que chaque regroupement est à représentations multiples. La symétrie nuit aux AG, car recombiner dans un même regroupement parent pourrait donner un regroupement de filiation très différent, ce qui va à l'encontre du principe de base des algorithmes génétiques selon lequel les parents devraient généralement produire une descendance d'une même adaptation. Dans des cas extrêmes, un AG classique pourrait donner un résultat même pire que celui d'une recherche entièrement aléatoire. Nous éclairons ce problème par deux exemples.

Illustrons le problème de symétrie dans un premier exemple avec des parents représentant le même regroupement mais en des combinaisons différentes. Pour plus de lisibilité, nous employons les lettres A à F comme allèles au lieu de nombres entiers. Prenons les deux chromosomes suivants :

	groupes représentés					
chromosome	A	B	C	D	E	F
ABCDEF	{1}	{2}	{3}	{4}	{5}	{6}
FEDCBA	{6}	{5}	{4}	{3}	{2}	{1}

pour le regroupement $\{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}\}$. Posons maintenant qu'une permutation ponctuelle permet d'obtenir de nouveaux chromosomes de filiation de ces parents. Si nous choisissons arbitrairement le centre des chromosomes comme point de permutation, nous obtenons la descendance suivante :

	groupes représentés					
chromosome	A	B	C	D	E	F
ABCCBA	{1; 6}	{2; 5}	{3; 4}	∅	∅	∅
FEDDEF	∅	∅	∅	{3; 4}	{2; 5}	{1; 6}

avec les deux, le regroupement est totalement autre $\{\{1; 6\}, \{2; 5\}, \{3; 4\}\}$: aucun groupe n'est reporté des parents à la descendance. Ainsi, l'adaptation de la descendance peut être sans lien aucun avec celle des parents et l'emploi de l'algorithme génétique se ramène à une recherche quasiment aléatoire. Voici un exemple avec les deux chromosomes classiques suivants :

	groupes représentés					
chromosome	A	B	C	D	E	F
AECFEC	{1}	∅	{3; 6}	∅	{2; 5}	{4}
DFFDAA	{5; 6}	∅	∅	{1; 4}	∅	{2; 3}

avec les deux, le regroupement est hétérogène : $\{\{1\}, \{3; 6\}, \{2; 5\}, \{4\}\}$ et $\{\{5; 6\}, \{1; 4\}, \{2; 3\}\}$. Par la même stratégie de permutation, nous obtenons la descendance suivante :

	groupes représentés					
chromosome	A	B	C	D	E	F
AECDAA	{1; 5; 6}	∅	{3}	{4}	{2}	∅
DFFFEC	∅	∅	{6}	{1}	{5}	{2; 3; 4}

pour les regroupements $\{\{1; 5; 6\}, \{3\}, \{4\}, \{2\}\}$ et $\{\{6\}, \{1\}, \{5\}, \{2; 3; 4\}\}$. À noter que les fils ont très peu en commun avec les parents, les seuls groupes préservés étant $\{1\}$ et $\{4\}$.

2.2 Algorithmes génétiques de regroupement

On peut s'attaquer au problème de symétrie en y allant de représentations d'opérateurs génétiques plus complexes (Galinier et Hao, 1999) ou en recourant aux techniques de classification automatique (Pelikan et Goldberg, 2000). Le risque avec ce mode de classification en grappes est que la diversité génétique se perde si les grappes en question sont trop étroites, d'où une éventuelle stagnation de la recherche (Prügel-Bennett, 2004). Il s'agit plutôt pour nous de poursuivre dans la voie déjà tracée en concevant un AGR (Falkenauer, 1998) dont on sait qu'il donne de bien meilleurs résultats que les AG classiques avec les problèmes de regroupement.

Les AGR sont expressément conçus pour résoudre de tels problèmes et ont trouvé de nombreux usages dans le déploiement de réseaux WiFi (Agustín-Blas, Salcedo-Sanz, Vidales, Urueta et Portilla-Figueras, 2011), la conception de réseaux sans fil (Brown et Vroblefski, 2004), le coupage de plaques d'acier (Hung, Sumichrast et Brown, 2003), l'aménagement d'ateliers de production (De Lit, Falkenauer et Delchambre, 2000) et l'analyse de réseaux sociaux (James, Brown et Ragsdale, 2010). L'heuristique peut être la même que pour les autres AG (sélection des parents, remplacement par filiation, etc.), mais le codage et les opérateurs génétiques diffèrent, c'est-à-dire la façon de traiter le tout comme des chromosomes et de procéder à la recombinaison et à la mutation. Nous allons illustrer les différences en reprenant nos exemples.

Les AGR représentent un regroupement comme une liste ordonnée de sous-ensembles en écartant les ensembles vides. Les parents dans le second exemple à la section 2.1 peuvent ainsi être représentés :

$$\langle\{1\}, \{3; 6\}, \{2; 5\}, \{4\}\rangle \quad \langle\{5; 6\}, \{1; 4\}, \{2; 3\}\rangle.$$

La mutation est simple avec l'AGR : un élément est déplacé d'un groupe à l'autre. Toutefois, l'opérateur de recombinaison est plus complexe. Nous choisissons une *section de permutation* dans chaque parent, soit

$\langle\{1\}, \{3; 6\}\rangle$ pour le 1er parent et $\langle\{1; 4\}\rangle$ pour le second. Nous *injectons* la 1re section de permutation dans le 2e parent à un point aléatoire et vice versa :

$$\langle\{1\}, \{3; 6\}, \underline{\{1; 4\}}, \{2; 5\}, \{4\}\rangle \quad \langle\{5; 6\}, \{1; 4\}, \underline{\{1\}}, \underline{\{3; 6\}}, \{2; 3\}\rangle.$$

Nous retranchons ensuite tout objet répété se trouvant déjà dans le parent récepteur :

$$\langle\varnothing, \{3; 6\}, \{1; 4\}, \{2; 5\}, \varnothing\rangle \quad \langle\{5\}, \{4\}, \{1\}, \{3; 6\}, \{2\}\rangle.$$

Enfin, nous retirons tout ensemble vide :

$$\langle\{3; 6\}, \{1; 4\}, \{2; 5\}\rangle \quad \langle\{5\}, \{4\}, \{1\}, \{3; 6\}, \{2\}\rangle.$$

C'est là la descendance. Il est clair que les deux fils ont beaucoup en commun avec les deux parents, puisque 5 des 7 groupes des parents sont préservés chez les fils, à savoir : $\{1\}, \{4\}, \{1; 4\}, \{2; 5\}$ et $\{3; 6\}$. Dans le premier exemple à la section 2.1, nous pouvons facilement vérifier que les deux fils représentent le même regroupement que chez les parents, comme on pouvait s'y attendre. Cette propriété de la recombinaison à injection dans l'AGR rend bien plus probable la similitude d'adaptation entre la descendance et l'ascendance, ce qui aide à son tour l'AGR à améliorer itérativement la population de chromosomes.

On pourrait observer que la représentation AGR du problème comporte toujours de l'asymétrie : tout regroupement demeure à représentations multiples par permutation des sous-ensembles de la liste ordonnée, mais les opérateurs génétiques sont quasiment indépendants de cet ordre qui n'a presque plus rien à voir. Le seul effet de cet ordre est de limiter le jeu d'injections possibles : dans le second exemple à la section 2.1, il est impossible d'injecter une section de permutation inexistante comme $\langle\{1\}, \{4\}\rangle$ du parent 1, parce que ces deux groupes ne sont pas adjacents. Nous pouvons supprimer cette limite à l'aide d'un opérateur génétique supplémentaire appelé *inversion*, lequel choisit une section du chromosome et l'inverse. Un exemple en est

$$\langle\{1\}, \{2\}, \underline{\{3; 6\}}, \underline{\{4\}}, \underline{\{5\}}\rangle \rightarrow \langle\{1\}, \{2\}, \underline{\{5\}}, \underline{\{4\}}, \underline{\{3; 6\}}\rangle.$$

Cela ne change pas le regroupement représenté par le chromosome, mais si on réordonne les groupes de ce chromosome, toutes les injections sont possibles.

L'injection, la mutation et l'inversion sont les opérateurs courants des AGR, mais il n'existe pas d'algorithme canonique. Les AGR tendent plutôt à s'ajuster à leurs usages particuliers et, en principe, tout AG peut s'adapter à un problème de regroupement par le choix des opérateurs. À la section 2.3, nous concevons un AGR en adaptation à notre problème.

2.2.1 À propos de la mise en œuvre

Pour plus de clarté, nous omettons les détails de mise en œuvre dans les descriptions de la section 2.2 et oublions, par exemple, que les chromosomes AGR se traitent habituellement en deux parties (et parfois plus). Il y a représentation classique comme plus haut, d'une part, puis liste des groupes non vides en

permutation, d'autre part. L'injection intervient en seconde partie sur les chromosomes parents et une certaine réidentification des groupes est nécessaire.

En temps normal, nous décidons d'avance du nombre d'itérations dans l'exécution de l'algorithme. Leur nombre doit être tel que l'AGR puisse converger vers la solution optimale après application des probabilités de mutation et d'inversion. Si la solution optimale est connue d'avance, nous pouvons régler l'algorithme pour qu'il s'arrête à ce point d'optimalité.

Nous déterminons ordinairement le nombre d'itérations par l'expérience acquise dans l'application de l'AGR à des variables cibles et auxiliaires semblables pour les mêmes ensembles de données ou avec l'ensemble de données et les variables cibles et auxiliaires propres au problème à résoudre. Nous pouvons avoir à expérimenter avec l'AGR (ou l'AG) avant de pouvoir estimer le nombre d'itérations nécessaires pour que la convergence s'opère. En fait, il est possible que l'AGR ou l'AG paraisse converger après un nombre donné d'itérations, mais que l'algorithme soit plutôt coincé dans un minimum local. Il serait bon que l'on hausse le nombre d'itérations et essaie diverses probabilités de mutation pour être sûr de converger vers un minimum global.

Cela implique qu'un certain nombre d'essais aient lieu avant le choix définitif des paramètres d'exécution de l'algorithme. Ainsi, que l'on sache que l'AGR convergera plus rapidement que l'AG s'ajoutera probablement comme facteur complexe dans l'amélioration de la durée totale du traitement. Dans les expériences que nous allons décrire, nous allons garder petit le nombre d'itérations, car nous voulons démontrer la convergence possible de l'AGR vers une solution dans ce nombre donné d'itérations.

Nous prenons alors les paramètres de mutation des exemples cités par Ballin et Barcaroli (2013) ou les paramètres par défaut dans Barcaroli (2014). Nous appliquons les opérateurs génétiques de regroupement avec inversion à l'AG conçu par Ballin et Barcaroli (2013) : ce sont là les opérateurs génétiques de regroupement d'un AGR. Nous comparons dans ce cas en rendement les différents opérateurs génétiques AG et AGR plutôt que d'expérimenter le paramétrage en variant le nombre d'itérations, la taille de la population de chromosomes, la probabilité de mutation ou le taux d'élitisme.

L'utilisateur peut choisir d'avance la probabilité de mutation. En temps normal, celle-ci devrait être telle qu'elle accroisse les chances que l'AGR décolle d'un minimum local sans dérégler l'évolution naturelle des chromosomes d'une génération à l'autre. Par ailleurs, nous avons arrêté la probabilité d'inversion à 0,01, parce que cela suffit au maintien de la diversité.

On peut décider par tâtonnement de la taille de la population de chromosomes. Il est souhaitable de s'attacher à la durée d'évaluation de chaque chromosome lorsqu'on établit cette taille : si les chromosomes sont trop nombreux dans l'ensemble, le laps de temps pourrait être démesuré lorsqu'on passe d'une itération à l'autre. Nous avons constaté que l'algorithme *bethel.r* (algorithme d'évaluation de Bethel-Chromy dans Barcaroli (2014)) prend plusieurs secondes à évaluer fût-ce un seul chromosome dans les ensembles de données plus abondants que nous allons employer (pour plus de détails, voir la section 4).

Nous renvoyons le lecteur à des études comme celle de Falkenauer (1998) pour un complément d'information sur la mise en œuvre des AGR (taux d'élitisme, par exemple).

2.3 Application au problème de stratification et de répartition de l'échantillon en combinaison

Comme nous l'avons mentionné, notre AGR est tiré de l'AG décrit dans Ballin et Barcaroli (2013) et représenté en R dans le paquet *SamplingStrata* (Barcaroli, 2014), mais avec des chromosomes et des opérateurs de regroupement au lieu des versions classiques. Ce changement est la seule nouveauté dans notre algorithme (sauf pour l'optimisation décrite à la section 4), mais son effet sur le rendement est marqué. Nous avons inséré cet AGR dans une version modifiée de la fonction appelée *rbga.r* dans le paquet *genalg* en R (Willighagen, 2005). Il est conçu pour s'agencer avec les autres fonctions dans *SamplingStrata* et s'appliquer au problème d'optimisation de stratification et de taille d'échantillon en combinaison. Il est résumé à la figure 2.1.

Suivant l'énoncé du problème dans Ballin et Barcaroli (2013), nous récapitulons ainsi la fonction de coût :

$$C(n_1, \ldots, n_H) = C_0 + \sum_{h=1}^{H} C_h n_h,$$

où C_0 est le coût fixe et C_h le coût moyen de l'interview d'une unité dans la strate h et où n_h est le nombre d'unités ou l'échantillon attribué à la strate h. Dans notre analyse, C_0 est fixé à 0 et C_h à 1. L'espérance de l'estimateur du « g^{e} » total de population est :

$$E\left(\hat{T}_g\right) = \sum_{h=1}^{H} N_h \overline{Y}_{h,g} \quad (g = 1, \ldots, G),$$

où $\overline{Y}_{h,g}$ est la moyenne des G variables cibles Y dans chaque strate h. La variance de l'estimateur est :

$$\text{VAR}\left(\hat{T}_g\right) = \sum_{h=1}^{H} N_h^2 \left(1 - \frac{n_h}{N_h}\right) \frac{S_{h,g}^2}{n_h} \quad (g = 1, \ldots, G). \tag{2.1}$$

La limite supérieure de variance ou de précision U_g s'exprime comme coefficient de variation (CV) pour chaque $\hat{T}_g$:

$$\text{CV}\left(\hat{T}_g\right) = \frac{\sqrt{\text{VAR}\left(\hat{T}_g\right)}}{E\left(\hat{T}_g\right)} \leq U_g. \tag{2.2}$$

Le problème se résume ainsi :

$$\begin{aligned} \min n &= \sum_{h=1}^{H} n_h \\ \text{CV}\left(\hat{T}_g\right) &\leq U_g. \end{aligned}$$

Algorithme génétique de regroupement (AGR)

Étape 1 : Initialisation

a) Créer aléatoirement une population de chromosomes de taille N_P.

Étape 2 : Sélection, partie 1

a) Ranger les chromosomes selon la taille d'échantillon.
b) Préserver les meilleurs chromosomes E pour la prochaine génération.

Étape 3 : Inversion

Inverser les groupes dans les chromosomes N_P avec la probabilité 0,01.

Étape 4 : Sélection, partie 2

Pour chacun des chromosomes restants $N_P - E$ dans la nouvelle génération :

a) Tirer les parents 1 et 2 des chromosomes N_P en question (les chromosomes de rang supérieur ont une plus grande probabilité de sélection).
b) Exécuter la permutation comme à la section 2.2.
c) Retrancher les groupes vides.
d) Renuméroter les groupes.

Étape 5 : Mutation

Opérer une mutation d'entiers dans les chromosomes $N_P - E$ à une probabilité choisie.

Étape 6 : Si le nombre d'itérations est inférieur au maximum

(et à titre facultatif si la taille d'échantillon est supérieure à la valeur désirée), passer à l'étape 2.

Figure 2.1 Pseudocode de notre AGR.

3 Comparaison des algorithmes génétiques

Comparons maintenant l'AG initial et notre AGR à l'aide d'ensembles de données appartenant au domaine public. Sauf avis contraire, nous adoptons dans tous les cas que nous présentons les paramètres suivants pour les deux algorithmes génétiques : $N_P = 20$; $U_g \equiv 0{,}05$; taux d'élitisme : 0,2; probabilité de mutation : 0,05.

3.1 Comparaison de l'ensemble de données sur l'iris

(Ballin et Barcaroli, 2013) utilisent l'ensemble de données sur l'iris (Anderson, 1935; Fisher, 1936; R Core Team, 2015) pour démontrer que, par l'AG qu'ils proposent, ils peuvent parvenir à la stratification optimale, c'est-à-dire à une stratification ou à un regroupement de strates atomiques qui donne une taille minimale d'échantillon. Cet ensemble est modeste et largement accessible. Il compte 150 observations pour cinq variables, à savoir la longueur et la largeur du sépale, la longueur et la largeur du pétale et l'espèce.

L'espèce est une variable catégorique à trois niveaux, iris soyeux, versicolore et de Virginie, chacun étant de 50 observations. Les quatre autres variables sont continues pour la longueur et la largeur en centimètres. Ballin et Barcaroli (2013) choisissent la longueur et la largeur du pétale comme variables d'intérêt, c'est-à-dire comme variables cibles. La longueur du sépale et l'espèce sont deux variables auxiliaires.

Ils convertissent la longueur du sépale en variable catégorique au moyen d'un algorithme à K moyennes (Hartigan et Wong, 1979) de manière à définir trois grappes (de 4,3 à moins de 5,5, de 5,5 à moins de 6,5, de 6,5 à 7,9). Le produit vectoriel entre l'espèce et la variante catégorique de la longueur du sépale engendre

neuf strates atomiques. Il reste qu'une strate atomique est vide et sans valeurs correspondantes pour la longueur et la largeur du pétale. Il n'y a donc que huit strates atomiques exploitables dans cet exemple.

Tableau 3.1
Reproduction du tableau de strates atomiques dans l'estimation de la taille minimale d'échantillon pour les variables cibles de l'ensemble de données sur l'iris dans Ballin et Barcaroli (2013), page 379

Strate	N	M1	M2	S1	S2	X1	X2	DOMAINE
[4,3; 5,5] (1)*soyeux	45	1,466667	0,244444	0,17127	0,106574	[4,3; 5,5] (1)	soyeux	1
[4,3; 5,5] (1)*versicolore	6	3,583333	1,166667	0,491313	0,205481	[4,3; 5,5] (1)	versicolore	1
[4,3; 5,5] (1)*de Virginie	1	4,5	1,7	0	0	[4,3; 5,5] (1)	de Virginie	1
[5,5; 6,5] (2)*soyeux	5	1,42	0,26	0,172047	0,08	[5,5; 6,5] (2)	soyeux	1
[5,5; 6,5] (2)*versicolore	35	4,268571	1,32	0,367051	0,189435	[5,5; 6,5] (2)	versicolore	1
[5,5; 6,5] (2)*de Virginie	23	5,230435	1,947826	0,318194	0,28873	[5,5; 6,5] (2)	de Virginie	1
[6,5; 7,9] (3)*versicolore	9	4,677778	1,455556	0,193091	0,106574	[6,5; 7,9] (3)	versicolore	1
[6,5; 7,9] (3)*de Virginie	26	5,876923	2,107692	0,494825	0,228579	[6,5; 7,9] (3)	de Virginie	1

Nous reproduisons les strates atomiques initiales au tableau 3.1, où M_g est la moyenne pour les valeurs Y_g correspondantes dans chaque strate atomique l_k et où S_g est l'écart-type de population de cette strate. Il y a 4 140 partitions possibles des huit strates atomiques. Il est donc possible de vérifier dans un laps de temps raisonnable la taille d'échantillon dans tout l'espace de recherche à l'aide de la fonction *bethel.r*. Cela est déjà fait (Ballin et Barcaroli, 2013), et nous savons que la taille minimale d'échantillon est de 11.

Ce test peut servir à établir si, avec le nouvel AG, nous trouvons fidèlement la taille minimale sans avoir à explorer tout l'espace de recherche. Nous employons $N_p = 10$ dans ce cas. La fonction *bethel.r* recherchera alors la taille minimale d'échantillon en nombres entiers plutôt qu'en nombres réels. Nous rangerons ensuite les chromosomes par taille d'échantillon en ordre croissant. Ainsi, nous reporterons les chromosomes de l'élite à la nouvelle itération et créerons les chromosomes restants par la méthode de recombinaison pour chaque algorithme.

Nous comparerons les nombres de chromosomes produits pour dégager la stratification optimale avec les deux algorithmes, tout comme le nombre d'itérations. Nous prévoyons que l'AGR sera plus efficace et mènera d'ordinaire à la solution optimale en moins d'itérations qu'avec l'AG.

Nous fixons le maximum d'itérations à 200, car si nous nous guidons sur Ballin et Barcaroli (2013), nous pouvons prévoir que les deux algorithmes trouveront la bonne solution en moins de 200 itérations. Nous avons donc ajouté un élément de code aux deux algorithmes pour qu'ils s'arrêtent lorsque la taille optimale d'échantillon, $n = 11,$ est atteinte et qu'ils indiquent le nombre d'itérations exécutées jusqu'à ce point d'optimalité. Notre traitement est différent de celui de Ballin et Barcaroli (2013) qui mentionnent le nombre de fois en 10 expériences que l'AG trouve la bonne solution pour un nombre donné d'itérations à pas d'accroissement de 25 à 200. Nous pensons cependant que notre approche démontrera mieux que l'AGR

peut trouver la bonne solution en moins d'itérations même dans l'expérience avec le petit ensemble de données sur l'iris.

Tableau 3.2
Résultats de l'expérience de l'ensemble de données sur l'iris avec l'AG et l'AGR

Nombre	a) AG			b) AGR		
	d'expériences	d'itérations	de chromosomes	d'expériences	d'itérations	de chromosomes
	1	14	228	1	11	180
	2	8	132	2	7	116
	3	17	276	3	6	100
	4	40	644	4	22	356
	5	31	500	5	9	148
	6	13	212	6	11	180
	7	15	244	7	8	132
	8	9	148	8	7	116
	9	15	244	9	9	148
	10	15	244	10	11	180
	11	14	228	11	3	52
	12	8	132	12	9	148
	13	17	276	13	27	436
	14	40	644	14	12	196
	15	31	500	15	16	260
	16	13	212	16	6	100
	17	15	244	17	20	324
	18	9	148	18	6	100
	19	15	244	19	7	116
	20	15	244	20	6	100
	21	16	260	21	11	180
	22	67	1 076	22	7	116
	23	19	308	23	8	132
	24	9	148	24	5	84
	25	11	180	25	7	116
	26	20	324	26	5	84
	27	32	516	27	6	100
	28	10	164	28	6	100
	29	37	596	29	9	148
	30	9	148	30	6	100

Le tableau 3.2 indique le nombre d'itérations (et de chromosomes produits) ayant permis de trouver $n = 11$ en 30 expériences avec les deux AG.

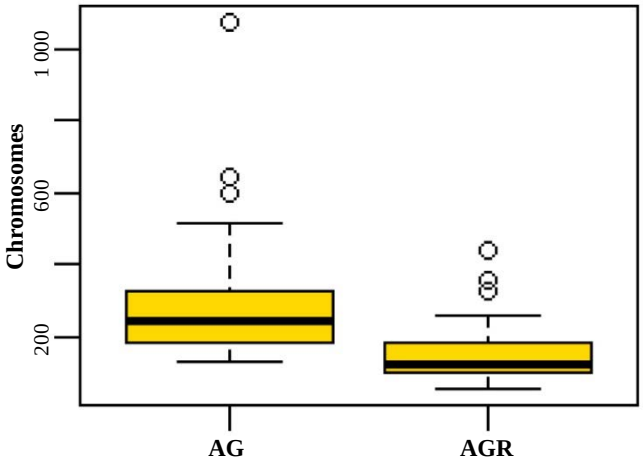

Figure 3.1 Distribution en diagramme de quartiles du nombre de chromosomes produits pour trouver $n = 11$ après 30 expériences avec l'AG et l'AGR.

La figure 3.1 présente la distribution du nombre de chromosomes produits en vue de trouver la solution optimale avec l'AG et l'AGR. Les diagrammes en quartiles indiquent que l'AGR aura normalement à produire moins de chromosomes pour trouver la solution optimale.

Tableau 3.3
Exemple de stratifications avec l'AG et l'AGR pour l'ensemble donné sur l'iris et $n = 11$

	Strate	Y1			Y2		
		N	Moyenne	E.-T.	Moyenne	E.-T.	Taille d'échantillon
AG	1	50	1,462	0,1685	0,246	0,1026	2
	2	50	4,26	0,4562	1,326	0,1911	3
	3	1	4,5	0	1,7	0	1
	4	23	5,2304	0,3112	1,9478	0,2824	3
	5	26	5,8769	0,4852	2,1077	0,2241	2
Total		150					11
AGR	1	23	5,2304	0,3112	1,9478	0,2824	3
	2	50	1,462	0,1685	0,246	0,1026	2
	3	26	5,8769	0,4852	2,1077	0,2241	2
	4	51	4,2647	0,4529	1,3333	0,1962	4
Total		150					11

Le tableau 3.3 illustre les stratifications avec l'AG et l'AGR dans la recherche de la taille optimale d'échantillon nécessaire au respect des contraintes de précision. Ballin et Barcaroli (2013) indiquent qu'un certain nombre de partitions sur les 4 140 partitions possibles mène à la taille minimale d'échantillon. Ces partitions sont d'une taille de 3 à 5 strates. On peut voir que l'AGR crée dans un plan de sondage des strates moins nombreuses et moins fragmentées. La même tendance peut s'observer dans les cas qui suivent.

3.2 Ensemble de données sur les municipalités suisses

L'ensemble de données présenté par Barcaroli (2014) porte sur les municipalités suisses en 2003. Chaque municipalité appartient à une de sept régions au niveau 2 de la Nomenclature des unités territoriales statistiques (NUTS), lequel correspond à l'échelon provincial. Chaque région compte un certain nombre de cantons ou de subdivisions administratives. On en dénombre 26 en Suisse. Les données, qui émanent de l'Office fédéral de la statistique suisse et figurent dans les paquets *sampling* et *SamplingStrata*, se composent de 2 896 observations (dont chacune vise une municipalité suisse en 2003). Il s'agit de 22 variables qui peuvent être examinées en détail dans Barcaroli (2014).

Les estimations cibles sont les totaux de population par catégorie d'âge dans chaque région suisse. Dans ce cas, les G variables cibles seront les suivantes :

Y1 : nombre d'hommes et de femmes âgés de 0 à 19 ans;
Y2 : nombre d'hommes et de femmes âgés de 20 à 39 ans;
Y3 : nombre d'hommes et de femmes âgés de 40 à 64 ans;
Y4 : nombre d'hommes et de femmes âgés de 65 ans et plus.

Nous considérons six variables auxiliaires formées par la même méthode de classification automatique à K moyennes que pour l'ensemble de données sur l'iris :

X1 : catégories de population totale dans la municipalité : 18;
X2 : catégories de superficie boisée dans la municipalité : 3;
X3 : catégories de superficie cultivée dans la municipalité : 3;
X4 : catégories de superficie en pâturage de montagne dans la municipalité : 3;
X5 : catégories de superficie bâtie dans la municipalité : 3;
X6 : catégories de superficie industrielle dans la municipalité : 3.

Nous traitons sept régions comme domaines de population dans le plan de sondage pour les distinguer des strates dans ce même plan, ce qui reproduit l'expérience décrite dans Barcaroli (2014). Le nombre de strates atomiques non vides est de 641 dans la population. Nous fixons la taille minimale de population de strates à deux et le nombre maximal d'itérations à 400. La figure 3.2 récapitule les résultats pour la taille d'échantillon et les strates après 30 expériences et dans chaque cas avec 400 itérations.

Figure 3.2 Diagramme de dispersion pour les strates et la taille d'échantillon avec l'AG et l'AGR après 30 expériences.

La figure 3.2 indique clairement que l'AGR donne une taille d'échantillon inférieure à celle de l'AG pour ces paramètres. La médiane de 246 pour l'AGR le cède du quart à la médiane de 328 de l'AG.

3.3 Microdonnées à grande diffusion de l'American Community Survey en 2015

Les États-Unis réalisent un recensement décennal depuis 1790. Au XX^{e} siècle, ce recensement a reçu une version longue et une version courte. Un sous-ensemble de la population était tenu de répondre à la version longue et le reste, à la version courte. À la suite du recensement de l'an 2000, la version longue est devenue la reprise annuelle de l'American Community Survey ou ACS (US Census Bureau, 2013). Le fichier-échantillon de microdonnées à grande diffusion de l'ACS (Public Use Microdata Sample ou PUMS) dans US Census Bureau (2016) est un échantillon des réponses effectives à l'ACS qui correspond à 1 % de la population des États-Unis. Le fichier PUMS compte 1 496 678 enregistrements décrivant chacun un logement individuel ou collectif. Le nombre de variables est de 235. On peut consulter tout le dictionnaire des données dans US Census Bureau (2016). Nous avons choisi les variables cibles suivantes :

1. revenu du ménage (12 derniers mois);
2. valeur foncière;
3. certaines charges de propriété mensuelles;
4. assurance incendie/dommage habitation/inondation (montant annuel),

nos variables auxiliaires sont les suivantes :

1. logements constitutifs;
2. durée d'occupation;
3. expérience de travail du chef de ménage et du conjoint;
4. situation de travail du chef de ménage ou du conjoint dans les ménages familiaux;
5. combustible de chauffage domestique;
6. date de construction initiale.

Le fichier PUMS pour lequel toutes les valeurs sont présentées se compose de 619 747 enregistrements. Les 51 États américains (d'après les définitions du recensement) sont nos domaines.

Dans les courbes de convergence à la figure 3.3, le trait noir représente la taille d'échantillon optimale ou minimale pour la population de chromosomes dans chaque itération et le trait rouge, la taille moyenne d'échantillon correspondante.

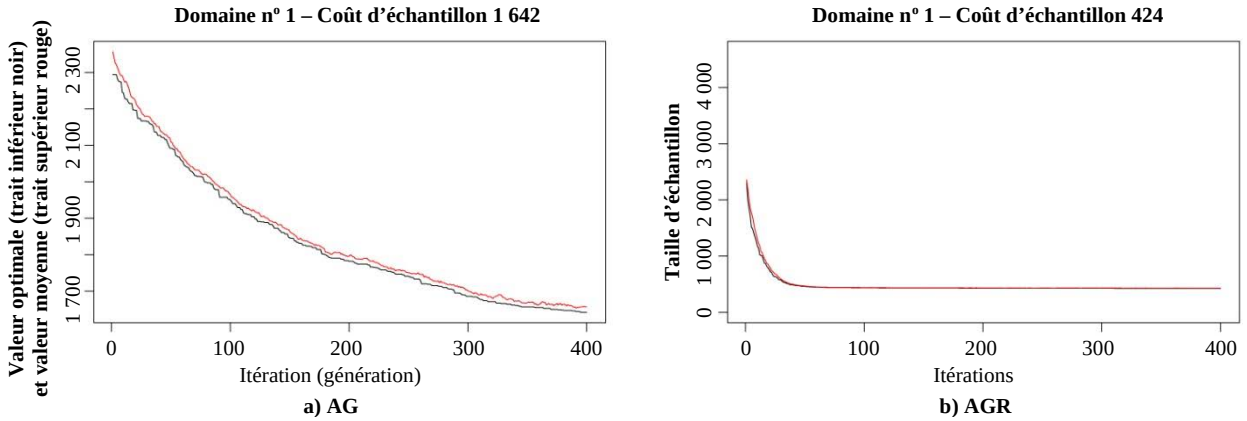

Figure 3.3 Courbes de convergence de la taille d'échantillon après la première expérience avec l'AG et l'AGR. À noter les différences d'échelle dans l'axe vertical.

L'AG semble réduire constamment la taille d'échantillon, mais sans atteindre de minimum local après 400 itérations. L'AGR paraît parvenir très rapidement à un minimum local ou global.

3.4 Données sur les prêts Kiva pour le défi Kaggle Data Science for Good

La plateforme en ligne de sociofinancement kiva.org présente un ensemble de données sur les prêts consentis en deux ans à des gens vivant dans des conditions de pauvreté et de marginalisation financière en vue d'un défi Kaggle Data Science for Good. Cet ensemble compte 671 205 enregistrements uniques. Nous avons choisi les variables cibles suivantes :

1. durée en mois;
2. nombre de prêteurs;
3. montant du prêt,

nos variables auxiliaires sont les suivantes :

1. secteur;
2. monnaie;
3. activité;
4. région;
5. partenariat,

le but est de créer des strates atomiques. Pour les variables en question, nous avons retranché tout enregistrement à valeurs manquantes. Nous avons ensuite retiré tout pays ayant moins de 10 enregistrements dans la base de sondage. La base résultante est de 614 361 enregistrements. La variable de codage de pays définit les 73 domaines du plan de sondage de cette expérience.

Tableau 3.4
Taille d'échantillon et strates dans les données sur les prêts Kiva avec l'AG et l'AGR après 100 itérations

AG		AGR		Réduction	
Taille d'échantillon	**Strates**	**Taille d'échantillon**	**Strates**	**Taille d'échantillon**	**Strates**
78 018	43 030	11 963	1 793	84,67 %	95,83 %

Le tableau 3.4 fait voir une diminution de 84,67 % de la taille d'échantillon et de 95,83 % du nombre de strates après 100 itérations. La figure 3.4 indique que, pour la même taille au départ de population de chromosomes dans le domaine 1 de l'ensemble de données sur les prêts Kiva, l'AGR parvenait à une bonne taille d'échantillon en moins de 100 itérations et que, après 10 000 itérations, l'AG n'avait pas encore convergé et la taille d'échantillon demeurait largement supérieure à celle de l'AGR.

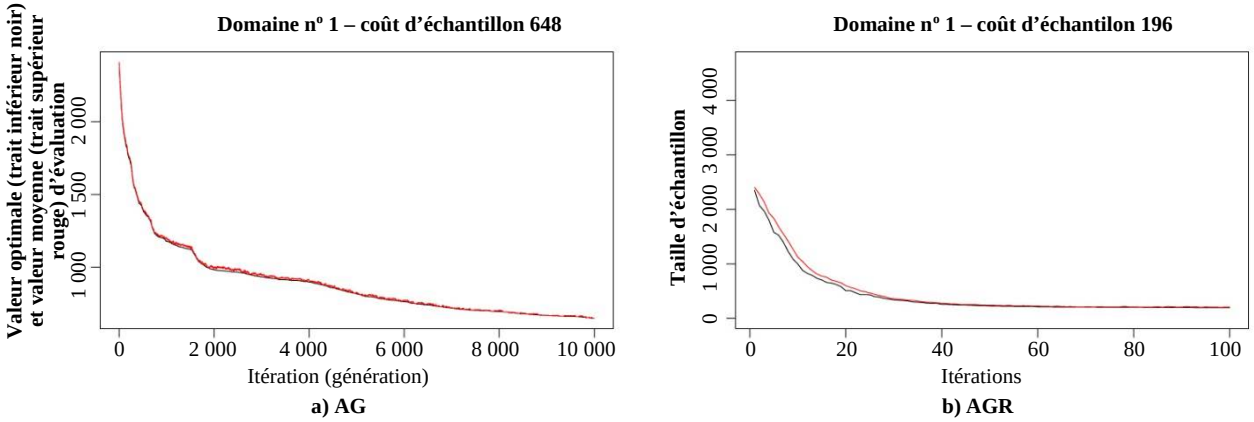

Figure 3.4 Courbes de convergence de la taille d'échantillon pour le premier domaine avec l'AG (10 000 itérations) et l'AGR (100 itérations) dans l'expérience de l'ensemble de données Kiva sur les prêts. À noter les différences d'échelle dans les axes vertical et horizontal.

3.5 Statistiques sur le commerce des produits de base des Nations Unies

Kaggle héberge aussi une copie des statistiques sur le commerce des produits de base de la Division de la statistique des Nations Unies. Celle-ci dispose d'enregistrements sur le commerce depuis 1962. Nous avons prélevé un sous-ensemble de données pour 2011 et retranché les enregistrements à observations manquantes. Nous avons ainsi obtenu un ensemble de données comptant 351 057 enregistrements. Nous avons choisi la variable cible suivante :

1. commerce (trade_usd)

pour la valeur de ce commerce en dollars américains. Nos variables auxiliaires sont les suivantes :

1. produit de base;
2. flux;
3. catégorie.

La variable produit de base est une description catégorique de la nature des produits de base (chevaux, vivants sans les pur-sang, par exemple). La variable flux décrit s'il y a importation, exportation, réimportation ou réexportation du produit de base. La variable catégorie décrit la catégorie (soie, engrais, etc.). Les 171 catégories de pays ou de régions ont été choisies comme domaines.

Tableau 3.5
Taille d'échantillon et strates des statistiques sur le commerce des produits de base des Nations Unies avec l'AG et l'AGR après 100 itérations

AG		AGR		Réduction	
Taille d'échantillon	**Strates**	**Taille d'échantillon**	**Strates**	**Taille d'échantillon**	**Strates**
288 638	191 000	84 181	16 555	70,84 %	91,33 %

3.6 Données du recensement américain de l'an 2000

L'extrait de l'Integrated Public Use Microdata Series ou IPUMS est un échantillon à 5 % des données du recensement américain de l'an 2000 (Ruggles, Genadek, Goeken, Grover et Sobek, 2017). C'est un fichier de 6 184 483 enregistrements. Les données en question ressemblent fort aux données de l'ACS, puisque celle-ci est une version annuelle de cet ensemble. Pour cette expérience, nous choisissons toutefois des combinaisons différentes de variables cibles et auxiliaires. La seule variable cible dans ce test représente habituellement un grand sujet d'intérêt dans les enquêtes auprès des ménages :

1. revenu total du ménage.

Nos variables auxiliaires sont les suivantes (il s'agit de variables sans doute accessibles dans les données administratives) :

1. coût annuel en assurance sur les biens;
2. coût annuel en combustible de chauffage domestique;
3. coût annuel en électricité;
4. valeur de l'habitation.

La variable de la valeur de l'habitation (VALUEH) présente le point milieu des intervalles de valeur foncière (5 000 est le point milieu de l'intervalle moins de 10 000), aussi en avons-nous fait une variable catégorique. Comme pour l'ensemble de données PUMS de l'ACS en 2015, nous avons prélevé un sous-ensemble pour lequel toutes les valeurs sont présentes. Le sous-ensemble dégagé compte 627 611 enregistrements. La région et la division de recensement forment le domaine dans cette expérience.

Tableau 3.6
Taille d'échantillon et strates pour les données du recensement américain de 2000 par région et division du recensement avec l'AG et l'AGR après 100 itérations

Division	Base de sondage		Solution AG		Solution AGR	
	Unités d'échantillonnage	Strates atomiques	Tailles d'échantillon	Strates	Tailles d'échantillon	Strates
Nouvelle-Angleterre	116 045	87 084	81 012	52 628	376	58
Atlantique centre	183 543	138 470	130 862	86 002	416	75
Centre nord-est	65 480	58 055	53 075	35 794	327	42
Centre nord-ouest	31 408	29 413	26 525	18 248	324	38
Atlantique sud	97 189	83 357	76 716	51 457	440	49
Centre sud-est	21 631	20 429	18 256	12 500	451	62
Centre sud-ouest	22 582	20 919	18 750	12 730	407	39
Rocheuses	26 765	25 041	22 161	14 791	351	30
Pacifique	62 968	54 864	50 136	33 653	358	49
Total	627 611	517 632	477 493	317 803	3 446	442

Les résultats font voir une taille d'échantillon de 3 446 avec l'AGR et de 477 493 avec l'AG après 100 itérations.

4 Mise en œuvre améliorée de l'évaluation de Bethel

Nous avons proposé et mis au point l'AGR pour qu'il puisse s'agencer avec le reste des fonctions dans *SamplingStrata*. Les autres fonctions du paquet sont donc demeurées inchangées. Cela comprend la fonction *bethel.r*, qui évalue l'adaptation des chromosomes dans chaque itération et qui est vorace en calcul. Dans le cas de l'ensemble de données PUMS par exemple, l'expérience a duré une trentaine de jours avec l'AG ou l'AGR pour 100 itérations.

Nous avons étudié les goulots d'étranglement du rendement dans *bethel.r* à l'aide du paquet *lineprof* en R. D'après notre analyse des résultats, la fonction appelée *chromy* dans *bethel.r* semble prendre le plus clair du temps de calcul. En poussant l'examen, nous avons constaté que *chromy* comporte une boucle WHILE paramétrée par défaut à 200 itérations. Ajoutons que *bethel.r* peut s'exécuter sur chaque chromosome de toute population chromosomique dans un ensemble de données de toute taille fonctionnelle (si nous avons la puissance de calcul pour le traiter) et pour tout nombre d'itérations. Plus l'ensemble de données augmente, plus le traitement s'allonge. Nous nous attendions à une amélioration du rendement grâce à une conversion en C++ de l'algorithme *bethel.r*, suivie d'une intégration en R par le paquet *Rcpp* (Eddelbuettel, 2013).

Tableau 4.1
Comparaison de rendement des ensembles de données qui précèdent à l'aide des versions R et Rcpp de l'algorithme de Bethel-Chromy

Ensemble de données	Enregis-trements	Domaines	Strates atomiques	Bethel μs	BethelRcpp μs	Facteur d'accélé-ration
iris	150	1	8	2 684,77	143,13	18,76
municipalités suisses	2 896	7	641	99 916	10 749,51	9,29
American Community Survey 2015	619 747	51	123 007	565 278 500	47 858 200	11,81
Données sur les prêts Kiva	614 361	73	84 897	826 297 710	82 894 480	9,97
Statistiques sur le commerce des produits de base de l'ONU 2011	351 057	171	350 895	139 749 810	87 555 870	1,6
Données du recensement américain de 2000	627 611	9	517 632	2 686 771	1 303 667	2,06

Le tableau 4.1 indique le temps médian d'exécution de l'algorithme de Bethel en cent fois pour les ensembles de données de notre analyse. Nos résultats confirment que la version en C++ de Bethel s'exécute plus rapidement que la version en R. Cette accélération pourrait dans la pratique faire la différence pour le nombre d'itérations pouvant s'exécuter dans *SamplingStrata* à cause des temps de traitement à prévoir pour *bethel.r*. Il reste que le rendement variera selon la taille et la complexité du problème. S'il y a accélération, c'est que, à comparer à *R,* C++ permet une communication à un moindre niveau avec l'ordinateur. Un autre facteur est la complexité de l'analyse effectuée en boucle dans chaque cas, et le fait que des données plus abondantes viendront restreindre la mémoire disponible. À noter aussi que nous avons comparé les versions C++ et R de Bethel comme deux fonctions autonomes. Nous n'avons pas comparé en rendement la version en C++ avec l'AGR et la version en R avec l'AG. Ce serait là un projet plus vaste visant à créer une version en C++ du paquet *SamplingStrata* et à intégrer celle-ci en R.

5 Conclusion et prochaines étapes

Nous avons créé un AGR comme solution de rechange à l'AG de *SamplingStrata* en R, puis comparé les deux algorithmes dans un certain nombre d'ensembles de données. L'AGR se compare favorablement à l'AG s'il s'agit de trouver la bonne solution et de respecter les contraintes avec des ensembles de données de moindre taille, mais il surpasse nettement l'AG avec de plus gros ensembles pour lesquels nous avons dû restreindre le nombre d'itérations. On en voit l'utilité pour les ensembles de données où le nombre d'itérations doit être restreint en raison de la charge de calcul. Nous avons également montré que le traitement est plus rapide grâce à l'intégration de la fonction *bethel.r* en C++ par le paquet Rcpp.

Notre recherche peut se poursuivre de plusieurs manières. Il est possible d'envisager d'autres techniques d'évaluation qui hâteront l'exécution de l'algorithme. Nous pourrions également entreprendre une recherche sur d'autres techniques d'apprentissage automatique pouvant permettre de résoudre notre problème.

Nous pourrions appliquer l'AGR à d'autres problèmes qui se posent avec des plans de sondage plus généraux sous réserve de modifications à prévoir seulement pour l'algorithme évaluant l'adaptation des chromosomes (algorithme de Bethel-Chromy). Au lieu, par exemple, de rechercher un échantillon aléatoire

simple stratifié pour le respect de contraintes de précision qui soit fondé sur des totaux ou des moyennes de population, l'AGR pourrait viser à un échantillonnage stratifié avec probabilité proportionnelle à la taille et à l'application d'un algorithme d'évaluation faisant intervenir des estimateurs (estimateurs de régression ou estimateurs de rapport, par exemple) ou paramètres (coefficient de corrélation, par exemple) plus généraux.

Nous pourrions en outre modifier l'algorithme d'évaluation et examiner des scénarios où les variances de population sont inconnues, auquel cas les données de recensements antérieurs, les dossiers administratifs ou des enquêtes par procuration pourraient servir à estimer la variance de population. Mentionnons néanmoins qu'estimer la variance de population avec un grand nombre de strates atomiques exigera une recherche plus attentive.

Disons enfin que les regroupements de strates atomiques opérés par l'AGR peuvent être d'une interprétation difficile. Ainsi, une variable auxiliaire ordinale aux valeurs de 1 à 4 peut se diviser d'une manière peu naturelle où les strates atomiques correspondant aux valeurs 1 et 3 et aux valeurs 2 et 4 formeraient des strates distinctes dans le plan de sondage. Il serait bon de songer à des tailles d'échantillon moins qu'optimales pour une stratification qui soit d'une interprétation plus facile. Nous pourrions, par exemple, imposer des contraintes aux regroupements admissibles, ce qui exigerait une recherche sur la formulation de contraintes appropriées d'admissibilité et sur leur mise en œuvre efficace avec l'AGR.

Remerciements

Nous désirons remercier Steven Riesz, de l'Economic Statistical Methods Division, et Brian J. McElroy, de l'Economic Reimbursable Survey Division, au sein du U.S Census Bureau qui, l'un et l'autre, ont répondu à nos questions au moment de choisir un ensemble de données de cet organisme statistique. Nous remercions également Giulio Barcaroli et Marco Ballin, coauteurs de Ballin et Barcaroli (2013), pour une vérification indépendante de notre AGR. Nos derniers remerciements et non les moindres vont au comité de rédaction et aux examinateurs de *Techniques d'enquête* pour leurs suggestions constructives à la suite de l'examen de l'article que nous leur avons soumis, et en particulier pour ce qu'ils proposent comme prochaines étapes.

Bibliographie

Agustín-Blas, L.E., Salcedo-Sanz, S., Vidales, P., Urueta, G. et Portilla-Figueras, J.A. (2011). Near optimal citywide WiFi network deployment using a hybrid grouping genetic algorithm. *Expert Systems with Applications*, 38(8), 9543-9556.

Anderson, E. (1935). The irises of the gaspe peninsula. *Bulletin of the American Iris society*, 59, 2-5.

Ballin, M., et Barcaroli, G. (2013). Détermination conjointe de la stratification et de la répartition optimales de l'échantillon en utilisant un algorithme génétique. *Techniques d'enquête*, 39, 2, 405-432. Article accessible à l'adresse https://www150.statcan.gc.ca/n1/fr/pub/12-001-x/2013002/article/11884-fra.pdf.

Barcaroli, G. (2014). SamplingStrata: An R package for the optimization of stratified sampling. *Journal of Statistical Software*, 61(4), 1-24.

Barcaroli, G. (2019). Optimization of sampling strata with the SamplingStrata package. https://cran.r-project.org/web/packages/SamplingStrata/vignettes/SamplingStrata.html, consulté le 29 avril 2019.

Bethel, J.W. (1985). An optimum allocation algorithm for multivariate surveys. *Proceedings of the Survey Research Section*, American Statistical Association, 209-212. https://www.overleaf.com/project/5ae8997d310d9a2939f40335.

Bethel, J. (1989). Répartition de l'échantillon dans les enquêtes à plusieurs variables. *Techniques d'enquête*, 15, 1, 49-60. Article accessible à l'adresse https://www150.statcan.gc.ca/n1/fr/pub/12-001-x/1989001/article/14578-fra.pdf.

Brown, E.C., et Vroblefski, M. (2004). A grouping genetic algorithm for the microcell sectorization problem. *Engineering Applications of Artificial Intelligence*, 17(6), 589-598.

Chromy, J.R. (1987). Design optimization with multiple objectives. *Proceedings of the Survey Research Section*, American Statistical Association.

De Lit, P., Falkenauer, E. et Delchambre, A. (2000). Grouping genetic algorithms: An efficient method to solve the cell formation problem.

Eddelbuettel, E. (2013). Seamless R and C++ Integration with Rcpp, *ISBN 978-1-4614-6867-7* 10.1007/978-1-4614-6868-4.

Falkenauer, E. (1998). *Genetic Algorithms and Grouping Problems*. New York: John Wiley & Sons, Inc.

Fisher, R.A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(2), 179-188.

Galinier, P., et Hao, J.K. (1999). Hybrid evolutionary algorithms for graph coloring. *Journal of Combinatorial Optimization*, 3(4), 379-397.

Hartigan, J.A., et Wong, M.A. (1979). Hybrid evolutionary algorithms for graph coloring.algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society*, *Series C* (Applied Statistics), 28(1), 100-108.

Hung, C., Sumichrast, R.T. et Brown, E.C. (2003). CPGEA: A grouping genetic algorithm for material cutting plan generation. *Computers & Industrial Engineering*, 44(4), 651-672.

James, T., Brown, E. et Ragsdale, C.T. (2010). Grouping genetic algorithm for the blockmodel problem. *IEEE Transactions on Evolutionary Computation*, 14(1), 103-111.

Pelikan, M., et Goldberg, D.E. (2000). Genetic algorithms, clustering, and the breaking of symmetry. *Proceedings of the Sixth International Conference on Parallel Problem Solving from Nature*.

Prügel-Bennett, A. (2004). Symmetry breaking in population-based optimization. *IEEE Transactions on Evolutionary Computation*, 8(1), 63-79.

R Core Team (2015). *R A Language and Environment for Statistical Computing*. Vienne, Autriche: R Foundation for Statistical Computing. https://www.R-project.org/.

Ruggles, S., Genadek, K., Goeken, R., Grover, J. et Sobek, M. (2017). Integrated public use microdata series: Version 7.0 [dataset]. Minneapolis: University of minnesota.

U.S. Census Bureau (2013). *American Community Survey Information Guide*. http://www.census.gov/content/dam/Census/programs-surveys/acs/about/ACS_Information_Guide.pdf, consulté le 15 février 2017.

U.S. Census Bureau (2016). *2015 ACS PUMS DATA DICTIONARY*. http://www2.census.gov/programs-surveys/acs/tech_docs/pums/data_dict/PUMSDataDict15.pdf, consulté le 15 février 2017.

U.S. Census Bureau (2016). *2015 ACS Public Use Microdata Sample (PUMS)*. Washington, D.C. https://factfinder.census.gov/faces/nav/jsf/pages/searchresults.xhtml?refresh=t#.

Willighagen, E. (2005). Genalg: R based genetic algorithm. *R Package Version 1*.