COMMUNAUTE FRANÇAISE DE BELGIQUE
ACADEMIE UNIVERSITAIRE WALLONIE-EUROPE
UNIVERSITÉ DE LIÈGE – GEMBLOUX AGRO-BIO TECH

COMPUTATION AND APPROXIMATION OF THE INVERSE OF RELATIONSHIP MATRICES BETWEEN GENOTYPED ANIMALS:

ALGORITHMS AND APPLICATIONS

ESSAI PRESENTE PAR PIERRE FAUX
EN VUE DE L'OBTENTION DU GRADE DE DOCTEUR EN SCIENCES AGRONOMIQUES
ET INGENIERIE BIOLOGIQUE

PROMOTEUR : NICOLAS GENGLER

2014

COMMUNAUTE FRANÇAISE DE BELGIQUE
ACADEMIE UNIVERSITAIRE WALLONIE-EUROPE
UNIVERSITÉ DE LIÈGE – GEMBLOUX AGRO-BIO TECH

COMPUTATION AND APPROXIMATION OF THE INVERSE OF RELATIONSHIP MATRICES BETWEEN GENOTYPED ANIMALS:

ALGORITHMS AND APPLICATIONS

ESSAI PRESENTE PAR PIERRE FAUX
EN VUE DE L'OBTENTION DU GRADE DE DOCTEUR EN SCIENCES AGRONOMIQUES
ET INGENIERIE BIOLOGIQUE

PROMOTEUR : NICOLAS GENGLER

2014

Faux Pierre. (2014). Computation and approximation of the inverse of relationship matrices between genotyped animals: Algorithms and Applications. (PhD Dissertation in English). Gembloux, Belgium, Gembloux Agro-Bio Tech - University of Liège, 139p., 21 tabl., 24 fig.

Abstract

The recent developments in molecular biology have made available thousands of genetic markers, allowing livestock genotyping at a reasonable cost and the subsequent development of genomic prediction. The single-step procedure, a unified approach of genomic prediction, requires inversion of two matrices gathering additive relationships between genotyped animals: the genomic relationship matrix (**G**) and a part of the additive relationship matrix ($\mathbf{A}_{22}$). The inverse of $\mathbf{A}_{22}$ may also be interesting for other applications. Matrix inverse can be constructed successively by, first, computing, for each animal, the vector containing contributions of other animals to its relationship and, secondly, adding the product of each vector of contributions by itself to a zeroed matrix. The objectives of this thesis were (1) to propose algorithms to compute or to approximate the vector of contributions and (2) to test the numerical efficiency of these algorithms (computing speed, memory use and, if needed, approximation accuracy). Computing contributions covered two points: (1) finding or approximating which contributions are different from zero, and (2) computing the value of contributions considered as non-zero. In the first approach, we considered that animals closely related have non-zero contributions and approximated their values by linear regression. This approach was extended in a recursive way. In the second approach, we empirically determined the set of non-zero contributions by a heuristic algorithm of pedigree exploration (only for the case of $\mathbf{A}_{22}$). Values were then computed either by linear regression, or using the already computed inverse. We also tested an approximation strategy: limiting the number of extracted generations of non-genotyped ancestors to reduce pedigree complexity. In a third approach, we followed the same heuristic algorithm as before but restricted the pedigree exploration to find out which animals have a non-zero contribution. Their values were approximated by linear regression. The presentation of the different approaches is followed by a general discussion in which the approaches are compared. It was found that the best compromise between speed, memory and approximation accuracy was achieved by the last approach for the case of $\mathbf{A}_{22}$. Use of this last approach simplified computations and therefore made predictions more feasible. However, for the case of **G**, no sufficient approximations could be reach in a reasonable time. Perspectives of other uses of algorithms developed and of future researches were drawn, as well as practical perspectives for animal breeding.

Faux Pierre. (2014). Construction et approximation de l'inverse de matrices de parenté entre animaux génotypés : algorithmes et applications. (Thèse de doctorat en Anglais). Gembloux, Belgique, Gembloux Agro-Bio Tech - Université de Liège, 139p., 21 tabl., 24 fig.

Résumé

De récents développements en biologie moléculaire ont rendu disponibles des milliers de marqueurs génétiques, permettant de génotyper les animaux de rentes à un coût modéré et, conséquemment, le développement de la prédiction génomique. La procédure « single-step », une approche unifiée de prédiction génomique, requiert l'inversion de deux matrices rassemblant les parentés additives entre animaux génotypés : la matrice de parenté génomique (**G**) et une partie de la matrice de parenté additive ($\mathbf{A}_{22}$). L'inverse de $\mathbf{A}_{22}$ peut également être intéressante pour d'autres applications. L'inverse d'une matrice peut être construite successivement : premièrement, en calculant, pour chaque animal, le vecteur qui contient les contributions des autres animaux aux parentés de celui-ci et, deuxièmement, en ajoutant le produit de chaque vecteur de contributions par lui-même à une matrice nulle. Les objectifs de cette thèse sont (1) de proposer des algorithmes pour construire ou approximer le vecteur des contributions et (2) de tester l'efficience numérique de ces algorithmes (temps de calcul, usage de mémoire et, si nécessaire, précision de l'approximation). Calculer les contributions recouvre deux aspects : (1) trouver ou approximer quelles sont les contributions non-nulles, et (2) calculer les contributions considérées comme non-nulles. Dans la première approche, nous considérons que les animaux fortement apparentés ont des contributions non nulles et approximons leur valeur par régression linéaire. Cette approche est étendue dans une implémentation récursive. Dans la seconde approche, nous déterminons empiriquement l'ensemble de contributions non-nulles par un algorithme heuristique d'exploration du pedigree (seulement pour le cas de $\mathbf{A}_{22}$). Les valeurs sont alors calculés soit par régression linéaire on en utilisant l'inverse déjà calculé à ce point. Nous testons aussi une stratégie d'approximation : limiter le nombre de générations extraites d'ancêtres non-génotypés pour réduire la complexité du pedigree. Dans une troisième approche, nous suivons le même algorithme heuristique qu'avant mais restreignons l'exploration du pedigree pour trouver quels animaux ont une contribution non-nulle. Leur valeur est approximée par régression linéaire. La présentation des différentes approches est suivie par une discussion générale dans laquelle les approches sont comparées. Il a été montré que le meilleur compromis entre consommation de temps et de mémoire et justesse de l'approximation était réalisé par la dernière approche dans le cas de $\mathbf{A}_{22}$. Utiliser cette dernière approche simplifie les calculs et donc rend les prédictions plus faisables. Cependant, pour le cas de **G**, aucune approximation suffisante n'a pu être atteinte en un temps raisonnable. Des perspectives d'autres utilisations des algorithmes et de futures recherches sont esquissées, aussi bien que des perspectives pratiques pour l'amélioration animale.

« Spirou, viens voir la nouvelle idée de Fantasio pour le bal costumé... C’est original, mais si on danse?...»

André Franquin. *Gaston Lagaffe*.

Acknowledgements

In early 2009, I came in Gembloux and knocked on the door of Prof. Nicolas Gengler. My everlasting interest for animal breeding brought me there. I was looking for a short-time position in an animal science research group, where I could simultaneously work and learn the basic and advanced principles of genetic evaluation. But Nicolas had a much better plan than a short-time position: a PhD in collaboration with CONVIS s.c. and University of Georgia (UGA). This plan allowed me to be part of a wonderful research group, to enjoy the work with them and to learn not only about genetic evaluation but also – mostly, should I say – about numerical calculus, programming and maths. It allowed me as well to travel in Europe and to USA where I met many other great people. For initiating the PhD project and for pushing me ahead in its achievement, I would address to Nicolas, my supervisor, sincere and warm thanks.

This was also made possible thanks to the thesis committee that advised me every year and to the thesis jury that read and reviewed the thesis. Therefore, I would like to personally acknowledge each member of these committee and jury. The thesis committee members were Prof. André Théwis, Prof. Rodolphe Palm and Prof. Yves Beckers, all from Gembloux Agro-Bio Tech (GxABT), Mr. Jean Stoll (formerly CONVIS s.c.), Mr. Romain Reding (CONVIS s.c) and Mrs. Jeanne Bormann (Administration et Services Techniques de l’Agriculture du Luxembourg, ASTA). Mrs Jeanne Bormann is also part of the thesis jury, as well as Prof. Frédéric Francis (President of the thesis jury) and Prof. Catherine Charles, both members of GxABT, Prof. Frédéric Farnir (Faculty of Veterinary Medicine of the University of Liège) and Dr. Ignacy Misztal (University of Georgia). I would like to address special thanks to Mrs. Jeanne Bormann and Prof. Catherine Charles that read the first draft of my thesis. Thanks to their helpful comments, my initial document has been improved significantly.

My gratitude also goes to Dr. Ignacy Misztal. During my “American” time, Ignacy delivered me so many advices. Those were precious and precise, numerous and … numeric, obviously! But they went far beyond the strictly professional area. Thank you, Ignacy, for all these moment we have spent aside of the work, talking about numerous and various subjects while seating at a good table or sharing a good drink.

Talking about prospects and discussing research points of view are an important part of the research work. In addition to the people mentioned here before, I would like to extend my gratitude to the following people for the fruitful discussions we had: Shogo Tsuruta (UGA), Selma Forni (Genus PIc, USA), Andrès Legarra (Institut National de la Recherche Agronomique, France), Ignacio Aguilar (Instituto Nacional de Investigacion Agropecuaria, Uruguay), Gregor Gorjanc (University of Ljubljana, Slovenia and Roslin Institute, Scotland) and Jérémie Vandenplas (GxABT).

I would also acknowledge the institutions that surrounded my project through a special thank to the following people: Mrs. Marie-Claude Marx and Mrs. Susana Pinto, from the Fonds National de la Recherche Luxembourg that funded the four first years of the PhD project; Mr. Romain Reding, Mr. Jeff Stirn and Mr. Armand Braun, from CONVIS s.c.; Mr. Alain Gillon from the Walloon Breeding Association (AWE). In addition, I would like to acknowledge the Ministry of Walloon Region for funding the remaining seven months of PhD.

My final thanks go to the research teams to which I have been part of: all members of the Animal Science Unit of Gembloux Agro-Bio Tech and of the Animal Breeding and Genetics Group of the Animal Dairy Science Department of the University of Georgia. Among those people, an additional thanks goes to: Valérie Arnould, with whom I have been closely working on Luxembourg data; Catherine Bastin, Frédéric Colinet and Jérémie Vandenplas, for revising some of my manuscripts; Frédéric Colinet, Sylvie Vanderick, Hana Bel Mabrouk, Aurélie Lainé and Marie-Laure Vanrobays, for their help in testing algorithms.

… Mais aussi…

Au-delà des personnes et institutions remerciées ci-dessus pour leur investissement direct dans mes travaux de thèse et leur réalisation, ma gratitude et mes remerciements vont à toutes celles et ceux qui m'ont accompagné au cours de mes années de doctorat.

Merci d'abord à tous les collègues de Gembloux ou d'Athens. Pêle-mêle et en tâchant de n'oublier aucun d'entre eux, merci à Valérie pour nos discussions sur la route d'Ettelbrück, à Frédéric pour ses précieuses compétences en Gestion Avancée des

Matières Administratives et Complexités Gembloutoises, à Marie-Laure pour ses traits d'humour discrets mais irrésistibles, à Marie pour avoir été, je pense, la collègue avec laquelle j'ai passé moins de temps au travail qu'en dehors, à Sylvie pour avoir été un pilier du « grand bureau », à Jérémie pour se prêter aussi facilement au photo-montage en Maître ou SuperGeek, à Catherine pour l'interminable liste de perronismes (oui, ce mot existe !) et, en parlant de perron, à Hedi pour nos discussions « à la marge » sur celui de la Zootechnie, à Hana, Laura, Aurélie, Purna, et aussi Amaury, Elisabeth, Hélène, Bernd, Alain, mais encore à El pour nos afterworks au Starbuck, à Joy, Kaori, Daniela L. et Shannon. Que de moments inoubliables n'avons-nous pas vécus ensemble ! Je pense particulièrement aux conférences, celles de Phoenix, Nantes, New Orleans et Denver, mais aussi aux jeux du mess, aux pauses d'un côté de l'océan et aux coffee-breaks de l'autre côté, aux innombrables verres de service, barbecues, pizza parties et autres réjouissances. Que d'endroits n'avons-nous pas visités – voire même colonisés – ensemble ! Plus d'une terrasse de café, ici et ailleurs, se souviennent de notre passage. Je me dois d'adresser à cet effet une dédicace spéciale à mes compagnons de voyage, Marie et Jérémie.

L'université, ce n'est pas qu'un seul groupe de recherche, ni une seule unité, ni même une seule faculté. C'est aussi un endroit riche de rencontres. Je profite de cette occasion pour adresser mes salutations à toutes ces personnes rencontrées durant les heures de travail mais à côté du travail et avec qui j'ai pu partager de nombreuses discussions, qu'il s'agisse de football américain avec Dennis, de football « belge » avec Sylvie, de course à pied avec Christophe ou d'autres sujets plus frivoles avec Yannick.

Merci aussi à tous mes amis pour leur soutien indéfectible et répété tout au long de la thèse, en particulier à Laurent pour ses messages transatlantiques, à Julien pour son art de la citation bien placée, à Alexis et Julien pour la série « On s'en fout », à Jean-François pour les nombreuses discussions scientifiques, à ceux, Françaises et Américains, à l'amitié desquels ce projet m'a porté, à Julien, Alexis, Augustin, Jérémie, Marie, Dana, Grégoire et Jean-Philippe pour les moments de décompression parfois bien nécessaires, à Augustin, Jean-Charles, Thibaut, Clément, Sophie, Géraldine, Hélène, Auriane, Ariane, Frantz et les autres colocataires que j'ai eus durant cette thèse. Merci également à celles et ceux qui m'ont hébergé, parfois au pied levé mais toujours avec une extrême gentillesse : Marta, Dana, Marie-Rose et Luc.

Merci, aussi et surtout, à ma famille, à Anne-Marie et Jacques, mes chers parents, pour leur présence et leurs conseils. Merci à Philippe-François pour nos discussions interminables sur l'amélioration génétique des bovins limousins, à Anne-Michelle pour d'autres discussions davantage axées sur la génétique moléculaire, à Hélène d'être un vrai petit cœur et, enfin, à Emilie et Gilles pour savoir « inverser » des matrices d'un poids considérable sans que cela n'implique la moindre opération en virgule flottante ! Merci aussi à mes grands-parents, tantes, oncles, cousines et cousins.

Por último, agradecer a Claudia por apoyarme durante los últimos meses de mi tesis y por ayudarme en los últimos toques finales. Gracias.

Enfin, ce n'est pas tant une forme de remerciement qu'il me plaît d'adresser ici qu'un rapide clin d'œil à quelques ... bovins ! Ce n'est d'ailleurs pas dans les habitudes académiques de remercier les sujets d'expérience, du moins lorsqu'il s'agit d'animaux. Mais c'est surtout parce que, d'une manière totalement fortuite, ils m'ont été d'une incommensurable, quoiqu'improbable, utilité que je tiens à mentionner ici les 643 individus du pedigree de « mes » animaux !

TABLE OF CONTENTS

GENERAL INTRODUCTION **1**
CONTEXT 2
IMPLICATIONS OF THE DOCTORAL RESEARCH IN CURRENT ANIMAL BREEDING 2
OBJECTIVES 4
THESIS OUTLINE 5
THESIS FRAMEWORK 7
REFERENCES 8

LITERATURE REVIEW **11**
MATRIX INVERSION 12
ADDITIVE AND GENOMIC RELATIONSHIP MATRICES BETWEEN GENOTYPED ANIMALS 13
Additive relationship matrix between genotyped animals *13*
Genomic relationship matrix *14*
TECHNIQUES FOR INVERSION OF RELATIONSHIP MATRICES 16
Introduction *17*
Additive relationship matrix *18*
Gametic relationship matrix *21*
Covariance matrices for marked QTL effects *23*
Dominance relationship matrix *27*
Epistasis Matrices *29*
Discussion and conclusions on inversion technique for relationship matrices *30*
REFERENCES 32

THE CLOSE-FAMILY APPROACH **37**
INTRODUCTION 38
MATERIAL AND METHODS 40
Approximation of Inverse of G and A_{22} *40*
Link with Inversion of A *44*
Applications of Algorithm *45*
Tests of Algorithm *46*
RESULTS AND DISCUSSION 48
Approximation of the inverse of G *48*
Estimation of GEBV with Approximate G^{-1} *48*
Approximation of the inverse of A_{22} *50*
Possible Improvements *53*
CONCLUSIONS 54
APPENDIX 55
Application 1 .... *55*
Application 2 .... *55*
REFERENCES 56

THE SPARSITY PATTERN ALGORITHM **59**
BACKGROUND 61
METHODS 63
Blockwise inversion of A_{22} *63*
Contribution of selected animals to relationships in A_{22}: characterizing the sparsity pattern of T^{1} *64*
An algorithm to set up the sparsity pattern *66*
Use of the sparsity pattern in blockwise inversion of A_{22} *67*
Experimental design for tests on real populations *69*
Two test software programs *71*
RESULTS 71
Characterizing the sparsity pattern: a numerical example *71*

Effect of accounting for sparsity on CPU time for inversion of A_{22} *73*
Effect of the number of extracted generations on accuracy of the inverse of A_{22} *74*
DISCUSSION 75
Computation time required by the algorithm to characterize the sparsity pattern *75*
Memory requirements of the algorithm to characterize the sparsity pattern *77*
Use of the algorithm to characterize the sparsity pattern on greater populations *77*
Computation time required by the algorithm for inversion of A_{22} using the sparsity pattern (Algorithm B) *80*
Memory requirements of the algorithm for inversion of A_{22} using sparsity pattern (Algorithm B) *82*
Number of generations to extract *82*
Practical use in a genomic background *83*
Current limits *84*
CONCLUSIONS 85
APPENDIX: INVERSION OF THE NUMERATOR RELATIONSHIP MATRIX USING THE INVERSE TRIANGULAR FACTOR 85
REFERENCES 88

THE RESTRICTED SPARSITY PATTERN ALGORITHM 91
INTRODUCTION 92
MATERIAL AND METHODS 93
Simulation of realistic test populations *93*
Sparsity in the inverse of A_{22} and SP algorithm *94*
Modified SP algorithm *97*
Implementation of the RSP algorithm *99*
Test protocol for accuracy of approximation *100*
RESULTS AND DISCUSSION 101
Approximation accuracy *101*
Computation time *104*
Memory requirements and sparsity *105*
CONCLUSIONS 105
REFERENCES 106

GENERAL DISCUSSION 109
ALGORITHMS AND STRATEGY FOR DETERMINATION OF CONTRIBUTORS IN THE SPECIFIC CASE OF A_{22} 110
Details on algorithms and strategies *111*
Implementation of the computation of the inverse or of its approximation in the specific case of A_{22} *115*
Comparative study between the different approximation algorithms *116*
ALGORITHMS FOR DETERMINATION OF CONTRIBUTORS IN THE SPECIFIC CASE OF G 122
REFERENCES 123

PERSPECTIVES AND CONCLUSIONS 125
PERSPECTIVES OF USE OF THE PROPOSED ALGORITHMS AND STRATEGIES 126
Use of the SP algorithm *126*
Use of the RSP algorithm *126*
PERSPECTIVES OF FUTURE RESEARCHES 128
On the SP algorithm *128*
On the RSP algorithm *128*
On taking advantage of parallel computations *128*
PERSPECTIVES FOR ANIMAL BREEDING 129
In the frame of an increasing number of genotypes *129*
In the frame of improved genetic evaluation systems *130*
In the frame of multi-line evaluations *130*
In the general frame of genomic selection *130*

CONCLUSIONS 132
REFERENCES 133
LIST OF TABLES 134
LIST OF FIGURES 136
LIST OF ABBREVIATIONS 139

Chapter I

GENERAL INTRODUCTION

Context

How to make the best estimation of the part of a performance that is transmitted to the next generation? This question is the central issue of animal breeding. Addressing this question enables to decide if the recorded trait can support selection. If well, best animals are chosen in order to improve the next generation for that particular trait.

Throughout history of genetic evaluations, main advances in animal breeding can be understood as the product of collaborative exchanges between availability of data (pedigrees, phenotypic records, and genomic information), technological advances in computer sciences and methodological developments in applied mathematics.

First of all, the recording of animal ancestry through the creation of pedigrees, recorded by an official herd-book society, allowed great advances in the availability of data for genetic evaluation. The first herd-books were published for Shorthorn cattle in 1822 and Hereford cattle in 1846 (Whetham, 1979), whereas cattle societies were established a few decades later (e.g. 1875 for Shorthorn, 1876 for Hereford). At that time, cattle breeders, as well as other animal breeders, were mainly concerned by increasing production yields. In the USA and several other countries, the first dairy herd improvement associations were established in the early years of the 20th century in USA. Official recording schemes rapidly increased the number of available data for animal breeders.

Applied genetics focused on the proper use of this growing amount of data. Availability of recorded pedigree opened the door to the analysis of the genetic structure of these records (Pearl, 1917; Fisher, 1918; Wright, 1922). The variance structure of the genetic effects was derived from the genetic structure of the population and the first robust method of genetic evaluation (Selection Index) was released by Hazel in 1943, followed by the Best Linear Unbiased Prediction (Henderson, 1953; Henderson, 1973).

This last method is the starting point of most of all current genetic evaluations. However, two decades passed before the method was implemented as the “animal model”. Why? Because of two main constraints: (1) the unavailability of computers with sufficient resources to implement the model and (2) the inversion of the additive relationship matrix (**A**), which has a cubical cost with classic inversion algorithms. The first constraint was overcome by technological development in computer science. “(…) [In 1954]*, most of the principles of BLUP were already available, but computing facilities were totally* *inadequate to utilize the method*", Henderson (1973) said. The second constraint became obsolete through the discovery of an algorithm that allows a direct computation of the inverse of **A** with a linear complexity (Henderson, 1976).

The "New Frontier" in animal breeding was crossed in the late 1980's: the use of molecular data in genetic evaluations. Molecular data were first used for detection of quantitative trait-loci (QTL) and subsequently integrated in genetic evaluations (e.g. Fernando and Grossman, 1989; Goddard, 1992; van Arendonk et al., 1994). The recent availability of dense genotypes (on thousands of bi-allelic single nucleotide polymorphisms) enabled a second way of integration of molecular information, called "genomic prediction". It was developed by Meuwissen et al. (2001) as a genome-wide strategy that avoids issues linked to detection and use of significant QTL. As explained by VanRaden (2007), the aim was no longer to attempt tracking individual QTL, but to integrate whole genome marker data directly into genetic merit contributions. In its initial implementation (GBLUP), genomic prediction is done in multiple steps (e.g. as detailed in VanRaden, 2008).

However, genomic prediction is quite demanding in need of computer power and methodological developments. The initial multi-steps methodology was recently transformed into a single-step approach (ssGBLUP; Misztal et al., 2009; Christensen and Lund, 2010), which unifies the use of the three main sources of information (pedigrees, genotypes and phenotypes). The core of this method is to replace the additive relationship matrix **A**, as used in a regular animal model, by a genomically-enhanced matrix **H** (Legarra et al., 2009; also presented in Bömcke et al., 2010), whose inverse has a very simple formulation. Nevertheless, computation of the inverse of **H** requires inversion of two relationship matrices between genotyped animals: the genomic relationship matrix (**G**) and a part of the additive relationship matrix ($\mathbf{A}_{22}$).

Their inversion is currently achieved using classic inversion algorithms, with cubic complexity. Even though parallel computing allowed computing time gains (Aguilar et al., 2011), the development of approximation methods becomes more and more relevant as the number of genotyped animals increases: VanRaden et al. (2013) reported more than 160,000 genotyped animals and Hickey (2013) highlights this decreasing cost of genotyping, coupled to the increase of imputation methods, should bring millions of genotyped animals within five years (from the year of his publication).

Objectives

Our central question is the following: is it possible to achieve inversion of **G** and $\mathbf{A}_{22}$ between genotyped animals without a cubical complexity?

Answering this question is covered by two mains objectives, which are to:

1. Propose algorithms and/or strategies to compute the inverse of these relationship matrices or to approximate their inverse;
2. Assess if computing time gains can be achieved with the proposed algorithms and, in case of approximation of the inverse, to assess if the approximation does not impact the use of the matrix in further computations.

The originality of the thesis lies in the way we address the central question: can we take advantage of the prior knowledge of the genetic structure of the population to reduce computations required by inversion of these two matrices? In that sense, this thesis aims to solve numerical problems by the help of genetic knowledge.

Implications of the doctoral research in current animal breeding

The main advantage of genomic prediction is to increase reliability of estimated breeding values (EBV) at a lower age (actually, as soon as DNA can be sampled); for instance, Calus et al. (2008) presented clear advantages of genomic selection, even at low marker densities.

The present dissertation was carried out in the frame of a project for Luxembourg dairy breeders (see below: *Thesis framework*). The Luxembourg dairy cattle population is a small-sized population (less than 40,000 cows currently in production). Predominant breed is Holstein, some being Red-Holsteins, upgraded from Friesian and Red-White cattle (Hammami, 2009) essentially since the 1970s. In Luxembourg, pedigree recording has been effective for decades and more than 90% of the total dairy cattle are under a milk recording system. Genomic prediction, and particularly the single-step procedure, is advantageous for small-sized populations, as the amount of information brought by genotypes is important in comparison to the amount of information brought by other sources (pedigrees, phenotypes). The ssGBLUP is also well adapted for novel traits with limited recording. Luxembourg is one of the few places worldwide where the fine milk composition (fatty acids content) is routinely recorded so far. As for other novel traits, genetic evaluation of milk composition traits may be less reliable due to the lack of evaluation history. Additional information brought by genotypes helps to fill this gap.

The methods outlined in this doctoral dissertation are dedicated to genetic evaluation systems, including those using a large number of genotypes. However, even if less fitted to its population, they can be applied to the specific case of Luxembourg. These methods were designed to speed up the routine genetic evaluations using genomic information.

Thesis outline

This doctoral thesis consists in a compilation of peer-reviewed articles that address the two main objectives in a coherent sequence of 4 chapters (Chapter II to V), a general discussion (Chapter VI) and perspectives and conclusions (Chapter VII).

Proposing algorithms for inversion of **G** and $\mathbf{A}_{22}$ presumes to define the operation of inversion and the features of these two matrices. These are the two first points addressed in Chapter II. In addition, Chapter II includes a literature review of the main computational techniques related to the use of relationship matrices in animal breeding. This review ends with the definition of a general framework for inversion of relationship matrices: relationship matrices can be inverted by using the prior knowledge of the population structure to set up the dependencies between the different levels of any genetic effect. In the case of **G** and $\mathbf{A}_{22}$, where each level of the genetic effect features a genotyped animal, setting up the contributions for any genotyped animal covers two aspects: (1) to determine which animals (later denoted as *contributors*) contribute to this animal, and (2) to compute the value of their contributions.

In Chapter III, a first algorithm is proposed to determine contributors of an animal: the close-family approach. This algorithm can be used in a recursive manner, allowing the proposal of a second algorithm: the recursive close-family approach. In both cases, an approximation of contributions is computed using ordinary least squares. Those two algorithms are applied to the case of **G** and $\mathbf{A}_{22}$. Use of the close-family approach is not time-expensive but not suited for the case of **G**. Use of the recursive close-family approach is prohibitive in terms of computing time, albeit well suited for both matrices. In addition, for the case of $\mathbf{A}_{22}$, results show that, even if the number of contributions to compute is large, a majority of these contributions are close to 0. Focusing on the determination of these contributions would be helpful to address the main question of this thesis.

Therefore, the research focuses on $\mathbf{A}_{22}$ in the upcoming chapters. The core of Chapter IV is the proposal of a heuristic algorithm that exhaustively searches the pedigree to find contributors of each genotyped animal. Computations of contributions are then restricted to these contributors, as other animals do not contribute. Moreover, in Chapter IV, a strategy is proposed to reduce the number of contributors, namely to extract pedigree of genotyped animals only on few generations. Results show that inversion and approximations are very fair; however, computations still have a cubical complexity with the order of the matrix. This complexity could be avoided if the relation between the number of contributors and the matrix order would be broken or, at least, tempered.

Consequently, a restricted search for contributors is proposed in Chapter V. This algorithm – actually, a restricted version of the core algorithm of Chapter IV – allows fair and close to sparse approximations of the inverse of $\mathbf{A}_{22}$. Results show that approximations have a limited impact on the further use of the inverse of $\mathbf{A}_{22}$.

Eventually, Chapter VI is a general discussion that starts by a comparative study of the different algorithms proposed for $\mathbf{A}_{22}$ and those proposed for **G**. For each algorithm, an implementation is proposed and tests are ran on the same computer in order to compare time and memory efficiencies and, if required, quality of approximation. Future perspectives of research and use of the proposed algorithms are discussed and general conclusions are drawn in the last chapter (Chapter VII).

A reminder of the thesis outline takes place in front of every chapter. In addition, Chapters II to V are followed by a summary of the main results outlined in the chapter and essential to follow the strategy of research. Also, other communications related to the topic of the chapter are listed at the end of the chapter, if applicable.

Thesis framework

The thesis research was initiated by October 2009 and supported by an “Aide à la Formation-Recherche” (AFR) grant issued by the Fonds National de la Recherche Luxembourg (FNR), under the project name “NextGenGES”.

The initial objectives of this project were the following: (1) methodological contribution to the development of the next generation genomic prediction methods, and (2) application of these methods to milk composition traits. Different constraints that appeared during the project forced a shift in the thesis objectives. These constraints were: (1) the unfitness between the purpose of methodological developments (drawing solutions for large matrices) and the size of the dairy cattle population from Luxembourg; (2) the unavailability of genotypes in Luxembourg; and (3) the increasing relevance of the methodological issues that appeared during the research.

Under the terms of this grant, the research was part of a public-private partnership between the cattle society from Luxembourg CONVIS s.c. and the host institution of this doctoral thesis, Gembloux Agro-Bio Tech, part of the University of Liège (Gembloux, Belgium). The major part of the doctoral research was done at Gembloux Agro-Bio Tech, under the supervision of Prof. N. Gengler, on pedigree data provided by CONVIS s.c., where a certain time of work was also spent. In addition, a third institution was involved in the project, the Animal and Dairy Science Department of the University of Georgia (Athens, GA, USA), where the training on computational techniques in animal breeding and the first methodological researches were carried out under the co-supervision of Dr. I. Misztal.

This grant initially covered three years, it was then renewed for a fourth year in October 2012 and it eventually ended in September 2013. For the remaining period of doctoral research (October 2013 to May 2014), the work done for this thesis contributed to the project “DairySNP” done by Gembloux Agro-Bio Tech and the Walloon Breeding Association (“Association Wallonne de l’Elevage”, AWE) and supported by the Ministry of Agriculture of the Walloon Region of Belgium.

Alongside to the thesis, a doctoral formation was successfully completed. The doctoral formation included, among others, the following main aspects: attendance to classes in animal breeding and genetics and big data management, active participation to international congresses, teaching assistance of the quantitative genetics class taught in

Gembloux by Prof. N. Gengler and publication of research outputs in peer-reviewed journals.

References

Aguilar I., Misztal I., Legarra A. and Tsuruta S., 2011. Efficient computation of the genomic relationship matrix and other matrices used in single-step evaluation. *J. Anim. Breed. Genet.*, 128, 422–428.

Bömcke E., Soyeurt H., Szydlowski M. and Gengler N., 2010. New method to combine molecular and pedigree relationships. *J. Anim. Sci.*, 89, 972–978.

Calus M.P.L., Meuwissen T.H.E., de Roos A.P.W. and Veerkamp R.F., 2008. Accuracy of Genomic Selection Using Different Methods to Define Haplotypes. *Genetics*, 178, 553–561.

Christensen O.F. and Lund M.S., 2010. Genomic prediction when some animals are not genotyped. *Genet. Sel. Evol.*, 42, 2.

Fernando R.L. and Grossman M., 1989. Marker assisted selection using best linear unbiased prediction. *Genet. Sel. Evol.*, 21, 1–11.

Fisher R.A., 1918. The Correlation between Relatives on the Supposition of Mendelian Inheritance. *Earth Environ. Sci. Trans. R. Soc. Edinb.*, 52, 399–433.

Goddard M.E., 1992. A mixed model for analyses of data on multiple genetic markers. *Theor. Appl. Genet.*, 83, 878–886.

Hammami H., 2009. *Genotype by Environment Interaction for Production Traits of Holsteins Using Two Countries as Model: Luxembourg and Tunisia*. Gembloux, Belgium: University of Liège.

Hazel L.N., 1943. The Genetic Basis for Constructing Selection Indexes. *Genetics*, 28, 476–490.

Henderson C.R., 1953. Estimation of Variance and Covariance Components. *Biometrics*, 9, 226–252.

Henderson C.R., 1973. Sire evaluation and genetic trends. In: *Proc. Anim. Breed. Genet. Symp. Honor Dr Jay Lush*., Am. Soc. Anim. Sci. and Am. Dairy Sci. Assoc., Poultry Sci. Assoc., Champaign, IL, 10-41.

Henderson C.R., 1976. A Simple Method for Computing the Inverse of a Numerator Relationship Matrix Used in Prediction of Breeding Values. *Biometrics*, 32, 69–83.

Hickey J.M., 2013. Sequencing millions of animals for genomic selection 2.0. *J. Anim. Breed. Genet.*, 130, 331–332.

Legarra A., Aguilar I. and Misztal I., 2009. A relationship matrix including full pedigree and genomic information. *J. Dairy Sci.*, 92, 4656–4663.

Meuwissen T.H.E., Hayes B.J. and Goddard M.E., 2001. Prediction of total genetic value using genome-wide dense marker maps. *Genetics*, 157, 1819–1829.

Misztal I., Legarra A. and Aguilar I., 2009. Computing procedures for genetic evaluation including phenotypic, full pedigree, and genomic information. *J. Dairy Sci.*, 92, 4648–4655.

Pearl R., 1917. Studies on Inbreeding. VII.-Some Further Considerations Regarding the Measurement and Numerical Expression of Degrees of Kinship. *Am. Nat.*, 51, 545–559.

van Arendonk J.A.M., Tier B. and Kinghorn B.P., 1994. Use of multiple genetic markers in prediction of breeding values. *Genetics*, 137, 319–329.

VanRaden P.M., 2007. Genomic measures of relationship and inbreeding. *Proc Interbull Annu. Meet.*, 33–36.

VanRaden P.M., 2008. Efficient Methods to Compute Genomic Predictions. *J. Dairy Sci.*, 91, 4414–4423.

VanRaden P.M., Null D.J., Sargolzaei M., Wiggans G.R., Tooker M.E., Cole J.B., Sonstegard T.S., Connor E.E., Winters M., van Kaam J.B.C.H.M., Valentini A., Van Doormaal B.J., Faust M.A. and Doak G.A., 2013. Genomic imputation and evaluation using high-density Holstein genotypes. *J. Dairy Sci.*, 96, 668–678.

Whetham E.H., 1979. The Trade in Pedigre Livestock 1850-1910. *Agric. Hist. Rev.*, 27, 47–50.

Wright S., 1922. Coefficients of Inbreeding and Relationship. *Am. Nat.*, 56, 330–338.

Chapter II

LITERATURE REVIEW

In genetic evaluations, relationship matrices establish the covariance structure of the genetic effect; each type of genetic effect (additive, dominance…) has its own type of relationship. Mixed model equations integrate the inverse of the relationship matrix of a certain type of genetic effect in order to obtain solutions for that type of effect. Therefore, computing the inverse of a relationship matrix without having to set up the matrix itself is of great interest in order to ease evaluations. In the third section of this chapter, the main algorithms to directly compute inverses of relationship matrices are reviewed. Beforehand, in the first and second sections, the operation of inversion and the features of the two matrices of interest ($\mathbf{A}_{22}$ and **G**) are introduced.

Matrix inversion

A square matrix **A** of order n is invertible if a matrix **B** of same order exists such as their product returns the identity matrix of same order: $\mathbf{AB} = \mathbf{I} = \mathbf{BA}$. **B** is called the inverse matrix of **A** and is denoted by $\mathbf{A}^{-1}$.

Matrix **A** is invertible if the determinant of **A** is different from zero. The determinant can be computed from the values of elements in **A**. Moreover, the determinant of a product of matrices is the product of the determinant of each matrix. Different methods allow computing the inverse of a matrix. These methods can be categorized into direct and iterative methods.

Direct methods require high accuracy in calculation to obtain proper solutions whereas iterative methods compensate round-off errors by a process of successive refinement (Rajagopalan, 1996). The most commonly implemented method is a direct one: the Gauss-Jordan method. This method applies a sequence of transformations of rows and columns on the original matrix and an identity matrix in order to convert the original matrix into an identity matrix. The inverse is then stored in the second matrix. The Cholesky factorization of symmetric matrix ($\mathbf{A} = \mathbf{LL}'$; Cholesky, 1910) can be used to compute its inverse by computing the inverse of **L** and multiplying it by its transpose. Another direct method is the Sherman-Morrison algorithm (Sherman and Morrison, 1950). This algorithm performs inversion in a line-wise manner: a zeroed matrix is updated, at each row, by the product of a vector of same length as the order of the matrix by its transpose. This vector is computed from the result of the previous inverse and the corresponding column in the original matrix. Sherman-Morrison algorithm is equivalent to the blockwise inversion algorithm (Banachiewicz, 1937) in which the Schur complement would be scalar.

Iterative methods work by successively improving approximations until a numerical convergence is reach. Speed of convergence is, however, dependent on the initial approximation. Among others, pre-conjugate gradient and bi-conjugate gradient stabilized methods are worth citing.

Additive and genomic relationship matrices between genotyped animals

If some animals are genotyped in a population, one can split this population between genotyped and non-genotyped animals. Such a basic splitting of the original population into two groups is feasible for any feature, e.g. splitting between recorded and non-recorded animals, between sexes or between breeds.

From a partition between genotyped and non-genotyped animals, two additive relationship matrices may be computed. The first one, $\mathbf{A}_{22}$, is the part of the additive relationship matrix that gathers relationships between genotyped animals. The second one, **G**, is a matrix of similarities between genotyped animals computed using genomic information. In a certain sense, **G** can be also be interpreted as part of a larger matrix: this larger matrix would be of the same size as **A**, only possible, however, if genomic information was available for all animals in population.

Additive relationship matrix between genotyped animals

Additive relationships coefficients are relationship measurements based on the knowledge of potential co-ancestries between two animals. Setting up such relationship coefficients requires having genealogical information, often streamlined in a table of triplets animal-sire-dam. Such table, as well as any triplet it contains, is called “pedigree”.

The additive relationship coefficients are due to Wright (1922) and have a range in $\left[0,2\right[$: two unrelated animals have a relationship coefficient of 0 and two clones would have a relationship coefficient of 2. Since rules to set up A treat two clones as two full-sibs (Emik and Terrill, 1949), a relationship coefficient of 2 is not reachable.

The relationship coefficient of an animal with itself is defined as 1 plus the inbreeding coefficient of that animal. The inbreeding coefficient (F) is the half of the relationship coefficient between the two parents.

Matrix **A** is symmetric and positive-definite. Non-singularity of **A** can be proved using the rules to compute the diagonal elements (Quaas, 1976) of its Cholesky factorization (see Henderson, 1976) and using the definition of additive relationship coefficients. For an animal with both parents known, the diagonal element of the Cholesky factorization is $\sqrt{\left(0.5-0.25(F_S+F_D)\right)}$, where F_S and F_D are respectively inbreeding coefficients of the sire and the dam of that animal. Inbreeding coefficients are the half of relationship coefficients and therefore have a range from 0 to 1 excluded. Consequently, the diagonal element of the Cholesky factorization is always positive, allowing existence of that factorization and non-singularity of the matrix. Inversion of **A** is detailed further in this chapter.

Matrix $\mathbf{A}_{22}$ is any part of **A** gathering relationship coefficients between animals chosen among all animals in population. In our case, this group is the group of genotyped animals. Matrix $\mathbf{A}_{22}$ has therefore the same structure and properties as **A**. The quickest and simplest way to compute $\mathbf{A}_{22}$ is the method of Colleau (2002). This method is derived from the inverted Cholesky factorization of **A** and it computes the additive relationship coefficients of an animal with the rest of the population by a double reading of an age-ordered pedigree. Applying the method for all genotyped animals and retaining only relationships with genotyped animals achieves computation of $\mathbf{A}_{22}$.

Genomic relationship matrix

Genomic relationship coefficients are relationship measurements based on similarities between individuals revealed by molecular markers.

Nejati-Javaremi et al. (1997) were the first to outline a method to derive the allelic relationship *TA* between two individuals *x* and *y* at a given locus *l*: they averaged the identity between the two alleles of an individual at this locus and both alleles of another individual at the same locus. The measure is repeated for all *L* available marker loci and averaged, returning the total allelic identity between the two individuals (equation II.1).

$$TA_{xy} = \frac{\sum_{l=1}^{L} TA_l}{L} \quad \text{(II.1)}$$

This method was, among others, implemented by Bömcke and Gengler (2009), using 16 microsatellites markers with at least 4 alleles, in order to derive a combined pedigree-genomic additive relationship.

The development of genome sequencing methods made hundreds of thousands of single nucleotide polymorphisms (SNP) available, opening the path to genomic selection (Meuwissen et al., 2001; Goddard and Hayes, 2007). Among the large number of SNP widely spread over the genome, bi-allelic markers are chosen to create assays of several thousands of SNPs (3,000, 10,000 or 50,000 on the most used beadchips). This availability considerably increased the accuracy of the relationship measurement, as well as its simplicity because SNPs are bi-allelic. Using bi-allelic genotypes (-1 and 1 for both homozygotes; 0 for heterozygote) in equation (II.1), one can define a matrix of total allelic relationship as in equation (II.2), where $\mathbf{Z}_0$ is a matrix containing, in row, bi-allelic genotypes and, in column, *m* SNPs.

$$\mathbf{TA} = \frac{\mathbf{Z}_0\mathbf{Z}_0' + m}{m} \qquad \text{(II.2)}$$

However, at a given locus, if an allele is less frequent than the other one, two individuals carrying this allele are more likely to be related than two animals carrying the frequent allele. In other words, common alleles are less informative than rare alleles to compute relationships. Thus, taking allelic frequencies into account matters.

VanRaden (2007) proposed a genomic relationship matrix (equation II.3) structurally similar to that derived (equation II.2) from Nejati-Javaremi et al. (1997), but that accounts for allelic frequency in both the genomic incidence matrix (**Z**) and the scaling factor (*d*).

$$\mathbf{G} = \frac{\mathbf{ZZ}'}{d} \qquad \text{(II.3)}$$

Matrix **Z** is obtained by subtraction of **P** from $\mathbf{Z}_0$. Each row of **P** contains the allelic frequencies of all alleles expressed as a difference from 0.5 and multiplied by 2, so that the *i*-th column of **P** is $2\left(f_i - 0.5\right)$, where f_i is the minor allelic frequency of the *i*-th SNP. As explain in VanRaden (2008), subtracting **P** from $\mathbf{Z}_0$ gives more credit to rare alleles than to common alleles. The scaling factor *d* is equal to $2\sum f_i(1-f_i)$ and, according to VanRaden (2008), makes **G** analogous to $\mathbf{A}_{22}$. This analogy has to be tempered for two reasons. Firstly, tuning is required to make both diagonal and off-diagonals values compatible with $\mathbf{A}_{22}$ (see Forni, 2011). Secondly, matrix $\mathbf{A}_{22}$ measures identity-by-descent (IBD) between animals whereas matrix **G** measures identity-by-state (IBS) between animals. IBS can be imputed either to co-ancestry or to randomly occurring mutations. Recent proposals create **G-IBD** matrices by tracing the gene flows through pedigrees (Villanueva et al., 2005) or using haplotypes (Hayes et al., 2009).

Matrix **G** (equation II.3) is positive semi-definite (VanRaden, 2007) but can be singular if two clones are genotyped, or if identical genotypes are found for two different animals because the number of SNPs is small. In such case, the pairs of rows and columns corresponding to these animals are the same. Also, matrix **G** can be singular if the total number of alleles is less than the number of genotyped individuals. Using a combined **G** made of a high proportion of **G** and a low proportion of $\mathbf{A}_{22}$ could circumvent singularity.

Even though, in this study, **G** is used as defined in equation (II.3) and made compatible with $\mathbf{A}_{22}$ using methods by Forni (2011), other genomic relationship matrices are worth citing (Leutenegger et al., 2003; Amin et al., 2007; Gianola and van Kaam, 2008).

Techniques for inversion of relationship matrices

[FROM: P. Faux and N. Gengler. 2014. A review of inversion techniques related to the used of relationship matrices in animal breeding. *Biotechnologie, Agronomie, Sociétés et Environnement*, (in press)]

Abstract

In animal breeding, prediction of genetic effects is usually obtained through the use of mixed models. For any of these genetic effects, mixed models require the inversion of the covariance matrix associated to that effect, which is equal to the associated relationship matrix times the associated component of the genetic variance. Given the size of many genetic evaluation systems, computing the inverses of these relationship matrices is not trivial. In this review, we aim to cover computational techniques that ease inversion of relationship matrices used in animal breeding for prediction of the following different types of genetic effects: additive effect, gametic effect, effect due to presence of marked quantitative trait loci, dominance effect and different epistasis effects. Construction rules and inversion algorithms are detailed for each relationship matrix. In the final discussion, we draw up a common theoretical frame to most of the reviewed techniques. Two computational constraints come out of this theoretical frame: setting up the matrix of

dependencies between levels of the effect and setting up some parts (diagonal or block-diagonal elements) of the relationship matrix to be inverted.

Keywords: animal breeding, quantitative genetics, breeding value.

Introduction

A simple model (equation II.4; see Kempthorne, 1955) describes a given phenotype (P) as the sum of the genotype (G) and the environment (E) of a particular animal.

$$P = G + E \qquad \text{(II.4)}$$

Based on equation (II.4), variations among phenotypic observations are therefore explained by genetic and environmental variations and by a potential interaction between genotype and environment. Genetic improvement of animals requires accurate estimation of the genetic variance component in order to predict the genetic values of animals. The structure of this variance component is based on knowledge of the biological processes involved in Mendelian inheritance.

In nearly all domestic species, animals have a diploid genome (with the exception of honey bees, where males are haploid). Then, during the production of gametes, a haploid copy of the diploid genome of the original animal (sire or dam) is made. However, haploid copies are produced from potentially different parts of the homologous chromosomes, following the process of recombination due to crossing-over. Thus, for any locus, a gamete carries a single copy of one of the two alleles carried by the parental genome. Both gametes eventually merge to create a new animal.

By the process described before, every new animal has a specific and unique genetic makeup. Genetic covariances among different animals arise because they have inherited similar alleles and allele combinations. Based on these covariances, associations among these animals can be defined as ratios between covariances and variances associated to a given genetic effect. Whether the interactions between alleles of the same locus (intra-locus interaction) and between loci (inter-loci interaction) are null or not, several types of genetic effects can be distinguished. In our study, we will cover and detail the following genetic effects: additive, gametic, effect due to marked QTL, dominance and the different types of epistasis effects.

When fitting a linear model with generalized least squares, use of the inverted covariance structure among observations allows obtaining Best Linear Unbiased Estimators. Prediction of genetic effects is usually obtained through the use of mixed models (Henderson, 1953; Henderson, 1973). These models are equivalent to models fitted using generalized least squares and, for every random effect, the inverse of the associated covariance structure is also needed.

Due to huge size of regular genetic evaluations, there is a substantial interest in computational techniques that make efficient use of covariance matrices in terms of computing time and memory requirements. Thus, our main objective is to review and explain in detail algorithms for inversion of relationships matrices useful in animal breeding. Completion of this objective involved the definition of the relationships between levels of the concerned genetic effect and the computation of the related matrices for each type of genetic effect listed above (additive, gametic, marked QTL effects, dominance and epistasis). Finally, we outline a general framework of inversion of relationship matrices in the final discussion.

It must be noted that the case of genomic relationship matrices has been willingly discarded in this study because no algorithm that directly sets up their inverses has been developed so far. The genomic relationships are made available by the use of dense marker chips (over than tens of thousands of markers) and give an accurate estimation of the observed relationship between two animals. For their computation, please refer to the work of VanRaden (2008), for additive genomic relationship matrix, and Su et al. (2012) for non-additive genomic relationship matrix.

Additive relationship matrix

Definition of the additive relationship

If interactions between alleles are considered null, the genetic (co)variance is said to be “additive”. Based on previous work by Pearl (Pearl, 1917a; Pearl, 1917b), Wright (1922) defined an additive relationship coefficient as the additive correlation between two animals *i* and *j* (equation II.5).

$$r_{ij} = \frac{Cov[i,j]}{\sqrt{Var[i] \cdot Var[j]}} = \frac{a_{ij}}{\sqrt{a_{ii} \cdot a_{jj}}} \qquad \text{(II.5)}$$

The r_{ij} coefficient is a correlation coefficient; it ranges from 0 to 1. The non-scaled coefficient of Wright, noted a_{ij}, is the additive genetic relationship coefficient and, from (II.5), is defined as equal to $r_{ij}\sqrt{a_{ii} \cdot a_{jj}}$. This coefficient is also often referred as the "numerator relationship" coefficient (due to its position in equation II.5). We will denote it as the "additive relationship coefficient" and the kind of relationship that it refers to as an "additive relationship" in our study. The matrix containing all these additive relationship coefficients will be denoted by **A** and called "additive relationship matrix".

Computation of the additive relationship matrix

Complete computation of the additive relationship matrix

The path coefficient method (Wright, 1922) enables the computation of the additive relationship between two animals. The process requires identification of all nearest ancestors shared between those two animals and counting of the number of generation steps between them. The path coefficient method can be automated and extended to computation of relationship coefficients in the whole population. The tabular method (Emik and Terrill, 1949; Henderson, 1976) performs the computation of additive relationship coefficients in a recursive manner. For a given animal, the relationship coefficients of this animal with all older animals are computed in a row by adding one half of the relationship coefficients in the rows of its parents. A prior step is required: organization of pedigree records in a sorted by generation list of triplets animal-sire-dam (Emik and Terrill, 1949; Mugnier et al., 1966). On a population of n animals, a square matrix of order n is created.

This algorithm has a complexity that is proportional to n^2, because, at each of the n loops it achieves, a linear combination of a vector of maximum length n is performed. Storage requirements follow the same trend and may quickly become prohibitive.

Partial computation of the additive relationship matrix

For this reason, and also because only a section of the additive relationship matrix may be of interest in large populations, algorithms that permit a partial computation of the additive relationship matrix have been developed.

Algorithms corresponding to two specific parts of the **A** matrix should be mentioned. The first one is an algorithm that computes the relationship coefficients of a particular animal with the rest of the population (e.g. Colleau, 2002). The second one is an algorithm that computes the diagonal elements of **A**, which reveals inbreeding coefficients (e.g. algorithms of Quaas, 1976; Meuwissen and Luo, 1992; Sargolzaei et al., 2005). The interest of these coefficients will be highlighted in the next sections.

Computation of the inverse of the additive relationship matrix

Matrix **A** is non-singular except in the presence of genetically identical animals (GIA; full-twins or clones). In such situations, contributions of Kennedy and Schaeffer (1989) and Oikawa and Yasuda (2009) are relevant.

In situations without GIAs, Henderson (1976) has proposed rules that allow computing the inverse of **A** without having to compute **A** explicitly. These rules are based on the simplicity of structure of matrices involved in the factorization of **A**: $\mathbf{A} = \mathbf{TDT}'$. According to Henderson (1976), matrix **T** can be computed recursively (equation II.6): the vector corresponding to the *i*-th row of **T**, from column 1 to (*i*-1), is equal to one half of corresponding parental vectors (say *s* and *d*). Diagonal value is 1 and upper triangular part is 0.

$$\mathbf{T}_{(i)} = \begin{bmatrix} \mathbf{T}_{(i-1)} & \mathbf{0} & 0 \\ \mathbf{p}'_{(i)}\mathbf{T}_{(i-1)} & 1 & \vdots \\ 0 & \cdots & 0 \end{bmatrix}, \text{where } \mathbf{p}'_{(i)} = \begin{bmatrix} 0 & \cdots & \overset{s}{0.5} & \cdots & \overset{d}{0.5} & 0 \end{bmatrix} \qquad \text{(II.6)}$$

Inverting the factorization of **A** and using it to compute the inverse of **A** (as $(\mathbf{T}^{-1})'\mathbf{D}^{-1}\mathbf{T}^{-1}$) does not require **T**, but the inverse of **T**. This latter has a very simple structure that comes by inversion of a triangular matrix (equation II.7).

$$\mathbf{T}^{-1}_{(i)} = \begin{bmatrix} \mathbf{T}^{-1}_{(i-1)} & \mathbf{0} & 0 \\ -\mathbf{p}'_{(i)} & 1 & \vdots \\ 0 & \cdots & 0 \end{bmatrix} \qquad \text{(II.7)}$$

The matrix **D** is diagonal: element $\mathbf{D}_{ii}$ is equal to $1 - .25 \cdot \sum_{\forall p \in \Pi_i} \mathbf{A}_{pp}$, where Π_i denotes the set of known parents (either 0, 1 or 2 parents known) of animal *i*. A correct computation of **D** requires to know the diagonal elements of **A**. Algorithms for computation of inbreeding coefficients mentioned in section "*Computation of the additive relationship matrix*" here above are of great interest. Among those, the algorithm by

Quaas (1976) is noteworthy as it is the first one to compute these elements for the particular purpose of the computation of the inverse of **A**.

Once matrix **D** has been computed, Henderson (1976) proposed a simple algorithm to set up the inverse (Algorithm II.1). The algorithm summarizes the product $(\mathbf{T}^{-1})'\mathbf{D}^{-1}\mathbf{T}^{-1}$ to *n* updates of a *n*-by-*n* matrix that was initially set to zero. Each update is a square block matrix of order 1 plus the number of known parents. This principle was demonstrated in Tier and Sölkner (1993) and van Arendonk et al. (1994).

Algorithm II.1. Direct computation of the inverse of the additive relationship matrix (**A**).

```
initialize $\mathbf{B} = \mathbf{D}^{-1}$ and $\mathbf{A}^{-1} = \mathbf{B}$, two matrices of order $n$
for $i = 1$ to $n$, do
    if any parent, say $p$, of the $i$-th animal is known,
        then add $-.5\mathbf{B}_{ii}$ to elements $\mathbf{A}^{-1}_{pi}$ and $\mathbf{A}^{-1}_{ip}$ and $.25\mathbf{B}_{ii}$ to element $\mathbf{A}^{-1}_{pp}$
    if both parents, say $p$ and $q$, of the $i$-th animal are known,
        then add $.25\mathbf{B}_{ii}$ to elements $\mathbf{A}^{-1}_{pq}$ and $\mathbf{A}^{-1}_{qp}$
```

The advantages of this algorithm are its low complexity (O(*n*)) and the low amount of memory required to store the very sparse output ($\mathbf{A}^{-1}$).

Gametic relationship matrix

Definition and uses of gametic relationships

In some situations, it may be interesting to express the additive genetic value of an individual in terms of the separate gametic contributions of each of their two parents (Schaeffer et al., 1989; Kennedy et al., 1988). Prediction of additive gametic values instead of additive genetic values allows reducing the size of the system to solve: the number of genetic effects is equal to the number of parents, necessarily lower than the total number of animals in the population. The covariance matrix used for random genetic (gametic) effects is called the "gametic relationship matrix" and denoted hereafter as $\mathbf{G}_a$. Quaas and Pollak (1980) have developed such a model, known as reduced animal model. This model also shows how each ancestor affects the genetic value of the individual. Gibson et al. (1988) have proposed a gametic model in which only one parental gamete expresses the genetic effect (autosomally inherited) of an individual. Others uses are:

analysis of haploid-diploids species such as the honey bee (Smith and Allaire, 1985) and analysis of gametic imprinting effects (Gibson et al., 1988; Schaeffer et al., 1989). Eventually, the usefulness of the gametic relationship matrix in computation of the dominance relationship matrix has been shown by Schaeffer et al. (1989). The derivation of **A** from the gametic relationship matrix has been described by Smith and Allaire (1985) and showed by Jamrozik and Schaeffer (1991). Matrix **A** is obtained by $\frac{1}{2}\mathbf{KGK}'$, where $\mathbf{K} = \mathbf{I} \otimes \begin{bmatrix} 1 & 1 \end{bmatrix}$ (Tier and Sölkner, 1993; van Arendonk et al., 1994).

Computation of the gametic relationship matrix

Smith (1984) proposed an algorithm to compute $\mathbf{G}_a$ that is inspired by the tabular method. For diploids species, the size of the matrix will be $N = 2n$, where *n* is the number of animals in population. Each animal has thus two rows/columns that correspond to both parental gametes. Construction rules are simply deduced from the tabular method: if the parent *p* is known, then the row elements below diagonal are equal to the half of the sum of corresponding elements in both lines of parent *p*; else if the parent *p* is unknown, these elements are null. The corresponding column is obtained by transposition.

Inversion of the gametic relationship matrix

Matrix $\mathbf{G}_a$ is non-singular within the same restriction as for matrix **A** (no clones).

The following algorithm (Algorithm II.2) was developed by Schaeffer et al. (1989) based on direct computation of the inverse of **A**. Animals are supposed to be ordered chronologically. For each animal, the first and second gametes are respectively due to the sire and dam. Computation of the diagonal elements is similar to that of Quaas (1976).

Algorithm II.2. Direct computation of the inverse of the gametic relationship matrix ($\mathbf{G}_a$) due to Schaeffer et al. (1989).

```
initialize a matrix $\mathbf{G}_a^{-1}$ of order $N$ and three vectors $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{d}$ of length $N$
for $k = 1$ to $N$, do
    set $\mathbf{d}(k) = \mathbf{v}(k) = \sqrt{1-\mathbf{u}(k)}$
    for $i = k + 1$ to $N$, do
```

```
        if the $k$-th gamete precedes any parental gamete, say $p$, of the $i$-th gamete,
            then add $.5\mathbf{v}(p)$ to $\mathbf{v}(i)$
        else set $\mathbf{v}(i)$ equal to 0
            add the square of $\mathbf{v}(i)$ to $\mathbf{u}(i)$
    set $c$ equal to the square of the inverse of $\mathbf{d}(k)$ and $\mathbf{G}_a^{-1}(k,k)$ equal to $c$
    if parental gametes, say $p$ and $m$, of the $k$-th gamete are known, then
        add $-.5c$ to $\mathbf{G}_a^{-1}(p,k)$, $\mathbf{G}_a^{-1}(m,k)$, $\mathbf{G}_a^{-1}(k,p)$ and $\mathbf{G}_a^{-1}(k,m)$
        add $.25c$ to $\mathbf{G}_a^{-1}(p,p)$, $\mathbf{G}_a^{-1}(p,m)$, $\mathbf{G}_a^{-1}(m,p)$ and $\mathbf{G}_a^{-1}(m,m)$
```

Covariance matrices for marked QTL effects

Definition of marked QTL covariance

Development of genetic engineering techniques leads to identify loci involved in determinism of quantitative traits (QTL) and to assist selection by use of markers linked to these QTL (Marked QTL, MQTL; Soller and Beckmann, 1983; Smith and Simpson, 1986). The following model (Fernando and Grossman, 1989) integrates effects of a causative QTL into BLUP.

$$y_i = \mathbf{x}_i'\beta + v_i^p + v_i^m + u_i + e_i \qquad \text{(II.8)}$$

In equation (II.8), a phenotypic value y_i is decomposed in environmental contributions $\mathbf{x}_i'\beta$, random additive genetic contributions: a contribution of the paternally inherited allele of a marked QTL (v_i^p), a contribution of the maternally inherited allele of the same marked QTL (v_i^m) and a residual additive contribution due to QTLs unlinked to the marker (u_i), and a random error contribution (e_i). Solving this mixed model requires the covariance matrix of the v_i values (called “MQTL matrix” and denoted as **G** hereafter), which is computed using both pedigree relationships and marker information.

Computation of the MQTL matrix

Fernando and Grossman (1989) have developed the “MQTL relationship” in a similar manner as the additive relationship. While this latter is based on the probability that alleles at a same locus for each animal are IBD, MQTL relationship is based on the conditional probability of the same event given information on a marker closely linked to the MQTL. This conditional probability is affected by the recombination rate *r* between the marker locus and the marked QTL (outlined and developed similarly in Chevalet et al., 1984): given that an animal inherited the paternal marker allele of its sire, the probability that he also inherited the paternal QTL allele of its sire is (1-*r*) whereas the probability that he inherited the maternal QTL allele of its sire is *r*. The MQTL relationship between two animals *i* and *j*, for both paternal and maternal alleles ($g_{i,j}^{p}$ and $g_{i,j}^{m}$), can thereby be computed recursively from the MQTL relationships between *s*, sire of *i*, and *j* ($g_{s,j}^{p}$ and $g_{s,j}^{m}$) and *d*, dam of *i*, and *j* ($g_{d,j}^{p}$ and $g_{d,j}^{m}$), given marker inheritance:

- if *i* inherits from its sire its paternal marker allele: $g_{i,j}^{p} = (1-r)\cdot g_{s,j}^{p} + r\cdot g_{s,j}^{m}$;
- if *i* inherits from its sire its maternal marker allele: $g_{i,j}^{p} = (1-r)\cdot g_{s,j}^{m} + r\cdot g_{s,j}^{p}$;
- if *i* inherits from its dam its paternal marker allele: $g_{i,j}^{m} = (1-r)\cdot g_{d,j}^{p} + r\cdot g_{d,j}^{m}$;
- if *i* inherits from its dam its maternal marker allele: $g_{i,j}^{m} = (1-r)\cdot g_{d,j}^{m} + r\cdot g_{d,j}^{p}$.

If no information on marker inheritance is available, then both paternal and maternal alleles have equal probability of being inherited and *r* is equal to 0.5. In such a case, the MQTL relationship is the corresponding gametic relationship. Matrix **G** has thus the same size as matrix $\mathbf{G}_a$ and, for computation purposes, is ordered in the same manner (parents before offspring; paternal allele before maternal allele). The computation goes through use of the recursive rules here above in a tabular method. van Arendonk et al. (1994) showed the recursion rule in matrix notation:

$$\mathbf{G}_{(i)} = \begin{bmatrix} \mathbf{G}_{(i-1)} & \mathbf{G}_{(i-1)}\mathbf{q}_i \\ \mathbf{q}_i'\mathbf{G}_{(i-1)} & 1 \end{bmatrix}, \quad \text{(II.6)}$$

where $\mathbf{G}_{(i-1)}$ is the MQTL matrix for gametes 1 to i-1 and $\mathbf{q}_i$ is a vector that has two non-zeros entries: $(1-r)$ to the position of the parental gamete whose allele was inherited and *r* to the position of the other parental gamete. An algorithm by Wang et al. (1995) also follows an identical tabular method but processes animal by animal (thus, 2 lines/rows at a time) instead of gamete by gamete. The tabular method for constructing **G** is therefore:

$$\mathbf{G}_{(i)} = \begin{bmatrix} \mathbf{G}_{(i-1)} & \mathbf{G}_{(i-1)}\mathbf{Q}_i \\ \mathbf{Q}_i'\mathbf{G}_{(i-1)} & \mathbf{C}_i \end{bmatrix},$$

where $\mathbf{C}_i$ is a 2-by-2 matrix with 1 on the diagonal and the inbreeding coefficient of animal *i* elsewhere and $\mathbf{Q}_i$ is a 2-by-(*i*-1) matrix with maximum 8 non-zeros elements, in all 4 columns corresponding to the 2 parental gametes. These elements are filled with the probability of descent for each offspring QTL allele from any parental QTL allele. It is worth noting that this algorithm accommodates situations where paternal or maternal origin of alleles cannot be determined.

A very similar algorithm was developed by Goddard (1992) for the covariance matrix between effects of potential QTL surrounded by two marker loci. In this algorithm, the relative position *p* of the QTL to the marker loci is used instead of the recombination rate of Fernando and Grossman (1989). Tracing inheritance of chromosome segments instead of marker loci enhances accuracy of the model. For genetic evaluation systems including many ancestors without marker information, Hoeschele (1993) showed that QTL effects were needed only for genotyped animals and common ancestors of these animals. Elimination of these equations led to a substantial reduction of the order of the covariance matrix. Such an algorithm that accounts for non-genotyped parents is also presented in Wang et al. (1995).

Direct computation of the inverse of the MQTL matrix

The algorithm of Fernando and Grossman (1989) follows the same approach as Henderson (1976) and Quaas (1976). Using a definition similar to that of their tabular method, they relate both effects of paternal and maternal MQTL (v_i^p and v_i^m) to their parental MQTL (v_s^p, v_s^m, v_d^p and v_d^m) effects in a simple linear model (equations II.10). In this model, coefficients ρ allocate r or $(1-r)$ accordingly with the inheritance turned up by marker information and ε_i^p and ε_i^m residual effects, whose covariance matrix $\mathbf{G}_\varepsilon$ is shown to be diagonal.

$$\begin{cases} v_i^p = \rho_{i,s}^p \cdot v_s^p + \rho_{i,s}^m \cdot v_s^p + \varepsilon_i^p \\ v_i^m = \rho_{i,d}^p \cdot v_d^p + \rho_{i,d}^m \cdot v_d^p + \varepsilon_i^m \end{cases} \quad \text{(II.10)}$$

Assuming that inbreeding coefficients are available, the algorithm proceeds through the pedigree and fills in the inverse of the MQTL matrix (initialized to a null matrix of order *N*) in three steps:

(1) compute the diagonal element d of $\mathbf{G}_\varepsilon$ as $2\rho_{i,s}^{p}\rho_{i,s}^{m}(1-F_s)$ for a paternal gamete or as $2\rho_{i,d}^{p}\rho_{i,d}^{m}(1-F_d)$ for a maternal gamete;

(2) set up a vector $\mathbf{q}$ equal to $\begin{bmatrix} -\rho_{i,s}^{p} & -\rho_{i,s}^{m} & 1 \end{bmatrix}'$ for a paternal gamete or equal to $\begin{bmatrix} -\rho_{i,d}^{p} & -\rho_{i,d}^{m} & 1 \end{bmatrix}'$ for a maternal gamete;

(3) add the product $d\mathbf{q}'\mathbf{q}$ to the inverse matrix to positions corresponding to each of its parental gamete and the current gamete itself.

The algorithm by van Arendonk et al. (1994) is equivalent to the previous one and is outlined under the form of the successive blockwise inversion of Tier and Sölkner (1993). It requires thus the computation of all MQTL relationships of the population. Equivalently, the algorithm of Wang et al. (1995) for direct computation of the inverse of the MQTL relationship matrix processes the two gametes of an animal at a time, as shown in equation (II.10) where $\mathbf{D}_i = \mathbf{C}_i - \mathbf{Q}_i'\mathbf{G}_{(i-1)}\mathbf{Q}_i$ is the Schur complement of $\mathbf{G}_{(i-1)}$.

$$\mathbf{G}_{(i)}^{-1} = \begin{bmatrix} \mathbf{G}_{(i-1)}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} -\mathbf{Q}_i \\ \mathbf{I}_2 \end{bmatrix} \mathbf{D}_i^{-1} \begin{bmatrix} -\mathbf{Q}_i' & \mathbf{I}_2 \end{bmatrix} \qquad \text{(II.10)}$$

Efforts in reducing computational costs of this algorithm have been outlined (Abdel-Azim and Freeman, 2001; Tuchscherer et al., 2004; Matsuda and Iwaisaki, 2002). The computing cost reduction performed by Sargolzaei et al. (2006), applying the indirect method of Colleau (2002) to the MQTL matrix, is also of great interest.

Computation and inversion of a covariance matrix for an animal model accounting for MQTL relationships

The closeness between gametic and MQTL relationship matrices has already been mentioned. Also, it has been mentioned that the additive genetic relationship matrix **A** could be retrieved from the gametic relationship matrix using an incidence matrix **K** (see section "*Definition and use of the gametic relationships*"). Similarly, it is worth noting that a modified **A** (noted hereafter $\mathbf{A}_M$) could be obtained from the MQTL matrix (van

Arendonk et al., 1994) as $\frac{1}{2}\mathbf{KGK}'$. Computation of the inverse is made successively in a similar manner as for **G** (see previous section). However, vectors $\mathbf{q}_i$ have non-trivial values. Their computation is therefore made using a construction of $\mathbf{A}_M$ similar to equation (II.9) for **G**. An analogous equation would express the *i*-th above diagonal column vector of $\mathbf{A}_M$ ($\mathbf{A}_{M,i}$) as $\mathbf{A}_{M,i} = \mathbf{A}_{M,(i-1)}\mathbf{q}_i$. Therefore, vectors $\mathbf{q}_i$ are obtained by the product $\mathbf{q}_i = \mathbf{A}_{M,(i-1)}^{-1}\mathbf{A}_{M,i}$. This product can be interpreted as a linear regression of the relationships between the (*i*-1) first animals on their relationships with the *i*-th animal.

Dominance relationship matrix

Definition of dominance

Dominance is defined by Fisher (1918) as the portion of the partitioned phenotypic variance that results from allelic interactions at the same locus. A dominance effect is the genetic effect carried on by a given allelic combination. When two animals share common ancestors, it becomes therefore likely that they carry an identical allelic combination. A dominance relationship coefficient scales this likelihood. Among others (epistasis effects), dominance is the non-additive genetic effect that is the more relevant in domestic species evaluation (Gengler et al., 1998).

Dominance relationship coefficient d_{ij} between animals *i* (having parents *s* and *d*) and *j* (having parents *p* and *m*) can be obtained from the additive relationship coefficients by the formula (Henderson, 1985): $d_{ij} = .25(a_{sp}a_{dm} + a_{sm}a_{dp})$. The matrix containing all dominance relationship coefficients is denoted by **D** and is called the dominance relationship matrix.

Computation of the dominance relationship matrix

Using formula above, **D** is computed using **A**. Also, a general recursion formula to compute **D** has been outlined in Smith and Mäki-Tanila (1990). Note that both **A** and **D** can easily be derived from the gametic relationship matrix (see section "*Definition and use of the gametic relationships*").

Computation of the inverse of the dominance relationship matrix

Because dominance is inherited through pairs of parents, two full-sibs have the same rows and columns in **D** and therefore **D** is not of full rank. To overcome this singularity, Hoeschele and VanRaden (1991) partitioned the dominance effects into sire X dam subclass effects (and a within-subclass deviation due to Mendelian sampling). They developed an inversion algorithm that sets up the inverse of the covariance matrix (noted **F**) of sire X dam subclass effects. The individual dominance effects are then related to these subclass effects. A recursive rule exists to compute the subclass effects (*f*). If *S* and *D* denote the sire and dam of an animal, *SS* and *DS*, the parents of its sire, *SD* and *DD*, the parents of its dam, the *S-D* subclass effect ($f_{S,D}$) is obtained by:

$$f_{S,D} = .5\left(f_{S,SD} + f_{S,DD} + f_{D,SS} + f_{D,DS}\right) - .25\left(f_{SS,SD} + f_{SS,DD} + f_{DS,SD} + f_{DS,DD}\right) + e, \quad \text{(II.11)}$$

where *e* is a segregation residual. Their method includes in three steps:

(1) identification of all filled sire X dam subclasses (among 8 potential subclasses in equation II.11) that provide relationship ties;

(2) direct computation of the inverse of **F** (see Algorithm II.3);

(3) computation of the inverse of **D** using an incidence matrix that relates dominance effects to subclass effects.

Algorithm II.3. Computation of inverse of **F**, matrix of *n* filled subclasses.

```
for $i$ = 1 to $n$, do
    set up $\mathbf{b}_i$, a row vector of length $k$, containing the coefficient $f$ as in equation II.11, that corresponds to each of the $k$ parental subclasses identified for subclass $i$
    set up the relationship matrix $\mathbf{F}_i$ of order $k$, containing the relationship coefficients between the $k$ parental subclasses
    compute $r_{ii}$ (variance coefficient for subclass $i$) as $(1-\mathbf{b}_i'\mathbf{F}_i\mathbf{b}_i)^{-1}$
    compute the contribution of subclass $i$ to the inverse of F as $r_{ii}^{-1}\begin{bmatrix} 1 & \mathbf{b}_i \end{bmatrix}'\begin{bmatrix} 1 & \mathbf{b}_i \end{bmatrix}$ and add it to $\mathbf{F}^{-1}$ at the proper positions
```

Epistasis Matrices

Definition of epistasis

Epistasis is a term that refers to interactions between loci (Bateson, 1909; Sinnot et al., 1950). Epistasis interactions used in animal breeding are (Cockerham, 1952; Cockerham, 1954):

- the effect of a particular allele of a first locus on a particular allele of the second locus, additive by additive interaction (AXA);
- the effect of a particular allele of a first locus on a particular allelic combination at the second locus (additive by dominance interaction, AXD), or;
- the effect of a particular allelic combination at a first locus on a particular allelic combination at the second locus (dominance by dominance interaction, DXD).

Other epistasis matrix can also be cited (additive by additive by additive, additive by additive by dominance, and so on; see Henderson, 1985).

Computation and inversion of the additive by additive relationship matrix

The AXA relationship matrix, denoted by $\mathbf{A}_A$ hereafter, can be formed rapidly by forming **A** using the tabular method and squaring each element (VanRaden and Hoeschele, 1991; Cockerham, 1954; Henderson, 1985; Kempthorne, 1955).

Chang et al. (1989) have developed a direct computation of the inverse of $\mathbf{A}_A$ constructed using only sire and maternal grand-sire information. Their algorithm fills in the inverse matrix through a quick reading of the pedigree. However, the subclass effect sire X dam is included in the Mendelian sampling effect. VanRaden and Hoeschele (1991) have solved this drawback by setting up an algorithm that accounts for all subclass effects as for dominance (see equation II.11). The relationships between AXA effects (**u**) are modelled by the linear relation $\mathbf{u} = \mathbf{Pu} + \mathbf{P}_b\mathbf{u}_b + \mathbf{m}$, in which **P** and $\mathbf{P}_b$ are incidence matrices, $\mathbf{u}_b$ is the vector AXA effects of unknown ancestors and ancestors combinations and **m**, the vector of AXA Mendelian sampling effects. After manipulations, the inverse of **U**, covariance matrix of **u** divided by the AXA variance component, can be expressed as $(\mathbf{I} - \mathbf{P}')\mathbf{R}^{-1}(\mathbf{I} - \mathbf{P})$, where **R** is the covariance matrix of $\mathbf{P}_b\mathbf{u}_b + \mathbf{m}$ divided by the AXA variance component. An algorithm – similar to that for dominance, see Algorithm II.3 - is proposed to compute the inverse of **U**. This algorithm proceeds as follows.

(1) Identification of all AXA subclass effects, written in an expanded list. These subclasses include all animals and parental combinations that provide relationship ties. Therefore, the size of the AXA effects covariance matrix (**U**) may be several times the number of animal; this increased size is nonetheless offset by the resulting sparseness of its inverse.

(2) Forward reading of the expanded list created at step (1). For each individual in this list, coefficients pertaining to the individual and its sire, dam and sire-dam subclass effect are added to $\mathbf{U}^{-1}$; for each sire-dam subclass, coefficients pertaining to that subclass and its ancestor subclasses are added to $\mathbf{U}^{-1}$. For both individual and sire-dam subclass, values and number of coefficients vary depending on the number of known sources.

In an inbred population, the effects of sire, dam and sire-dam subclass are correlated and the values of coefficients are affected by inbreeding.

Computation and inversion of other epistasis matrices

Others fore-mentioned epistasis matrices are computed similarly as the AXA matrix, by a Hadamard product of dominance and/or additive genetic matrices.

Their inversion may be performed by classical inversion algorithms (Henderson, 1985; Palucci et al., 2007). A general methodological frame to solve a model including any epistasis effect (also, dominance effect) without inversion of the relationship matrix of this effect has been presented by Schaeffer (2003). This method computes solutions of the desired effects as a selection index from the additive genetic solutions and iteratively corrects the observations for these desired effects and computes additive genetic solutions until convergence is reached.

Discussion and conclusions on inversion technique for relationship matrices

A general framework for inversion of variance-covariance matrices of genetic effects may be drafted through the different kinds of genetic effects described and their associated relationship matrices.

The variance-covariance matrix of a genetic effect vector **v** is usually defined as the product (equation II.13) of a relationship matrix, say **W**, and the genetic variance component associated to this effect, say σ_v^2.

$$Var(\mathbf{v}) = \mathbf{W} \cdot \sigma_v^2 \qquad \text{(II.13)}$$

The vector **v** is modelled by a linear model: $\mathbf{v} = \mathbf{Bv} + \mathbf{e}$, where **B** is an incidence matrix that gathers dependencies between elements of **v** and **e** is a term accounting for a residue due to the particular element itself (or, undue to dependencies between elements of **v**). It has to be noted that elements of **v** must be ordered such that any element only depends of elements preceding him; that is matrix **B** must be lower triangular. Removing recursion of this model returns $\mathbf{v} = (\mathbf{I} - \mathbf{B})^{-1} \cdot \mathbf{e}$. Variance of **v** can thereby be expressed in terms of variance of the residual term **e** (equation II.14).

$$Var(\mathbf{v}) = \left(\mathbf{I} - \mathbf{B}\right)^{-1} \cdot Var(\mathbf{e}) \cdot \left(\mathbf{I} - \mathbf{B}'\right)^{-1} \qquad \text{(II.14)}$$

The covariance among residual terms is usually null, because these terms refer to the own specificity of the effect (individual, gamete or subclass). Consequently, the variance-covariance matrix of **e** is the product of a diagonal matrix, say **D**, by σ_v^2. Thereby, equating equations (II.13) and (II.14), it comes out that the relationship matrix associated to any of these described genetic effects can be expressed as $\mathbf{W} = \left(\mathbf{I} - \mathbf{B}\right)^{-1} \cdot \mathbf{D} \cdot \left(\mathbf{I} - \mathbf{B}'\right)^{-1}$, and a general expression of its inverse is:

$$\mathbf{W}^{-1} = \left(\mathbf{I} - \mathbf{B}'\right) \cdot \mathbf{D}^{-1} \cdot \left(\mathbf{I} - \mathbf{B}\right) \qquad \text{(II.15)}$$

It is worth noting that this expression is the inverse of the root-free Cholesky factorization of **W**, for which the lower triangular factor is $\left(\mathbf{I} - \mathbf{B}\right)^{-1}$. It has been proposed (Henderson, 1976) and shown (Tier and Sölkner, 1993) that setting up the inverse of **W** using formula in (II.15) sums up to adding the contributions of a list of numbered levels of effect (individuals, gametes, subclass effects) to a null matrix. This successive addition can be achieved for *x* levels at a time. Usually, *x* is equal to 1 but may be greater than 1 in some situations (Wang et al., 1995; Sargolzaei et al., 2006; Smith and Mäki-Tanila, 1990). If we assume the following partitions for the relationship matrix **W** after *i* additions of *x* levels ($\mathbf{W}_{(i)}$) and the corresponding matrix **B**:

$$\mathbf{W}_{(i)} = \begin{bmatrix} \mathbf{W}_{11} & \mathbf{W}'_{21} \\ \mathbf{W}_{21} & \mathbf{W}_{22} \end{bmatrix} \text{ and } \mathbf{B}_{(i)} = \begin{bmatrix} \mathbf{B}_{11} & \mathbf{0} \\ \mathbf{B}_{21} & \mathbf{B}_{22} \end{bmatrix},$$

then the *i*-th addition of *x* levels to the inverse returns the matrix $\mathbf{W}_{(i)}^{-1}$:

$$\mathbf{W}_{(i)}^{-1} = \begin{bmatrix} \mathbf{W}_{11}^{-1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} + \begin{bmatrix} -\mathbf{B}'_{21} \\ \mathbf{I}_{(x)} \end{bmatrix} \left[\mathbf{W}_{22} - \mathbf{B}_{21}\mathbf{W}'_{21}\right]^{-1} \begin{bmatrix} -\mathbf{B}_{21} & \mathbf{I}_{(x)} \end{bmatrix}. \qquad \text{(II.16)}$$

The computational step in equation (II.16) requires to know sub-matrices $\mathbf{W}_{22}$, $\mathbf{W}_{21}$ and $\mathbf{B}_{21}$.

Therefore, we conclude by defining a computationally efficient algorithm for inversion of a genetic relationship matrix, on the basis of equation (II.16), as an algorithm that provides means to set up these sub-matrices ($\mathbf{W}_{22}$, $\mathbf{W}_{21}$ and $\mathbf{B}_{21}$) at a reduced computational cost.

Setting up $\mathbf{B}_{21}$ often requires no computation because the dependency coefficients between levels of effects in **v** are *a priori* known (e.g. additive and gametic relationships). In some cases, e.g. MQTL matrices, few computations are required to set up these coefficients. Also, as shown by van Arendonk et al. (1994), these coefficients can be obtained by partitioned matrix theory. The original model of dependencies between levels in **v** can also be simplified by adding sub-levels, what enables to set up $\mathbf{B}_{21}$ more readily (Hoeschele and VanRaden, 1991; VanRaden and Hoeschele, 1991).

Setting up $\mathbf{W}_{22}$ and $\mathbf{W}_{21}$ is either implicit (e.g. gametic relationships and additive and dominance relationships of non inbred populations have all diagonal elements equal to 1), either requires explicit computation of the relationship matrix (e.g. MQTL matrices). In this second case (e.g. additive and dominance relationships of inbred populations and derived epistasis matrices), computation efficiency can be greatly enhanced using algorithms of partial computation of **A** (e.g. Quaas, 1976; Colleau, 2002).

References

Abdel-Azim G. and Freeman A.E., 2001. A rapid method for computing the inverse of the gametic covariance matrix between relatives for a marked Quantitative Trait Locus. *Genet. Sel. Evol.*, 33, 153-173.

Amin N., van Duijn C.M. and Aulchenko Y.S., 2007. A Genomic Background Based Method for Association Analysis in Related Individuals. *PLoS ONE.*, 2, e1274.

Banachiewicz T., 1937. Zur Berechnung der Determinanten, wie auch der Inversen und zur darauf basierten Au ösung der Systeme linearen Gleichungen. *Acta Astron. Ser C.*, 3, 41–67.

Bateson W., 1909. *Mendel's Principles of Heredity*, Cambridge: Cambridge University Press.

Bömcke E. and Gengler N., 2009. Combining microsatellite and pedigree data to estimate relationships among Skyros ponies. *J. Appl. Genet.*, 50, 133–143.

Chang H.L., Fernando R.L. and Gianola D., 1989. Inverse of an Additive × Additive Relationship Matrix Due to Sires and Maternal Grandsires. *J. Dairy Sci.*, 72, 3023–3034.

Chevalet C., Gillois M. and Khang J.V.T., 1984. Conditional probabilities of identity of genes at a locus linked to a marker. *Genet. Sel. Evol.*, 16, 1–13.

Cholesky A.-L., 1910. Sur la résolution numérique des systèmes d'équations linéaires. *Bull. Sabix Société Amis Bibl. Hist. École Polytech.*, 81–95.

Cockerham C.C., 1952. *Genetic Covariation Among Characteristics of Swine*, Doctoral dissertation: Iowa State College (United States of America).

Cockerham C.C., 1954. An Extension of the Concept of Partitioning Hereditary Variance for Analysis of Covariances among Relatives When Epistasis Is Present. *Genetics*, 39, 859–882.

Colleau J.-J., 2002. An indirect approach to the extensive calculation of relationship coefficients. *Genet. Sel. Evol.*, 34, 409-421.

Emik L.O. and Terrill C.E., 1949. Systematic Procedures for Calculating Inbreeding Coefficients. *J. Hered.*, 40, 51–55.

Fernando R.L. and Grossman M., 1989. Marker assisted selection using best linear unbiased prediction. *Genet. Sel. Evol.*, 21, 467-477.

Fisher R.A., 1918. The Correlation between Relatives on the Supposition of Mendelian Inheritance. *Earth Environ. Sci. Trans. R. Soc. Edinb.*, 52, 399–433.

Forni S., 2011. Different genomic relationship matrices for single-step analysis using phenotypic, pedigree and genomic information. *Genet. Sel. Evol.*, 43, 1.

Gengler N., Misztal I., Bertrand J.K. and Culbertson M.S., 1998. Estimation of the dominance variance for postweaning gain in the U.S. Limousin population. *J. Anim. Sci.*, 76, 2515–2520.

Gianola D. and van Kaam J.B.C.H.M., 2008. Reproducing Kernel Hilbert Spaces Regression Methods for Genomic Assisted Prediction of Quantitative Traits. *Genetics*, 178, 2289–2303.

Gibson J.P., Kennedy B.W., Schaeffer L.R. and Southwood O.I., 1988. Gametic models for estimation of autosomally inherited genetic effects that are expressed only when received from either a male or female parent. *J. Dairy Sci.*, 71 (Suppl. 1).

Goddard M.E., 1992. A mixed model for analyses of data on multiple genetic markers. *Theor. Appl. Genet.*, 83, 878–886.

Goddard M.E. and Hayes B.J., 2007. Genomic selection. *J. Anim. Breed. Genet.*, 124, 323–330.

Hayes B.J., Visscher P.M. and Goddard M.E., 2009. Increased accuracy of artificial selection by using the realized relationship matrix. *Genet. Res.*, 91, 47.

Henderson C.R., 1953. Estimation of Variance and Covariance Components. *Biometrics*, 9, 226-252.

Henderson C.R., 1973. Sire evaluation and genetic trends. In: *Proc. Anim. Breed. Genet. Symp. Honor Dr Jay Lush.*, Am. Soc. Anim. Sci. and Am. Dairy Sci. Assoc., Poultry Sci. Assoc., Champaign, IL, 10-41.

Henderson C.R., 1976. A Simple Method for Computing the Inverse of a Numerator Relationship Matrix Used in Prediction of Breeding Values. *Biometrics*, 32, 69–83.

Henderson C.R., 1985. Best Linear Unbiased Prediction of Nonadditive Genetic Merits in Noninbred Populations. *J. Anim. Sci.*, 60, 111–117.

Hoeschele I. and VanRaden P.M., 1991. Rapid Inversion of Dominance Relationship Matrices for Noninbred Populations by Including Sire by Dam Subclass Effects. *J. Dairy Sci.*, 74, 557–569.

Hoeschele I., 1993. Elimination of Quantitative Trait Loci Equations in an Animal Model Incorporating Genetic Marker Data. *J. Dairy Sci.*, 76, 1693–1713.

Jamrozik J. and Schaeffer L.R., 1991. An equivalent gametic model for animal dominance genetic linear model. *J. Anim. Breed. Genet.*, 108, 343–348.

Kempthorne O., 1955. The Theoretical Values of Correlations between Relatives in Random Mating Populations. *Genetics*, 40, 153–167.

Kennedy B.W., Schaeffer L.R. and Sorensen D.A., 1988. Genetic Properties of Animal Models. *J. Dairy Sci.*, 71, 17–26.

Kennedy B.W. and Schaeffer L.R., 1989. Genetic Evaluation Under an Animal Model When Identical Genotypes Are Represented in the Population. *J. Anim. Sci.*, 67, 1946–1955.

Leutenegger A.-L., Prum B., Génin E., Verny C., Lemainque A., Clerget-Darpoux F. and Thompson E.A., 2003. Estimation of the Inbreeding Coefficient through Use of Genomic Data. *Am. J. Hum. Genet.*, 73, 516–523.

Matsuda H. and Iwaisaki H., 2002. A recursive procedure to compute the gametic relationship matrix and its inverse for marked QTL clusters. *Genes Genet. Syst.*, 77, 123–130.

Meuwissen T.H.E. and Luo Z., 1992. Computing inbreeding coefficients in large populations. *Genet. Sel. Evol.*, 24, 305-313.

Meuwissen T.H.E., Hayes B.J. and Goddard M.E., 2001. Prediction of total genetic value using genome-wide dense marker maps. *Genetics*, 157, 1819–1829.

Mugnier M., Sutter J. and Goux J.-M., 1966. Organigrammes pour l'étude mécanographique de la parenté et de la fécondité dans une population. *Popul. Fr. Ed.*, 21, 75–98.

Nejati-Javaremi A., Smith C. and Gibson J.P., 1997. Effect of total allelic relationship on accuracy of evaluation and response to selection. *J. Anim. Sci.*, 75, 1738–1745.

Oikawa T. and Yasuda K., 2009. Inclusion of genetically identical animals to a numerator relationship matrix and modification of its inverse. *Genet. Sel. Evol.*, 41:25.

Palucci V., Schaeffer L.R., Miglior F. and Osborne V., 2007. Non-additive genetic effects for fertility traits in Canadian Holstein cattle. *Genet. Sel. Evol.*, 39, 1–13.

Pearl R., 1917a. The Probable Error of a Mendelian Class Frequency. *Am. Nat.*, 51, 144–156.

Pearl R., 1917b. The Selection Problem. *Am. Nat.*, 51, 65–91.

Quaas R.L., 1976. Computing the Diagonal Elements and Inverse of a Large Numerator Relationship Matrix. *Biometrics*, 32, 949–953.

Quaas R.L. and Pollak E.J., 1980. Mixed model methodology for farm and ranch beef cattle testing programs. *J. Anim. Sci.*, 51, 1277–1287.

Rajagopalan J., 1996. *An Iterative Algorithm for Inversion of Matrices*. Montreal: Concordia University.

Sargolzaei M., Iwaisaki H. and Colleau J.-J., 2005. A fast algorithm for computing inbreeding coefficients in large populations. *J. Anim. Breed. Genet.*, 122, 325–331.

Sargolzaei M., Iwaisaki H. and Colleau J.-J., 2006. Efficient computation of the inverse of gametic relationship matrix for a marked QTL. *Genet. Sel. Evol.*, 38, 253-264.

Schaeffer L.R., Kennedy B.W. and Gibson J.P., 1989. The Inverse of the Gametic Relationship Matrix. *J. Dairy Sci.*, 72, 1266–1272.

Schaeffer L.R., 2003. Computing simplifications for non-additive genetic models. *J. Anim. Breed. Genet.*, 120, 394–402.

Sherman J. and Morrison W.J., 1950. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *Ann. Math. Stat.*, 124–127.

Sinnot E.W., Dünn L.C. and Dobzhansky T., 1950. *Principles of genetics*, New York: McGraw-Hill Book Co.

Smith C. and Simpson S.P., 1986. The use of genetic polymorphisms in livestock improvement. *J. Anim. Breed. Genet.*, 103, 205–217.

Smith S.P., 1984. Dominance relationship matrix and inverse for an inbred population. *Unpubl. Mimeo Dept. Dairy Sci; Ohio State Univ.*

Smith S.P. and Allaire F., 1985. Efficient selection rules to increase non-linear merit: application in mate selection. *Genet. Sel. Evol.*, 17, 387–406.

Smith S.P. and Mäki-Tanila A., 1990. Genotypic covariance matrices and their inverses for models allowing dominance and inbreeding. *Genet. Sel. Evol.*, 22, 65-91.

Soller M. and Beckmann J.S., 1983. Genetic polymorphism in varietal identification and genetic improvement. *Theor. Appl. Genet.*, 67, 25–33.

Su G., Christensen O.F., Ostersen T., Henryon M. and Lund M.S., 2012. Estimating Additive and Non-Additive Genetic Variances and Predicting Genetic Merits Using Genome-Wide Dense Single Nucleotide Polymorphism Markers. *PLoS ONE*, 7, e45293.

Tier B. and Sölkner J., 1993. Analysing gametic variation with an animal model. *Theor. Appl. Genet.*, 85, 868–872.

Tuchscherer A., Mayer M. and Reinsch N., 2004. Identification of gametes and treatment of linear dependencies in the gametic QTL-relationship matrix and its inverse. *Genet. Sel. Evol.*, 36, 621–642.

van Arendonk J.A.M., Tier B. and Kinghorn B.P., 1994. Use of multiple genetic markers in prediction of breeding values. *Genetics*, 137, 319–329.

VanRaden P.M. and Hoeschele I., 1991. Rapid Inversion of Additive by Additive Relationship Matrices by Including Sire-Dam Combination Effects. *J. Dairy Sci.*, 74, 570–579.

VanRaden P.M., 2007. Genomic measures of relationship and inbreeding. *Proc Interbull Annu. Meet.*, 33–36.

VanRaden P.M., 2008. Efficient methods to compute genomic predictions. *J. Dairy Sci.*, 91, 4414-4423.

Villanueva B., Pong-Wong R., Fernandez J. and Toro M.A., 2005. Benefits from marker-assisted selection under an additive polygenic genetic model. *J. Anim. Sci.*, 83, 1747–1752.

Wang T., Fernando R.L., van der Beek S., Grossman M. and van Arendonk J., 1995. Covariance between relatives for a marked quantitative trait locus. *Genet. Sel. Evol.*, 27, 251-274.

Wright S., 1922. Coefficients of Inbreeding and Relationship. *Am. Nat.*, 56, 330–338.

BRIEF SUMMARY OF CHAPTER II

- The inverse of the additive genetic relationship matrix (**A**) may be computed using an algorithm that successively updates a zeroed matrix. This algorithm has a linear complexity with the order of the matrix (number of animals in population).
- Inverses of most of the other genetic relationship matrices follows the same strategy: (1) definition of the levels of the genetic effect, (2) for each level, direct computations of dependencies with levels preceding the current one, and (3) for each level, successive update of a zeroed matrix with the vector of contributions. This strategy features a general framework for inversion of relationship matrices.
- We propose to assess the use of the same framework for computation and/or approximation of the inverse of $\mathbf{G}$ and $\mathbf{A}_{22}$. As each level of the genetic effect features a genotyped animal in these specific cases, the aim will be to determinate (or approximate the determination) of the dependencies of this animal with other genotyped animals. It covers two aspects: (1) to determine which genotyped animal (later denoted as contributor) contributes to the current animal, and (2) to compute the value of its contribution.

Chapter III

THE CLOSE-FAMILY APPROACH

In the previous chapter, a framework was proposed for inversion of relationship matrices. This framework dictates to determine dependencies between genotyped animals. In this chapter, we propose to implement the framework to **G** and $\mathbf{A}_{22}$ using two algorithms to approximate the determination of contributors.

In the first algorithm, contributors of an animal are its close-related genotyped animals.

In the second algorithm, contributors are firstly found for the initial matrix. Then, a residual matrix is computed and the same approach is applied to obtain an approximation of the inverse of that residual matrix. The process is then recursively applied until convergence.

[FROM: P. Faux, N. Gengler and I. Misztal. 2012. A recursive algorithm for decomposition and creation of the inverse of the genomic relationships matrix. *Journal of Dairy Science*, 95: 6093-6102]

Abstract

Some genomic evaluation models require creation and inversion of a genomic relationship matrix (**G**). As the number of genotyped animals increases, **G** becomes larger and thus requires more time for inversion. A single-step genomic evaluation also requires inversion of the part of the pedigree relationship matrix for genotyped animals ($\mathbf{A}_{22}$). A strategy was developed to provide an approximation of the inverse of **G** ($\tilde{\mathbf{G}}^{-1}$) that may also be applied to the inverse of $\mathbf{A}_{22}$ ($\tilde{\mathbf{A}}_{22}^{-1}$). The algorithm was based on direct inversion of the pedigree relationship matrix (**A**), which involves a recursion for each animal and its parents. Decomposition of **G** was similar to that for **A** except that more relatives were involved for each animal. The $\tilde{\mathbf{G}}^{-1}$ was computed as the matrix product $(\mathbf{T}^{-1})'\mathbf{D}^{-1}\mathbf{T}^{-1}$, where $\mathbf{T}^{-1}$ and $\mathbf{D}^{-1}$ are inverses of the triangular and diagonal matrices **T** and **D**, respectively. The weights for each relative were determined by variables of regression of the genomic relationships among all genotyped animals older than a given animal on the genomic relationships for that animal. The resulting estimators were used to create $\mathbf{T}^{-1}$. Then, $\mathbf{T}^{-1}\mathbf{G}(\mathbf{T}^{-1})'$ resulted in a new matrix that is close to diagonal and also needs to be inverted. The inverse of that matrix was approximated with the same decomposition as for approximation of the inverse of **G** ($\tilde{\mathbf{G}}^{-1}$), and the procedure was repeated in successive rounds of recursion until a matrix was obtained that was close enough to diagonal to be inverted element by element. Two applications of the approximation algorithm were tested in a single-step genomic evaluation of US Holstein final score, and correlation coefficients between estimated breeding values based on either real or approximated $\mathbf{G}^{-1}$ were compared. Approximations came closer to $\mathbf{G}^{-1}$ as the number of recursion rounds increased. Approximations were even more accurate and faster for $\mathbf{A}_{22}$. Time-saving strategies are needed to reduce the computing time required for the algorithm.

Keywords: genomic selection, relationship matrix, matrix inversion, algorithm

INTRODUCTION

Meuwissen et al. (2001) proposed a method to predict breeding values that includes molecular information from dense marker panel that cover the whole genome. This method, called genomic prediction, relies on the assumption that markers are expected to be in linkage disequilibrium with potential Quantitative trait loci (QTL). Among different methods proposed to implement genomic prediction, VanRaden (2007) suggested that a genomic relationship matrix (**G**) could be computed based on molecular knowledge of alleles shared between individuals and that mixed models using such a matrix instead of a pedigree-based matrix (**A**) could predict genetic effects more accurately than those using **A**. The main structural difference between **A** and **G** is that individuals assumed to be unrelated based on pedigree almost always share fractions of their genomes. Therefore, **G** is expected to be dense, whereas **A** may be sparse, and use of **G** in genomic BLUP (GBLUP) consequently requires inversion of a dense matrix.

Legarra et al. (2009) developed a method that includes pedigree, phenotypic, and genomic information in one step, which was also shown by Christensen and Lund (2010) and mentioned by Bömcke et al. (2011). That method requires inversion of **G** as well as inversion of the part of **A** that represents pedigree-based relationships among genotyped animals ($\mathbf{A}_{22}$). Although **A** may be sparse and able to be inverted using the rules of Henderson (1976) and Quaas (1976), $\mathbf{A}_{22}$ is expected to be less sparse and cannot be inverted using those rules, because it contains relationships from a non-genotyped ancestor. The **G** and $\mathbf{A}_{22}$ matrices can be created and inverted by standard procedures such as those in the program PREGSF90 (Aguilar et al., 2011) by using a package of linear algebra kernels (Basic Linear Algebra Subprograms; Lawson et al., 1979) that are able to multiply matrices efficiently. Using PREGSF90, Aguilar et al. (2011) showed that creation and inversion of $\mathbf{A}_{22}$ and **G** for 30,000 animals with 40,000 SNP required approximately 3 hr. However, their computations increase cubically with the number of animals and are unsuitable for very large numbers of genotypes.

The inverse of **A** ($\mathbf{A}^{-1}$) can be calculated quickly and with minimum storage, because it is based on a recursion formula involving only 3 individuals: an animal and its parents. For direct inversion of **G**, genomically enhanced EBV (**GEBV**) of a genotyped animal is assumed to be dependent on all other genotyped animals. The first objective of this study was to develop and evaluate a recursion formula for the inverse of **G** ($\mathbf{G}^{-1}$) that includes only a fraction of genotyped animals. The second objective was to determine whether such a formula is also suitable for creation of the inverse of $\mathbf{A}_{22}$ ($\mathbf{A}_{22}^{-1}$).

Material and Methods

Approximation of Inverse of G and $\mathbf{A}_{22}$

State of the art

Development of approximations will be based on root-free Cholesky factorization. If the genomic relationship matrix **G** is a symmetric positive definite matrix, then **G** is invertible and may be factorized as $\mathbf{G} = \mathbf{LL}'$. The decomposition of its inverse is thus $(\mathbf{L}^{-1})'\mathbf{L}^{-1}$, what is equal, if $\mathbf{L} = \mathbf{T}\sqrt{\mathbf{D}}$, to the following root-free Cholesky factorization of $\mathbf{G}^{-1}$:

$$\mathbf{G}^{-1} = (\mathbf{T}^{-1})'\mathbf{D}^{-1}\mathbf{T}^{-1} \quad \text{(III.1)}$$

We will focus hereafter on approximating the inverse of the root-free Cholesky factor ($\mathbf{T}^{-1}$) rather than the root-free Cholesky factor itself ($\mathbf{T}$).

Even if less popular than approximations of the Cholesky factor, sparse approximations of the inverse of the Cholesky factor of a nonsingular coefficient matrix, for instance, **M**, are frequently used for computation of preconditioners for conjugate gradient calculations of linear systems involving **M**. In a comparative study, Benzi and Tuma (1999) identify two main approaches of computing such sparse approximations. In the first approach – to which our algorithm belongs –, the sparse approximation of the inverse of the Cholesky factor is directly computed from **M**. Information about triangular factors of **M** is not required. Among others in the same approach, a method, by Kolotilina and Yeremin (1993) might be noticed that computes a factorized sparse approximate inverse (FSAI) by minimizing the Frobenius norm $\|\mathbf{I} - \mathbf{PQ}_A\|_F$, where **P** is the sparse approximation of $\mathbf{Q}_A^{-1}$ and $\mathbf{Q}_A$ is the Cholesky factor of a symmetric **M**, between the approximate and the real triangular factor. In contrast, the second approach gathers methods that require an incomplete factorization of **M** and that will use this incomplete factorization to obtain a sparse approximation of the inverse of the Cholesky factor. Nevertheless, our aim is not to use a sparse approximation of factors of the inverse for preconditioning, but well for approximating the inverse of the matrix.

Algorithm for Approximation of Inverse of Cholesky factor

Based on equation (III.1), we propose an approximation ($\tilde{\mathbf{G}}^{-1}$) of $\mathbf{G}^{-1}$, expressed as

$$\tilde{\mathbf{G}}^{-1} = \tilde{\mathbf{T}}'\tilde{\mathbf{D}}^{-1}\tilde{\mathbf{T}}, \quad \text{(III.2)}$$

where $\tilde{\mathbf{T}}$ is an approximation of $\mathbf{T}^{-1}$ and $\tilde{\mathbf{D}}^{-1}$ is an approximation of the inverse of $\mathbf{D}$.

The method that we developed to create a sparse $\tilde{\mathbf{T}}$ will be illustrated through an example pedigree (Figure 1) for 4 animals (*s*, *d*, *j*, and *i*). Animals *s* and *d* are parents of *i*, and animal *j* represents any other animal older than *i* and related to *i* through ancestors of *s* and *d*. Additional ancestors that explain the relationships among *j*, *s*, and *d* are omitted. For the example pedigree, assume that genomic relationships (*g*) among animals are

$$\begin{bmatrix} g_{ss} & g_{sd} & g_{sj} & g_{si} \\ g_{ds} & g_{dd} & g_{dj} & g_{di} \\ g_{js} & g_{jd} & g_{jj} & g_{ji} \\ g_{is} & g_{id} & g_{ij} & g_{ii} \end{bmatrix} = \begin{bmatrix} 1.13 & 0.04 & 0.17 & 0.58 \\ 0.04 & 1.02 & 0.32 & 0.47 \\ 0.17 & 0.32 & 0.98 & 0.22 \\ 0.58 & 0.47 & 0.22 & 1.09 \end{bmatrix}$$

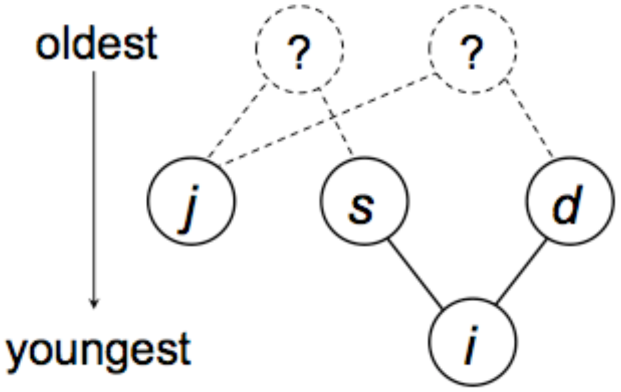

Figure 1 **Pedigree relationships between 4 animals *s*, *d*, *j*, and *i*, where *s* and *d* are parents of *i* and *j* is any animal older than *i* and related to both *s* and *d* through ancestors.**

Genomic relationships stretch across the whole pedigree, potentially across breeds. Therefore we will introduce the concept of close family where "close-family" of any animal *i* (Ω_i) may be defined by a genomic relationship threshold *p* as

$$\forall (j < i), \Omega_i = \left\{ \left| g_{ij} \right| \geq p \right\}. \quad \text{(III.3)}$$

Note that, due to the condition on j, the close-family is restricted to animals older than i. In the current example, we have defined p as equal to 0.15.

The algorithm will fill in $\tilde{\mathbf{T}}$ as follows, for any animal i in pedigree:

i. select all animals in the close-family of i, i.e. animals j as defined by expression (III.3) ;

ii. perform the regression of relationships between those animals on relationships that they share with animal i;

iii. fill in positions in $\tilde{\mathbf{T}}$ that correspond to animals selected in Ω_i with the opposite of solutions and set diagonal element of $\tilde{\mathbf{T}}$ to 1.

For the example mentioned before, we run here the algorithm for the last animal in pedigree, i:

i. $\Omega_i = \left\{ s, d, j \right\}$, because all animals in pedigree have a genomic relationship with i greater than p=0.15;

ii. the regression to be performed is, here also, illustrated by the example:

$$\begin{bmatrix} g_{ss} & g_{sd} & g_{sj} \\ g_{ds} & g_{dd} & g_{dj} \\ g_{js} & g_{jd} & g_{jj} \end{bmatrix} \cdot \boldsymbol{\beta} + \boldsymbol{\varepsilon} = \begin{bmatrix} g_{si} \\ g_{di} \\ g_{di} \end{bmatrix} \Leftrightarrow \begin{bmatrix} 1.13 & 0.04 & 0.17 \\ 0.04 & 1.02 & 0.32 \\ 0.17 & 0.32 & 0.98 \end{bmatrix} \cdot \boldsymbol{\beta} + \boldsymbol{\varepsilon} = \begin{bmatrix} 0.58 \\ 0.47 \\ 0.22 \end{bmatrix}, \qquad \text{(III.4)}$$

where $\boldsymbol{\beta}$ is a regression coefficients vector and $\boldsymbol{\varepsilon}$, a vector of errors of estimation; by ordinary least-squares, solutions $\hat{\boldsymbol{\beta}}$ of this regression are

$\begin{bmatrix} 0.50 & 0.44 & -0.01 \end{bmatrix}'$ and estimation errors are $\begin{bmatrix} 0 & 0 & > -0.01 \end{bmatrix}'$;

iii. $\tilde{\mathbf{T}}$ is filled in with opposite of these solutions and its diagonal elements are set to 1, which gives for the example the final $\tilde{\mathbf{T}}$:

$$\tilde{\mathbf{T}} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -0.14 & -0.31 & 1 & 0 \\ -0.50 & -0.44 & 0.01 & 1 \end{bmatrix}$$

As one can see, $\tilde{\mathbf{T}}$ is filled-in only in positions where the corresponding values in the lower triangular part of **G** are greater than *p*. This threshold was here fixed to 0.15. This threshold allows to control the sparsity of $\tilde{\mathbf{T}}$. This sparse approximation has to be compared with the matrix that it aims to approximate (see equation III.1), which actually is, by complete Cholesky factorization and inversion of the triangular factor:

$$\mathbf{T}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -0.04 & 1 & 0 & 0 \\ -0.14 & -0.31 & 1 & 0 \\ -0.50 & -0.44 & 0.01 & 1 \end{bmatrix}$$

The next step is now to compute **D** (see equation III.1) and to approximate its inverse ($\tilde{\mathbf{D}}^{-1}$).

Recursive Formula and Approximation of D^{-1}

A quick rearrangement of equation (III.1) gives:

$$\mathbf{D} = \mathbf{T}^{-1}\mathbf{G}(\mathbf{T}^{-1})' \qquad \text{(III.5)}$$

Replacing $\mathbf{T}^{-1}$ by its approximation $\tilde{\mathbf{T}}$ in the formula here above allows computing a **D** that is no longer diagonal, but close to diagonal. For the current example, **D** would be:

$$\mathbf{D} = \begin{bmatrix} 1.13 & 0.04 & 0 & 0 \\ 0.04 & 1.02 & 0 & 0 \\ 0 & 0 & 0.86 & 0 \\ 0 & 0 & 0 & 0.59 \end{bmatrix}.$$

Thus, **D** approaches becoming a diagonal matrix. Nevertheless, **D** has to be inverted or its inverse approximated ($\tilde{\mathbf{D}}^{-1}$). If **D** is almost a diagonal matrix, then inversion of only its diagonal elements may be a good approximation of its inverse. However, if **D** is not close enough to being diagonal, the algorithm given above can also be used to approximate $\mathbf{D}^{-1}$ instead of $\mathbf{G}^{-1}$.

Therefore by its recursive use, the algorithm leads to the following $\tilde{\mathbf{D}}^{-1}$ in round *n*. Similar to approximation in equation (III.2),

$$\tilde{\mathbf{D}}_{n-1}^{-1} = \tilde{\mathbf{T}}_n'\tilde{\mathbf{D}}_n^{-1}\tilde{\mathbf{T}}_n. \qquad \text{(III.6)}$$

equation (III.6) involves choice of the new parameter p, construction of the new matrix $\tilde{\mathbf{T}}_n$, and computation of the new matrix $\tilde{\mathbf{D}}_n = \tilde{\mathbf{T}}_n\tilde{\mathbf{D}}_{n-1}\tilde{\mathbf{T}}'_n$.

After t rounds of recursion, when the off-diagonal elements of $\tilde{\mathbf{D}}_n$ are considered to be small enough,

$$\tilde{\mathbf{G}}_t^{-1} = \left[\tilde{\mathbf{T}}'_1\tilde{\mathbf{T}}'_2(\ldots)\tilde{\mathbf{T}}'_t\right]\tilde{\mathbf{D}}_t^{-1}\left[\tilde{\mathbf{T}}_t(\ldots)\tilde{\mathbf{T}}_2\tilde{\mathbf{T}}_1\right], \qquad \text{(III.7)}$$

where $\tilde{\mathbf{D}}_t^{-1}$ is the matrix formed by the inverse of the diagonal elements of $\tilde{\mathbf{D}}_t$. If $\tilde{\mathbf{T}}_f = \left[\tilde{\mathbf{T}}_t(\ldots)\tilde{\mathbf{T}}_2\tilde{\mathbf{T}}_1\right]$, then equation (III.7) becomes

$$\tilde{\mathbf{G}}_{\mathrm{t}}^{-1} = \tilde{\mathbf{T}}'_{\mathrm{f}}\tilde{\mathbf{D}}_{\mathrm{t}}^{-1}\tilde{\mathbf{T}}_{\mathrm{f}}. \qquad \text{(III.8)}$$

Link with Inversion of A

The pedigree-based relationship matrix (**A**) contains relationships (a) that are obtained iteratively with the rule:

$$a_{ij} = 0.5a_{sj} + 0.5a_{dj}, \qquad \text{(III.9)}$$

where i and j are 2 animals (with j older than i) and s and d are the parents of i. Therefore, a_{ij} represents the pedigree-based relationship between i and j; a_{sj}, the relationship between s and j; and a_{dj}, the relationship between d and j.

For the example pedigree (Figure 1), the additive relationships among the 4 animals s, d, j and i may be:

$$\begin{bmatrix} a_{ss} & a_{sd} & a_{sj} & a_{si} \\ a_{ds} & a_{dd} & a_{dj} & a_{di} \\ a_{js} & a_{jd} & a_{jj} & a_{ji} \\ a_{is} & a_{id} & a_{ij} & a_{ii} \end{bmatrix} = \begin{bmatrix} 1.00 & 0.00 & 0.20 & 0.50 \\ 0.00 & 1.00 & 0.30 & 0.50 \\ 0.20 & 0.30 & 1.00 & 0.25 \\ 0.50 & 0.50 & 0.25 & 1.00 \end{bmatrix}.$$

Applying the algorithm of construction of $\tilde{\mathbf{T}}$ to the case of **A**, with the same threshold p (p=0.15), leads, for the fourth animal (i), to the regression in (III.10).

$$\begin{bmatrix} a_{si} \\ a_{di} \\ a_{ji} \end{bmatrix} = \begin{bmatrix} a_{ss} & a_{sd} & a_{sj} \\ a_{ds} & a_{dd} & a_{dj} \\ a_{js} & a_{jd} & a_{jj} \end{bmatrix} \boldsymbol{\beta} + \boldsymbol{\varepsilon} \Leftrightarrow \begin{bmatrix} 0.50 \\ 0.50 \\ 0.25 \end{bmatrix} = \begin{bmatrix} 1.00 & 0.00 & 0.20 \\ 0.00 & 1.00 & 0.30 \\ 0.20 & 0.30 & 1.00 \end{bmatrix} \boldsymbol{\beta} + \boldsymbol{\varepsilon}, \qquad \text{(III.10)}$$

which, due to the iterative rule of computation of pedigree-based relationships (see equation III.9), has trivial solutions. Whatever j is, solutions will always be $\beta_s = 0.5, \beta_d = 0.5, \beta_j = 0$, i.e. always zeros for all animals except for parents, for which solution are 0.5. Filling in $\tilde{\mathbf{T}}$ with opposite of these solutions is actually what Henderson (1976) proposed in his decomposition of $\mathbf{A}^{-1}$, used for direct computation of $\mathbf{A}^{-1}$:

$$\mathbf{A}^{-1} = (\mathbf{T}_A^{-1})'\mathbf{D}_A^{-1}\mathbf{T}_A^{-1}. \qquad \text{(III.11)}$$

The equivalence is easily shown when replacing unknowns by their trivial solutions (0.5) in equation (III.9):

$$\begin{bmatrix} a_{si} \\ a_{di} \\ a_{ji} \end{bmatrix} = \begin{bmatrix} a_{ss} & a_{sd} & a_{sj} \\ a_{ds} & a_{dd} & a_{dj} \\ a_{js} & a_{jd} & a_{jj} \end{bmatrix} \begin{bmatrix} 0.50 \\ 0.50 \\ 0 \end{bmatrix} + \boldsymbol{\varepsilon} \Rightarrow - \begin{bmatrix} a_{ss} & a_{sd} & a_{sj} \\ a_{ds} & a_{dd} & a_{dj} \\ a_{js} & a_{jd} & a_{jj} \end{bmatrix} \begin{bmatrix} 0.50 \\ 0.50 \\ 0 \end{bmatrix} + \begin{bmatrix} a_{si} \\ a_{di} \\ a_{ji} \end{bmatrix} \begin{bmatrix} 1 \end{bmatrix} = \boldsymbol{\varepsilon}, \quad \text{(III.12)}$$

and rearranging this last equation, which gives:

$$\begin{bmatrix} -0.50 & -0.50 & 0 & 1 \end{bmatrix} \begin{bmatrix} a_{ss} & a_{ds} & a_{js} \\ a_{sd} & a_{dd} & a_{jd} \\ a_{sj} & a_{dj} & a_{jj} \\ a_{si} & a_{di} & a_{ji} \end{bmatrix} = \boldsymbol{\varepsilon}' = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}. \qquad \text{(III.13)}$$

The product in equation (III.12) is the product $\tilde{\mathbf{T}}\mathbf{A}$ restricted to line i of $\tilde{\mathbf{T}}$ and columns s, d, and j of $\mathbf{A}$. As constructed here above, $\tilde{\mathbf{T}}$ is thus always equal to the highly sparse $\mathbf{T}_A^{-1}$ of equation (III.11). Furthermore, errors of estimation are always zero (equation III.13) and, in respect to equation (III.3), they correspond to off-diagonals of **D**, which ensures **D** to always be diagonal, and thus easily inverted.

What our algorithm performs on **G** is thus inspired by what is done in the direct computation of $\mathbf{A}^{-1}$, but extended, in our case, to all animals closely-related to a given animal, rather than only to his parents.

Applications of Algorithm

Two computing bottlenecks occur: construction of each $\tilde{\mathbf{T}}$ and matrix multiplications. In both cases, the number of estimates (k) for each line of $\tilde{\mathbf{T}}$ (i.e., the value chosen for p) is critical. A lower p results in longer time for ordinary least square estimation and a less sparse $\tilde{\mathbf{T}}$. Consequently, 2 different applications of the algorithm were examined. For the first application, *k* was defined by *p*. For the second application, *k* was predefined. Numbers of additional relatives were equal for each recursion round. Therefore, computation time was nearly the same for each creation of $\tilde{\mathbf{T}}$ for both applications. Details on the two applications are in the Appendix.

Tests of Algorithm

Computation of G and A22

The genomic relationship matrix was here computed as (VanRaden,2008): $\mathbf{G} = \frac{\mathbf{ZZ}'}{2\sum_{i=1}^{m} p_i(1-p_i)}$, where **Z** is a matrix of *n* rows (*n* = number of genotyped animals) and *m* columns (*m* = total number of markers) that contains genotypes centered around the frequency p_i of the second allele at locus *i*. As discussed by Forni et al. (2011), matrix compatibility can be obtained by using observed allelic frequency (rather than fixed allelic frequency) and rescaling the genomic relationship matrix so that the mean for diagonal elements is 1; that is, on the same scale as diagonal elements of $\mathbf{A}_{22}$. In addition, off-diagonal elements of **G** were also scaled to be comparable with off-diagonal elements of $\mathbf{A}_{22}$. All computations were performed using PREGSF90, which also created $\mathbf{A}_{22}$ using algorithm by Colleau (2002).

Phenotypic records, Pedigree and Genotypes

The algorithm was tested using data for the official May 2009 US Holstein genetic evaluation for final score, which were provided by Holstein Association USA Inc. (Brattleboro, VT). A total of 10,553,183 final score records from 6,296,878 cows were available as well as 9,120,198 pedigree records. A total of 6,931 bulls had been genotyped using the BovineSNP50 BeadChip (Illumina, San Diego, CA). After removal of uninformative and low-quality SNP, 38,416 SNP were used to estimate genomic relationship coefficients. For computational ease, a reduced sample of 1,718 genotypes was used. That sample included the 800 youngest genotyped animals and all their genotyped ancestors.

Evaluations were calculated with an animal model that included fixed effects for management group (herd-year-classifier), age group by classification year, and lactation stage by classification year as described by Tsuruta et al. (2004). Variance components were those used for national evaluation of final score. Heritability was equal to 0.35 and repeatability was equal to 0.67. Evaluations were calculated using approximations of $\mathbf{G}^{-1}$ and $\mathbf{A}_{22}^{-1}$ from both applications of the algorithm.

Tests on Approximations

Quality of approximation of $\mathbf{G}^{-1}$ was tested by mean square difference (**MSD**) between $\mathbf{G}^{-1}$ and $\tilde{\mathbf{G}}^{-1}$ calculated as a weighted sum of square differences (*n* denotes here the size of the matrix):

$$MSD = \left[\sum_{i=1}^{n} \sum_{j=1}^{n} (\mathbf{G}_{ij}^{-1} - \tilde{\mathbf{G}}_{ij}^{-1})^2 \right] / n^2.$$

Moreover, the degree of sparsity of $\tilde{\mathbf{T}}_f$ that may be returned after any number of rounds of recursion was measured as the percentage of elements that equaled zero in the lower off-diagonal part of the matrix. Because multiplications of $\tilde{\mathbf{T}}_f$ provide the inverse approximation, a high percentage of elements in $\tilde{\mathbf{T}}_f$ equal to zero (i.e., a high degree of sparsity) should result in rapid computational time for the algorithm. Corresponding MSD and percentage of elements of $\tilde{\mathbf{T}}_f$ that equalled zero were also calculated for $\mathbf{A}_{22}^{-1}$ and its approximation ($\tilde{\mathbf{A}}_{22}^{-1}$).

Even if differences between $\mathbf{G}^{-1}$ and $\tilde{\mathbf{G}}^{-1}$ exists, they might not be relevant for genomic predictions. Therefore, each $\tilde{\mathbf{G}}^{-1}$ was used in the single-step procedure of Misztal et al. (2009) to compute approximated GEBV for final score including phenotypic, pedigree, and genomic information for 1,718 genotyped animals. Linear regressions of GEBV computed using approximated inverse of **G** on GEBV computed with real $\mathbf{G}^{-1}$ were estimated. Correlations between both sets of GEBV (computed using either real or approximated inverse) and standard deviations of GEBV computed using approximated inverse were also estimated.

Results and Discussion

Approximation of the inverse of G

The MSD and the percentage of elements of the lower triangular part of $\tilde{\mathbf{T}}_f$ that equalled zero are shown in Table 1 for the first application of the algorithm (five rounds of recursion with a smaller *p* at each round) and in Table 2 for the second application of the algorithm (eight rounds of recursion with a maximum *k* of 50 for all rounds).

Table 1 **Mean square differences (MSD) between real and approximated inverses of the genomic relationship matrix ($\mathbf{G}^{-1}$)** and percentages of elements that equaled zero (*t*0) in the lower part (excluding diagonals) of a triangular matrix used in approximating $\mathbf{G}^{-1}$ for 5 rounds of recursion of an approximation algorithm that defines the number of estimates for each line of the triangular matrix by the genomic relationship threshold (*p*) for 1,718 genotyped US Holsteins evaluated for final score in May 2009.

Recursion round	*p*	*MSD*[1]	*t0 (%)*
1	0.210	66.53×10^{-4}	92.35
2	0.017	34.87×10^{-4}	10.78
3	0.009	16.81×10^{-4}	2.28
4	0.005	5.97×10^{-4}	0.86
5	0.003	1.82×10^{-4}	0.49

[1] Mean Square Difference (MSD)

As for any approximation, MSD between $\mathbf{G}^{-1}$ and $\tilde{\mathbf{G}}^{-1}$ was always higher than MSD between $\mathbf{G}^{-1}$ and $\mathbf{A}_{22}^{-1}$ (81.49×10^{-4}), which suggests that any $\tilde{\mathbf{G}}^{-1}$ would be intermediate to $\mathbf{A}_{22}^{-1}$ and $\mathbf{G}^{-1}$. With more rounds of recursion, $\tilde{\mathbf{G}}^{-1}$ became closer to $\mathbf{G}^{-1}$, whereas $\tilde{\mathbf{T}}_f$ became denser. The close relationships among the Holstein bulls may partly explain the loss of sparsity. A population that included different breeds (or even different lines within breed) would be expected to have different results. However, in such multi-breed situation, our method could have an additional advantage, because animals from different breeds would not be closely-related (i.e. they would share genomic relationships lower than the threshold *p*), and would therefore generate an inverted matrix that separates breeds.

Estimation of GEBV with Approximate $\mathbf{G}^{-1}$

Slopes of linear regression of GEBV obtained using approximated inverse of **G** (denoted hereafter as "approximated GEBV") on GEBV obtained using real inverse of **G** (denoted hereafter as "real GEBV"), correlation coefficients between both sets of GEBV and standard deviations of approximated GEBV are in Table 3 for the first algorithm application and in Table 4 for the second application. Regardless of the algorithm application or the rounds of recursions, all slopes were higher than that for linear regression of EBV calculated without genomic information (hereafter denoted as "traditional EBV") on real GEBV (0.59).

Table 2 **Mean square differences (MSD) between real and approximated inverses of the genomic relationship matrix ($\mathbf{G}^{-1}$)** and percentages of elements that equaled zero (*t*0) in the lower part (excluding diagonals) of a triangular matrix used in approximating $\mathbf{G}^{-1}$ for 8 rounds of recursion of an approximation algorithm that defines the maximum number of estimates to be 50 for all rounds for 1,718 genotyped US Holsteins evaluated for final score in May 2009.

Recursion round	*MSD*[1]	*t0 (%)*
1	80.12×10^{-4}	94.27
2	53.02×10^{-4}	30.73
3	40.16×10^{-4}	11.07
4	32.04×10^{-4}	6.48
5	26.14×10^{-4}	4.43
6	21.59×10^{-4}	3.26
7	18.16×10^{-4}	2.50
8	15.37×10^{-4}	1.99

[1]Mean Square Difference (MSD)

All correlation coefficients were also larger than the correlation coefficient between traditional EBV and real GEBV (0.76). The standard deviations of all approximated GEBV ranged between the standard deviation of traditional EBV (10.69) and the standard deviation of real GEBV (13.77). The first application of the algorithm needed only three rounds to achieve results similar to those obtained after seven rounds of the second application. For both applications, a very similar relationship was observed between sparsity of the approximated $\mathbf{G}^{-1}$ and the correlation between GEBV: a decreasing sparsity in the factorization of $\mathbf{G}^{-1}$ leads to more similar GEBV. A strong increase in correlation between approximated and real GEBV was also observed in the first rounds of recursion for both applications, indicating that an important part of the genomic relationships was recovered.

Table 3 **Slopes of linear regression of GEBV based on approximated genomic relationship matrix (approximated GEBV) from 5 rounds of recursion** of an approximation algorithm that defines the number of estimates for each line of the triangular matrix by the genomic relationship threshold (*p*) on GEBV based on the actual genomic relationship matrix (real GEBV), coefficient of correlation (*r*) between approximated and real GEBV and standard deviations of approximated GEBV (*s*) for 1,718 genotyped US Holsteins evaluated for final score in May 2009.

Recursion round	*p*	*Slope*	*r*	*s*
1	0.210	0.68	0.79	11.81
2	0.017	0.86	0.91	13.04
3	0.009	0.92	0.97	13.54
4	0.005	0.97	0.99	13.54
5	0.003	0.99	> 0.99	13.66

Table 4 **Slopes of linear regression of GEBV based on approximated genomic relationship matrix (approximated GEBV) from 8 rounds of recursion** of an approximation algorithm that defines the maximum number of estimates to be 50 for all rounds on GEBV based on the actual genomic relationship matrix (real GEBV), coefficient of correlation (*r*) between approximated and real GEBV and standard deviations of approximated GEBV (*s*) for 1,718 genotyped US Holsteins evaluated for final score in May 2009.

Recursion round	*Slope*	*r*	*s*
1	0.75	0.77	13.39
2	0.84	0.87	13.36
3	0.87	0.90	13.30
4	0.89	0.93	13.23
5	0.92	0.95	13.41
6	0.94	0.96	13.47
7	0.95	0.97	13.55
8	0.96	0.98	13.59

Approximation of the inverse of $\mathbf{A}_{22}$

All $\tilde{\mathbf{A}}_{22}^{-1}$ were calculated with the same sequence of thresholds p used in the first application of the algorithm to calculate $\tilde{\mathbf{G}}^{-1}$ (Table 1) and the same value of k as in the second application (Table 2) for $\tilde{\mathbf{G}}^{-1}$. For each application, after the same number of recursion rounds, MSD were smaller for $\tilde{\mathbf{A}}_{22}^{-1}$ than for $\tilde{\mathbf{G}}^{-1}$, and $\tilde{\mathbf{T}}_f$ were much sparser (Table 5 for the first algorithm application and Table 6 for the second application). Furthermore, the non-zero elements of $\tilde{\mathbf{T}}_f$ tended to have a higher proportion of small elements (<0.01) when calculating $\tilde{\mathbf{A}}_{22}^{-1}$ compared with $\tilde{\mathbf{G}}^{-1}$ (Figure 2 for the first algorithm application and Figure 3 for the second application). Computational enhancement by removal of the smallest coefficients should thus be possible for calculating $\tilde{\mathbf{A}}_{22}^{-1}$. The results suggest that the algorithm is suitable for inversion of $\mathbf{A}_{22}$.

Table 5 **Mean square differences (MSD) between real and approximated inverses of the part of the pedigree-based relationship matrix that represents relationships among genotyped animals ($\mathbf{A}_{22}^{-1}$)** and percentages of elements that equaled zero (*t*0) in the lower part (excluding diagonals) of a triangular matrix used in approximating $\mathbf{A}_{22}^{-1}$ for 5 rounds of recursion of an approximation algorithm that defines the number of estimates for each line of the triangular matrix by the genomic relationship threshold (*p*) for US Holsteins evaluated for final score in May 2009.

Recursion round	*p*	MSD[1]	*t*0 (%)
1	0.210	2.11×10^{-4}	92.70
2	0.017	1.01×10^{-4}	88.32
3	0.009	0.79×10^{-4}	85.20
4	0.005	0.52×10^{-4}	81.88
5	0.003	0.30×10^{-4}	79.39

[1]Mean Square Difference (MSD)

Table 6 **Mean square differences (MSD) between real and approximated inverses of the part of the pedigree-based relationship matrix that represents relationships among genotyped animals ($\mathbf{A}_{22}^{-1}$)** and percentages of elements that equaled zero (*t*0) in the lower part (excluding diagonals) of a triangular matrix used in approximating $\mathbf{A}_{22}^{-1}$ for 8 rounds of recursion of an approximation algorithm that defines the maximum number of estimates to be 50 for all rounds for US Holsteins evaluated for final score in May 2009.

Recursion round	*MSD[1]*	*t0 (%)*
1	11.23×10^{-4}	94.49
2	4.83×10^{-4}	76.88
3	0.63×10^{-4}	71.46
4	0.14×10^{-4}	70.46
5	0.05×10^{-4}	69.80
6	0.02×10^{-4}	68.98
7	0.01×10^{-4}	67.91
8	$<0.01 \times 10^{-4}$	66.74

[1]Mean Square Difference (MSD)

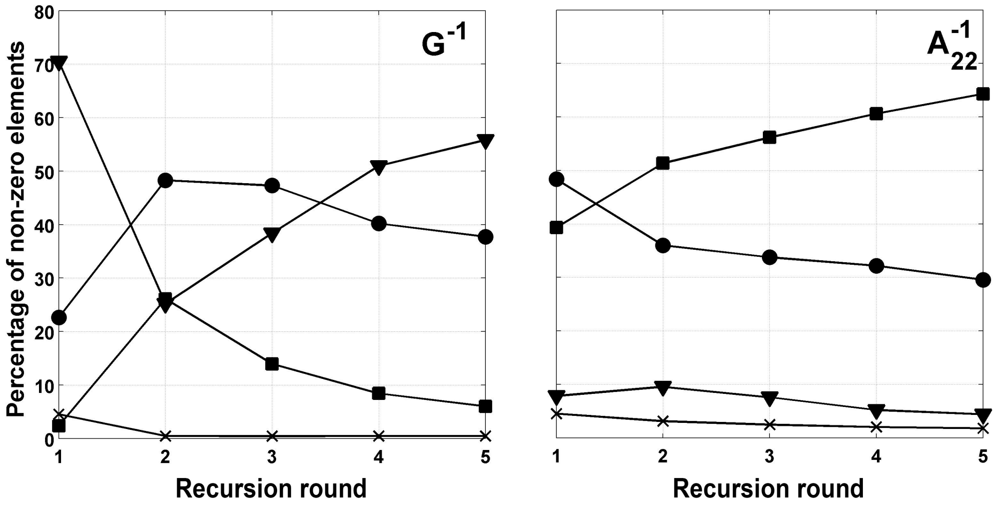

Figure 2 **Distribution of off-diagonal elements in the lower part (excluding diagonals) of a triangular matrix used in five rounds of recursion** of an algorithm that defines the number of estimates for each line of the triangular matrix by the genomic relationship threshold for US Holsteins evaluated for final score in May 2009 for approximation of the inverse of the genomic relationship matrix ($\mathbf{G}^{-1}$) and the part of the pedigree-based relationship matrix that represents relationships among genotyped animals ($\mathbf{A}_{22}^{-1}$); number of elements denoted as × between 10^{-1} and 1, ▲ between 10^{-2} and 10^{-1}, ● between 10^{-3} and 10^{-2}, ■ between 10^{-4} and 10^{-3} (element numbers of $<10^{-4}$ were considered to equal zero).

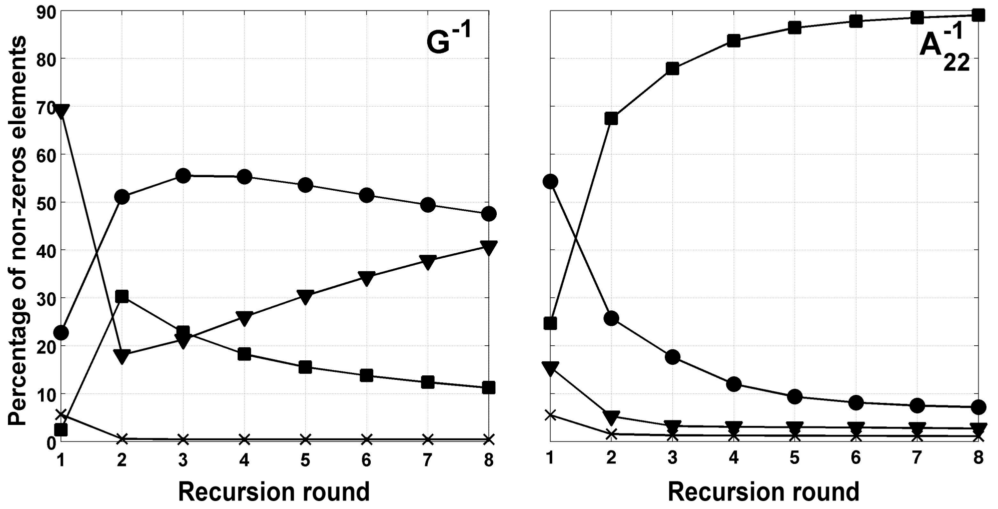

Figure 3 **Distribution of off-diagonal elements in the lower part (excluding diagonals) of a triangular matrix used in eight rounds of recursion** of an algorithm that defines the maximum number of estimates for each line of the triangular matrix to be 50 for all rounds for US Holsteins evaluated for final score in May 2009 for approximation of the inverse of the genomic relationship matrix ($\mathbf{G}^{-1}$) and the part of the pedigree-based relationship matrix that represents relationships among genotyped animals ($\mathbf{A}_{22}^{-1}$); number of elements denoted as × between 10^{-1} and 1, ▲ between 10^{-2} and 10^{-1}, ● between 10^{-3} and 10^{-2}, ■ between 10^{-4} and 10^{-3} (element numbers of $<10^{-4}$ were considered to equal zero).

Possible Improvements

The algorithm described provides a new insight on inversion of **G**. Each round of recursion returns an approximation that is better than the previous one, with convergence achieved in a few rounds. Consequently, the main issue for the proposed algorithm is the computing time. Presently, no efforts were made to study optimization, but reduction of computing time could be achieved in three ways. First, matrix computations could be optimized, e.g., as in Aguilar et al. (2011). Unfortunately, this algorithm would still have a cubic cost. Selecting animals on the basis of pedigree and creating genomic relationships only when required by this selection might avoid complete creation of **G**, and thus storage in RAM of the complete **G**. For the next rounds, elements of **D** would be kept in memory only if they exceed the next threshold *p*. Second, the number of animals selected for ordinary least squares could be decreased without compromising approximation accuracy. Hayes et al. (2009) stated that the gain in accuracy by including different breeds in the same relationship matrix is low or close to zero. Also, Muir (2007) found by simulation that genomic predictive ability strongly decays with generations under strong selection (compared with random selection). That finding was corroborated by practical studies in layer chickens (Wolc et al., 2011). Therefore, a recursion algorithm may eliminate animals that are more than one to two generations apart or are from another line or breed. With actual data sets, the number of such animals may be approximately constant even with increasing numbers of genotypes. Therefore, time for computation of any $\tilde{\mathbf{T}}$ would be equal regardless of the number of genotypes.

The third way to reduce computing time is related to the selection of regression variables for ordinary least squares. Two methods to select animals were presented: selection of all animals based on a genomic relationship threshold (first algorithm application) and selection of a specified number of closest-related animals (second algorithm application). For both selection methods, genomic (or $\mathbf{A}_{22}$) relationships for a given animal are defined by genomic (or $\mathbf{A}_{22}$) relationships among all its relatives. However, some relationships with relatives could be the same for two closely related animals; i.e., the incidence matrix in the ordinary least squares equations (equation III.4) of two animals could contain identical blocks. Thus, some matrix manipulation could easily lead to solutions for the second animal based on solutions for the first animal. This method of reducing computing time achieves some of the advantage realized from decreasing the number of animals selected for ordinary least squares. Moreover, animals are so far selected only on the basis of their closeness (using a threshold *p*) to deduct linear combinations between their relationships. This criterion (closeness) could to be enhanced; as for the case of **A**, some close animals (for instance, half-sibs) might not be needed whereas close parents of close animals would be. If *x* is an animal of interest for estimation of genomic relationships of a given animal *y* closely related to it, then, *z*, parent of *x* though not related to *y*, might be helpful for estimation because it would explain not the similarity but the difference between *x* and *y*.

The development of this approximation algorithm was based on the assumption that using a complete **G** for genomic evaluation is appropriate, and algorithm approximations were compared with **G** that links all genotyped animals. However, the inclusion of some distant relationships in **G** could be detrimental to evaluation accuracy. In such a case, the approximated **G**, which assumes contributions of closely related individuals from recent generations and from the same line, would result in higher accuracy. For example, if a few breeds were evaluated together with no predictability across breeds, $\mathbf{G}^{-1}$ from regular algorithms would contain nonzero off-diagonal elements across breeds. Those elements, which should be zero, would be a source of "noise" in GEBV. By selecting only closely related animals, the recursive approximation algorithm would set those elements to zero automatically and can be potentially more beneficial. However additional research is needed to clarify this point.

Conclusions

This recursive algorithm is powerful enough to approximate the inverse of a matrix such as **G** or $\mathbf{A}_{22}$. As the number of rounds of recursion increases, the inverse approximation becomes closer to the real inverse. The coefficient of correlation between approximated and real GEBV showed a strong increase in the first recursion rounds indicating that few rounds may be enough to recover important genomic relationships. Currently used applications of the algorithm can be optimized to achieve efficient selection of closely related animals. The algorithm may also be particularly suitable in the case of the inversion of $\mathbf{A}_{22}$, because it achieves a highly sparse factorization of this matrix within a few rounds of recursion. As quality of approximation depends on the number of rounds of recursion, the computing time required by each additional round is crucial and has to be optimized. As shown before, several computational improvements that could be developed in the future might reduce computational time needed.

Appendix

Application 1

The genomic matrix **G** first is ordered from oldest to youngest animal. For any row *i* from 2 to *n*, the algorithm proceeds as follows:

1. Select individuals older than animal *i* and with a genomic relationship with animal *i* that is larger than the parameter *p* defined for this recursion round. If none can be selected under that condition, take all individuals older than animal *i* in round 1 or ignore the line (i.e., none is selected) in the next rounds.
2. Perform ordinary least squares regression of the genomic relationships among those animals on their genomic relationships with animal *i*.
3. Update the lower triangular matrix $\tilde{\mathbf{T}}$ with the regression estimates on the off-diagonals and 1 on the diagonal.
4. Then, as in equation (III.6), $\tilde{\mathbf{T}}_1\mathbf{G}\tilde{\mathbf{T}}_1'$ in round 1 returns $\mathbf{D}_1$, and $\tilde{\mathbf{T}}_x\tilde{\mathbf{D}}_{x-1}\tilde{\mathbf{T}}_x'$ in each of the next rounds returns $\mathbf{D}_x$.
5. Finally, $\tilde{\mathbf{T}}_1$ in round 1 or $\tilde{\mathbf{T}}_f\tilde{\mathbf{T}}_x$ in each of the next rounds updates $\tilde{\mathbf{T}}_f$.

After *t* rounds, a last product returns $\tilde{\mathbf{G}}_t^{-1}$ using equation (III.8), where $\tilde{\mathbf{D}}_t^{-1}$ is made up of diagonal elements of $\mathbf{D}_t$. Steps 1 and 3 occur at each round but are different for round 1 and successive rounds. Steps 2 and 4 occur at each round. Step 5 does not occur for round 1. The assignation of a value to threshold *p* in step 1 has been made arbitrarily. This value nonetheless depends on the distribution of off-diagonals of the matrix which the inverse has to be approximated and may be deduced from this distribution.

Application 2

Only step 1 of application 1 is changed for application 2. Instead of selecting all animals that are older than animal *i* and have a genomic relationship with animal *i* that is larger than parameter *p*, a maximum of *k* animals is selected. If *i* is $\leq(k + 1)$, all animals are selected, and if *i* is $>(k + 1)$, the *k* animals with the largest genomic relationships with animal *i* are selected. This modification avoids regression with more than *k* estimators. In addition, each $\tilde{\mathbf{T}}$ created is sparser, and the increment is equal from each round to the next (*k* estimators are added).

Acknowledgements

The present project (NextGenGES–PHD-09-118) is supported by the National Research Fund, Luxembourg, as part of a private public partnership between CONVIS (Ettelbruck, Luxembourg) and Gembloux Agro-Bio Tech of the University of Liège. This project was partially supported by AFRI grants 2009-65205-05665 and 2010-65205-20366 from USDA NIFA. The authors gratefully acknowledge the financial support (project D31-1233/S1) of the Ministry of Agriculture of the Walloon Region of Belgium. Nicolas Gengler, who was Senior Research Associate of the National Fund for Scientific Research (FNRS) until December 31, 2011, acknowledges his support. The authors thank Holstein Association USA Inc. (Brattleboro, VT) and the Cooperative Dairy DNA Repository (Beltsville, MD) for providing genotypic data. Editorial help by Suzanne Hubbard is gratefully acknowledged. Helpful comments by reviewers are acknowledged.

References

Aguilar I., Misztal I., Legarra A. and Tsuruta S., 2011. Efficient computation of the genomic relationship matrix and other matrices used in single-step evaluation. *J. Anim. Breed. Genet.*, 128, 422–428.

Benzi M. and Tuma M., 1999. A comparative study of sparse approximate inverse preconditioners. *Appl. Numer. Math.*, 30, 305–340.

Bömcke E., Soyeurt H., Szydlowski M. and Gengler N., 2010. New method to combine molecular and pedigree relationships. *J. Anim. Sci.*, 89, 972–978.

Christensen O.F. and Lund M.S., 2010. Genomic prediction when some animals are not genotyped. *Genet. Sel. Evol.*, 42, 2.

Colleau J.-J., 2002. An indirect approach to the extensive calculation of relationship coefficients. *Genet. Sel. Evol.*, 34, 409–421.

Forni S., 2011. Different genomic relationship matrices for single-step analysis using phenotypic, pedigree and genomic information. *Genet. Sel. Evol.*, 43, 1.

Hayes B.J., Bowman P.J., Chamberlain A.C., Verbyla K. and Goddard M.E., 2009. Accuracy of genomic breeding values in multi-breed dairy cattle populations. *Genet. Sel. Evol.*, 41, 51.

Henderson C.R., 1976. A Simple Method for Computing the Inverse of a Numerator Relationship Matrix Used in Prediction of Breeding Values. *Biometrics*, 32, 69–83.

Kolotilina L.Y. and Yeremin A.Y., 1993. Factorized sparse approximate inverse preconditionings I: theory. *SIAM J Matrix Anal Appl.*, 14, 45–58.

Lawson C.L., Hanson R.J., Kincaid D.R. and Krogh F.T., 1979. Basic Linear Algebra Subprograms for Fortran Usage. *ACM Trans Math Softw.*, 5, 308–323.

Legarra A., Aguilar I. and Misztal I., 2009. A relationship matrix including full pedigree and genomic information. *J. Dairy Sci.*, 92, 4656–4663.

Meuwissen T.H.E., Hayes B.J. and Goddard M.E., 2001. Prediction of total genetic value using genome-wide dense marker maps. *Genetics*, 157, 1819–1829.

Misztal I., Legarra A. and Aguilar I., 2009. Computing procedures for genetic evaluation including phenotypic, full pedigree, and genomic information. *J. Dairy Sci.*, 92, 4648–4655.

Muir W.M., 2007. Comparison of genomic and traditional BLUP-estimated breeding value accuracy and selection response under alternative trait and genomic parameters. *J. Anim. Breed. Genet.*, 124, 342–355.

Quaas R.L., 1976. Computing the Diagonal Elements and Inverse of a Large Numerator Relationship Matrix. *Biometrics*, 32, 949–953.

Tsuruta S., Misztal I., Lawlor T.J. and Klei L., 2004. Modeling final scores in US Holsteins as a function of year of classification using a random regression model. *Livest. Prod. Sci.*, 91, 199–207.

VanRaden P.M., 2007. Genomic measures of relationship and inbreeding. *Proc Interbull Annu. Meet.*, 33–36.

VanRaden P.M., 2008. Efficient Methods to Compute Genomic Predictions. *J. Dairy Sci.*, 91, 4414–4423.

Wolc A., Arango J., Settar P., Fulton J.E., O'Sullivan N.P., Preisinger R., Habier D., Fernando R., Garrick D.J. and Dekkers J.C., 2011. Persistence of accuracy of genomic estimated breeding values over generations in layer chickens. *Genet Sel Evol.*, 43, 23.

BRIEF SUMMARY OF CHAPTER III

- Two algorithms were proposed to approximate the determination of contributors: the close-family approach and the recursive close-family approach. The contributions are then computed by linear regression.
- Use of the close-family approach is not time-expensive but not suited for the case of **G**. Use of the recursive close-family approach should be prohibitive in terms of computing time, albeit well suited for both matrices.
- In the case of $\mathbf{A}_{22}$, results show that a majority of the contributions are close to 0. It would therefore be meaningful to address the following questions: which contributions are actual zeros? Which contributions are actual non-zeros?

RELATED PUBLICATIONS

- P. Faux, I. Misztal and N. Gengler. 2010. Working on a Method to Compute Inverse of Genomic Relationship Matrix from Sparse Matrices. At: *ADSA-PSA-AMPA-CSAS-ASAS Joint Annual Meeting 2010*. Oral presentation in Denver, CO (USA).
- P. Faux, I. Misztal and N. Gengler. 2011. A recursive method of approximation of the inverse of genomic relationship matrix. At: *ADSA-ASAS Joint Annual Meeting 2011*. Oral presentation in New-Orleans, LA (USA).
- P. Faux and N. Gengler. 2012. Optimizing genomic prediction: Strategies to obtain inverse of large relationship matrices. At: *17th National Symposium on Applied Biological sciences*. Poster in Leuven (Belgium).

Chapter IV

THE SPARSITY PATTERN ALGORITHM

Introduced in the previous chapter, the close-family algorithm makes a brute approximation of the sparsity pattern of $\mathbf{A}_{22}$. Two types of error are then possible: discarding relevant contributors and considering non-contributing animals as contributors.

The pedigree contains the genealogical knowledge that is used to compute $\mathbf{A}$ and, consequently, to compute $\mathbf{A}_{22}$. In this chapter, we propose a heuristic algorithm that exhaustively searches the pedigree to find dependencies between genotyped animals; such a search results in setting up the sparsity pattern of the inverse Cholesky factor of $\mathbf{A}_{22}$ and, by logical matrix product, of $\mathbf{A}_{22}^{-1}$.

The inverse of $\mathbf{A}_{22}$ is then computed using prior information on its sparsity pattern to avoid useless computations.

[FROM: P. Faux and N. Gengler. 2013. Inversion of a part of the additive relationship matrix using pedigree information. *Genetics Selection Evolution*, 45:45]

Abstract

Background. In recent theoretical developments, the information available (e.g. genotypes) divides the original population into two groups: animals with this information (*selected* animals) and animals without this information (*excluded* animals). These developments require inversion of the part of the pedigree-based numerator relationship matrix that describes the genetic covariance between *selected* animals ($\mathbf{A}_{22}$). Our main objective was to propose and evaluate methodology that takes advantage of any potential sparsity in the inverse of $\mathbf{A}_{22}$ in order to reduce the computing time required for its inversion. This potential sparsity is brought out by searching the pedigree for dependencies between the *selected* animals. Jointly, we expected distant ancestors to provide relationship ties that increase the density of matrix $\mathbf{A}_{22}$ but that their effect on $\mathbf{A}_{22}^{-1}$ might be minor. This hypothesis was also tested.

Methods. The inverse of $\mathbf{A}_{22}$ can be computed from the inverse of the triangular factor ($\mathbf{T}^{-1}$) obtained by Cholesky root-free decomposition of $\mathbf{A}_{22}$. We propose an algorithm that sets up the sparsity pattern of $\mathbf{T}^{-1}$ using pedigree information. This algorithm provides positions of the elements of $\mathbf{T}^{-1}$ worth to be computed (i.e. different from zero). A recursive computation of $\mathbf{A}_{22}^{-1}$ is then achieved with or without information on the sparsity pattern and time required for each computation was recorded. For three numbers of *selected* animals (4000; 8000 and 12 000), $\mathbf{A}_{22}$ was computed using different pedigree extractions and the closeness of the resulting $\mathbf{A}_{22}^{-1}$ to the inverse computed using the fully extracted pedigree was measured by an appropriate norm.

Results. The use of prior information on the sparsity of $\mathbf{T}^{-1}$ decreased the computing time for inversion by a factor of 1.73 on average. Computational issues and practical uses of the different algorithms were discussed. Cases involving more than 12 000 *selected* animals were considered. Inclusion of 10 generations was determined to be sufficient when computing $\mathbf{A}_{22}$.

Conclusions. Depending on the size and structure of the *selected* sub-population, gains in time to compute $\mathbf{A}_{22}$ are possible and these gains may increase as the number of *selected* animals increases. Given the sequential nature of most computational steps, the

proposed algorithm can benefit from optimization and may be convenient for genomic evaluations.

Background

For a population of *n* animals, the numerator relationship matrix (**A**), is an *n*-by-*n* matrix with the following properties:

(1) a_{ij} is the numerator relationship coefficient between two animals *i* and *j* among *n*, as defined by Wright (1922);

(2) diagonal element a_{ii} is equal to $1+F_i$, where F_i is the inbreeding coefficient (Wright, 1922) of animal *i*;

(3) **A** is non-singular and symmetric: for two animals *i* and *j* among *n*, $a_{ij} = a_{ji}$.

Because the numerator relationship matrix describes the additive similarity between animals, it is an important element explaining genetic (co)variances between animals and has numerous applications in the field of animal genetics, the most important one being its use in setting up the mixed model equations for estimation of breeding values (Henderson, 1973).

In some situations, a particular type of information (genomic information, foreign genetic evaluation, phenotypes on a particular trait, etc.) is only available for some animals, which are selected for this particular purpose, while other animals are excluded. The original population can therefore be split into two sub-populations:

(1) a sub-population composed of animals called "*excluded*" hereafter;

(2) a sub-population composed of animals called "*selected*" hereafter.

Splitting the original population in this way leads to the following partition of **A**:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix}.$$

The four blocks include the relationships between *excluded* animals ($\mathbf{A}_{11}$), between *excluded* and *selected* animals ($\mathbf{A}_{12}$ and $\mathbf{A}_{21}$) and between *selected* animals ($\mathbf{A}_{22}$).

Recent methodological developments in animal breeding require inversion of $\mathbf{A}_{22}$, for example for genotyped animals in the context of genomic prediction using a single-step procedure (Misztal et al., 2009; Christensen and Lund, 2010; Gengler et al., 2012). Another example concerns external animals when integrating foreign information into a local genetic evaluation (Vandenplas and Gengler, 2012). It is also noteworthy that the pedigree-based relationship matrix $\mathbf{A}_{22}$ and the genomic relationship matrix (**G**; VanRaden, 2008) show structural similarities: both matrices express polygenic/genomic similarities among animals inherited from ancestors that are not represented in these matrices. Thus, the present research on $\mathbf{A}_{22}$ can be extended to genomic relationships in **G**.

Based on the original work by Henderson (1976) on inversion of **A**, a general framework for the inversion of relationship matrices follows (see Appendix). Henderson outlined a method that is based on the root-free factorization of **A** and showed the high sparsity of the inverse triangular factor of **A**. An efficient use of this sparsity then allows direct computation of $\mathbf{A}^{-1}$ as a sum of individual contributions based on a chronological reading of the pedigree. Applying partitioned matrix theory, van Arendonk et al. (1994) gave a general expression for the sum of individual contributions outlined by Henderson (1976): an additional row/column in **A** leads to updating its inverse by increasing the order by 1 and by summing the square of a very sparse vector to $\mathbf{A}^{-1}$. The very sparse vector is the corresponding row (below the diagonal) of the inverse triangular factor. All details on these developments are given in Appendix.

When required, the inverse of $\mathbf{A}_{22}$ is currently obtained by brutal inversion algorithms (e.g. generalized inverse algorithm). In these algorithms, any potential sparsity occurring in the matrix to invert or in its inverse is brought out by matrix computations. In contrast, the main objectives of this paper were to investigate how potential sparsity in the inverse triangular factor of $\mathbf{A}_{22}$ can be characterized using only the pedigree, thus without requiring matrix computations, and then use the sparsity pattern of the inverse triangular factor of $\mathbf{A}_{22}$ in the computation of its inverse. Whereas the structure of the inverse triangular factor is known for **A** (positions are given by the pedigree; values are *a priori* known), no information is available on the structure of the inverse triangular factor of $\mathbf{A}_{22}$, neither on the positions of non-zero elements nor on the values of these elements.

Moreover, the inverse triangular factor of $\mathbf{A}_{22}$ may be close to dense. Therefore, we addressed our objective in the following five steps:

(1) inversion of $\mathbf{A}_{22}$ with an algorithm that uses the inverse triangular factor;

(2) development of an algorithm that uses pedigree information to find the positions of the non-zero elements (sparsity pattern) in the inverse triangular factor of $\mathbf{A}_{22}$;

(3) inversion of $\mathbf{A}_{22}$ with the algorithm of step (1) but restricting computations to the non-zero elements identified by the algorithm in step (2);

(4) assessment of the time reduction when computing the inverse as in step (3) instead of as in step (1);

(5) and evaluation of the effect of the number of generations in the pedigree used to compute $\mathbf{A}_{22}$, in order to reduce density of the inverse triangular factor.

Methods

Blockwise inversion of A₂₂

For simplicity, we assume that we are working on the last *selected* animal, indexed as animal *n*. Similarly to inversion of **A** (see equation IV.A.6 in Appendix), assume that $\mathbf{A}_{22}$ is partitioned in a sub-matrix **Z**, of order (*n*-1), a (*n*-1)-long vector **y**, and a scalar *m* as:

$$\mathbf{A}_{22} = \begin{bmatrix} \mathbf{Z} & \mathbf{y} \\ \mathbf{y}' & m \end{bmatrix} \qquad \text{(IV.1)}$$

Using blockwise inversion, $\mathbf{A}_{22}^{-1}$ can be recursively computed using the following equation:

$$\mathbf{A}_{22}^{-1} = \begin{bmatrix} \mathbf{Z}^{-1} & \mathbf{0} \\ \mathbf{0}' & 0 \end{bmatrix} + \frac{1}{s} \cdot \begin{bmatrix} -\mathbf{Z}^{-1}\mathbf{y} \\ 1 \end{bmatrix} \begin{bmatrix} -\mathbf{y}'\mathbf{Z}^{-1} & 1 \end{bmatrix} \qquad \text{(IV.2)}$$

where *s* is a scalar equal to $m - \mathbf{y}'\mathbf{Z}^{-1}\mathbf{y}$.

Computing $\mathbf{b} = \mathbf{Z}^{-1}\mathbf{y}$ and defining $\alpha = s^{-1}$ simplifies equation (IV.2) as follows:

$$\mathbf{A}_{22}^{-1} = \begin{bmatrix} \mathbf{Z}^{-1} & \mathbf{0} \\ \mathbf{0}' & 0 \end{bmatrix} + \alpha \begin{bmatrix} -\mathbf{b} \\ 1 \end{bmatrix} \cdot \begin{bmatrix} -\mathbf{b}' & 1 \end{bmatrix} \qquad \text{(IV.3)}$$

Similarly, as for **A** (see Appendix), there is a link between vector **b** and the root-free Cholesky factorization of $\mathbf{A}_{22}$ ($\mathbf{A}_{22} = \mathbf{TDT}'$), in that $-\mathbf{b}'$ corresponds to the last row of the inverse triangular factor of $\mathbf{A}_{22}$ ($\mathbf{T}^{-1}$).

Equation (IV.3) shows that $\mathbf{A}_{22}^{-1}$ can be constructed recursively by adding a vector product to the previous result ($\mathbf{Z}^{-1}$). This recursive construction of $\mathbf{A}_{22}^{-1}$ will be called "Algorithm A" and implies, from the second row to the last row, the computation of the whole vector **b**.

If an animal and its parents are all *selected*, vector **b** is as sparse as in the case of **A**, i.e. the only non-zero elements of **b** correspond to parents. Restricting computations to these elements, i.e. discarding computations involving elements that we know equal 0, results in saving computing time. Such a case is, however, highly trivial. In the next sections, we propose a method to deal with more complex cases.

Contribution of *selected* animals to relationships in $\mathbf{A}_{22}$: characterizing the sparsity pattern of $\mathbf{T}^{-1}$

For animal *n*, vector **b** is the row of $\mathbf{T}^{-1}$ that spans from column 1 to column (*n*-1). By definition ($\mathbf{b} = \mathbf{Z}^{-1}\mathbf{y}$ thus $\mathbf{Zb} = \mathbf{y}$), vector **b** contains the required coefficients to compute relationships (**y**) of animal *n* with the *n*-1 preceding animals from the relationships between those *n*-1 preceding animals (**Z**). In the case of **A**, only known parents of animal *n* are required to compute its relationships with the preceding animals. Therefore, only positions of known parents have a value different from zero in vector **b**. In the case of $\mathbf{A}_{22}$, some *selected* animals replace the parents if they are *excluded*: the value in **b** of these *selected* animals is different from 0, which means that they are needed to compute relationships between *selected* animals (**y**) from the relationships between all *selected* animals (**Z**). This can be illustrated by the example pedigree in Figure 4 and Tables 7 and 8, which specify $\mathbf{A}_{22}$ and $\mathbf{T}^{-1}$ for the example pedigree. Three cases are outlined and detailed in the following:

(i) animal *G* has two known parents, *E* and *F*. Animal *E* is *excluded*; its parent *C* (grandparent of G) is thus required ($\mathbf{T}^{-1}_{CG} \neq 0$) to explain the relationship between *C* and *G* ($\mathbf{A}_{22;CG} = 0.25$).

(ii) animal *K* has one known parent, *F*, that is also *selected*. Any relationship that *K* shares with other *selected* animals is necessarily and only explained by *F* ($\forall X \neq F, \mathbf{T}^{-1}_{KX} = 0$).

(iii) animal *L* has one known parent, *E*, that is *excluded*. Its *selected* halfsib (*G*) and the *selected* parent of *G* (*F*, which is unrelated to *L*) are required, among others, to explain any relationship that *L* shares with other *selected* animals.

Table 7 **Matrix A₂₂ for the example of Figure 4**

	C	***F***	***G***	***I***	***J***	***K***	***L***
C	1.00		0.25		0.25		0.25
F		1.00	0.50			0.50	
G	0.25	0.50	1.00	0.06	0.06	0.25	0.25
I			0.06	1.00			0.06
J	0.25		0.06		1.00		0.06
K		0.50	0.25			1.00	
L	0.25		0.25	0.06	0.06		1.00

Empty cells are 0.

Table 8 **Inverse triangular factor ($\mathbf{T}^{-1}$) of A₂₂ for the example of Figure 4**.

	C	***F***	***G***	***I***	***J***	***K***	***L***
C	1.00						
F		1.00					
G	-0.25	-0.50	1.00				
I	0.02	0.05	-0.09	1.00			
J	-0.25				1.00		
K		-0.50				1.00	
L	-0.18	0.13	-0.27	-0.05			1.00

Empty cells are 0.

Animals that are required to explain relationships of a given *selected* animal with other *selected* animals will hereafter be denoted as the *contributors* of this *selected* animal. *Contributors* of a *selected* animal can be found by an exhaustive search of *selected* animals that replace any *excluded* parent of the *selected* animal. Their determination uses the pedigree and returns which elements of **b** (and thereby of $\mathbf{T}^{-1}$) are worth computing because they are expected to be non-zero. By subtraction, we obtain which elements are zeros, which is referred to as the “sparsity pattern” of $\mathbf{T}^{-1}$ in the following. In the next sub-section, we propose a heuristic algorithm that streamlines the determination of the sparsity pattern of $\mathbf{T}^{-1}$. Similar methodologies (Gilbert, 1994; George and Liu, 1980) have been developed for the triangular factor of a symmetric-positive definite matrix rather than the inverse of the triangular factor.

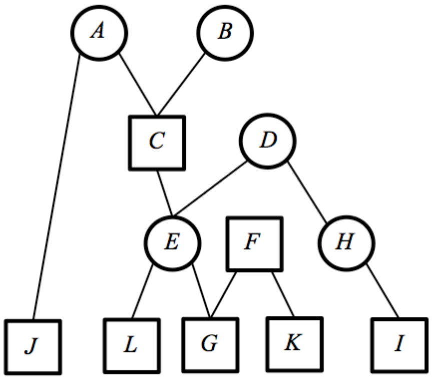

Figure 4 **Small example: a population of 12 animals.** Genealogical tree for a population of 12 animals, partitioned in sub-populations 1 (*excluded*, circle) and 2 (*selected*, square). Alphabetical order gives the birth order.

An algorithm to set up the sparsity pattern

Our proposed heuristic algorithm to set up the sparsity pattern of the inverse triangular factor of $\mathbf{A}_{22}$ (see pseudo-code below) requires two inputs: the pedigree (of length n_0, renumbered and ordered: parents precede progeny) and the subpopulation to which any animal belongs: *excluded* (population status is 1) or *selected* (population status is 2). The purpose of the algorithm is to complete two vectors of variable length for any animal *i*. The first vector ($\mathbf{r}_{(i)}$) contains references to *excluded* parents of animal *i*. The second vector ($\mathbf{c}_{(i)}$) contains *selected contributors* of animal *i*. The positions of non-zeros in the i-th row in $\mathbf{T}^{-1}$ (sparsity pattern) includes any position of the *i*-th row that is listed in $\mathbf{c}_{(i)}$.

```
Initialize a vector $\mathbf{x}$ as the integer sequence from 1 to $n_0$.

For any animal $i$ in the whole population ($i$ goes from 1 to $n_0$),

        (1) initialize two vectors $\mathbf{c}_{(i)}$ and $\mathbf{r}_{(i)}$ as empty vectors

        (2) if the status of animal $i$ is 2, then append element $i$ to $\mathbf{c}_{(i)}$; or else if the
            status of animal $i$ is 1, append element $i$ to $\mathbf{r}_{(i)}$
```

(3) if the sire s of animal i is known and its status is 2, then append element s to $\mathbf{c}_{(i)}$; or else if s is known but its status is 1, append vector $\mathbf{r}_{(s)}$ to $\mathbf{r}_{(i)}$

(4) if the dam d of animal i is known and its status is 2, then append element d to $\mathbf{c}_{(i)}$; or else if d is known but its status is 1, append vector $\mathbf{r}_{(d)}$ to $\mathbf{r}_{(i)}$

(5) if the status of animal i is 2 and the vector $\mathbf{r}_{(i)}$ is not empty, then:

a. select all elements of $\mathbf{x}$ that are at positions given in $\mathbf{r}_{(i)}$, remove duplicates and gather them in a temporary list $\mathbf{t}$

b. for any element k in list $\mathbf{t}$,

i. Append to $\mathbf{c}_{(i)}$ the elements of vector $\mathbf{c}_{(k)}$ not yet in $\mathbf{c}_{(i)}$

ii. Select elements of $\mathbf{x}$ that are equal to k and replace them by i;

or else if the status of animal i is 1 or if the vector $\mathbf{r}_{(i)}$ is empty, do nothing.

If the whole population was *selected* (i.e. $\mathbf{A}_{22} = \mathbf{A}$, every animal has status 2), it can be easily deduced from the algorithm that only the animal itself (in step (1)) and its known sire and dam (in steps (2) and (3)) would enter vector $\mathbf{c}_{(i)}$. The corresponding $\mathbf{T}^{-1}$ would be highly sparse, as it is for $\mathbf{A}$. This also means that if numerous parents are *selected*, then this algorithm is expected to run very fast.

An example of the use of this algorithm is given in the Results section.

Use of the sparsity pattern in blockwise inversion of A$_{22}$

The algorithm for blockwise inversion of $\mathbf{A}_{22}$ (Algorithm A, summarized in equation IV.3) is modified to account for sparsity and will be called Algorithm B. For simplicity, we still consider the last *selected* animal (animal n). Algorithm B reduces computations to obtain $\mathbf{b}$ from $\mathbf{y} = \mathbf{Zb}$ (equations IV.2 and IV.3) by three procedures, depending on the number (k) of elements in the corresponding vector $\mathbf{c}_{(n)}$ and the length of $\mathbf{b}$ (n-1).

The first procedure (called *EMPTY*) is used when $k = 0$ ($\mathbf{c}_{(n)}$ is empty). If so, only α is added to element $\mathbf{A}^{-1}_{22,nn}$. The value of α is just the inverse of $\mathbf{A}_{22,nn}$.

The second procedure (called *PROD*, for matrix PRODuct) is used when k is smaller than but relatively close to (n-1). In such a case, we perform a line-wise partition (equation IV.4) of **b** and $\mathbf{Z}^{-1}$ between non-zeros (of subscript u) and null (subscript v) entries of **b** in order to avoid useless computations:

$$\begin{bmatrix} \mathbf{b}_u \\ \mathbf{b}_v \end{bmatrix} = \begin{bmatrix} \mathbf{b}_u \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{Z}^u \\ \mathbf{Z}^v \end{bmatrix} \mathbf{y} \Rightarrow \quad \mathbf{b}_u = \mathbf{Z}^u \mathbf{y} \qquad \text{(IV.4)}$$

Since (n-1) is the number of elements in **b** and k the number of elements in $\mathbf{b}_u$, k dot products (of (n-1)-long vectors) would be performed instead of (n-1) dot products (of (n-1)-long vectors).

The third procedure (called *LS*, for Linear System of lower size) is used when k is much smaller than (n-1). In such a case, we extend the previous partition of **b** to a blockwise partition of **Z** and **y** (the non-zero and zero elements of **b** are respectively indexed by u and v):

$$\begin{bmatrix} \mathbf{b}_u \\ \mathbf{b}_v \end{bmatrix} = \begin{bmatrix} \mathbf{Z}_{uu} & \mathbf{Z}_{uv} \\ \mathbf{Z}_{vu} & \mathbf{Z}_{vv} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{y}_u \\ \mathbf{y}_v \end{bmatrix} \qquad \text{(IV.5)}$$

Then, applying partitioned matrix theory to equation (IV.5) returns the following expressions for $\mathbf{b}_u$ and $\mathbf{b}_v$ (with $\mathbf{S}_Z = \mathbf{Z}_{vv} - \mathbf{Z}_{vu}\mathbf{Z}_{uu}^{-1}\mathbf{Z}_{uv}$):

$$\begin{cases} \mathbf{b}_u = \mathbf{Z}_{uu}^{-1}\mathbf{y}_u \quad + \quad \mathbf{Z}_{uu}^{-1}\mathbf{Z}_{uv}\mathbf{S}_Z^{-1}\mathbf{Z}_{vu}\mathbf{Z}_{uu}^{-1}\mathbf{y}_u \quad - \quad \mathbf{Z}_{uu}^{-1}\mathbf{Z}_{uv}\mathbf{S}_Z^{-1}\mathbf{y}_v \\ \mathbf{b}_v = -\mathbf{S}_Z^{-1}\mathbf{Z}_{vu}\mathbf{Z}_{uu}^{-1}\mathbf{y}_u \quad + \quad \mathbf{S}_Z^{-1}\mathbf{y}_v \end{cases}.$$

Vector $\mathbf{b}_u$ can be expressed in terms of $\mathbf{b}_v$ ($\mathbf{b}_u = \mathbf{Z}_{uu}^{-1}\mathbf{y}_u - \mathbf{Z}_{uu}^{-1}\mathbf{Z}_{uv}\mathbf{b}_v$) and, since $\mathbf{b}_v$ is a vector of zeros, it comes that computing $\mathbf{b}_u$ shrinks to compute only $\mathbf{Z}_{uu}^{-1}\mathbf{y}_u$. In other words, the linear system $\mathbf{Zb} = \mathbf{y}$ is replaced by a linear system of lower size $\mathbf{Z}_{uu}\mathbf{b}_u = \mathbf{y}_u$, and solving it is valuable only if the number of operations required to solve it is lower than the number of operations to achieve the product in procedure *PROD*. We chose the less expensive procedure (*PROD* or *LS*) by estimation of the number of expected floating-point multiplications.

Experimental design for tests on real populations

In order to evaluate Algorithm B in comparison with regular inversion (Algorithm A), different $\mathbf{A}_{22}$ were computed on the basis of a real pedigree provided by the Luxembourg breeders society CONVIS. This pedigree includes dairy cows from Luxembourg with their ancestors tracing back up to 24 generations and contains 387 499 animals. Statistics of the pedigree data are Table 9.

Table 9 **Statistics of the population used** (dairy cows from Luxembourg)

Total number of animals	*387 499*
Number of cows	366 773
Number of bulls	20 726
Number of animals by birth year class:	
Before 1950	5441
From 1950 to 1974	24 577
From 1975 to 1999	229 016
From 2000 to 2012	128 465
Maximum number of generations of pedigree	39
Average number of generations[1] for animals in different birth year classes:	
Before 1950	3.28
From 1950 to 1974	6.49
From 1975 to 1999	19.03
From 2000 to 2012	25.11
Pedigree completeness: number of animals with (% of the pedigree):	
Both parents unknown:	70 167 (18.1%)
Dam known, sire unknown	69 721 (18.0%)
Sire known, dam unknown	17 141 (4.4%)
Both parents known	230 470 (59.5%)

[1] For a given animal, the number of generations is computed as the number of generations between this animal and its most distant ancestor.

Selected sub-populations of three sizes (4000, 8000 and 12 000 animals) were designed and are identified hereafter as the three size scenarios S4k, S8k and S12k. Animals of the selected sub-populations were randomly chosen from a pool of animals born after 1999 (128 465 animals) on the assumption that only recent animals could be of interest (those being genotyped or in production).

Because a pedigree with a lower number of extracted generations is expected to provide a sparser $\mathbf{T}^{-1}$, the impact of the number of extracted generations was also evaluated for each size scenario. This enabled us to assess how many extracted generations were required in the pedigree to compute a $\mathbf{A}_{22}^{-1}$ that is a sufficient approximation to the $\mathbf{A}_{22}^{-1}$ computed using all available ancestors in the pedigree, which will be referred to as the "real inverse". Extracting no animals other than *selected* animals refers to "generation 0": the population is only made of *selected* animals. When extracting one generation of ancestors ("generation 1"), *excluded* parents enter the population. When extracting two generations of ancestors ("generation 2"), *excluded* grandparents also enter the population, and so on. Details on the number of animals extracted and the percentage of extraction after each generation, considered as the ratio between the number of animals in the population and the maximum number of animals available in the pedigree, are outlined in Figure 5.

Deviations from the real inverse were measured by the following norm: $N = {tr\left((\mathbf{A}_{22}^{(g)} - \mathbf{A}_{22}^{(f)})'(\mathbf{A}_{22}^{(g)} - \mathbf{A}_{22}^{(f)})\right)} \Big/ {tr\left((\mathbf{A}_{22}^{(f)})'\mathbf{A}_{22}^{(f)}\right)}$, where $\mathbf{A}_{22}^{(g)}$ is the inverse of $\mathbf{A}_{22}$ computed using g extracted generations and $\mathbf{A}_{22}^{(f)}$ is the real inverse. This norm can be interpreted as the average difference between the value of any element of $\mathbf{A}_{22}^{(g)}$ and its corresponding value in $\mathbf{A}_{22}^{(f)}$. The two matrices are equal when N is equal to 0.

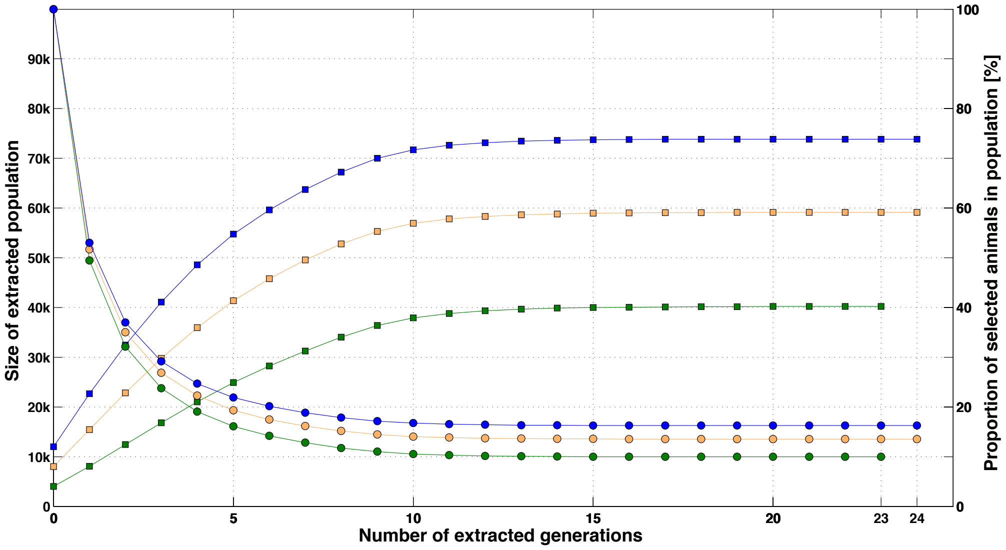

Figure 5 **Pedigree extraction facts.** Generation by generation extraction of the pedigree of the *selected* population for three size scenarios (green: S4k; orange: S8k; blue: S12k): number of extracted animals (■) and proportion of *selected* animals in the extracted population (●), expressed as a percentage. Extraction went up to 23 generations for scenario S4k and up to 24 generations for scenarios S8k and S12k.

Matrix $\mathbf{A}_{22}$ was computed in two steps. Inbreeding coefficients were first computed for each size scenario and number of extracted generations. The average inbreeding coefficient was never greater than 1.23% and the greatest inbreeding coefficient was 44.53%. Matrix $\mathbf{A}_{22}$ was then computed using the method of Colleau (2002).

Two test software programs

In order to evaluate potential gains in time when using Algorithm B instead of Algorithm A to invert $\mathbf{A}_{22}$, we developed two test programs in Fortran 95. The programs were neither optimized for speed, nor parallelized. Therefore, all comparisons have to be interpreted as relative figures.

The first program applies the recursive construction of the inverse, as outlined in Algorithm A (equations IV.2 and IV.3). Potential null entries in **y** are checked to avoid useless computations when performing product $\mathbf{Z}^{-1}\mathbf{y}$.

The second program restricts the same recursive construction of the inverse to non-zero elements by procedures *EMPTY*, *PROD* and *LS*. Potential null entries in **y** are also taken into account when performing the product $\mathbf{Z}^{u}\mathbf{y}$ (procedure *PROD*). The linear system $\mathbf{Z}_{uu}\mathbf{b}_{u} = \mathbf{y}_{u}$ (procedure *LS*) is solved by factorization and by backward and forward substitutions.

For both programs, computing time was recorded using Fortran intrinsic subroutine CPU_TIME. For the program that uses Algorithm B, computing time includes the time required to determine the sparsity pattern. All computations and file storage were performed using double precision (15 digits). Each job was repeated 20 times on an Intel® Xeon® 64-bit processor (RAM: 8 Gb, cache size: 6 Mb, clock speed: 3 GHz).

Results

Characterizing the sparsity pattern: a numerical example

The algorithm to characterize the sparsity pattern was applied to the example pedigree of Figure 4 and specified in Table 10 (including animal status). The algorithm starts by initializing a vector **x** equal to $\left[1,2,3,4,5,6,7,8,9,10,11,12\right]$. Then, we consecutively treat each animal depending on its status and the status of its parents.

Table 10 **Renumbered pedigree for the example of Figure 4.**

	Number	*Sire*	*Dam*	*Status*
A	1	-	-	1
B	2	-	-	1
C	3	1	2	2
D	4	-	-	1
E	5	3	4	1
F	6	-	-	2
G	7	6	5	2
H	8	-	4	1
I	9	8	-	2
J	10	1	-	2
K	11	6	-	2
L	12	-	5	2

Population status of the animal is given in a 4th column: 1 for *excluded*, 2 for *selected*.

Animal 1. Status 1 and unknown parents. Thus, $\mathbf{r}_{(1)} = [1]$, $\mathbf{c}_{(1)} = [-]$ and $\mathbf{x} = \mathbf{x}$.

Animal 2. Status 1 and unknown parents. Thus, $\mathbf{r}_{(2)} = [2]$, $\mathbf{c}_{(2)} = [-]$ and $\mathbf{x} = \mathbf{x}$.

Animal 3. Status 2 and known parents (1 and 2; both status 1). Thus, $\mathbf{c}_{(3)} = [3]$ and $\mathbf{r}_{(3)} = [1,2]$. The list of elements of $\mathbf{x}$ that match $\mathbf{r}_{(3)}$ is $[1,2]$. Then, $\mathbf{c}_{(3)} = [3,\mathbf{c}_{(1)},\mathbf{c}_{(2)}] = [3]$ and any element of $\mathbf{x}$ equal to 1 or 2 is replaced by 3, returning $\mathbf{x} = [3,3,3,4,5,6,7,8,9,10,11,12]$.

Animal 4. Status 1 and unknown parents. Thus, $\mathbf{r}_{(4)} = [4]$, $\mathbf{c}_{(4)} = [-]$ and $\mathbf{x} = \mathbf{x}$.

Animal 5. Status 1 and known parents (status 1 and 2). Thus, $\mathbf{c}_{(5)} = [3]$ and $\mathbf{r}_{(5)} = [5,\mathbf{r}_{(4)}] = [5,4]$. No list to set up because animal has status 1; $\mathbf{x} = \mathbf{x}$.

Animal 6. Status 2 and unknown parents. Thus, $\mathbf{r}_{(6)} = [-]$, $\mathbf{c}_{(6)} = [6]$ and $\mathbf{x} = \mathbf{x}$.

Animal 7. Status 2 and known parents (status 1 and 2). Thus, $\mathbf{c}_{(7)} = [7,6]$ and $\mathbf{r}_{(7)} = [\mathbf{r}_{(5)}] = [5,4]$. The list of elements of $\mathbf{x}$ that match $\mathbf{r}_{(7)}$ is $[5,4]$. Then, $\mathbf{c}_{(7)} = [7,6,\mathbf{c}_{(5)},\mathbf{c}_{(4)}] = [7,6,3]$ and any element of $\mathbf{x}$ equal to 5 or 4 is replaced by 7, returning $\mathbf{x} = [3,3,3,7,7,6,7,8,9,10,11,12]$.

Animal 8. Status 1 and one known parent (status 1). Thus, $\mathbf{r}_{(8)} = [8,\mathbf{r}_{(4)}] = [8,4]$, $\mathbf{c}_{(8)} = [-]$ and $\mathbf{x} = \mathbf{x}$.

Animal 9. Status 2 and one known parent (status 1). Thus, $\mathbf{c}_{(9)}=\left[9\right]$ and $\mathbf{r}_{(9)}=\left[\mathbf{r}_{(8)}\right]=\left[8,4\right]$. The list of elements of $\mathbf{x}$ that match $\mathbf{r}_{(9)}$ is $\left[8,7\right]$. Then, $\mathbf{c}_{(9)}=\left[9,\mathbf{c}_{(8)},\mathbf{c}_{(7)}\right]=\left[9,7,6,3\right]$ and any element of $\mathbf{x}$ equal to 8 or 7 is replaced by 9, returning $\mathbf{x}=\left[3,3,3,9,9,6,9,9,9,10,11,12\right]$.

Animal 10. Status 2 and one known parent (status 1). Thus, $\mathbf{c}_{(10)}=\left[10\right]$ and $\mathbf{r}_{(10)}=\left[\mathbf{r}_{(1)}\right]=\left[1\right]$. The list of elements of $\mathbf{x}$ that match $\mathbf{r}_{(10)}$ is $\left[3\right]$. Then, $\mathbf{c}_{(10)}=\left[10,\mathbf{c}_{(3)}\right]=\left[10,3\right]$ and any element of $\mathbf{x}$ equal to 3 is replaced by 10, returning $\mathbf{x}=\left[10,10,10,9,9,6,9,9,9,10,11,12\right]$.

Animal 11. Animal has status 2 and has one known parent (status 2). Thus, $\mathbf{c}_{(11)}=\left[11,6\right]$ and $\mathbf{r}_{(11)}=\left[-\right]$. No list to set up because $\mathbf{r}_{(11)}$ is empty; $\mathbf{x}=\mathbf{x}$.

Animal 12. Status 2 and one known parent (status 1). Thus, $\mathbf{c}_{(12)}=\left[12\right]$ and $\mathbf{r}_{(12)}=\left[\mathbf{r}_{(5)}\right]=\left[5,4\right]$. The list of elements of $\mathbf{x}$ that match $\mathbf{r}_{(12)}$ is $\left[9\right]$. Then, $\mathbf{c}_{(12)}=\left[12,\mathbf{c}_{(9)}\right]=\left[12,9,7,6,3\right]$ and any element of $\mathbf{x}$ equal to 9 is replaced by 12, returning $\mathbf{x}=\left[10,10,10,12,12,12,12,12,12,10,11,12\right]$.

Vectors $\mathbf{c}_{(i)}$ of the *selected* animals (3, 6, 7, 9, 10, 11 and 12) contain the non-zero elements of $\mathbf{T}^{-1}$ (Table 11) and these match with $\mathbf{T}^{-1}$ in Table 8.

Table 11 **Sparsity pattern of $\mathbf{T}^{-1}$ for the example of Figure 4**.

	C	***F***	***G***	***I***	***J***	***K***	***L***
C	X						
F		X					
G	X	X	X				
I	X	X	X	X			
J	X				X		
K		X				X	
L	X	X	X	X			X

X indicates non-zero entries.

Effect of accounting for sparsity on CPU time for inversion of $\mathbf{A}_{22}$

Algorithms A and B were both applied to the matrices created by different pedigree extractions of the three size scenarios. The elapsed CPU time results (averaged over 20 repetitions) are shown in Figure 6. Taking sparsity into account (Algorithm B) instead of using an inversion algorithm with cubic complexity (Algorithm A) reduced the elapsed CPU time for computing the inverse. For instance, the relative gains in computing speed of Algorithm B for the fully extracted pedigree were 1.67 faster for S4k, 1.75 faster for S8k, and 1.77 faster for S12k.

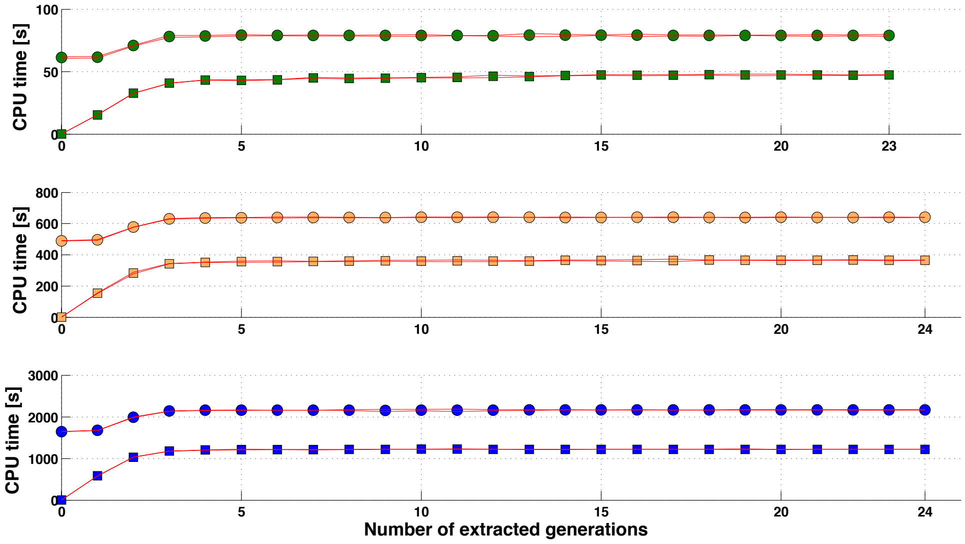

Figure 6 **CPU time required for inversion of $\mathbf{A}_{22}$ by two algorithms.** Elapsed CPU time required for inversion of $\mathbf{A}_{22}$ of three different sizes (green: 4000; orange: 8000; blue: 12000), computed using pedigrees with different numbers of extracted generations, by algorithms B (■) and A (▼). Red lines show upper and lower confidence intervals (99%; 20 repetitions).

Effect of the number of extracted generations on accuracy of the inverse of $\mathbf{A}_{22}$

For each size scenario, $\mathbf{A}_{22}$ was computed using different numbers of extracted generations and the inverses were compared (Figure 7) to $\mathbf{A}_{22}^{-1}$ computed using the fully extracted pedigree (after 23, 24 and 24 generations respectively for scenarios S4k, S8k and S12k) by computing the norm *N*. As shown in Figure 7, regardless of the size of the matrix, the norm stabilized after 14 generations to values less than 1E-13, which can be attributed to errors due to precision.

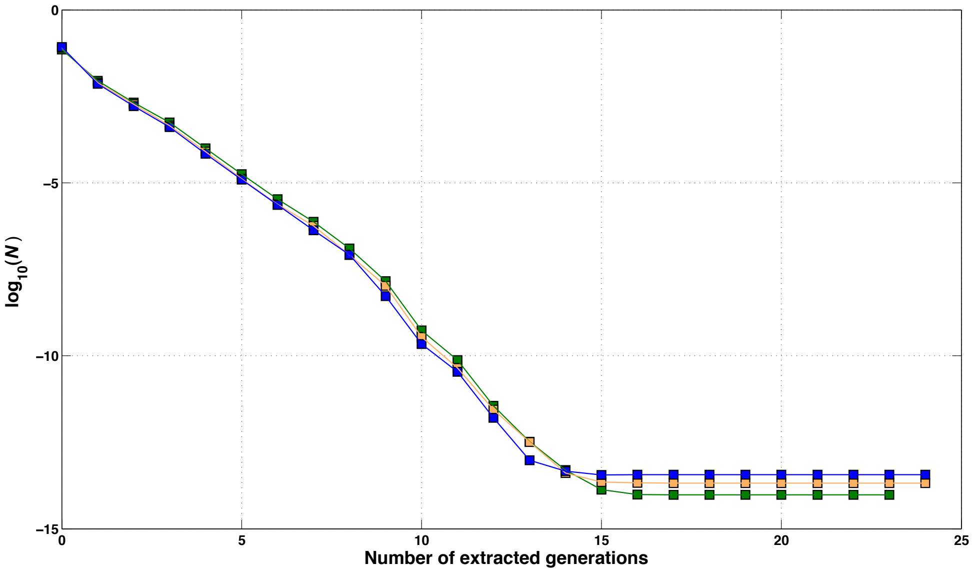

Figure 7 **Effect of the depth of the pedigree on $\mathbf{A}_{22}^{-1}$.** Differences, as base-10 logarithm of the norm *N*, between $\mathbf{A}_{22}^{-1}$ based on a pedigree with a limited number of extracted generations and $\mathbf{A}_{22}^{-1}$ based on a fully extracted pedigree, for three size scenarios (green: S4k; orange: S8k; blue: S12k).

Discussion

Computation time required by the algorithm to characterize the sparsity pattern

Figure 8 shows the elapsed CPU time (averaged over 20 repetitions) when running the proposed algorithm to determine the sparsity pattern of $\mathbf{T}^{-1}$ on populations with different numbers of *selected* animals (4000; 8000; 12 000) and that were extracted from several generations. The curves of the three size scenarios (S4k, S8k and S12k) presented a similar behaviour. When the population consists only of *selected* animals (generation 0), the elapsed time was less than 1 second (S4k: 0.03 s, S8k: 0.11 s and S12k: 0.29 s). For this case, only non-zero entries occur for *selected* sires or dams of *selected* animals, *a fortiori* present in the pedigree. Then, elapsed CPU time increased linearly up to the 15th extracted generation, although at a different rate for the different size scenarios. Beyond that point, adding ancestors did not affect the elapsed time. These results have to be related with pedigree extraction (Figure 5): does it make sense to spend more time for additional generations? Almost all available ancestors have entered the population after extracting 10 generations (between 95-99% of the number of animals in the last extraction round). However, elapsed CPU time continued to increase at the same rate from generations 10 to 15. For instance, in scenario S12k, adding ~3% of the final population cost an additional ~4 seconds (or ~22% of the total elapsed time). The usefulness of this small group of remote ancestors for inversion of $\mathbf{A}_{22}$ is discussed hereafter (sub-section "*Number of generations to extract*").

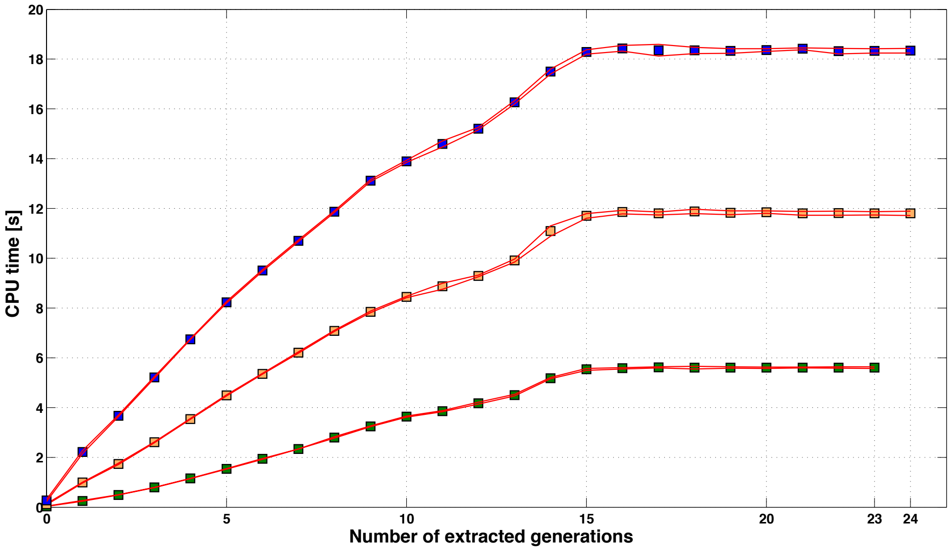

Figure 8 **CPU time required for determination of the sparsity pattern of $\mathbf{T}^{-1}$.** Elapsed CPU time required by the proposed algorithm for determination of the sparsity pattern of $\mathbf{T}^{-1}$, by number of extracted generations, for three size scenarios (green: S4k, orange: S8k and blue: S12k). Red lines show upper and lower confidence intervals (99%; 20 repetitions).

For the fully extracted population (after 23, 24 and 24 generations for scenarios S4k, S8k and S12k, respectively), there was a close-to-linear relationship between the size of the *selected* population and the elapsed CPU time (approximately 6 seconds for 4000 additional animals in the *selected* sub-population). The effective computational complexity of this algorithm is difficult to establish, however, because it mostly depends, first, on how the population was split (for instance, a *selected* sub-population that includes mainly a few lines or families would not contain that many *excluded* parents) and, secondly, on how the population is structured (depth of the pedigree, effective size of the population, average inbreeding). The embedded loop in the algorithm (step (4b) in the pseudo-code) is the main computational bottleneck and performs k iterations. In a population of n_0 animals, if k is related to the two factors mentioned above (i.e. splitting and structure of the population), then the computing time required by the algorithm would behave as $n_0 \cdot k$, where k would be a case-specific factor. This agrees with the observations in Figure 8.

Memory requirements of the algorithm to characterize the sparsity pattern

For a population of n_0 animals with *n selected* animals, vectors $\mathbf{c}_{(i)}$ and $\mathbf{y}_{(i)}$ have the greatest RAM requirements. In our implementation, vector $\mathbf{y}_{(i)}$ stores few elements (positions of *excluded* ancestors) for all animals (thus ~n_0 integers). For *selected* animals, vector $\mathbf{c}_{(i)}$ stores non-zero positions and includes approximately $n \cdot (n \cdot \bar{d})$ integers, where $\bar{d}$ is the average density of $\mathbf{T}^{-1}$ (number of non-zeros in the lower part of $\mathbf{T}^{-1}$ averaged by line). For *excluded* animals, $\mathbf{c}_{(i)}$ accounts for potential *selected* ancestors, therefore including approximately $(n_0 - n) \cdot \bar{a}$ integers, where $\bar{a}$ is the average number of *selected* ancestors per *excluded* animal. Memory would thus be allocated for approximately $n^2\bar{d} + (n_0 - n) \cdot \bar{a}$ integers. None of these integers may be declared as 3-byte integers when n_0 is lower than 2^{24} (i.e. when pedigree contains less than 16.77 millions of animals).

Use of the algorithm to characterize the sparsity pattern on greater populations

If additional animals are *selected*, then the proportion of *selected* animals in the population would likely increase. In fact these additional animals would either bring new *excluded* ancestors (case 1), share ancestors with already *selected* animals (case 2), or have no registered parents in the pedigree (case 3). The two last cases are expected to be more important as the number of *selected* animals increases. Therefore, matrix $\mathbf{T}^{-1}$ of such a population should get sparser. These expectations were confirmed by randomly picking animals from the pool of 128 465 animals born after 1999, simulating eight larger *selected* sub-populations of 16 000 up to 128 000 animals. Table 12 gives sizes and proportions of the *selected* sub-populations. Using a computer with higher memory resources (64 Gb of RAM), the sparsity pattern of these new situations was computed. Then, the degree of sparsity was assessed as the percentage of null entries in the lower triangular part of $\mathbf{T}^{-1}$ for these new situations, as well as for previous size scenarios. The results in Figure 9 show that the degree of sparsity remained the same for low percentages of *selected* animals in the population (lower than 20%), while the degree of sparsity increased linearly beyond approximately 20k animals in these specific cases. The average degree of sparsity by number of *selected* animals corresponded to the average number of *contributors* for a given animal in a given size situation. Figure 10 shows that the average number of *contributors* was linearly related to the number of *selected* animals up to ~80k *selected* animals, beyond which the average number of *contributors* was constant. We expected the average number of contributors to decrease as the number of *selected* animals increased. These new *selected* animals would then cover more of the relationships due to *excluded* animals. Note that the average number of contributors would be less than 2 if all animals were *selected* (i.e. $\mathbf{A}_{22} = \mathbf{A}$).

Table 12 **Populations extracted for different sets of *selected* animals**

Number of *selected* animals	**Size of the extracted population**	**Proportion of *selected* animals in the extracted population (%)**
4 000	40 196	9.95
8 000	59 120	13.53
12 000	73 864	16.25
16 000	87 237	18.34
32 000	127 809	25.04
48 000	159 259	30.14
64 000	183 750	34.83
80 000	204 637	39.09
96 000	222 546	43.14
112 000	238 130	47.03
128 000	252 147	50.76

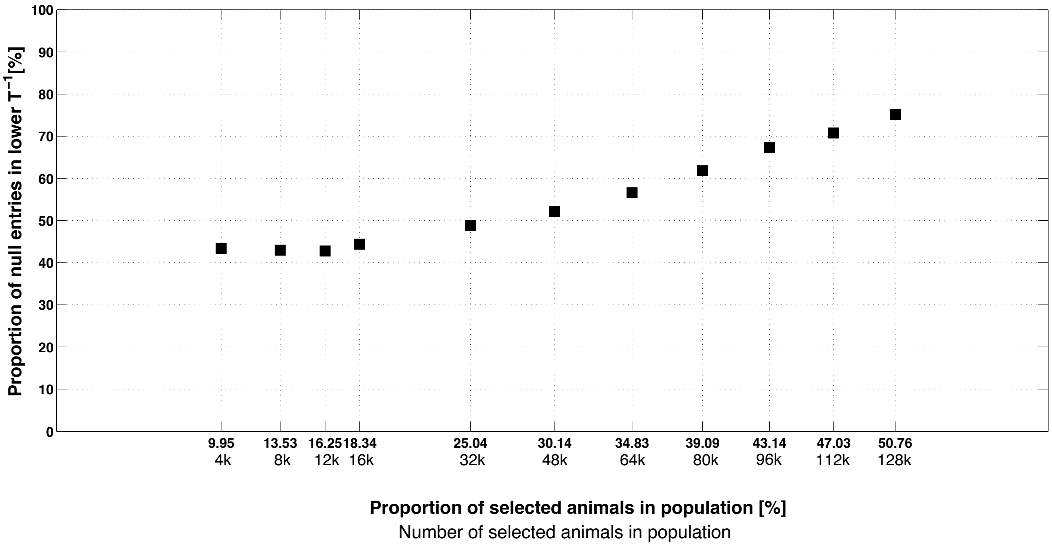

Figure 9 **Degree of sparsity of $\mathbf{T}^{-1}$.** Proportion of null entries in the lower triangular part of $\mathbf{T}^{-1}$ for different proportions (%) and numbers (thousands of animals) of *selected* animals in an extracted population.

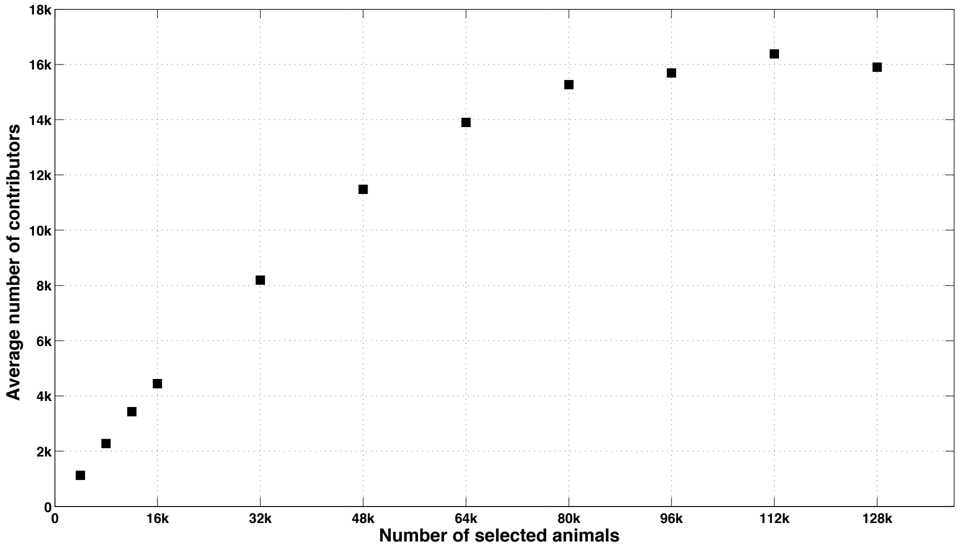

Figure 10 **Average number of *contributors*.** Average number of *contributors* by line of $\mathbf{T}^{-1}$, for different numbers of *selected* animals (in thousands of animals).

Computation time required by the algorithm for inversion of $\mathbf{A}_{22}$ using the sparsity pattern (Algorithm B)

When running Algorithm B, the procedure (*EMPTY*, *PROD* or *LS*) to compute vector **b** was chosen according to the estimated number of floating-point multiplications to be performed. A view of this choice along all (n-1) lines of $\mathbf{T}^{-1}$ is given in Figure 11 for each size scenario ($\mathbf{A}_{22}$ was always computed using a fully extracted pedigree). Due to prior reordering of the pedigree by generation, the first lines of $\mathbf{T}^{-1}$ correspond to founders (unrelated animals) and are thus empty. Procedure *LS* occurred less than procedures *EMPTY* and *PROD* but was evenly distributed among line numbers, particularly for scenario S12k.

Table 13 **Estimated computational complexity[1] of Algorithm B**

Procedure	Complexity for line *i*	Proportion	Complexity on *n* lines
EMPTY	1	p_E	$p_E \cdot O(n)$
LS	$O(k^3)+O(k^2)+O(k)$	p_L	$p_P \cdot \left[O(n\cdot k^3)+O(n\cdot k^2)+O(n\cdot k)\right]$
PROD	$O(k^2)+O(k\cdot i)$	p_P	$p_P \cdot \left[O(n\cdot k^2)+O(n^2\cdot k)\right]$
Total	$O(k^3)+O(k^2)$ $+O(k\cdot i)+O(k)$	1	$p_P\cdot O(n^2\cdot k)+p_L\cdot O(n\cdot k^3)+(p_L+p_P)\cdot O(n\cdot k^2)$ $+p_L\cdot O(n\cdot k)+p_E\cdot O(n)$

Matrix is of order *n* and average number of *contributors* is *k*;
[1] computational complexity is assessed as the expected number of floating-point multiplications to be performed.

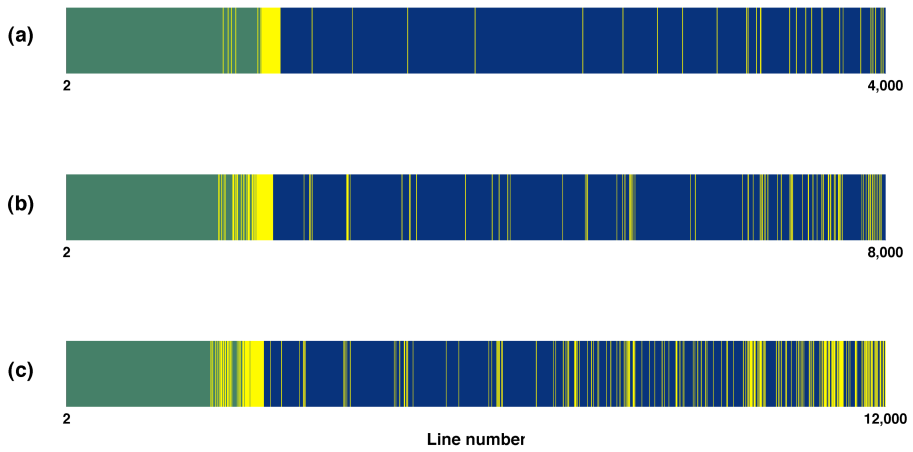

Figure 11 **Procedure choice when running algorithm B.** Procedure choice (green: *EMPTY*; yellow: *LS*; blue: *PROD*) when running algorithm B, along all lines of $\mathbf{T}^{-1}$, for inversion of matrix $\mathbf{A}_{22}$ with a fully extracted pedigree, for three size scenarios [(a): S4k; (b): S8k ; (c): S12k].

Considering Algorithm B led to estimation of the computational complexities based on the expected number of floating-point multiplications involved in the different tasks achieved by Algorithm B, as specified in Table 13. Total complexity is detailed for treatment of one line and for treatment of one full matrix of order *n* in Table 13, where treatment refers to all tasks to be performed, i.e. computing **b** and adding **bb**′ to the previous inverse. If *k* (average number of *contributors*) is considered as independent of *n*, the most complex term is $O(n^2 \cdot k)$, which is required when using the *PROD* procedure (proportion p_P of the total). The *PROD* procedure is used less frequently for greater matrices (see Figures 11 and 12 beyond 80k animals). Treating *k* as independent of *n* is also a more reasonable assumption for greater matrices (Figure 10), since *k* is undoubtedly related to *n* for smaller matrices. The total complexity for a matrix of order *n* becomes:

$$\begin{aligned}&(\bar{d})^3 p_L O(n^4) + \left[\bar{d}\, p_P + (\bar{d})^2 (p_L + p_P)\right] \cdot O(n^3)\\ &+\bar{d}\, p_L O(n^2) + p_E O(n)\end{aligned},$$

where $\bar{d}$ represents the average density of the matrix. The most complex term ($(\bar{d})^3 p_L O(n^4)$) is tempered by two very low coefficients: the proportion of times the *LS* procedure is used (p_L), which may be very low for small matrices (Figure 12), and the cube of the average density ($\bar{d}$), which was lower than 0.5 in our examples (Figure 9) for matrices of order beyond 32 000. Thus, Algorithm B seems more suitable for large matrices than for small matrices, regardless of whether there is dependence between *n* and *k* or not.

The issue of numerical stability was also addressed. When using procedure *PROD*, the result of the previous iteration was used in the current iteration through α and **b**. Accumulating errors could lead to instabilities and/or divergences. However, in *LS* procedure, the result of the previous iteration does not affect the **b** that is computed. Choosing the *LS* procedure at regular intervals among iterations using the *PROD* procedure (see Figure 11) stops the accumulation of errors that could have resulted from continuously choosing the *PROD* procedure. Therefore, interlacing choices for both procedures is a good way to prevent numerical instability. Independence between iterations also allows procedure *LS* to parallelized.

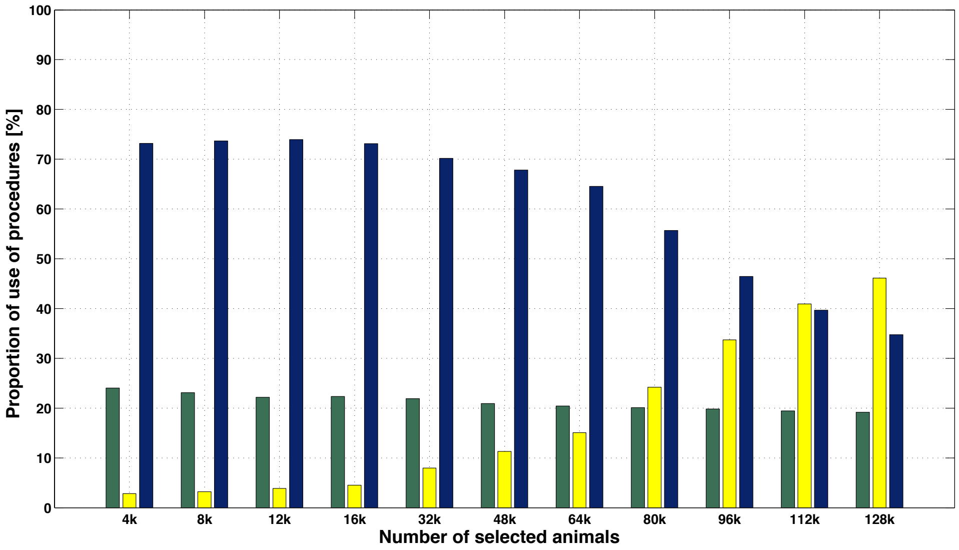

Figure 12 **Proportional use of different procedures in algorithm B.** Proportional (%) use of the three procedures in algorithm B (green: *EMPTY*; yellow: *LS*; blue: *PROD*), for different numbers of *selected* animals (in thousands of animals).

Memory requirements of the algorithm for inversion of $\mathbf{A}_{22}$ using sparsity pattern (Algorithm B)

Algorithm B requires allocation of more than twice the RAM than Algorithm A because it cannot store the results of the inversion in the input matrix. This is due to procedure *LS* working on different parts of $\mathbf{A}_{22}$. However, since elements that are required for *LS* are identified when determining the sparsity pattern, they could be stored separately in order to reduce the amount of RAM required. For that reason, sparsity patterns should be established prior to computation of $\mathbf{A}_{22}$ to determine which relationships are worth being computed.

Number of generations to extract

The depth of the pedigree to be used for instance in genetic evaluations, is still a question of debate, and often moderately deep pedigrees are used, especially when only recent data is analyzed.

Results in Figure 7 suggest that pedigree from a limited number of generations (5 to 10) is sufficient to compute $\mathbf{A}_{22}^{-1}$ with reasonable accuracy. The explanation is that distant ancestors do not greatly enhance a relationship. For instance, a common ancestor to animals *i* and *j* that enters the pedigree after *g* extracted generations and that is older than any selected animal, can only add up to 2^{-2g} to the value of the relationship between *i* and *j*. In generation *g, i* and *j* can have a maximum of 2^{g} common ancestors. Therefore, extracting an additional generation can increase the relationship between *i* and *j* by only up to $\delta = 2^{-g}$. Regardless of the number of animals added to the pedigree when extracting generation 10, the maximum change brought to any relationship reduces to less than 0.001, which would have a minor effect on the inverse scale, as confirmed by Figure 7.

However, computing time required for determination of the sparsity pattern increases linearly after 10 generations (Figure 8). Thus, limiting extraction of pedigree to 10 generations appears to be a good balance between taking into account relationships due to distant ancestors and computing time. Applying a similar study to pedigree extractions for routine genetic evaluations would be meaningful and may lead us to consider extracting a number of generations instead of a birth year limit, which is current common practice.

Practical use in a genomic background

For genomic evaluations, two specific situations where $\mathbf{A}_{22}^{-1}$ is needed may require the use of Algorithm B. First, as explained above and shown in equation (IV.3), the inverse of the matrix is computed recursively by adding a block specific to the current animal to the previous inverse. At each genomic evaluation, $\mathbf{A}_{22}^{-1}$ could therefore be stored in a file and reused at the next evaluation cycle. At each evaluation, the matrix would be enhanced by adding newly genotyped animals. However, this approach has some limits:

(1) Animals have to be listed by generation order and only animals younger than those already genotyped can be added because older animals may cause changes in the sparsity pattern. This could be easily implemented in a cattle breed such as Holstein, where only few animals are key ancestors of the breed.

(2) The resulting file may be large but this could be reduced by sparse storage approaches.

Meyer et al. (2013) recently applied a similar methodology for computation of the inverse of the genomic relationship matrix (**G**): their methodology also updates the previous inverse of **G**, necessitating its storage on disk from an evaluation to the next one.

Secondly, when using a pedigree of only one extracted generation, which contains genotyped animals and their ungenotyped parents, inversion of $\mathbf{A}_{22}$ is even faster (Figure 6) and the inverse seems to be a reasonable approximation of $\mathbf{A}_{22}^{-1}$ computed with a full extracted pedigree (see Figure 7 and discussion here above). Such a fair approximation of $\mathbf{A}_{22}^{-1}$ may be useful as a preconditioner to solve $\mathbf{A}_{22}\mathbf{x} = \mathbf{v}$, for instance, as required in the iterative solution of MME of single-step genomic BLUP (best linear unbiased prediction) proposed by Legarra and Ducrocq (2012).

Current limits

The algorithm to determine the sparsity pattern of the inverse triangular factor of $\mathbf{A}_{22}$ is obviously useful only in inversion algorithms that use the inverse triangular factor. For other inversion algorithms, the algorithm to determine the sparsity pattern should not be useful.

Inversion algorithms that use the inverse triangular factor are useful in certain cases (e.g., for updating an inverted matrix or for obtaining quick approximations), but they would be less efficient, in terms of computing time, for the single purpose of inversion. The time required by Algorithms A and B was compared with the time required by subroutine "dkmxhf.f90" (K. Meyer, University of New England, Australia), which is a regular and efficient inversion algorithm. For inversion of the three different orders of $\mathbf{A}_{22}$ (4000, 8000 and 12 000), computing times of dkmxhf.f90 were lower than computing times obtained with Algorithm A and similar to those obtained with Algorithm B (accounting for sparsity). For small numbers of extracted generations, computing times were slightly lower for Algorithm B than dkmxhf.f90, but were greater when greater numbers of generations were extracted. However, the computing speed of Algorithm B can benefit from several optimizations (e.g., parallelization of the *LS* procedure and use of specific libraries for matrix products).

For computational ease, a small population (less than 1 million animals) was used in this study. Gains in computing time have to be tested for other sizes of population. This study was also restricted to only one population by size scenario and used repetitions (20) of the algorithm on the same data. Use of a Holstein population may also be criticized because although the average computed inbreeding was never greater than 1.23%, such a population has few key ancestors. Having the key ancestors in the *selected* sub-population might avoid density, because they would be *contributors* of many other *selected* animals.

Conclusions

The determination of the sparsity pattern of $\mathbf{T}^{-1}$ using pedigree information is a prior step that allows gains in computing time for inversion based on the use of $\mathbf{T}^{-1}$. This allowed the computing time for inversion of matrices of three different sizes (4000, 8000 and 12 000 *selected* animals) to be reduced by a factor 1.73 on average. Gains in computing time are expected to be higher if the number of *selected* animals exceeds 80 000. Memory requirements for inversion of such a matrix would increase and the algorithm would become numerically more stable, since the *LS* procedure would become more important than the *PROD* procedure. Moreover, computation of the inverse by a recursive method may be very helpful in the case of genomic prediction, where a new batch of younger *selected* animals at each upcoming evaluation must be added to the previous inverse matrix already computed.

The results on the number of pedigree generations required for the *selected* animals suggest that no more than 14 generations should be extracted. If the working precision is less than 15 digits, this can even be reduced. A good balance between computing time for determination of the sparsity pattern and accuracy may be achieved with 10 extracted generations.

Appendix: Inversion of the numerator relationship matrix using the inverse triangular factor

The numerator relationship matrix (**A**) can be factorized as

$$\mathbf{A} = \mathbf{TDT}'. \qquad \text{(IV.A.1)}$$

Henderson (1976) proposed a recursion rule to compute the triangular factor **T**:

$$\mathbf{T}_{(i)} = \begin{bmatrix} \mathbf{T}_{(i-1)} & \mathbf{0} \\ \mathbf{b}'_{(i)}\mathbf{T}_{(i-1} & 1 \end{bmatrix} \qquad \text{(IV.A.2)}$$

In equation (IV.A.2), $\mathbf{T}_{(i-1)}$ and $\mathbf{T}_{(i)}$ are two matrices of respective sizes (*i*-1) and *i*. They refer **T** computed after, respectively, (*i*-1) and *i* recursions. Vector $\mathbf{b}_{(i)}$ is a vector of parental contributions: it summarizes the linear dependency between parents and offspring. This vector is null except on positions corresponding to parents of *i* where it is equal to 0.5. Henderson (1976) also showed that the inverse triangular factor ($\mathbf{T}^{-1}$) only contains three different values: 0, 1and -0.5, since it is obtained by triangular matrix inversion (equation IV.A.3). The elements of the diagonal are equal to 1 and the lower off-diagonal elements are equal to the vector $-\mathbf{b}'_{(i)}$ corresponding to the i^{th} animal; they contain thus only 0 and -0.5 elements.

$$\mathbf{T}^{-1}_{(i)} = \begin{bmatrix} \mathbf{T}^{-1}_{(i-1)} & \mathbf{0} \\ -\mathbf{b}'_{(i)} & 1 \end{bmatrix} \qquad \text{(IV.A.3)}$$

Besides **T**, the diagonal matrix **D** is computed one element at a time according to Henderson (1976) and Quaas (1976). At the i^{th} recursion $\mathbf{D}_{(i)}$ has the form:

$$\mathbf{D}_{(i)} = \begin{bmatrix} \mathbf{D}_{(i-1)} & \mathbf{0} \\ \mathbf{0}' & d_{ii} \end{bmatrix} \qquad \text{(IV.A.4)}$$

Replacing equations (IV.A.2) and (IV.A.4) in (IV.A.1) shows that the recursion rule for computation of **T** is actually identical to that of the tabular method (equation IV.A.5.3; Emik and Terril, 1949; Henderson, 1976), since it computes the last below-diagonal row in $\mathbf{A}_{(i)}$ as a linear combination of rows in $\mathbf{A}_{(i-1)}$.

$$\mathbf{A}_{(i)} = \mathbf{T}_{(i)}\mathbf{D}_{(i)}\mathbf{T}_{(i)}{}' \qquad \text{(IV.A.5.1)}$$

$$= \begin{bmatrix} \mathbf{T}_{(i-1)}\mathbf{D}_{(i-1)}\mathbf{T}'_{(i-1)} & \mathbf{T}_{(i-1)}\mathbf{D}_{(i-1)}\mathbf{T}'_{(i-1)}\mathbf{b}_{(i)} \\ \mathbf{b}'_{(i)}\mathbf{T}_{(i-1)}\mathbf{D}_{(i-1)}\mathbf{T}'_{(i-1)} & \mathbf{b}'_{(i)}\mathbf{T}_{(i-1)}\mathbf{D}_{(i-1)}\mathbf{T}'_{(i-1)}\mathbf{b}_{(i)} + d_{ii} \end{bmatrix} \qquad \text{(IV.A.5.2)}$$

$$= \begin{bmatrix} \mathbf{A}_{(i-1)} & \mathbf{A}_{(i-1)}\mathbf{b}_{(i)} \\ \mathbf{b}'_{(i)}\mathbf{A}_{(i-1)} & \mathbf{b}'_{(i)}\mathbf{A}_{(i-1)}\mathbf{b}_{(i)} + d_{ii} \end{bmatrix} \qquad \text{(IV.A.5.3)}$$

Replacing $\mathbf{b}'_{(i)}\mathbf{A}_{(i-1)}\mathbf{b}_{(i)} + d_{ii}$ in equation (IV.A.5.3) by a_{ii} (the equivalence can be easily shown) expresses the tabular method as in van Arendonk et al. (1994):

$$\mathbf{A}_{(i)} = \begin{bmatrix} \mathbf{A}_{(i-1)} & \mathbf{A}_{(i-1)}\mathbf{b}_{(i)} \\ \mathbf{b}'_{(i)}\mathbf{A}_{(i-1)} & a_{ii} \end{bmatrix} \qquad \text{(IV.A.6)}$$

Applying the partitioned matrix theory to equation (IV.A.6), van Arendonk et al. (1994) structured $\mathbf{A}^{-1}$ as a sum of n updates of a null matrix (recursion rule in equation IV.A.7) involving multiplication of a sparse vector ($-\mathbf{b}_{(i)}$) by itself.

$$\mathbf{A}_{(i)}^{-1} = \begin{bmatrix} \mathbf{A}_{(i-1)}^{-1} & \mathbf{0} \\ \mathbf{0}' & 0 \end{bmatrix} + \frac{1}{d_{ii}} \begin{bmatrix} -\mathbf{b}_{(i)} \\ 1 \end{bmatrix} \begin{bmatrix} -\mathbf{b}'_{(i)} & 1 \end{bmatrix} \qquad \text{(IV.A.7)}$$

The sparse vector $-\mathbf{b}_{(i)}$ is actually the transpose of the i-th below-diagonal row of $\mathbf{T}^{-1}$ (see equation IV.A.3). Such a construction of $\mathbf{A}^{-1}$ requires thus to know the following:

(1) the positions and values of non-zero elements in $\mathbf{b}_{(i)}$, i.e. the structure of $\mathbf{T}^{-1}$;

(2) some elements of the original matrix, to compute d_{ii} as $a_{ii} - \mathbf{b}'_{(i)}\mathbf{A}_{(i-1)}\mathbf{b}_{(i)}$.

After meeting these requirements (determination of the structure of the inverse triangular factor and computation of some elements of the original matrix), the same framework was extended to the inversion of other relationship matrices used in animal breeding: e.g. gametic relationship matrix (Schaeffer et al., 1989), dominance (Hoeschele and VanRaden, 1991) and epistasis (VanRaden and Hoeschele, 1991) effects or covariance matrix of marked QTL effects (Fernando and Grossman, 1989).

Acknowledgements

Luxembourg breeders association CONVIS is gratefully acknowledged for providing the pedigree data used in this study. PF acknowledges the following peoples and institution: Fonds National de la Recherche Luxembourg (FNR) for funding through AFR grant, I Misztal for hosting at Animal and Dairy Sciences Department of University of Georgia and for providing computing facilities, G Gorjanc for hosting in Animal Science Department of University of Ljubljana, J Vandenplas for helpful discussions, C Charles and L Lecharlier for mathematical advising, C Bastin, JFDumasy and D. TeCroney for editing help, and several members of the Numerical Genetics, Genomics and Modeling group of Gembloux Agro-Bio Tech for testing the algorithm. Comments and edits by two anonymous reviewers and by editors are gratefully acknowledged. The authors acknowledge the financial support (project D31-1274/S1 and D31-1274/S2 "*DairySNP*") of the Ministry of Agriculture of the Walloon Region of Belgium.

References

Christensen O.F. and Lund M.S., 2010. Genomic prediction when some animals are not genotyped. *Genet. Sel. Evol.*, 42, 2.

Colleau J.-J., 2002. An indirect approach to the extensive calculation of relationship coefficients. *Genet. Sel. Evol.*, 34, 409–421.

Emik L.O. and Terrill C.E., 1949. Systematic Procedures for Calculating Inbreeding Coefficients. *J. Hered.*, 40, 51–55.

Fernando R.L. and Grossman M., 1989. Marker assisted selection using best linear unbiased prediction. *Genet. Sel. Evol.*, 21, 1–11.

Gengler N., Nieuwhof G.J., Konstantinov K.V. and Goddard M.E., 2012. Alternative single-step type genomic prediction equations. In *Book of Abstracts of the 63rd Annual Meeting of the European Association of Animal Production: 27-31 August 2012; Bratislava*. Wageningen: Wageningen Academic Publishers, p. 131.

George A. and Liu J.W., 1980. An Optimal Algorithm for Symbolic Factorization of Symmetric Matrices. *SIAM J. Comput.*, 9, 583–593.

Gilbert J.R., 1994. Predicting structure in sparse matrix computations. *SIAM J. Matrix Anal. Appl.*, 15, 62–79.

Henderson C.R., 1973. Sire evaluation and genetic trends. In: *Proc. Anim. Breed. Genet. Symp. Honor Dr Jay Lush*., Am. Soc. Anim. Sci. and Am. Dairy Sci. Assoc., Poultry Sci. Assoc., Champaign, IL, 10-41.

Henderson C.R., 1976. A Simple Method for Computing the Inverse of a Numerator Relationship Matrix Used in Prediction of Breeding Values. *Biometrics*, 32, 69–83.

Hoeschele I. and VanRaden P.M., 1991. Rapid Inversion of Dominance Relationship Matrices for Noninbred Populations by Including Sire by Dam Subclass Effects. *J. Dairy Sci.*, 74, 557–569.

Legarra A. and Ducrocq V., 2012. Computational strategies for national integration of phenotypic, genomic, and pedigree data in a single-step best linear unbiased prediction. *J. Dairy Sci.*, 95, 4629–4645.

Meyer K., Tier B. and Graser H.-U., 2013. Technical note: Updating the inverse of the genomic relationship matrix. *J. Anim. Sci.*

Misztal I., Legarra A. and Aguilar I., 2009. Computing procedures for genetic evaluation including phenotypic, full pedigree, and genomic information. *J. Dairy Sci.*, 92, 4648–4655.

Quaas R.L., 1976. Computing the Diagonal Elements and Inverse of a Large Numerator Relationship Matrix. *Biometrics*, 32, 949–953.

Schaeffer L.R., Kennedy B.W. and Gibson J.P., 1989. The Inverse of the Gametic Relationship Matrix. *J. Dairy Sci.*, 72, 1266–1272.

van Arendonk J.A.M., Tier B. and Kinghorn B.P., 1994. Use of multiple genetic markers in prediction of breeding values. *Genetics*, 137, 319–329.

Vandenplas J. and Gengler N., 2012. Comparison and improvements of different Bayesian procedures to integrate external information into genetic evaluations. *J. Dairy Sci.*, 95, 1513–1526.

VanRaden P.M. and Hoeschele I., 1991. Rapid Inversion of Additive by Additive Relationship Matrices by Including Sire-Dam Combination Effects. *J. Dairy Sci.*, 74, 570–579.

VanRaden P.M., 2008. Efficient Methods to Compute Genomic Predictions. *J. Dairy Sci.*, 91, 4414–4423.

Wright S., 1922. Coefficients of Inbreeding and Relationship. *Am. Nat.*, 56, 330–338.

BRIEF SUMMARY OF CHAPTER IV

- An algorithm was developed to set up the sparsity pattern of the inverse Cholesky factor of $\mathbf{A}_{22}$ (i.e. to define, for each genotyped animal, which other animals were contributors of that animal). This pre-processing step, is applied on the pedigree before inversion of $\mathbf{A}_{22}$. It does not require much time because it does not involve floating-point operations but a pedigree search.
- Used in the frame of the Sherman-Morrison algorithm for inversion of matrices (successive update of a zeroed matrix), knowing the sparsity pattern allows avoiding useless computations. Therefore, computing time is reduced when using the sparsity pattern instead of not using it.
- A fair approximation of $\mathbf{A}_{22}^{-1}$ can be obtained by the following strategy: (1) computing $\mathbf{A}_{22}$ after extraction of few (1, 2 or 3) generations of non-genotyped ancestors of genotyped animals, (2) computing the sparsity pattern of the inverse of this $\mathbf{A}_{22}$, and (3) using the sparsity pattern when inverting $\mathbf{A}_{22}$ with the Sherman-Morrison algorithm.

RELATED PUBLICATIONS

- P.Faux and N. Gengler. 2013. Strategies for inversion of the additive relationship matrix among genotyped animals. In: *Book of abstracts of the 64th Annual Meeting of the European Association for Animal Production*. **Oral presentation** in Nantes (France).

Chapter V

THE RESTRICTED SPARSITY PATTERN ALGORITHM

In genotyped populations, it comes out that the inverse of $\mathbf{A}_{22}$ is frequently dense. Therefore, setting up its sparsity pattern and using it for inversion of $\mathbf{A}_{22}$ is not helpful because useless computations only concern few elements of the inverse. However, we pointed out that most of the contributions were small. Moreover, the absolute value of a contribution tends to decreases as the number of pedigree branches to search from the animal up to its contributor increases. Therefore, we propose to restrict the pedigree search for establishing the sparsity pattern to a limited number of branches. The so-approximated sparsity pattern is then used in approximation of the inverse of $\mathbf{A}_{22}$. In order to assess the applied interest of these approximations, tests are run using the approximated inverses in a traditional mixed model.

[FROM: P. Faux and N. Gengler. (submitted). A method to approximate the inverse of a part of the additive relationship matrix. *Journal of Animal Breeding and Genetics*]

Abstract

Single-step genomic predictions need the inverse of the part of the additive relationship matrix between genotyped animals ($\mathbf{A}_{22}$). Gains in computing time are feasible with an algorithm that sets up the sparsity pattern of $\mathbf{A}_{22}^{-1}$ (SP algorithm) using pedigree searches, when $\mathbf{A}_{22}^{-1}$ is close to sparse. The objective of this study is to present a modification of the SP algorithm (RSP algorithm) and to assess its use in approximating $\mathbf{A}_{22}^{-1}$ when the actual $\mathbf{A}_{22}^{-1}$ is dense. The RSP algorithm sets up a restricted sparsity pattern of $\mathbf{A}_{22}^{-1}$ by limiting the pedigree search to a maximum number of searched branches. We have tested its use on 4 different simulated genotyped populations, from 10,000 to 75,000 genotyped animals. Accuracy of approximation is tested by replacing the actual $\mathbf{A}_{22}^{-1}$ by its approximation in an equivalent mixed model including only genotyped animals. Results show that limiting the pedigree search to 4 branches is enough to provide accurate approximations of $\mathbf{A}_{22}^{-1}$, which contain about 80% of zeros. Computing approximations is not expensive but may require a great amount of memory (at maximum, ~81 minutes and ~55 Gb of RAM for 75,000 genotyped animals using parallel processing on 4 threads).

Keywords: matrix computations, ssGBLUP.

Introduction

In a unified approach (single-step genomic BLUP - ssGBLUP - , Misztal et al., 2009; Christensen and Lund, 2010), all the three information sources (pedigrees, phenotypes and genotypes) are used to compute genomically-enhanced breeding values (GEBV). The originality of this method lies on the use of a modified additive relationship matrix **H** (Legarra et al., 2009), accounting for both pedigree-based and genomic relationships, instead of the additive relationship matrix **A**. The ssGBLUP requires inversion of two symmetric matrices whose order is equal to the number of genotyped animals: **G**, genomic relationship matrix and $\mathbf{A}_{22}$, part of the additive relationship matrix gathering relationships between genotyped animals.

As for any symmetric positive-definite matrix, their inversion has a cubical complexity and is performed so far by direct inversion algorithms (e.g. as described in Aguilar et al., 2011). The case of **G** has been investigated by different studies performing computation (Meyer et al., 2013) or approximation (Misztal et al., 2014) of its inverse at lower computational costs. For the case of $\mathbf{A}_{22}$, we recently proposed a heuristic algorithm ("SP algorithm" for Sparsity Pattern algorithm) that sets up the sparsity pattern of its inverse using pedigree information (Faux and Gengler, 2013). It showed that $\mathbf{A}_{22}^{-1}$ might be sparse in some situations, depending on the structure of the pedigree of genotyped animals. In such a case, computing time savings are possible because computations are restricted to non-zero elements.

However, this method is useless when $\mathbf{A}_{22}^{-1}$ is dense or close to dense. In this study, we present a method to approximate $\mathbf{A}_{22}^{-1}$ by zeroing elements of the inverse Cholesky factor of $\mathbf{A}_{22}$. Our aim is to assess its efficiency in terms of approximation accuracy and computer resources (time and memory) consumption for different orders of $\mathbf{A}_{22}$.

Material and Methods

Simulation of realistic test populations

In the frame of applied researches on implementation of genomic evaluation for dairy cattle in the Walloon Region of Belgium, 2,427 genotyped animals (1,855 males and 574 females) were available. Because this number is limited in regard to our main objective that is to test approximations for huge matrix orders, we propose to simulate test populations that mimic the structure of this real population (denoted hereafter as "P2").

Matrix $\mathbf{A}_{22}^{-1}$ of P2 contains ~20% of elements equal to 0. This is due to the fact that genotyped animals in P2 are mainly born in same years (2000 to 2011) and are therefore highly likely to share the same non-genotyped ancestors.

In addition to genotyped animals in P2, a certain number of animals are randomly picked in the pedigree used for official genetic evaluations of dairy cattle in Walloon Region of Belgium. Animals are chosen in year of birth classes from 2000 to 2011 so that each class contains a proportionate number to its number in P2, in order to have a final number of genotyped animals equal to 10,000, 25,000, 50,000 or 75,000.

Eventually, a maximum of 6 generations are extracted for all genotyped animals, giving the 4 populations (P10, P25, P50 and P75) detailed in Table 14.

Table 14 **Details on populations used in the study**: type (real or simulated), sizes and sparsity degrees.

Population	*Type*	*Number of genotyped animals*	*Total number of animals*	*Sparsity degree*[1]
P2	Real	2,427	17,677	20.31
P10	Simulated	10,000	74,529	9.46
P25	Simulated	25,000	148,446	8.91
P50	Simulated	50,000	243,005	10.62
P75	Simulated	75,000	322,634	10.44

[1]Sparsity degree is expressed as the percentage of elements equal to zeros in the corresponding $\mathbf{A}_{22}^{-1}$.

Sparsity in the inverse of $\mathbf{A}_{22}$ and SP algorithm

For any genotyped animal *i*, its relationships with other genotyped animals ($\mathbf{A}_{22;\,1:i-1,i}$) can be computed as a linear combination $\mathbf{b}_i$ of the relationships between animals preceding him ($\mathbf{A}_{22;\,1:i-1,1:i-1}$):

$$\mathbf{A}_{22;\,1:i-1,i} = \mathbf{A}_{22;\,1:i-1,1:i-1} \cdot \mathbf{b}_i \qquad \text{(V.1)}$$

We name the *j*-th element ($\forall j, j = 1:i-1$) of $\mathbf{b}_i$ "contribution" of the *j*-th animal to the *i*-th animal and we call a "contributor" of the *i*-th animal any animal *j* whose contribution is not zero. Vector $\mathbf{b}_i$ is related to the root-free Cholesky factorization of $\mathbf{A}_{22}$ ($\mathbf{A}_{22} = \mathbf{TDT}'$) as $\mathbf{b}_i' = -\mathbf{T}_{i,1:i-1}^{-1}$.

Applying simple searching rules to the pedigree of each genotyped animal, the SP algorithm finds out which genotyped animals are contributors of a given genotyped animal. Defining a "branch" as a parent-offspring connection, the rules are the following, for the last animal in an age-ordered pedigree:

(1) explore ascending branches:

   a. if an ancestor is genotyped, then add it to the list of contributors and stop exploration in this branch;

   b. if an ancestor is not genotyped, then add it in a temporary list and keep exploring in this branch until genotyped ancestors or founders are found;

(2) explore descending branches of each non-genotyped ancestor in the temporary list and add any of their genotyped progenies in the list of contributors;

(3) apply the same rules (1) and (2) to each genotyped progeny found through rule(2) until no more animals are in the temporary list of non-genotyped ancestor.

For other animals than the last one in pedigree, animals younger than him must simply be ignored. This algorithm has been implemented in a sequential way in Faux and Gengler (2013). As an example, it can be applied to the last animal (animal 12) in the genealogical tree in Figure 13.

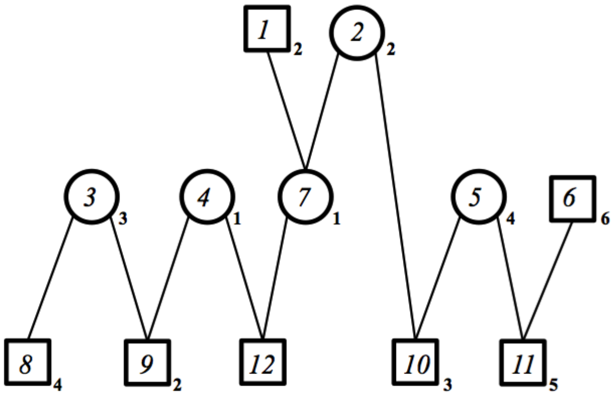

Figure 13 **Example genealogical tree of 12 animals**. Genotyped and non-genotyped animals are respectively tagged with squares and circles. Indexes show the number of branches to search starting from animal 12 in order to find its contributors.

In this figure, the animal identifier gives the age order (1 is the oldest animal and 12 is the youngest) and square tags identify genotyped animals.

(1) Apply rule (1) to animal 12: animal 1 is a genotyped ancestor, thus a contributor of animal 12, and animals 4, 7 and 2 are non-genotyped ancestors, thus left in a temporary list.

(2) Apply rule (2) to animals 4, 7 and 2: 2 genotyped progenies are found (animals 9 and 10). Thus, they join animal 1 in the list of contributors.

(3) Following rule (3), apply rules (1) and (2) to animals 9 and 10. Two other genotyped progenies (animals 8 and 11) also enter the list of contributors.

(4) Following rule (3), apply rules (1) and (2) to animals 8 and 11. A genotyped ancestor (animal 6) enters the list of contributors and the search is over as we have searched all available branches.

In this example, all genotyped animals are contributors of the last animal in pedigree. Therefore, vector of contributions $\mathbf{b}_i$ is fully dense (last row in $\mathbf{T}^{-1}$, see Table 15). As $\mathbf{A}_{22}^{-1} = \left(\mathbf{T}^{-1}\right)' \mathbf{D}^{-1}\mathbf{T}^{-1}$, the sparsity pattern of $\mathbf{A}_{22}^{-1}$ can be obtained from the sparsity pattern of $\mathbf{T}^{-1}$: all cross-positions of non-zero entries in a row of $\mathbf{T}^{-1}$ are non-zero entries in $\mathbf{A}_{22}^{-1}$. Therefore, in our example, $\mathbf{A}_{22}^{-1}$ is fully dense because $\mathbf{b}_i$ is too (Table 15).

Table 15 **Actual $\mathbf{T}^{-1}$ (below diagonal) and $\mathbf{A}_{22}^{-1}$ (diagonal and above diagonal)** for the example in Figure 13.

	1	***6***	***8***	***9***	***10***	***11***	***12***
1	1.07	0.01	-0.02	0.08	0.04	-0.01	-0.29
6		1.36	-0.00	0.01	0.19	-0.73	-0.03
8			1.07	-0.29	-0.01	0.00	0.08
9			-0.25	1.15	0.04	-0.01	-0.31
10					1.11	-0.37	-0.16
11		-0.5			-0.25	1.46	0.05
12	-0.25	-0.02	0.07	-0.27	-0.14	0.05	1.17

Empty cells denote zeros. No null entries out of 49 for $\mathbf{A}_{22}^{-1}$ (dense).

The SP algorithm only determines the non-zero elements in $\mathbf{b}_i$ not their values, which are solutions of equation (V.1). To restrict their computation only to contributors, a possibility is to partition equation (V.1) between non-zero (subscript u_i) and zero (subscript v_i) entries in $\mathbf{b}_i$:

$$\begin{bmatrix} \mathbf{A}_{u_i,i} \\ \mathbf{A}_{v_i,i} \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{u_i,u_i} & \mathbf{A}_{u_i,v_i} \\ \mathbf{A}_{v_i,u_i} & \mathbf{A}_{v_i,v_i} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{b}_{u_i} \\ \mathbf{b}_{v_i} \end{bmatrix} \qquad \text{(V.2)}$$

and to compute $\mathbf{b}_{u_i}$ as $\mathbf{A}_{u_i,u_i}^{-1}\mathbf{A}_{u_i,i}$. Once contributions are computed, computation of $\mathbf{A}_{22}^{-1}$ can be successively achieved, from the first animal in pedigree to the last one, by adding the following to proper positions of a zeroed matrix:

$$\alpha_i^{-1} \cdot \begin{bmatrix} -\mathbf{b}_{u_i} \\ 1 \end{bmatrix} \cdot \begin{bmatrix} -\mathbf{b}'_{u_i} & 1 \end{bmatrix}$$

where $\alpha_i = \mathbf{A}_{i,i} - \mathbf{A}_{i,u_i}\mathbf{b}_{u_i}$. When there are few non-zero contributions, then $\mathbf{A}_{22}^{-1}$ tends to be sparse and computing time gains are possible because inversion involves few operations.

Conversely, computing time gains are null when $\mathbf{A}_{22}^{-1}$ is dense or close to dense. However, it was reported that a majority of non-zero contributions were close to 0 (Faux et al., 2012). As an example, Figure 14 shows a distribution of the absolute value of non-zero contributions, in a real case (population P2), among absolute value classes. Each black bar gives the percentage of non-zero contributions whose absolute values, on a logarithmic scale, is in the corresponding class. In addition, each grey bar gives the cumulative percentage of non-zero contributions pertaining to lower classes. About 86% of contributions pertain to classes lower than class -3, meaning that 86% of non-zero contributions are lower than 10^{-3}.

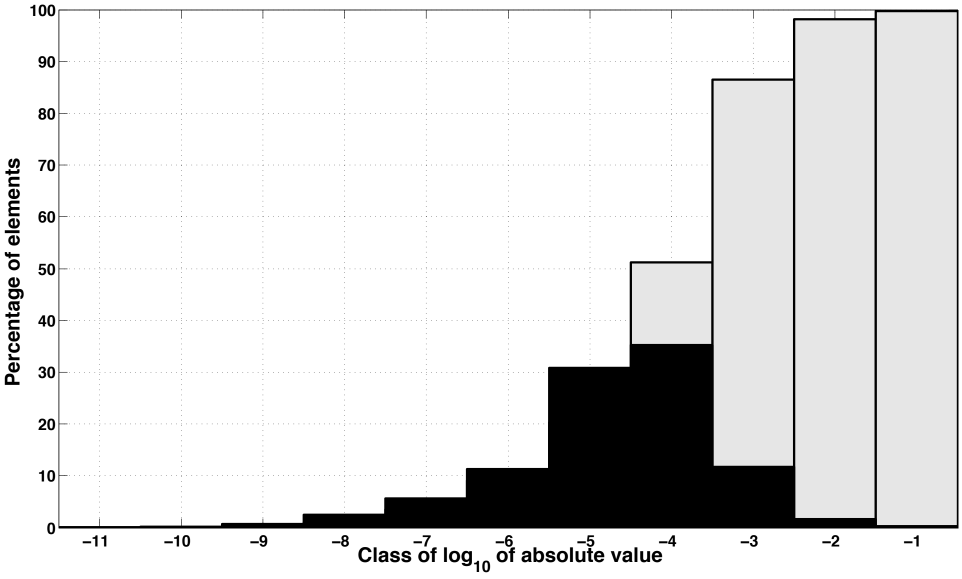

Figure 14 **Distribution of absolute value of 2,249,997 contributions of a real genotyped population (P2)** among logarithmic classes (black bars) and percentage of contributions lower, in absolute value, than a certain logarithmic class (grey bars).

Modified SP algorithm

Why are these contributions so small? In Figure 15, we investigated the 2,249,997 non-zero contributions among the 2,427 genotyped animals in population P2. This figure shows, on a logarithmic scale, how the average absolute value of a contribution decreases as the contributor becomes more distant, i.e. as more branches have to be searched to find this contributor. On average, contributions from a contributor that is located at 4 branches are approximately equal to 10^{-2}. We observe the same decrease in our small example: in Figure 13, indexes show the number of branches to explore to connect animal 12 to other animals and the contributions (last row of Table 15) also decreases as the contributor is more distant. Therefore, it would be meaningful to restrict the computations of contributions to contributors found after searching a limited number of branches.

We modify the SP algorithm by adding a criterion *k*: for any genotyped animal, the search for contributors stops as soon as *k*, a maximum of number of branches to search, is reached ("RSP algorithm" for Restricted Sparsity Pattern algorithm).

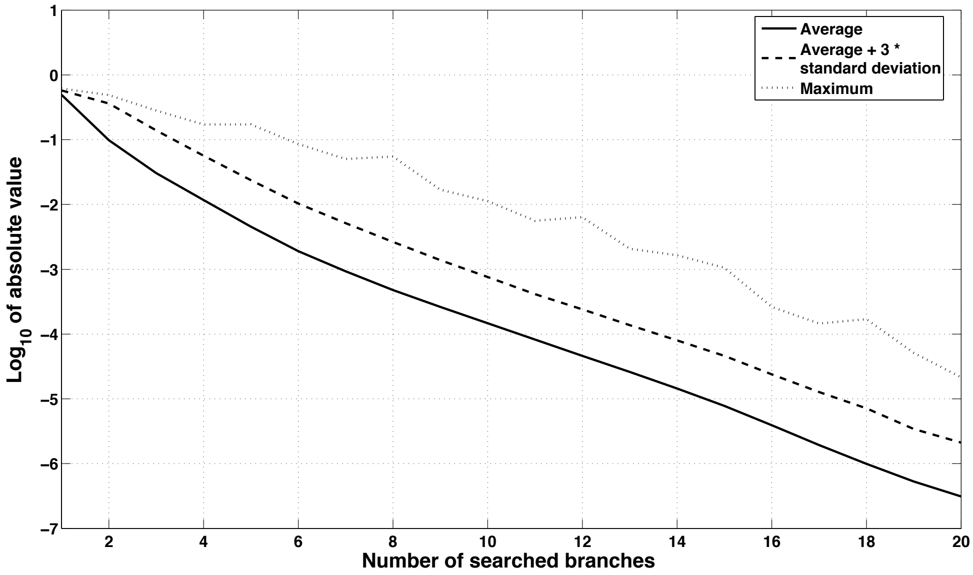

Figure 15 **Average contributions, on a logarithmic scale, in terms of the number of branches to search to connect the animal to its contributor, for a real genotyped population (P2).** Plain, dashed and dotted lines respectively show the average, the average plus 3 times the standard deviation and the maximum of absolute values of contributions found at each number of searched branches.

As an example, let us use the RSP algorithm for animal 12 in Figure 13. When *k* = 1, no contributors can be found. When *k* = 2, two contributors are found: animals 9 and 1. Considering only 2 contributors instead of 6 would greatly reduce density of the inverse (see Table 16: the percentage of zeros in $\mathbf{A}_{22}^{-1}$ is 57%). When *k* = 3, one additional contributor is found: animal 10. When *k* = 4, we add animal 8. When *k* = 5, we add animal 11 and when *k* = 6, animal 6.

To compute contributions, we assume, in equation (V.2), that only contributors found within the *k* first branches are indexed by u_i and consequently, that $\mathbf{b}_{v_i} \approx 0$. Computing relevant contributions as $\mathbf{A}_{u_i,u_i}^{-1}\mathbf{A}_{u_i,i}$ is therefore an approximation. Greater is *k*; better is the approximation, albeit less sparse: using example (Fig. 13), compare Tables 16 and 17 (approximated $\mathbf{T}^{-1}$ and $\mathbf{A}_{22}^{-1}$ respectively for $k = 2$ and $k = 4$) and Table 15 (actual $\mathbf{T}^{-1}$ and $\mathbf{A}_{22}^{-1}$).

Table 16 **Approximated $\mathbf{T}^{-1}$ (below diagonal) and $\mathbf{A}_{22}^{-1}$ (diagonal and above diagonal), when searching the pedigree for contributors located at maximum 2 branches**, for the example in Figure 13.

	1	***6***	***8***	***9***	***10***	***11***	***12***
1	1.07			0.07			-0.29
6		1.36			0.18	-0.73	
8			1.07	-0.27			
9			-0.25	1.14			-0.29
10					1.09	-0.36	
11		-0.5			-0.25	1.45	
12	-0.25			-0.25			1.14

Empty cells denote zeros. 28 null entries out of 49 for $\mathbf{A}_{22}^{-1}$ (sparse at ~57%).

Table 17 **Approximated $\mathbf{T}^{-1}$ (below diagonal) and $\mathbf{A}_{22}^{-1}$ (diagonal and above diagonal), when searching the pedigree for contributors located at maximum 4 branches**, for the example in Figure 13.

	1	***6***	***8***	***9***	***10***	***11***	***12***
1	1.07		-0.02	0.08	0.04		-0.29
6		1.36			0.18	-0.73	
8			1.07	-0.29	-0.01		0.08
9			-0.25	1.15	0.04		-0.31
10					1.11	-0.36	-0.15
11		-0.5			-0.25	1.45	
12	-0.25		0.07	-0.27	-0.12		1.17

Empty cells denote zeros. 16 null entries out of 49 for $\mathbf{A}_{22}^{-1}$ (sparse at ~33%).

Implementation of the RSP algorithm

The RSP algorithm requires keeping in memory all elements of the original matrix requested for computation of approximated contributions, i.e. elements in matrix $\mathbf{A}_{u_i,u_i}$ and vector $\mathbf{A}_{u_i,i}$ for each genotyped animal. Those requested elements actually correspond to the non-zero elements in the approximated inverse, i.e. to cross-positions of non-zero elements in $\mathbf{T}^{-1}$.

Therefore, we implemented the following sequence of operations: (1) computation of the sparsity pattern of $\mathbf{T}^{-1}$ using RSP algorithm; (2) computation of the sparsity pattern of $\mathbf{A}_{22}^{-1}$ from the sparsity pattern of $\mathbf{T}^{-1}$; (3) allocation of two sparse matrices with as much non-zero entries as expected in $\mathbf{A}_{22}^{-1}$; (4) column-wise computation of $\mathbf{A}_{22}$ (i.e. one genotyped animal at a time) using method by Colleau (2002); for each animal, only requested elements are stored at their proper places in a sparse matrix allocated at the previous step; (5) eventually, inversion itself by the computation of approximated contributions and their addition to proper places in the second sparse matrix.

In order to reduce even more the sparsity of the approximated inverse, we choose to add to the final inverse only contributions whose absolute value is greater than 10^{-3}.

Operations (1) and (5) can be parallelized as they perform independent computations for all genotyped animals. The implementation described here before has therefore also been implemented with OpenMP directives (http://www.openmp.org/).

Test protocol for accuracy of approximation

We test the approximation method for $k = 1$ to 4 and on 4 different orders of $\mathbf{A}_{22}$ (10,000, 25,000, 50,000 and 75,000). To avoid time-wasting and memory-demanding computations, actual inverses were not computed. We appreciate the quality of approximations by using them in a model equivalent to a classic mixed model and comparing the estimated breeding values (EBV) returned by both models.

The test protocol is the following: (1) simulation of true breeding values (TBV) and phenotypes (3 phenotypes per animal, with 3 different heritabilities: 0.10, 0.30 and 0.50) only for genotyped animals, using the simulation method by Van Vleck (1994); (2) prediction of EBV for all animals, genotyped or not, using a single-trait mixed model in which an overall mean is the sole fixed effect and an additive genetic effect and a normal error are random effects (model 1); (3) considering the same observations (thus, with observations only for genotyped animals: 1 observation per genotyped animal), prediction of EBV for only genotyped animals using a model equivalent to the previous one, but with a reduced number of levels for the genetic effect (only genotyped animals) and using the approximated $\mathbf{A}_{22}^{-1}$ in the mixed model equations (model 2). If we would have used the actual $\mathbf{A}_{22}^{-1}$ instead of the approximated $\mathbf{A}_{22}^{-1}$ in model 2, then solutions of model 2 and solutions of model 1 for genotyped animals would have been equal.

Therefore, comparisons involve 2 types of EBV: those of genotyped animals computed with model 1 (EBV1) and those computed with different models 2 using different approximations of $\mathbf{A}_{22}^{-1}$ (EBV2*k*, where *k* stands for the maximum number of branches searched by RSP algorithm in this approximation). Spearman rank correlations between EBV1 and EBV2*k* and the linear regression of EBV2*k* on EBV1 are computed, as well as the variance of the difference (EBV1-EBV2*k*). Also, reliabilities are computed, for EBV1 and EBV2*k*, as the squared correlation between TBV and EBV.

Results and Discussion

Approximation accuracy

On Figure 16, reliabilities of EBV1 (black bars) and EBV2*k* (white and grey bars) are given for the 4 orders of $\mathbf{A}_{22}$ and the 3 simulated sets of phenotypes (each with a different heritability). White and grey bars are reliabilities using, from left to right, a *k* (maximum number of searched branches) from 1 to 4 in the approximation of $\mathbf{A}_{22}^{-1}$. Comparisons have to be done between each group of 5 bars (same heritability and matrix order), not between groups of different heritabilities and/or matrix order.

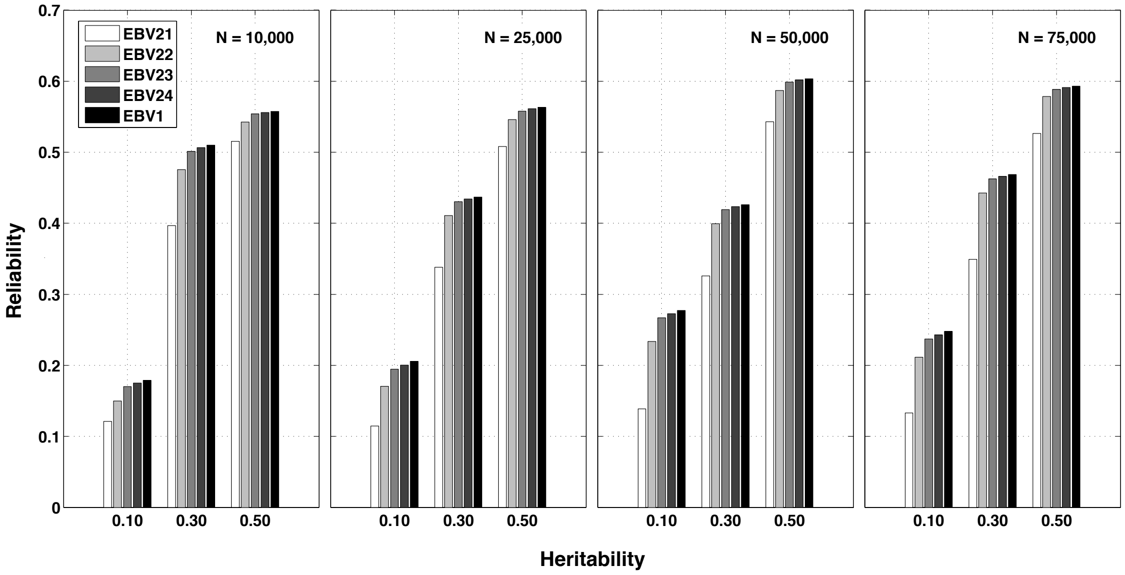

Figure 16 **Reliabilities (as squared correlations between true and estimated breeding values) of EBV1, EBV21, EBV22, EBV23 and EBV24 for 4 different orders of $\mathbf{A}_{22}$ and 3 different heritabilities (0.10, 0.30 and 0.50).**

Using *k* = 4 does not highly impact the reliability of breeding values. In the worst case (h^2 = 0.10; matrix order = 25,000), reliabilities of EBV1 and EBV24 are respectively 20.58% and 20.02% and in the best case (h^2 = 0.50; matrix order = 50,000), reliabilities of EBV1 and EBV24 are respectively 60.33% and 60.18%. Moreover, the slopes and intercepts of the linear regressions of EBV1 on EBV24 are respectively close to 1 and 0 (Table 18), showing that no inflation/deflation affects the estimation of breeding values.

The ratio between variance of the difference (EBV1-EBV24) and variance of EBV1 ranges from 0.27% ($h^2 = 0.30$; matrix order = 50,000) to 1.97% ($h^2 = 0.10$; matrix order = 10,000), suggesting that no one breeding value in EBV24 was completely different from those in EBV1. For a visual appreciation, this worst case ($h^2 = 0.10$; matrix order = 10,000) is plotted on Figure 17.

In addition, when $k = 4$, there are no major re-rankings between EBV, as Spearman's rank correlations in Table 18 suggest: all correlations are above 99%, except for the fore-mentioned worst case (98.88%).

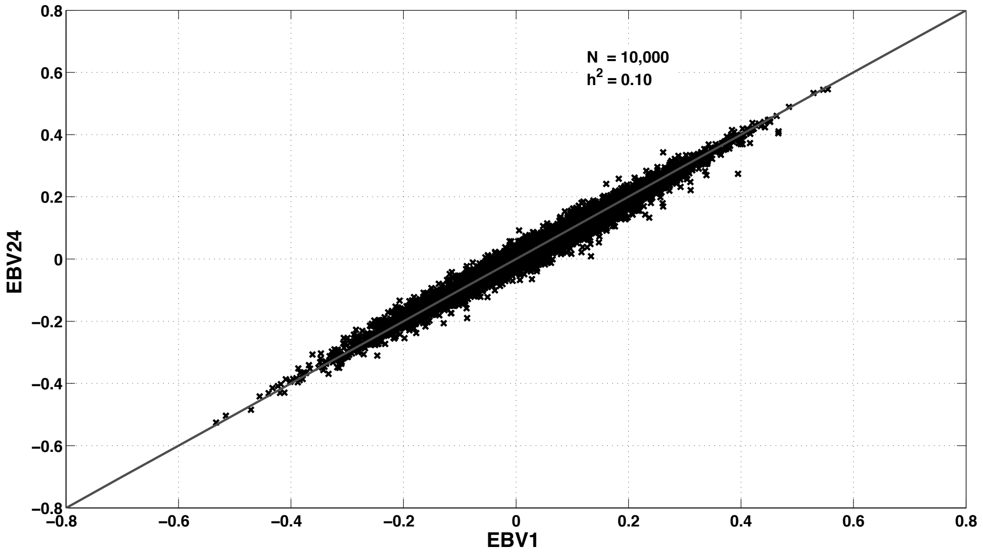

Figure 17 **Estimated breeding values of 10,000 genotyped animals using an animal model (EBV1) vs. estimated breeding values using an equivalent model with approximated $\mathbf{A}_{22}^{-1}$ for which the maximum number of searched branches (*k*) was equal to 4 (EBV24).** Heritability was 0.10. The identity line is in grey.

Table 18 **Spearman rank correlations between EBV1 and EBV24, slope and intercept of the linear regression of EBV24 on EBV1, variances of EBV1 and variances of the difference between EBV1 and EBV24** for 4 different orders of $\mathbf{A}_{22}$ and 3 heritabilities per order.

Matrix order		10,000			25,000			50,000			75,000	
Heritability	0.1	0.3	0.5	0.1	0.3	0.5	0.1	0.3	0.5	0.1	0.3	0.5
Rank correlation	98.88%	99.61%	99.79%	99.08%	99.65%	99.83%	99.31%	99.65%	99.85%	99.08%	99.64%	99.82%
Slope	0.98	0.99	1.00	0.98	0.99	1.00	0.98	0.99	1.00	0.98	0.99	1.00
Intercept	0.00	0.02	0.00	0.00	-0.03	0.02	0.00	-0.01	-0.01	0.00	0.01	0.02
Var(EBV1)	0.018	0.179	0.273	0.018	0.179	0.273	0.027	0.125	0.322	0.024	0.150	0.298
Var(EBV1-EBV24)	3.5E-04	1.3E-03	1.0E-03	3.1E-04	8.6E-04	8.8E-04	3.2E-04	8.0E-04	8.6E-04	3.8E-04	9.9E-04	9.6E-04

Computation time

On a Intel® Core™ i7 with 12 computing units (clock speed: 3.20 GHz; RAM: 64 Gbytes), the computing times were averaged on 10 repetitions, for approximations of $\mathbf{A}_{22}^{-1}$ when *k*, the maximum number of searched branches, is equal to 4 (Fig. 18). These times covered the three following operations: computation of the sparsity pattern of $\mathbf{T}^{-1}$, computation of the sparsity pattern of $\mathbf{A}_{22}^{-1}$ and approximation of the inverse.

For 75,000 genotyped animals, computing the approximated inverse required about 4 hours using the serial implementation and about 81 minutes using the parallel implementation on 4 threads, in which respectively ~5 and ~3 minutes were dedicated to computation of sparsity patterns. As the actual $\mathbf{A}_{22}^{-1}$ was not computed, these computing times can only be compared to other values in literature. For instance, Aguilar et al. (2011) reported inversion times using generalized inverse and optimized subroutines for parallel computation: about 3,160 ($10^{3.5}$) seconds for the actual $\mathbf{A}_{22}^{-1}$ an order of 25,000. In our implementation, the approximated $\mathbf{A}_{22}^{-1}$ of an order of 25,000 was 453 ($10^{2.65}$) seconds with serial and 150 ($10^{2.18}$) seconds with parallel (4 threads) implementations.

For all cases (4 different sizes and 4 values of *k*), speed-ups using parallel implementation with 4 threads were about 3 times compared to use of serial implementation.

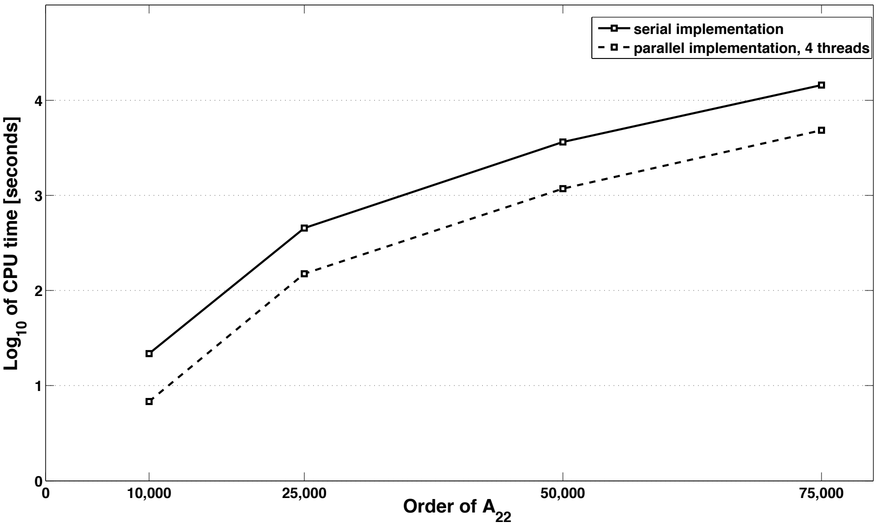

Figure 18 **Computing times for approximated inverses of different orders of** $\mathbf{A}_{22}$, when the number of branches searched equals 4, for serial (plain line) and parallel (dashed line) implementations.

Memory requirements and sparsity

As matrices were sparse and stored in sparse structures, the memory peaks for approximation of the inverse when $k = 4$ were limited: about 1.53, 6.24, 12.80 and 29.47 Gb for the respective orders 10,000, 25,000, 50,000 and 75,000, using serial implementation.

Parallel implementation required more memory at the consumption peak, about twice of the serial implementation when using 4 threads (e.g. 55.51 Gb for 75,000 genotyped animals). The reason is that the sparse matrix that stored the inverse was explicitly duplicated for each thread and reduced into a single one by the end of the parallel region, in order to avoid between-threads competition to access the same element.

When $k = 1$, the RSP algorithm only explores one branch and thus only select as contributors the parents of the current animal, if those were genotyped. The resulting approximated inverse is as sparse as a $\mathbf{A}^{-1}$ of same order (more than 99.98% of zeros in all cases outlined). When $k = 4$, the approximated $\mathbf{A}_{22}^{-1}$ is definitely less sparse: from ~78% (order of 25,000) to ~84% (order of 10,000). But, even if less sparse, it means that only ~16% to ~22% of the memory needed to store the actual inverse is required when using the approximated inverse.

Conclusions

The absolute value of a contribution to the additive relationships of a genotyped animal decreases, in average, as the contributor becomes more distant of this genotyped animal in terms of pedigree branches. The algorithm to set up the sparsity pattern of $\mathbf{A}_{22}^{-1}$ using pedigree information can be modified (RSP algorithm) in order to restrict the pedigree search to a limited number of branches. The restricted sparsity pattern can then be used in approximation of $\mathbf{A}_{22}^{-1}$. If the search for contributors of a genotyped animal does not exceed more than 4 branches to search, then the approximation can be quickly computed, even for large matrices (orders of 25,000 to 75,000). Using this approximated $\mathbf{A}_{22}^{-1}$ in a mixed model showed that prediction of breeding values was not highly impacted; therefore that such approximation is accurate.

Acknowledgements

P. Faux gratefully acknowledges the financial support of the Fonds National de la Recherche Scientifique Luxembourg (FNR). The authors acknowledge the technical and financial support of the Ministry of Agriculture of the Walloon Region of Belgium through the project "DairySNP" (D31-1274). F. Colinet, S. Vanderick (University of Liège) and the Walloon breeding association (AWE) are acknowledged for preparing and providing the pedigree data of genotyped animals.

References

Aguilar I., Misztal I., Legarra A. and Tsuruta S., 2011. Efficient computation of the genomic relationship matrix and other matrices used in single-step evaluation. *J. Anim. Breed. Genet.*, 128, 422–428.

Christensen O.F. and Lund M.S., 2010. Genomic prediction when some animals are not genotyped. *Genet. Sel. Evol.*, 42, 2.

Colleau J.-J., 2002. An indirect approach to the extensive calculation of relationship coefficients. *Genet. Sel. Evol.*, 34, 409–421.

Faux P., Gengler N. and Misztal I., 2012. A recursive algorithm for decomposition and creation of the inverse of the genomic relationship matrix. *J. Dairy Sci.*, 95, 6093–6102.

Faux P. and Gengler N., 2013. Inversion of a part of the numerator relationship matrix using pedigree information. *Genet. Sel. Evol.*, 45, 45.

Legarra A., Aguilar I. and Misztal I., 2009. A relationship matrix including full pedigree and genomic information. *J. Dairy Sci.*, 92, 4656–4663.

Meyer K., Tier B. and Graser H.-U., 2013. Technical note: Updating the inverse of the genomic relationship matrix. *J. Anim. Sci.*

Misztal I., Legarra A. and Aguilar I., 2009. Computing procedures for genetic evaluation including phenotypic, full pedigree, and genomic information. *J. Dairy Sci.*, 92, 4648–4655.

Misztal I., Legarra A. and Aguilar I., 2014. Using recursion to compute the inverse of the genomic relationship matrix. *J. Dairy Sci.*

Van Vleck L.D., 1994. Algorithms for Simulation of Animal Models with Multiple Traits and with Maternal and Non-Additive Genetic Effects. *Rev Bras. Genet.*, 17, 53–57.

BRIEF SUMMARY OF CHAPTER V

- The pedigree search for establishment of the sparsity pattern of the inverse Cholesky factor of $\mathbf{A}_{22}$ can be limited to a defined number of pedigree branches. However, it requires a non-sequential implementation of the algorithm that exhaustively searches the pedigree to set up this sparsity pattern.
- Using an approximated $\mathbf{A}_{22}$ (based on a approximated sparsity pattern of $\mathbf{A}_{22}^{-1}$) instead of a regular $\mathbf{A}_{22}$ in a simple mixed model on simulated phenotypes does not greatly impact the computation of breeding values, if the pedigree searches for contributors are stopped once a maximum of 4 branches were explored.
- Pedigree search for dependencies as well as computation of the vector of contributions and the subsequent update of a zeroed matrix to obtain the approximated $\mathbf{A}_{22}^{-1}$ are independent operations for each genotyped animal. Therefore, this approximation of $\mathbf{A}_{22}^{-1}$ supports parallelization and, using 4 threads, the approximation was sped up by 3.

RELATED PUBLICATIONS

- P. Faux and N. Gengler. 2014. Efficient approximations of the inverse of a part of the additive relationship matrix. At: *2014 World Congress on Genetics Applied to Livestock Production*. Poster in Vancouver (Canada).

Chapter VI

GENERAL DISCUSSION

Based on the framework for inversion of relationship matrices shown in Chapter II, different algorithms and strategies were proposed in Chapters III to V. All can be used with $\mathbf{A}_{22}$ whereas only those of Chapter III can be used with **G**. In this final chapter, we first draw a comparative study between the different algorithms applied to each specific case of matrix. This comparative study is needed because tests in the previous chapters were run on different machines.

The literature review in Chapter II gave the two main axes of our research. First, most of the relationship matrices used in animal breeding are inverted using the algorithm of Sherman and Morrison (1950). The main reason is its original use by Henderson (1976) and Quaas (1976) for the specific case of **A**. By analogy, we proposed to follow the same algorithm for the two relationship matrices concerned by our study ($\mathbf{A}_{22}$ and **G**).

Secondly, the core of this algorithm is the computation, animal by animal, of the contributions of animals preceding the current animal to relationships of this animal with the rest of the population. This computation covers two aspects: (1) determining which animals have a non-zero contribution (those are later called *contributors*); and (2) assessing the value of their contribution.

Approximations of the inverses of $\mathbf{A}_{22}$ or **G** can be made by an approximation of the search for contributors and a subsequent approximation of their contributions.

Throughout chapters III to V, different algorithms and strategies were proposed to determine contributors (or to approximate their determination), for the specific cases of $\mathbf{A}_{22}$ (Chapters III, IV and V) and of **G** (Chapter III). Also, different implementations of the Sherman-Morrison algorithm have been proposed to compute the values of actual or approximated contributions. In the next subsections, we present a comparative study of these proposed algorithms, strategies and implementations.

Algorithms and strategy for determination of contributors in the specific case of $\mathbf{A}_{22}$

In Chapter IV, an algorithm was proposed to exhaustively search the pedigree in order to genuinely determine the contributors of any animal in $\mathbf{A}_{22}$. These contributors being the animals having a non-zero value in the lower part of the inverse Cholesky factor of $\mathbf{A}_{22}$ ($\mathbf{T}^{-1}$), this algorithm establishes the sparsity pattern of $\mathbf{T}^{-1}$. We later denote it as "SP algorithm" (for "Sparsity Pattern algorithm").

From Chapters III to V, three different algorithms for approximating the determination of contributors have been proposed. These are the following: the close-family approach ("CF" algorithm), the recursive close-family approach ("RCF" algorithm) and the restricted sparsity pattern algorithm" ("RSP" algorithm). In addition, in Chapter IV, a strategy that consists to compute the sparsity pattern of $\mathbf{T}^{-1}$ using a pedigree with fewer extracted generations of non-genotyped ancestors for genotyped animals has been proposed (sparsity pattern algorithm, with limited extracted generations or “SPG” algorithm).

Details on algorithms and strategies

The SP algorithm

Rules to establish the sparsity pattern of $\mathbf{T}^{-1}$ were first enunciated in Chapter IV (section “*Material and Methods*”). However, a more comprehensive formulation of this algorithm was given in Chapter V (section “*Material and Methods*”). Following the latter, an example is given for animal 16 in Figure 19: exploration of ascending branches reveals animals 4, 6 and 10 as contributors, exploration of descending branches from non-genotyped ancestors reveals animals 11, 12 and 13 as contributors and eventually, applying the same rules for those last contributors reveals animal 8 as being a contributor too.

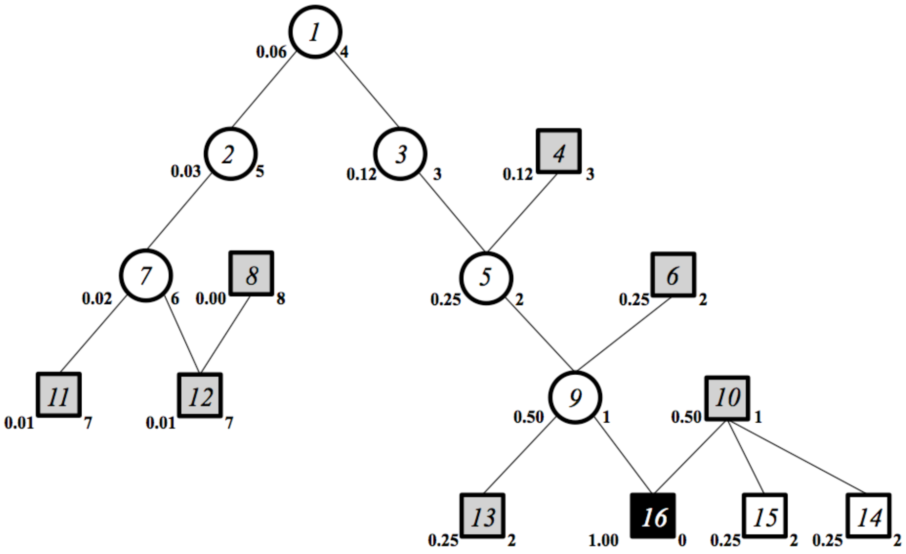

Figure 19 **Genealogical tree of 16 animals (10 genotyped, in squares)**, showing the actual contributors (in grey) of animal 16th. Identifiers give age order (1: oldest; 16: youngest). Left and right subscripts are respectively the additive relationship with animal 16th and the number of branches to reach the contributor from animal 16th.

The CF algorithm

Close-family of a genotyped animal was firstly introduced, in equation (III.3), such as the set of genotyped animals older than him and sharing a relationship greater or equal to a threshold *p* with this animal. This algorithm does not require any pedigree search; contributors are simply selected on the value of their relationships with the current animal. Note that, if the close-family threshold p is set to 0, then all animals are considered as contributors and, consequently, the non-contributing animals obtain a null contribution while actual contributors obtain their actual contribution. However, two types of errors may be encountered: discarding actual contributors (e.g. animals 8, 11 and 12 in the illustrated example of Figure 20) or considering non-contributing animals as contributors (e.g. animals 15 and 14 in Figure 20).

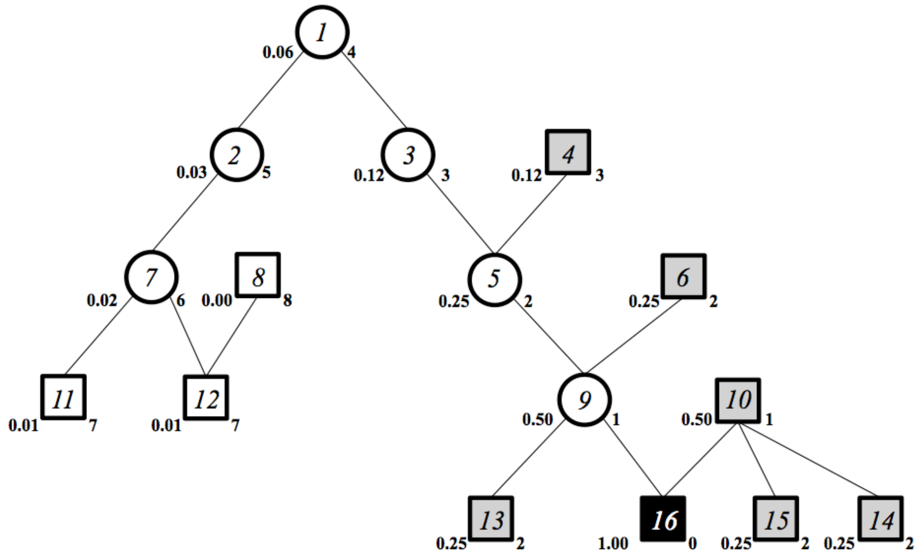

Figure 20 **Genealogical tree of 16 animals (10 genotyped, in squares)**, showing the contributors (in grey) of animal 16th found using the CF algorithm (considering the animals sharing an additive relationship greater or equal to 0.10 as members of the same close-family). Identifiers give age order (1: oldest; 16: youngest). Left and right subscripts are respectively the additive relationship with animal 16th and the number of branches to reach the contributor from animal 16th.

The RCF algorithm

Multiplying $\mathbf{A}_{22}$ by the $\mathbf{T}^{-1}$ approximated by the CF algorithm and its transpose (equation III.5) produces a matrix $\mathbf{D}$ whose inverse is approximated, in the CF algorithm, by an element-wise inversion of the diagonal. However, an approximation of the inverse of $\mathbf{D}$ can be obtained by computing an approximated $\mathbf{T}^{-1}$ for this matrix in a similar manner as the CF algorithm achieved for $\mathbf{A}_{22}$. The same operation is then recursively applied on the resulting $\mathbf{D}$ until this matrix is diagonal. Using two rounds of recursion (the first with a threshold equal to 0.10 and the second with a threshold equal to 0.005) in our illustrated example (Figure 21), we considered more animals as contributors than with the CF algorithm. However, we still face the same problems, namely discarding contributors (animal 8) and considering non-contributing animals as contributors (animals 15 and 14).

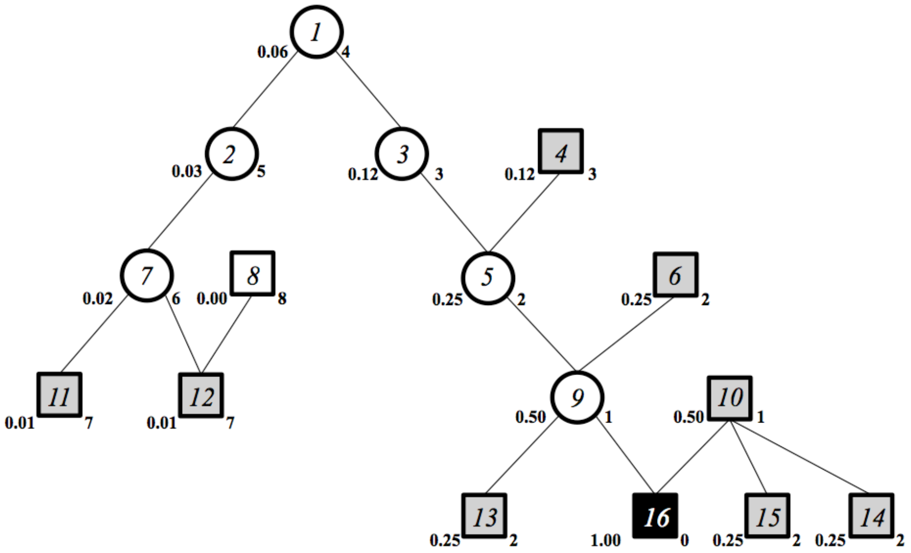

Figure 21 **Genealogical tree of 16 animals (10 genotyped, in squares)**, showing the contributors (in gray) of animal 16th found using the RCF algorithm (considering two rounds of recursion with the respective thresholds: 0.10 and 0.005). Identifiers give age order (1: oldest; 16: youngest). Left and right subscripts are respectively the additive relationship with animal 16th and the number of branches to reach the contributor from animal 16th.

The fore-mentioned problems of misattributing contributors were circumvented by the development of the SP algorithm. However, the number of contributors can be large in regard to matrix order whereas most of them still have small contributions. This led us to restrict the number of contributors by limiting the pedigree search to the close-neighbourhood of the current animal. It was achieved by two means: the SPG strategy and the RSP algorithm.

The SPG strategy

In this strategy, the fully-extracted pedigree is replaced by a less deep pedigree for genotyped animals: extraction of non-genotyped ancestors of genotyped animals is only made for few generations. This pedigree is then used for both computation of $\mathbf{A}_{22}$ and determination of contributors using the SP algorithm. It means consequently that the approximation of $\mathbf{A}_{22}^{-1}$ is actually the inverse of an approximation of $\mathbf{A}_{22}$, since it was computed by ignoring some pedigree links. This strategy is illustrated (Figure 22) by considering only 2 generations of non-genotyped ancestors of genotyped animals, what breaks links between genotyped animals.

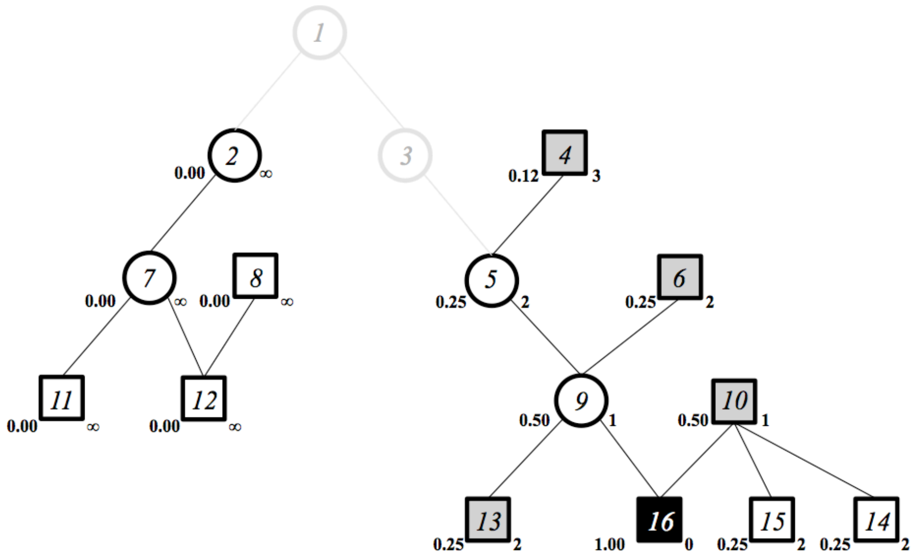

Figure 22 **Genealogical tree of 16 animals (10 genotyped, in squares)**, showing the contributors (in gray) of animal 16th found using the SPg strategy (performing extraction of 2 generations of non-genotyped ancestors of genotyped animals). Identifiers give age order (1: oldest; 16: youngest). Left and right subscripts are respectively the additive relationship with animal 16th and the number of branches to reach the contributor from animal 16th. Animals and branches in grey are ignored.

The RSP algorithm

The previous strategy has one main drawback: when the genotyped population is made of animals of the same generation sharing a lot of common ancestors, then extracting only two or three generations of non-genotyped ancestors does not help to reduce the number of contributors. We have then proposed (Chapter V) to restrict pedigree search for establishment of the sparsity pattern to a limited number of branches in order to discard contributors with small contributions. This option is illustrated in Figure 23, where the pedigree search is stopped once a maximum number of 4 branches were searched. In this example and for this animal, it led to consider the same contributors as with the SPG strategy.

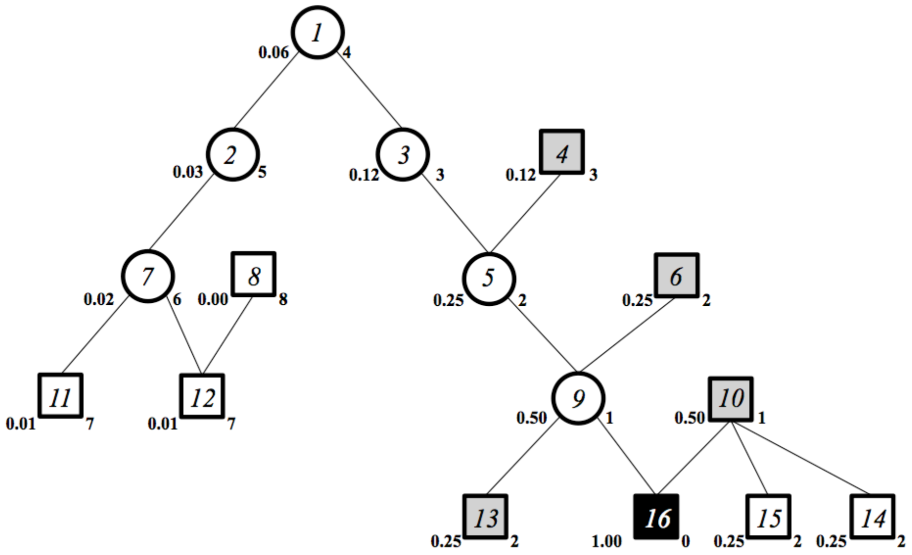

Figure 23 **Genealogical tree of 16 animals (10 genotyped, in squares)**, showing the contributors (in gray) of animal 16th found using the RSP algorithm (stopping the pedigree search when a maximum of 4 branches were explored). Identifiers give age order (1: oldest; 16: youngest). Left and right subscripts are respectively the additive relationship with animal 16th and the number of branches to reach the contributor from animal 16th.

Implementation of the computation of the inverse or of its approximation in the specific case of A_{22}

The Sherman-Morrison algorithm for inversion was formulated in different manners throughout the chapters of this thesis (equations II.13 and IV.3 – where it is referred as blockwise inverse– or partially in equation V.1). Based on equation (IV.3), the core of the algorithm is the computation of a contribution vector (**b**), for any genotyped animal. This vector is then weighted by a factor (α), multiplied by its transpose ($\mathbf{b}'$) and added to the previous results.

Following this formulation, in Chapter IV, we proposed three procedures to obtain the vector of contributions:

(1) when there are no contributors, procedure *EMPTY*: only the weighting factor α is computed;

(2) when the number of contributors is relatively small in regard of the row number, procedure *LS*: **b** is computed by solving a low-sized linear system;

(3) when the number of contributors is relatively high in regard of the row number, procedure *PROD*: **b** is computed on the basis of the inverse updated at the previous iteration.

Procedure *EMPTY* is the less likely to occur; only founders require it. Procedures *LS* and *PROD* have an opposite advantage/disadvantage: *PROD* requires the inverse of the previous iteration whereas *LS* does not (the main loop of an algorithm using *LS* may be parallelized); *PROD* is featured for large number of contributors whereas *LS* is not (complexity increases cubically with number of contributors, see Table 13).

We tested the algorithms and strategy here before (SP, CF, RCF, RSP algorithms and SPG strategy) with the following implementations for each of them.

The SP algorithm was implemented with only *PROD* procedure. The vector of contributions is obtained by computing the vector-column of $\mathbf{A}_{22}$ (using method of Colleau, 2002) corresponding to the current animal and multiplying the inverse updated at the previous iteration by this vector-column, only for contributors. The inverse is then updated before moving to next genotyped animal in pedigree. The SPG strategy uses the same implementation.

Approximations made with CF, RCF and RSP algorithms cannot use the *PROD* procedure: computing an approximation of contributions, then adding it to the previous inverse and using this one to compute contributions for next animals quickly leads to numerical divergence. Therefore, the CF, RCF and RSP algorithms are implemented using *LS* procedure. For the comparisons here after, none of these implementations was taking advantage of parallelization, albeit operations can be made in parallel for all genotyped animal (in the case of CF, RCF and RSP algorithms) and matrix operations for a single animal can be improved by use of multi-thread libraries (e.g. MKL, LAPACK).

Note that, here below, the names of algorithms and strategy will refer to their implementation rather than only to the algorithm/strategy itself.

Comparative study between the different approximation algorithms

Test protocol and materials

Comparisons involved different sizes of matrices (4,000; 8,000; 16,000 and 32,000) obtained from genotyped populations designed as in Chapter IV. The test protocol was identical to that of Chapter V: one TBV and one phenotype were simulated using the method of Van Vleck (1994) for each genotyped animal assuming a heritability of 0.30. The solutions of a model with an overall mean and an additive genetic effect were computed using a regular animal model. An equivalent model that uses the approximation of the inverse of $\mathbf{A}_{22}$ times the genetic variance component as variance matrix for the genetic effect was used to compute approximated EBVs for genotyped animals. The Spearman rank correlation between these approximated EBVs and the actual EBVs is recorded for each type of approximation to assess re-rankings of animals due to the use of approximations. The reliability of approximated and actual EBVs are also computed. Eventually, the distance norm *N* between a matrix and its approximation is computed for each approximation of $\mathbf{A}_{22}^{-1}$ (*N* was introduced in Chapter IV).

As $\mathbf{A}_{22}$ is computed and used one column at a time in the implementation of the SP algorithm, the recorded computing times cover two operations: computation of $\mathbf{A}_{22}$ and computation/approximation of $\mathbf{A}_{22}^{-1}$.

For each type of approximations (CF, RCF, SPG and RSP algorithms), the ranges of approximation parameters (close-family threshold(s), number of generations to extract and maximum number of branches to search) were limited to relevant approximations only. Each $\mathbf{A}_{22}^{-1}$ was also obtained using the Fortran 95 subroutine DKMXHF by K. Meyer (University of New-England, Australia).

All computations were made on a Intel® Core™ i7 computer with 12 computing units (clock speed: 3.20 GHz; RAM: 64 Gbytes). Results for computing times and memory requirements are given in Table 19 and results for quality of approximation in Table 20.

Results

In Table 20, one may consider that an approximation is accurate at a first level (Spearman correlation above 99%, what matches with a norm *N* less than 1E-2) or at a second level (Spearman correlation above 99.5%, what matches with a norm *N* less than 5E-3). First level of approximation accuracy happens with CF when p=010, with RCF when p_1=0.10 and p_2=0.01, with SPG when g=2 or more and with RSP when k=4 or more. In regard to computing time for the actual inverse (DKMXHF and SP algorithms in Table 19), computing times by RCF are prohibitive and those using SPG when g=2 or more are in the same range as the actual inverses. Therefore, for a first level approximation, the challenge is between CF (when p=0.10) and RSP (when k=4). Second level of approximation accuracy challenges computing time of the actual inverse only with RSP, when k=5.

Table 19 **Computing requirements of different algorithms for computing or approximating the inverse of $\mathbf{A}_{22}$**. Computing times (in minutes) and maximum required random-access memory (in Gbytes) for computing or approximating the inverse of 4 different orders of $\mathbf{A}_{22}$ using different algorithms, as well as the percentage of elements treated as zero in the computed (actual or approximated) inverse.

				CF		***RCF***	***SPG***				***RSP***			
Size		***DKMXHF***	***SP***	*p=0.20*	*p=0.10*	p_1=0.10 p_2=0.01	*g=0*	*g=1*	*g=2*	*g=3*	*k=2*	*k=3*	*k=4*	*k=5*
4,000	*Time*	0.267	0.649	0.022	0.033	0.739	0.003	0.283	0.491	0.568	0.025	0.032	0.043	0.072
	Memory	0.066	2.279	0.147	0.200	0.262	0.127	0.220	0.250	0.257	0.018	0.018	0.022	0.062
	%0	-	43.142	99.642	85.805	60.027	99.974	65.178	48.260	44.192	99.891	99.050	95.239	79.784
8,000	*Time*	2.095	4.791	0.069	0.186	4.235	0.010	2.622	4.008	4.521	0.078	0.103	0.159	0.442
	Memory	0.255	5.275	0.753	0.824	1.031	0.502	0.915	1.002	1.025	0.038	0.049	0.095	0.297
	%0	-	42.408	99.465	83.288	61.441	99.987	59.449	46.341	43.107	99.897	98.594	92.423	74.537
16,000	*Time*	17.195	36.513	0.252	1.856	71.966	0.045	22.869	32.582	35.194	0.252	0.369	0.757	3.828
	Memory	1.006	9.310	3.006	3.253	4.440	2.004	3.727	3.995	4.051	0.136	0.196	0.457	1.472
	%0s	-	43.682	99.214	80.548	62.282	99.993	56.701	46.525	44.458	99.899	97.990	89.929	70.331
32,000	*Time*	146.956	309.773	0.883	20.445	732.868	0.294	206.483	280.591	299.162	0.879	1.613	4.635	44.529
	Memory	4.009	42.951	12.014	12.685	17.553	8.032	33.723	34.492	34.672	0.527	0.842	2.096	5.904
	%0	-	47.803	98.923	78.570	63.355	99.996	57.414	50.058	48.347	99.894	97.314	88.010	68.681

Approximations parameters: *p* (CF) is the close-family threshold; p_1 and p_2 (RCF) are respectively the threshold of the first and second rounds of recursion; *g* (SPG) is the number of extracted generations of non-genotyped ancestors of genotyped animals; *k* (RSP) is the maximum number of searched pedigree branches.

Table 20 **Quality of approximation of the inverse of $\mathbf{A}_{22}$ by different algorithms**. Spearman rank correlations (ρ) and reliabilities (REL) of actual and approximated EBV for 4 different numbers of genotyped animals using different algorithms, as well as the difference norm *N* between the actual and approximated inverses of $\mathbf{A}_{22}$.

			CF		***RCF***	***SPG***				***RSP***			
Size		***Actual inverse***	*p=0.20*	*p=0.10*	*p1=0.10* *p2=0.01*	*g=0*	*g=1*	*g=2*	*g=3*	*k=2*	*k=3*	*k=4*	*k=5*
4,000	*ρ (%)*	100	97.61	99.43	99.83	94.26	98.66	99.80	99.96	96.81	98.60	99.17	99.68
	REL(%)	33.24	32.11	32.77	33.05	31.23	32.61	33.06	33.23	31.52	32.58	32.56	32.82
	N	0	0.023	0.008	0.004	0.071	0.009	0.002	0.001	0.025	0.013	0.007	0.004
8,000	*ρ (%)*	100.00	97.97	99.50	99.80	93.14	99.24	99.86	99.97	97.34	99.00	99.49	99.77
	REL(%)	34.52	33.28	34.25	34.39	30.35	34.17	34.39	34.51	32.72	33.95	34.11	34.30
	N	0	0.022	0.008	0.004	0.080	0.008	0.002	<0.001	0.022	0.012	0.006	0.003
16,000	*ρ (%)*	100.00	97.40	99.43	99.79	91.21	98.47	99.73	99.96	96.31	98.50	99.34	99.78
	REL(%)	37.14	35.54	36.78	37.15	30.98	36.25	37.06	37.17	34.76	36.21	36.80	37.11
	N	0	0.019	0.007	0.004	0.088	0.007	0.001	<0.001	0.020	0.010	0.005	0.003
32,000	*ρ (%)*	100.00	97.92	99.53	99.81	90.34	99.03	99.84	99.97	96.96	98.93	99.55	99.83
	REL(%)	38.12	36.65	37.86	38.00	31.09	37.43	38.00	38.09	35.89	37.44	37.86	37.99
	N	0	0.016	0.007	0.003	0.091	0.005	0.001	<0.001	0.016	0.008	0.004	0.002

Approximations parameters: *p* (CF) is the close-family threshold; p_1 and p_2 (RCF) are respectively the threshold of the first and second rounds of recursion; *g* (SPG) is the number of extracted generations of non-genotyped ancestors of genotyped animals; *k* (RSP) is the maximum number of searched pedigree branches.

Discussion

For these three different approximations (CF when p=0.10 and RSP when k=4 or k=5), the computing time ratios between the elapsed time for DKMXHF and the elapsed times for each approximation, for the 4 different orders of $\mathbf{A}_{22}$, are plotted in Figure 24.

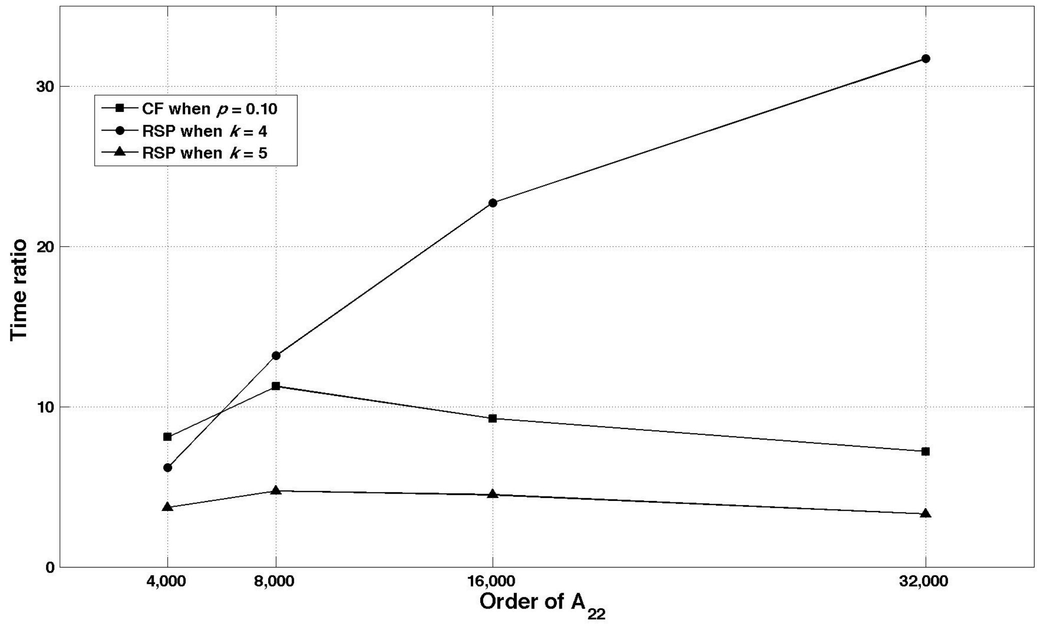

Figure 24 **Computing time ratios** between computing the actual inverse using DKMXHF and the approximated inverse using 3 approximations (CF when the close-family threshold is set to 0.10 and RSP when the number of searched branches is limited to a maximum of 4 or 5), for 4 different orders of $\mathbf{A}_{22}$.

This figure shows that, for a comparable approximation, RSP (when k=4) is a better choice than CF (when p=0.10): the speed-up for having an approximation instead of the actual inverse increases up to ~31 times faster with the order of the matrix with RSP when k=4, whereas it slightly decreases for orders above 8,000 with CF when p=0.10. As explained above, CF algorithm makes two types of errors (discarding relevant contributors/considering non-contributing animals as contributors), whereas RSP only removes distant contributors, expected to have smaller contributions (see Figure 15). Therefore, as the order of the matrix increases, the number of animals considered as contributors keeps growing with CF whereas it stays stable with RSP, what would explain the greater computing times required by CF. In addition, the resulting approximation is ~10% sparser using RSP when k=4 than CF when p=0.10. It may be worth to detail the elapsed time between the different operations when using the RSP algorithm.

For instance, here below are the proportions of total time (4.635 min; see Table 19) covered by each operation to approximate inversion of a matrix of order 32,000 with RSP when *k*=4:

- setting up the restricted sparsity pattern of $\mathbf{T}^{-1}$: 10% of total time;
- setting up the restricted sparsity pattern of $\mathbf{A}_{22}^{-1}$: less than 1% of total time;
- computing $\mathbf{A}_{22}$: 14% of total time;
- computing the contributions: 44% of total time;
- adding the product of the vector of contributions to $\mathbf{A}_{22}^{-1}$: 31% of total time.

The last operation would be unnecessary if a rearrangement of equations avoids the use of the explicit $\mathbf{A}_{22}^{-1}$. In our parallelized implementation, this operation is also critical: it demands more memory to avoid thread competitions when accessing the same element in memory. Circumventing this operation may therefore allow using more threads without requiring prohibitive amounts of memory.

Memory requirements depend on the implementation: number of variables to store (e.g. only $\mathbf{A}_{22}$ and few working vectors in the case of DKMXHF vs. $\mathbf{A}_{22}$, $\mathbf{A}_{22}^{-1}$, $\mathbf{T}^{-1}$ and $\mathbf{T}_f^{-1}$ in the case of RCF), storage type for each of these variables (sparse or dense; triangular or full) and percentage of zeros in $\mathbf{A}_{22}^{-1}$. This latter factor (also shown in Table 19) affects the required memory if the main variables ($\mathbf{A}_{22}^{-1}$ or $\mathbf{A}_{22}$) are stored sparse, e.g. as with the current implementation of RSP algorithm. When accounting for these three factors in each implementation, the memory requirements are the same for the different algorithms, except for SP algorithm. In that case, additional memory is required for storage of an important sparsity pattern.

It might be interesting to relate on different points these algorithms to the work of Chow (2000) who introduces power of sparsified matrices to compute sparse approximate preconditioners. The approximate inverses proposed in this contribution follow the same scheme as the CF algorithm: an approximate sparsity pattern is computed retaining values of the original matrix upon a given threshold. Following the nomenclature in this paper, the RCF algorithm can be qualified as an *adaptive* procedure to produce an approximate inverse of $\mathbf{A}_{22}$: a first approximation of the sparsity pattern is performed, then a minimization problem is solved, the initial pattern is updated and the process is repeated until the approximation is good enough. Eventually, the strategy followed by Chow

(2000) is similar to our strategy in that sense that it aims to reduce computations by a pre-processing step in which an approximate sparsity pattern is computed. However, floating-point operations are still required in that study during the pre-processing step. In contrast, the main advantage of RSP algorithm is that it does not involve floating-point operations in the pre-processing step that computes an approximate sparsity pattern.

Conclusions

The best strategy to approximate the inverse of $\mathbf{A}_{22}$ when the order of the matrix exceeds tens of thousands would be to use the RSP algorithm, limiting the number of searched pedigree branches to 4 or 5. The results also show that there is no need to extract more than 3 generations of non-genotyped ancestors of genotyped animals. For instance, using a pedigree including only 2 generations of non-genotyped ancestors should reduce memory and time required by the establishment of the approximated sparsity pattern without compromising the quality of approximation.

Algorithms for determination of contributors in the specific case of G

Since the rules that prevail for the computation of $\mathbf{A}_{22}$ (Henderson, 1976; Colleau, 2002) are not the same as for **G** (VanRaden, 2008; Leutenegger et al., 2006; Bömcke and Gengler, 2009) , only the analogy between $\mathbf{A}_{22}$ and **G** might dictate to use the same (actual or approximate) sparsity pattern for both. The comparative study between approximation algorithms for the specific case of **G** is limited to the CF and RCF algorithms. Their use for this matrix was already introduced and discussed in Chapter III. However, no numerical tests were made in that chapter.

Therefore, tests were conducted and results were compared with actual inversion using subroutine DKMXHF. Tests involved the 2,427 genotyped animals available in Walloon Region of Belgium. This population was introduced in Chapter V as “P2” (see details in Table 14). In order to assess the quality of approximation, phenotypes were simulated for genotyped animals using the method of Van Vleck (1994) and a ssGBLUP with an overall mean and 17,677 levels of genetic effect, each one matching one animal was used to estimate GEBV for the whole population. The same model was also solved with the actual inverse of **G** in order to compute Spearman rank correlations between the actual and approximated GEBVs for all animals in population as well as only for genotyped populations. The approximation parameters (thresholds) were chosen so that computing times were not prohibitive.

Results (Table 21) are not good: the best Spearman correlation (97.438%) is lower than 99% for GEBVs of genotyped animals. Spearman correlations for GEBVs of all animals are even worst. Moreover, finding the appropriate close-family threshold has a cost: it requires multiple trials, increasing the computing cost of these algorithms.

Table 21 **Computing requirements of CF and RCF algorithms for computing or approximating the inverse of G and quality of approximation of its inverse.** Computing times (in seconds) and maximum required random-access memory (in Mbytes) for computing or approximating the inverse **G** for 2,427 animals, as well as the percentage of elements treated as zero in the approximated inverse and the Spearman rank correlations between GEBVs computed using approximated inverse and GEBVs computed using actual inverse, for all 17,677 animals in population (ρ_{ALL}) and for 2,427 genotyped animals (ρ_{GENO}).

		CF				***RCF***
	Actual inverse	*p=0.20*	*p=0.15*	*p=0.10*	*p=0.05*	*p1=0.10* *p2=0.01*
Time	9.021	0.116	0.260	2.376	147.153	5.704
Memory	70.100	70.704	71.452	76.448	141.428	93.740
%0	-	94.908	80.232	21.726	0.337	67.277
ρ_{ALL} *(%)*	100	83.527	69.522	60.561	66.270	74.198
ρ_{GENO} *(%)*	100	90.940	92.246	93.193	97.438	92.233

In conclusion, these two algorithms are not suited at all for approximation of the inverse of **G**. Better approximations can be found through partition of the genotyped population between young and proven animals as in Misztal et al. (2014).

References

Bömcke E. and Gengler N., 2009. Combining microsatellite and pedigree data to estimate relationships among Skyros ponies. *J. Appl. Genet.*, 50, 133–143.

Chow E., 2000. A priori sparsity patterns for parallel sparse approximate inverse preconditioners. *SIAM J. Sci. Comput.*, 21, 1804–1822.

Colleau J.-J., 2002. An indirect approach to the extensive calculation of relationship coefficients. *Genet. Sel. Evol.*, 34, 409–421.

Henderson C.R., 1976. A Simple Method for Computing the Inverse of a Numerator Relationship Matrix Used in Prediction of Breeding Values. *Biometrics*, 32, 69–83.

Leutenegger A.-L., Labalme A., Génin E., Toutain A., Steichen E., Clerget-Darpoux F. and Edery P., 2006. Using genomic inbreeding coefficient estimates for homozygosity mapping of rare recessive traits: application to Taybi-Linder syndrome. *Am. J. Hum. Genet.*, 79, 62–66.

Misztal I., Legarra A. and Aguilar I., 2014. Using recursion to compute the inverse of the genomic relationship matrix. *J. Dairy Sci.*

Quaas R.L., 1976. Computing the Diagonal Elements and Inverse of a Large Numerator Relationship Matrix. *Biometrics*, 32, 949–953.

Sherman J. and Morrison W.J., 1950. Adjustment of an inverse matrix corresponding to a change in one element of a given matrix. *Ann. Math. Stat.*, 124–127.

VanRaden P.M., 2008. Efficient Methods to Compute Genomic Predictions. *J. Dairy Sci.*, 91, 4414–4423.

Van Vleck L.D., 1994. Algorithms for Simulation of Animal Models with Multiple Traits and with Maternal and Non-Additive Genetic Effects. *Rev Bras. Genet*., 17, 53–57.

Chapter VII

PERSPECTIVES AND CONCLUSIONS

The achievement of this thesis provides three types of perspectives: perspectives of use of the developed algorithms and strategies, perspectives of future researches and perspectives for animal breeding. The last type goes beyond the framed objectives of the thesis and sketches potential benefits for breeders of research fields related to the thesis. Final conclusions are eventually drawn at the end of this chapter.

Perspectives of use of the proposed algorithms and strategies

Use of the SP algorithm

As shown in the comparative study here before, the SP algorithm sets up the full sparsity pattern of $\mathbf{A}_{22}^{-1}$ but is not helpful to reduce computations time compared to other algorithms. However, the SP algorithm is useful even as a stand-alone algorithm to study the structure of the population. Using it as a step prior to inversion or other computations involving $\mathbf{A}_{22}$ (e.g. factorization) reveals sparsity in the inverse matrix without prohibitive computational time (in our last implementation, the sparsity pattern of $\mathbf{T}^{-1}$ is returned within less than a minute for 50,000 animals). Having the sparsity pattern of $\mathbf{A}_{22}^{-1}$ provides an additional clue for the choice of algorithm to perform the upcoming computations. The cases outlined in the comparative study were having close to dense $\mathbf{A}_{22}^{-1}$, due to the way genotyped animals were chosen in the population and to the structure of this population. This might be no longer the case in other situations. Conversely, if the genotyped population is relatively small and $\mathbf{A}_{22}^{-1}$ is found dense by SP algorithm, then a classical inversion has to be preferred. Dependencies (however, without the value of the contribution) of breeding values between genotyped animals are also available through the use of the SP algorithm.

Use of the RSP algorithm

The most interesting prospective use of RSP algorithm would be to replace, in ssGBLUP, $\mathbf{A}_{22}^{-1}$ by an approximated $\mathbf{A}_{22}^{-1}$ that would require less time to be computed without compromising accuracy of GEBVs. Research work on that topic is currently in progress at the University of Georgia. Theoretical developments other than ssGBLUP also require $\mathbf{A}_{22}^{-1}$ (e.g. Vandenplas and Gengler, 2012) and might work with approximated inverses.

Also, using an $\mathbf{A}_{22}$ made of recorded animals (in an equivalent model as in Chapter V) instead of a matrix $\mathbf{A}$ containing all animals in population may substantially reduce the number of equations for variance component estimation. Reduced animal model were already proposed (Quaas and Pollak, 1980; Quaas, 1988) but a model including only recorded animals and, therefore, the part of $\mathbf{A}$ containing relationships between those animals, was, to our knowledge, not developed so far. There are two constraints to such a model. Firstly, $\mathbf{A}_{22}^{-1}$ should be obtained at a low computational cost. Secondly, $\mathbf{A}_{22}^{-1}$ has to be as sparse as possible, since variance component estimation requires inversion of the left-hand side (LHS) of the mixed model equations (MME). These two constraints may be overcome by the use of the RSP algorithm. In order to improve sparsity, other animals than only recorded ones might be included in $\mathbf{A}_{22}$.

As an illustration, let us use a small data set provided by CONVIS s.c., and designed for variance component estimation (Arnould et al., 2014). This data set is made of repeated observations on 8,811 cows sharing a fully-extracted pedigree of 22,876 animals. If $\mathbf{A}_{22}$ is made of the 8,811 recorded cows, the SP algorithm assesses the actual sparsity of its inverse as equal to 48.1% of null elements. Such $\mathbf{A}_{22}$ would be time-expensive to invert. Moreover, the LHS of the MME of such a reduced model would be too dense to be easily inverted by sparse inversion (e.g. as in Misztal and Perez-Enciso, 1993). If the top 1,000 non-recorded animals with the greatest number of offspring are added to $\mathbf{A}_{22}$, then the order of $\mathbf{A}_{22}$ is 9,811 but the actual sparsity of its inverse is 98.8% of null elements. The time required for inversion is dramatically reduced whereas the sparsity degree is not yet high enough for sparse matrix inversion of the LHS. Using the RSP algorithm with k=5 increases the sparsity degree up to 99.9%, what is enough for sparse matrix inversion of the LHS. This illustrative example shows how the RSP algorithm could be used to decrease the order of the LHS of MME. In such case, a “full” model in which the additive genetic effect accounts for 22,876 rows and columns would be replaced by a “reduced” model in which the additive genetic effect accounts for 9,811 rows and columns, however with an approximated inverse of its covariance matrix.

It is eventually worth noting that a similar approach could be used to assess how genotyping few important animals could affect the sparsity of $\mathbf{A}_{22}$ (made, in this case, of genotyped animals).

Perspectives of future researches

On the SP algorithm

A first and main perspective of future research concerns the SP algorithm. Its exploration rules were empirically set up; no formal evidence of these rules was found. However, they were tested on many different pedigrees, with or without inbreeding, and were not falsified. Different trials were made to transfer the heuristic rules to a graph theory problem (as in Chow, 2000), but did not bring any results so far. Concerning the SP algorithm, researches should also test if some permutations of animals would make additional sparsity emerge.

On the RSP algorithm

The RSP algorithm is a simplification of the SP algorithm based on the view that the distance between a contributor and the animal it refers to affects the value of its contribution. Other factors may also affect these values. For instance, the number of equivalent contributors: if *x* contributors (instead of only one) are found by the same pedigree path (e.g. x half-sibs found only through the dam, without other connections to other ancestors), then the contribution of each of the *x* contributors is equal to the contribution that would have a single one divided by their number. Therefore, listing the different equivalent paths would produce a lower system to solve. Dividing a so-computed contribution by the number of equivalent contributors would return the proper contributions. In addition, when a lot of contributors are found within the first searched branches in regard to those found in next searched branches, they explain most of the relationships of the animal. The search may thus be stopped once a sufficient number of contributors are found.

On taking advantage of parallel computations

Implementations (CF, RCF and RSP) in which the vector of contributions is computed independently can be parallelized. The end of Chapter IV already discussed this point. Moreover, parallelization was implemented in Chapter V for the RSP algorithm: in average, the computation of the approximation was sped up by 3 when using 4 threads. Nevertheless, the process required more random-access memory because the variable that stores the inverse was explicitly duplicated and reduced by the end of the parallel region. In addition, computing time required by the step of update of this duplicated inverse is not negligible. Computing times were recorded for both operations (computation of the vector of contributions and update of the inverse) in our prototypic implementations: CPU times were close for both (the whole CPU-time is segmented in section *Discussion* of Chapter VI). Research efforts should therefore be put on modifying the equations that will further use $\mathbf{A}_{22}^{-1}$ in order to avoid update of the inverse, so that the only remaining computing bottle-neck would be the computations of contributions.

On the issue of parallelization, Meyer and Tier (2013) recently showed the interest of Graphical Processing Units (GPU) for computing the inverse of **G**. For approximation of the inverse of $\mathbf{A}_{22}$ by the RSP algorithm, it would also be relevant to test the use of GPU since the linear systems to solve independently have low sizes (e.g. 790 equations on average for the case of 75,000 animals outlined in Chapter V).

Perspectives for animal breeding

In the frame of an increasing number of genotypes

As mentioned in Chapter I, the growing number of genotyped animals increases the need for computations strategies (Hickey, 2013). These strategies include, in the case of **G**, approximations (Misztal et al., 2014) or higher computing resources (Meyer et al., 2013). For this reason the RSP algorithm is helpful, even if it returns an approximation rather than the real inverses. As shown in the discussion of Chapter IV, the number of contributors is no more coupled to the order of the matrix once the order goes beyond ~80,000 genotyped animals. We may thus expect the number of approximate contributors when using the RSP algorithm with $k = 4$ or $k = 5$ to follow the same trend. As a consequence, the RSP algorithm would become a very interesting way to avoid complete inversion of such very large $\mathbf{A}_{22}$.

In addition, the structure of the upcoming genotyped population will matter. As more families (both parents + offspring) will be genotyped in the future, $\mathbf{A}_{22}$ will become closer to **A**. The SP and RSP account for that by stopping exploration of pedigree if genotyped animals are reached in the two first searched branches (from animal to sire and to dam). However, it still has to integrate the fact that computations are reduced if full-sibs (with non-genotyped parents) are genotyped. In such case, it can be easily shown that the vector of contributions of an animal is the vector of contributions of the full-sib times a factor plus a contribution for the full-sib.

In the frame of improved genetic evaluation systems

As explained in Chapter I, two types of advances made national evaluations on a large scale (i.e. including several millions of animals, using an animal model rather than sire / sire-maternal-grand-sire models) possible, in the late 20th century: technological developments (availability of computer with higher resources, at a better prize) and methodological developments that eased setting up the evaluations (inversion techniques of relationship matrices, see Chapter II). Nowadays, the use of molecular information, as well as the growing size of number of evaluated traits, often requiring complex modelling, has increased the demand for both technological and methodological developments. The final output of this thesis is to ease computations linked to a particular case of a time-consuming operation (inversion of $\mathbf{A}_{22}$). Reduced time for achieving computations would mean that evaluations can be more frequent and may also become continuous (e.g. as proposed in Misztal et al., 1991). Such an improvement would be highly profitable for breeders: decision-making would indeed be more efficient as frequently updated breeding values would be available.

In the frame of multi-line evaluations

The SP algorithm, as well as other algorithms arising from this one (SPG and RSP), is useful in the frame of multi-line evaluations. Indeed, this algorithm only uses the genealogical information to find out which genotyped animals do contribute to other ones, i.e. for a given genotyped animal, which other genotyped animals will impact its breeding value. In multi-breeds evaluation, breeds can easily be identified what allows to easily break the genomic dependencies over breeds, if needed. In multi-lines, however, these dependencies are not that obvious. For that reason, exploring the pedigree with the SP algorithm is useful to set up the sets of contributors for each animal and avoid additional computations.

In the general frame of genomic selection

The development of the algorithms presented in this thesis is, beyond their use, framed in the field of genomic selection. Therefore, the research led in the frame of this thesis, as well as discussions with peers and attendances to scientific workshops and conferences, was helpful to understand the challenges and opportunities that can be brought to different breeding areas by genomic selection.

A major opportunity brought by the "genomics" era is the use of the molecular information from the SNP to maintain genetic diversity through a better estimation of relationship and inbreeding coefficients. Using that "gross information" represented by SNP, the research has mainly be focusing these last years on selection, i.e. on obtaining better estimation of breeding values by including an additional source of information. The use for conservation purposes has been neglected.

As outlined in an extension article (Faux et al., 2014), this extra-source of information can be useful when pedigrees are incomplete or missing. Some methodological developments, using algorithms described above and the ssGBLUP theory, showed that genomically-enhanced inbreeding coefficients could be easily computed for the dairy cattle population from Luxembourg (Faux and Gengler, 2014).

For this population, access to the genomic information is therefore critical. Since milk composition traits are routinely recorded in Luxembourg (Arnould et al., 2012), the breeders from Luxembourg have access to phenotypes for novel traits that can be an "exchange currency" to obtain genotypes. By sharing genotypes with them, external collaborators could obtain GEBV on these novel traits for their animals. As shown in a short study on this topic (Faux et al., 2012), the 440 currently genotyped animals are not yet enough to improve significantly the reliability of breeding values. However, by a joint effort of genotyping more local cows (keeping access to these genotypes) and a more proactive strategy to promote the phenotypes, this could generate the needed phenotypes.

Conclusions

We have proposed several algorithms that either set up, or approximate the sparsity pattern of $\mathbf{A}_{22}$ and **G**. Once set up, the (actual or approximate) sparsity pattern is used to compute the inverse of these matrices. If the actual sparsity pattern is known (only possible for the specific case of $\mathbf{A}_{22}$) then the inverse is computed by using the previously computed inverse to obtain the vector of contributions (Sherman-Morrison algorithm). If the sparsity pattern is approximate (as it was only the case for **G** using CF and RCF algorithms and as it was the case for $\mathbf{A}_{22}$ using different algorithms) then the inverse is computed by solving a linear system of lower size to obtain the vector of contributions. In both cases, the weighted product of the vector of contributions by its transpose updates the inverse.

It was shown that using approximated sparsity patterns greatly help to reduce the time needed for computation of approximation of $\mathbf{A}_{22}^{-1}$. Restricting the sparsity pattern to the first searched branches (RSP algorithm) returned the best approximations. If the number of searched branches is limited to 4, the approximation is up to 31 times faster than the actual inverse and does not greatly impact futures computations using the inverse. The approximation is even better if the number of searched branches is limited to 5, but then the approximation process is only 4 times faster than the actual inversion.

Conversely, it was shown that the approximations proposed for the sparsity pattern of **G** and for its inverse were not sufficient to not impact computations of GEBVs using the ssGBLUP procedure.

As the number of genotyped animals increases, approximations using the RSP algorithm should become more interesting as they take a reasonable advantage of the *a priori* knowledge of the population structure to set up an approximate sparsity pattern of $\mathbf{A}_{22}^{-1}$. This algorithm may be improved to be even more efficient: fewer computations for better approximations should be possible. Moreover, it offers potentialities for variance component estimation.

References

Arnould V., Gengler N. and Soyeurt H., 2012. Effect of the milk recording time on the genetic parameters of milk production and mid-infrared milk components in Luxembourg dairy cattle. *J. Dairy Sci.*, 95.

Arnould V., Reding R., Delvaux C., Bormann J., Gillon A. and Bertozzi C., 2014. Estimating daily yield and content of major fatty acids from single milking. In *Communications in Agricultural and Applied Biological Sciences*. 19th National Symposium on Applied Biological Sciences. Gembloux, Belgium.

Chow E., 2000. A priori sparsity patterns for parallel sparse approximate inverse preconditioners. *SIAM J. Sci. Comput.*, 21, 1804–1822.

Faux P., Arnould V., Soyeurt H. and Gengler N., 2012. Feasibility of genomic prediction of fatty acids composition in milk of dairy cattle of Luxembourg using single-step procedure. In 2012 ADSA-ASAS Joint Annual Meeting. Phoenix, AZ: Journal of Dairy Science, pp. 401–402.

Faux P., Bormann J., Reding R. and Gengler N., 2014. Importance stratégique de l'information généalogique et génomique pour une estimation correcte de la consanguinité. *Lëtzebuerger Ziichter*, 25–28.

Faux P. and Gengler N., 2014. Efficient computation of genomically-enhanced inbreeding coefficients. In *Communications in Agricultural and Applied Biological Sciences*. 19th National Symposium on Applied Biological Sciences. Gembloux, Belgium.

Hickey J.M., 2013. Sequencing millions of animals for genomic selection 2.0. *J. Anim. Breed. Genet.*, 130, 331–332.

Meyer K. and Tier B., 2013. Utility of Graphic Processing Units for dense matrix calculations in computing and inverting genomic relationship matrices. In *Proc. Assoc. Advmt. Anim. Breed. Genet*. Napier, New-Zealand, pp. 270–273.

Meyer K., Tier B. and Graser H.-U., 2013. Technical note: Updating the inverse of the genomic relationship matrix. *J. Anim. Sci.*

Misztal I., Lawlor T.J., Short T.H. and Wiggans G.R., 1991. Continuous Genetic Evaluation of Holsteins for Type. *J. Dairy Sci.*, 74, 2001–2009.

Misztal I. and Perez-Enciso M., 1993. Sparse matrix inversion for restricted maximum likelihood estimation of variance components by expectation-maximization. *J. Dairy Sci.*, 76, 1479–1483.

Misztal I., Legarra A. and Aguilar I., 2014. Using recursion to compute the inverse of the genomic relationship matrix. *J. Dairy Sci.*

Quaas R.L. and Pollak E.J., 1980. Mixed model methodology for farm and ranch beef cattle testing programs. *J. Anim. Sci.*, 51, 1277–1287.

Quaas R.L., 1988. Additive Genetic Model with Groups and Relationships. *J. Dairy Sci.*, 71, Supplement 2, 91–98.

Vandenplas J. and Gengler N., 2012. Comparison and improvements of different Bayesian procedures to integrate external information into genetic evaluations. *J. Dairy Sci.*, 95, 1513–1526.

List of Tables

Table 1 **Mean square differences (MSD) between real and approximated inverses of the genomic relationship matrix ($\mathbf{G}^{-1}$)** and percentages of elements that equaled zero (*t*0) in the lower part (excluding diagonals) of a triangular matrix used in approximating $\mathbf{G}^{-1}$ for 5 rounds of recursion of an approximation algorithm that defines the number of estimates for each line of the triangular matrix by the genomic relationship threshold (*p*) for 1,718 genotyped US Holsteins evaluated for final score in May 2009. 48

Table 2 **Mean square differences (MSD) between real and approximated inverses of the genomic relationship matrix ($\mathbf{G}^{-1}$)** and percentages of elements that equaled zero (*t*0) in the lower part (excluding diagonals) of a triangular matrix used in approximating $\mathbf{G}^{-1}$ for 8 rounds of recursion of an approximation algorithm that defines the maximum number of estimates to be 50 for all rounds for 1,718 genotyped US Holsteins evaluated for final score in May 2009. 49

Table 3 **Slopes of linear regression of GEBV based on approximated genomic relationship matrix (approximated GEBV) from 5 rounds of recursion** of an approximation algorithm that defines the number of estimates for each line of the triangular matrix by the genomic relationship threshold (*p*) on GEBV based on the actual genomic relationship matrix (real GEBV), coefficient of correlation (*r*) between approximated and real GEBV and standard deviations of approximated GEBV (*s*) for 1,718 genotyped US Holsteins evaluated for final score in May 2009. 50

Table 4 **Slopes of linear regression of GEBV based on approximated genomic relationship matrix (approximated GEBV) from 8 rounds of recursion** of an approximation algorithm that defines the maximum number of estimates to be 50 for all rounds on GEBV based on the actual genomic relationship matrix (real GEBV), coefficient of correlation (*r*) between approximated and real GEBV and standard deviations of approximated GEBV (*s*) for 1,718 genotyped US Holsteins evaluated for final score in May 2009. 50

Table 5 **Mean square differences (MSD) between real and approximated inverses of the part of the pedigree-based relationship matrix that represents relationships among genotyped animals ($\mathbf{A}_{22}^{-1}$)** and percentages of elements that equaled zero (*t*0) in the lower part (excluding diagonals) of a triangular matrix used in approximating $\mathbf{A}_{22}^{-1}$ for 5 rounds of recursion of an approximation algorithm that defines the number of estimates for each line of the triangular matrix by the genomic relationship threshold (*p*) for US Holsteins evaluated for final score in May 2009. 51

Table 6 **Mean square differences (MSD) between real and approximated inverses of the part of the pedigree-based relationship matrix that represents relationships among genotyped animals ($\mathbf{A}_{22}^{-1}$)** and percentages of elements that equaled zero (*t*0) in the lower part (excluding diagonals) of a triangular matrix used in approximating $\mathbf{A}_{22}^{-1}$ for 8 rounds of recursion of an approximation algorithm that defines the maximum number of estimates to be 50 for all rounds for US Holsteins evaluated for final score in May 2009. 51

Table 7 **Matrix $\mathbf{A}_{22}$ for the example of Figure 4** 65

Table 8 **Inverse triangular factor ($\mathbf{T}^{-1}$) of $\mathbf{A}_{22}$ for the example of Figure 4**. 65

Table 9 **Statistics of the population used** (dairy cows from Luxembourg) 69

Table 10 **Renumbered pedigree for the example of Figure 4.** 72
Table 11 **Sparsity pattern of $\mathbf{T}^{-1}$ for the example of Figure 4**. 73
Table 12 **Populations extracted for different sets of *selected* animals** 78
Table 13 **Estimated computational complexity[1] of Algorithm B** 80
Table 14 **Details on populations used in the study**: type (real or simulated), sizes and sparsity degrees. 94
Table 15 **Actual $\mathbf{T}^{-1}$ (below diagonal) and $\mathbf{A}_{22}^{-1}$ (diagonal and above diagonal)** for the example in Figure 13 96
Table 16 **Approximated $\mathbf{T}^{-1}$ (below diagonal) and $\mathbf{A}_{22}^{-1}$ (diagonal and above diagonal), when searching the pedigree for contributors located at maximum 2 branches**, for the example in Figure 13 99
Table 17 **Approximated $\mathbf{T}^{-1}$ (below diagonal) and $\mathbf{A}_{22}^{-1}$ (diagonal and above diagonal), when searching the pedigree for contributors located at maximum 4 branches**, for the example in Figure 13 99
Table 18 **Spearman rank correlations between EBV1 and EBV24, slope and intercept of the linear regression of EBV24 on EBV1, variances of EBV1 and variances of the difference between EBV1 and EBV24** for 4 different orders of $\mathbf{A}_{22}$ and 3 heritabilities per order. 103
Table 19 **Computing requirements of different algorithms for computing or approximating the inverse of $\mathbf{A}_{22}$**. Computing times (in minutes) and maximum required random-access memory (in Gbytes) for computing or approximating the inverse of 4 different orders of $\mathbf{A}_{22}$ using different algorithms, as well as the percentage of elements treated as zero in the computed (actual or approximated) inverse. 118
Table 20 **Quality of approximation of the inverse of $\mathbf{A}_{22}$ by different algorithms**. Spearman rank correlations (ρ) and reliabilities (REL) of actual and approximated EBV for 4 different numbers of genotyped animals using different algorithms, as well as the difference norm N between the actual and approximated inverses of $\mathbf{A}_{22}$ 119
Table 21 **Computing requirements of CF and RCF algorithms for computing or approximating the inverse of G and quality of approximation of its inverse.** Computing times (in seconds) and maximum required random-access memory (in Mbytes) for computing or approximating the inverse **G** for 2,427 animals, as well as the percentage of elements treated as zero in the approximated inverse and the Spearman rank correlations between GEBVs computed using approximated inverse and GEBVs computed using actual inverse, for all 17,677 animals in population (ρ_{ALL}) and for 2,427 genotyped animals (ρ_{GENO}) 123

List of Figures

Figure 1 **Pedigree relationships between 4 animals *s*, *d*, *j*, and *i*, where *s* and *d* are parents of *i* and *j* is any animal older than *i* and related to both *s* and *d* through ancestors.** 41

Figure 2 **Distribution of off-diagonal elements in the lower part (excluding diagonals) of a triangular matrix used in five rounds of recursion** of an algorithm that defines the number of estimates for each line of the triangular matrix by the genomic relationship threshold for US Holsteins evaluated for final score in May 2009 for approximation of the inverse of the genomic relationship matrix ($\mathbf{G}^{-1}$) and the part of the pedigree-based relationship matrix that represents relationships among genotyped animals ($\mathbf{A}_{22}^{-1}$); number of elements denoted as × between 10^{-1} and 1, ▲ between 10^{-2} and 10^{-1}, ● between 10^{-3} and 10^{-2}, ■ between 10^{-4} and 10^{-3} (element numbers of $<10^{-4}$ were considered to equal zero). 52

Figure 3 **Distribution of off-diagonal elements in the lower part (excluding diagonals) of a triangular matrix used in eight rounds of recursion** of an algorithm that defines the maximum number of estimates for each line of the triangular matrix to be 50 for all rounds for US Holsteins evaluated for final score in May 2009 for approximation of the inverse of the genomic relationship matrix ($\mathbf{G}^{-1}$) and the part of the pedigree-based relationship matrix that represents relationships among genotyped animals ($\mathbf{A}_{22}^{-1}$); number of elements denoted as × between 10^{-1} and 1, ▲ between 10^{-2} and 10^{-1}, ● between 10^{-3} and 10^{-2}, ■ between 10^{-4} and 10^{-3} (element numbers of $<10^{-4}$ were considered to equal zero). 52

Figure 4 **Small example: a population of 12 animals.** Genealogical tree for a population of 12 animals, partitioned in sub-populations 1 (*excluded*, circle) and 2 (*selected*, square). Alphabetical order gives the birth order. 66

Figure 5 **Pedigree extraction facts.** Generation by generation extraction of the pedigree of the *selected* population for three size scenarios (green: S4k; orange: S8k; blue: S12k): number of extracted animals (■) and proportion of *selected* animals in the extracted population (●), expressed as a percentage. Extraction went up to 23 generations for scenario S4k and up to 24 generations for scenarios S8k and S12k. 70

Figure 6 **CPU time required for inversion of $\mathbf{A}_{22}$ by two algorithms.** Elapsed CPU time required for inversion of $\mathbf{A}_{22}$ of three different sizes (green: 4000; orange: 8000; blue: 12000), computed using pedigrees with different numbers of extracted generations, by algorithms B (■) and A (▼). Red lines show upper and lower confidence intervals (99%; 20 repetitions). 74

Figure 7 **Effect of the depth of the pedigree on $\mathbf{A}_{22}^{-1}$.** Differences, as base-10 logarithm of the norm *N*, between $\mathbf{A}_{22}^{-1}$ based on a pedigree with a limited number of extracted generations and $\mathbf{A}_{22}^{-1}$ based on a fully extracted pedigree, for three size scenarios (green: S4k; orange: S8k; blue: S12k). 75

Figure 8 **CPU time required for determination of the sparsity pattern of $\mathbf{T}^{-1}$.** Elapsed CPU time required by the proposed algorithm for determination of the sparsity pattern of $\mathbf{T}^{-1}$, by number of extracted generations, for three size scenarios (green: S4k, orange: S8k and blue: S12k). Red lines show upper and lower confidence intervals (99%; 20 repetitions). 76

Figure 9 **Degree of sparsity of $\mathbf{T}^{-1}$.** Proportion of null entries in the lower triangular part of $\mathbf{T}^{-1}$ for different proportions (%) and numbers (thousands of animals) of *selected* animals in an extracted population. 79
Figure 10 **Average number of *contributors*.** Average number of *contributors* by line of $\mathbf{T}^{-1}$, for different numbers of *selected* animals (in thousands of animals). 79
Figure 11 **Procedure choice when running algorithm B.** Procedure choice (green: *EMPTY*; yellow: *LS*; blue: *PROD*) when running algorithm B, along all lines of $\mathbf{T}^{-1}$, for inversion of matrix $\mathbf{A}_{22}$ with a fully extracted pedigree, for three size scenarios [(a): S4k; (b): S8k ; (c): S12k]. 80
Figure 12 **Proportional use of different procedures in algorithm B.** Proportional (%) use of the three procedures in algorithm B (green: *EMPTY*; yellow: *LS*; blue: *PROD*), for different numbers of *selected* animals (in thousands of animals). 82
Figure 13 **Example genealogical tree of 12 animals**. Genotyped and non-genotyped animals are respectively tagged with squares and circles. Indexes show the number of branches to search starting from animal 12 in order to find its contributors. 95
Figure 14 **Distribution of absolute value of 2,249,997 contributions of a real genotyped population (P2)** among logarithmic classes (black bars) and percentage of contributions lower, in absolute value, than a certain logarithmic class (grey bars). 97
Figure 15 **Average contributions, on a logarithmic scale, in terms of the number of branches to search to connect the animal to its contributor, for a real genotyped population (P2).** Plain, dashed and dotted lines respectively show the average, the average plus 3 times the standard deviation and the maximum of absolute values of contributions found at each number of searched branches. 98
Figure 16 **Reliabilities (as squared correlations between true and estimated breeding values) of EBV1, EBV21, EBV22, EBV23 and EBV24 for 4 different orders of $\mathbf{A}_{22}$ and 3 different heritabilities (0.10, 0.30 and 0.50).** 101
Figure 17 **Estimated breeding values of 10,000 genotyped animals using an animal model (EBV1) vs. estimated breeding values using an equivalent model with approximated $\mathbf{A}_{22}^{-1}$ for which the maximum number of searched branches (*k*) was equal to 4 (EBV24).** Heritability was 0.10. The identity line is in grey. 102
Figure 18 **Computing times for approximated inverses of different orders of $\mathbf{A}_{22}$**, when the number of branches searched equals 4, for serial (plain line) and parallel (dashed line) implementations. 104
Figure 19 **Genealogical tree of 16 animals (10 genotyped, in squares)**, showing the actual contributors (in grey) of animal 16th. Identifiers give age order (1: oldest; 16: youngest). Left and right subscripts are respectively the additive relationship with animal 16th and the number of branches to reach the contributor from animal 16th. 111
Figure 20 **Genealogical tree of 16 animals (10 genotyped, in squares)**, showing the contributors (in grey) of animal 16th found using the CF algorithm (considering the animals sharing an additive relationship greater or equal to 0.10 as members of the same close-family). Identifiers give age order (1: oldest; 16: youngest). Left and right subscripts are respectively the additive relationship with animal 16th and the number of branches to reach the contributor from animal 16th. 112
Figure 21 **Genealogical tree of 16 animals (10 genotyped, in squares)**, showing the contributors (in gray) of animal 16th found using the RCF algorithm (considering two rounds of recursion with the respective thresholds: 0.10 and 0.005). Identifiers give age order (1: oldest; 16: youngest). Left and right subscripts are respectively the

additive relationship with animal 16th and the number of branches to reach the contributor from animal 16th. 113

Figure 22 **Genealogical tree of 16 animals (10 genotyped, in squares)**, showing the contributors (in gray) of animal 16th found using the SPg strategy (performing extraction of 2 generations of non-genotyped ancestors of genotyped animals). Identifiers give age order (1: oldest; 16: youngest). Left and right subscripts are respectively the additive relationship with animal 16th and the number of branches to reach the contributor from animal 16th. Animals and branches in grey are ignored.114

Figure 23 **Genealogical tree of 16 animals (10 genotyped, in squares)**, showing the contributors (in gray) of animal 16th found using the RSP algorithm (stopping the pedigree search when a maximum of 4 branches were explored). Identifiers give age order (1: oldest; 16: youngest). Left and right subscripts are respectively the additive relationship with animal 16th and the number of branches to reach the contributor from animal 16th. 115

Figure 24 **Computing time ratios** between computing the actual inverse using DKMXHF and the approximated inverse using 3 approximations (CF when the close-family threshold is set to 0.10 and RSP when the number of searched branches is limited to a maximum of 4 or 5), for 4 different orders of $\mathbf{A}_{22}$ 120

List of abbreviations

AXA	additive by additive epistasis
BLUP	best linear unbiased predictor
DNA	desoxyribonucleic acid
DXD	dominance by dominance epistasis
EBV	estimated breeding value
GBLUP	genomic BLUP
GEBV	genomically-enhanced breeding value
GIA	genetically identical animals
IBD	identical by descent
IBS	identical-by-state
LHS	left-hand side of an equation
MME	mixed model equations
MQTL	marked quantitative trait locus
MSD	mean square difference
QTL	quantitative trait locus
RAM	random-access memory
REL	reliability of estimated breeding values
SI	selection index
SNP	single nucleotide polymorphism
ssGBLUP	single-step genomic BLUP
TBV	true breeding value