Efficient Computations of Genomic Relationship Matrix and other Matrices Used in the Single-Step Evaluation

I. Aguilar[*†], I. Misztal[*], A. Legarra[‡] and S. Tsuruta[*]

Introduction

Genomic evaluations are currently performed using multiple step procedures (Hayes *et al.* (2009); VanRaden *et al.* (2009)). A typical evaluation requires 1) traditional evaluation with an animal model, 2) extraction of pseudo-observations such as deregressed evaluations, 3) estimation of genomic effects for genotyped animals, and possibly 4) combining the genomic index with traditional parent averages and breeding values (Hayes *et al.* (2009); VanRaden *et al.* (2009)). Genomic effects also can be estimated with a simple model that includes a genomic relationship matrix derived from genotypes and variances of the SNP marker effects (Nejati-Javaremi *et al.* (1997); VanRaden (2007)).

Recently, Misztal *et al.* (2009) proposed a single-step procedure (SSP) for genetic evaluation in which the pedigree-based relationship matrix is augmented by contributions from the genomic relationship matrix. Legarra *et al.* (2009) derived a joint relationship matrix based on pedigree and genomic relationships. An inverse of such joint relationship matrix allows straightforward application of the single-step approach in genetic evaluations (Aguilar *et al.* (2010); Christensen and Lund (2010)).

VanRaden (2008) presented methods to create genomic relationship matrices. The kernel of such methods involves a matrix multiplication operation. Specific subroutines for such operations are already available (Basic Linear Algebra Subroutines BLAS; Dongarra *et al.* (1988). An optimized version of BLAS subroutines (Automatically Tuned Linear Algebra Software, ATLAS; Whaley & Dongarra (1998)) allows taking into account features of a specific processor (memory speed and cache size) in several such subroutines.

Modifications of current software for genetic evaluations and variance component estimation to implement SSP (Aguilar *et al.* 2010) require inverses of the genomic relationship matrix and the relationship matrix for genotyped animals. The objectives of this research were to present efficient computing options to create such relationships matrices based on genomic markers and pedigree information, as well as their inverses.

Material and methods

Data. Creation of the genomic relationship matrix (G) was based on simulations. A matrix of incidences of SNP maker information (Z) was simulated for a panel of 40K SNPs, with values corresponding to gene content of the second allele (0, 1 and 2). Number of genotyped animals varied from 1,000 to 30,000. Pedigree-based relationship matrix (A_{22}) was

[*] Department of Animal and Dairy Science, University of Georgia, Athens, GA, United States

[†] Instituto Nacional de Investigación Agropecuaria, Las Brujas, Uruguay

[‡] INRA, UR631 SAGA, BP 52627, 31326 Castanet-Tolosan, France

constructed using a pedigree data of 9,100,106 US Holstein provided by Holstein USA Inc. (Brattleboro, VT).

Methods. Let the inverse of the joint relationship matrix based on both pedigree and genomic information (Aguilar *et al.* (2010); Christensen and Lund (2010)):

$$\mathbf{H}^{-1} = \mathbf{A}^{-1} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{G}^{-1} - \mathbf{A}_{22}^{-1} \end{bmatrix}$$

where A^{-1} is the inverse of the numerator relationship matrix, G^{-1} is the inverse of the genomic relationship matrix and A_{22}^{-1} is the inverse of the relationship matrix based on pedigree information corresponding to the genotyped animals.

Following VanRaden (2008), genomic relationships (G) were created as: G = ZZ'/k; where k is a scaling parameter and Z is an incidence matrix for SNP effects with elements equal to the number of copies of second allele and centered by the allele frequency. Computations of ZZ'/k were computed in Fortran 95 by several methods: 1) a simple three "do" loops, with centering the matrix Z through indirect memory access and scaling within loops (ORIG); 2) modification to optimize the indirect memory access (OPTM); and 3) OPTM plus reorganization of loops, and exclusion of the scaling operation from the main loop (OPTML). Having separate operations for matrix multiplication and scaling allows using general subroutines to compute ZZ'. Also, matrix multiplications of the form ZZ' were computed by OPTML, by the original BLAS subroutine DGEMM, and by their optimized versions as in ATLAS or in Intel's Math Kernel Library (MKL).

Matrix inversion was by a converted Fortran 95 code of a generalized inverse algorithm from the BLUPF90 package (Misztal *et al.* 2002) and by the LU factorization as implemented in LAPACK (Anderson *et al.* 1990). Such subroutines are available either in ATLAS or in MKL libraries.

Creation of the based-pedigree relationship matrix for genotyped animals (A_{22}) was evaluated using two methods. The first method was using the tabular method and the second was following formulas as presented in Misztal *et al.* (2009), which use the algorithm of Colleau (2002).

Computations. All programs were run on an Opteron 64-bit processor with a clock speed of 3.02 GHz and a cache size of 1Mbyte. Some programs were also run on a Xeon 64-bit processor with a clock speed of 3.5 GHz and a cache size of 6Mbytes.

Results and discussion

Results using the alternative loop codes are presented in Table 1. Computing time using ORIG was 10 times slower on the Opteron system, most likely because of its lower cache memory. Large improvement was achieved with the OPTM on the Opteron but not the Xenon system. An alternative explanation is that the Intel compiler that was used in the study was efficient in optimizing codes on Xenon but not on Opteron systems. Almost 4 times speedup was obtained on both computers with OPTML.

Figure 2 shows the computing time for OPTML, the BLAS subroutine for matrix multiplication (DGEMM), and its optimized version as in ATLAS libraries. The lowest computing time was with ATLAS-DGEMM subroutines. The performance of the OPTML shows a trend similar to ATLAS-DGEMM, but slightly slower.

Table 1: Computing time (m) for alternative codes for creation of the G matrix[1] on different machines.

		Algorithms		
Processor	Cache	Original	Memory optimized	Memory & loop optimized
Xeon 3.5 GHz	6 Mbyte	24	26	7
Opteron 3.02 GHz	1 Mbyte	265	59	17

[1] using 6,500 animals and 40K SNP markers

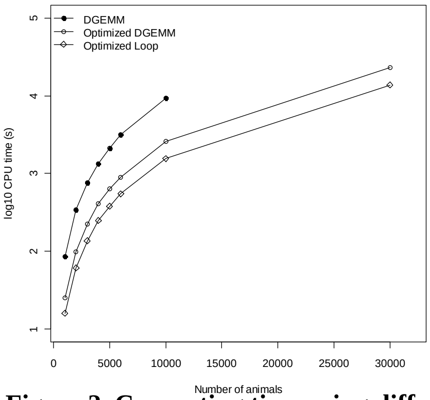

Figure 2. Computing time using different matrix multiplications algorithms

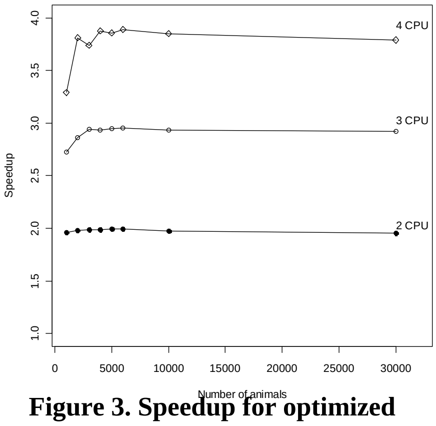

Figure 3. Speedup for optimized DGEMM for multiple processors using OpenMP

Matrix multiplication with large matrices requires optimization to fully utilize the cache memory. This operation requires fine tuning for specific system architectures. Simple modifications (OPTM) were successful in reducing the run time. Also, a simple rearrangement of the codes allow the compiler to do an automatic optimization (vectorization) and speed-up computations from 4 to 15 times, depending on the processor. However, using automatic code generation as in ATLAS-DGEMM results in a code that runs faster with no additional programming.

Optimized implementation of DGEMM in MKL allows parallel processing. Figure 3 shows the results for the optimized implementation of DGEMM in the MKL using up to 4 processors. The speedup with 3 processors and 5000 genotypes was 2.93, which was close to an ideal one.

Inversion of matrices for different number of genotypes was always faster using LAPACK compared with the generalized inverse. For the largest genomic relationship matrix (30,000 animals) the inversion took approximately 13 h with the generalized inverse but only 3.4 h using the optimized version of LAPACK. Further reductions in computing time could be obtained by parallel processing using OpenMP directives. A speed up of 3.35 was attained using four processors and the largest matrix.

Creation of the relationship matrix based on pedigree information for 6500 genotyped animals using the tabular method requires 311 s and 12.1 Gbyte of memory. The same computations with the Colleau method (2002) require 45 s and 322 Mbyte. The tabular method uses more memory as it requires storage for a dense matrix for all genotyped animals and their ancestors (approximately 57,000 individuals for 6,500 genotyped animals) while the Colleau method needs only a few vectors with dimension equal to the number of genotyped animals. Memory requirements for the tabular method can be reduced by splitting the pedigree file in several groups, but at cost of additional computations (VanRaden, personal communication, 2009).

Conclusion

We presented methods for efficient creation of matrices required for an efficient implementation of the single-step evaluation. Optimizations were by modifications of the existing code, using the efficient automatic optimization provided as open source software, or by commercial libraries. With all the optimizations, the creation of the genomic relationship matrix for 30,000 animals with 40K SNPs each, takes about 1 hr, with a similar time to obtain its inverse.

Acknowledgments

This study was partially funded by the Holstein Association USA Inc. and by AFRI grants 2009-65205-05665 and 2010-65205-20366 from the USDA NIFA Animal Genome Program. The authors thank P.M. VanRaden from Animal Improvement Programs Laboratory, USDA (Beltsville, MD), for providing the original software.

References

Aguilar, I., Misztal I., Johnson D. L. *et al*. (2010). *J. Dairy Sci.* 93: 743–752.

Anderson, E., Bai, Z., Dongarra, J. *et al (1990) In Proc ACM/IEEE Conference.*2-11

Christensen, O. F., Lund, M. S. (2010). *Genet. Sel. Evol*., 42–2

Colleau, J. J. (2002) *Genet. Sel. Evol.* 34:409–421

Dongarra, J., Croz, J., Hammarling, S. *et al*. (1988). *ACM Trans. Math. Softw.* 14, 1–17.

Hayes, B. J., P. J. Bowman, A. J. Chamberlain *et al*. (2009). *J. Dairy Sci*. 92:433–443.

Nejati-Javaremi, A., C. Smith, J. P. Gibson (1997). *J. Anim. Sci*. 75, 1738–1745.

Legarra, A., Aguilar, I., Misztal, I. (2009). *J. Dairy .Sci.,* 92:4656–4663.

Misztal, I., S. Tsuruta, T. Strabel, *et al*. (2002). In *Proc7th WCGALP*. 28–07

Misztal, I., A. Legarra, I. Aguilar. (2009). *J. Dairy Sci*. 92: 4648–4655.

VanRaden, P. M. (2007). *Interbull Bull*. 37, 33–36.

VanRaden, P. (2008). *J. Dairy Sci.*, 91:4414-4423.

VanRaden, P. M., Van Tassell, C. P., Wiggans, G. W. *et al*. (2009). *J. Dairy Sci*. 92:16–24.

Whaley, R. C., Dongarra, J. (1998). *In Proc ACM/IEEE Conference* (CD-ROM)