Acceleration of Computations in AI REML for Single-step GBLUP Models

***Y. Masuda**†, I. Aguilar‡, S. Tsuruta†, I. Misztal†.**
*Obihiro University of Agriculture and Veterinary Medicine, Obihiro, Japan,
†University of Georgia, Athens, USA, ‡Instituto Nacional de Investigación Agropecuaria, Canelones, Uruguay.

ABSTRACT: The objective of this study was to evaluate the advantage of the YAMS package over the FSPAK package in average-information (AI) REML for single-step GBLUP models. Data sets from broiler and Holsteins were used in this study. (Co)variance components were estimated with the AIREMLF90 program which could switch YAMS and FSPAK for sparse operations. The YAMS package used the BLAS and LAPACK libraries using all the 16 cores on CPU. For a single-trait model applied to the data contained over 15,000 genotyped animals, FSPAK took over 4 hours to finish the first 5 rounds while YAMS took 20 minutes. For a 4-trait model applied to the same data set, FSPAK failed in the sparse factorization while YAMS took 5 hours to finish the first 5 rounds. The use of YAMS can dramatically increase speed and stability of AIREMLF90 for single-step GBLUP models.

Keywords: single step GBLUP; supernodal methods; variance component estimation.

INTRODUCTION

Average-information (AI) REML (Gilmour, Thompson, and Cullis 1995) is the most popular method to estimate (co)variance components in animal breeding (Misztal 2008). The most time-consuming operation in AI REML is a calculation of traces in the first derivative of a likelihood function. Misztal and Pérez-Enciso (1993) demonstrated that the traces could be efficiently calculated using a sparse factorization (George and Liu 1981) followed by a sparse inversion (Erisman and Tinney 1975) of the left hand side of mixed model equations (LHS). A sparse inverse contains selected elements of an inverse matrix and can be computed with lower cost compared to a full inverse. Existing packages that compute the factor and sparse inverse, e.g. FSPAK (Pérez-Enciso, Misztal, and Elzo 1994), were designed for very sparse LHS and were slow to factorize and invert LHS when complex models were applied.

In genomic era, a regular animal model was extended to a single-step GBLUP model (ssGBLUP; Aguilar, Misztal, and Johnson et al. 2010). In this model, the elements of the additive relationship matrix for genotyped animals are replaced with the corresponding elements of the genomic relationship matrix (GRM) created with genomic markers. The ssGBLUP has been studied extensively for its predictability of genetic merits (Misztal, Aggrey, and Muir 2013). However, a few studies estimated genetic parameters using AI REML in ssGBLUP because of high computing cost of the sparse operations for LHS containing a large dense-matrix, i.e. the inverse of a GRM. The FSPAK would take long time to finish the computations or even crash especially when 5,000 or more genotyped animals were considered.

Masuda, Baba, and Suzuki (2014) developed a sparse package, YAMS. This software implements the supernodal methods (Ng and Peyton (1993); Campbell 1995), which exploit the dense structure in a matrix and can efficiently perform the sparse factorization and inversion for LHS containing dense blocks. All dense operations (multiplications and additions between matrices) are performed with the numerical libraries, BLAS and LAPACK (Anderson, Bai, and Bischof et al. 1999).

The objective of this study was to present the advantage of YAMS over the FSPAK package in AI REML for the ssGBLUP models. Computing options to achieve better performance on YAMS were also discussed.

MATERIALS AND METHODS

Data. Data from field records (Table 1) were used in this study. Models 1 through 4 were from commercial flocks in broiler. Additive genetic effects, residual effects, and other random effects (only applied to the first trait) were considered. Model 5 was a single-trait animal model applied to the final score in US Holsteins (Tsuruta, Misztal, and Klei et al. 2002). Prior to the estimation of parameters, each GRM was calculated with the preGSf90 program (Aguilar, Misztal, and Legarra et al. 2011) and stored in a file.

Software. (Co)variance components were estimated with the AIREMLF90 program (Misztal, Tsuruta, and Strabel et al. 2002), which implements the algorithm by Jensen, Mäntysaari, and Madsen et al. (1997). The program was modified to switch FSPAK and YAMS using an option in a parameter file. The default algorithm of ordering was MMD (Liu 1985) in FSPAK, and AMD (Amestoy, Davis, and Duff 1996) in YAMS. Detailed algorithms for the sparse operations employed in FSPAK and YAMS were described by Pérez-Enciso, Misztal, and Elzo (1994) and Masuda, Baba, and Suzuki (2014), respectively.

Benchmarks. All programs were compiled with the Intel Fortran Compiler (Intel Corporation, Santa Clara, CA). The YAMS package used the multithreaded versions of BLAS and LAPACK in the Math Kernel Library (MKL; Intel Corporation, Santa Clara, CA). The Analyses were performed on a computer with Intel Xeon E5-2689 CPU (2.9GHz), which had 16 cores. All the 16 cores were simultaneously used in BLAS and LAPACK. Computing time for the first and second rounds in AI REML was measured. The computing time was split into several parts: preparation (setting up equations), finding the ordering, symbolic factorization, numerical factorization, sparse inversion, and the remaining operations.

Table 1. Description of models and the number of nonzero elements in a factor of LHS.

		Number[1]			
Model	Traits	FIXED	PED	GENO	NZE
1	1	470	213,297	15,723	1.2×10^8
2	2	940	213,297	15,723	5.1×10^8
3	3	1,410	213,297	15,723	1.1×10^9
4	4	1,880	213,297	15,723	2.0×10^9
5	1	1,837	100,775	34,506	6.0×10^8

[1]FIXED = total levels of fixed effects, PED = animals in a pedigree file, GENO = genotyped animals, NZE = nonzero elements in a factor.

Additional experiments were conducted to investigate efficiency of the number of active cores to be used in sparse factorization and inversion with YAMS. Usefulness of another algorithm for ordering, METIS (Karypis and Kumar 1998), was also investigated.

RESULTS AND DISCUSSION

Comparisons between FSPAK and YAMS. Table 2 shows the computing time to finish the first and second round in AI REML using FSPAK and YAMS. FSPAK crashed during the factorization except Model 1. For Model 1, YAMS finished the computing about 10 and 14 times faster than FSPAK in the first and second round, respectively. To finish the first 5 rounds, FSPAK took over 4 hours while YAMS took 20 minutes. For the most complex model (Model 4), YAMS took 5 hours to finish the first 5 rounds. This computing time is reasonable to conduct a number of analyses with AI REML.

Table 2. Computing time (sec.) in first and second rounds of AI REML with FSPAK and YAMS.

	FSPAK		YAMS	
Model	1st	2nd	1st	2nd
1	3,857	2,808	397	199
2	NA[1]		1,103	595
3	NA		2,480	1,746
4	NA		5,115	3,381
5	NA		2,156	1,360

[1]FSPAK crashed during the sparse factorization.

Table 3 shows the detailed computing time for various operations in the first round with FSPAK and YAMS. In the numerical factorization and inversion for Model 1, YAMS was 20 and 29 times faster than FSPAK, respectively. Advantage of YAMS in the preparation step was from the AMD-ordering and symbolic factorization (Table 4). As the number of traits increased, more computing time tended to be spent for non-sparse operations.

Table 3. Computing time (sec.) of operations in first round of AI REML with FSPAK and YAMS.

		Operations[1]			
Package	Model	PREP	FAC	INV	MISC
FSPAK	1	1,006	717	2,077	51
YAMS	1	235	36	72	53
	2	668	86	150	199
	3	1,315	251	428	485
	4	2,466	556	975	1,117
	5	922	363	593	278

[1]PREP = creations of LHS and relationship matrices on memory, finding ordering, and symbolic factorization, NFAC = numerical factorization, SPINV = sparse inversion, MISC = other operations related to calculations for an AI matrix and gradients.

Table 4. Computing time (sec.) of operations in first round of AI REML for Model 1 with different computing options.

			Operations[1]		
Package	Cores	Ordering	O+S	FAC	INV
FSPAK	1	MMD	839	717	2,077
YAMS	1	METIS	81	1,075	3,262
	1	AMD	67	85	181
	2	AMD	67	60	127
	4	AMD	68	46	92
	8	AMD	67	40	79
	16	AMD	68	36	72

[1]O+S = finding ordering and symbolic factorization, FAC = numerical factorization, INV = sparse inversion.

Computing options. Table 4 presents the computing time for operations in the first round for Model 1 with different number of cores and the METIS-ordering algorithm. The AMD-ordering was more advantageous over the METIS-ordering. After ordering with AMD, all elements of the inverse of GRM in LHS were reordered and put together into a dense block in the bottom-right of the factor (Figure 1). The YAMS package can efficiently process such a large dense-block using BLAS and LAPACK. The METIS algorithm tended to spread nonzero elements around the factor and resulted in poor performance with YAMS. Even when only 1 core was used, YAMS was much faster than FSPAK. The computing time reduced as the number of active cores increased.

The performance of the numerical factorization and sparse inversion with YAMS depends largely on the optimization level of BLAS. Optimized BLAS libraries are available as proprietary software such as the Intel Math Kernel Library and the AMD Core Math Library (ACML; AMD, Sunnyvale, CA), or free software such as

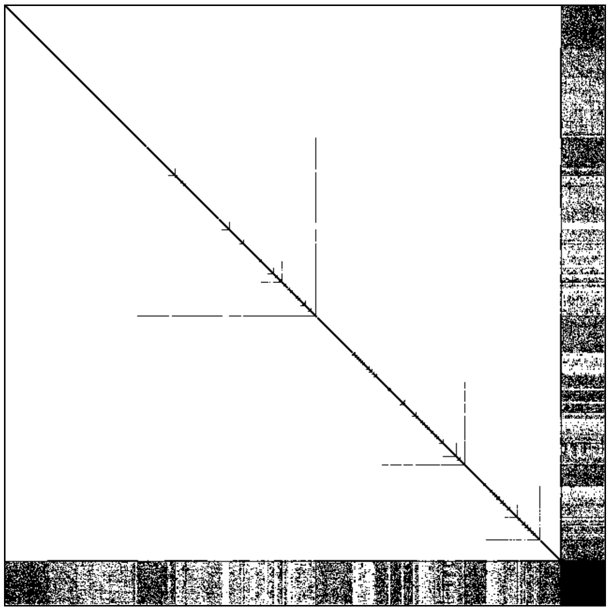
Figure 1: The distribution of nonzero elements in LHS ordered with AMD for the model 1 (a single-trait model)

OpenBLAS (www.openblas.net), GotoBLAS (www.tacc.utexas.edu/tacc-software/gotoblas2), and ATLAS (http://math-atlas.sourceforge.net).

Besides AIREMLF90, YAMS was also incorporated into the REMLF90 program with the EM REML and the BLUPF90 program to solve mixed model equations (Misztal, Tsuruta, and Strabel et al. 2002). These programs would accelerate estimation of (co)variance components as well as prediction of GEBV and estimation of prediction error variances in ssGBLUP.

CONCLUSION

The YAMS package dramatically improves speed and stability in (co)variance component estimation with AI REML for ssGBLUP models. The combination with AIREMLF90 was capable of estimating genetic parameters in a 4-trait model with a GRM from over 15,000 genotyped animals. Assuming cubic cost with the number of genotypes, analyses with 60,000 genotypes are possible within one day computing if adequate memory is available. The YAMS package is also useful for other operations requiring a factor or sparse inverse of LHS, such as calculation of accuracies for GEBV.

LITERATURE CITED

Amestoy, P. R., Davis, T. A., and Duff, I. S. (1996). *SIAM J. Matrix Anal. Appl.* 17:886–905.

Anderson, E., Bai, Z., Bischof, C. et al. (1999). LAPACK Users' Guide third edition. Soc. Ind. Appl. Math., Philadelphia, PA.

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

Aguilar, I., Misztal, I., Legarra, A. et al. (2011). *J. Anim. Breed. Genet.* 128:422–428.

Campbell, Y. E. (1995). PhD Thesis, University of Florida, Gainesville, FL.

Erisman, A. M., and Tinney, W. F. (1975). *Comm. ACM.* 18:177–179.

George, A., and Liu, J. W. H. (1981). Computer Solution of Large Sparse Positive Definite Systems. Prentice Hall Professional Technical Reference, NJ.

Gilmour, A. R., Thompson, R., and Cullis, B. R. (1995). *Biometrics* 51:1440-1450.

Jensen, J., Mäntysaari, E. A., Madsen, P. et al. (1997). *J. Ind. Soc. Agri. Stat.* 49:215–236.

Karypis, G., and Kumar, V. (1998). *SIAM J. Sci. Comput.* 20:359–392.

Liu, J. W. H. (1985). *ACM Trans. Math. Softw.* 11:141–153.

Masuda, Y., Baba, T., and Suzuki, M. (2014). *J. Anim. Breed. Genet.* (in press).

Misztal, I., and Perez-Enciso, M. (1993) *J. Dairy Sci.* 76:1479.

Misztal, I., Tsuruta, S., Strabel, T. et al. (2002). Proc 7th WCGALP, Communication No. 28-07.

Misztal, I. (2008). *J. Anim. Breed. Genet.* 125:363–370.

Misztal, I., Aggrey, S. E., and Muir, W. M. (2013). *Poult. Sci.* 92:2530–2534.

Ng, E. G., and Peyton, B. W. (1993). *SIAM J. Sci. Comput.* 14:761–769.

Pérez-Enciso, M., Misztal, I., and Elzo, M. A. (1994). Proc 5th WCGALP. 22:87–88.

Tsuruta, S., Misztal, I., Klei, L. et al. (2002). *J. Dairy Sci.* 85:1324–1330.