J. Dairy Sci. 92:2971–2975
doi:10.3168/jds.2008-1929

Technical note: Derivation of equivalent computing algorithms for genomic predictions and reliabilities of animal merit

I. Strandén*[1] and D. J. Garrick†‡
*MTT Agrifood Research Finland, FIN-31600 Jokioinen, Finland
†Department of Animal Science, Iowa State University, Ames 50014
‡Institute of Veterinary, Animal and Biomedical Sciences, Massey University, Palmerston North, New Zealand

ABSTRACT

Conventional prediction of dairy cattle merit involves setting up and solving linear equations with the number of unknowns being the number of animals, typically millions, multiplied by the number of traits being simultaneously assessed. The coefficient matrix has been large and sparse and iteration on data has been the method of choice, whereby the coefficient matrix is not stored but recreated as needed. In contrast, genomic prediction involves assessment of the merit of genome fragments characterized by single nucleotide polymorphism genotypes, currently some 50,000, which can then be used to predict the merit of individual animals according to the fragments they have inherited. The prediction equations for chromosome fragments typically have fewer than 100,000 unknowns, but the number of observations used to predict the fragment effects can be one-tenth the number of fragments. The coefficient matrix tends to be dense and the resulting system of equations can be ill behaved. Equivalent computing algorithms for genomic prediction were derived. The number of unknowns in the equivalent system grows with number of genotyped animals, usually bulls, rather than the number of chromosome fragment effects. In circumstances with fewer genotyped animals than single nucleotide polymorphism genotypes, these equivalent computations allow the solving of a smaller system of equations that behaves numerically better. There were 3 solving strategies compared: 1 method that formed and stored the coefficient matrix in memory and 2 methods that iterate on data. Finally, formulas for reliabilities of genomic predictions of merit were developed.
Key words: breeding value, computing method, dairy cattle, equivalent model

Practical computational considerations for the formulae published by Garrick (2007), VanRaden (2007, 2008), and reliabilities in VanRaden (2008) are given. A simple model for genomic prediction was assumed, where each individual has 1 observation for a trait, marker loci are biallelic, and unknown gene effects are additive. The model was based on the BLUP methods described by Meuwissen et al. (2001) and can be written as $\mathbf{y} = \mathbf{Xb} + \mathbf{Zu} + \mathbf{e}$, where $\mathbf{y}$ is n × 1 vector of observations, $\mathbf{Xb}$ is a vector of means, $\mathbf{Z}$ is a matrix containing 1 column to represent substitution effects for each marker locus; that column typically contains values 0, 1, or 2 to represent the number of copies of either one of the alleles; $\mathbf{u}$ is a vector of random chromosome fragment or SNP effects, and $\mathbf{e}$ is random residual vector. The random effects are assumed uncorrelated with variance matrices $\mathrm{Var}(\mathbf{u}) = \mathbf{I}\sigma_u^2$ and $\mathrm{Var}(\mathbf{e}) = \mathbf{R}$, where $\mathbf{I}$ is the identity matrix and $\mathbf{R}$ is diagonal in a single trait setting. Vector $\mathbf{u}$ contains the additive genetic effects that correspond to allele substitution effects for each marker or haplotype. This model can be readily extended to include observations based on progeny-test merit rather than phenotypes, other uncorrelated random effects in the model, or a general diagonal variance structure for the genetic effects $\mathbf{u}$, as is appropriate for BayesA or BayesB, wherein each locus can have its own variance (Meuwissen et al., 2001).

Assuming that variance components are known and the fixed effects $\mathbf{b}$ can be estimated, the genetic effects for each marker can be solved from the mixed model equations (Henderson, 1984)

$$\hat{\mathbf{u}} = \left(\mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z} + \lambda\mathbf{I}\right)^{-1}\mathbf{Z}'\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right), \qquad [1]$$

where scalar $\lambda = \dfrac{1}{\sigma_u^2}$. Define $\mathbf{a}$ to be the vector of breeding values or transmitting abilities for animals or sires, computed as a matrix-vector product by $\mathbf{a} = \mathbf{Zu}$, and $\mathbf{G} = \dfrac{\mathbf{ZZ}'}{\boldsymbol{\nu}_p}$ (VanRaden, 2008), where $\boldsymbol{\nu}_p = 2\Sigma p_i(1 - p_i)$ and

Received November 25, 2008.
Accepted January 23, 2009.
[1]Corresponding author: ismo.stranden@mtt.fi

the terms p_i and $(1 - p_i)$ are the frequencies of the alternate alleles at locus i. In that case, $\lambda = \lambda_a \boldsymbol{\nu}_p = \frac{2\Sigma p_i \left(1 - p_i\right)}{\sigma_a^2}$, where $\lambda_a = \frac{1}{\sigma_a^2}$ and σ_a^2 is total genetic variance. For more details on these parameters, see VanRaden (2008).

Equivalent Computations

Our derivations rely on the following identity (Householder, 1964), known as the matrix inversion lemma:

$$\left(\mathbf{Q} + \mathbf{BSC}'\right)^{-1} = \mathbf{Q}^{-1} - \mathbf{Q}^{-1}\mathbf{B}\left(\mathbf{S}^{-1} + \mathbf{C}'\mathbf{Q}^{-1}\mathbf{B}\right)^{-1}\mathbf{C}'\mathbf{Q}^{-1}, \quad [2]$$

where the matrices are of the order that allow the operations, and $\mathbf{S}$ and $\mathbf{Q}$ are nonsingular matrices. The matrix inversion lemma allows for derivation of equivalent computational algorithms relating breeding values $\mathbf{a}$ and genomic effects $\mathbf{u}$.

First, an equivalent model for solving the breeding values, $\mathbf{a}$, is developed. From the definition of a breeding value as the sum of genomic effects; that is, $\mathbf{a} = \mathbf{Zu}$, premultiply [1] by $\mathbf{Z}$, so that

$$\hat{\mathbf{a}} = \mathbf{Z}\left(\mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z} + \mathbf{I}\lambda\right)^{-1}\mathbf{Z}'\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right).$$

Noting that $\left(\mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z} + \mathbf{I}\lambda\right)^{-1} = \left(\mathbf{I}\lambda + \mathbf{Z}'\mathbf{R}^{-1}\mathbf{Z}\right)^{-1}$ and applying [2] to this inverse ($\mathbf{Q} = \mathbf{I}\lambda$, $\mathbf{B} = \mathbf{C} = \mathbf{Z}'$, $\mathbf{S} = \mathbf{R}^{-1}$) gives

$$\left(\mathbf{I}\lambda^{-1} - \lambda^{-1}\mathbf{Z}'\left(\mathbf{R} + \mathbf{ZZ}'\lambda^{-1}\right)^{-1}\mathbf{Z}\lambda^{-1}\right),$$

leading to the equality:

$$\hat{\mathbf{a}} = \mathbf{Z}\left(\mathbf{I}\lambda^{-1} - \lambda^{-1}\mathbf{Z}'\left(\mathbf{R} + \mathbf{ZZ}'\lambda^{-1}\right)^{-1}\mathbf{Z}\lambda^{-1}\right) \mathbf{Z}'\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right).$$

This can be simplified to

$$\hat{\mathbf{a}} = \lambda_a^{-1}\left(\mathbf{I} - \mathbf{G}\left(\mathbf{R}\lambda_a + \mathbf{G}\right)^{-1}\right)\mathbf{GR}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right). \quad [3]$$

Note that $\mathbf{ZZ}'\lambda^{-1} = \mathbf{G}\lambda_a^{-1}$. Applying [2] again ($\mathbf{Q} = \mathbf{G}$, $\mathbf{B} = \mathbf{C} = \mathbf{I}$, $\mathbf{S} = \mathbf{R}\lambda_a$), gives

$$\left(\mathbf{R}\lambda_a + \mathbf{G}\right)^{-1} = \left(\mathbf{G}^{-1} - \mathbf{G}^{-1}\left(\mathbf{R}^{-1}\lambda_a^{-1} + \mathbf{G}^{-1}\right)^{-1}\mathbf{G}^{-1}\right),$$

which used in [3] results in

$$\hat{\mathbf{a}} = \lambda_a^{-1}\left(\mathbf{I} - \mathbf{G}\left(\mathbf{G}^{-1} - \mathbf{G}^{-1}\left(\mathbf{R}^{-1}\lambda_a^{-1} + \mathbf{G}^{-1}\right)^{-1}\mathbf{G}^{-1}\right)\right) \mathbf{GR}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right),$$

provided the matrix $\mathbf{ZZ}'$ can be inverted; that is, at least as many loci as animals and no 2 animals have identical genotypes. This simplifies to

$$\hat{\mathbf{a}} = \left(\mathbf{R}^{-1} + \lambda_a\mathbf{G}^{-1}\right)^{-1}\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right), \quad [4]$$

which is the equivalent algorithm given by Garrick (2007).

Formula [4] can be arrived from a different model. Consider $\mathbf{y} = \mathbf{Xb} + \mathbf{Ia} + \mathbf{e}$, where $\mathbf{a} = \mathbf{Zu}$ is random vector of breeding values, and $\mathbf{e}$ is the random residual as defined earlier. Now, $\mathrm{Var}\left(\mathbf{a}\right) = \mathbf{ZZ}'\boldsymbol{\sigma}_u^2$ and $\mathrm{Var}\left(\mathbf{e}\right) = \mathbf{R}\boldsymbol{\sigma}_e^2$. Using [1] gives

$$\hat{\mathbf{a}} = \left(\mathbf{R}^{-1} + \left(\mathbf{ZZ}'\boldsymbol{\sigma}_u^2\right)^{-1}\right)^{-1}\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right)$$

and recognizing that $\mathbf{ZZ}'\boldsymbol{\sigma}_u^2 = \mathbf{G}\boldsymbol{\sigma}_a^2$, proof of [4] is straightforward.

Application of some more matrix algebra to [4] gives

$$\hat{\mathbf{a}} = \left(\mathbf{I} + \mathbf{R}\lambda_a\mathbf{G}^{-1}\right)^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right)$$

or

$$\hat{\mathbf{a}} = \mathbf{G}\left(\mathbf{G} + \mathbf{R}\lambda_a\right)^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right), \quad [5]$$

which was given in VanRaden (2008). Note that no inverse of $\mathbf{G}$ or $\mathbf{ZZ}'$ is needed to use [5].

The same steps can be done for $\hat{\mathbf{u}}$ in [1]. After applying [2] and some reordering as above to derive [3], we find

$$\hat{\mathbf{u}} = \lambda^{-1}\mathbf{Z}'\left(\mathbf{I} - \left(\mathbf{R}\lambda_a + \mathbf{G}\right)^{-1}\mathbf{G}\right)\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right).$$

This is somewhat different from [3] where premultiplication by matrix $\mathbf{Z}$ changed the $\mathbf{Z}'$ present here to be

G, and subsequently allowed some further reordering. Application of the matrix inversion lemma [2] the same way as to [3] gives

$$\hat{\mathbf{u}} = \lambda^{-1}\mathbf{Z}'\left(\mathbf{I} - \left(\mathbf{G}^{-1} - \mathbf{G}^{-1}\left(\mathbf{R}^{-1}\lambda_a^{-1} + \mathbf{G}^{-1}\right)^{-1}\mathbf{G}^{-1}\right)\mathbf{G}\right) \mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right),$$

which can be simplified to be

$$\hat{\mathbf{u}} = \lambda^{-1}\mathbf{Z}'\mathbf{G}^{-1}\left(\mathbf{R}^{-1}\lambda_a^{-1} + \mathbf{G}^{-1}\right)^{-1}\mathbf{R}^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right)$$

or (VanRaden, 2008)

$$\hat{\mathbf{u}} = \boldsymbol{\sigma}_a^{-2}\boldsymbol{\sigma}_u^2\mathbf{Z}'\left(\mathbf{G} + \mathbf{R}\lambda_a\right)^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right). \quad [6]$$

Computational Considerations

Formulas [5] and [6] involve a common term $\left(\mathbf{G} + \mathbf{R}\lambda_a\right)^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right)$ defined as $\mathbf{s}$, representing the solution to equation $\left[\mathbf{G} + \lambda_a\mathbf{R}\right]\mathbf{s} = \mathbf{y} - \mathbf{X}\hat{\mathbf{b}}$ [7].

Now with this definition, [5] can be rewritten to be $\hat{\mathbf{a}} = \mathbf{G}\mathbf{s}$ and [6] as $\hat{\mathbf{u}} = \boldsymbol{\sigma}_a^{-2}\boldsymbol{\sigma}_u^2\mathbf{Z}'\mathbf{s}$. Equation system [7] can be solved in many ways. Setting up the equations into memory can take considerable time although the solving time can be quick by direct methods (VanRaden, 2008).

The linear system [7] can be rewritten as

$$\left[\mathbf{Z}\mathbf{Z}'\lambda_a^{-1} + \mathbf{R}\right]\mathbf{s} = \lambda_a^{-1}\left(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}\right), \quad [8]$$

which allows a fast implementation using iteration on data (Schaeffer and Kennedy, 1986; Misztal and Gianola, 1987). In that context, every round of Jacobi or a preconditioned conjugate gradient method requires computing the product of a coefficient matrix by a vector. The original problem in [1] has matrix by vector $\mathbf{v}_p$ multiplication of form $\mathbf{Z}'\mathbf{Z}\mathbf{v}_p$ whereas the equivalent system in [8] has multiplication of form $\mathbf{Z}\mathbf{Z}'\mathbf{v}_n$, where p and n represent the order of the vectors; namely, the number of SNP loci and animals, respectively. When computations are done in the order from right to left; that is, $\mathbf{Z}'(\mathbf{Z}\mathbf{v}_p)$ or $\mathbf{Z}(\mathbf{Z}'\mathbf{v}_n)$, the number of multiplications is the same for computing $\mathbf{Z}'\mathbf{Z}\mathbf{v}_p$ or $\mathbf{Z}\mathbf{Z}'\mathbf{v}_n$. Thus, each iteration has computationally the same number of operations when solving unknowns in the original and equivalent system of equations by iteration on data despite the difference in the number of unknowns.

Computing performance of different iteration strategies can be approximated by considering the most intensive step. In Jacobi iteration and the preconditioned conjugate gradient method, the work is dominated by computing the product of the coefficient matrix and a vector. The number of floating point operations (**flops**) representing a multiplication, addition, and assignment (Golub and Van Loan, 1985), can be estimated for different iteration on data strategies. The strategies were I) coefficient matrix stored in memory (as in VanRaden, 2008); II) conventional iteration on data with matrix by vector product within iteration; III) the 2 step approach from right to left. Comparisons were restricted to flops to obtain the product $\mathbf{Z}\mathbf{Z}'\mathbf{v}_n$, recalling $\mathbf{Z}$ is a n by p matrix, with a row for each genotyped animal and a column for each biallelic locus. Let there be m iterations to solve for $\mathbf{s}$ in equation [7] or [8], with that solution premultiplied by $\mathbf{G}$, after convergence, to estimate merit of individual animals. The work to compute $\mathbf{G}\mathbf{s}$ is required in all the strategies and therefore ignored.

In strategy I, forming the $\mathbf{G}$ matrix in [6] requires making the product sum of matrices $\mathbf{Z}\mathbf{Z}'$. This can be computed in 1 of 2 ways, either as p outer products of each column of $\mathbf{Z}$; that is, $\mathbf{Z}\mathbf{Z}' = \sum_{i=1}^{p}\mathbf{z}_i\mathbf{z}_i'$, where $\mathbf{z}_i$ is column i of $\mathbf{Z}$ or as n^2 operations involving inner products of each row of $\mathbf{Z}$ with every column of $\mathbf{Z}'$ (i.e., row of $\mathbf{Z}$). The former approach requires access to $\mathbf{Z}$ by column; that is, the genotypes of all animals for one locus at a time, whereas the latter approach requires repeated access to all loci for pairs of animals. In either approach, pn^2 flops are performed. VanRaden (2008) reported that computing the $\mathbf{G}$ matrix increased effort by the number of markers times number of bulls squared, which supports the simple derivation. Once $\mathbf{G}$ has been formed and stored, on each iteration a product of an n by n matrix times an n vector is computed. Thus, n^2 flops are done each iteration. In total the first strategy requires $n^2\ (p + m)$ flops.

Strategy II, conventional iteration on data, is very similar to strategy I. However, elements of the coefficient matrix would be computed every iteration, rather than being stored. Computing $\mathbf{G}$ as $\mathbf{Z}\mathbf{Z}'$ via outer products each iteration and multiplying the elements as formed would require, over m iterations, $2pn^2m$ flops. The second method using outer products is inferior to the first, provided the coefficient matrix can be stored to memory, because $pm > p + m$ when p and m are greater than 2. Provided $\mathbf{Z}$ can be stored in memory, elements of $\mathbf{G}$ can be more efficiently computed as inner products of pairs of rows of $\mathbf{Z}$, with each element then multiplied by the relevant elements of the solution

Table 1. Relative performance and approximate number of floating point operations (flops) required to set up and iteratively solve equations[1] to estimate genomic merit for 3 iteration strategies considered in relation to the number of animals (n), the number of genotypes per animal (p), and the number of iterations (m)

	Strategy[2]		
	I	II	III
Storage	n^2	np	np
Setup flops[3]	n^2p	N/A[4]	N/A
Total flops for m iterations	$n^2(p + m)$	$2pn^2m$	$2pnm$
	Work[5] to setup and solve equations		
	Strategy I / Strategy III	Strategy II / Strategy III	Strategy III billion flops
(m = 1,000, p = 50,000 loci)			
n = 2,000 animals	1.0	2,000	200
n = 10,000	5.1	10,000	1,000
n = 25,000	12.8	25,000	2,500
(m = 5,000,[6] p = 50,000)			
n = 2,000	0.2	2,000	1,000
n = 10,000	1.0	10,000	5,000
n = 25,000	2.8	25,000	12,500

[1]Equations to solve are $\left[\mathbf{G} + \lambda_a \mathbf{R}\right]\mathbf{s} = \mathbf{y} - \mathbf{X}\hat{\mathbf{b}}$ with $\mathbf{G} = \mathbf{ZZ}'$.

[2]Strategy I forms and stores the coefficient matrix before iteration, strategy II uses iteration on data to repeatedly create and discard elements of the coefficient matrix, whereas strategy III takes advantage of multiplicative factors of the major component of the coefficient matrix. Strategy I forms **G** and stores the coefficient matrix for iteration; strategy II stores **Z** and accesses it each iteration by column to form contributions to elements of **G** by outer product that are used then discarded; strategy III stores **Z** and accesses it each iteration by column to form contributions to elements of $\mathbf{Gs} = \mathbf{ZZ}'\mathbf{s}$ in 2 steps by calculating for each column i=1,...,p, of Z scalar $t_i = \mathbf{z}_i'\mathbf{s}$ followed by ith vector contribution $\mathbf{z}_i t_i$ to the product **Gs**, where $\mathbf{z}_i$ is column i of Z.

[3]A floating point operation is a multiplication, addition, and assignment (Golub and Van Loan, 1985).

[4]N/A = not applicable.

[5]Work is presented as the approximate flops in strategy III, whereas it is the ratio of work in strategy I or II relative to the work in strategy III. Only the flops required in the dominant matrix vector computations were quantified.

[6]Preconditioned conjugate methods are guaranteed to converge in fewer iterations than the number of equations and seldom require as many as 2,000 iterations. This number of iterations allows for circumstances where analyses are conducted across several traits; for example, 10 traits at 500 iterations per trait.

vector requiring $(p + 1)n^2m$ flops or approximately half the work of the outer product approach for large p.

Strategy III, the 2-step approach from right to left, typically requires the smallest number of flops in current scenarios. Product $\mathbf{Z}(\mathbf{Z}'\mathbf{v}_n)$ is calculated in 2 steps requiring $2np$ flops in each iteration. This strategy is similar in computational load to VanRaden (2008) although there the original equation system was iterated. Thus, the experience of a linear increase in computing time observed in VanRaden (2008) can be supported by this analysis. In total, the third strategy needs $2npm$ flops.

The relative performance of these strategies is sensitive to the number of animals with genotypes, the number of loci per animal, and the number of iterations required to solve the equations. The approximate numbers of flops required for some realistic values of these parameters in cattle evaluation settings are in Table 1. Strategy II is always at least an order of magnitude more work than the other strategies for the scenarios considered. Strategy I gets better relative to strategy III as the number of loci increases (results not shown) or the number of iterations increases. More iterations can be the result of analyzing numerous traits for the same animals. Strategy I gets worse relative to strategy III as the number of animals increases. Strategy III is typically less work than strategy I, unless many iterations are required for scenarios with few animals genotyped and many loci per animal. There is little practical difference between 50,000 and 300,000 loci (results not shown) for the 2,000 to 25,000 genotyped animals considered in the scenarios shown in Table 1.

VanRaden (2008) reported that the computing time of the original system by iteration on data was more than the computing time by a direct method where the coefficient matrix was stored in memory although the number of bulls was more than twice the number of iterations. This contradicts the simple analysis; however,

VanRaden (2008) solved different equation systems by different methods: the original system was solved by an iterative method, whereas the equivalent model used a direct method.

Solving properties of equation system [8] are often more favorable than the original system [1]. Condition number of the coefficient matrix in [8] can be expected to be lower than in [1]. A low condition number means a well-conditioned problem, which leads to fewer iterations to achieve the same accuracy in solutions for an iterative method (Golub and Van Loan, 1985). The condition number of the original system can be expected to become worse when more highly correlated SNP effects are included into the model. On the other hand, adding more SNP effects does not much influence the correlation structure between animals in the matrix $\mathbf{ZZ}'$. Thus, fewer iterations are expected for an iterative method using equation [8] than for the original system of equation [1].

In practice, methods storing the coefficient matrix in memory have an advantage over iteration on data methods. Computing the $\mathbf{G}$ matrix needs to be done only once when solving several traits (VanRaden, 2008). Consequently, the computational overhead of building the coefficient matrix in the memory becomes diluted across the number of traits.

Reliabilities

Reliabilities for breeding values are often needed. Consider model $\mathbf{y^*} = \mathbf{a} + \mathbf{e}$, where $\mathbf{y^*}$ is vector of observations corrected for fixed effects ($\mathbf{y^*} = \mathbf{y} - \mathbf{X}\hat{\mathbf{b}}$), $\mathbf{a} = \mathbf{Zu}$ is a random vector of breeding values, and $\mathbf{e}$ is a random residual as defined earlier, with $\text{Var}(\mathbf{a}) = \mathbf{ZZ}'\boldsymbol{\sigma}_u^2$ and $\text{Var}(\mathbf{e}) = \mathbf{R}$. In general, reliability of animal i is $r_i^2 = 1 - \left(PEV_i \middle/ \boldsymbol{\sigma}_G^2\right)$, where PEV_i is the prediction error variance of the breeding value of animal i, and $\boldsymbol{\sigma}_G^2$ is genetic variance. Prediction error variance for animal i is the ith diagonal element in matrix $\left(\mathbf{R}^{-1} + \left(\mathbf{ZZ}'\boldsymbol{\sigma}_u^2\right)^{-1}\right)^{-1}$, and the genetic variance is the ith diagonal element in matrix $\mathbf{ZZ}'\boldsymbol{\sigma}_u^2$. The formula can be simplified.

Consider prediction error variance matrix

$$\begin{aligned} PEV &= \left(\mathbf{R}^{-1} + \left(\mathbf{ZZ}'\right)^{-1}\lambda\right)^{-1} \\ &= \left(\mathbf{R}^{-1} + \mathbf{G}^{-1}\lambda_a\right)^{-1}. \end{aligned}$$

The genetic variance matrix is

$$\begin{aligned} \text{Var}(\mathbf{a}) &= \mathbf{ZZ}'\lambda^{-1} \\ &= \mathbf{G}\lambda_a^{-1}. \end{aligned}$$

Thus, a formula for reliability of animal i is (VanRaden, 2008)

$$\left\{\mathbf{G} - \lambda_a\left(\mathbf{R}^{-1} + \mathbf{G}^{-1}\lambda_a\right)^{-1}\right\}_i \Big/ \left\{\mathbf{G}\right\}_i \quad [9]$$

where notation $\left\{\mathbf{M}\right\}_i$ refers to the diagonal element i of matrix $\mathbf{M}$. The numerator in [9] can be rewritten (VanRaden, 2008)

$$\begin{aligned} \mathbf{G} - \lambda_a\mathbf{G}\left(\mathbf{G} + \mathbf{R}\lambda_a\right)^{-1}\mathbf{R} &= \mathbf{G}\left(\mathbf{I} - \lambda_a\left(\mathbf{G} + \mathbf{R}\lambda_a\right)^{-1}\mathbf{R}\right) \\ &= \mathbf{G}\left(\mathbf{G} + \mathbf{R}\lambda_a\right)^{-1}\mathbf{G}. \end{aligned} \quad [10]$$

If [10] is calculated by direct inversion that derives all elements of the inverse, not just the diagonal, the inverse $\left(\mathbf{G} + \mathbf{R}\lambda_a\right)^{-1}$ or its factor can be used to directly solve breeding values in [7].

CONCLUSIONS

Equivalent computing algorithms allow the solving of unknowns in less computing time and with less memory than needed by the conventional approaches. When several traits are analyzed or reliabilities are needed, iteration on data techniques can be inferior to constructing the coefficient matrix in memory and reusing it.

REFERENCES

Garrick, D. J. 2007. Equivalent mixed model equations for genomic selection. J. Dairy Sci. 90(Suppl. 1):376. (Abstr.)

Golub, G. H., and C. F. Van Loan. 1985. Matrix Computations. 476 pp. The John Hopkins University Press, Baltimore, MD.

Henderson, C. R. 1984 Applications of Linear Models in Animal Breeding. University of Guelph, Guelph, Ontario, Canada.

Householder, A. S. 1964. The theory of matrices in numerical analysis. Republished and reissued by Dover Publications, New York, 2006.

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

Misztal, I., and D. Gianola. 1987. Indirect solution of mixed model equations. J. Dairy Sci. 70:716–723.

Schaeffer, L. R., and B. W. Kennedy. 1986. Computing strategies for solving mixed model equations. J. Dairy Sci. 69:575–579.

VanRaden, P. M. 2007. Genomic measures of relationship and inbreeding. Interbull Bull. 37:33–36.

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