Bpop: an efficient program for estimating base population allele frequencies in single and multiple group structured populations

Ismo Strandén and Esa A. Mäntysaari

Natural Resources Institute Finland (Luke), Jokioinen, Finland

e-mail: ismo.stranden@luke.fi

Base population allele frequencies (AF) should be used in genomic evaluations. A program named Bpop was implemented to estimate base population AF using a generalized least squares (GLS) method when the base population individuals can be assigned to groups. The required dense matrix products involving $(\mathbf{A}_{22})^{-1}\mathbf{v}$ were implemented efficiently using sparse submatrices of $\mathbf{A}^{-1}$, where **A** and $\mathbf{A}_{22}$ are pedigree relationship matrices for all and genotyped animals, respectively. Three approaches were implemented: iteration on pedigree (IOP), iteration in memory (IM), and direct inversion by sparsity preserving Cholesky decomposition (CHM). The test data had 1.5 million animals genotyped using 50240 markers. Total computing time (the product $(\mathbf{A}_{22})^{-1}\mathbf{1}$) was 53 min (1.2 min) by IOP, 51 min (0.3 min) by IM, and 56 min (4.6 min) by CHM. Peak computer core memory use was 0.67 GB by IOP, 0.80 GB by IM, and 7.53 GB by CHM. Thus, the IOP and IM approaches can be recommended for large data sets because of their low memory use and computing time.

Key words: allele frequency, computer software, genomic evaluation, relationship matrix

Introduction

Base population allele frequencies (AF) are recommended to be used in computation of the genomic relationship matrix used in the genomic evaluation (VanRaden 2008) and estimation of the genomic compliant relationship matrix among metafounders (Garcia-Baccino et al. 2017). For example, the single-step GBLUP method assumes that the pedigree and genomic based relationship matrices have the same scale and base population definition (Christensen et al. 2012, Mäntysaari et al. 2020) which may be achieved by using base population AF. Garcia-Baccino et al. (2017) reviewed methods for the estimation of base population AF when the pedigree of genotyped animals is known for single and multiple group populations. In two of the presented three methods, i.e., generalized least squares (GLS) and maximum likelihood (ML), computationally the most challenging step is the product $\mathbf{v} = (\mathbf{A}_{22})^{-1}\mathbf{s}$ where vector **s** is a function of marker genotypes and $\mathbf{A}_{22}$ is the pedigree based relationship matrix between the genotyped animals. When the number of genotyped animals is large, the need to invert a large $\mathbf{A}_{22}$ may make the method computationally unfeasible because this matrix is often dense.

An alternative to the brute force inversion of a large $\mathbf{A}_{22}$ matrix is to solve the vector **v** in linear system of equations $\mathbf{A}_{22}\mathbf{v} = \mathbf{s}$, e.g., by an iterative method such as preconditioned conjugate gradient (PCG) iteration. However, when the number of genotyped animals is large, storing and using the dense $\mathbf{A}_{22}$ matrix will slow down the computations considerably. Thus, solving by an iterative method may take too long. Strandén et al. (2017) presented an alternative computational approach for the GLS method (McPeek et al. 2006) where explicit calculation of $(\mathbf{A}_{22})^{-1}$ is avoided. They used equality $(\mathbf{A}_{22})^{-1}\mathbf{s} = (\mathbf{A}^{22} - \mathbf{A}^{21}(\mathbf{A}^{11})^{-1}\mathbf{A}^{12})\mathbf{s}$ where $\mathbf{A}^{ij}$, i,j = 1,2,are submatrices of $\mathbf{A}^{-1}$ which are often sparse, and numbers 1 and 2 refer to the non-genotyped and the genotyped animals, respectively. Product $\mathbf{A}^{ij}\mathbf{s}$ can be calculated either using pedigree information without making matrix $\mathbf{A}^{ij}$ (Henderson 1976, Quaas 1976) or after $\mathbf{A}^{ij}$ has been computed using pedigree information. Computationally the most challenging task is the product $(\mathbf{A}^{11})^{-1}\mathbf{x}$ where $\mathbf{x} = \mathbf{A}^{12}\mathbf{s}$. This product requires using either an iterative or a sparse matrix solver (Strandén et al. 2017).

Aldridge et al. (2018) compared two approaches to compute the GLS method estimates. One method used the direct inversion of the $\mathbf{A}_{22}$ matrix approach and another used the sparse matrix solver approach as in Strandén et al. (2017). Aldridge et al. (2018) estimated base population AF for 1670 markers using genotypes from 100 078 animals. According to their results, the direct $\mathbf{A}_{22}$ matrix inversion approach took more than 1 day but the algorithm using sparse matrices took about 49 seconds. The direct inversion approach needed 118.5 GB of memory, but the sparse matrix approach needed only 1.3 GB of memory.

Manuscript received March 2020

Strandén et al. (2017) used the sparse matrix approach in the iterative PCG method to solve single-step GBLUP (Aguilar et al. 2010, Christensen and Lund 2010) where the product $(\mathbf{A}_{22})^{-1}\mathbf{d}$ was calculated in every iteration of the PCG algorithm. They presented and tested three approaches and found the sparse matrix solver approach to be the fastest (Strandén et al. 2017). In single-step GBLUP, the product $(\mathbf{A}_{22})^{-1}\mathbf{d}$ needs to be computed every iteration using a different **d** vector. Note that the product $(\mathbf{A}_{22})^{-1}\mathbf{d}$ above is due to the mixed model equations of ssGBLUP, and not due to PCG iterations used in the GLS method for the base population AF estimation in the current study. In the GLS method, however, the product $\mathbf{v} = (\mathbf{A}_{22})^{-1}\mathbf{s}$ needs to be computed only once and the same **v** vector is used in every AF calculation. Thus, when the number of AF to be estimated increases, time to compute $\mathbf{v} = (\mathbf{A}_{22})^{-1}\mathbf{s}$ can become less significant. Because the two other approaches, named IOP and IM (see "Solving approaches in Bpop" in Material and Methods) in Strandén et al. (2017) were computationally simpler than the sparse matrix solver approach called CHM, and needed less computing memory, the simpler approaches may be better in AF estimation for large data sets having millions of genotyped animals.

In this study, we describe Bpop program for estimation of base population AF using the GLS method. We consider one and multiple group populations in AF estimation. We implement and compare the three algorithms, called IOP, IM and CHM, presented in Strandén et al. (2017). We use a large genomic data from cattle to illustrate performance of the developed approaches.

Material and methods

Base population allele frequency estimation

Base population AF in a single population can be estimated using the GLS model in McPeek et al. (2004). Let **M** be an *n* by *m* genotype matrix for *n* individuals and *m* SNP (single-nucleotide polymorphism) markers. Genotype is coded 0 for homozygote AA, 1 for the heterozygote AB, and 2 for the homozygote BB. For each marker *i*, the approach uses a GLS model where the only fixed effect is the unknown general mean μ_i:

$$\mathbf{m}_i=\mathbf{1}_n\mu_i+\mathbf{e},$$

where $\mathbf{m}_i$ is marker genotype column *i* in **M**, $i = 1,\ldots m$, $\mathbf{e}\sim(\mathbf{0}, \mathbf{A}_{22}\sigma^2)$, $\mathbf{A}_{22}$ is pedigree relationship matrix of the genotyped animals, and σ^2 is common variance. The variance of gene content σ^2 is assumed to be the same for all genotypes and need not be known (e.g. Garcia-Baccino et al. 2017). Solving this GLS model gives estimator

$$\hat{\mu}_i=\mathbf{1}_n'(\mathbf{A}_{22})^{-1}\mathbf{m}_i/(\mathbf{1}_n'(\mathbf{A}_{22})^{-1}\mathbf{1}_n). \qquad [1]$$

Base population AF is half of this, i.e., $\hat{p}_i=\frac{1}{2}\hat{\mu}_i$.

This approach can be generalized for an admixed population using a genetic groups model having *r* groups (Garcia-Baccino et al. 2017). Let **Q** be an *n* by *r* matrix of fractions of genetic groups represented in individuals where each row sums to one. The **Q** matrix can be calculated using pedigree information where offspring group proportions are calculated as mean of the parent group proportions and unknown parent is assigned to a group. The GLS model can be presented as

$$\mathbf{m}_i=\mathbf{Q}\boldsymbol{\mu}_i+\mathbf{e},$$

where $\boldsymbol{\mu}_i$ is an r by 1 vector of unknown general means of the groups. Note that this GLS model assumes that the variance of gene content is the same in all groups (Garcia-Baccino et al. 2017). Estimator to the base population AF of the groups for marker *i* are solutions

$$\hat{\mathbf{p}}_i=\frac{1}{2}(\mathbf{Q}'(\mathbf{A}_{22})^{-1}\mathbf{Q})^{-1}\mathbf{Q}'(\mathbf{A}_{22})^{-1}\mathbf{m}_i, \qquad [2]$$

which is an *r* by 1 vector. In some cases, it is useful to estimate the AF using the observed genotypes, i.e., ignore the pedigree structure. This can be done using simplified equations: $\hat{p}_i=\mathbf{1}_n'\mathbf{m}_i/(2n)$ and $\hat{\mathbf{p}}_i=\frac{1}{2}(\mathbf{Q}'\mathbf{Q})^{-1}\mathbf{Q}'\mathbf{m}_i$. We call these least squares (LS) estimators. Note that the computation of observed genotype data AF neglects the pedigree-based covariance structure between the genotyped animals.

Computational algorithms

Computationally, the most challenging terms in formulas [1] and [2] are due to $(\mathbf{A}_{22})^{-1}$, particularly when *n* is large. Note that the computations involving $(\mathbf{A}_{22})^{-1}$ need to be done for each group and marker, i.e., in total *mr* times. Fortunately, all terms involving $(\mathbf{A}_{22})^{-1}$ are independent from the marker genotypes in **M** and can be factored as a common multiplier which is calculated only once.

For the single group case, formula [1] can be written as $\hat{p}_i = \mathbf{cm}_i$ where $\mathbf{c}=\frac{1}{2}\mathbf{1}_n^{'}(\mathbf{A}_{22})^{-1}/\left(\mathbf{1}_n^{'}(\mathbf{A}_{22})^{-1}\mathbf{1}_n\right)$ is a row vector of length *n*. Note that the **c** vector can be written

$$\mathbf{c}=\frac{1}{2}\mathbf{f}'\left(\mathbf{1}_n^{'}\mathbf{f}\right)^{-1} \qquad [3]$$

where $\mathbf{f}=(\mathbf{A}_{22})^{-1}\mathbf{1}_n$ is an *n* by 1 vector. For the multiple group case, formula [2] can be written $\hat{\mathbf{p}}_i=\mathbf{Cm}_i$ where $\mathbf{C}=\frac{1}{2}(\mathbf{Q}'(\mathbf{A}_{22})^{-1}\mathbf{Q})^{-1}\mathbf{Q}'(\mathbf{A}_{22})^{-1}$ is an *r* by *n* matrix. Like for the **c** vector, the **C** matrix can be expressed

$$\mathbf{C}=\frac{1}{2}(\mathbf{Q}'\mathbf{F})^{-1}\mathbf{F}' \qquad [4]$$

where $\mathbf{F}=(\mathbf{A}_{22})^{-1}\mathbf{Q}$ is an *n* by *r* matrix. Note that for the multiple group case, $(\mathbf{A}_{22})^{-1}$ is needed in *r* multiplications due to the *r* columns in **Q**. Thus, solving each of the AF requires multiplication of the marker vector $\mathbf{m}_i$ by a constant vector **c** or matrix **C**. An alternative approach is to calculate all AF simultaneously by $\hat{\mathbf{p}}=\mathbf{cM}$ and $\hat{\mathbf{P}}=\mathbf{CM}$ for the single and multiple group cases, respectively.

Computations of the **f** vector and the **F** matrix need the inverse matrix $(\mathbf{A}_{22})^{-1}$. As described in Strandén et al. (2017), the computations involving this matrix can be made efficiently using submatrices of the inverse of the full relationship matrix **A**. Animals can be assigned to two sets: set 1 has the non-genotyped ancestors of genotyped animals, set 2 has the genotyped animals. Then, the relationship matrix **A** and its inverse $\mathbf{A}^{-1}$ can be expressed by submatrices referring to the two sets:

$$\mathbf{A}=\begin{bmatrix}\mathbf{A}_{11} & \mathbf{A}_{12}\\ \mathbf{A}_{21} & \mathbf{A}_{22}\end{bmatrix} \text{ and } \mathbf{A}^{-1}=\begin{bmatrix}\mathbf{A}^{11} & \mathbf{A}^{12}\\ \mathbf{A}^{21} & \mathbf{A}^{22}\end{bmatrix}.$$

According to the rules of matrix algebra, inverse of the $\mathbf{A}_{22}$ matrix can be expressed using elements of the inverse matrix $\mathbf{A}^{-1}$:

$$(\mathbf{A}_{22})^{-1}=\mathbf{A}^{22}-\mathbf{A}^{21}\left(\mathbf{A}^{11}\right)^{-1}\mathbf{A}^{12}. \qquad [5]$$

Computations involving sub-matrices of $\mathbf{A}^{-1}$ can be made by using the pedigree list (Henderson 1976, Quaas 1976). Note that while the whole population pedigree can hold millions of animals, the matrix **A** can be restricted to include only genotyped animals and their ancestors. For example, non-genotyped progeny of these do not contribute information to AF. This reduction of the **A** matrix to genotyped animals and their ancestors can reduce substantially the amount of computations.

Consider calculating $\mathbf{v}=(\mathbf{A}_{22})^{--1}\mathbf{s}$ where **s** is an *n* by 1 vector. According to formula [5], we need to compute $\mathbf{v}=(\mathbf{A}_{22})^{-1}\mathbf{s}=(\mathbf{A}^{22}-\mathbf{A}^{21}(\mathbf{A}^{11})^{-1}\mathbf{A}^{12})\mathbf{s}$. This calculation can be split into the following three steps (Strandén et al. 2017):

1) $\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix}=\begin{bmatrix}\mathbf{A}^{12}\\ \mathbf{A}^{22}\end{bmatrix}\mathbf{s}$

2) $\mathbf{y}_1=\left(\mathbf{A}^{11}\right)^{-1}\mathbf{x}_1$

3) $\mathbf{v}=\mathbf{x}_2-\mathbf{A}^{21}\mathbf{y}_1$

Steps 1) and 3) involve sparse submatrices $\mathbf{A}^{12}$ and $\mathbf{A}^{21}$ of $\mathbf{A}^{-1}$. Because these steps compute a submatrix by vector product, the calculations can be performed using pedigree information without making the submatrices (Henderson 1976, Quaas 1976). Step 2) can be calculated by solving $\mathbf{y}_1$ in $\mathbf{A}^{11}\mathbf{y}_1=\mathbf{x}_1$ which can be done by alternative approaches (see “Solving approaches in Bpop”).

The above three steps can be used to calculate vector **f** in equation [3], $\mathbf{c}=\mathbf{f}'/\left(2\mathbf{1}_n^{'}\mathbf{f}\right)$, by assigning vector **s** to be $\mathbf{1}_n$, and the result **v** equals **f**. For the computation of equation [4], $\mathbf{C}=\frac{1}{2}(\mathbf{Q}'\mathbf{F})^{-1}\mathbf{F}'$, the three steps need to be done *r* times, separately for each column of **Q**, to compute columns of **F**. Hence, vector **s** is a column of **Q**, and the result **v** is the corresponding column in **F**.

Bpop program

We have written an easy-to-use software, called Bpop, to calculate estimates to the base population AF using the described approaches. To address large populations and numbers of genotyped animals, Bpop has been written with Fortran 95.

The program expects at least names of pedigree and marker genotype files which include the input information for the calculations. By default, the relationship matrix computations, $\mathbf{v}=(\mathbf{A}_{22})^{-1}\mathbf{s}$, do not use inbreeding coefficients because the program does not compute them. The pedigree information can be supported by inbreeding coefficients from a file such that the relationship matrix computations include inbreeding coefficients. All the information to compute elements of the $\mathbf{A}_{22}$ matrix is included in the pedigree of the genotyped animals and their ancestors. The Bpop program prunes the given pedigree to contain only the genotyped animals and their ancestors before the AF estimation.

The pedigree file is expected to have a line for each animal. Each row has three numbers. The first column has the animal ID code and the next two columns have its parent ID codes. All ID codes must be positive 64-bit integer numbers which can be at most 2^{63}-1. Unknown parent ID code is either zero or a negative number.

The genotypes are expected to be in a text file. The first column in the genotype file has the ID code numbers. The genotypes of an individual are on the same line after its ID code. The genotypes are assumed to have count of one of the alleles, i.e., numbers zero, one or two, as described earlier. By default, the genotypes are expected to be space separated integer numbers. Option "-FMT" (Table 1) enables user to give a fixed format with Fortran syntax. For example, -FMT "(i10,26x,50240i1)" assumes there to be a 10-digit ID code, 26 spaces, and 50240 genotypes without space separation. Number of markers in the format (50240 in the example above) has to be equal or more than in the genotype file. The program calculates and uses the actual number of markers on the first row of the genotype file if a too large number of genotypes is given in the format.

The base population AF are computed by equation [1] when no information on groups is provided. The base population animals can be assigned to groups for which the base population AF are estimated using equation [2]. The group numbers should be negative integer numbers in the place of unknown parents, i.e., any negative integer value as a parent indicates that the animal has a missing parent. These group numbers are needed for the base population animals only. In other words, the program calculates elements of the **Q** matrix for each animal by tracing the pedigree to the base population identified by the base population unknown parent group numbers. In order to estimate base population AF for the groups using equation [2], option "-group n" has to be given where n is equal to or more than the number of groups in the pedigree. More Bpop program options are in Table 1.

Table 1. Bpop program requires always the file names of pedigree and marker genotype (**M** matrix) as user input. Optional input, their description and defaults are given below.

Option	Description
-info	print program instructions, current option values and stop.
-nthr n	Number of threads. Default is 1.
-a <file>	output file name for the estimated base population allele frequencies. Default: marker genotype file appended with "_AF_MTH" where MTH equals the calculation method (IOP, IM, or CHM).
-F <file>	input file name for the inbreeding coefficients file (INPUT). Defaut file format: <id> <number> <inbreeding coefficient>
-Fcol c	column number for the inbreeding coefficients in the -F file option. Default is 3.
-m1 c	change default column number (2) of the first marker in the marker file.
-FMT FMT	format given for <id code> and <genotypes>, e.g. -FMT "(i2,1x,6i1)".
-CR v	convergence statistic threshold value in iterative solving, default 10^{-5}.
-groups n	allele frequencies by group (negative unknown parent number), n= maximum number of groups in the pedigree. Default: n=1, and negative parent numbers are ignored.
-proportions <file>	write the group proportions in **Q** by genotyped animal to file. Default: no file created.
Calculation method (only one can be used):	
-IOP	iteration on pedigree.
-IM	iteration in memory (default).
-CHM	CHOLMOD approach.

In the Bpop program, the base population AF are computed by $\hat{p}_i=\mathbf{cm}_i$ for a single group population, and by $\hat{\mathbf{p}}_i=\mathbf{Cm}_i$ for a multiple group population, where *i* is the marker number, and $\mathbf{m}_i$ is column *i* in the marker matrix **M**. The marker genotypes **M** are assumed to be in a file where each line has all markers for an individual. In order to save memory and allow large genotyped populations, the **M** matrix is not stored to memory. Instead, the genotype file is read line by line, i.e., rows of matrix **M** are processed. The **c** vector or the **C** matrix is kept in memory. Multiplication of each marker row of **M** by **c** or **C** is performed to all markers simultaneously, and the result is accumulated to the AF estimates. For example, consider AF matrix $\hat{\mathbf{P}}=\mathbf{CM}$. It is calculated by sum

$\hat{\mathbf{P}}=\sum_{j=1}^{n}\mathbf{c}_{.j}\mathbf{m}_{.j}$, where $\mathbf{c}_{.j}$ is column *j* of **C** and $\mathbf{m}_{.j}$ is row *j* of **M**.

Solving approaches in Bpop

Computation of GLS estimates can be done following the three steps described earlier in section "*Computational algorithms*". Two of the steps, numbered 1) and 3) are matrix times vector multiplications. Computationally most challenging is step 2) which requires calculating $\mathbf{v}=(\mathbf{A}_{22})^{-1}\mathbf{s}$, i.e., solving $\mathbf{y}_1$ in $\mathbf{A}^{11}\mathbf{y}_1=\mathbf{x}_1$. This solving of a linear system of equations can be done in many ways, and the Bpop program allows choosing one of three alternatives (Table 1). Two of the approached, options named IOP and IM, use preconditioned conjugate gradient (PCG) iteration, and the third option, named CHM, uses direct solving by CHOLMOD library (Davis and Hager 2009, Chen et al. 2008).

In the IOP and IM approaches, PCG iteration is used to solve $\mathbf{y}_1$ in $\mathbf{A}^{11}\mathbf{y}_1=\mathbf{x}_1$. In PCG, the core iteration step involves multiplication of a vector by the $\mathbf{A}^{11}$ matrix. In the IOP aproach, the required computations are done by reading pedigree list of the genotyped animals and their ancestors without ever explicitly forming the $\mathbf{A}^{11}$ matrix (see Appendix). Because no $\mathbf{A}^{11}$ matrix was formed for the IOP approach, the RAM memory need is expected to be small. In the IM approach, the $\mathbf{A}^{11}$ matrix is stored in memory as a sparse matrix and used in PCG.

The IOP and IM methods (options "-IOP" and "-IM") use the preconditioned conjugate gradient (PCG) method in solving $\mathbf{y}_1$ in $\mathbf{A}^{11}\mathbf{y}_1=\mathbf{x}_1$. Diagonal of the $\mathbf{A}^{11}$ matrix is used as the preconditioner in the PCG method. Convergence statistic at the iteration round *k* is

$$c_k=\sqrt{\frac{\left(\mathbf{A}^{11}\mathbf{y}_1^{[k]}-\mathbf{x}_1\right)'\left(\mathbf{A}^{11}\mathbf{y}_1^{[k]}-\mathbf{x}_1\right)}{\mathbf{x}_1{'}\mathbf{x}_1}}$$

where $\mathbf{y}_1^{[k]}$ is vector of solutions at round *k*, and $\mathbf{x}_1$ is the right-hand side. Convergence is assumed when c_k is less than 10^{-5}. The default convergence limit can be changed using option "-CR" (Table 1).

In the CHM approach, the sparse $\mathbf{A}^{11}$ matrix is built in memory as in the IM approach. Inverse of the $\mathbf{A}^{11}$ matrix can be dense although the $\mathbf{A}^{11}$ matrix is sparse. Consequently, sparse Cholesky factorization of $\mathbf{A}^{11}$ by CHOLMOD library is used in solving of $\mathbf{A}^{11}\mathbf{y}_1=\mathbf{x}_1$, i.e., a direct method is used instead of PCG iteration. The factorization is done with minimal fill-ins of the (sparse) matrix. CHOLMOD includes high-performance left-looking supernodal factorization and solving methods (Ng and Peyton 1993) based on LAPACK (Anderson et al. 1999) and BLAS (Basic Linear Algebra Subprograms) (Dongarra et al. 1990). The use of CHOLMOD requires two steps: ordering/factorization, and solving. The first step is done only once, and the direct solving is done for each vector **s** which is $\mathbf{1}_n$ or a column in **Q**.

Study design and data

The three approaches (IOP, IM and CHM) were tested in estimation of AF using beef cattle data from the Irish Cattle Breeding Federation (ICBF). The full pedigree had 10.26 million animals of which 1.50 million were genotyped. The genotyped animals had an ancestor pedigree of 1.83 million non-genotyped animals. The animals had been genotyped using different versions of ICBF IDB SNP chip but before the analyses these were imputed to the standard Illumina Bovine SNP50 Bead Chip (Illumina, San Diego, USA). There were 50240 markers from 29 bovine autosomes available for the analysis. Original pedigree of the genotyped animals had 46 unknown parent groups which were determined by breed of animal with unknown parent(s). Groups having average proportion in the **Q** matrix lower than 0.01% were combined to one group such that the final number of groups was 24. Average proportion of the combined group in the **Q** matrix was 0.01%.

It was first verified whether all three approaches gave the same solutions, and after that they were compared according to computing time to calculate $\mathbf{v}=(\mathbf{A}_{22})^{-1}\mathbf{s}$, i.e., step 2), total computing time, and peak random access memory (RAM) use. Number of PCG iterations is reported for the IOP and IM approaches. The full 1.50 million genotyped data were used in estimation of base population AF for two base populations: a single group and a 24-group population. In addition, single population AF were computed using randomly sampled 0.1, 0.5 and 1 million genotyped animals in order to investigate scalability of computations.

We used a multi-core computer with two Intel® Xeon® E5-2680 v2 (2.8 GHZ) processors. A single thread was mostly used. However, parallel computing using at most 10 CPU cores was tested when computations used the CHM approach.

The Bpop program has command line options (Table 1). An example command line to estimate base population AF using the ICBF data set:

```
Bpop -a IM.dat -FMT “(i10,26x,50240i1)” -F ICBF.inbr ICBF.ped ICBF_geno.dat
```

In this case, the default approach (IM) will be used in the computations because no computing approach is given. The inbreeding coefficients are in file “ICBF.inbr”, the pedigree is in file “ICBF.ped” and the genotypes in file “ICBF_geno.dat”. The estimated base population AF will be written to file “IM.dat”. Because no “-groups” option was given, a single population is assumed. Including command “-groups 50” would estimate base population AF for the groups given in the pedigree defined as negative parent numbers for the animals without known parents. The number 50 in the option “-groups” is at least as large as the number of groups in pedigree.

Results and discussion

The studied three approaches gave the same or almost the same AF estimates. Correlations between AF from any two approaches were 1.00000 for the single population analysis with the largest difference of 0.0001 in estimated base population AF for any marker by any two approaches. For the 24-group case, the correlations between AF for the approaches were 1.00000 and the larger difference was 0.0002. Thus, the PCG iteration based and the direct solver based approaches reached almost the same solutions.

Computing time due to calculating the **c** vector in [3] took only a fraction of the total computing time (Table 2). Consequently, differences in total computing time between the three approaches were small. The CHM approach was the slowest because the extra computing time due to making the factorization took all the computing time benefits attained by fast solving of the **c** vector. The factorization took 4.38 min and 1.15 min with one and ten cores, respectively. When the number of genotyped animals was reduced, the total computing time reduced as well (Table 3). In general, all approaches showed similar total computing times.

Table 2. Number of PCG iterations (N), wall clock time for computing the c factor (T_c), total wall clock time (T_T) and peak memory use (RAM) in single group allele frequency estimation when using all genotypes and computations are based on iteration on pedigree (IOP), iteration in memory (IM), or sparse Cholesky factorization (CHM).

Approach	N iteration	T_c (min)	T_T (min)	RAM (GB)
IOP	400	1.18	52.5	0.67
IM	381	0.32	50.9	0.80
CHM	–	4.63	55.9	7.53
CHM, parallel	–	1.42	50.4	8.20

The approaches needed different amount of RAM (Table 2) which was expected. The IOP approach required least amount of RAM. In IOP, only the pedigree list is read, and computations require some extra memory. In the IM and CHM approaches, it was necessary to have the sparse matrix $\mathbf{A}^{11}$ of size 1828434, i.e., number of ancestors to the genotyped animals, in RAM. The matrix was stored in compressed sparse row format where each non-zero coefficient value was stored in a double precision real and its column number in a 32-bit integer. The number of non-zero elements in $\mathbf{A}^{11}$ was 4140064. Thus, less than 0.001% of the elements in $\mathbf{A}^{11}$ were non-zero. The needed additional memory due to this sparse matrix was about 130 MB (Table 2). Note that the memory need of 130 MB includes also some extra memory allocation for the sparse matrix because the space was allocated before the actual number of non-zeros in the $\mathbf{A}^{11}$ matrix was known. The IM approach showed some speed benefit over the other approaches although the total computing time was not much affected (Table 2).

Table 3 illustrates the effect of increase in the number of genotyped animals to peak RAM. The IOP and IM approaches showed only a modest increase in peak RAM but RAM increase for the CHM approach was faster. However, even in the CHM approach, the increase in peak RAM was quite linear in the number of genotyped animals.

The CHM method is based on a direct solving approach where reordering of the equations is used to minimize both the memory use and the number of computations in the factorization and solving steps. The additional RAM in the CHM approach over the IM approach increased as the number of genotyped animals increased. When all 1.50 million genotyped animals were used in the computations, the CHM method needed ten times more memory than merely storing the $\mathbf{A}^{11}$ matrix as in the IM approach (Table 3).

Table 3. Total wall clock time in minutes (T_T) and peak memory use in giga bytes (RAM) in single group allele frequency estimation when number of genotyped animals was 100,000 (100K), 500,000 (500K), 1,000,000 (1M), and 1,500,000 (1.5M), and computations use iteration on pedigree (IOP), iteration in memory (IM), or sparse Cholesky factorization (CHM).

	100K		500K		1M		1.5M	
Approach	T_T	RAM	T_T	RAM	TT	RAM	TT	RAM
IOP	3.7	0.63	17.4	0.63	35.0	0.63	55.0	0.67
IM	3.7	0.63	16.5	0.65	32.5	0.73	50.9	0.80
CHM	3.8	0.95	17.5	3.20	34.8	5.53	55.9	7.53

The subsets of genotyped animals in Table 3 were a random sample from the 1.50 million genotyped animals. In practice, number of genotyped animals increases due to genotyping of young animals. Most of the new genotyped animals can be expected to have at least one of the parents genotyped. Consequently, the number of non-zero elements in $\mathbf{A}^{11}$ can be expected to increase slowly. Thus, the IM approach will continue to have a low memory need and the memory increase in the CHM approach can be expected to be tolerable in the future (Masuda et al. 2016, Taskinen et al. 2017). In the analysis of our data sets, the number of non-zero elements in $\mathbf{A}^{11}$ matrix was 0.97 million, 2.62 million, 3.52 million, and 4.14 million, for the cases of 0.10, 0.50, 1.00 and 1.50 million genotyped animals, respectively. Thus, number of non-zero elements increased at a slower pace than the number of genotyped animals.

Performance of the approaches adopted in this study to compute term $(\mathbf{A}_{22})^{-1}$ times a vector has been investigated in solving mixed model equations from a ssGBLUP model. Strandén et al. (2017) used PCG iteration to solve ssGBLUP. In regular ssGBLUP, the $\mathbf{A}_{22}$ matrix was computed and inverted. The three approaches using formula [5] for $(\mathbf{A}_{22})^{-1}$ were used to decrease total computing time. They found that the solver computing time increased, but preprocessing time decreased, when formula [5] was used. Time to solve MME increased by c. 25% by IOP, c. 11% by IM, and c. 2% by CHM in comparison to the regular ssGBLUP. However, the total computing time decreased by almost 30% when using CHM because there was no need to make and invert $\mathbf{A}_{22}$. Advantage of the CHM approach is expected to increase, when the number of genotyped animals increases because computing time for inverting $\mathbf{A}_{22}$ increases qubically in number of genotyped animals but the increase in computing time due to formula [5] is linear (Masuda et al. 2016). In contrast, in the base population AF estimation, total computing time by the CHM approach was often more than by the IOP and IM approaches because the preprocessing time to make the factorization was substantial in comparison to the need to compute the $\mathbf{c}$ vector only once.

Aldridge et al. (2018) estimated base population AF using the GLS method for a single population. They compared a direct sparsity preserving solving approach (GLS_Sparse) similar to our CHM approach to an approach where the matrix was explicitly made and inverted (GLS_Full). The GLS_Sparse approach needed 49 seconds and 1.3 GB RAM with 1,670 SNP markers from 100,078 genotyped animals. The GLS_Full approach needed about 32 h and 118.2 GB of RAM. When the number of markers was increased to 50,100, the GLS_Sparse approach took 6 minutes and 37.6 GB of RAM. These numbers support the conclusion that the implemented implicit computing approach for the $\mathbf{A}_{22}$ matrix is efficient. The smaller memory need by the Bpop program than the GLS_Sparse approach in the 100,000 genotyped case can be due to differences in the direct solver or having the $\mathbf{M}$ matrix in memory or other reasons. Thus, direct comparison of programs should be treated with caution.

We did not consider any approach where the $\mathbf{A}_{22}$ matrix would be formed explicitly because the number of genotyped animals in our data was so large. In practice, when the number of genotyped animals increases sufficiently, preprocessing time to make $\mathbf{A}_{22}$ becomes unfeasible due to the large memory requirements. For example, when genotypes are available from 1.5 million animals, dense square matrix of the size 1.5 million stored in double precision would take about 18 terabytes. When a dense lower triangle or packed matrix is used, storing the $\mathbf{A}_{22}$ matrix in single precision would still require 4.5 terabytes.

In this study, benefits of using parallel computing were small (Table 2). The CHOLMOD library can be compiled to use one CPU or several CPUs. Thus, it allows exploiting parallel computing. Parallel computing decreased time in the CHOLMOD computing steps but, as already noticed, this step takes only a fraction of the total computing time. In addition to the parallel CHOLMOD, we tested parallel computing in the AF estimation by using parallel versions of DAXPY and DGEMV subroutines available in LAPACK/BLAS from the Intel Math Kernel Library© (Intel Math Kernel Library Reference Manual, 2014). Parallel computing did not give much advantage here either. In the current implementation, one row of **M** is in memory at a time which leads to calling the parallel subroutines as many times as there are genotyped animals. Benefits from parallel computing are likely to be larger when the full **M** matrix is in RAM as in Aldridge et al. (2018). Then, one matrix times matrix product using DGEMM or SGEMM subroutine call can be used and, thus, the parallel subroutines use data from all animals simultaneously. However, having the **M** matrix in RAM would increase the peak RAM substantially. The **M** matrix for our full genomic data would take about 600 GB in double precision, which has to be used in order to use the DGEMM subroutine. If single precision subroutine SGEMM is considered to give sufficient numerical accuracy, then the RAM need is halved. A hybrid approach would allow reading and using genotypes from several individuals at a time. This hybrid approach would have a lower RAM use than storing the full **M** matrix in RAM and, perhaps, more efficient matrix times matrix computations by DGEMM than in the current implementation.

Table 4. Average number of PCG iterations (N), wall clock time for calculating the **C** matrix in Formula [4] (T_C), total wall clock time (T_T) and peak memory use (RAM) in multi group allele frequency estimation when computation use iteration on pedigree (IOP), iteration in memory (IM), or sparse Cholesky factorization (CHM).

Approach	N iteration	T_C (min)	T_T (min)	RAM (GB)
IOP	300	10.5	89.7	1.68
IM	286	5.3	85.5	1.68
CHM	–	4.7	91.3	8.38
CHM, parallel	–	1.8	85.3	9.05

Estimation of base population AF to groups decreased differences between the approaches. Table 4 has computing times from estimating base AF for the 24 groups. In general, the results were similar to those for single group in Table 2. However, the difference in peak RAM was not as large between the approaches for the multiple group case as for the single group case. This is most likely due to the increase in RAM need due to the 24 groups each of which required a column in the **Q** matrix. In other words, the group proportions for the genotyped animals and their ancestors were temporarily needed in the computation of the **Q** matrix which was stored in RAM in double precision and took almost 650 MB. The difference in peak RAM between IOP and IM approaches was minimal because memory was deallocated after the full **Q** matrix had been made, and the $\mathbf{Q}_2$ submatrix of only the genotyped animals was used in the base AF computations. The additional memory needed to store $\mathbf{A}^{11}$ in the IM approach was so small that the deallocated RAM from the full **Q** memory was more than enough for its use.

Conclusions

A computationally efficient program called Bpop was written to estimate base population AF using a GLS approach. Computationally the most demanding step involves inverse of pedigree relationship matrix between genotyped animals, $(\mathbf{A}_{22})^{-1}$, times a vector or a matrix. The $\mathbf{A}_{22}$ matrix is dense. We presented and implemented three alternative approaches which do not require explicitly making the $(\mathbf{A}_{22})^{-1}$ matrix. These approaches used an equivalent matrix formula of $(\mathbf{A}_{22})^{-1}$ involving sparse matrices. This formulation allowed use of marker data from many genotyped animals. The computing step involving the reformulated $(\mathbf{A}_{22})^{-1}$ matrix was very fast. The three approaches had small differences in total computing time but had larger differences in the needed amount of peak computer memory. Thus, choice of the computing approach can be made based on the available computer memory.

Availability and requirements

The program Bpop is provided free of charge for the scientific community, but users are required to credit its use in any publication. Commercial users must contact the authors. Bpop executable is available for Linux upon request from the corresponding author. The program is under ongoing development, and due to the number of features, some combinations of options may not have been tested thoroughly.

Acknowledgements

The authors acknowledge the Irish Cattle Breeding Federation (ICBF) for providing the beef cattle data.

References

Aldridge, M.N., Vandenplas, J. & Calus, M.P.L. 2018. Efficient and accurate computation of base generation allele frequencies. Journal of Dairy Science 102:1364–1373. https://doi.org/10.3168/jds.2018-15264

Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenny, A. & Sorensen, D. 1999. LAPACK Users' Guide. 3rd ed. SIAM. Philadelphia, PA, USA. https://doi.org/10.1137/1.9780898719604

Aguilar, I., Misztal, I., Johnson, D.L., Legarra, A., Tsuruta, S. & Lawlor, T.J. 2010. Hot topic: A unified approach to utilize phenotypic, full pedigree, and genomic information for genetic evaluation of Holstein final score. Journal of Dairy Science 93:743–752. https://doi.org/10.3168/jds.2009-2730

Chen, Y., Davis, T.A., Hager, W.W. & Rajamanickam, S. 2008. Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate. ACM Transactions on Mathematical Software 35:22. https://doi.org/10.1145/1391989.1391995

Christensen, O. & Lund, M.S. 2010. Genomic prediction when some animals are not genotyped. Genetics Selection Evolution 42:2. https://doi.org/10.1186/1297-9686-42-2

Christensen, O.F., Madsen, P., Nielsen, B., Ostersen, T. & Su, G. 2012. Single-step methods for genomic evaluation in pigs. Animal. 6:1565:1571. https://doi.org/10.1017/S1751731112000742

Davis, T.A. & Hager, W.W. 2009. Dynamic supernodes in sparse Cholesky update/downdate and triangular solves. ACM Transactions on Mathematical Software 35:27. https://doi.org/10.1145/1462173.1462176

Dongarra, J.J., Du Croz, J., Duff, I.S. & Hammarling, S. 1990. A set of level-3 basic linear algebra subprograms. ACM Transactions on Mathematical Software 16:1–17. https://doi.org/10.1145/77626.79170

Garcia-Baccino, C.A., Legarra, A., Christensen, O.F., Misztal, I., Pocrnic, I., Vitezica, Z.G. & Cantet, R.J.C. 2017. Metafounders are related to Fst fixation indices and reduce bias in single-step genomic evaluations. Genetics Selection Evolution 49:34. https://doi.org/10.1186/s12711-017-0309-2

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. https://doi.org/10.2307/2529339

McPeek, M.S., Wu, X. & Ober, C. 2004. Best linear unbiased allele-frequency estimation in complex pedigrees. Biometrics 60:359–367. https://doi.org/10.1111/j.0006-341X.2004.00180.x

Mäntysaari, E.A., Koivula, M. & Strandén, I. 2020. Symposium review: Single-step genomic evaluations in dairy cattle. Journal of Dairy Science. https://doi.org/10.3168/jds.2019-17754

Masuda, Y., Misztal, I., Tsuruta, S., Legarra, A., Aguilar, I., Lourenco, D.A.L., Fragomeni, B.O. & Lawlor, T.J. 2016. Implementation of genomic recursions in single-step genomic best linear unbiased predictor for US Holsteins with a large number of genotyped animals. Journal of Dairy Science 99:1968–1974. https://doi.org/10.3168/jds.2015-10540

Ng, E. & Peyton, B. 1993. Block sparse Cholesky algorithms on advanced uniprocessor computers. SIAM Journal of Scientific Computing 14:1034–1056. https://doi.org/10.1137/0914063

Quaas, R.L. 1976. Computing the diagonal elements and inverse of a large numerator relationship matrix. Biometrics 32:949–953. https://doi.org/10.2307/2529279

Strandén, I. & Lidauer, M. 1999. Solving large mixed models using preconditioner conjugate gradient iteration. Journal of Dairy Science 82:2779–2787. https://doi.org/10.3168/jds.S0022-0302(99)75535-9

Strandén, I., Matilainen, K., Aamand, G. & Mäntysaari, E.A. 2017. Solving efficiently large single-step genomic best linear unbiased prediction models. Journal of Animal Breeding and Genetics 134:264–274. https://doi.org/10.1111/jbg.12257

Taskinen, M., Mäntysaari, E.A. & Strandén, I. 2017. Single-step SNP-BLUP with on-the-fly imputed genotypes and residual polygenic effects. Genetics Selection Evolution 49:36. https://doi.org/10.1186/s12711-017-0310-9

VanRaden, P.M. 2008. Efficient methods to compute genomic predictions. Journal of Dairy Science 91:4414–4423. https://doi.org/10.1111/jbg.12257

Appendix: Solving $\mathbf{y}_1=(\mathbf{A}^{11})^{-1}\mathbf{x}_1$ iteratively

Bpop program has three alternative approaches to solve the equation $\mathbf{A}^{11}$ $\mathbf{y}_1$= $\mathbf{x}_1$.In two of the approaches, preconditioned conjugate gradient (PCG) iteration is used. These two alternatives use either a $\mathbf{A}^{11}$ matrix stored in memory (the IM approach) or a pedigree list information without explicitly making $\mathbf{A}^{11}$ (the IOP approach) for the necessary computations in PCG iteration. All three approaches use the same rules as those used to make the full $\mathbf{A}^{-1}$ using a pedigree list (Henderson 1976, Quaas 1976). In the PCG method (e.g., Strandén and Lidauer 1999) used by the IM and IOP approaches, each iteration requires multiplication of the current search direction vector $\mathbf{d}_1$ by the coefficient matrix $\mathbf{A}^{11}$. Thus, every iteration of PCG calculates product $\mathbf{s}$= $\mathbf{A}^{11}\mathbf{d}_1$. In the following, we will be described the computational steps for this product in the IOP approach.

The product $\mathbf{s}$= $\mathbf{A}^{11}\mathbf{d}_1$can be computed by considering the full $\mathbf{A}^{-1}\mathbf{d}$ matrix product rules, where $\mathbf{d}_2$=$\mathbf{0}$, and by updating only those elements that change vector , i.e., non-genotyped ancestors to the genotyped animals. According to the rules of making $\mathbf{A}^{-1}$ (Henderson 1976, Quaas 1976), the product $\mathbf{A}^{-1}\mathbf{d}$ can be computed by proceeding the pedigree list (in any order) and updating elements of individual *i*, its sire *s* and dam *d* in the result vector $\mathbf{s}$. For example, when both parents are known, update for individual *i* and its parents is

$$\mathbf{s}_{\text{update}}=\text{f}_\text{i}\begin{bmatrix}-0.5\\-0.5\\1\end{bmatrix}\begin{bmatrix}-0.5 & -0.5 & 1\end{bmatrix}\begin{bmatrix}\mathbf{d}_\text{s}\\\mathbf{d}_\text{d}\\\mathbf{d}_\text{i}\end{bmatrix}$$

where $\text{f}_i = 4/(4–k–\text{F}_s–\text{F}_d)$, *k*=2 is the number of known parents to individual *i*, F_s is inbreeding coefficient of sire, and F_d is inbreeding coefficient of dam. Corresponding update formulas are available for individuals with only one or no known parents. Note that the computations can be efficiently performed from right to left such that no matrices are stored (Strandén and Lidauer 1999):

1) $$\text{c}=\begin{bmatrix}-0.5 & -0.5 & 1\end{bmatrix}\begin{bmatrix}\mathbf{d}_\text{s}\\\mathbf{d}_\text{d}\\\mathbf{d}_\text{i}\end{bmatrix}$$

2) $$\mathbf{s}_{update}=\begin{bmatrix}-0.5cf_i\\-0.5cf_i\\cf_i\end{bmatrix}.$$

Because the IOP approach is used to calculate $\mathbf{s}$=$\mathbf{A}^{11}\mathbf{d}_1$, not $\mathbf{A}^{-1}\mathbf{d}$, the multiplications of $\mathbf{d}$ can be limited to the non-genotyped ancestors of the genotyped animals. Likewise, the update is applied to these non-genotyped ancestors as well.

Figure 1 has a Fortran-type pseudo code for the multiplication s=$\mathbf{A}^{11}\mathbf{d}_1$. The 'pedigree' table has pedigree information for all genotyped animals and their ancestors. For every individual, the pedigree has three numbers: individual, sire and dam ID numbers. In the Bpop program, the full pedigree has been pruned to have the genotyped animals and their ancestors. Furthermore, the animal ID codes have been renumbered to be consecutive integer numbers from one to the number of animals N. There are three vectors. Vector $\mathbf{F}$ has the pre-calculated inbreeding coefficients, vector $\mathbf{d}$ has the current values to be multiplied by $\mathbf{A}^{11}$, and vector $\mathbf{s}$ will have the result of the multiplication.

Fig. 1. Pseudo code example of calculation of product $\mathbf{s}=\mathbf{A}^{11}\mathbf{d}_1$

```
$\mathbf{s}$ = 0
do i=1,N
  (id, sire, dam) = pedigree(i) ! ID, its sire and dam numbers
  ! step 1): compute multiplier c
  k = 0 ! number of known parents
  f = 0 ! sum of parent inbreeding coefficients
  c = 0 ! result from the first step multiplication
  if (sire > 0) then ! sire known?
    k = k + 1
    f = f + $\mathbf{F}_{sire}$
    if (sire in $\mathbf{A}^{11}$) c = c – 0.5*$\mathbf{d}_{sire}$
  end if
  if (dam > 0) then ! dam known?
    k = k + 1
    f = f + $\mathbf{F}_{dam}$
    if (dam in $\mathbf{A}^{11}$) c = c – 0.5*$\mathbf{d}_{dam}$
  end if
  if (id in $\mathbf{A}^{11}$) c = c + $\mathbf{d}_{id}$
  ! Step 2): update vector s
  if (c is nonzero) then
    d = 4/(4-k-f)
    if (sire in $\mathbf{A}^{11}$) $\mathbf{s}_{sire}$= $\mathbf{s}_{sire}$ – 0.5*d*c
    if (dam in $\mathbf{A}^{11}$) $\mathbf{s}_{dam}$ = $\mathbf{s}_{dam}$ – 0.5*d*c
    if (id in $\mathbf{A}^{11}$) $\mathbf{s}_{id}$ = $\mathbf{s}_{id}$ + d*c
  end if
end do
```