ASReml Update

What's new in Release 4.2

A R Gilmour
VSN International, Hemel Hempstead, United Kingdom

R Thompson
Rothamsted Research, Harpenden, United Kingdom

June 2021

ASReml Update. What's new in Release 4.2

The following significant changes have been made in ASReml 4.2.

- ASReml can now fit a bivariate GL(M)M model (page 10).
- Pedigree trimming has been implemented along with the option of absorbing ancestors without data (page 11). Trimming strips animals out of the pedigree that are not needed for the analysis.
- The variance function `rr`*k*`(`*f*`)` has been added as an alias for `xfa`*k*`(`*f*`)` with all specific variances set to 0.0 (page 2).
- Memory access has been increased; ASReml 4.2 can utilise up to 96 Gbyte work space if it is available (page 6).
- Some internal routines have been reorganised to run substantially faster (page 6).

Contents

1 General changes

1.1 Model functions

1.1.1 Design functions

These new design functions assist in setting up particular design models.

`hs(`*f*`,`*k*`)`	is a factor with level *1* if factor *f* is coded 1 or *k* and zero otherwise.
`ref(`*f*`,`*k*`)`	creates a factor using levels from *f* except that level *k* is set to zero. This effectively sets *k* (default 1) as the reference level of the factor *f* when `mu` and `ref(`*f*`,`*k*`)` are fitted and the level *j* effect of `ref(`*f*`,`*k*`)` estimates the difference of level *j* and *k* of the factor *f* effects

`zero(`*k*`)`
`zero`
	is used to include *k* column(s) of zeros in the model. It is primarily intended to be used to set aside space for fitting the factors of an extended factor analytic (XFA*k*) or reduced rank (RR*k*) model applied across several model terms.
	For example:
	`str(ani age.ani zero(1).ani xfa1(2).ani)`
	`zero` is equivalent to `zero(1)`

1.1.2 Reduced rank variance matrix function

The `xfak()` variance structure model function can fit reduced rank variance structures by setting specific variances to zero. A new form of this function, `rrk()`, is available which automatically sets all the specific variances ($\mathbf{\Psi}$) to zero. `rrk()` is formally just a synonym for `xfak()` which means that if you explicitly supply initial values, you need to supply the $\mathbf{\Psi}$ values as zero.

An almost equivalent way of fitting an `xfak()` structure is to fit an `rrk()` term and a `diag()` term, for example `rrk(trial).entry + diag(trial).entry` in place of `xfak(trial).entry`. The `rr+diag` form may run faster when there is a relationship matrix associated with `entry`.

1.1.3 Revised spatial analysis processing (!ARLIMIT)

Conventional spatial analysis in ASReml has focussed on fitting an AR $\times$ AR residual structure, which for multiple blocks is conveniently specified as `residual sat(Site).ar1(Row).ar1(Column)`.

Two recently identified issues have been addressed:
1. where a few sites have only 1 row/column, ASReml now allows you to define `ar(Row)` when there is just 1 row: it fixes the AR parameter but does not use it.
2. where the correlation parameter goes to the boundary (± 0.999); the qualifier `!ARLIMIT [`ρ`]` resets the boundary limit for the magnitude of the AR parameter. The default for the new limit (`!ARLIMIT` without an argument) is 0.75.

1.1.4 Fixed covariate adjustment option (!TAU)

When using the two-step procedure of fitting a spatial model with Genotype random to estimate spatial parameters and then holding these fixed to obtain adjusted Genotype means, a covariate is sometimes also fitted in the first model, whose coefficient needs to be used in the second run to

adjust the data. The process involves two steps. First we specify `!TAU` x to nominate the covariate whose coefficient is to be captured, or the position number of the covariate. The default position is 1. Second we specify `$Tau` to substitute the coefficient into the job code.

```
!RENAME !ARG 1 2
Form adjusted variety means with weights  !DOPART $1
yield
plantheight
variety *
row * column *
adjyld !=plantheight !*$Tau !*-1 !+yield
yhv.csv !skip 1   !Tau plantheight
!PART 1
yield ~ mu plantheight mv !r variety
residual ar1(row).ar1(column)
!PART 2
!DF -1 !HOLD !CONTINUE !MAXIT 1
adjyld ~  mu variety mv
residual ar1(row).ar1(column)
predict variety !TWOSTAGE
```

This job fits two models. In the first, `$Tau` has a value of 1 when used to calculate `adjyld` but `adjyld` is not used in the first model. The coefficient value is updated at the end of the the first model fit to the regression coefficient obtained in that fit for `plantheight`. In the second model, we analyse `adjyld` which has been corrected for `plantheight`. This example does not take into account the mean plantheight.

1.1.5 VCC parameter number offset (!OFFSET)

Variance parameter constraint (VCC) allows variance parameters to be constrained to be equal (typically). This mechanism requires specification of parameter numbers, but a simple change to the model can change the parameter numbers. Therefore we have added an offset qualifier which can be

specified on the first VCC data line, which allows adjustment of all the parameter numbers. For example, in a multienvironment trial, there were 64 experiments representing 12 year/location combinations. A spatial analysis was required where the spatial parameters (variances and correlations) were constrained to be the same for experiments at each particular year/location. This was achieved by the code

```
...
!VCC 12
yield ~ mu YrLoc mv !r xfa1(YrLoc).Geno ...

residual at(expt).ar1(Row).ar1(Column)
+ 19 22 25 28 31          !BLOCK 3  !OFFSET 19  # 5  AN
+ 34 37 40                !BLOCK 3              # 3  BK
+ 43 46 49 52 55          !BLOCK 3              # 5  BL
+ 58 61 64                !BLOCK 3              # 3  CG
+ 67 70 73                !BLOCK 3              # 3  GW
+ 76 79 82                !BLOCK 3              # 3  KV
+ 85 88 91                !BLOCK 3              # 3  LC
+ 94 97 100 103 106 109   !BLOCK 3              # 6  MN
+ 112 115 118 121 124 127 !BLOCK 3              # 6  NH
+ 130 133 136 139 142     !BLOCK 3              # 5  PN
+ 145 148 151 154 157 ... !BLOCK 3              # 18 RS
+ 199 202 205 208         !BLOCK 3              # 4  WR
```

The 64 experiments define 192 residual variance parameters arranged as 64 sets of 3. The coding of `expt` was such that experiments were nested within the 12 `YrLoc` classes. The parameter numbers 19:210 pertain to the initial model fitted but after adding additional model terms, the parameter numbers changed to 38:229. Rather than having to retype all the numbers, the `!OFFSET 19` qualifier adds 19 to the numbers as they are read in. The offset value remains in force for subsequent VCC statements unless reset.

The `!BLOCK 3` qualifier means that the line is actually interpreted as three lines:
`+ 34 37 40 !BLOCK 3 # 3 BK`
is interpreted as

```
+ 34 37 40 # pertaining to the expt variances
+ 35 38 41 # pertaining to the expt row correlations
+ 36 39 42 # pertaining to the expt column correlations
```

1.1.6 Gamma constraints (!FREEGH)

The gamma constraint code letters `P, U, F` and `Z` affect the way parameters are updated. `P` keeps parameters within the 'parameter space', `U` allows parameters to go outside the 'parameter space', `Z` fixes the parameter (say a covariance) at zero and `F` holds a parameter at its inital value. A new code letter has been introduced. `H` holds the parameter for the first h iterations and then the code changed to `P` so that the parameter can be estimated. h has a default value of 3; the value can be set with the `!FREEGH` h data line qualifier.

1.1.7 Working variables (!WVR)

The (data file line) qualifier `!WVR` reports working variables to a `.wvr` file.

1.2 Licensing

ASReml 4.2 has migrated to a new license system underpinned by Reprise technology. This system is widely used by software vendors and allows for much more robust license management. It allows delivery and management of your license entitlements via the cloud without installing license server software on your systems. Further details about the licensing can be found on the ASReml knowledgebase: `https://asreml.kb.vsni.co.uk/knowledge-base/vsni-new-licensing/`

1.3 Workspace

We have reworked some of the core routines to allow access to a larger workspace (up from 32 to 96 Gbyte workspace) and a more efficient matrix absorption process.

Also, the use of memory has been changed to separate different kinds of data into different arrays in accordance with more modern programming conventions. This results in the need to allocate more memory for a particular job than was needed in ASReml 4.1.

Workspace can be set either on the command line (with the `-W`m option) or with the `!WORKSPACE` m qualifier on the first line of the job (above the title line). The former takes precedence. In either case, m is interpreted in gigabytes. The argument m can include a decimal point and ASReml reports the space available in Gbytes to one decimal place and a value of m less than 0.2 is interpreted as 0.2 Gbytes.

The default workspace in ASReml 4.2 is 2 Gbytes, which is ample for the majority of runs. The minimum allocated is 0.2 Gbytes; the maximum allocated depends on what is available on the PC. We recommend that the workspace requested not exceed the RAM available on your machine (commonly 8 or 16 Gbyte). If your system cannot provide the requested workspace, the request will be diminished until it can be satisfied. On multi-user systems, do not unnecessarily request the maximum or other users may be unhappy. ASReml reports the actual amount of primary workspace used in a job at the end of the .asr file. Sometimes the allocation of primary workspace means that ASReml cannot access sufficient secondary workspace. ASReml will report this and suggest the primary workspace be reduced.

1.4 Performance issues

Timing information useful for comparing execution time between models and/or builds of ASReml is available in the `.asl` file if the `!LOGFILE` and

`!DEBUG` qualifiers (or command line options `-DL`) are set on the first line of the job (above the TITLE line). Running the command `grep '>>' job.asl` at the command prompt (`grep` is a command-line option used to find a specific string from inside a file but it can be used only in Linux. For Windows, the `grep` alternative is `findstr`.

As an example we give the timings of a complex multi environment spatial analysis fitting an `XFA1` model interacted with `ide(Geno)` and `grm1(Geno)`.

```
>> >>  Process                 CPU_time   SumCPU    Clock    SumClock
>> >>  GRM SG inversion: sec   78.19      78.19     11.45    11.45
>> >>  Get NRM/GRM: sec        1.06       79.25     0.14     11.59
>> >>  Getting Started: sec    2.08       81.33     1.89     13.48
>> >>  Before Order : sec      0.86       82.19     0.85     14.34
>> >>  Order found: sec        291.52     373.70    291.52   305.86
FILLIN 101600575 ==>> 133331138 1.31, #SR: 51625, AvLen: 2583, \\
MxLen: 8458, AvFlops: 6677255
>> >>  C reordered: sec        46.86      420.56    46.86     352.72
>> >>  C absorbed: sec         669.86     1090.42   86.26     438.98
>> >>  AI formed: sec          6.81       1097.23   6.82      445.80
>> >>  Ci formed: sec          3150.66    4247.89   406.95    852.75
>> >>  Ci reordered: sec       23.02      4270.91   21.75     874.50
>> >>  Score formed: sec       4.91       4275.81   4.90      879.41
>> >>  Iteration complete: sec 0.00       4275.81   0.01      879.42
>> >>  Finished: sec           1.36       4277.17   1.36      880.78
```

Typically, the major components are `Order found`, `C absorbed` and `Ci formed`. The `INFILLIN` line reports the size of the `C` matrix before and after absorption, and the ratio of the values. There are 10 areas where ASReml 4.2 is faster than the 2015 version of 4.1.

1. Inversion of sparse G matrices has been improved and uses multiple processors.
2. Time to work out an efficient equation order.
3. The absorption routine has been changed to first perform a symbolic absorption so as to establish the complete infill pattern. Then memory

can be accessed in a more linear fashion. In jobs with relatively dense G matrices, this can reduce the iteration time up to 40%.

4. Reordering the C matrix into the order required for analysis.
5. Nodal absorption is implemented in ASReml 4.2. It operates when there are C matrix links between consecutive rows.
6. Multiprocessors are used in the absorption of C and the C inverse step.
7. Improvements in formation of the Average Information matrix.
8. Reordering the C inverse matrix back into the model for calculation of the scores.
9. Improvements in formation of the scores.
10. The processing of `!PRESENT` clauses was not efficient and in an analysis predicting 50,000 genotypes having a hierarchical genotype structure, it took 2 hours to form the predict design matrix when the actual run otherwise only required a minute. Reorganising the process has resolved the issue.

In the example given, it was taking 532 minutes per iteration plus 113 minutes to obtain the equation order in ASReml 4.1. In ASReml 4.2 it now takes 10 mins per iteration (+ 5 minutes to determine equation order).

2 New procedures

2.1 VPREDICT

`VPREDICT` allows the user to calculate common functions of the variance components such as heritability. These functions are reported in the `.pvc` file and also to the `.xml` file when that form of output is requested.

New functionality in ASReml 4.2 includes:

- VPREDICT statements can be written over multiple lines if the incomplete lines end with COMMA (,).
- `W` <filename> i[: j] writes component[s] i [to j] to a file ($filename$`.vpc`) and their variance matrix to another file ($filename$`.vpv`). The format is described in the User Guide.
- Functions `K` and `M` facilitate calculation of transformed variance components. A model may generate an estimated variance-covariance matrix $\boldsymbol{\Sigma}$ for s effects $\boldsymbol{u}$ and there might be interest in using $\boldsymbol{K}$ a $(r \times s)$ matrix to transform $\boldsymbol{u}$ to $\boldsymbol{Ku}$ with derived symmetric $(r \times r)$ variance matrix $\boldsymbol{M} = \boldsymbol{K\Sigma K}'$. For instance, we might have a yield on a plot, one of its row neighbours and one of its column neighbours estimated using a spatial autoregressive model and wish to compute the variance

matrix of sums and differences of these three yields. If $\boldsymbol{\Sigma}$ is the residual variance of the three plots and $\boldsymbol{K}$ a matrix allowing the formation of sums or differences then we wish to form $\boldsymbol{M} = \boldsymbol{K\Sigma K'}$.

`K` *label v*
defines a vector (v) of coefficients
`M` *label i:j m:n*`[!DIAG]`

Specifies $\boldsymbol{K}$ a $r \times c$ matrix derived from the set of r vectors of length c starting with *firstveclabel* and ending with *lastveclabel*. If all the required vectors start with the same initial symbols *initialvecsymbols* can be used to specify $\boldsymbol{K}$. *m:n* specifies $\boldsymbol{S}$ an unstructured symmetric $s \times s$ variance matrix. Note that *m:n* can be replaced by a structure label.

There are two cases if $c = s$ the matrices $\boldsymbol{K}$ and $\boldsymbol{S}$ are conformable and $\boldsymbol{M} = \boldsymbol{KSK'}$ is computed. If the columns of $\boldsymbol{S}$ are a multiple (s/c) of the columns of $\boldsymbol{K}$ then a partitioned form of a $rs/c \times rs/c$ variance matrix $\boldsymbol{M}$ is formed with $(r \times r)$ sub-matrix $\boldsymbol{Mij} = \boldsymbol{KSijK'}$ with $\boldsymbol{S}$ partitioned into $c \times c$ sub-matrices $\boldsymbol{Sij}(i, j = 1, ..., s/c)$. If `!DIAG` is set only the diagonal elements of $\boldsymbol{M}$ are computed. See **13.2.7** of the User Guide for more details.

2.2 Bivariate GLM(M)

ASReml 4 allowed a bivariate analysis of a binomially distributed variate and a linear model variate (normally distributed variate with an identity link). ASReml 4.2 has extended this bivariate analysis of generalized linear models. Both variates are now allowed to be distributed with Normal, Binomial, Poisson, Gamma or Negative Binomial distributions. Because multinomial threshold models are an extension of GLM with composite link functions they may not be included in a bivariate analysis. The GLMM qualifiers related to a variate should be specified immediately after the response variate is nominated. If a link function is specified, it must follow the distribution qualifier. Examples include:
`Scald !BINOMIAL !PROBIT FootRot !BINOMIAL !PROBIT ~ Trait ...`

```
Scald !BINOMIAL !PROBIT weight ~ Trait ...
```
Although ASReml will provide an analysis of grouped binomial data it is usually more statistically efficient, if possible, to expand to the binary data and get a more efficient estimate of the residual covariance. The procedure used is to scale the specified residual variance matrix by the inverse GLM weights, so the values in the specified residual variance matrix are actually dispersion factors, typically initialized with variances of 1.0. For instance:
```
residual units.us(Trait !INIT 1 0.5 1)
#residual units.us(Trait !INIT 1 0.5 1 !GFPF) # to fix the dis-
persion
```
The algorithm used is a heuristic extension to the standard PQL method used in the univariate case.

2.3 More pedigree qualifiers

Pedigree pre-processing has been extended to allow removal of unnecessary individuals, those with no data or descendants with data, sometimes called trimming, and recursive removal of base individuals (individuals with unknown parents) that have no data and only one offspring, sometimes called pruning. Both these operations do not change the likelihood or parameter estimates but can save computation. There is the facility to also limit the number of generations of ancestors in the pedigree; this again can save computation but can change estimates as it implicitly assumes the ancestors left out have no genetic variation. There is an option to form and use a sparse inverse relationship matrix just on a specified set of individuals.

The *pedigree line qualifiers* `!TRIM` *filename* [*field*] [`!SKIP` *k*] [`!NOPRUNE`] [`!KEEP` *g*] [`!REDUCE`] invoke the following processes:

- Read field *field* (default 1) of *filename* (typically the data file) and identify individuals in the pedigree with a data record in *filename*. It does not check whether any particular response variable is missing for that record.
- Write a reduced pedigree file containing those individuals with a data record and their immediate ancestors. The new file has the `.TRM`

filename extension appended to its name. Use `!KEEP` g to limit the number of ancestoral generations to include. Use at least 2 if fitting maternal effects, more if inbreeding is present. The pedigree lines are flagged either `!RETAIN` or `!REMOVE` to distinquish those in the data file from their ancestors (see `!REDUCE`).

Using `!TRIM ... !KEEP 2` a pedigree which contained 7269 animals over 14 generations, the trimmed pedigree contained 1997 individuals over 3 generations, being 856 animals with data, 532 parents of these animals and 609 grandparents. Execution time for the job reduced from 30 s per iteration to 3 s per iteration and the LogL increased 0.18. Note that this can save computation but can change estimates as it implicitly assumes the ancestors left out have no genetic variation.

The `!REDUCE` qualifier requests ASReml check whether `!RETAIN` appears on each line of the (sorted) pedigree file. If present, it forms the sparse $\boldsymbol{A}^{-1}$ matrix and then absorbs the individuals not flagged `!RETAIN`. Generally this will make the reduced $\boldsymbol{A}^{-1}$ much denser and so is only useful in special situations where family size is small or the number of ancestors retained is small (definitely less than 10,000). Using `!TRIM` and `!REDUCE` on the job with 856 animals, execution time increased to 540 s per iteration (a 7 trait analysis)!

```
pedigree.csv !SKIP 1 !ALPHA !TRIM HfrAndCow4.csv 1 !SKIP 1 !RE-
DUCE
```

Note that the first `!SKIP` pertains to the pedigree file and should be specified before the `!TRIM` qualifier; the second pertains to the data file and should appear after the filename.

The `!SORT` qualifier preprocesses the pedigree file writing the sorted file with filename extension `.SRT`. Genetic groups with no members will be dropped by the `!SORT` process. In an effort to save preprocessing effort, if the pedigree file is specified with the `.SRT` extension and the file already exists, ASReml will assume the sorting has already been performed to create the file. If the pedigree file is specified with the `.SRT` extension and the file does not exist, it will be created if the base name file exists. Thus:

- `pedigree.txt !SORT` creates and uses `pedigree.txt.SRT`

- `pedigree.txt.SRT !SORT` uses `pedigree.txt.SRT` as it is if it exists
- `pedigree.txt.SRT !SORT` creates and uses `pedigree.txt.SRT` if `pedigree.txt` exists but `pedigree.txt.SRT` does not exist
- `pedigree.txt.SRT` also creates and uses `pedigree.txt.SRT` if `pedigree.txt` exists but `pedigree.txt.SRT` does not exist.

Normally ASReml recursively removes base individuals (individuals with unknown parents) that have no data and only one offspring, sometimes called pruning. The `!NOPRUNE` qualifier instructs ASReml not to prune the trimmed pedigree.

The `!UPPER` qualifier was introduced for the case of a pedigree where names have not been recorded consistently with respect to case. All lower case characters are converted to upper case.

The `!CSKIP` c qualifier instructs ASReml to ignore the first c columns of the pedigree file. A pedigree file typically has 3 or 4 fields being the identifiers for the individual, its Sire, its Dam maybe its sex or inbreeding value (f). This is intended to facilitate reading a data file as a pedigree file when the file has c other fields at the beginning of each line.

2.4 GRM file modification

An `!ADD` d qualifier adds a constant $0.000001{*}d$ to nonzero diagonal elements of the GRM matrix supplied. `!NONULL` in conjunction with `!ADD` d to add $0.000001 * d$ to all diagonal elements. This is intended for when when the matrix elements are not of sufficient precision resulting in the matrix not being positive definite.

2.5 Selection index calculations using !GINDEX

This qualifier `!GINDEX` *ModelTerm Coefficients* should be placed after the datafile line and before the model line. The qualifier is motivated by wishing to predict effects that are based on linear combinations of objective traits that are not measured. In principle these objective traits could be included directly in a mixed model but it is sometimes more computationally feasible to calculate the predicted values by using the idea of a selection index and form the index from the predictions of the measured trait. If the i-th index for an individual is $\boldsymbol{\alpha_i u_0}$ with linear combination $\boldsymbol{\alpha}_i$ of objective traits effects $\boldsymbol{u_0}$ then this index can be predicted with $\boldsymbol{b_i}\widetilde{\boldsymbol{u_m}}$ with $\boldsymbol{b_i} = \boldsymbol{\alpha_i G_{om} G_{mm}^{-1}}$ where $\boldsymbol{G_{om}}$ is the covariance between objective and measured trait effects and $\boldsymbol{G_{mm}}$ is the variance of measured trait effects and $\boldsymbol{u_0}$ is the prediction of the effects for the measured traits. Note that this qualifier is envisaged to be used to provide predictions of effects using one round of iteration and using known values of the variance parameters.
For example,

```
harvey.dat !MAXIT 1
!GINDEX us(Trait).nrm(animal),
0.25 0.75 # Index 1
0.70 0.30 # Index2
!ASSIGN INITS 19.96 8.985 117.2
!ASSIGN INITR 134.8 -66.45 655.2
ADG Y3   Trait Trait.line ,
!r us(Trait $INITS).nrm(animal)
residual id(units).us(Trait $INITR)
```

generates predicted values for 2 indices using animal effect predicted values for traits `ADG` and `Y3`. The term `us(Trait)` identifies $\boldsymbol{G_{mm}}$. Forty coefficients can be supplied and the number is a multiple of the number of traits. The computed selection and prediction error standard errors are written to the `.sli` file. Note that the prediction error standard errors are calculated from the prediction of $\boldsymbol{b_i u_m}$. The prediction error of $\boldsymbol{b_i u_m}$ also includes an extra term dependent on $(\boldsymbol{b_i u_m - a_i u_o})$, the extra variation from predicting the indices using measured traits instead of objective traits. This variation depends on $\boldsymbol{G_{oo} - G_{om} G_{mm}^{-1} G_{mo}'}$ but at present this is not calculated by ASReml .