

BLUPF90 - a flexible mixed model program in Fortran 90

(Preliminary)

Ignacy Misztal, Animal and Dairy Science, University of Georgia

e-mail: ignacy@uga.edu tel. (706) 542-0952, fax (706) 583-0274

November 17, 1997 - May 9, 2002

INTRODUCTION

Mixed model packages in animal breeding are either comprehensive and fast but hard to modify or less comprehensive, less efficient but easier to modify. Although the comprehensive packages suit the times when they were developed, they run out of date. For example, none of the packages available in 1994 (see my 1994 WCGALP paper) supported the now-popular random regressions. Adding new features in comprehensive programs is hard because of their complexity. Less comprehensive programs are easier to modify but are less efficient, support only a narrow class of models, and require substantial computer efficiency.

New developments in computer hardware and software relaxes some efficiency requirements on computer programs. With larger memory and faster processor(s), it is actually advantageous to write less efficient but simpler programs. Object-oriented languages such as C++ and partly Fortran 90 allow for creation of classes for common operations so that the number of details in programming can be greatly reduced without sacrificing efficiency. Fortran 90 has an extra advantage to support a variety of matrix operations.

BLUPF90 is a mixed model program written in Fortran 90 (Misztal, 1999) . This program was developed step by step in my graduate level course and focuses on simplicity and model flexibility. It supports general multiple trait models with missing values and different models/trait, and random regressions. Memory allocation is automatic. Sparse matrix operations are supported through an easy-to-use sparse matrix module SPARSEM. Currently, all variables are stored in memory and only solutions are provided; solutions to models with up to a few millions of equations can be obtained. Extra functionality such as prediction error variance and REML variance components for problems of moderate size can be added quickly and without much programming; Please look at nce.ads.uga.edu/~ignacy for existing modifications to BLUPF90.

Computing details

The program is modular. All details of the model are localized a structure called *model*. Storage of the left hand side (LHS) of the system of equations is in matrix structures in memory. Solutions are obtained either iteratively by SOR or exactly by FSPAK. All equations are ordered within traits. Using the sparse matrix storage and solution by iteration, the program needs about 15 bytes for each nonzero of the LHS, which is upper stored, and an extra 12 bytes per equation. With an average of 10 nonzero elements per row, which would correspond to 5 in upper-storage only, the memory requirements would be approximately 87 bytes/equations. Thus in the current version, BLUPF90 can solve about 340,000 equations with 30 Mbytes of memory and almost 6 million with 512 Mbytes. Dense systems of equations would reduce that limit.

Files

Parameter file

BLUPF90 is driven by a parameter file, with a general format as shown below. Keywords in capital need to be included exactly as they appear. Fields in italic are those replaced by the user.

```
# Each keyword can be preceded by comments, each starting with
#
DATAFILE
name of data file
NUMBER_OF_TRAITS
number of traits
NUMBER_OF_EFFECTS
number of effects
OBSERVATION(S)
position of observations in data file (one per trait)
WEIGHT(S)
position of weight(s) in data file (one per trait); blank if
all weights equal
EFFECTS: POSITIONS_IN_DATAFILE NUMBER_OF_LEVELS TYPE_OF_EFFECT
[EFFECT NESTED]
one line per effect:
    position of effect (one per trait)
    number of levels
    type of effect (one of cross for crossclassified or cov
for covariable)
    position of effect where nested (one per trait, optional)

RANDOM_RESIDUAL VALUES
residual variance covariance matrix (full stored)
RANDOM_GROUP
number of correlated effects (order as in EFFECTS above);
correlated effects must be consecutive
RANDOM_TYPE
type of random effect: one of diagonal, add_animal,add_sire,
add_an_upg,
    add_an_upginb or par_domin

FILE
relationship file; blank line if file unnecessary
(CO)VARIANCES
variance covariance matrix for given correlated effects and
traits(full stored)
```

The section starting with “RANDOM-GROUP” is repeated for each random effect or each group of random effects. Each keyword can be preceded by a comment starting with #, however, comments cannot separate (co)variance matrices. See APPENDIX for examples of parameter files.

Different models per trait and missing values

The value 0 is used as a code for missing values for both data and parameter files. If an effect is missing for one trait, assign 0 to its position. For example, assume that trait 1 has a contemporary group on position 1 with 1000 levels and trait 2 has a contemporary group on

position 7 with 2000 levels. This can be coded in the parameter file as:

```
1 0 1000 cross
```

```
0 7 2000 cross
```

and, when effects are specific to one trait, can be merged into one line

```
1 7 2000 cross
```

where the number of levels is greater of the two. A similar merge of two effects is in the example of Appendix B.

Data file

The data file is in free format containing real and integer values only. BLUPF90 expects all effects to be numbered from 1 consecutively. For renumbering, use another program, e.g., RENUMMAT on my anonymous FTP. A renumbering program specific to BLUPF90 called RENUMF90 is available as a binary for Linux.

Random effects and Pedigree files

There are a few types additive effects, each with a different pedigree format.

a) additive sire (add_sire)

The pedigree file has the following format:

```
animal number, sire number, maternal grandsire number
```

where unknown sire and/or maternal grandsire numbers are replaced by 0.

b) additive animal (add_animal)

The pedigree file (add_animal) has the following format:

```
animal number, sire number, dam number
```

where unknown sire and/or dam numbers are replaced by 0.

c) additive animal with unknown parent groups (add_an_upg)

The pedigree file has the following format:

```
animal number, sire number, dam number, parent code
```

where sire and/or dam numbers can be replaced by unknown parent numbers, and parent code = 3 - number of known parents, i.e., 1 (both parents known), 2 (one parent known), and 3 (both parents unknown).

d) additive animal with unknown parent groups and inbreeding (add_an_upginb)

The pedigree file has the following format:

```
animal number, sire number, dam number, inb/upg code
```

where sire and/or dam numbers can be replaced by unknown parent numbers, and

$$\text{inb/upg code} = 4000 / [(1+m_s)(1-F_s) + (1+m_d)(1-F_d)]$$

where m_s (m_d) is 0 whenever sire (dam) is known, and 1 otherwise, and F_s (F_d) is the coefficient of inbreeding of the sire (dam). For example, the inb/upg code for the animal with both parents known is 2000.

e) parental dominance

The pedigree class file has the following format:

```
s-d s-sd s-dd ss-d ds-d ss-sd ss-dd ds-sd ds-dd code
```

where x-y is a combination number of animals x and y, s is sire, d is dam, sd is sire of dam, etc. Code is a number of 0 to 255 and refers to the combination of missing subclasses. If one line is:

p s₈ s₇ s₆ s₅ s₄ s₃ s₂ s₁ code

then code = $\sum(a_i 2^{**i})$, where a_i=0 if s_i=1 and 1 otherwise. For example, the code for a line with all nonzero parental subclasses is 255. For a line with only zero parental subclasses, If classes are ordered so that lines with zero parental subclasses, code=0. If lines are ordered so that p for parental classes with code=0 are ordered last, they may be omitted and will added automatically. The parental dominance file can be obtained from program RENDOMN.

BLUPF90 facilitates inclusion of other relationship matrices, e.g., for autocorrelation

Output file

The solutions are printed in file called "solutions"

Comments and bugs

Many crashes and errors involving the word "hash" result from too few levels defined in the parameter file.

BLUPF90 computes generalized solution by several methods and with different precisions. With different methods of solving, the values of solutions change, but estimable functions should be the same. With different precisions, solutions may be slightly different.

References

Misztal, I. 1997. SPARSEM - sparse matrix module in Fortran 90. Available on my home page (<http://nce.ads.uga.edu/~ignacy/f90> or at <ftp://num.ads.uga.edu>).

Misztal, I. 1994. Comparison of software packages in animal breeding. Proc. 5th World Congress Gen. Appl. Livest. Prod. Vol. 22:3-10. Available on my home page.

I. Misztal. 1990. JAA20 - a mixed model using with iteration on data. *ibid.*

Misztal, I. 1999. Complex models, more data: simpler programming. Proc. Inter. Workshop Comput. Cattle Breed. '99, March 18-20, Tuusula, Finland. *Interbull Bul.* 20:33-42.

L.R. Schaeffer. 1985. Course - Advances in estimating breeding values and population parameters. Technical University Berlin, Germany.

L. R. Schaeffer and J.C.M. Dekkers. 1994. Random regressions in animal models for test-day production in dairy cattle. Proc. World Cong. Genet. Appl. Livest. Prod., University of

Guelph, Canada, 19:443-446.

Appendix A

Single trait “USDA-type” animal model. This example is from the documentation of program JAA20.

$$y_{ijkl} = hys_i + hs_{ij} + p_k + a_k + e_{ijkl}$$

where

y_{ijkl} - production yield
 hys_i - fixed herd year season
 hs_{ij} - random herd x sire interaction
 p_k - random permanent environment
 a_k - random animal

and

$$\text{var}(hs_{ij}) = .1, \text{var}(p_k) = .05, \text{var}(a_k) = .5, \text{var}(e_{ijkl}) = 1$$

Data file (ic)

Format: animal/hys/p/hs/y

```
1 1 1 1 10
2 1 2 1 11
3 2 3 2 15
4 2 4 3 13
5 3 5 4 14
6 3 6 3 12
```

Relationship file (is)

Format: animal/dam/sire/code

```
1 12 8 2
2 1 8 1
3 2 9 1
4 7 10 1
5 12 11 2
6 1 10 1
7 13 14 3
8 5 11 1
9 13 8 2
10 7 14 2
11 13 14 3
```

Parameter file

```
# Example of single-trait animal model with one fixed effect
DATAFILE
```

```
ic
```

```
NUMBER_OF_TRAITS
```

```
1
```

```
NUMBER_OF_EFFECTS
```

```
4
```

```
OBSERVATION(S)
```

```
5
```

```
WEIGHT(S)
```

```
EFFECTS: POSITIONS_IN_DATAFILE NUMBER_OF_LEVELS TYPE_OF_EFFECT
```

```

[EFFECT NESTED]
2 3 cross
3 6 cross
4 4 cross
1 14 cross
RANDOM_RESIDUAL VALUES
1
RANDOM_GROUP
2
RANDOM_TYPE
diagonal
FILE

(CO)VARIANCES
.1
RANDOM_GROUP
3
RANDOM_TYPE
diagonal
FILE

(CO)VARIANCES
.05
RANDOM_GROUP
4
RANDOM_TYPE
add_animal_upg
FILE
is
(CO)VARIANCES
.5

```

Execution

```

/home/ignacy/f90/examples blupf90
name of parameter file?exiap

```

```

BLUPF90 1.00

```

```

Parameter file:          exiap

```

```

Data file:              ic

```

```

Number of Traits        1
Number of Effects       4
Position of Observations 5
Position of Weight (1)  0
Value of Missing Trait/Observation 0

```

```

EFFECTS
# type                position (2)      levels
[positions for nested]
1 cross-classified    2                 3
2 cross-classified    3                 6
3 cross-classified    4                 4
4 cross-classified    1                 14

```

Residual (co)variance Matrix
1.000

Random Effect 1
Type of Random Effect: diagonal
trait effect (CO)VARIANCES
1 2 0.100

Random Effect 1
Type of Random Effect: diagonal
trait effect (CO)VARIANCES
1 3 0.050

Random Effect 1
Type of Random Effect: additive animal
Pedigree File: is

trait effect (CO)VARIANCES
1 4 0.500

REMARKS

(1) Weight position 0 means no weights utilized
(2) Effect positions of 0 for some effects and traits means
that such effects are missing for specified traits

Data record length = 5

original G
0.10

inverted G
10.00

original G
0.05

inverted G
20.00

original G
0.50

inverted G
2.00

solutions stored in file: "solutions"

/home/ignacy/f90/examples cat solutions

trait/effect level solution

1	1	1	11.8589
1	1	2	13.7539
1	1	3	14.7086
1	2	1	-0.0088
1	2	2	0.0088
1	2	3	-0.0159
1	2	4	0.0159
1	2	5	0.0321
1	2	6	-0.0321
1	3	1	0.0000
1	3	2	-0.0079
1	3	3	-0.0081
1	3	4	0.0161
1	4	1	-1.7627

1	4	2	-0.9553
1	4	3	1.4288
1	4	4	-0.9206
1	4	5	-1.0781
1	4	6	-2.3474
1	4	7	0.8511
1	4	8	-0.1521
1	4	9	3.8926
1	4	10	-2.7717
1	4	11	0.8528
1	4	12	-3.1911
1	4	13	7.9976
1	4	14	-6.3340

Appendix B

Example of multiple trait sire model (from L.R. Schaeffer notes of 1985).

Models

Trait 1: $Y_{1i} = h_i + s_{1j} + e_{1ijk}$
Trait 2: $Y_{2i} = h_i + s_{2j} + e_{2ijk}$

where

h - fixed herd
s - random sire

and

$\text{var}(s) = A \begin{bmatrix} 10 & 10 \\ 10 & 20 \end{bmatrix}$, $\text{var}(e) = I \begin{bmatrix} 8 & 6 \\ 6 & 17 \end{bmatrix}$

Data file (lrsdat)

Format: h:/s/Y₁/Y₂

```
1 0 1 3.4 0
2 0 2 1.3 0
1 1 3 .8 50.3
2 1 4 4.5 52.6
0 1 5 0 55.0
```

Pedigree file (lrsrel)

Format: bull/sire/MGS

```
1 3 0
2 0 5
3 0 0
4 0 0
5 0 0
```

Parameter file (lrsex)

Example of two trait sire model with unequal models

DATAFILE

lrsdat

NUMBER_OF_TRAITS

2

NUMBER_OF_EFFECTS

2

OBSERVATION(S)

4 5

WEIGHT(S)

EFFECTS: POSITIONS_IN_DATAFILE NUMBER_OF_LEVELS TYPE_OF_EFFECT

[EFFECT NESTED]

1 2 2 cross

3 3 5 cross

RANDOM_RESIDUAL VALUES

10 10

10 20

RANDOM_GROUP

2

RANDOM_TYPE

add_sire

FILE
lrsrel
(CO)VARIANCES
8 6
6 17

Execution

/home/ignacy/f90/examples blupf90
name of parameter file?lrsex

BLUPF90 1.00

Parameter file: lrsex

Data file: lrsdat

Number of Traits 2
Number of Effects 2
Position of Observations 4 5
Position of Weight (1) 0
Value of Missing Trait/Observation 0

EFFECTS

#	type	position (2)	levels
1	cross-classified	1 2	2
2	cross-classified	3 3	5

[positions for nested]

Residual (co)variance Matrix

10.000	10.000
10.000	20.000

Random Effect 1
Type of Random Effect: additive sire
Pedigree File: lrsrel

trait	effect	(CO)VARIANCES	
1	2	8.000	6.000
2	2	6.000	17.000

REMARKS

(1) Weight position 0 means no weights utilized
(2) Effect positions of 0 for some effects and traits means
that such effects are missing for specified traits

Data record length = 5

original G
8.00 6.00
6.00 17.00
inverted G
0.17 -0.06
-0.06 0.08

solutions stored in file: "solutions"

/home/ignacy/f90/examples cat solutions

trait	effect	level	solution
1	1	1	2.3877
2	1	1	52.4449
1	1	2	3.2180
2	1	2	0.0000
1	2	1	0.2243
2	2	1	-0.0210
1	2	2	-0.8217
2	2	2	-0.3866
1	2	3	-0.4969
2	2	3	-0.7512
1	2	4	0.6178
2	2	4	-0.0769
1	2	5	0.2217
2	2	5	1.0851

Appendix C

This test-day model example comes from the paper of Schaeffer and Dekkers (WCGALP94 18:443)

Model

$$y_{ijkl} = h_i + \$_1 X_{1j} + \$_2 X_{2j} + a_k + (_{1k} X_{1j} + (_{2k} X_{2j} + e_{ijkl}$$

where

y_{ijkl} - yield of test day

h_i - test day effect

X_{1j} - days in milk

X_{2j} - log(days in milk)

$\$_1, \$_2$ - fixed regressions

a_k - random animal

$(_{1k}, (_{2k}$ - random regressions for each animal

and

$$\text{var}(e_{ijkl}) = 1; \text{var}(a_k, (_{1k}, (_{2k}) = [2.25 \ 4 \ -.7; 4 \ 1375 \ 12; \ -.7 \ 12 \ 94]^{-1}$$

Data file (lrsrrdat)

Format: h/a/ X_1 / X_2 /y

```
1 1 73 1.42985 26
1 2 34 2.19395 29
1 3 8 3.64087 37
2 1 123 0.908127 23
2 2 84 1.28949 18
2 3 58 1.65987 25
2 4 5 4.11087 44
3 1 178 0.538528 21
3 2 139 0.785838 8
3 3 113 0.992924 19
3 4 60 1.62597 29
4 2 184 0.505376 1
4 3 158 0.657717 15
4 4 105 1.06635 22
4 5 14 3.08125 35
5 3 218 0.335817 11
5 4 165 0.614366 14
5 5 74 1.41625 23
5 6 31 2.28632 28
6 3 268 0.129325 7
6 4 215 0.349674 8
6 5 124 0.90003 17
6 6 81 1.32586 22
```

Relationship file (lrsrrrel)

Format: animal/sire/dam

```
1 9 7
2 10 8
3 9 2
4 10 8
5 11 7
6 11 1
7 0 0
8 0 0
```

```
9 0 0
10 0 0
11 0 0
```

Parameter file (exlrsrr)

Example of single-trait random-regression model

DATAFILE

lrsrrdat

NUMBER_OF_TRAITS

1

NUMBER_OF_EFFECTS

6

OBSERVATION(S)

5

WEIGHT(S)

EFFECTS: POSITIONS_IN_DATAFILE NUMBER_OF_LEVELS TYPE_OF_EFFECT

[EFFECT NESTED]

1 6 cross

3 1 cov

4 1 cov

2 11 cross

3 11 cov 2

4 11 cov 2

RANDOM_RESIDUAL VALUES

1

RANDOM_GROUP

4 5 6

RANDOM_TYPE

add_animal

FILE

lrsrrrel

(CO)VARIANCES

```
.447906   -0.001334   0.003506
-0.001334   0.000732  -0.000103
 0.003506  -0.000103   .010678
```

Execution

/home/ignacy/f90/examples blupf90

name of parameter file?exlrsrr

BLUPF90 1.00

Parameter file: exlrsrr

Data file: lrsrrdat

```
Number of Traits          1
Number of Effects         6
Position of Observations   5
Position of Weight (1)     0
Value of Missing Trait/Observation 0
```

EFFECTS

```
# type          position (2)      levels
[positions for nested]
1 cross-classified      1                6
```

2	covariable	3	1	
3	covariable	4	1	
4	cross-classified	2	11	
5	covariable	3	11	2
6	covariable	4	11	2

Residual (co)variance Matrix

1.000

correlated random effects 4 5 6
 Type of Random Effect: additive animal
 Pedigree File: lrsrrrel

trait	effect	(CO)VARIANCES		
1	4	0.448	-0.001	0.004
1	5	-0.001	0.001	0.000
1	6	0.004	0.000	0.011

REMARKS

- (1) Weight position 0 means no weights utilized
- (2) Effect positions of 0 for some effects and traits means that such effects are missing for specified traits

Data record length = 5

original G

0.45	0.00	0.00
0.00	0.00	0.00
0.00	0.00	0.01

inverted G

2.25	4.00	-0.70
4.00	1375.09	11.95
-0.70	11.95	94.00

solutions stored in file: "solutions"

/home/ignacy/f90/examples cat solutions

trait/effect level solution

1	1	1	19.9496
1	1	2	20.3729
1	1	3	20.6095
1	1	4	19.7278
1	1	5	18.6035
1	1	6	17.8500
1	2	1	-0.0498
1	3	1	5.2912
1	4	1	-0.4430
1	4	2	0.2704
1	4	3	-0.7288
1	4	4	1.1019
1	4	5	-0.1626
1	4	6	-0.4828
1	4	7	-0.0988
1	4	8	0.4574
1	4	9	-0.6288
1	4	10	0.4574
1	4	11	-0.1872
1	5	1	0.0369
1	5	2	-0.0661

1	5	3	0.0068
1	5	4	-0.0054
1	5	5	0.0069
1	5	6	0.0167
1	5	7	0.0133
1	5	8	-0.0238
1	5	9	0.0350
1	5	10	-0.0238
1	5	11	-0.0008
1	6	1	-0.0370
1	6	2	0.0325
1	6	3	-0.0479
1	6	4	0.0767
1	6	5	-0.0149
1	6	6	-0.0377
1	6	7	-0.0103
1	6	8	0.0364
1	6	9	-0.0480
1	6	10	0.0364
1	6	11	-0.0145

Appendix D

This model was used for studies on multibreed evaluation in beef cattle. It is provided as an example of a model with maternal effect and different models per trait.

Model (in concise form, with most indices omitted)

$$\begin{aligned}y_1 &= cg_1 + bt + mbt + a + M + e \\y_2 &= cg_2 + bt + mbt + a + M + pe + e \\y_3 &= cg_3 + bt + mbt + a + e\end{aligned}$$

where

y_{1-3} - birth weight, weaning weight, and gain
 cg_{1-3} - contemporary groups separate for each trait
br - breed type
mbt - maternal breed type
a - additive effect
m - maternal effect
pe - permanent environmental effect of the dam

Data file (data.out)

Format:

1. contemporary group for trait 1
2. contemporary group for trait 2
3. contemporary group for trait 3
4. animal breed type
5. maternal breed type
6. animal id
7. dam id
8. birth weight
9. weaning weight
10. gain

Relationship file (pedi.outok)

Format:

animal
sire or unknown parent group
dam or unknown parent group
"1 + number of missing parents"

Parameter file (exlrsrr)

DATAFILE

data.out

NUMBER_OF_TRAITS

3

NUMBER_OF_EFFECTS

6

OBSERVATION(S)

8 9 10

WEIGHT(S)

EFFECTS: POSITIONS_IN_DATAFILE NUMBER_OF_LEVELS TYPE_OF_EFFECT

[EFFECT

NESTED]

1 2 3 133085 cross

4 4 4 181 cross

5 5 0 165 cross

6 6 6 1724112 cross

7 7 0 1724112 cross

0 7 0 1724112 cross

RANDOM_RESIDUAL VALUES

26.3 0.0 0.0

0.0 1312.9 0.0

0.0 0.0 1246.3

RANDOM_GROUP

4 5

RANDOM_TYPE

add_an_upg

FILE

pedi.outok

(CO)VARIANCES

22.9 36.3 18.6 -4.6 0.0 0.0

36.6 500.2 110.8 0.0 -91.6 0.0

18.6 110.8 313.0 0.0 0.0 0.0

-4.6 0.0 0.0 10.1 0.0 0.0

0.0 -91.6 0.0 0.0 419.1 0.0

0.0 0.0 0.0 0.0 0.0 0.0

RANDOM_GROUP

2

RANDOM_TYPE

diagonal

FILE

(CO)VARIANCES

0.263 0.0 0.0

0.0 13.129 0.0

0.0 0.0 12.463

RANDOM_GROUP

3

RANDOM_TYPE

diagonal

FILE

(CO)VARIANCES

0.263 0.0 0.0

0.0 13.129 0.0

0.0 0.0 0.0

RANDOM_GROUP

6

RANDOM_TYPE

diagonal

FILE

(CO)VARIANCES

0.0	0.0	0.0
-----	-----	-----

0.0	45.5	0.0
-----	------	-----

0.0	0.0	0.0
-----	-----	-----

Appendix D

A single-trait random regression model for test-day milk is using cubic Legendre polynomials.

Model

$$y_{ijkl} = \text{hym}_{ij} + \sum_{m=1}^4 \alpha_m(l)h_{im} + \sum_{m=1}^4 \alpha_m(l)u_{km} + \sum_{m=1}^4 \alpha_m(l)p_{im} + e_{ijkl}$$

where

y_{ijkl} - test day milk

hym_{ij} - herd-year-test for herd i and year-test j

h_i - effects of herd i

$\alpha_m(l)$ - value of m -th Legendre polynomial at point corresponding to $\text{DIM}=l$

u - additive effects

p - permanent environmental effects

Data file (datarr)

Format:

1. herd
2. herd-year-test
- 3-6. values of Legendre polynomials
7. weight for residuals: $100/\text{var}(e_{ijkl})$
8. test day
9. animal

Relationship file (pedirr)

Format:

animal
sire
dam

Parameter file (exrr3)

DATAFILE

datarr

NUMBER_OF_TRAITS

1

NUMBER_OF_EFFECTS

13

OBSERVATION(S)

8

WEIGHT(S)

7

EFFECTS: POSITIONS_IN_DATAFILE NUMBER_OF_LEVELS TYPE_OF_EFFECT

2 3726 cross #herd-year-test

3 84 cov 1 #herd

4 84 cov 1

5 84 cov 1

6 84 cov 1

3 21874 cov 9 #additive
4 21874 cov 9
5 21874 cov 9
6 21874 cov 9
3 21874 cov 9 #pe
4 21874 cov 9
5 21874 cov 9
6 21874 cov 9

RANDOM_RESIDUAL VALUES

100

RANDOM_GROUP

6 7 8 9

RANDOM_TYPE

add_animal

FILE

pedirr

(CO)VARIANCES

(4 x 4 matrix)

RANDOM_GROUP

10 11 12 13

RANDOM_TYPE

diagonal

FILE

(CO)VARIANCES

(4 x 4 matrix)

Appendix F

Design of the program

The model is completely described in the module MODEL.

```

module model
implicit none

!      Types of effects
integer,parameter::effcross=0,& !effects can be cross-classified
                    effcov=1    !or covariables

!      Types of random effects
integer, parameter :: g_fixed=1,&    ! fixed effect
                    g_diag=2, &    ! diagonal
                    g_A=3, &      ! additive animal
                    g_A_UPG=4, &   ! additive animal with unknown
                                !   parent groups
                    & g_A_UPG_INB=5, & ! additive animal with unknown
                                !   parent groups and inbreeding
                    & g_As=6,&    ! additive sire
                    g_PD =7, &    ! parental dominance
                    g_last=8     ! last type

character (40)      ::  parfile, &    !name of parameter file
                    datafile    !name of data set

integer :: ntrait,&          !number of traits
         neff,&             !number of effects
         miss=0            !value of missing trait/effect

integer,allocatable :: pos_y(:)      !positions of observations
integer ::          pos_weight      ! position of weight of records; zero if none

integer,allocatable :: pos_eff(:,:),& !positions of effects for each trait
                    nlev(:),&       !number of levels
                    effecttype(:),& !type of effects
                    nestedcov(:,:),& !position of nesting effect for each trait
                                ! if the effect is nested covariable
                    & randomtype(:),& ! status of each effect, as above
                    randomnumb(:)    ! number of consecutive correlated effects

character (40),allocatable:: randomfile(:) ! name of file associated with given
                                         ! effect

real, allocatable :: r(:,:),&      !residual (co)variance matrix
                    rinv(:,:),&   ! and its inverse
                    g(:,:,:)      ! The random (co)variance matrix for each trait
end module model

```

The core of the program is presented below.

```

program BLUPF90
use model;use sparsem; use sparseop
implicit none
real,allocatable :: y(:),&          ! observation value
                    indata(:)      ! one line of input data

real ::          weight_y          ! weight for records

type (sparse_hashm)::xx            ! X'X in sparse hash form
type (sparse_ija):: xx_ija        ! X'X in IJA form, for use with FSPAK only
real, allocatable:: xy(:),sol(:)  !X'Y and solutions

real,allocatable :: weight_cov(:,:)
integer,allocatable:: address(:,:) ! start and address of each effect
integer :: neq,io,&                ! number of equations and io-status
         data_len,&                ! length of data record to read
         i,j,k,l                    ! extra variables
real:: val, dat_eff

!
call read_parameters
call print_parameters
neq=ntrait*sum(nlev)
data_len=max(pos_weight,maxval(pos_y),maxval(pos_eff))
print*,'Data record length = ',data_len
allocate (xy(neq), sol(neq),address(neff,ntrait),&

```

```

        weight_cov(neff,ntrait),y(ntrait),indata(data_len))
call zerom(xx,neq); xy=0
!
call setup_g                ! invert R matrices

open(50,file=datafile)      !data file

! Contributions from records
do
  read(50,*,iostat=io)indata
  if (io.ne.0) exit
  call decode_record
  call find_addresses
  call find_rinv
  do i=1,neff
    do j=1,neff
      do k=1,ntrait
        do l=1,ntrait
          val=weight_cov(i,k)*weight_cov(j,l)*weight_y*rinv(k,l)
          call addm(val,address(i,k),address(j,l),xx)
        enddo
      enddo
    enddo
    do k=1,ntrait
      do l=1,ntrait
        xy(address(i,k))=xy(address(i,k))+rinv(k,l)*y(l)*weight_cov(i,k) &
          *weight_y
      enddo
    enddo
  enddo
enddo
!
! Random effects' contributions
do i=1,neff
  select case (randomtype(i))
  case (g_fixed)
    continue                ! fixed effect, do nothing
  case (g_diag)
    call add_g_diag(i)
  case (g_A, g_As, g_A_UPG,g_A_UPG_INB)
    call add_g_add(randomtype(i),i)
  case (g_PD)
    call add_g_domin(i)
  case default
    print*,'unimplemented random type',randomtype(i)
  endselect
enddo

if (neq < 15) then
  print*,'left hand side'
  call printm(xx)
  print '( ' ' right hand side:'' ,100f8.1)',xy
endif

  call solve_iterm(xx,xy,sol)

! Comment the line above and uncomments the lines below only if
! solutions by FSPAK are desired
!xx_ija=xx;
!call fspak90('solve',xx_ija,xy,sol)

if (neq <15) print '( ' ' solution:'' ,100f7.3)',sol
call store_solutions

```

Warning

When the program is modified to solve for different left hand sides repeatedly by FSPAK, e.g., in variance component procedures, always factorize before solving:

```

call fspak90('factorize',xx_ija)
call fspak90('solve',xx_ija,xy,sol)

```

The solving step assumes that the first factorization is intact, and subsequent solves involve

the initial factorization.

Appendix F

Extensions to the parameter file

BLUPF90 can be modified so that the parameter file can contain additional lines with the following format:

```
OPTION name str1 str2 ....
```

This line(s) can be read by subroutine

```
call getoption(name,n,x,xc)
```

where str1, str2 are strings separated by spaces. If the line with a given name cannot be located, n=-1. Otherwise, the subroutine assigns: xc(1)=str1, xc(2)=str2,...and attempts to decode strings into real values: x1=value(str1),...; n contains the number of strings. x and xc are optional and their dimensions may be smaller than n in which case some strings/values are not stored. Upon exit, unit 40 points to the next line in the parameter file to the one located.

Example

Suppose that BLUPF90 was modified to add an autoregressive effect. Details for that effect are specified in the following parameter line:

```
OPTION autoregressive effect 2 rho .82 variance 500.0
```

Then, assuming declarations:

```
integer :: n, x(20)
```

```
character (10) :: xc(20)
```

the call

```
call getoption('autoregressive',n,x,xc)
```

will result in n=6, xc(1)='effect', xc(2)='2', xc(3)='rho',..., and x(1)=0, x(2)=2.0, x(3)=0, x(4)=.82, ..., x(6)=500.0.