PEST User's Manual

Eildert Groeneveld

Institute of Animal Husbandry and Animal Behaviour
Federal Agricultural Research Centre (FAL)
31535Neustadt 1
Hoeltystr.10
Germany

e-mail: groeneveld@tzv.fal.de
tel: 05034/871155 (Germany)
fax:05034/871143

CHAPTER 1 Introduction

PEST is a software package for multivariate prediction and estimation. It covers fixed, random and mixed models. PEST reads raw data and translates class codes like month names integer or character identification of animals into internal representation. Identifications can be up to 16 characters long. A parameter file contains commands to PEST. It consists of input data description, the statistical model, output, data transformations etc. Wherever possible meaningful defaults are used. Three modes of treatment of the mixed model equations are available. They are (A) sparse storage of coefficients in memory, (B) Gauss-Seidel/Jacobi iteration on data with complete storage of diagonal blocks, and (C) Gauss-Seidel iteration on data with storage of level elements only. Memory requirements decrease from the first to the latter while CPU requirements increase. Because all three types can be combined in one model run placing some effects into part A while large effects like litter and animal can be placed in B or C. This allows most efficient usage of a given amount of memory with many possibilities to tune setting up and solving mixed linear models. PEST has been written at the Department of Animal Sciences at the University of Illinois, USA, by Eildert Groeneveld, Milena Kovac and Tianlin Wang. The following publications are relevant in regard to algorithms and scope of PEST:

- Eildert Groeneveld and Milena Kovac. A Generalized Computing Procedure for setting Up and Solving Mixed Linear Models. 1990, Journal of Dairy Science 73:513-531

- Eildert Groeneveld, Milena Kovac, and Tianlin Wang. PEST, a general purpose BLUP package for multivariate prediction and estimation. Proceedings of the 4th World Congress on Genetics applied to Livestock Production, 1990, Edinburgh, 488-491
- Eildert Groeneveld, Rod Bunge, Tianlin Wang and Rohan L. Fernando, Hypothesis Testing in Multivariate Mixed Models and its Implementation in PEST, 42nd Annual Meeting of the European Association of Animal Production, Berlin, Sept. 8-12, 1991
- Eildert Groeneveld, Milena Kovac, Tianlin Wang, and Rohan L. Fernando. Computing algorithms in a general purpose BLUP package for multivariate prediction and estimation. Arch. Tierz. 35 (1992), 399-412
- Eildert Groeneveld, Milena Kovac. Performance Characteristics of Different Solving Strategies in Multivariate Mixed Models. Livestock Production Science, 30 (1992) 319-331
- Eildert Groeneveld, Milena Kovac, Tianlin Wang and Rohan L. Fernando. Computing algorithms in a general purpose BLUP package for multivariate prediction and estimation. Archives of Animal Breeding, 35 (1992) 4, 399-412

Further features of PEST are:

- treatment of missing values
- different incidence matrices for some or all traits
- any number of fixed and random effects
- any number of covariables
- polynomials up to order 20
- inclusion of relationship among animals
- treatment of inbreeding
- genetic group models
- heterogeneous variances
- dumping of old solutions to be used for presetting the solutions vector for following runs as required in routine prediction of breeding values.
- setting the average of breeding values for base parents to zero.
- tests of hypothesis for fixed and mixed uni and multi variate models.
- treatment of animal, sire, sire dam models.
- Prediction error variances for BLUPs and standard errors of BLUEs
- covariance matrix of fixed and random effects

CHAPTER 2 System Requirements

2.1 System Requirements

PEST is written in FORTRAN 77. It consists of around 15000 lines of source code in 250 routines.Ported version exist for SUN workstations, VAX under VMS, MACINTOSH for the ABSOFT compiler, IBM mainframes under CMS, and CRAYs under UNIX.These versions should be ready to compile and run. All system dependent routines are included.

For (relatively) easy porting to other machines systems dependent instructions are concentrated in a few routines. These are timing routines and a routine that allows submission of the parameter file name as a parameter behind the program name.

Solving large sets of mixed model equations may be computationally challenging. This is particularly true for multiple trait models. As accuracy is of some importance all computations are done in extended precision on 64 bits. Furthermore, memory requirements may be substantial as large buffers may need to be stored. PEST provides a very general and thus flexible strategy to solve mixed model equations. Its objective has not been to minimize memory requirements, (which is becoming less of an issue at a remarkable pace).Together with the large amount of program code PEST does require a certain minimum amount of processing capability.

Before installation the buffer size to be used by the package has to be decided on. Its dimension is set in the main programs in kilobytes. This is the only declaration a user has to make. PEST does its own memory management at run time. It may be useful to have executable modules with different buffer sizes: small problems could be solved by a 3 or 5 MB version while a large of 15 or 20 MB version could deal with very large problems comprising hundreds of thousand equations.The program code without data buffers requires around 400 kB to execute. With data buffers added this a too large to run on DOS systems. Further, it should be noted that most computations are performed on 64 bit reals which are often much slower than 32 bit floating point operations. The smallest machines to run PEST successfully are PC's under OS/2, the Macintosh with at least 2 MB memory, and the lower end UNIX systems like SUNs SPARCstation and their clones.PEST reads raw data and codes it as required for further processing. In this process temporary files are created, which may be substantial in size depending on the original file size. Thus, sufficient disk space has to be available.

2.2 Installation

A complete distribution comprises the following files:

- MANUAL.ps
- PEST.fortran
- README
- SYNTAX.ERR
- b_pest
- comp.list
- course.dat
- de_mpa.dat
- de_mpa.ped
- hb_el.dat
- hb_el.ped
- ihan.dat
- mid.dat
- mid.ped
- mid800.dat
- mu_mo.dat
- olistp1 through olistp30
- p1 through p30
- verify.f

The filenames will differ depending on the file naming conventions of the respective operating system. The complete PEST source code is contained in the file PEST.f provided this is a validated port for one the following computers/operating systems:

- SUN under SUN OS

- VAX under VMS
- HP 9000 under UNIX
- MAC under MAC OS with ABSOFT fortran
- CRAY under UNICOS
- IBM under CMS
- DOS with 386 and numerical coprocessor. This version comes as an executable file and does not require a FORTRAN compiler for installaton.

A complete installation consists of the following steps:

- adapt source code
- compile and link PEST
- adapt parameter files
- run all 30 pfiles
- check output against olistpn

2.2.1 Adaptation of source code

As described above PEST does its own memory management dynamically at runtime. The buffer pool is based on only two vectors IV and CV which are equivalenced. Equivalencing character and non character variables is not supported by the FORTRAN 77 standard but is supported on the vast majority of compilers. If the compiler does not support this feature installation of PEST will be impossible.

The size of this buffer has to be adjusted at installation in accordance with the real memory on the target machine.Small problems can be solved with a buffer of 1 MB or less. Large problems like national genetic evaluation of breeding programs may require values of 20MB. The following lines contain the parameters needed for installation:

```
program pest
integer*4 nwords, kb, int4 , real4 , real8
parameter (kb         =    3*1024)
parameter (int4       =    4)
parameter (real4      =    4)
parameter (real8      =    8)
parameter (nwords =(kb*1024)/8)
character*8      cv(nwords)
real     *8      rv(nwords)
equivalence     (cv,rv)
```

The variable KB has to be set by the installer. To make this a 20 MB version the ‘3’ has to be replaced by a ‘20’, to make this a .5 MB version the expression ‘3*1024’ should be replaced by ‘512’ as some compilers do not allow mixed mode arithmetic in parameter statements.

2.2.1.1 file handling

Default files to be used by PEST are all initalized in the ‘block data fnamin’. Most of the open statements are concentrated in the routine OPENFI (the exceptions are the parameter file and the file SYNTAX.ERR which contains error descriptions). The EFFLIST file is accessed in direct access mode. The record length that is used in OPENFI is in

terms of bytes. Some filesystems specify this length in words. Thus, at installation time this value has to be checked. In the case of LRECL being the record length in words and the word length being4 bytes a line should be inserted before the open that reads LRECL=LRECL/4.

PEST uses a variety of files to store results. The default files are specified in the BLOCK DATA FNAMIN subprogram. The standard setting for UNIX systems is:

```
 data filena   /'    ', '                    ',
*'                   ', '                    ',
*'                   ',
*'                   ', '                    ',
*'list.pest'         ,'                      ',
*'datafile.pest', 'relation.pest',
*'efflist.pest',     'mem_map.pest',
*'                   ', 'prediction.pest',
*'/home/pest/SYNTAX.ERR'/
```

If these entries are not appropriate for your system, change them to appropriate values.Of particular importance is the location of the file SYNTAX.ERR. It is accessed once an exception is detected by PEST. It should therefore reside in an directory where each user has read access. This would normally be the same directory that contains the PEST executable. All other files in the block data statement will be created locally in the current directory of the user who started PEST. As PEST creates the binary files and EFFLIST with the same default names **PEST should not be started twice from the same directory** while the first job is still running. The second run may delete the original EFFLIST and binary files with unpredictable results.

2.2.2 Compilation

As PEST is rather large compilation may take a long time. It might be necessary to increase the size of the symbol table. Consult your compiler handbook for details. It is important not to forget the optimization option, as this may dramatically speed up execution.

Some compilers have an option that generates code to check if indexes do not exceed the defined dimensions of vectors and matrices. This option must not be invoked as boundaries are deliberately exceeded.

For compilation the static memory model should be chosen where possible.

2.2.3 Adaptation of parameter files

Parameter files contain the control statements that parameterize PEST. The are identical on all computers. Filename, however, do have to be changed to the system requirements. As a general rule, the character strings between the quotes in infile and outfile statements are passed unchanged to the FORTRAN open statement. Thus, they have to

contain the filenames as required by the respective FORTRAN/operating system. Furthermore, the filenames have, obviously, to match those of the actual data files. Under UNIX one filename may be ‘coursedata’ while on an IBM under CMS it may have to be ‘/COURSEDA DAT A’.

2.2.4 Run parameter files and verify correct execution

After all parameter files have been adapted to the current system run each pfile one by one.

The output produced has names vlistp1 through vlistp30. The file verify.batch contains the commands to run the complete suite for UNIX systems.

After the test suite has been processed the original list files and the newly created have to be compared.This is supported by the program verify.f which is distributed with Unix versions. Compile and run it. VERIFY takes its input from a file called listfiles which contains the name of the listfiles to be processed, however, without the ‘v’ or ‘o’ prefix. VERIFY creates the file ‘pestcomp’ which contains all lines with discrepancies. Thoroughly investigate it. For each run a few discrepancies are to be expected as some of the timing information will be printed. This has been left in on purpose, as to give an idea how your machine compares to the SUN4 used to generate the originals.

Some of the solutions may be different which may not indicate a problem. If the target machine uses a different format for the representation of real numbers results may be different for iterative procedures on the third decimal digit. This should not be of any concern.

2.2.5 Ports to new operating systems

As indicated above verified ports exist for five operating systems. The following is intended as a help for those who want port PEST to another operating system.

2.2.5.1 System dependent routines

The following routines operating system dependent and should be checked:

- **dtime**
- **time**
- **fdate**
- **ior**
- **getarg**

dtime is subroutine that returns the cumulative CPU in seconds for the current process. It is called:

```
real*4 cputime
....
call dtime (cputime)
```

The integer function **time** returns the system time in seconds from some arbitrary base. It is used to determine the total time of a job spent in the computer. If it is replaced by a function that always returns the same number the wall clock time listed will always be 00:00:00.

The routine **fdate** returns in a character string of length 24 the current time and date. The calling sequence is:

```
character*24 date
...
call fdate (date)
```

On a SUN workstation date contains: “Sat Sep 1 12:50:36 1990”. The format of this date is irrelevant but will be printed out on the list output.

If no timing or date routines exist on the target computer they can be replaced by dummy routines that return a constant:

```
subroutine dtime (time)
real*4 time
time = 0
return
end
```

and for the date routine:

```
subroutine fdate (string)
character*24 string
string = '****** IIII ******'
return
end
```

Finally the wallclock time:

```
function time ()
integer time
time = 0
return
end
```

The hashing routine performs requires a logical AND and logical exclusive OR. No FORTRAN default exists for these routines. Many compilers support logical operators of the following kind:

```
 m = bit(n) .and. j
loc = loc .xor. bit(32-n)
```

while others require:

```
m = iand(bit(n),j)
loc = ieor (loc, bit(32-n))
```

Check with your compiler hand book to find out which method is supported.

The **GETARG** routine is used to pass the command line parameter on to PEST. On some systems (like UNIX) PEST may be called like:

```
>pest pfile
```

pfile being the parameter file containing the PEST language commands. **GETARG** is called in the routine edgm as:

```
character*80 pfname
.....
call getarg (1,pfname)
```

Under SUN OS (UNIX) it returns the first parameter after the program name itself, i.e. for the above example it would return in pfname the string ‘pfile’. If your system does not support return of command line parameters you can replace getarg by the following dummy subroutine:

```
subroutine getarg (no,string)
integer*4 no
character*(*) string
string = ‘ ‘
return
end
```

If STRING is empty the user will be prompted by PEST to enter the parameter file name:

```
orion{eg}44: pest
         ************************************************
         *                    P E S T                   *
         *     Multivariate Prediction and ESTimation   *
         *               10.0 MB version                *
         *                                              *
         *     E. Groeneveld, M. Kovac, and T. Wang     *
         *         Department of Animal Sciences        *
         *            University of Illinois            *
         *           Wed Sep  5 09:08:22 1990           *
         ************************************************
Enter pfile name, please :
```

2.2.6 Comments about installations on various machines

The following is a list of problems that were encountered while porting PEST to the various platfoms. They are just a collection and by no means complete.

2.2.6.1 MacIntosh

The ports were made with the ABSOFT compiler, version 2.4. When starting the compiler certain options have to be chosen:

☒ M 68020/68030
☒ P 68881/68882
☒ H STATIC
☒ U * = UNIT 9

For a production version of PEST do not create a symbol table and not the N option :'Display cards at runtime'.This will only slow down execution.

The file SYNTAX.ERR should best be put in the system folder. Remember that filenames are case sensitive on the Macintosh. Thus, SYNTAX.ERR should be in capital letters, the way it is defined in block data FNAMIN subprogram described above.

After having compiled PEST you need to set the application size. Highlight the PEST icon and press 'apple I'. This will bring up a window that describes the application. At the bottom is a field that contains the application size which should be around 300kb. This value does not include any buffers. Thus, if you specified prior to compilation 2000kb in PEST you should enter into that box 2400. If this number is too small PEST will abort with the rather terse message: Application has unexpectedly quit.

2.2.6.2 IBM CMS

Contrary to most other systems IBM FORTRAN is not content with the standard FORTRAN 77 OPEN statement as it requires a FILEDEF statement. Unfortunately, IBM FORTRAN or standard CMS does not have a routine that allows it to issue a FILEDEF statement. Apparently, there are different routines floating around, some seem to be user written, others apparently can be purchase from IBM. The routine name for issueing FILEDEFs chosen in PEST is CMSFNC. It has the format:

```
call cmsfnc (length(strfil),strfil,ier)
```

STRFIL is a character variable containing the FILEDEF statement, the first parameter contains its length in number of characters, while ier return the error code. The installer has to ensure that this routine is available.

Unfortunately, the IBM FORTRAN is sensitive to fixed and variable length records. The README file on the distribution medium contains a whole set of copy commands that changes all parameter files to fixed length record types. One such command for parameter file p1.s.a is:

```
copy p1 s a = = = (recfm f lrec 140 replace
```

Likewise, if data files (and relationship files) are to be read in free format they also have to be of a fixed length type with a record length of 140.

The compiler used in the port was IBMs FORTVS2.

2.2.6.3 CRAY under UNICOS

Installation on the CRAY should be rather unadventurous. Compile with the following option:

```
cf77 -m4 PESTCRAY.f -o PEST
```

The -m4 suppresses warnings that are otherwise will flood the screen. These are declaration of the form INTEGER*4 variable which are not ANSI standard and, thus, result in a warning.

Memory requirements on a CRAY are larger for the same model and data set than on other machines. The reason is that on the CRAY an integer word - the way memory allocation is handled in PEST - uses 64 bits instead of 32.

CHAPTER 3 The User Interface

3.1 General remarks

The parameter file (pfile) drives PEST. It contains all the specifications required to run PEST. A pfile is created by the user via his/her favorite text editor in either lower or upper letters. A pfile is divided into SECTIONS which may be grouped in any order.

The user interface is designed such that as little as possible has to be specified by the user to run various statistical analysis. Table 1 gives an example of a complete parameter file which performs a multivariate genetic evaluation of boars for the traits backfat and daily gain including pedigree information. Thus, implementing a genetic evaluation amounts to creating a file - say pfile1 -, entering the 31 line in table 1 with an editor and invoking PEST by typing pest pfile1. After the job has completed the list file will contain BLUEs for the fixed effects liveweight, month of test, and society, and BLUP's for each boar for the traits backfat and daily gain. If it is decided to modify the model by also fitting a regression of daily gain on live weight the corresponding term has to be added to the last line in pfile1 and PEST has to be run again. Thus, different models can be tested very rapidly.

TABLE 1 Sample parameter file.

```
comment sample_job
  This is a sample job performing a multivariate genetic evaluation
  of boars for the traits backfat and daily gain with different
  incidence matrices. Data and pedigree file are read in free format.
  Defaults are used.
data
  infile = 'input.dat'
  input
     boar             800
     month_of_test     15
     liveweight         0
     backfat            0
     daily_gain         0
     society            6
relationship
  rel_for boar
  infile = 'pedig.d'
  input
     boar
     m_p
     f_p
ve
    2.2   32.3
   32.2 3200.
vg
  vg_for boar
    1.2   10.4
   10.4 2240.
model
   backfat    = live_weight(society) month_of_test society boar
   daily_gain =                      month_of_test society boar
```

The general command structure of PEST is given in Chapter 7. Wherever possible defaults are being used. The parameter file is used to modify these defaults. Thus, if the syntax in the pfile is not correct, the defaults will not be changed and the faulty command be discarded. A parameter file consists of section names, key words, options and user specified information.

Each SECTION - written in bold capital letters throughout the text and in Chapter 7 - describes a certain aspect connected to performing a fixed or mixed model analysis: the data section describes all input variables, the model section the statistical model. These two are the only mandatory sections.All variables referenced somewhere in the parameter file have to be defined, i.e.read in in the data section. A section always starts in the first column and is followed by a body of text which consists of key words, options and user supplied input. Sections can be written in any order.

KEYWORDS - also written in bold capital letters throughout the text and in Chapter 7 - start anywhere beginning at column 2. They either change a default to a different state like the DISK key word or they are either followed by a textstring specified by the user (given in italic) like in INFILE = '*datafile*'. The equal sign can be omitted.

OPTIONs are appended to user input by starting with one of the following characters: { [(. They are convenient means of adding more specific information to a command as in the OUTFILE = '*filename*' [TEXT case.

Commands in square brackets [] are optional and can thus be omitted. In these cases defaults will take effect which are underlined in Chapter 7.

Separating options and key words by a vertical bar | means to choose only one from the list. A bracket { means that the enclosed term can be repeated. Wherever a n or constant is given in the syntax the user has to specify a numerical value.

A 'C' in the first column indicated a comment lines which can be inserted anywhere in the parameter file. It is ignored by PEST and gives the user the possibility to add whatever comments are deemed useful.

In the following each section will be described.

3.1.1 Sections

Each section starts in the first column with a legal section name. These are:

- COMMENT
- RELATIONSHIP
- DATA mandatory
- MODEL mandatory
- HYPOTHESIS
- PRINTOUT
- SOLVER
- SYSTEM_SIZE
- STARTING_VALUES
- VE
- VG

Only the DATA and MODEL sections are mandatory.

3.1.2 Keywords

Each section may have a number of KEYWORDS. These start anywhere between and including column 2 and 132 and may or may not be followed by parameters. Keywords are protected words that must not be used as variable names within the section they be-

long to. They can be written in upper or lower case or a mixture of both just like the section names. Key words are:

abs_avg_change
abs_max_change
base_zero
birthdate
c11
canonical
cholesky
chisquare
contrast
dense
disk
f_p
group
hetero_in
inbreeding
infile
input
int
ioc
iod
iod_gs
k1a1 through k50a10
k1c1 through k50c10
line
memory_map
m_p
max_char
max_iter
max_iter_iod
max_iter_iod_gs
non_zero
outfile
page
pev
rel_for
relax
relax_iod
relax_iod_gs
scaling
sign_digits
smp
stand_avg_change
stand_max_change
stop
stop_iod
stop_iod_gs
treated_as_missing

3.1.3 Filenames

Wherever file names are required these have to comply with the specifications for FORTRAN open statements of the respective operating systems. All filenames are put between quotes. The string between the quotes is transferred unchanged to the FORTRAN open statement. Comments may be introduced anywhere in the pfile by starting the line with a ‘c’ or ‘C’ in the first column. Furthermore, blank lines can be inserted anywhere.

PEST operates nearly exclusively on the symbolic names for traits and effects. The link to data is made only in the data section linking the name to the data columns. Names can be as long as 32 characters, they may include any printable character except ‘.

3.1.4 Overriding defaults

The parameter file is scanned repeatedly during the initial phase of processing in PEST. Amongst others built in defaults are overridden by the content of the parameter file. If keywords are not spelled correctly, they will not be identified and, thus, the defaults will not be overridden. This is important to know if results are not as expected.

3.2 The Sections

COMMENT

3.2.1 COMMENT Section

The comment section is intended for documentation purposes for each run. It has the following form:

[**COMMENT** [*header*]]
An unlimited number of lines can be added.
Only the first five will be printed in the list file.

The *header* is printed as part of the header on every page in the print output. It can be used as convenient reminder of the specific run. Up to five lines of the following comment block are printed on the fist page of the list output. The user can supply any information deemed desirable to have on the list output. Comments in excess of 5 lines are skipped. The comment section aisles useful to document the job described in the pfile. An example may be:

COMMENT PROJECT_1
This is an implementation of an animal model for the three traits backfat, age and number of piglets born. The data set is derived from field test of boars and litter recording schemes for gilts. It composes data from the years 1980 through 1988. The model includes LITTER, HERDS, and ANIMAL as random effects and LOCATION, WEIGHT, and BREED as fixed effects. Data are generated by the SQL procedure extract_2.sql.

The string PROJECT_1 will be printed in the heading of each page. The first five lines will, furthermore, be printed in the comment block on the first page of the list output. The COMMENT section can be omitted.

3.2.2 DATA Section

DATA

The data section defines the structure of input data and as such is mandatory. Its general format is:

DATA
INFILE = *filename*
INPUT
{*variable_i no_of_levels* [*starting_column*,*length* [,*decimal_digits*]]}
[**MAX_CHAR** = 8|16]
[**DISK**]
[**OUTFILE** = *filename* [<**TEXT**|**BINARY**]

It has the keywords: INFILE, INPUT, OUTFILE, and DISK the latter two being optional. PEST converts the input data on the basis of information supplied in the data section into a format required for setting up and solving the mixed model equations. This is a two step process: firstly, data are read on the basis of the format specifications. Those traits and effects specified in the model and transformation section are then recoded. During this process a considerable amount of memory is required. This will however, usually not exceed the requirements for setting up and solving the system of equations in the case of animal models. With sire models this may not be so. As an output a data file is produced that serves as input to PEST in the later stages. As a default PEST uses the filename “datafile.pest”. Specifying the keyword OUTFILE = filename will use “filename” instead.Its simplest format is:

```
DATA
    INFILE =' input.data'
    INPUT
            animal          1000
            herd             200
            month             12
            liveweight         0
            backfat            0
            daily_gain         0
```

This format implies a data file with the name input.data. It contains ASCII or EBCDIC data in free format, i.e. columns separated by either blanks or comma. The columns are given the names animal, liveweight, backfat and daily_gain. The column behind the variable names in the file gives the number of levels for each variable. This number does not have to be exact, it does have to be large enough to accommodate the actual number of levels. If a too small number is chosen the coding process will be aborted and an appropriate message issued. It should be noted that in the case of including the relationship the number of levels in the data section has to be large enough to also accommodate the relatives that may not have records. In other words: if the data file contains records on 1000 animals and the pedigree file contains 1500, the number of levels specified in the data section has to be at least 1500. Continuous variables get a 0 value.Effects are interpreted as character strings. This allows use of numeric and non numeric values like names for animals or months names. The data file could look like:

```
Paul      0001      January      123.      12.1      671
John      0020          May      119.      15.6      789
```

If numeric values are to be used for one variable they should have leading zeros. This ensures appropriate sorting. This format can be achieved by writing this column in the FORTRAN format (I4.4).The input statement should contain all variables in the data file. This allows easy changes in the model definition later on.Where data are not available in free format additional data have to be supplied in DATA section. Assume the following data set:

```
Paul  0001  January      123.      12.1      671
John  0020      May      119.      15.6      789
```

the data section looks like:

```
DATA
      INFILE =' input.data'
      INPUT    MAXLEVEL          START      LENGTH DECIMALS
      animal   1000                  1           5
      herd     200                   8           4
      month    12                   12           7
      liveweight0                   19           5        0
      backfat  0                    24           4        1
      daily_gain0                   28           3        0
      age      0                    31           3        0
      breed    5
```

It should be noted that PEST will read fixed format much faster than free format. Thus, with larger data sets always fixed format should be used.With iterative procedures data has to be read repeatedly. With, say 3 effects in either IOD or IOD_GS, the complete data file has to be read 3 times for each round of iteration. As a default data are stored in memory and read from there. This the fastest procedure. However, if the data set is large relative to available memory, it may not be practical. Specifying the keyword DISK in the data section leads to data being read from disk. This may drastically reduce memory requirement at the expense of increased processing time. The appropriate data section would read:

```
DATA
      INFILE =' input.data'
      DISK
      INPUT    MAXLEVEL          START      LENGTH DECIMALS
      animal   1000                  1           5
      ......
```

The DISK keyword is optional and has no parameters. Its format is:

```
DISK
```

This switches the default from reading from memory to disk. If the DISK keyword is specified sufficient disk space has to be available. The data format is binary with only those effects and traits that are specified in the model section. The performance degradation depends very much on the speed of the external memory, i.e. disk. It will be worthwhile to inquire into the fastest disk available in the computer configuration. Some systems have solid state disks (RAM disks) where access times come close to those from and to memory. In these cases reading from DISK could be the overall most efficient option.

The INFILE keyword is mandatory. Its format is:

```
INFILE = 'FILENAME'[recoded]
```

The option “recoded” is optional. Class effects have to be recoded, i.e. each of the effects has to be coded numerically starting at 1. If data are available in this format the data preparation part of PEST is performed faster.

The OUTFILE keyword is optional with the format:

```
OUTFILE = 'FILENAME'[TEXT]
```

If the option TEXT is specified an output file in ASCII or EBCDIC format will be produced which can be inspected by any text editor.

Internally, missing values are represented by a string of eight '1'. These are also written to the text file. Thus, if further use of the recoded data is intended, the user should be aware of this fact.!

The MAX_CHAR keyword allows to specify the maximum length of the identification. Its format is

MAX_CHAR = 8

The default is 16 characters. If 8 is specified the format in the data section must not specify a longer field than 8. In this case no free format is allowed as this always assumes a maximum field length on 16. The effect of specifying 8 characters instead of 16 is a reduction in memory and disk requirements which may be crucial in very large problems.

If MAX_CHAR=8 is used labels from the relationship file also must not be longer than 8 characters. Furthermore, its data input section has to have a fixed format.

HYPOTHESIS

3.2.3 HYPOTHESIS Section

A variety of hypothesis can be specified in the Hypothesis Section. The hypothesis are of the general type:

$$K'b-m \qquad \text{(EQ 1)}$$

[**HYPOTHESIS**]
{{**CONTRAST** = *constant* | **0**}
{*address constant*}}
[CHI]
[PEV]
[C11 number]
{OUTFILE = *filename*|**c11.pest {binary}}**
{FORMAT '(legal fortran format|10F12.5)'}

The following tests and their side conditions are possible:

1. univariate fixed and mixed models based of an estimate of the error variance performing an F test.
2. univariate fixed and mixed models using the user supplied residual variance for an Chi square test
3. multivariate fixed and mixed model using the user supplied residual variance for an Chi square test
4. a contrast may consist of up to 50 elements (this limit may be increased prior to compilation time), no limit with second format.
5. no limits exist on the number of contrasts and their elements
6. all effects have to be placed in SMP.

Hypotheses are specified by a set of keywords specifying the addresses in the mixed model equations and the coefficients that are used in the contrast. Assume the following output from PEST: a multivariate analysis has been performed for the two traits backfat and daily gain. The year equations are the first equations in the mixed model:

```
 ........................ PEST UIUC V2.6     ......................
Wed Aug 21 16:01:54 1991  P4                                page  4
    YEARS                 BACKFAT    DAILYGAIN

    81                     0.319      -21.145
    82                    -0.156      -12.868
    83                    -0.058      -16.437
    84                    -0.061       -7.410
    85                     0.000        0.000
    86                     0.023       24.192

    HERD                  BACKFAT    DAILYGAIN

    0001                  10.156      727.868
    0002                  13.000      656.500
    0003                  10.298      542.385
    0004                   9.399      538.951
    0005                   9.200      566.000
    0006                   9.503      539.611
```

A composite hypothesis is to be tested: the difference between year 81 and the average of years 82 through 85 for backfat is equal to zero and the difference between herd 2 and 3 for daily gain is 30 g. The hypothesis section for this test would be:

hypothesis
c... first contrast:
contrast = 0
1 1
3 -1/4
5 -1/4
7 -.25
9 -.25
c..second contrast
contrast = 30.
16 1.
18 -1.

The resultant output for this particular data set and the covariance matrix supplied is:

```
    ---------------------------------------------------------------

    H y p o t h e s i s  T a b l e  :
    ________________________________

  Contrast  MME address Coefficients Deviations
 ____________________________________________________
     1            1     1.0000
     1            3    -0.2500
     1            5    -0.2500
     1            7    -0.2500
     1            9    -0.2500

       ____________________________________________________________
       Test value [MU1]    :        0.000000
       Estimated value     :        0.391163 +-       0.148589
       ____________________________________________________________

     2           16     1.0000
     2           18    -1.0000

       ____________________________________________________________
       Test value [MU2]    :       30.000000
       Estimated value     :      115.590983 +-      40.732798
```

```
Prob (Chi_S >   11.346) =  0.0034

Covariance of Estimators:

1  0.2208E-01        0.
2      0.          1659.
```

The first contrast among the years gives an estimate of .39 mm backfat. The second contrast between herd 2 and 3 for trait daily gain is 115.6 g. The composite hypothesis that the difference between year 81 and the following 4 is zero and that the difference between herd 2 and 3 is 30 g daily gain is rejected at the .3% level on the basis of an Chi square test.

A test that the effect of 1981 equals that of 1982 in a univariate model would be written in the hypothesis section as:

```
hypothesis
c addresses of first contrast (year 81 - 82)
    contrast = 0
    1   1.
    2  -1.
```

The addresses to be specified are the sequential numbers from the PEST list output. In multiple trait models, the numbering is done within levels, starting with the first trait, continuing with the second and so on.

The corresponding output is:

```
    H y p o t h e s i s  T a b l e  :

Contrast  MME address Coefficients Deviations
    1            1     1.0000
    1            2    -1.0000

Test value [MU1]    :        0.000000
Estimated value     :        0.477263 +-       0.166619

Prob (F >    8.205) =  0.0043
Resid.stand.dev.    :        1.475523

Covariance of Estimators:
 1  0.2776E-01
```

This test is based on the F statistic with the residual variance estimated from the data. In this case the residual standard deviation of backfat is 1.48mm.

The keyword **CHI** specifies a Chi-square test in univariate fixed and mixed models based on the supplied residual covariance. Note, if no VE is specified a value of 1. will be assumed for the residual variance.

The prediction error variances (actually their square root) of BLUPs and standard errors of the BLUEs can be computed by specifying the keyword PEV in the hypothesis section. The computational efford will be substantial, as the system of equations has to be solved as tdimx times, with tdimx being the total number of equations. Therefore, this may be practical only for relatively small systems. The PEVs are appended to the solutions in the list output file, one PEV for each trait.

Sometime, the covariance matrix of the fixed effects are useful to have. They can be computed by specifying the following:

```
HYPOTHESIS
        C11 11
        outfile my.c11
        format '(''c11..'',11F15.8)'
```

The keyword C11 specifies that the inverse of the coefficient matrix be computed for the first 11 equations and columns. If **number** is ommitted, the complete inverse will be generated. Thus, if the user wants to have the inverse of the coefficient matrix pertaining to the fixed part of the system (i.e. C_{11}) he/she should do the following:

- place the fixed effects first in the model section.
- specify the number of equations under number.

Again, it should be remembered that the system of equations has to be solved as many times as indicated by **number**. If number is not specified the default will take effect, which means that the inverse of the complete coefficient matrix will be generated.

In some cases it might be useful to be able to further process the inverse by some piece of program. In this situation the **binary** option may be useful. The resulting file cannot be read by a texteditor as it contains only binary information. For each equation one record is written. Notice that the data are in extended real format (double precision or real*8).

Default output is a ascii/ebcdic for which a fortran format may be specified.

If no **outfile** is specified the results will be written to the default **c11.pest**.

MODEL

3.2.4 MODEL Section

The model section is mandatory as it describes the statistical model. Its format is:

MODEL
{*traitname_i* = [**INT**] {*independent_variable*} [<**PREDICT**]}

The model section consists of one line (equation) for each trait in a multivariate analysis. Analyzing backfat with effects herd and month and the covariable liveweight is specified as:

```
MODEL
        backfat = liveweight month herd
```

Because liveweight is specified in the data statement as having zero levels it is automatically taken to be a covariable. If an additional intercept is to be fitted the model statement becomes:

```
MODEL
        backfat = int liveweight month herd
```

or

```
MODEL
        backfat = liveweight month herd [int
```

A multiple trait analysis is specified by a second line:

```
MODEL
        backfat = liveweight month herd
        dailygain = liveweight month herd
```

Multiple incidence matrices are automatically generated by specifying only those effects that are to be defined for each trait. In the above model it does not make too much sense to specify liveweight as a covariable for daily gain as it is a part of the trait. It may be more appropriate to adjust for a constant age. The model would look like:

```
MODEL
        backfat = liveweight month herd
        dailygain = age month herd
```

If a genetic evaluation is to be performed with an animal model the model could become:

```
MODEL
        backfat = liveweight month herd animal
        dailygain = age month herd animal
```

A quadratic regression is fitted by specifying:

```
MODEL
        backfat = liveweight liveweight*liveweight month herd animal
```

Polynomials of up to order 20 can be specified.

Nesting of covariables is specified by writing:

```
        backfat = liveweight(breed) breed month herd animal
```

or

```
        backfat = liveweight*breed breed month herd animal
```

The same notation is used for specifying interaction:

```
backfat = liveweight breed month herd breed*herd animal
```

All effects can be nested. Nesting dams within sires can be written as:

```
yield = dam(sires)
```

PEST translates nested effects into interactions, which are statistically identical. While this is a very flexible setup, it does generates more equations which will also turn up on the list output.

Interactions can be specified up to an order of twenty.

Predicted values may be calculated by appending the option PREDICT behind the last effect for each trait equation. Only traits with the PREDICT option will be predicted. The output is written to the file PREDICTION.PEST. The format is:

```
backfat = liveweight*breed breed month herd animal [predict
```

An intercept can be fitted by adding the keyword INT to the list of effects:

```
backfat = int liveweight(breed) breed month herd animal
```

PRINTOUT

3.2.5 PRINTOUT Section

The printout section allows specification of print output. Its general format is:

[**PRINTOUT**]
[**OUTFILE** = *filename*|"**LIST.PEST**"]
[**OUTPUT**]
{*traitname_i any_legal_FORTRAN_format*}
[**PAGE** = *n*|**54**]
[**LINE** = *n*|**74**]
[**SIGN_DIGITS** = *n*|**5**]
[**XPX** *format*]
[**XPY** *format*]
[**BASE_ZERO**]
[**MEMORY_MAP**]

The section is optional with defaults that should cover most of printing needs. Data formats are calculated on the basis of the actual results in the solutions vector. They may, however, be overridden by format statements in the print output. All list output will be written to the specified file with the name given after the keyword OUTFILE. If the operating system allows this it can be either a disk file or a printer. If no file is specified LIST.PEST will be assumed.

LINE_LEN and PAGE_LEN specify the page layout. LINE_LEN gives the maximum number of characters in each line, whereas PAGE_LEN is the number of lines after which a page break is performed. On every page break a page heading is printed. Furthermore, column headings for the current effects and traits will be repeated. Thus, if the user is interested only in the output of the solutions for further processing the page length should be set to a very high value. Control L is used as a form feed character.

SIGN_DIGITS The number of digits can be specified globally for all traits by setting SIGN_DIGITS. The default is 4. The format for each single trait can be specified by the keyword OUTPUT followed by the trait name and its format. This will override any defaults or globally specified formats like SIGN_DIGITS. Thus, if it is desired to have 7 total digits on all traits except for one which should have 1 decimal point, one would set the significant digits to seven and specify the trait under output:

```
sign_digits7
output
        dailygain          (f6.1)
```

XPX and XPY lead to the listing of the left hand side and right hand side of the mixed model equations. This is only useful for small examples. A valid FORTRAN format may be specified behind XPX and XPY:

```
XPX    (25F4.1)
```

or

```
XPX    ‘(‘MME :’,25f4.1)’
```

If a character string is to be included as in the last example the complete format has to be surrounded by quotes.

In breeding programs it may be useful to set the average of the base generation to zero. All other solutions for the effect are then modified accordingly. This is done by the keyword BASE_ZERO. This has no effect on the solving process but is performed only prior to printing the results. Base animals are identified in the list output by an appended *.

RELATIONSHIP

3.2.6 RELATIONSHIP Section

The RELATIONSHIP section specifies pedigree data and their use in the model. Its format is:

[**RELATIONSHIP**]
REL_FOR {*effect_name_i*}
INFILE = *filename*
[**MAX_CHAR** = **8**|**16**]
INPUT
ANIMAL [*starting column, length*]
M_P|**M_GP** [*starting column, length*]
F_P|**F_GP** [*starting column, length*]
[**BIRTHDATE**] [*starting column, length*]
[**GROUP**]
[**INBREEDING**]
[**DISK**]
[**UNDEFINED** = *string*]
[**OUTFILE** = *filename*[<**TEXT**|**BINARY**]]

The keyword REL_FOR specifies the effect which is correlated, in this case animal. Pedigree data are stored in file pedig.data with four columns animal, M_P (male parent), F_P (female parent), and BIRTHDATE. The latter is optional and only required if inbreeding is to be considered. This is indicated by the keyword INBREEDING. The

birthdate is taken to be a numerical value of the format YYMMDD or a number that orders the animals according to their age.

Animal, parent, grandparent models can also be evaluated by specifying the **M_GP** for male grandparent and **F_GP** for the female grandparent. The relationship file has to contain the respective codes.

Animals always have to come first under INPUT whereas the other variables can be in any order. This variable can also be one of the variables listed in the data and model section, such as SIRE.

Sire dam model are specified by the following line in the relationship section:

RELATIONSHIP

rel_for sire dam

Obviously, both sire and dam have to be specified in the model section. Some restrictions apply to sire dam models: both sire and dam have to be together in one solver which must not be IOD_GS. If sire dam do not have a common pedigree file no special restrictions exist.

A genetic group model is specified by the keyword GROUP. In this case it is the responsibility of the user to assign animals to the desired groups. No unknown parents are allowed in this case, i.e. every record must have a complete entry. For each unknown parent a genetic group has to be assigned by the user.

Without genetic groups animals in the pedigree file may contain undefined parents.The default of ‘0’ can be changed to any string by the keyword UNDEF as shown above.

As a default pedigree data are read from memory during setting up and solving the mixed model equations. If this should become a bottle neck the keyword DISK will lead to reading data from disk, which may lead to a much increased computation time if either solver IOD or IOD_GS is used.

Remember that the identification of animals in the data file have to be contained in the pedigree file. The comparison is done on character strings as defined in the input format. Should this not be identical animals from the data file will not be found in the pedigree tables. An appropriate message is issued during coding of the data file.

SOLVER

3.2.7 SOLVER Section

The solver section deals with the choices of solvers, the distribution of effects of the three parts of coefficient storage, stopping criteria, relaxation parameters, and maximum number of iterations allowed. Its format is:

[**SOLVER**]
[**IOC**|**SMP**|**DENSE**] [{*effect_name_i*}] [<**STOP** = *n*, **RELAX** = *n*, **MAX_ITER** = *n*]
[**IOD**] [{*effect_name_i*}] [<**STOP** = *n*, **RELAX** = *n*, **MAX_ITER** = *n*]

[**IOD_GS**] *effect_name* [<**STOP** = *n*, **RELAX** = *n*, **MAX_ITER** = *n*]
{[**STOP**|**STOP_IOC**|**STOP_IOD**|**STOP_IOD_GS** = *n* |.001]}
{[**RELAX**|**RELAX_IOC**|**RELAX_IOD**|**RELAX_IOD_GS** = *n* |1.0]}
{[**MAX_ITER**|**MAX_ITER_IOC**|**MAX_ITER_IOD**|**MAX_ITER_IOD_GS** = *n* |1000]}
[**ABS_MAX_CHANGE**|**ABS_AVG_CHANGE**|**STAND_MAX_CHANGE**|**STAND_AVG_CHANGE**]

Available solvers are , thus, SMP, IOC, DENSE, IOD, and IOD_GS. The first three are used for storage of all coefficients in memory, while the last two pertain to effects that are to be solved by iteration on data.If the solver section is empty or does not exist, IOC with all effects specified in the model is assumed as the default. This implies half storage of nonzero elements in memory and subsequent solving of the system by Gauss-Seidel. If sufficient memory is available to store all coefficients for the specified model this is usually the fastest option to solve a set of mixed model equations. Thus, the section:

```
SOLVER
      IOC
```

is equivalent to no solver section, as it invokes the default setting of placing all effects into the coefficient part.

Iterative procedures require stopping criteria. Four stopping criteria are available as listed in Chapter 7. The default criterion used by PEST is the absolute maximum change of the solutions from one round to the next. It may be specified by the key word ABS_MAX_CHANGE in the solver section. The other criteria are the standardized maximum change (STAND_MAX_CHANGE) which results in the same number of significant digits in multivariate evaluations irrespective of the differences in magnitude of the numerical values of the traits. Maximum and standardized average change are specified by the key words ABS_AVG_CHANGE and STAND_AVG_CHANGE. The default value of the stopping criterion of .001 can be overridden by specifying a different value under the solver section. It specifies the number of significant digits, .001 would be four significant digits, whereas it would four decimal digits for the two absolute stopping criteria. The type of stopping criterion is globally applied to all equations that are solved iteratively.

The maximum number of iterations can be specified for each of the solvers IOC, IOD and IOD_GS as indicated in Chapter 7. The default ‘end of round relaxation’ factors can also be overridden. They can be either specified as key words or, preferably, as an option appended to the solvers.

As stated above IOC is an iterative procedure and as such approaches the final solution only asymptotically. If nothing else is specified default stopping criterion, relaxation parameters and maximum number of rounds are used. These three values can be specified as:

```
SOLVER
      IOC
      stop = .001                    ]
      relax=1.2
      MAX_ITER=1000
```

This sets the absolute stopping criterion to .001, i.e. the iterative process will be stopped if the largest absolute change of all solutions from one round to the next is smaller than .001. RELAX and max_iter are set accordingly.

A more convenient way to specify parameters for iteration is in the use of options by giving the parameters with the same name (stop, relax and max_iter) behind each solver and its effects:

```
SOLVER
        IOC       [stop = .1, relax=1.2, max_iter=299
        IOD_GS    [stop= .2, relax=1.3
```

The stopping criterion should be chosen in accordance with the purpose of the respective run. If only the ranking of animals is required accuracy can be smaller. On the other hand if genetic trends are to be evaluated accuracy has to be higher. Depending on the accuracy chosen quite a few rounds may be needed until convergence is reached. 1000 rounds are not uncommon. This number of rounds required may also vary considerably with the model used. No hard rules can be given for the choice of the relaxation parameter. It depends largely on the particular system of equation to be solved. Values around 1.2 to 1.3 sometime give drastic reduction in the number of rounds needed, sometimes halving them if compared to a relaxation factor of 1.0 (which, of course, is the original Gauss-Seidel procedure).

The maximum number of rounds should be chosen in accordance with the desired accuracy. Unless rapid depletion of a CPU account is an issue, MAX_ITER should be left at around 1000 or 2000 rounds.

SMP is the direct solver from the Yale sparse matrix package. Its advantage over iterative procedures is that the solution produced is a direct converged result. On the other hand it is usually more costly in terms of both memory and CPU requirements. As it is a direct procedure stopping criteria, relaxation parameters and maximum number of iterations are meaningless. The direct sparse solver is specified as:

```
SOLVER
        SMP
```

DENSE specifies another direct solver, based on complete storage of all coefficients including zeros. It is, thus, limited to small problems. Advantages? It is invoked by writing:

```
SOLVER
        DENSE
```

IOC, SMP and DENSE are mutually exclusive.Of the three solvers that operate on stored coefficients IOC is most efficient in regard to memory utilization. Also, for most of the applications where BLUP's are of interest it is fastest. If completely converged solutions are required, SMP will be appropriate.

Two solvers can be chosen for iteration on data. They are IOD and IOD_GS. IOD specifies Gauss-Seidel iteration on data. It, thus, requires less memory than any of IOC, SMP, and DENSE. IOD stores the complete diagonal blocks of all effects. If the relationship is included the procedure becomes Jacobi iteration on data. It should be noted that this procedure may not converge, particularly in multiple trait models. Some or all effects may be declared for IOD. To specify solving all effects under IOD specify:

```
SOLVER
        IOD
```

Iteration on data requires the data to be read once for each effect. This includes intercepts and covariables, although each may consist of one equation only. This is why intercepts and covariables will always by put into part A and solved by IOC (unless specified otherwise). As IOC is usually the fastest procedure effects with a rather small number of levels, i.e. equations, should be put into part A or conversely, only those effects should be put in IOD that are too large to be accommodated in part A. This why the syntax assumes specification of those effects behind IOD that are to be solved by this procedure. For the above analysis ANIMALS should be placed there with the other effects being solved by IOC. This is specified by:

```
SOLVER
        IOD animal
```

The general rule is by default effects are solved by iteration on coefficients half stored in memory. Those effects specified for solvers IOD and IOD_GS will be taken out of IOC and solved where specified with the exception of intercepts and covariables which will always automatically be put back into IOC.

IOD_GS is a pure Gauss-Seidel solver with iteration on data. Only the one diagonal block is stored at a time. Thus, its memory requirements are minimal. Only one effect is allowed in this part. This will usually be an effect with a large number of levels, like animal. Relationships can be included. The specification of an effect for this solver is:

```
SOLVER
        IOD_GS animal
```

In conjunction with the above model this statement would place the animal effect into the most memory efficient part C treating the other effects in part A with coefficients stored in memory and all equations except the animal equations being solved by iteration on coefficients. With two large effects like litter and animal in genetic evaluations in swine the solver section could be written as:

```
SOLVER
        IOD litter
        IOD_GS animal
```

This would place the two large effects in the memory saving iteration on data, while all other fixed effects - which are usually much smaller - would be solved simultaneously.

STARTING_VALUES

3.2.8 STARTING_VALUES Section

The starting_values sections allows to specify input and out relating to presetting the solutions vector. Its format is:

[**STARTING_VALUES**]
[**OUTFILE** = *filename*]
[**INFILE** = *filename*]

Iterative procedures give the possibility to preset the solutions vector before the solving process is commenced. In routine genetic evaluations all data that have been accumulated so far are usually reprocessed. Only a limited amount of new data is added. This is

particularly true in species where continuous selection takes place like in swine or chicken. Presetting the solutions vector to solutions obtained in the previous round can dramatically cut down processing time. Before loading of previous solutions can be done, old solutions have to be dumped to disk. This is performed by the following sequence in the parameter file:

```
STARTING_VALUES
    OUTFILE = filename
```

Loading these solution is initiated by:

```
STARTING_VALUES
    INFILE = filename
```

If PEST would be used in continuous genetic evaluation in a breeding program, say in weekly intervals, the parameter file would have to be modified prior to each run to reflect the changing filenames for dumping and loading:

```
STARTING_VALUES
    INFILE = last_weeks_solutions
    OUTFILE= this_weeks_solutions
```

The models must not change from one run to the next and the amount of data used must not be less. If this is necessary, the genetic evaluation has to start with an empty solutions vector, i.e. no starting value file should be specified for input.

The data required for initializing the solutions vector are stored in three files. The first file contains information about the model and the date and time it was run. This file has the original name specified under outfile. The second file contains the identification labels of all levels. depending on the keyword MAX_CHAR these are either eight or sixteen characters long. Thus, reducing MAX_CHAR to 8 will halve the size of this file (and indeed the corresponding memory during execution of PEST). The file is the same except a '1' is appended. The third file containing the solutions is identified by an appended '2'. All three files are required at loading and have to reside in the same directory.

SYSTEM_SIZE

3.2.9 SYSTEM_SIZE section

This section determines the number of non zero elements expected for the mixed model equations. Its format is:

[**SYSTEM_SIZE**]
NON_ZERO = *constant*

This section is only considered if some or all equations are set up in memory. Setting non_zero to a exceedingly high value may exhaust available memory and termination of PEST. If it is set to a value lower than the actual number of non zero elements during setup of the MME PEST will also stop. If this section is omitted and some or all of the equations are set up in memory PEST estimates the expected number of non zero elements on the basis of the system size. This estimated number may be either too large or too small. In the latter case PEST would terminate processing. If this condition occurs

the SYSTEM_SIZE section should be introduced in the parameter file and an estimate used for non_zero. If this value is set, it will always take precedence over the estimate of PEST. After PEST has successfully completed it writes, amongst others, the actual number of non zero elements to the list file. This value can be used in the parameter file for further runs. PEST uses hashing to store the non zero coefficients. A very tight buffer because of small NON_ZERO may lead to poor performance during this process. Increasing its size will help in such situations. The effect of unnecessarily high values of NON_ZERO may, on the other hand, lead to thrashing or swapping on machines with a too little real memory. In such cases the actual number of non zero elements reported in the General Information Table on the list output should be used to determine a sensible value for the non zero keyword. If, as an example, the default calculates a value of 300,000 and the actual number is 82134 the following entry in the parameter file would prevent waste of memory:

```
SYSTEM_SIZE
        non_zero = 83000
```

TRANSFORMATION

3.2.10 TRANSFORMATION Section

The TRANSFORMATION section allows some manipulation of data as they are read in. Its format is:

[**TRANSFORMATION**]
[**TREATED_AS_MISSING**]
{*traitname_i lower_bound,missing_value,upper_bound*}
[**SCALING**]
{*traitname_i*-|+|*|/ *constant*}
{*covariate_i*-|+|*|/ *constant*}
[**CHOLESKY**]
[**CANONICAL**]

Key words are TREATED_AS_MISSING, SCALING CHOLESKY, and CANONICAL. The former allows the definition of what values are to be considered missing values. In a multiple trait situation the correct residual covariance matrices will be generated corresponding to the pattern of missing values. The format is:

```
Treated_as_missing
    TraitnameLower Bound        Point Upper Bound
```

For each trait in the model, and only a trait, not a covariable, can missing values be defined. The lower and upper bound specify values if undercut or exceeded will be treated as missing. The middle value gives one value which if encountered in the data is considered missing. The following example sets lower and upper bounds for feed conversion efficiency:

```
treated_as_missing
    FCE     2.0 .0 3.6
```

Only values larger 2.0 and smaller than 3.6 will be considered legal. The next example takes values between -2 and +100 as legal values but also declares 99 illegal:

```
treated_as_missing
        trait       -2                          99 100
```

If no upper and lower bounds are to be imposed and only one zeros are to be excluded one can write:

```
treated_as_missing
        trait       none 0. none
```

This is also the default setting.

The keyword SCALING allows arithmetic operations on each continuous trait in the data. Its format is:

```
scaling
    weight -130
```

This would subtract 130 from each weight. Other legal operands are addtion and multiplication.

When covariables are used in the model it should always be scaled to a value of around zero, leading to a potentially much faster convergence. Thus, if the covariable is body weight with values around 130kg scale the variable as indicated above.

As also the coefficient of variation (CV) is based on scaled data, its value can be nonsensical, depending on the type of scaling. If its value is too large to fit the print format a ??? will be printed instead and an explanatory comment given at the bottom of the DATA INFORMATION table.H

The keyword CHOLESKY performs a Cholesky decomposition in multivariate models. This may lead to reduced overall processing time in multiple trait models. If the model has different incidence matrices Cholesky decomposition cannot and will not be performed.

The keyword CANONICAL performs canonical transformation on data. This can and will be done only for restricted models: one random effect, no missing values, no heterogeneous variances, no multiple incidence matrices. The present version does use the same amount of data storage as for the multivariate analysis. However, convergence may be much faster.

VE

3.2.11 Residual Variance Section

Multi variate analysis requires specification of the residual covariance structure. This is done in the VE section. The format is:

[**VE**]
covariance_matrix | **HETERO_IN** *effect_name*
{**VE_FOR** *effect_level_i*

covariance_matrix_i}

A square matrix of the order of number of traits in the model has to be specified. The first element corresponds to the residual variance of the first trait, the second entry in the first row is the residual covariance between the first and the second trait, and so on.

Heterogeneous residual covariances are also specified in this section. It requires specification of the effect for which heterogeneous variances are known. E.g. residual heterogeneous variances for each breed in the model are specified as:

```
VE
        HETERO_IN breed
        VE_FOR  DUROC
                1.2                     2.3
                2.3                     5.6
        VE_FOR  HAMPSHIRE
                1.3                     2.5
                2.5                     6.0
        VE_FOR  LANDRACE
                1.1                     2.3
                2.3                     6.7
```

Residual covariances for each level of the effect has to be specified!

VG

3.2.12 Variances of Random Effects Section

Covariances for the random effects are specified much the same as above. Its format is:

[**VG**]
{**VG_FOR** *effect_name_i*
covariance_matrix_i}

The additive genetic covariance for animals would be written as:

```
VG
        VG_FOR animal
                1.2                     2.3
                2.3                     5.6
```

In a sire model the same covariance matrix could be used, dividing each value by 4:

```
VG
        VG_FOR sire
                1.2/4                   2.3/4
                2.3/4                   5.6/4
```

Variances for other random effects are specified in the same way. No defaults are available for mixed models, thus, the VE and VG sections are mandatory.

CHAPTER 4 Examples of Input/Output

This chapter will deal will aspects of output that PEST generates and what the corresponding parameter file looks like. Discussion will be in the frame work of sample parameter files which can be found on the distribution medium.

4.1 The parameter file

TABLE 2

The first parameter file P1 is a sire model for swine data:

```
COMMENT P1
SIRE Model with the following data
 0082 0001 0001 '12-DEC-82' 176.0 126.0 10.0  8.9 715.0
 0075 0211 0002 '16-DEC-85' 213.0 132.0 12.0 10.5 619.0
 123456789.123456789.123456789.123456789.123456789.123456789.
DATA
 INFILE = 'coursedata'
 INPUT
  sire      111  1 4
  dam      1111  6 4
  herd      100 11 4
  years      10 24 2
  age         0 27 6 1
  weight      0 33 6 1
  backfat     0 39 6 1
  adj_backfat 0 45 6 1
  dailygain   0 51 6 1
MODEL
 backfat   = years herd sire
 dailygain = years herd sire
```

```
c Residual covariance matrix
VE
 2.2  32.3
 32.3 3200
c...Additive genetic covariance matrix:
VG
 VG_FOR sire
 1.2/4 10.4/4
 10.4/4 2240/4

c...covariance matrix for herds:
 VG_FOR herd
 1.3 -3.36
 -3.36 1500
SOLVER /IOC IOD IOCD SMP
 ioc
 stop = .001
printout
 outfile 'scr.list'
```

The model specifies a two trait analysis for backfat and daily gain with the effects years, herd and sire. Sire and herd are random as there are two entries in the VG section (VG_FOR sire and VG_FOR herd). Year does not have a covariance structure specified and is, thus, considered fixed. Note that all of the input in the parameter file is in free format. The order of the sections is immaterial, so is the order of traits in the model. It has, however, to be noted that the covariance components have to be specified in the same order as the traits are listed in the model section! This is very important to notice as there is no way to check for sensible covariance matrices.

4.2 The Runtime Output

During execution PEST prints information on the list device, usually the CRT. The following is the output for the above parameter file.

```
 pest PEST.manual.p1
         ************************************************
         *                    P E S T                   *
         *     Multivariate Prediction and ESTimation   *
         *               15.0 MB version                *
         *                                              *
         *     E. Groeneveld, M. Kovac, and T. Wang     *
         *         Department of Animal Sciences        *
         *            University of Illinois            *
         *           Mon Mar 12 12:04:24 1990           *
         ************************************************
 Terms in part A (IOC       )
          1 YEARS
          2 HERD
          3 SIRE
     0.1% used out of  15360. kb. at  R_FILTER
 --> Start coding data.
   So far     500 data records have been read.
   So far    1000 data records have been read.
 In total    1127 data records have been read.
     0.1% used out of  15360. kb. at DataFilter
 --> Start Setting up and Solving Equations.
     0.8% used out of  15360. kb. at BLUP_IOD
MME done.
     IOC at round    10 max. change :     3.256196
     IOC at round    20 max. change :     2.260296
     IOC at round    30 max. change :     1.448869
```

```
     IOC at round    40 max. change :     0.857848
     IOC at round    50 max. change :     0.455053
     IOC at round    60 max. change :     0.192526
     IOC at round    70 max. change :     0.102660
     IOC at round    80 max. change :     0.119010
     IOC at round    90 max. change :     0.130172
     IOC at round   100 max. change :     0.135378
     IOC at round   110 max. change :     0.136861
     IOC at round   120 max. change :     0.128216
     IOC at round   130 max. change :     0.114377
     IOC at round   140 max. change :     0.098489
     IOC at round   150 max. change :     0.082552
     IOC at round   160 max. change :     0.067705
     IOC at round   170 max. change :     0.054520
     IOC at round   180 max. change :     0.043204
     IOC at round   190 max. change :     0.033744
     IOC at round   200 max. change :     0.026002
     IOC at round   210 max. change :     0.019779
     IOC at round   220 max. change :     0.014856
     IOC at round   230 max. change :     0.011017
     IOC at round   240 max. change :     0.008063
     IOC at round   250 max. change :     0.005819
     IOC at round   260 max. change :     0.004136
     IOC at round   270 max. change :     0.002890
     IOC at round   280 max. change :     0.001979
     IOC at round   290 max. change :     0.001323
rankxx=   12
 --> Start Processing Output.
     0.1% used out of  15360. kb. at HypothesisTesting
         ***************************************************
              The list output is on file: scr.list
         ***************************************************
THANKS FOR USING THE SOFTWARE, GOOD LUCK!
```

After PEST has been started it prints a banner that gives the size of the data buffer with which this particular version has been compiled at installation time. This particular version has a rather big buffer of 15MB. The output after the banner gives the distribution of the effects over the three classes of solvers. As a default all effects are placed in part A (this is storage of coefficients in memory) and solved by IOC. This is explicitly specified in the parameter file, which would not have been necessary.

PEST has three distinct phases (table 2): in the first data are coded, i.e. the level codes - which may be up to 16 characters long - are translated into numerical values. Where requested relationship data are coded first. As a result a binary output file is produced. If no output file name is specified the default name relationship.pest is used. Notice that these files can become very large for large problems.

Memory is managed dynamically as PEST proceeds through its three phases. The amount memory used is indicated by the printout :.01% used out of 16360 KB at r_filter which codes the relationship file, and the same statistic at DataFilter. Thereafter setting up the mixed model equations start, followed by another statistic reflecting current memory usage. MME indicates that the mixed model equations have been set up and that the solving process started. In this example solving is done by Gauss-Seidel iteration on sparse half stored coefficient. Every 10 rounds a statistic is printed giving the maximum absolute change in the solutions from one round to the next. The default stopping criterium is .001 which was reached after about 290 rounds. This is followed by the last phase of PEST the processing of output which includes the recoding of data. Finally

the termination of PEST is announced with a reminder where the list output is written to.

4.3 List Output

Every list output is divided into a general statistic that describes the job, including processing time, memory utilization, model information and basic statistics. This is followed by a printout of the solutions together with the identification of the levels. Each page starts with a page header which contains the version number of PEST together with the date and time the job was run. Furthermore, it also contains in the middle of the second line the string which was placed behind the comment section name in the pfile. In this case it is P1.

4.3.1 Comment Information

As already described above the comment section allows to store descriptive comments with the parameter file the first five lines of which will be printed.In this example only 4 lines were specified, thus, all of them could be printed. There is no limit to the number of comment lines in the comment section.

4.3.2 General Information

This block gives some global statistics regarding the job. It starts with the available buffer size of the current version, followed by the degree of utilization, in this case mere .9 %. The total number of equations is 290 with 2169 non zero elements. This number refers to the half stored matrix. The second number in the last line is either the number of non zero elements specified in the SYSTEM_SIZE section or an estimate by PEST which is based on the number of equations. This default cannot always be correct. If it is too small PEST will stop processing and issue a corresponding message. In this case the user has to set system_size to an appropriate value and start PEST again.

The next line gives the number of records in the data file, followed by the parameter, the list and data file. The last line in this block gives the MEMORY as a binary input.As a default PEST read data always from memory which increases performance in the case of iteration on data (solvers IOD and IOD_GS). However, the penalty is increased memory demand. If the key word DISK is specified in the data and pedigree section data will not be loaded into memory but rather read from disk. This will only be a disadvantage if either IOD or IOD_GS is used. In the case of this example not performance degredation is to be expected as data are only read once.

4.3.3 Run Time Information

The block Run Time Information gives statistics about CPU and memory utilization. It is important in tuning PEST for a certain application.

4.3.4 Data File Information

This section gives some basic statistics regarding the input data before any transformations are performed. It gives the minimum, maximum, average, standard deviation and

coefficient of variation for all continuous variables in the model, i.e. traits and covariables. It furthermore gives a count of missing values in the data set. The latter is obviously defined according to the specifications in the PFILE or the defaults in use.

4.3.5 Solver Information

This section describes the distribution of effects over the three classes of solvers, the stopping criteria used, the maximum number of iterations allowed and finally the actual number of iterations performed. This particular PFILE specifies all effects to be solved by IOC. This is why the columns referring to IOD and IOD_GS are empty. As can be seen 297 round were required with an actual value for the stopping criterium being .0009.

4.3.6 Model Information

This section list which traits are defined for which effect. This pfile has the same incidence matrix for all traits. This is why we have an x for each trait/effect combination. The listing of the covariance matrices follows. Always check this part. Wrong covariances are hard to detect from the solutions part. With missing values all the different residual covariance matrices are listed.

4.3.7 Solutions to the Mixed Model equations

The solutions to the mixed model equations are printed according to the format specifications in the PRINTOUT section and/or the built in defaults. It has to be noted that the numerical value of the solutions may change with the ordering of the effects in the model and also with the change in solvers. While this may be disturbing initially it has to be noticed that only estimable functions are invariant and should always give the same numerical values.These are all random effects.Thus, the solutions for herd and sire should always be the same, irrespective of the solvers used or the placement of the effects in the model.

```
..................... PEST UIUC V1.5    .....................
 Mon Mar 12 13:58:34 1990  P1                          page   1

.......................C o m m e n t s.....................
. SIRE MODEL                                                 .
. 0082 0001 0001 '12-DEC-82' 176.0 126.0  10.0   8.9 715.0     .
. 0075 0211 0002 '16-DEC-85' 213.0 132.0  12.0  10.5 619.0     .
. 123456789.123456789.123456789.123456789.123456789.123456789 .
.............................................................


     G e n e r a l  I n f o r m a t i o n:
     ____________________________________


     Useable memory        :  15.000 MB
     Memory used           :   0.85 %
     # of equations        :       290
     # of nonzero elements :      2159        4219
     # of data records     :      1127
```

```
Parameter   file      : PEST.manual.p1
Results     file      : scr.list
Input data  file(raw) : /home/eg/verify/coursedata
Input data  file(bin) : MEMORY
```

R u n T i m e I n f o r m a t i o n:

CPU time spend in	hour:min:sec	memory(kb)
data preparation	00:00:04	14
setting up and solving	00:00:12	103
Total wall clock time / memory	00:00:18	130

D a t a F i l e I n f o r m a t i o n:

	BACKFAT	DAILYGAIN
Total # of records	1127	
missing value	.	.
# of missing	0	0
Maximum	16.0000	717.0000
Minimum	6.0000	347.0000
Mean	9.7915	537.6389
Stand. deviation	1.5433	50.8990
Coef.of Variation %	15.7620	9.4671

S O L V E R I n f o r m a t i o n:

```
 ..................... PEST UIUC V1.........................
Mon Mar 12 13:58:52 1990  P1                          page   2

                Part A             Part B            Part C

       SOLVERS    IOC        Gauss-Seidel/Jacobi   Gauss-Seidel

.............................................................
       TERMS      YEARS
                  HERD
                  SIRE
.............................................................
       Stop        0.00100
       Relax       1.00000
       Max_I          1000
       ..........
       act.  stop. 0.00098
       act.# iter.     296
```

```
  M o d e l  I n f o r m a t i o n:
  ---------------------------------

#  effect         type   levels   traits......
-------------------------------------------------------------
                                    BACKFAT     DAILYGAIN
1  YEARS            F        6        x           x
2  HERD             R       28        x           x
3  SIRE             R      111        x           x
-------------------------------------------------------------

  Covariances (residual)
  ----------------------

       1......
                 2.200      32.300
                32.300    3200.000

  Covariances (random effects)
  ----------------------------

       1......SIRE
                 0.300        2.600
                 2.600      560.000
       2......HERD
                 1.300       -3.360
                -3.360     1500.000
```

4.4 Missing values, different incidence matrix, relationship

The following is a more complicated model: a genetic evaluation of boars and gilts for daily gain, backfat (measured on boars) and number of piglets born for gilts. While this is not a particularly useful model it does serve the purpose of demonstrated how a more complicated model can be implemented. Table 3 gives the parameter file for this model.

Note that in this case no format is specified under the data section, only the maximum number of levels are given. Also, free format is assumed for the pedigree file.

The transformation section specifies .0 as missing values, no upper and lower bounds are set. The keyword DISK under the data section specifies that data are not to be loaded into memory but rather read repeatedly from disk.

The model has a different incidence matrix for each trait. Liveweight nested within locations is fitted to backfat, while age at first litter is a covariable for the number of piglets born.

TABLE 3 parameter file 2:

```
RELATIONSHIP
      rel_for animal
      disk
      infile = 'hb_el.ped'
      undefined = '0000000'
      input
           animal
           m_p
           f_p
COMMENT P20
   These are herdbook data boars. Using the disk option to reduce
   memory requirements.
transformation
  treated_as_missing
   backfat     none .0 none
   daily_gain  none .0 none
   #_born      none .0 none
system_size
   non_zero = 820

DATA
      disk
      INFILE  =  'hb_el.dat'
      INPUT  [  VAR_NAME    MAXLEVEL START_COLUMN VAR_LENGTH
DECIMAL ]
                 animal          1290
                 month_of_test    100
                 live_weight        0
                 month_of_litter  100
                 age_at_litter      0
                 backfat            0
                 daily_gain         0
                 #_born             0
                 loc               10
MODEL

       backfat     =  live_weight(loc) month_of_test  loc animal
       daily_gain  =                   month_of_test  loc animal
       #_born      =  age_at_litter    month_of_litter  animal

VE
            2.2    32.3    .01
           32.3  3200.0  12.00
             .01   12.0   5.60

VG
    VG_FOR animal
           1.2 10.4 .01
          10.4 2240 8.8
          .01 8.8   4.3
SOLVER
       ioc           [stop = .1, relax=1.0 max_iter=1000
       iod_gs animal [stop = .1  max_iter=200
printout
      outfile 'vlistp20'
```

The VG sections specify this to be an animal model. Note that the order of the traits in the model section has to correspond to the covariances in the VE and VG section.

Two types of solvers will be engaged in solving this problem. The large animal effect is placed in IOD_GS which requires least memory.The rest remains by default in part A for which IOC is specified. The stopping values is set to .01 which will also be used for IOD_GS, as nothing to the contrary is specified. The maximum number of round for IOC is set to 1000. In real life data sets the litter effect would have to be accommodated. As it is also rather large, one would place it into IOD.

For the other two the default of 200 will be in effect. The keyword **nsig_digit** under the printout section request an extended number of digits on output.

Table 4 gives the output of PEST during execution.

TABLE 4 Output during execution:

```
orion{eg}64: pest p20
         ************************************************
         *                    P E S T                   *
         *     Multivariate Prediction and ESTimation     *
         *               10.0 MB version                  *
         *                                                *
         *     E. Groeneveld, M. Kovac, and T. Wang       *
         *         Department of Animal Sciences          *
         *            University of Illinois              *
         *           Wed Sep  5 09:08:22 1990             *
         ************************************************
 Terms in part A (IOC       )
          1 LIVE_WEIGHT       1 LOC
          2 MONTH_OF_TEST     2
          3 LOC               3
          4 AGE_AT_LITTER     4
          5 MONTH_OF_LITTER   5
 Terms in part C (IOD_GS    )
          1 ANIMAL            1
 --> Start coding relationships.
 number of non_base ANIMAL     &             =     983
 number of base parents/groups               =     303
 number of total ANIMAL     &                =    1286
     0.4% used out of  10240. kb. at  R_FILTER
 --> Start coding data.
 In total     796 data records have been read.
     0.4% used out of  10240. kb. at Data Filter
     0.5% used out of  10240. kb. at Relationship
     0.8% used out of  10240. kb. at Relationship Matrix
 --> Start Setting up and Solving Equations.
     0.7% used out of  10240. kb. at BLUP_IOD
     Non zero coefficients stored:       769, filled:  72.1%
 --> Setting up Mixed Model Equations done.

 IOC...:   10   0.05   0.00   0.02
 IOC...:   20   0.05   0.00   0.02
 IOC...:   30   0.04   0.00   0.02
 IOC...:   40   0.04   0.00   0.02
 IOC...:   50   0.04   0.00   0.02
 IOC...:   60   0.04   0.00   0.02
 .....
 .....
```

```
 IOC...:  400   0.01   0.00   0.01
 IOC...:  410   0.01   0.00   0.01
 IOD_GS:    1  0.8 42.2  0.6

 IOC...:   10  0.01  0.00  0.01
 IOC...:   20  0.01  0.00  0.01
 IOC...:   30  0.01  0.00  0.01
 IOC...:   40  0.01  0.00  0.01
 IOC...:   50  0.01  0.00  0.01
 IOD_GS:    2  0.4 20.2 -0.1

 IOC...:   10  0.01  0.00  0.01
 IOC...:   20  0.01  0.00  0.01
 IOC...:   30  0.01  0.00  0.01
 IOD_GS:    3  0.3 16.6 -0.1

 IOC...:   10  0.01  0.00  0.01
 IOC...:   20  0.01  0.00  0.01
 IOC...:   30  0.01  0.00  0.01
 IOD_GS:    4  0.3 12.2 -0.1

 IOC...:   10  0.01  0.00  0.01
 IOC...:   20  0.01  0.00  0.01
 IOC...:   30  0.01  0.00  0.01
 IOD_GS:    5  0.2  8.6  0.0

 IOC...:   10  0.01  0.00  0.01
 IOC...:   20  0.01  0.00  0.01
 IOD_GS:    6  0.2  6.0  0.0

 IOC...:   10  0.01  0.00  0.01
 IOC...:   20  0.01  0.00  0.01
 IOD_GS:    7  0.1  4.4  0.0

 IOC...:   10  0.01  0.00  0.01
 IOD_GS:    8  0.1  3.5  0.0

 IOC...:   10  0.01  0.00  0.01
 IOD_GS:    9  0.1  2.8  0.0

 IOC...:   10  0.01  0.00  0.01
 IOD_GS:   10  0.1  2.3  0.0
 IOD_GS:   11  0.0  1.8  0.0
 IOD_GS:   12  0.0  1.5  0.0
 IOD_GS:   13  0.0  1.2  0.0
 IOD_GS:   14  0.0  1.0  0.0
 ...
 ...
 IOD_GS:   29  0.0  0.2  0.0
 IOD_GS:   30  0.0  0.1  0.0
 IOD_GS:   31  0.0  0.1  0.0
 IOD_GS:   32  0.0  0.1  0.0
 IOD_GS:   33  0.0  0.1  0.0
     0.7% used out of  10240. kb. at After BLUP_IOD
     0.7% used out of  10240. kb. at Before printing.
 --> Start Processing Output.
      0.7% used out of  10240. kb. at After printing.
         ***************************************************
          The list output is on file: vlistp20
         ***************************************************
THANKS FOR USING THE SOFTWARE, GOOD LUCK!
STOP: PEST
```

The output immediately following the banner give the distribution of effects over the solvers. Liveweight, location, age at litter, and month of test are solved simultaneously

in the first part by iteration on coefficients.These equations are, thus, set up in a half stored sparse format directly in memory.

The animal effect is solved in part C by IOD_GS, which is also Gauss-Seidel iteration on data. As can be seen a double block iteration is performed: first IOC iteratively solves the equations of effects liveweight, location, age at litter, and month of test. Then IOD_GS solves the animal equations. Then IOC starts the whole process over again. While initially it took more than 400 rounds to convergence in IOC, this number goes down from each overall round to the next. The reason is, of course, that each following IOC block starts already from previous solutions which come successively better, so that in later IOD_GS rounds IOC requires less than 10 rounds and, thus, does not produce a print out.

In IOD_GS, the equations for the three traits are solved simultaneously. Contrary, IOC does not recognize the trait substructure of the mixed model equations but rather solves one equation a the time. Had the animal effect (together with the relationship) been placed in IOD, would the solving strategy have been Jacobi. In this case it is important to perform under relaxation, thus, setting relax_iod to values around .7. Values of 1. and above may not lead to convergence!

The pedigree file is coded as the first mayor computational effort. Depending on its size this may take some time. So far pedigree files of sizes up to 850,000 have been coded which may take a couple of hours. At this point a binary file is created which contains the pedigree information in a format that results in minimum computations during later iterations. PEST reports 164 base parents,i.e.parents with no known ancestors. The data file need not contain these bases animals with a record of their own. If they are present PEST will skip them and generate them again.

In the next step the data are coded. Each animal identification is checked against those in the pedigree file. No animal in the data file is allowed that does not have an entry in the pedigree file. Should this occur PEST will terminate.

The next step is setting up and solving the mixed model equations. In the case with all effects in part A, i.e. IOC, SMP or DENSE, the process of setting up and solving is separate. If, however, IOD and/or IOC_GS is employed the process in interlaced. depending on the number of the equations that are being placed in part A this process may take some time. A hashing technique is used to store the non zero coefficients. If the buffer become very tight the process may slow down considerably. If this occurs the system size section should be used and a higher value supplied. PEST lists each 10,000 elements stored on the screen, together with the degree of filling and the hit rate. The latter is a statistic that gives the number of accesses for each new element stored. Values around 50 are not uncommon.

As can be seen from the list output for PFILE.2 different incidence matrices are used for all of the traits. Accordingly, the solutions part contains only values (solutions) for those traits that are defined for the respective effect.

TABLE 5 Output for PFILE.2

```
..... PEST UIUC V1.3     .....
 Tue Feb  6 18:18:44 1990  HERDBOOK DATA                          page   1

    .....C o m m e n t s.....
    . THESE ARE HERDBOOK DATA BOARS                                .
    ...............................................................

     G e n e r a l  I n f o r m a t i o n:
     ____________________________________

     Useable memory        :  14.648 MB
     Memory used           :   0.85 %
     # of equations        :      4116
     # of nonzero elements :       769         820
     # of data records     :       796
     Parameter   file      : p.8
     Results     file      : list.p8
     Input data  file(raw) : hb_el_zl.dat
     Input rel.  file(raw) : /home/eg/verify/hb_el_zl.pedig
     Input data  file(bin) : datafile.pest
     Input rel.  file(bin) : relation.pest

     ____________________________________

     R u n  T i m e  I n f o r m a t i o n:
     _____________________________________

     ____________________________________________________________
     CPU time spend in                   hour:min:sec  memory(kb)
       data preparation                :  00:00:18          39
       setting up and solving          :
       solving the system of equations :  00:00:55          71
     Total wall clock time / memory    :  00:01:22         127
     ____________________________________________________________

     D a t a  F i l e  I n f o r m a t i o n:
     ________________________________________

                      LIVE_WEIGH AGE_AT_LIT    BACKFAT DAILY_GAIN     #_BORN

_____________________________________________________________________________
     Total # of records        796
     missing value                                .0         .0         .0
     # of missing                                 148        148        648
     Maximum                170.00     483.00     16.00     748.00     22.00
     Minimum                 96.00     366.00      7.00     464.00      7.00
     Mean                   129.95     390.46     10.74     574.88      9.36
     Stand. deviation        11.54      18.53      1.51      46.86      1.66
     Coef.of Variation %      8.88       4.75     14.09       8.15     17.72

_____________________________________________________________________________


 ..... PEST UIUC V1.3     .....
 Tue Feb  6 18:20:20 1990  HERDBOOK DATA                          page   2
```

```
R E L A T I O N S H I P  I n f o r m a t i o n:
_____________________________________________

   number of base animals     :     264
   number of non-base animals :    1023
   total number of animals    :    1287
_____________________________________________

S O L V E R  I n f o r m a t i o n:
__________________________________

             Part A              Part B               Part C
_____________________________________________________________________
   SOLVERS    IOC           Gauss-Seidel/Jacobi        Gauss-Seidel
   ..........................................................
   TERMS      LIVE_WEIGHT*LOC                         ANIMAL
              MONTH_OF_TEST
              LOC
              AGE_AT_LITTER
              MONTH_OF_LITTER
   ..........................................................
   Stop        0.01000                                 0.01000
   Relax       1.00000                                 1.00000
   Max_I          1000                                     200
   ..........
   act.  stop. 0.00819                                 0.00997
   act.# iter.    10.0 ( 660)                               66
_____________________________________________________________________

M o d e l  I n f o r m a t i o n:
_________________________________

#  effect         type   levels   traits......
_____________________________________________________
                                  BACKFAT    DAILY_GAIN #_BORN
1  LIVE_WE.LOC      C        9        x          -         -
2  MONTH_OF_TEST    F       15        x          x         -
3  LOC              F        9        x          x         -
4  AGE_AT_LITTER    C        1        -          -         x
5  MONTH_OF_LITTER  F       51        -          -         x
6  ANIMAL           A     1287        x          x         x
_____________________________________________________


..................... PEST UIUC V1.3     .....................
Tue Feb  6 18:20:20 1990  HERDBOOK DATA                 page   3

   Covariances (residual)
   ______________________

        1......
                  2.200      32.300       0.000   ....   missing data.
                 32.300    3200.000       0.000   ....   missing data.
                  0.000       0.000       0.000   ....   missing data.
        2......
                  0.000       0.000       0.000   ....   missing data.
                  0.000       0.000       0.000   ....   missing data.
                  0.000       0.000       5.600   ....   missing data.
```

```
Covariances (random effects)
____________________________

     1......ANIMAL
                1.200       10.400        0.010
               10.400     2240.000        8.800
                0.010        8.800        4.300

------------------------------------------------------------


........................... PEST UIUC V1.3     ............................
Tue Feb  6 18:20:20 1990  HERDBOOK DATA                          page   4


LIVE_WEIGHT.*.LOC   BACKFAT DAILY_GAIN   #_BORN

LIVE_WEI*01           0.080
LIVE_WEI*02           0.062
LIVE_WEI*03           0.087
LIVE_WEI*06           0.025
LIVE_WEI*07           0.075
LIVE_WEI*08           0.080
LIVE_WEI*09           0.084
LIVE_WEI*10           0.000
LIVE_WEI*19          -0.017


MONTH_OF_TEST     BACKFAT DAILY_GAIN   #_BORN

0000                0.000       0.00
8705               -3.922     496.13
8706               -3.395     508.93
8707               -3.937     498.72
8708               -4.427     487.11
8709               -4.257     496.33
8710               -4.183     506.69
8711               -4.051     506.10
8712               -5.020     526.57
8801               -3.811     564.39
8802               -3.909     487.64
8803               -3.900     505.46
8804               -4.495     504.41
8805               -4.073     502.89
8806               -3.953     509.20

LOC               BACKFAT DAILY_GAIN   #_BORN

01                  4.217      73.75
02                  6.717      70.02
03                  3.587      76.06
06                 11.896      97.72
07                  5.191      79.31
08                  4.396      78.27
09                  3.807      74.63
10                  0.000       0.00
19                 17.017       9.07

AGE_AT_LITTER     BACKFAT DAILY_GAIN   #_BORN

AGE_AT_LITTER                            0.009

MONTH_OF_LITTER   BACKFAT DAILY_GAIN   #_BORN
```

```
    0000                                  0.000
    7812                                  8.945
    8011                                  6.386
    8101                                  5.259
    8102                                  7.680
    8104                                  3.745
    8108                                  5.507
    8109                                  7.102


..... PEST UIUC V1.3     .....
Tue Feb  6 18:20:20 1990  HERDBOOK DATA                    page   5

    MONTH_OF_LITTER   BACKFAT DAILY_GAIN   #_BORN
    8202                                  9.129
    8204                                  6.475
    8206                                  5.364
    8207                                  4.717
    8208                                  6.281
    8211                                  5.710
    8301                                  4.976
    8302                                  9.159
    8304                                  5.098
    8308                                  8.901
    8310                                  5.910
    8311                                  6.929
    8312                                  5.693
    8401                                  7.569
    8402                                  4.100
    8404                                  7.791
  ........
  ........ lines skipped

    ANIMAL            BACKFAT DAILY_GAIN   #_BORN

    0000019             0.000       0.00    0.000
    0000021             0.000       0.00    0.000
    0000026             0.000       0.00    0.000
    0000112             0.000       0.00    0.000
    0000138             1.080     -19.27   -0.256
    0000149             0.745      23.96    0.077
    0000238            -0.113       0.44   -0.008


..... PEST UIUC V1.3     .....
Tue Feb  6 18:20:21 1990  HERDBOOK DATA                    page   6

    ANIMAL            BACKFAT DAILY_GAIN   #_BORN
    0000255            -0.194      -9.82   -0.035
    0000267             0.000       0.00    0.000
    0000277             0.245     -11.94   -0.138
    0000299             0.177       8.26   -0.642
    0000333            -0.372      32.54   -0.102
    0000377             0.105     -13.87   -0.059
    0000400            -0.202     -19.75   -0.075
    0000652             0.000       0.00    0.000
    0000679             0.000       0.00    0.000
    0000700             1.314       4.74   -0.017
    0000701             0.715      74.62    0.283
......
...... lines skipped

..... PEST UIUC V1.3     .....
Tue Feb  6 18:20:24 1990  HERDBOOK DATA                    page  30

    ANIMAL            BACKFAT DAILY_GAIN   #_BORN
```

```
1002416*            0.042      -0.07   -0.291
1002497*            0.000       0.00    0.000
1002502*           -0.201      -2.50    0.197
1002669*            0.037      19.35   -0.124
1002965*            0.000       0.00    0.000
1003141*           -0.123      -7.61   -0.028
1003576*            0.000       0.00    0.000
1006264*           -0.100      -5.32    0.300
1007015*            0.032      -3.44   -0.015
1007173*           -0.045      -0.53   -0.001
1020824*           -0.141       2.35    0.005
1020825*            0.071      -7.11   -0.472
1080736*            0.007       3.22    0.039
1080800*           -0.082      11.80    0.050
1080840*           -0.022       0.56    0.003
1080858*            0.036      12.32    0.049

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

CHAPTER 5 Tuning PEST

PEST can be tuned to adapt to a large variety of constraints. Tuning is performed by

- placing effects in the three classes of solvers (A, B, C)
- by choosing the appropriate solvers for class A
- choosing the optimal stopping criterion for the three classes
- setting the ceiling of maximum number of iterations
- using the appropriate relaxation parameters for A, B, C
- placing data on disk or in memory
- placing relationship info on disk or in memory
- choosing Cholesky decomposition
- choosing canonical transformation
- reloading old solutions
- compiling PEST with the appropriate memory size

5.1 Solver classes

The three solver classes A, B, C are the mayor means of tuning PEST for speeding up the solving processes and influencing the memory requirements. A wise choice of solv-

ers will drastically reduce computation time and should, therefore, be clearly understood.

5.1.1 Solver class A

SMP

IOC

DENSE

All effects are placed by default into class A. The mixed model equations are set up in memory prior to the solving process. Thus, coded data are read only once which is prior to solving. Depending on the choice among the three solvers SMP, DENSE and IOC memory requirements may vary drastically. SMP and IOC, both, operate on sparse (i.e. only non zero) half stored coefficient matrices. The right hand sides have to be stored in full for all three solvers. They do however vary in their work space requirements: IOC requires above the IA, JA, A vectors a work space for the modified RHS whereas SMP requires much larger work buffers. DENSE requires half storage of the complete coefficient matrix including zeros. For smaller systems up to 500 equations DENSE may be more memory efficient whereas SMP will be better above this value. The user should consult the run time information section in the print output to decide which solver is best suited to the problem. All three solvers solve the MME without consideration for its particular structure with DENSE and SMP giving converged solutions whereas IOC as an iterative procedure stops after the specified convergence criterion is satisfied. For practical purposes like ranking animals converged solutions may not be needed thus giving solutions much faster than would be possible for either DENSE and SMP.

DENSE and SMP may, however, be superior to IOC when part A is used in conjunction with the other two solvers. In this case part A has to be solved for each iteration in part B and C. Ordering and factorization in SMP and decomposition in DENSE is only done once. Thereafter this step is skipped and solving is very much faster. This is exactly the above described case.

No inherent restrictions apply to this solver class: all effects can be placed here, all models that are allowed in PEST can by specified. The only limitations are those of memory requirements and CPU time. Note that in an animal model storage for the non zero coefficient will quickly exceed that for the data.

The order of effects in the mixed model equations is determined by the order of the effects in the MODEL section. As a general rule effects with the smallest number of levels should be placed first for faster hashing and convergence. This means that covariables and intercepts should always be placed first followed by the other effects.

The section SYSTEM_SIZE allows specification of buffers for PEST to store the non zero elements. If not specified a built in algorithm tries to estimate the number of non zero elements on the basis of the number of mixed model equations. This will not always work. The estimate may either be too large, thus wasting memory or too small in which case one of the following may happen:

- PEST will stop with the message that the system size is too small and that the non_-zero keyword should be used to increase the buffer.
- if SMP is used the ordering driver may abort as there is not enough work space available.
- The process of hashing may be severely slowed down due to crowding in the hash buffer. This is indicated by a printout on the screen which reports the number of ele-

ments stored so far, the degree of filling and the hit rate. The latter is the ratio of attempts to locate a free place to store a coefficient to the number of coefficients actually stored. If this value goes up to 100 or over it may be advisable to increase the system size.

```
Non zero coeffients stored: 5000, filled: 8.3% hit rate: 5.7
Non zero coeffients stored:10000, filled:16.5% hit rate:10.2
Non zero coeffients stored:15000, filled:24.8% hit rate:11.8
Non zero coeffients stored:20000, filled:33.1% hit rate:12.9
```

If all effects are placed in part A data can be read from disk without much loss of processing time for reduced memory requirements as data are read only once. In this case specify DISK in the data and relationship section.

5.1.2 Solver class B

IOD

This solver stores the diagonal blocks of the mixed model equation system and solves them by Gauss-Seidel on iteration. The solver keyword is IOD. For each effect in IOD data has to be read once. This makes storage of data important. The fastest option is the default in PEST which loads data into memory and accesses them directly during iteration. Depending on the size of the data file this may take up a substantial amount of memory. Should this not be available can data be read from disk. As described above the keyword DISK in the data and relationship section have to be set.

If relationship is used in this section The solving strategy becomes Jacobi for this effect. It has to be noted that in this case the relaxation parameter most likely has to be set to a value of below 1.0. If this is not done the system of equations may not converge. Setting RELAX_IOD to .8 has proven a reasonable value but its optimum value may depend on the data set and the model used.

In the multiple trait case, the trait by trait equations are solved simultaneously. This leads generally to a faster convergence that IOC which does not take advantage of the structure of the MME.

Some restrictions apply to this solver class: firstly, covariables and intercept are always placed in part A, even if specified under IOD. This makes sense because otherwise data will have to be read once for those one or two equations only. Secondly, sire dam models with relationship are allowed only in part A. In the case without relationship they can be placed in IOD.

5.1.3 Solver class C

IOD_GS

This stores one block of diagonal elements of the mixed model equations only. This makes it the most memory efficient procedure. Solving is by Gauss-Seidel with simultaneous solution of the trait by trait equations in the multiple trait case. The solver is invoked by IOD_GS followed by the effect name.

IOD_GS has the most restrictions: it is designed to deal with one large effect, like the animal effect. This may include the relationship information. As long as data are read from memory - the default in PEST - data do not have to be sorted. If, however, the disk

keyword is chosen input data have to be sorted according to the effect placed in IOD_GS.

The relaxation parameter RELAX_IOD_GS can be set separate for this solver, the same applying to the maximum number of iterations (MAX_ITER_IOD_GS) and the stopping criterium (STOP_IOD_GS). Note that if a different ceiling is set for part B and C always the smaller of both values will be effective as for each iteration in part B there is one iteration in part C.

Again, the keyword DISK can be specified for both data and relationship records. If there is enough memory to hold either data or pedigree information one can be specified as to be read from disk while the other will be held in memory.

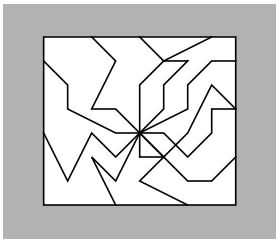
5.1.4 Distribution of effects over solvers

Always place effects with a small number of levels in part A. Typical examples are breeds, covariables, seasons, herds. The total number of equations can be well in excess of 1000. Large effects should be placed into part B. Litter is a typical example for this class. Remember that for every effect you add to part B data have to be read once more. Part C would finally contain the largest effect, typically the animal effect.

If sire models a run, it would be wise to place them into part A and specify data input to be read from disk, as the data file may be very large whereas the system of equations is not.

It should, however, be noted that even if all equations would fit into part A it might be more efficient to put one or some effects in part B or C. The speedup is achieved by the simultaneous solution for all equations in part A which usually leads to faster convergence which - sometimes - can be substantial. For a full discussion of this issue see: E. Groeneveld and M. Kovac: Performance Characteristics of Different Solving Strategies in Multivariate Mixed Models. Livestock Production Science, 30 (1992) 319-331

CHAPTER 6 Troubleshooting



Most of the errors detected by PEST will be self explanatory. The comments and explanations are stored in the file SYNTAX.ERR or a slightly modified name depending on the operating system.

6.1 Syntax and runtime errors

SYNTAX.ERR must not be modified! If it is changed PEST may not stop when required and, thus, produce unpredictable results. The following gives a list of this file:

```
--------- Syntax and runtime errors for PEST ----------

00.      This explains how errors and warnings are handled.
         NN.NN  message code
         X      error/warning status
         Text   Message, length is 65 characters per line, at least one line

         Messages are coded with a section number and sequential number:
         Section:
              00.  Introduction text
              01.  COMMENT section
              02.  SYSTEM_SIZE section
```

03. DATA section
04. RELATIONSHIP section
05. MODEL section
06. SOLVER section
07. PRINTOUT section
08. HYPOTHESIS section
09. TRANSFORMATION section
10. COVARIANCE section
11. STARTING_VALUES
30. Syntax for values, arithmetic operations
50. Runtime messages
90. System errors
99. Dummy, internal use, comments
M1. OPTIMIZATION
MM. In development

Messages are divided into three groups:
S = very serious and system errors. Stops the program instantly.
E = error. Stop it after 5 such messages have been accumulated, at the end of parameter file
W = warning. There is slight disagreement. The program will proceed. The user must check output for results may be wrong.

---

01. COMMENT section
01.99 E Undefined keyword in COMMENT section!

---

02. SYSTEM_SIZE section
02.10 S Limit of nonzero elements for sparse storage is exceeded. Remedy: increase NON_ZERO in parameter file.
02.99 E Undefined keyword in SYSTEM_SIZE section!

---

03. DATA section
03.01 S Section is missing.
03.10 E Length of an classified variable under DATA section exceeded limit specified with keyword MAX_CHAR. Check format or use default value for MAX_CHAR (16)!
03.11 S Do not mix free and fixed format.
03.99 E Undefined keyword in DATA section!

---

04. RELATIONSHIP section
04.05 W More than two effects in relationship.
04.10 E Maximum numbers of levels for effect in relationship section must be greater than 1! Check DATA INPUT section.
04.11 S Do not mix free and fixed format.
04.20 S Variable names allowed in relationship section are: “animal”, “m_p” for sire, “m_gp” for grandsire, “f_p” for dam, “f_gp” for granddam and “birthdate”. Check your list of variables! The error may also occur if you mistype the following keyword or section. Pay attention to word:
04.21 E Check the variable list in the relationship section. Input should contain columns for “animal” and two ancestors from the list “m_p”, “m_gp”, “f_p”, “f_gp”. “birthdate” must be specified only if you want inbreeding to be included. If you want to skip a column on input, you should use a format.

04.22 E The animal must always be the first variable under the INPUT keyword. Move the following line up and use correct format:
04.25 W Keyword INBREEDING will be ignored because BIRTHDATE is not contained in pedigree file.
04.99 E Undefined keyword in RELATIONSHIP section!
---
05. MODEL section
05.01 S MODEL section is missing: specify the model in the parameter file!
05.02 S MODEL section exists twice in the parameter file. Delete one.
05.05 E Trait name must appear on the LeftHandSide of the model equation!
05.06 E Put intercept as first effect in the model.
05.07 S Trait name must appear in the first line of the model section!
05.08 S Trait name must appear only once in the model section! Repeated trait is:
05.09 E Term name must appear only once in the model equation! Repeated term is:
05.10 E SIRE and DAM must be adjacent in MODEL.
05.15 E Variable used in the MODEL is not specified in DATA section. Check name:
05.16 E Parentheses are not symmetric in the model section. Check nested factors. Check also other sections where options are specified. Start options with '{', '[', or '<' or enclose them in '(' and ')'.
05.80 S Reserved word is used in the model definition. Word:
05.99 E Undefined keyword in MODEL section!
---
06. SOLVER section
06.10 E Only one effect allowed in solver IOD_GS.
06.11 E No interaction allowed in solver IOD_GS.
06.15 E Covariate or classified variable name must appear in DATA INPUT section and in the MODEL section. Check name:
06.99 E Undefined keyword in SOLVER section!
---
07. PRINTOUT section
07.15 E Trait name must appear in MODEL section. Only traits with a specified model are considered in PEST!
07.99 E Undefined keyword in PRINTOUT section!
---
08. HYPOTHESIS section
08.10 W Hypothesis testing is not supported for the solver combination below. Use SMP solver only.
08.99 E Undefined keyword in HYPOTHESIS section!
---
09. TRANSFORMATION section
09.10 W Invalid or missing parameter under keyword SCALING!
09.15 E Trait or covariate name must appear in DATA INPUT section and in the MODEL section. Check name:
09.20 E Non numeric number under keyword TREATED_AS_MISSING.
09.21 W Missing value is not specified. Default (dot) will be used.
09.99 E Undefined keyword in TRANSFORMATION section!
---
10. VE or VG (COVARIANCE) section
10.10 E The order of matrices required in MODEL section and provided under COVARIANCE (VE or VG) section does not agree. Line:

10.20 E Keyword FOR was expected! Check number of rows for covariance matrices. See also Chapter 3.10 in User's Manual.
10.21 E Value for keyword IN_FORM is not supported!
10.22 E Two covariance matrices specified for the same effect!
10.23 E Non numeric character found in (co)variances.
10.24 S Covariance matrix is not symmetric positive definite.
Level for residual covariance matrix or random effect number:
10.99 E Undefined keyword! Keyword:

---

11. STARTING_VALUES section
11.99 E Undefined keyword in STARTING_VALUES section! Keyword:

---

30. GENERAL RULES APPLIED TO PARAMETER FILE
30.01 W Meaningless arithmetic operations. The second value required for the arithmetic operation is 0. The required operation is skipped!
30.02 E Unexpected character in parameter file near string followed. The value expected should be an integer, real or double precision number. Delimiters allowed are blank, +, -, /, *, comma, and all types of brackets. This message may occur also whenever the size of a matrix required is bigger than the matrix provided in a parameter file.
30.10 W Semipositive matrix containes negative values on diagonal.
30.90 E Section is empty. If section is not required, delete it. Section:
30.99 E Illegal section name in parameter file. Section:

---

50. RUNTIME - CODING DATA AND COMPUTATION
50.01 S Data file is empty. Misspelled name in parameter file?
50.02 S Relationship file is empty. Misspelled name in parameter file?
50.03 S Relationship file is not coded. Delete option RECODED!
50.10 S Data file has to be sorted by the effect put in IOD_GS solver!
50.11 S Your special data structure and solver combination is not handled by PEST: reading data from disk in IOD_GS with animals that have records but no pedigree and the relationship included will under certain circumstances lead to a loss of records.
This situation has occured now. You have the following options as workarounds:
1. do not use the DISK option in the DATA section.
2. use IOD or IOC/SMP instead of IOD_GS.
3. if because of memory constraints you must use the DISK option together with IOD_GS do the following:
- specify OUTFILE = 'FILE.coded' (TEXT in both the RELATIONSHIP and DATA section and run PEST to produce these files (one iteration is sufficient).
- format the files appropriately modifiy the PFILE and sort the data file by the animal number.
- specify the sorted data file (and the coded relationship file) as input in the RELATIONSHIP and DATA section:
INFILE='FILE.coded' (RECODED
This will prevent coding and the job will run through.
Remember that the codes in the relationship and data file have to be in the same format.
50.20 E Each animal with a record must be contained in a relationship file.
Check animal:

50.30 S There should be at least one main effect in the model.
50.50 S Maximum number of levels provided in parameter file was exceeded. The following number was too small:
50.60 E A level value from a previous run was not found in the new data file: all levels from the previous run have to be included in the current.
50.70 S Diagonal element of D is 0. Hints: check birthdate.
50.80 S Matrix is not semipositive definite!
50.81 W Rank of the coefficient matrix has unexpectedly changed!
50.99 S Undefined keyword or operator in TRANSFORMATION section! Check the word:
50.LOAD E A level from an old file is not found in the new system. All levels from the previous run have to be included in the current.
50.LOAD1 S New model and old solutions file are incompatible. The model must not change when reloading old solutions.
50.LOAD2 S Length of the identification in a new and old system is different.
50.RANKXX W Number of parameters is greater than number of observations. Thus, there is no error mean square and the rank does not make sense - it is ignored.
50.TWICE E Records in relationship file are repeated. Only one pedigree record per animal is allowed. Clean your relationship file.
50.CONV W System does not converge!
50.ESTIM W Column K' is not estimable!
50.KFULL W K is not full column rank!
50.ADDRES W Address for hypothesis test is outside the system of mixed model equations. Address has been set to 1. Check hypothesis section in parameter file and make appropriate changes. The test is meaningless!!!

---

90. SYSTEM MESSAGES
90.01 S File "SYNTAX.ERROR" is missing. The program cannot proceed without it!
90.EBMME S Illegal addresses in sparse matrix. Check format in data and relationship file!
90.F S Variable was not mapped. Sorry, call support!
90.GAMMQ S Something wrong in gamma function. Sorry, call support!
90.HELP S Sorry, call support!
90.MAP S PEST has not enough memory. Place some effects in IOD or IOD_GS, use DISK option in DATA and RELATIONSHIP file, use option MAX_LENGTH, check NON_ZERO value, or recompile PEST with more memory.
90.OPEN S File can not be opened. It does not exist or its name in the parameter file is misspelled. Consult your FORTRAN manual for "IOSTAT" error messages. IOSTAT value:
90.CLOSE S File can not be closed. Consult your FORTRAN manual for "IOSTAT" error messages. IOSTAT value:
90.READ S I/O problem reading a file. Consult your FORTRAN manual for "IOSTAT" error messages. IOSTAT value:
90.SDRV S Non zero flag in solution driver. Try to increase NON_ZERO in the parameter file. Flag:
90.PIVOT W The system contains empty equations. This is ok in runs with with missing values and different models for the traits. Check the number of missing equations (zero pivots):

90.ODRV S Non zero flag in ordering driver. If FLAG is 9 or 10 increasing
NON_ZERO under the SYSTEM_SIZE section may help. Flag:
90.VALUE S Unexpected character when reading integer or real value.
90.WRITE S I/O problem writing a file. Consult your FORTRAN manual for
"IOSTAT" error messages. IOSTAT value:
90.NSFEA S This feature/keyword/option is presently not supported:
90.INBGG W The combination of genetic groups and inbreeding has not
been sufficiently tested. Use with care!!!!

99.99 123456789 123456789 123456789 123456789 123456789 123456789 12345

M0. GENERAL REMARKS (milena)
M0.MISS S The parameter file does not contain sufficient information.
Check statement:
M0.25 E Effect listed is unknown! Check spelling of effects in COVARIANCE
and RELATIONSHIP section. Unknown effect:

M04. RELATIONSHIP section (milena)
M04.02 S File name has to be presented with INPUT keyword

MX. COVARIANCE section (milena)
MX.10 S The effect name has to be specified after keyword FOR.
Use RESIDUAL or effect name for random effects.
MX.COV S Covariance section is missing.
MX.VAL S Value specified is not allowed for keyword FORM.
Specified value was:
MX.OR S Effects listed in COMPONENTS FOR ... OR ... have to
be in the model for the same traits! Effects are:
MX.TEMP S Effect_list does not contain effect:

M1. OPTIMIZATION section (milena)
M1.ALGOR S Optimization algorithm must be specified. Use keyword ALGORITHM
and try: amoeba, uphill, powell, rosen, roslin, svarun
M1.NOALG S Specified algorithm is not suported yet.
Try: amoeba, uphill, powell, rosen, roslin, svarun.
M1.JOB W Job name is not specified! Default value is 'none'.

MM. RUN TIME ERRORS (milena)
MM.01 S More records than expected in file:
MM.ITER E Maximum number of function evaluations or rounds exceeded! Number
of rounds:
MM.02 E In parameter file, you specified file with covariances. File does
not exist! Check file name in FROM option:

KONEC Parameter file has been processed. Just checking if errors occured.

6.2 Various problems

6.2.1 The level codes appear in an unsorted order:

1

10

11

111

Remedy: write data file with leading zeros by using fortran format I4.4.

6.2.2 System of equations does not converge

When a random effect with relationship is placed in IOD the solving strategy is Jacobi. This may require underrelaxation for convergence. Set RELAX_IOD = .8 or some other value below 1.

6.2.3 Files disappear, nonsensical list output

PEST uses default files for intermediate storage. If PEST is started again in the same directory it may overwrite an intermediate file generated from the first run which is still running. When this run tries to reuse data dumped to disk before it may be incompatible as it has been overwritten by the second run. Do not start PEST from the same directory while the first job is still running.

6.2.4 Bad hit rate

When part A is used (either IOC, DENSE or SMP) the non zero elements are hashed into a buffer. As the buffer fills up the hit rate deteriorates. With values above 100 it may be worthwhile to abort the process. The first option is to increase NON_ZERO in the SYSTEM section. Secondly, covariables should always be placed first in the MODEL section.

6.2.5 Slow convergence

Some systems converge very slowly. 700 rounds are not uncommon. This is particularly true with IOC in multiple trait models as not block iteration is performed. Multi variate models with highly different incidence matrices may converge very slowly. Relationship and genetic groups also tends to reduce rate of convergence. In cases of slow convergence check the following options:

- is the stopping criterion appropriate. A value of .0001 does not make much sense for genetic evaluation of daily gain if the objective is ranking of animals.
- balance effects over the solvers. Place effects with few levels in IOC. They will be solved simultaneously, which typically improves rate of convergence.
- place effects with fewest levels first in the model
- in repeated analysis (continuous genetic evaluations) dump and load old solutions.

CHAPTER 7 PEST Command structure

Section names and keywords are written in **CAPITAL** letters in the following text.This is not a requirement for PEST! Optional section are bracketed [ ]. User input is written in *italic letters*.These brackets {} indicate repetition of the construct it encloses. Default values are underlined.

Sections can be in any order. Likewise, keywords can be in any order within sections but must not be used elsewhere.

[**COMMENT** [*header*]]
An unlimited number of lines can be added. Only the first five will be printed in the list file.

DATA
INFILE = *filename*
INPUT
{*variable_i no_of_levels* [*starting_column,length* [,*decimal_digits*]]}
[**MAX_CHAR** = 8|16]
[**DISK**]
[**OUTFILE** = *filename* [<**TEXT**|**BINARY**]

[**HYPOTHESIS**]

[**CHI**]
{{**CONTRAST** = *constant* | **0**}
{*address constant*}}

[**PEV**]
[**C11** [*number of equations*]
[**outfile**=*filename*|"**c11.pest**" [<**BINARY**]]
[**format**=' *any_legal_FORTRAN_format*']]

MODEL
{*traitname_i* = [**INT**] {*independent_variable*} [<**PREDICT**]}

[**PRINTOUT**]
[**OUTFILE** = *filename*|"**LIST.PEST**"]
[**OUTPUT**]
{*traitname_i any_legal_FORTRAN_format*}
[**PAGE** = *n*|**54**]
[**LINE** = *n*|**74**]
[**SIGN_DIGITS** = *n*|**5**]
[**XPX**]
[**XPY**]
[**BASE_ZERO**]
[**MEMORY_MAP**]

[**RELATIONSHIP**]
REL_FOR {*effect_name_i*}
INFILE = *filename*
[**MAX_CHAR** = **8**|**16**]
INPUT
ANIMAL [*starting column, length*]
M_P|**M_GP** [*starting column, length*]
F_P|**F_GP** [*starting column, length*]
[**BIRTHDATE**] [*starting column, length*]
[**GROUP**]
[**INBREEDING**]
[**DISK**]
[**UNDEFINED** = *string*]
[**OUTFILE** = *filename*[<**TEXT**|**BINARY**]]

[**SOLVER**]
[**IOC**|**SMP**|**DENSE**] [{*effect_name_i*}] [<**STOP** = *n*, **RELAX** = *n*, **MAX_ITER** = *n*]
[**IOD**] [{*effect_name_i*}] [<**STOP** = *n*, **RELAX** = *n*, **MAX_ITER** = *n*]
[**IOD_GS**] *effect_name* [<**STOP** = *n*, **RELAX** = *n*, **MAX_ITER** = *n*]
{[**STOP**|**STOP_IOC**|**STOP_IOD**|**STOP_IOD_GS** = *n* |.001]}
{[**RELAX**|**RELAX_IOC**|**RELAX_IOD**|**RELAX_IOD_GS** = *n* |1.0]}
{[**MAX_ITER**|**MAX_ITER_IOC**|**MAX_ITER_IOD**|**MAX_ITER_IOD_GS** = *n* |1000]}
[**ABS_MAX_CHANGE**|**ABS_AVG_CHANGE**|
STAND_MAX_CHANGE|**STAND_AVG_CHANGE**]

[**STARTING_VALUES**]
    [**OUTFILE** = *filename*]
    [**INFILE** = *filename*]

[**SYSTEM_SIZE**]
    NON_ZERO = *constant*

[**TRANSFORMATION**]
    [**TREATED_AS_MISSING**]
        {*traitname_i lower_bound,missing_value,upper_bound*}
    [**SCALING**]
        {*traitname_i*-|+|*|/ *constant*}
        {*covariate_i*-|+|*|/ *constant*}
    [**CHOLESKY**]
    [**CANONICAL**]

[**VE**]
    covariance_matrix | **HETERO_IN** *effect_name*
        {**VE_FOR** *effect_level_i*
        covariance_matrix_i}

[**VG**]
        {**VG_FOR** *effect_name_i*
        covariance_matrix_i}

CHAPTER 8 Known Problems

The following gives a list, a short description and possible workaround for known bugs in PEST. The present list reflects the status at version 3.0.

8.1 SCALING

In the transformation section operations can be performed on data as they are read in. If the order of traits in the transformation section differs from the order in the data section the wrong traits are scaled. The workaround is obvious: keep the traits in the same order. Also, check the DATA INFORMATION section in the list file. From Revision 2.9.2 on the means and standard deviations and all other statistics of this section are based on the scaled data. Previously, these statistics were generated prior to scaling. Thus, no apparent difference could be noted when scaling was applied. However, the equation system was setup using scaled data.

CHAPTER 9 Revision History

1. October 11, 1990: introduction of standard errors of estimates in hypothesis: There are many changes to many subroutines.
2. Tue Oct 16 09:46:02 CDT 1990: certain ordering in DATA section and MODEL lead to the wrong START element being used. In turn animals could not be found in the relationship file although they existed. The change is in RFILTER:

```
      do  1010  lp=1,datvn
      if (class(lp).eq.'M') then
      k=k+1
      elseif (class(lp).eq.'A') then
      k=k+1
      begin=start(k)
      maxani =maxlev  (lp)
      goto 30
      endif
 1010 continue

change into:

      do  1010  lp=1,datvn
      if (class(lp).eq.'M'.or.class(lp).eq.'C') then
      k=k+1
      elseif (class(lp).eq.'A') then
      k=k+1
      begin=start(k)
      maxani =maxlev  (lp)
      goto 30
      endif
```

```
1010 continue
```

3. Thu Oct 18 09:48:50 CDT 1990: change in hypothesis test. Insert before the last endif in routine HYPOTE:

```
     do  1090 i = 1, ncol
     do  1100 j = 1, ncol
     kvkori (i,j) = kvkori (i,j) * (ev/ve(1,1))
1100 continue
1090 continue
```

4. Thu Oct 18 09:49:34 CDT 1990: bug in EDGM, exchange parameters of routine I4NULL. Was:

```
     call i4null (iv(f('maxlevel')),datvn)
```

should be:

```
     call i4null (datvn,iv(f('maxlevel')))
```

5. Thu Oct 18 10:42:50 CDT 1990 in EDGM move the ENDIF as indicated:

```
1010 continue
     end if
     call map ('ka'       ,maxi*ncol,   bytei4 )
     call map ('kc'       ,maxi*ncol,   byter8 )
     call readco     (cv(pfile),nl,ncontr ,ncol,maxi,
    *hyp,mu,iv(f('kc')),iv(f('ka')))
     if (hyp) then
```

to:

```
1010 continue
     call map ('ka'       ,maxi*ncol,   bytei4 )
     call map ('kc'       ,maxi*ncol,   byter8 )
     call readco     (cv(pfile),nl,ncontr ,ncol,maxi,
    *hyp,mu,iv(f('kc')),iv(f('ka')))
     end if
     if (hyp) then
```

6. Wed Oct 24 16:09:53 CDT 1990, **Version 2.1**.This includes a new interface for hypothesis test, and some bugs are removed. Zero pivots are detected in ECLEAN. Individual changes are to numerous to be listed.

7. Mon Nov 5 16:35:47 CST 1990 insert in BLUPC behind:

```
     reldon   = .false.
the following line:
     endfla   = .false.
```

8. Tue Nov 20 14:32:20 CST 1990, **Version 2.2**. Numerical values in the parameter file can be expressions like -1/3 or 2+4/12.

9. Wed Dec 5 16:00:38 CST 1990, **Version 2.3**. Inbreeding and genetic groups bombed out. Changes in EDGM. Change in CHREAL for CRAYs.

10. Tue Apr 2 08:38:18 MET DST 1991, A problem with the F-test under an animal model. In EDGM change the lines:

```
if (ng.gt.0) then
   rankz = rankz - ng + 1
end if

to:

if (gmodel.and.ng.gt.0) then
   rankz = rankz - ng + 1
end if
```

11. Fri Jun 14 16:07:47 MET DST 1991, **Version 2.4**: the wrong coefficient was chosen for animals with only one parent missing. The routine DIAGII is replaced by:

```
function diagdi  (type, gen1, gen2)
integer*4 type
real*4    diagdi  , gen1, gen2
diagdi   = 1.
if (type .eq. 2 .or. type .eq. 4) then
diagdi   = diagdi   + gen1
end if
if (type .gt. 2) then
diagdi   = diagdi   + gen2
end if
diagdi   = 2./diagdi
return
end
```

12. Tue Jul 2 10:31:18 MET DST 1991, **Version 2.5**: some data records were skipped when reading data from disk in IOD_GS with animals that do not have pedigree data. A new routine DATA_CHECK has been introduced that terminates PEST should this condition occur.

13. Wed Aug 21 15:27:03 MET DST 1991, **Version 2.6**: animal, parent,grandparents implemented in relationship. Programmed stops built in for some special conditions that are not supported.

14. Thu Aug 29 17:16:50 MET DST 1991, **Version 2.7**: an element of the RHS was overwritten under the following circumstances: a fixed effect of a multiple trait model is solved in IOD, at least one equation empty due to **all** records of a particular level having missing values for one and the same trait.

15. Thu Sep 19 11:31:58 MET DST 1991, **Version 2.8**:In hypothesis testing estimability of certain contrasts was not determined correctly. However, estimates and their standard errors were affected. The output format in hypothesis testing was changed. Original codes were added to listoutput.

16. Tue Oct 29 08:01:35 MET 1991: When an animal was in the data file but not in the relationship file PEST would not stop gracefully but abort. In EDGM insert after the call to DFSWIT the lines:

```
   call syntax       ('KONEC',1,
  *'Errors have occured while coding data.')
```

17. Thu Nov 7 08:09:22 MET 1991: contrary, to th manual the default for missing values is not .0 but rather 11111111. To change this modify routine TRANSF and replace

```
   do  1030  i=1,ntrait
   npoint(i)=11111111.
   mpoint(i)='.0'
   mrange(i,1)=-99999999.
   mrange(i,2)=99999999.
1030 continue
```

by

```
   do  1030  i=1,ntrait
   npoint(i)=0.
   mpoint(i)='.0'
   mrange(i,1)=-99999999.
   mrange(i,2)=99999999.
1030 continue
```

18. Wed Feb 5 07:54:51 MET 1992: in hypothesis testing symbolic factorization of the coefficient matrix can be skipped. With multiple test this results in much faster execution. In routine HYPOTHE replace the line:

```
      call sdrvd(tdimx ,p,ip,ia,ja,a,km(1,j),sol,
     *nsp,isp,rsp,esp,1,flag,tol,rank)
```

by

```
      call sdrvd(tdimx ,p,ip,ia,ja,a,km(1,j),sol,
     *nsp,isp,rsp,esp,3,flag,tol,rank)
```

19. Wed Feb 5 08:07:46 MET 1992: In hypothesis testing, the error degrees of freedom are reported incorrectly in the list file. This does the affect the test itself which is correct. To correct the bug change the following line in routine HYPOTE from :

```
      rankx  = rank - rankz
      dfe = rankx
```

to:

```
      rankx  = rank - rankz
      dfe = ntot - rankx
```

20. Wed Feb 5 09:51:03 MET 1992: For hypothesis testing the following modifications drastically reduce the memory requirements: In routine EDGM change the line from:

```
      nsp = tdima *3+4*nze + 1
```

to:

```
      nsp = tdima *3+2*nze + 1

      also in routine SMPSOL change line:

      call  odrvd(tdimx ,ia,ja,a,p,ip,nsp,isp,2,flag)
```

to:

```
      call  odrvd(tdimx ,ia,ja,a,p,ip,2*nsp,isp,2,flag)
```

21. Wed Mar 25 09:04:56 MET 1992: in prt2 a modification was made to the automatic computation of the output format: with very small numbers for the solutions the previous format was too short, thus, generating “****”. Also effected is routine minmax.

22. Wed Apr 22 09:02:51 MET DST 1992: In some instances one record was skipped when using the IOD_GS solver. Replace in routine DATASO line 46:

```
      do 1040 ii = i + 1, datlen
```

to:

```
      do 1040 ii = i + 1, datlen + 1
```

23. Wed May 6 08:25:25 MET DST 1992: Version 2.9.1. In some instances the automatic format generator did not produce a suitable output. The routine MINMAX has been rewritten and a new routine NLEAD been introduced. Furthermore, routines NDIGand PRT2 are modified. The keyword SIGN_DIGITS (default set to 5) now produces output that really delivers the specified number of significant digits within each effect.

24. Tue Dec 1 14:43:54 MET 1992: Version 2.9.1. The Table DATA INFORMATION has been reorganized to cope with a larger number of traits. The effects of SCALING are now reflected by this table. The number traits has been increased to 30 (previously we had a limit of 10). The PEST message file SYNTAX.ERR has changed.

25. Mon Jan 25 09:00:16 MET 1993: Version 2.9.2. Some minor changes and modifications in the User’s Guide. Some modifications have been made to have PEST fit into the system of multivariate REML (co)variance component estimation.

26. Tue Apr 27 12:26:38 MET DST 1993: Version 3.0. The old hypothesis format is phased out. Prediction error variance is added. Also, the inverse of a part or the complete coefficient matrix can be computed. A large number of subroutines is affected.

27. Thu Jun 3 14:23:16 MET DST 1993: Version 3.1. A bug that occured when genetic groups and the disk option was specified for solver IOD_GS which carried the animal effect got fixed. Quite a few modules are affected.
28. Wed Jan 31 09:21:00 MET 1996. If some of the traits have no observations at all a division by zero may occur in data_info. This has been patched.