The Survival Kit v6.12

User's Manual

Vincent Ducrocq, Johann Sölkner and Gábor Mészáros

January 21, 2014

Contents

Preface 7
1 History 7
2 Class of models supported 7
3 Main features 7
4 The programs 9
5 Hardware requirements and compilation 10
5.1 Compilation under UNIX 10
5.2 Compilation under MS Windows 10
6 Last Changes 11
7 Disclaimer 13
8 Availability 13

1 Data Preparation Program (PREPARE - prepare.txt) 15
9 Syntax 15
9.1 NRECMAX Statement 16
9.2 NEFMAX Statement 16
9.3 NDIMAX Statement 17
9.4 FILES Statement 17
9.5 INPUT Statement 18
9.6 TIME Statement 18
9.7 CENSCODE Statement 19
9.8 DISCRETE_SCALE Statement 19
9.9 IDREC Statement 20

9.10 TRUNCATE Statement 20
9.11 TIMECOV Statement 21
9.12 CONVDT Statement 21
9.13 CLASS Statement 22
9.14 JOINCODE Statement 22
9.15 CORRELATION Statement 22
9.16 TIMEDEP Statement 23
9.17 COMBINE Statement 23
9.18 OUTPUT Statement 24
9.19 FUTURE Statement 24
9.20 PEDIGREE Statement 24
9.21 FORMOUT Statement 26
10 Sorting of the recoded file 26
10.1 COX Program 26
10.2 WEIBULL Program 27
10.3 Sorting of "future records" 28

2 Programs COX and WEIBULL 29

11 Parameter file "1" - varlist.txt 29
11.1 Syntax 29
11.2 TIME Statement 30
11.3 CODE Statement 30
11.4 ID Statement and PREVIOUS_TIME Statement 30
11.5 COVARIATE Statement 30
11.6 DISCRETE_SCALE Statement 31
11.7 JOINCODE Statement 31
11.8 Format Statement 32
12 Parameter file "2" - cox.txt / weibull.txt 32
12.1 Syntax 32
12.2 NRECMAX Statement 34
12.3 NDIMAX Statement 34

12.4 NEFMAX Statement 34
12.5 NSTRAMAX Statement 34
12.6 NTIMMAX Statement 35
12.7 NSTIMAX Statement 35
12.8 NLEVOUT Statement 35
12.9 NPGAUSS and NITER_GAUSS Statement 35
12.10 NVEC_BF Statement 36
12.11 NO_LOGARITHM Statement 36
12.12 NO_HESSIAN Statement 36
12.13 NZE_HESSIAN Statement 36
12.14 WITH_MGD Statement 37
12.15 LOGFILE Statement 37
12.16 FILES Statement 37
12.17 TITLE Statement 38
12.18 ITE_QUASI Statement 38
12.19 STRATA Statement 38
12.20 ORIGIN Statement 39
12.21 RHO_FIXED Statement 39
12.22 MODEL Statement 40
12.23 COEFFICIENT Statement 41
12.24 <ONLY_>STATISTICS Statement 41
12.25 RANDOM Statement 42
12.25.1 Model types 43
12.26 INTEGRATE_OUT Statement 44
12.27 ANIMAL_SOLUTION Statement 46
12.28 CORRELATION Statement 46
12.29 TEST Statement 47
12.30 STD_ERROR Statement 47
12.31 DENSE_HESSIAN and FORCE_POSITIVE Statements . 48
12.32 BASELINE Statement 48

12.33 KAPLAN Statement 49
12.34 SURVIVAL Statement 49
12.35 RESIDUALS Statement 50
12.36 CONSTRAINTS Statement 50
12.37 CONVERGENCE_CRITERION Statement 51
12.38 STORAGE Statement 51
12.39 STORE_SOLUTIONS and READ_OLD_SOLUTIONS Statement 51

3 A small example **53**

4 Analysis of very large data sets **69**
13 PREPARE program 69
14 Before running the COX or WEIBULL programs 70
15 COX and WEIBULL programs 70

5 Database creation tutorial **73**
16 General example 73
17 Database creation 74
17.1 Input database 75
17.2 The SAS code 75
18 Tips and Tricks 77

Preface

1 History

The Survival Kit was initially intended to fill a gap in the software available to animal breeders who generally tend to use extremely large data sets and want to estimate random effects. Methods of survival analysis have primarily been developed in the area of clinical biometrics where data sets and number of levels of effects are usually smaller. Theory about random effects in survival analysis (frailty models) was less developed than for linear models and there were no programs publicly available which deal with such models.

Although developed by animal breeders for animal breeders, the programs of the Survival Kit should therefore be interesting for people from other areas encountering similar problems of large models and random effects. To make the Survival Kit user-friendly, commands used in the parameter files mimic the SAS command language.

2 Class of models supported

The models supported by the Survival Kit belong to the following class of univariate proportional hazards models with a single continuous or discrete response time:

$$\lambda(t, \mathbf{x}(t), \mathbf{z}(t)) = \lambda_{0j}(t) \exp\{\mathbf{x}(t)'\mathbf{b} + \mathbf{z}(t)'\mathbf{u}\} \quad (1)$$

where $\lambda(t, \mathbf{x}(t), \mathbf{z}(t))$ is the hazard function of an individual depending on time t, $\lambda_{0j}(t)$ is the baseline hazard function, $\mathbf{x}(t)$ is a vector of (possibly time-dependent) fixed covariates with corresponding parameter vector $\mathbf{b}$, $\mathbf{z}(t)$ is a vector of random (possibly time-dependent) covariates with corresponding parameter vector $\mathbf{u}$.

For a detailed statistical presentation of the methodology of survival analysis, see (Cox, 1972; Prentice and Gloeckler, 1978; Cox and Oakes, 1984; Kalbfleisch and Prentice, 1980; Klein and Moeschberger, 1997)

3 Main features

Baseline hazard function:

The (possibly stratified) baseline hazard function $\lambda_{0j}(t)$ may either be unspecified or follow a Weibull hazard distribution $\lambda_{0j}(t) = \lambda_j \rho_j (\lambda_j t)^{\rho_j - 1}$. In the first case, if the failure time variable is t is continuous, (1) defines a Cox model. Estimates of $\mathbf{b}$ and $\mathbf{u}$ are obtained using what is known as a partial likelihood, the part of the full likelihood in which the baseline hazard function does not appear. When the failure time variable is discrete with few categories and many observations with the same failure time (ties), the Cox's partial

likelihood approach is no longer valid. Because the baseline hazard function can then be described with few parameters, these can be estimated together with **b** and **u** using an approach due to Prentice and Gloeckler(1978).

The second case corresponds to a Weibull model, a common type of regression model that has been shown to be flexible and often adequate for biological data. The exponential model is a particular case of a Weibull model, with ρ equal to 1.

Fixed covariates:

Any number of fixed covariates is supported. They may either be discrete (class) variables or continuous. There is no explicit limit to the number of levels of a discrete covariate (the limit will usually be a function of the computer memory available). An intercept (grand mean) is always implicitly included in the Weibull mode. When there is only one stratum and no covariate specified, this intercept is equal to $\rho \log \lambda$ where ρ and λ are the two Weibull parameters. Covariates may be time-dependent ($\mathbf{x}(t)$), where the dependency is modelled through "piecewise" constant hazard functions with jumps at times corresponding to calendar dates (e.g., January 1st) or linked to the individual itself (e.g., starting or stopping of smoking when the effect of smoking on survival is investigated).

Random covariates:

The random (discrete) covariates in vector $\mathbf{z}(t)$ may be defined to follow a log-gamma or a normal distribution. They may also follow a multivariate normal distribution where the covariance structure between individuals is modelled by the matrix of genetic relationships, a typical approach to describe the additive genetic values of individuals in animal breeding. The log-gamma was chosen because $\exp(\mathbf{u})$ is then gamma distributed, a usual assumption for frailty models. The two parameters of the gamma distribution are taken to be equal so that the expectation $E(\exp(\mathbf{u})) = 1$. With the normal distribution, $E(\mathbf{u}) = 0$. The distribution parameters (Gamma parameter for the log-gamma distribution and variance for the normal or multivariate normal distribution) may be either pre-specified or estimated alongside with the effects in the model (Ducrocq and Casella, 1996). Several random effects may be specified in the same model, but only one may involve a relationship matrix. Random effects may also be time-dependent. Although the expression *frailty term* is used to generally describe random effects in survival analysis, it was originally introduced to account for individual heterogeneity of observations as is done by the error term in the linear model. Such a term may also follow one of the distributions mentioned above and is technically treated in the same way as the other random effects.

Strata:

Stratification may be used to separate groups of individuals with different baseline hazard functions $\lambda_{0j}(t)$ where j is serving as group indicator. Together with time-dependent co-

variates, this is another mean of relaxing the required assumption of proportional hazards for all individuals over the total observation period. Only one variable (possibly time dependent) may be chosen as strata variable. The number of strata is not restricted.

In addition to estimation of fixed and random effects (and their distribution parameters), the Survival Kit offers options for calculating asymptotic standard errors of effects (only for smaller models, where the matrix of second derivatives may be actually set up), a sequence of likelihood ratio tests and different ways of setting constraints to deal with dependencies in the model. As a special feature, different values of the survivor function may be estimated for individuals with preset covariate structure. This way, it is possible to calculate estimated median survival time (for example) or survival probability to a specified age for combinations of covariate values of special interest. Generalized, martingale and deviance residuals (Cox and Snell, 1966; Klein and Moeschberger, 1997) can also be computed.

4 The programs

The Survival Kit is a set of Fortran programs, called PREPARE, COX and WEIBULL (each of these programs consists from several source code files) and two module files: *prepinclu.f* holding parameter definitions for PREPARE and *parinclu.f* with parameter definitions for COX and WEIBULL. These modules are included in each of the programs (via a Fortran *use* statement). The package works stand-alone, i.e. does not rely on any subroutines from mathematical subroutine libraries. The optimisation routines used are partly taken from public domain subroutine libraries (Liu and Nocedal, 1989; Perez-Enciso et al., 1994) and are integrated in the programs. PREPARE is used to prepare the data for the actual analysis done with either COX or WEIBULL. Data preparation includes recoding of class variables and in the presence of time-dependent covariates splitting up individual records into so-called elementary records with each elementary record covering only the time span from one change in any time-dependent covariate to the next. The recoded file may therefore have many more records than the original one.

The estimation of effects under the proportional hazards model described above is then performed by COX or WEIBULL, depending on whether the baseline hazard function is assumed to be unspecified in the Cox model or it is assumed to follow the two-parameter Weibull hazard distribution. Specifications for both models are similar, but it is computationally easier and less time consuming to estimate the parameters of distributions of the random effects under the Weibull model.

5 Hardware requirements and compilation

The programs have been written in Fortran 90 (initially in Fortran 77 up to version 5) and have been tested on PC (using Lahey's Fortran compiler) and on several UNIX platforms. No system routines are used, **except a timing subroutine ("second()" for UNIX platform) which can be replaced or switched off without any consequence**.
The size of the program may be varied through changes in parameters affecting the maximum number of records and maximum number of levels of effects to be estimated. These parameters are defined in for the data preparation via PREPARE and for programs COX and WEIBULL. From version 6.0 onwards there is no need to recompile the programs to change some specified parameters. For detailed description see the syntax for PREPARE and parameter file "2". In case you change any other parameter in *prepinclu* or *parinclu* files, it is necessary to recompile the programs.

5.1 Compilation under UNIX

Survival Kit version 5 and above consists from source code located in several different files. From this reason a special approach is needed. In case of recompilation one needs to use the supported file. To use it one have to type:

/usr/ccs/bin/make -f makefile

where "/usr/ccs/bin" is the location of the program "make". In other words, this part of the input will differ at various workstations.

5.2 Compilation under MS Windows

In case of Lahey Fortran 95 compiler there are several things one should be aware of. Program similar to *Makefile* is already included into Lahey, called *Automake*, located in Lahey compiler "Bin" directory. To use it, copy all files related to Automake to the *same* working directory (along with file "plusfort.fig") as you have the source files to compile, and start it with file "AM.bat" via command prompt, double clicking on it or from the Lahey itself (third button from top on the right side). Configuration file for the Automake is "Automake.fig" editable with "AMedit.exe" or any text editor.

In case of problems with *Automake*

- Check in the "automake.fig" file if the path for the executable file was correctly set. In case you move the automake files used previously, and the path for target

executable file is not changed, you will get no executables at place where you expect them to be.

- If you remove a file or subroutine from the folder, delete the dependency file "automake.dep" manually (re-created in the next run). It stores the program structure, and in case of changes it can still search for deleted files or skip the newly added ones.

- Many useful information are in Lahey's User's Guide (it has a chapter about automake) or on the Lahey Forum and FAQ online.

6 Last Changes

- In version 3.0, the programs PREPAREC and WEIBULLC were made available for very large applications. A few bugs were corrected and some features (like the JOINCODE, COEFFICIENT, RESIDUALS, INTEGRATE_OUT ... WITHIN ... statements, choice of input/output formats, inclusion of groups of unknown parents, use of left-truncated records with COX) were added.

- In version 3.1, the main addition was the possibility to properly analyze *discrete* failure time data. Discrete failure times with very few distinct observed values occur for example when survival time is expressed in years or parities. Then, it becomes more difficult to find proper parametric proportional hazards models and the semi-parametric approach of Cox (Cox model) is no longer suitable (an exact calculation of the partial likelihood is not feasible in general in presence of many "ties"). The approach of Prentice and Gloeckler (1978) for grouped data is much more satisfying: a full likelihood is written, involving a limited number of parameters describing the baseline hazard function. Using a reparameterization of Prentice and Gloeckler's model, it is possible to transform the grouped data model into a form *close to* an exponential regression model including a particular time-dependent covariate (*time_unit*), with changes at every time point. The PREPARE and WEIBULL programs have been modified to accommodate such a model (note however that the resulting model is *not* a Weibull model). The *only* change that is required from the user is the specification that the time scale is discrete (keyword: DISCRETE_SCALE) in the first parameter file. The use of D6 (ddmmyy) formats with years larger or equal to year 2000 is now possible (option NBUG_2000 in the *parinclu* file).

- In version 3.12 (April 2000), another set of small bugs was corrected. The statement *SIRE_DAM_MODEL* was made obsolete (*JOINCODE* does the same thing and should be preferred). It is now possible to use both the statements *STRATA* and *INTEGRATE_OUT ... WITHIN ...* at the same time.

- In version 5.0 the three programs were separated from each other in a sense, that there was no single *parinclu* file for all of them. Instead of this a *prepinclu* was introduced for the Prepare program. A number of new keywords were introduced, such as ITE_QUASI forcing to start with a number of iterations of quasi-newton algorithm before switching to full Newton. Also the ability to produce estimates for approximate animal model (Ducrocq, 2001) via keyword ANIMAL_SOLUTION, and to include solutions for the random effect already integrated out, via the BACK_SOLVE keyword. Starting from v5.0 the source code contains elements of Fortran 90, which means that Fortran 77 compilers are no longer sufficient for compilation.

- In version 6.0 parameters related to the size of the database were changed to variables and matrices with fixed size were changed to allocatables. This allowed to introduce several new keywords designed to change these crucial variables in the parameter (text) files without recompilation of the programs. Moreover *prepinclu* and *parinclu* files were changed to modules and variables not used in prepare deleted from *prepinclu*. Also the estimation of variances for two random effects at the same time is possible in both COX and WEIBULL.

- In version 6.1 it is possible to account for the correlated nature of two random effects using the CORRELATION keyword.

7 Disclaimer

The "Survival Kit" can be freely used for non-commercial purposes **provided its use is being credited** and can be freely distributed. **Use it at your own risk**. Citation (the paper is open access):

Mészáros,G., Sölkner,J., Ducrocq,V. (2013) The Survival Kit: Software to analyze survival data including possibly correlated random effects. Computer Methods and Programs in Biomedicine, 110(3): 503-510.

There is no technical support, but questions can be directed to:

Dr. Vincent Ducrocq
UMR 1313 Génétique Animale et Biologie Intégrative (GABI)
Institut National de la Recherche Agronomique (INRA)
78352 Jouy-en-Josas France
Phone: + 33 1 34 65 22 04
email: vincent.ducrocq@jouy.inra.fr

or:

prof. Johann Sölkner
Universität für Bodenkultur
Gregor Mendel Strasse 33
1180 Vienna Austria
Tel: + 43 1 47654 3272
email: johann.soelkner@boku.ac.at

or:

Dr. Gábor Mészáros
Universität für Bodenkultur
Gregor Mendel Strasse 33
1180 Vienna Austria
Tel: + 43 1 47654 3259
email: gabor.meszaros@boku.ac.at

8 Availability

The programs and this manual can be retrieved on the Web in compressed form at:

http://www.nas.boku.ac.at/nuwi-survivalkit.html
or
http://www4.jouy.inra.fr/gabi/les-Recherches/Developpement-d-outils

Chapter 1

Data Preparation Program (PREPARE - prepare.txt)

When called, PREPARE will search for the name of a parameter file called *prepare.txt* which should be in the same directory as the executable and data files. This parameter file provides the program with information about the files and variables to be used. In particular, the dependent and censoring variables are defined, class variables and variables that may be combined into new variables are stated. Information relating to the time scale may either be given in absolute values relative to the start of the individual observation or via dates. In the latter case, the date of the start of the observation has also to be given and the program will transform the information given through dates into information about time relative to the starting point of the observation. Levels of variables defined in the CLASS statement will be recoded from 1 to *no. of levels* for use by the analysis programs COX and WEIBULL. When time-dependent variables are defined (or when it is specified that the time scale is discrete with few categories), the program will "cut" the observation into pieces (called elementary records) each ranging from the time when there is one change in any of the time-dependent covariates to the next.

9 Syntax

The following statements may be used with PREPARE. Statements written in capitals are obligatory. Names between < > are omitted when they are not needed. The sequence of statements should be as shown. Comments may be included in the parameter file. The start of a comment is /*, the end is */, i.e. /* *This text is a comment* */. The text between the two delimiters may be on more than one line. No more than 150 characters should appear on one line, but a same statement can cover several lines: each statement ends with a semi-colon (;).

nrecmax *value*;
nefmax *value*;
ndimax *value*;
FILES *file1 file2 file3 file4*;
INPUT *variable type variable type ... variable type*;
TIME *variable <variable <variable>>*;
CENSCODE *variable value*;
discrete_scale;
idrec *variable*;
truncate *variable*;
timecov *variable type values*;
convdt *variable=variable–variable variable=variable–variable ...*;
class *variables*;
joincode *variable variable*;
correlation *variable variable*;
timedep *variable typediff variable typediff ...*;
combine *variable=variable+variable variable=variable+variable+variable ...*;
OUTPUT *variables*;
future *file5 file6*;
pedigree *file7 file8 variable <variable> <WITH_MGD>;*
formout *type*;

9.1 NRECMAX Statement

NRECMAX *value*;

The NRECMAX statement defines the maximum number of elementary records without necessity to recompile the entire program. With time dependent variables this number may be much larger than the number of records in your original datafile. In case this statement is omitted or negative value is given, default value from *prepinclu* will be used. *Note:* When NRECMAX is used, it should be specified *before* the FILES statement.

9.2 NEFMAX Statement

NEFMAX *value*;

The NEFMAX statement holds the maximum number of discrete (CLASS) and continuous covariates without necessity to recompile the entire program. Time dependent covariates have to be counted twice, because they require two positions each in the recoded data file. In case this statement is omitted or a negative value is given, default value from *prepinclu*

will be used.
Note: When NEFMAX is used, it should be specified *before* the FILES statement.

9.3 NDIMAX Statement

NDIMAX *value*;

The NDIMAX statement defines the maximum total number of effects in the model (size of vector of solutions). In case this statement is omitted or a negative value is given, default value from *prepinclu* will be used.
Note: When NDIMAX is used, it should be specified *before* the FILES statement.

9.4 FILES Statement

FILES *file1 file2 file3 file4*;

The FILES statement is used to define the names of the files needed in the process of data preparation for the actual survival analysis.

- *file1* is the name of the original data file (must exist in your directory).
- *file2* is the name of the recoded file produced by the program to be used by subsequent programs COX or WEIBULL. This file may be much larger than the original data file when time dependent covariates are used in the analysis, as so-called elementary record are written spanning the times between any changes in time dependent covariates.
- *file3* is the name of a file containing the original and new codes (after internal recoding) for each variable in the class statement. The file name is required even when no class statement is defined. In this case it will be an empty file.
- *file4* is the name of a file containing information about the variables in the data set to be used by subsequent COX or WEIBULL. It is one of two parameter files required by those programs. From v6 onwards this file name is set to *varlist.txt* by default, i.e. this name should appear as fourth in the parameter file for PREPARE.

Note for UNIX users: If the UPPER_CASE flag in file *prepinclu* or *parinclu* (see later for a detailed description of the parameters in *prepinclu* or *parinclu*) is set to .TRUE., all lowercase letters in the parameter file are transformed into upper case internally (to avoid troubles with variable names typed in different ways by the user). This implies that on systems where file names are case sensitive (mainly UNIX), the file name of *file1* has to be upper case to be recognised by the program. Also, the names of the files produced

by PREPARE (*file2* to *file4*) will be upper case even if stated lower case in the FILES statement.

If the UPPER_CASE flag is set to .FALSE., all names (other than keywords, which can be either upper or lower case) used in parameter files are case sensitive (they are not transformed).

9.5 INPUT Statement

INPUT *variable type variable type ... variable type*;

In the INPUT statement, variables names and their respective types are defined. The input file (*file1*) is read in free format implying that variables have to be separated by blanks and that all variables in the data set (whether needed or not in the current analysis) have to be specified. Variable names may be up to 8 characters long (otherwise the name is truncated), and may be comprised of any combination of characters, numbers and symbols, except for the following symbols: ; + − = (). The variable names should be separated by spaces (using TABs may result in an error). The following variable types are allowed:
I4 - integer number
R4 - real number
D6 - date with 6 digits (ddmmyy)
D8 - date with 8 digits (ddmmyyyy)
Note that the "Y-2000 bug" is avoided for D6 dates by adding 100 to the value of "yy" when yy is less than NBUG_2000 (set in the *prepinclu* file).

9.6 TIME Statement

TIME *variable* < *variable* < *variable* >>;

In the TIME statement, the dependent variable is defined. This is usually time, but may be other things like lifetime production, etc. Time values of 0 should be avoided. The time statement may have three different forms:

- **TIME** *variable*;

  The variable must have been specified in the input statement and contains the time till death or censoring of an individual (must be of type I4).

- **TIME** *variable1 variable2*;

variable1 is the time variable as above, *variable2* is a date (D6 or D8) indicating the beginning of the observation (must also be contained in list of input variables). This is necessary for models with time dependent covariates when changes in risk are given by dates and not by times in the TIMECOV or TIMEDEP statements.

- **TIME** *variable1 variable2 variable3*;

  variable1 is the name of the time variable. It may or may not be contained in the list of variables in the INPUT statement. *variable2* and *variable3* are dates (D6 or D8) indicating the beginning and end of the observation. They must have been defined in the INPUT statement. If *variable1* is not found in the input statement, its value will be calculated as difference between end and beginning of the observation.

Note: Only one dependent variable may be specified. If different dependent variables are of interest, separate analyses have to be run all of them starting with the data preparation step.

9.7 CENSCODE Statement

CENSCODE *variable value*;

In the CENSCODE statement, the censoring variable is defined. This is an indicator variable. The *variable* name must be identical to one of the names specified in the INPUT statement and must contain the censoring code (must be of type I4). *value* defines a number indicating (right) censored records. All other values of the variable indicate death (failure).

9.8 DISCRETE_SCALE Statement

DISCRETE_SCALE;

The DISCRETE_SCALE statement specifies that the failure time variable (*variable1* in the TIME statement) is expressed in discrete units with *few* (say, < 20) distinct values. This is in preparation for, later on, an analysis using a grouped data model (Prentice and Gloeckler, 1978) with the WEIBULL program (although it is *not* a Weibull model).

The immediate consequence of the DISCRETE_SCALE statement is the implicit definition and calculation of a time-dependent covariate called *time_unit*, taking distinct values at time $0, 1, 2, 3, \cdots,$ failure time $- 1$. An elementary record is created for each value of this *time_unit* variable.

9.9 IDREC Statement

The statement can have two mutually exclusive alternative forms:

IDREC *variable*;
IDREC *STORE_PREVIOUS_TIME*;

- In the IDREC statement, a variable is defined that is unique to each record. This variable may be needed for sorting the elementary records when using a Weibull model and is therefore obligatory when using WEIBULL as analysis program. The variable must be of type I4.

  Note: The same variable may also appear in the class statement (e.g., animal identification for an animal model. In this case, the variable will appear twice in the recoded *file2* that is a result of this program (first, unrecoded as the 3rd variable on the record, and second, recoded as one of the class variables).

- The alternate formulation (IDREC *STORE_PREVIOUS_TIME*) is needed in the special case when later on in the WEIBULL program, a (log-gamma) random time-dependent covariate is integrated out (and only then). The beginning of each interval is then stored at the place where the identity of the record is normally stored.
  Note: if the "INTEGRATE_OUT *variable1* WITHIN variable2" statement is used in the WEIBULL parameter file, the basic form "IDREC *variable*" is maintained (no need of "STORE_PREVIOUS_TIME").

9.10 TRUNCATE Statement

In the TRUNCATE statement, information is provided for handling left truncated records. For these records, the origin point (e.g. date of birth) is known but occurs before the beginning of the study period, so that the covariate status between origin point and begin of study period is unknown. It has the following form

TRUNCATE *variable*;

variable holds the time of the truncation point. It must be of type I4, D6 or D8. If of type I4, it specifies the time between the beginning of the observation and the point in time when the individual actually entered the study (truncation point). If of type D6 or D8, it holds the truncation date and time will be calculated as difference between this date and the date specified in the first argument of the time statement (types must be identical). If the variable is coded 0 in the data set, the record is treated as untruncated (independently from data type).

Left truncated records are now (in version 3.0 and above) handled by **both** the WEIBULL and the COX programs.

9.11 TIMECOV Statement

TIMECOV *variable type values*;

The TIMECOV statement is needed when the hazard changes with time (this may be calendar time or time as measured from the beginning of each observation) independent from special (stochastic) events.

- *variable* defines the name of the timecov variable. This must be a new name not specified previously (in the INPUT or TIME statements).
- *type* may be either integer (I4) or a date (D6 or D8).
- *values* indicates a list of values (with a maximum number specified through the parameter MAXFIG in the *parinclu* file) defining points in time when the hazard changes.

If *type* is integer, *values* indicate changes in the hazard rate relative to the beginning of each observation. If *type* is a date, it determines points in the calendar time, when the risk changes for all individuals. The change on the time axis of the individual is then calculated as difference between the respective date given and the date indicating the starting point of the observation in the TIME statement. Therefore, when using a TIMECOV statement with *type* as a date, the TIME statement must hold the date of the beginning of the observation as second argument and the date of the end of the observation as third argument. The variable types of those arguments must be identical.

Note: Only one TIMECOV statement can be handled by the program (in version 3.0 and above).

9.12 CONVDT Statement

CONVDT *variable=variable–variable variable=variable–variable ...*;

With the CONVDT statement, the difference between two date variables is calculated and written onto a new time variable. The variable to the left of the equality sign must be a new name, the variables to both sides of the minus sign must be dates of the same type (D6 or D8). This statement will be used for calculation of covariates (e.g., age at first calving from date of birth and date of calving) but not for the dependent variable, which is processed in the TIME statement.

Note: the variable to the left of the minus sign is the date later in time (e.g., date of first calving), to the right is the one earlier in time (e.g. date of birth).

9.13 CLASS Statement

CLASS *variables*;

The CLASS statement lists the names the classification variables to be used in the analysis. Independent variables defined in the MODEL statements of the COX or WEIBULL programs which are not defined in the CLASS statement will be treated as continuous covariates. All variables defined in the class statement will be internally recoded (list of codes in *file3*). Codes are transformed back to original values for the listing of the results from COX and Weibull. The recoded values are needed, though, in the CONSTRAINTS statement of COX and WEIBULL (see there).

9.14 JOINCODE Statement

JOINCODE *variable variable*;

The JOINCODE statement specifies that the two variables indicated must be recoded together, as if they pertained to a single list. This is necessary, e.g., when sires and maternal grand-sires are specified as different variables in the input statement but must appear in a single recoded list: sires can also appear as maternal grand-sires and their additive genetic effect as grand-sires is 0.5 times their effect as sires. This is also one way (out of 2) for the definition of sire-dam models: sires and dams are recoded together using the JOINCODE option and an 'overall' relationship matrix is used for both sires and dams.

Both variable names must have been declared previously in the CLASS statement.

9.15 CORRELATION Statement

CORRELATION *variable variable*;

The CORRELATION statement specifies the two variables used as correlated random effects in Cox or Weibull. The effect of this keyword is similar to the one of JOINCODE (i.e. the two variables are coded together), but the JOINCODE keyword does not show up in the *varlist.txt* and the list of old and new codes is written out for both variables in *codelist.txt*. CORRELATION and JOINCODE keywords should not be used together in the same run.

Both variable names must have been declared previously in the CLASS statement.

9.16 TIMEDEP Statement

TIMEDEP *variable typediff variable typediff .....*;

Time dependent covariates, i.e., independent variables whose value may change during the time an individual is observed (may be CLASS or continuous covariates) are defined with this statement. *variable* names must be defined in the INPUT statement.

In the input file (*file1*), any changes of value of these variables with time have to be given at the end of each record as triplets in the following way (see also section about files):

- consecutive number of the time dependent covariate. This means that one has to count on which position in the record the variable (with its initial value) is located and this value has to be supplied.
- date or time of change in covariate value.
- new covariate value.

typediff relates to the way the difference between beginning of the observation and the actual change of the covariate is given. If it is I4, it is given as time, if it is D6 or D8, it is calculated as the difference between the date given in b) and the date describing the beginning of the observation in the TIME statement (both dates must be of the same type).

One of the most frequent problems (errors) that users encounter with the Survival Kit is related to an inadequate preparation of these triplets (for example, with time of change $= 0$ (or even ≤ 0) or dates of change before the origin (or the truncation) point or after the failure date.

9.17 COMBINE Statement

COMBINE *variable* $=$ *variable* $+$ *variable variable* $=$ *variable* $+$ *variable* $+$ *variable...*;

The COMBINE statement is used to to produce 'interaction effects' between two or three variables defined in the CLASS statement. Continuous variables are not allowed. The variable to the left of the equality sign must be a new name. If one of the variables combined is time dependent (defined in the TIMEDEP statement), the resulting variable will also be time dependent.

9.18 OUTPUT Statement

OUTPUT *variables*;

In the OUTPUT statement, the variables defined in the *variables* list will be written to the output file (*file2* of the FILES statement). This file will then be used for the analysis with the COX or WEIBULL programs. The *variables* must be of type I4 or R4 when defined in the INPUT statement. All variables newly defined in subsequent statements (TIME, CONVDT, COMBINE) are also allowed. No date variables (D6 or D8) may be included into the output file!

9.19 FUTURE Statement

FUTURE *file5 file6*;

In the FUTURE statement, you provide information about a set of records, for which you want a printout on predicted values of the survivor curve (quantile values or functional values of the survival function at given times).

- *file5* is the name of the original file holding the 'future' records (must exist in your directory). The structure of the file must be exactly identical to that of *file1*

- *file6* is the name of the recoded file produced by the program identical in structure to the *file2* data file. It will be used by COX or WEIBULL to produce values related to the corresponding survival functions. Note: if you have a future statement, you also need to have an IDREC statement, as elementary records have to be sorted by identity of the original records.

9.20 PEDIGREE Statement

In the PEDIGREE statement, you give information about a pedigree file for including relationships between individuals into the model. If you use the JOINCODE or CORRELATION in Prepare then the first variable name from JOINCODE or CORRELATION should be used here. Depending on the model to be used later, it can have three alternative forms:

PEDIGREE *file7 file8 variable;* (for sire or animal models,
or for sire-dam models if "JOINCODE sire dam;" is used)

PEDIGREE *file7 file8 variable1 variable2;* (for sire-dam models)
PEDIGREE *file7 file8 variable WITH_MGD;* (for maternal grand dam models)

- *file7* is the name of the original pedigree file (must exist).
- *file8* is the name of the recoded pedigree file produced by the program.
- *variable* is the name of the random variable (animal or sire) in the data file (must be stated in INPUT and OUTPUT).
- *variable1* and *variable2* (for sire-dam models, unless JOINCODE is used) are the names of the sire and dam variables in the data file (must be stated in INPUT and OUTPUT)
- *WITH_MGD* should be used, when maternal grand dam is included into evaluation. In this case *only* the pedigree file will consist from *five* columns.

FORM OF THE PEDIGREE FILE:
The pedigree file has to consist of four (five) columns:

- 1 – identity of animal
- 2 – sex code in sire-dam models (1 = male, 2 = female, -1 for group of unknown parents); may be used for other purposes in other situations
- 3 – identity of sire
- 4 – identity of dam (or maternal grandsire)
- 5 – identity of maternal grand dam (when WITH_MGD option is used)

Unknown parents are defined as 0 or *with negative values* corresponding to groups of unknown parents.
Important: there should be one line defined for *each* animal, *each* parent and *each* group of unknown parents.
See "Model types" section for the RANDOM statement of Weibull and Cox for more detailed information about sire and animal models.

9.21 FORMOUT Statement

FORMOUT *type*;
FORMOUT *FIXED_FORMAT* I*n* F*n.d*;

With the FORMOUT statement, the format of the recoded output files is specified (these are *file2* of the FILES statement and *file6* of the FUTURE statement). If a fixed format is chosen, the type FIXED_FORMAT must be followed by the Fortran description of formats for integers and reals ("I*n* F*n.d*", e.g., "I8 F12.5" for integers with 8 digits and reals with 12 digits, including 5 decimal figures). The other admitted type is FREE_FORMAT

- FREE_FORMAT is the default option.
- FIXED_FORMAT is useful when you need to sort output files but your sorting routine is not really adequate when the records are not strictly presented in columns (especially on PCs). Be careful: if the formats for integers and/or reals are not big enough, the results will be incorrect.

10 Sorting of the recoded file

For the use in COX or (sometimes) in WEIBULL, the recoded file produced by PREPARE (*file2* of the FILES statement) has to be sorted. The sorting order is different for COX and WEIBULL. A sorting routine is **not** included in the Survival Kit, as it is often machine dependent. The specification of a fixed format (FORMOUT FIXED_FORMAT) may be necessary when compiling the Survival Kit with some (PC) compilers, where the output of a single record in free format is put into separate lines when it exceeds 80 characters. You have to find out!

10.1 COX Program

This is the sorting order (from highest to lowest level) for COX:

- *STRATA variable: in ascending order* (this is necessary only when a STRATA statement is used in COX; see below)
- *TIME variable: in descending order* (the TIME variable is always the first variable in the recoded file)
- *CENSCODE variable: in ascending order* (the censoring variable is always the second variable in the recoded file)

10.2 WEIBULL Program

Several special cases exist[1]. These cases are *mutually exclusive*:

- A STRATA statement is used in WEIBULL. The sorting order should be:
  - *STRATA variable: in ascending order*
  - *IDREC variable: in ascending order* (the IDREC variable is always the third variable in the recoded file)
  - *TIME variable: in ascending order* (the TIME variable is always the first in the recoded file).
- A LOGGAMMA random variable is defined in the WEIBULL parameter file. This random variable is algebraically integrated out in order to reduce the number of parameters to estimate or to obtain an exact marginal posterior distribution. This is done using the INTEGRATE_OUT statement in the parameter file (see below).

  Again two situations exist:
  - If the INTEGRATE_OUT statement is followed by the name of one variable only (e.g., INTEGRATE_OUT sire), the sorting order should be:
    - *the variable to be integrated out: in ascending order*
    - *IDREC variable: in ascending order* (the IDREC variable is always the third variable in the recoded file)
    - *TIME variable: in ascending order* (the TIME variable is always the first in the recoded file).

    Note that if the variable to be integrated out is time-dependent (e.g. herd-year-season), the statement IDREC STORE_PREVIOUS_TIME **must** have been used.
  - If the INTEGRATE_OUT statement is followed by the name of two variables (e.g., INTEGRATE_OUT year WITHIN herd) and if the original records were already sorted by herd, the statement IDREC STORE_PREVIOUS_TIME is **not needed**, and there is no need to sort according to the variable "year". Records should simply be sorted by IDREC variable and ascending TIME variable. This is the natural way records are recoded and therefore, the need to sort the elementary records disappears.
- **General case:**

  In all other situations, there is no need to sort: the recoded data set is naturally recoded by IDREC variable and ascending TIME variable.

[1] also applies to the grouped data model of Prentice and Gloeckler (1978)

10.3 Sorting of "future records"

The sorting of "future records" (*file6* of the FUTURE statement) is identical to the sorting order for WEIBULL (general case), irrespective of whether COX or WEIBULL is used later on.

Chapter 2

Programs COX and WEIBULL

Depending on the parameterisation of the baseline hazard function, either program COX or program WEIBULL may be called after data preparation with PREPARE. Using the same recoded data set, various alternative survival analyses may be carried out with both programs. The programs require two different parameter files. The first is produced by PREPARE. The second parameter file essentially describes the model of analysis. Most statements used in the second parameter file may be called by both programs, a few (mainly related to parameters for log-gamma distributed random effects) may only be called by WEIBULL. Generalized residuals are computed only by COX.

11 Parameter file "1" - varlist.txt

As pointed out above, this parameter file is produced by PREPARE (*file4* of the FILES statement of PREPARE). The default name is set to *varlist.txt*. Note that you normally will **not** want to change anything in this file. This section is written to help the user in understanding the file.

11.1 Syntax

The following statements are used in this parameter file.

TIME *position*;
CODE *position*;
ID *position* or **PREVIOUS_TIME** *position*;
COVARIATE *variable nlevels pos1 pos2 variable nlevels pos1 pos2 ....*;
discrete_scale;
joincode *variable1 variable2*;

format_type;

11.2 TIME Statement

TIME *position*;

position is a figure which labels the position of the dependent (TIME) variable on the recoded data file (*file2* of the FILES statement of PREPARE).
Note: The position of the time variable is always 1.

11.3 CODE Statement

CODE *position*;

position is a figure which labels the position of the censoring code on the recoded data file (*file2* of the FILES statement of PREPARE). In this file, censored records are always coded 0, uncensored records are coded 1, but other codes are also used: -2 for the first elementary record refering to a truncated observation, -1 for an elementary record indicating a change in a time-dependent covariate
Note: The position of the censoring code is always 2.

11.4 ID Statement and PREVIOUS_TIME Statement

The two statements are mutually exclusive.

ID *position*;
PREVIOUS_TIME *position*;

position labels the position of the record identification code (ID Statement) or of the beginning of the time period for an elementary record in the special case described in the IDREC *STORE_PREVIOUS_TIME* statement of program PREPARE (required when a time-dependent covariate is integrated out in the WEIBULL program, and only then).
Note: The position of the identification variable is always 3.

11.5 COVARIATE Statement

COVARIATE *variable nlevels pos1 pos2 variable nlevels pos1 pos2 ....*;

The COVARIATE statement gives information about the covariates that might be used in the model of analysis. These are all variables in the OUTPUT statement of the PREPARE

parameter file except for the TIME and CODE variables described above.

The covariates are listed one per line, first continuous and then class covariates. *variable* is the name of the covariable, *nlevels* gives the number of different levels found for class variables (zero for continuous covariates), and *pos1* and *pos2* give the position(s) of the covariate on the recoded file (*file2* in the FILES statement of the PREPARE parameter file). When *pos1* and *pos2* are identical, the covariate is time independent. When *pos1* and *pos2* are different (subsequent) numbers, the covariate is time dependent and the covariate values on *pos1* and *pos2* give the status of the covariate before and after the change of the hazard function of an individual due to a change in this (or another) time dependent covariate.

11.6 DISCRETE_SCALE Statement

DISCRETE_SCALE;

The DISCRETE_SCALE statement indicates that the recoded file is prepared to be used with the WEIBULL program to fit the grouped data model of Prentice and Gloeckler (1978).

It is the direct result of the usage of the DISCRETE_SCALE in the PREPARE parameter file. The consequence is the definition of a specific time-dependent covariate called *time_unit*, which changes values at each time point $(1, 2, \cdots)$ between the origin and the observed failure time or censoring time. A corresponding number of elementary records is therefore created.

The statement also avoids the need to repeat it in the user-supplied parameter file "2". If one wants to run a regular Weibull model, one can simply comment out this statement in parameter file "1" by adding /* and */. However, if a lot of elementary records were created because of the definition of the *time_unit* time dependent variable, it may be more efficient to start again, running PREPARE after deleting the DISCRETE_SCALE statement in its own input parameter file.

11.7 JOINCODE Statement

JOINCODE *variable1 variable2*;

The JOINCODE statement indicates that the two variables *variable1* and *variable2* were recoded together, in a single list. This is the case, e.g., when sires and maternal grand-sires are specified as different variables but used in a sire-maternal grand sire model: sires may appear as maternal grand-sires and their additive genetic effect as grand-sires is 0.5 times their effect as sires. This statement is the direct result of the usage of the JOINCODE statement in the PREPARE parameter file.

11.8 Format Statement

format_type;

format_type displays the format of the recoded (output) files from PREPARE (in the PREPARE parameter files, these are *file2* of the FILES statement and *file6* of the FUTURE statement).

If a fixed format is chosen, *format_type* is FIXED_FORMAT and is followed by the Fortran description of formats for integers and reals ("I*n* F*n.d*", e.g., "FIXED_FORMAT I8 F12.5" for integers with 8 digits and reals with 12 digits, including 5 decimal figures). The other possible expressions for *format_type* is:
FREE_FORMAT; *(= default)*

12 Parameter file "2" - cox.txt / weibull.txt

In this parameter file, the statistical model used in the data analysis is described together with various options regarding storage, statistical tests and output. Most statements and options are valid for both COX and WEIBULL programs. Therefore, a parameter file with same structure can be used with both programs. However, a few statements apply to only one of these programs and should be commented out when the other program is run, in order to avoid warning or error messages. These will be indicated below.*cox.txt* should be used as the default file name for COX, and *weibull.txt* as the default file name for the WEIBULL program.

The grouped data model of Prentice and Gloeckler (1978) can also be fitted using the WEIBULL program, although it is not a Weibull model. For the user, the parameter file "2" has exactly the same form as for the Weibull model, the use of the grouped data model being already specified in the parameter file "1" through the DISCRETE_SCALE keyword. The latter will *automatically* generate an extra time-dependent covariate *time_unit* which will appear in the model (do not repeat it in the MODEL statement). Note that a few statements are not applicable to such a model (e.g., RHO_FIXED or INTEGRATE_OUT),

12.1 Syntax

The following statements are used in this parameter file. Statements written in capitals are obligatory. Names between < > are omitted when they are not needed. The sequence of statements should be as shown.
Comments may be included in the parameter file. The start of a comment is /*, the end is */, i.e. /* *This text is a comment* */. The text between the two delimiters may be

more than one line.
Sometimes the statement or option names are longer than 8 characters (e.g., the statement **read_old_solutions** shown below) In this case, only the first 8 characters (i.e. **read_old** in our example) are significant to the programs, the rest may or may not be written by the user.

/* parameters */
nrecmax *value*;
ndimax *value*;
nefmax *value*;
nstramax *value*;
ntimmax *value*;
nstimax *value*;
nlevout *value*;
npgauss *value*;
niter_gauss *value*;
nvec_bf *value*;
no_logarithm *value*;
no_hessian;
nze_hessian *value*;
with_mgd;
logfile < *filename* > ;
/* keywords */
FILES *file1 file2 file3 <file4 file5 file6>*;
title *Title of analysis*;
ite_quasi *value*;
strata *variable*;
origin *variable* or *value*;
rho_fixed *value*;
MODEL<*variables*>;
coefficient *variable value*;
<**only_**>**statistics**;
random *variable* <**estimate** <**moments**>> *distribution <rules> parameter<s>*
<repeat sequence for next variable<s>> ;
integrate_out < **joint_mode** > < **back_solve** > *variable1* < **within** *variable2* >;
animal_solution *filename*;
correlation *variable variable* <**estimate**> *value*;
test <*type<s> of test*>;
std_error;
dense_hessian;
force_positive;

baseline;
kaplan;
survival *file7 file8 option*;
residual *file7 file8*;
constraints *options*;
convergence_criterion *value*;
storage on_disk or **in_core**;
store_solutions;
read_old_solutions;

12.2 NRECMAX Statement

NRECMAX *value*;

NRECMAX is the maximum number of elementary records in recoded data file. With time-dependent variables this number may be much larger than the number of records in your original data file.

12.3 NDIMAX Statement

NDIMAX *value*;

NDIMAX is the maximum total number of levels of effects in the model (size of the vector of solutions).

12.4 NEFMAX Statement

NEFMAX *value*;

NEFMAX is the maximum number of covariates. These are both discrete (CLASS) covariates and continuous covariates. Time dependent covariates have to be counted twice, because the states before and after change occupy two positions on the recoded data file.

12.5 NSTRAMAX Statement

NSTRAMAX *value*;

NSTRAMAX is the maximum number of strata, i.e. levels of the strata variable defined in the STRATA statement.

12.6 NTIMMAX Statement

NTIMMAX *value*;

NTIMMAX is the largest possible value of the (dependent) time variable. It is necessary as an upper limit for an efficient computation of log(time) and exp(time) and when calculating functional values of the survivor curve (SURVIVAL statement of COX and WEIBULL). In case this statement is omitted or a negative value is given, default value from *parinclu module* will be used.

12.7 NSTIMAX Statement

NSTIMAX *value*;

NSTIMAX denotes the maximum number of distinct times or quantiles that can be defined in the SURVIVAL statement options SPECIFIC and QUANTILE. In case this statement is omitted or a negative value is given, default value from *parinclu module* will be used.

12.8 NLEVOUT Statement

NLEVOUT *value*;

NLEVOUT is the maximum number of levels for the random effect which is integrated out via INTEGRATE_OUT statement. In case this statement is omitted or a negative value is given, default value from *parinclu* will be used.

12.9 NPGAUSS and NITER_GAUSS Statement

NPGAUSS *value*; **NITER_GAUSS** *value*;

These two parameters are used in the numerical integration to get moments of the marginal posterior of a random effect variance parameter (when the MOMENTS option in the RANDOM statement is used). If they are omitted or a negative value is given, default value from *parinclu* will be used.

- NPGAUSS defines the number of points used during Gauss-Hermite integration (has to be between 3 and 5).
- NITER_GAUSS gives the number of iterations in the Gauss-Hermite integration process.

12.10 NVEC_BF Statement

NVEC_BF *value*;

NVEC_BF defines the number of vectors used to store the approximation of the Hessian (the rank of approximation of Hessian matrix). Values between 3 and 20 might be used. In case this statement is omitted or negative value is given, default value from *parinclu* will be used.

12.11 NO_LOGARITHM Statement

NO_LOGARITHM *value*;

NO_LOGARITHM defines the form of the *rho* parameter used in WEIBULL maximalisation routine. This statement overrides the default settig in *parinclu* (negative values are skipped without notice). There are *two* possibilities:

- NO_LOGARITHM 1; when *rho* is used, or
- NO_LOGARITHM 0; when *log(rho)* is used in maximalisation routine.

12.12 NO_HESSIAN Statement

NO_HESSIAN;

The Hessian matrix is needed to compute standard deviations of estimated effects (STD_ERROR statement) and to find dependencies (option FIND in the CONSTRAINT statement). It is fully stored in COX and in WEIBULL when DENSE_HESSIAN is stated.
By using NO_HESSIAN the Hessian matrix is omitted (not needed), otherwise it is stored by default.

12.13 NZE_HESSIAN Statement

NZE_HESSIAN *value*;

Declares the number of nonzero elements in the Hessian matrix. If it is omitted or a negative value is given, default value from *parinclu* will be used.

12.14 WITH_MGD Statement

WITH_MGD;

WITH_MGD is to indicate whether the maternal grand dam was included into the evaluation. In this case the same keyword should be used in the PEDIGREE statement of PREPARE.

12.15 LOGFILE Statement

logfile < *filename* > ;

If LOGFILE is present in the parameter file, the output log (i.e. computation details) will be saved to a file instead of being printed to the screen. The < *filename* > is optional. If no name specified by the user, the log will be saved to *logfile.txt* .

12.16 FILES Statement

FILES *file1 file2 file3 <file4 file5 file6>*;

The FILES statement is used to define the names of the files needed for running COX or WEIBULL.

file1 is the name of the recoded data file (*file2* of FILES Statement in PREPARE). This file must exist in your directory.

file2 is the name of the file containing the original and new codes (after internal recoding) for each CLASS variable. (*file3* of FILES Statement in PREPARE). The file name is required even when no CLASS variables were defined in PREPARE. In this case it will be an empty file.

file3 is the name of the output file where the results of the analysis are listed.

file4 is the (recoded) pedigree file *file8* of the PEDIGREE Statement of PREPARE. This file is not obligatory and will only be needed for analyses where the additive genetic relationship structure between individuals is considered.

file5 is only needed when for very large applications, solutions from a previous run are to be read in because the READ_OLD_SOLUTIONS statement is used (see below).

file6 is only needed when for very large applications, solutions are to be stored for a subsequent run with the STORE_SOLUTIONS statement (see below).

Files 1, 2 must exist in your directory, files 4 and 5 must exist when subsequent statements make use of them; file 3 and 6 will be new files (**Note:** existing files with the same name are overwritten without warning). In situations where files 5 or 6 are required but not files 4 or 4 and 5, dummy names have to be stated for the latter ones.

Note for UNIX users: Unless the UPPER_CASE flag in file *parinclu* (see later for a detailed descriptions of the parameters in *parinclu*) is set to .TRUE., all lowercase letters in the parameter file are transformed into upper case internally (to avoid troubles with variable names typed in different ways by the user). This implies that on systems where file names are case sensitive (mainly UNIX), the file name of *file1* has to be upper case to be recognised by the program. Also, the names of the files produced by PREPARE (*file2* to *file4*) will be upper case even if stated lower case in the FILES statement. If UPPER_CASE is set to .FALSE. in file *parinclu*, file and variable names will not be recognized if typed differently.

12.17 TITLE Statement

TITLE *Title of analysis*;

The TITLE of analysis may fill the rest of the line (up to position 150) after the statement name. It will be written at the beginning of the output files (*file3* of the FILES statement above and *file8* of the SURVIVAL statement below).

12.18 ITE_QUASI Statement

ite_quasi *value*;

This keyword specifies the number of quasi-newton iterations before switching to full Newton steps. If the user specifies *ite_quasi* 20 the program will switch to full Newton algorithm after 20 quasi-newton iterations. If *ite_quasi* is set to 0, the program will start with the full Newton.

12.19 STRATA Statement

STRATA *variable*;

Stratification is an extension of the proportional hazards model. With stratification, the assumption of the proportionality of hazards is restricted to individuals within subgroups of the population, where grouping is defined by the *variable* indicated in the STRATA statement. Only one variable can be used as strata variable. If a combination of variables is more sensible (e.g. year-season), the individual variables may be combined in the COMBINE statement of PREPARE.

Note: Do not forget that in models with strata, data have to be sorted by strata (ascending) and within strata by the time variable (descending) and the censoring variable (ascending). See section on *sorting* in Chapter 1.

12.20 ORIGIN Statement

ORIGIN *variable*;
or
ORIGIN *value*;

Using the ORIGIN statement a shift in the time scale is assumed. This statement is permitted only with the WEIBULL program. There are two possibilities:

- If an integer constant is specified, the program takes this as a starting time point for the evaluation. The value of the constant should be lower than the time of the first event.
- In case a CLASS variable is specified here, the program will assume a new starting point for each change of the variable.

It is possible to use the ORIGIN and STRATA statements together, for example:

STRATA parity;
ORIGIN parity;

This results to a definition of a new Weibull distribution for each parity.

12.21 RHO_FIXED Statement

RHO_FIXED *value*;

This statement can be used only with the WEIBULL program (with COX, a warning message is printed and the statement is ignored). It specifies a fixed value for the Weibull

parameter ρ. For example, RHO_FIXED 1.0 will constrain ρ to be 1, therefore defining an exponential regression model instead of a more general Weibull model.

This statement is ignored when a grouped data model is fitted (then ρ is *always* assumed to be 1).

12.22 MODEL Statement

MODEL < *variables* >;

The MODEL statement specifies the independent variables affecting the dependent variable described in the TIME statement (remember that only one dependent variable may be specified there). The *variables* listed must be a subset of the variables defined in the COVARIATE statement of Parameter file 1.

Model building capabilities: Discrete (class) and continuous variables may be included. No covariate name at all is also accepted.

no covariate: if no variable is specified between MODEL and the semi-colon, the COX model can be used to simply compute a nonparametric estimate of the survivor curve (KAPLAN statement, see below) for each stratum. The WEIBULL program with no covariate specified will lead to the fit of a 2-parameter Weibull function for each stratum.

discrete covariates: they have to be stated in the CLASS and COMBINE statements of the parameter file for PREPARE.
continuous covariates: all variables in the MODEL which have not been specified via CLASS and COMBINE. Polynomial regression models may be fitted by stating *variable**N* where *N* is the power to which the covariate value is taken. For example, for a third order polynomial in *varx*, the statement would be:
MODEL *varx varx**2 varx**3 + any number of other discrete and continuous covariates*;
Covariables are always automatically centered for computations (i.e. their mean value is subtracted from each observation). For polynomial expressions, the values are first taken to the power of *N*, the mean value is then calculated and subtracted for this new variable. The mean value (or the sum of mean values if there are more than just one continuous covariate) is printed in the output of COX and is incorporated to the intercept in the output of WEIBULL. This is important to know when you want to draw the regression curves).

interactions between discrete covariates: They may only be fitted by combining the original values of two or three variables into a new one via the COMBINE statement of PREPARE.
Interactions between class and continuous covariates (i.e. individual regressions) and

nested models are currently not supported.
Variables are treated as fixed unless they are stated in the RANDOM statement described below.

When a grouped data model (= Prentice and Gloeckler's model) is fitted, an extra time-dependent covariate *time_unit* will be *automatically* generated and will appear in the model (do not repeat it in the MODEL statement). The purpose of this covariate is the estimation of the baseline hazard function at the same time as the regression parameters.

12.23 COEFFICIENT Statement

COEFFICIENT <variable value <variable value> >;

The COEFFICIENT statement specifies that the covariate *variable* will always be multiplied by the coefficient *value*. This is especially useful when used in conjunction with the statement JOINCODE in PREPARE. For example, a typical sire-maternal grand-sire used in animal breeding will be obtained by specifying:
JOINCODE sire mgs; (in the data preparation parameter file, for PREPARE)
and
COEFFICIENT mgs 0.5;

Also, the best possible way to define a sire-dam model is:

JOINCODE sire dam; (in the data preparation parameter file, for PREPARE)
and
COEFFICIENT sire 0.5 dam 0.5 ;

12.24 <ONLY_>STATISTICS Statement

ONLY_STATISTICS;
STATISTICS;

These statements can be used only with the WEIBULL program (they are ignored otherwise). They request the computation and the printing of a number of elementary statistics related to each level of the class variables specified in the MODEL statement (number of observations, number of observed failures, age at failure, average value of continuous covariates for all observations and for the uncensored ones, etc..). If STATISTICS is preceded by ONLY_ , the program will stop as soon as these values have been printed.

12.25 RANDOM Statement

RANDOM *variable* < *ESTIMATE* < *MOMENTS* >> *distribution* < *rules* > *parameter* < *s* >< *repeat previous sequence for next variable* < *s* >>;

The RANDOM statement gives information on *variables* to be treated as random. The distribution parameters of the random covariates are either assumed to be known or they may be estimated (optional parameter *ESTIMATE*).

distribution allows to specify the distribution that the random variable is assumed to follow. The user may choose from 3 alternatives:
LOGGAMMA: the levels of the random effect follow a log-gamma distribution. This is identical with assuming a gamma frailty term. The frailty term, say w, is defined as a multiplicative term to the usual hazard function with fixed effects only, i.e, $w = \exp\{\mathbf{z}(t)\mathbf{u}\}$.
NORMAL: the levels of the random effect are independently normally distributed. This assumption is not so common in survival analysis but for rather large gamma parameters ($\gamma > 10$), the log-gamma and normal are similar and the normal distribution is needed to make the step to the multivariate normal distribution described next.
MULTINORMAL rules: the levels of the random effect follow a multivariate normal distribution with covariances between levels being induced by genetic relationships. Two types of relationships are allowed and may be stated via the *rules* parameter: *USUAL_RULES* are the relationships under an animal model; *MGS_RULES* relate to a sire model or a sire-maternal grandsire model, accounting for male relationships. Note that *SIRE_DAM_MODEL* that was used in earlier versions to relate sires and dams in a sire-dam model in which sires and dams had been recoded separately is now obsolete and not accepted any more. Instead, simply use the *JOINCODE* option in PREPARE.F and *...MULTINORMAL USUAL_RULES sire ...* for sire-dam models). When MULTINORMAL is used, information about the covariances between individuals has to be provided via a pedigree file (*file4* of the FILES statement).

parameter is the distribution parameter (gamma parameter with the log-gamma distribution and the variance with normal or multivariate normal distributions). It may be preset, in which case one value of the parameter has to be provided or it may be estimated (see ESTIMATE option below), in which case three values have to be given (see below)

ESTIMATE: when stated after the variable name, the distribution parameter is estimated as the mode of its marginal posterior density which is approximated by Laplacian integration.
The *parameter* value is replaced by three values: the first two values are the bounds of the interval to be searched, the third value gives the final tolerance (accuracy). Using the ESTIMATE option, the parameter of only one random variable may be estimated at one time. If the estimated value is at one of the bounds prespecified, a warning message is printed and the analysis should be repeated changing the values of the bounds. (Example:

you stated parameters 0.01 0.1 0.001 as lower bound, upper bound and tolerance, which means that the program must look for the mode of the marginal posterior density in the interval]0.01, 0.1[and the searching process is stopped when the current interval for the estimate is smaller than 0.001. If the program output tells you "the mode is between 0.0100 and 0.0109 and the best value is 0.0104", you need to reset the bounds, e.g., to 0.005 and 0.011 and start again). The STORE_SOLUTIONS and READ_OLD_SOLUTIONS statements described below may be used to avoid starting from scratch again.
When the INTEGRATE_OUT JOINT_MODE statement is used (only with a Weibull model and for a log-gamma random effect), the ESTIMATE statement should not be used (for the log-gamma random effect): the gamma parameter is then estimated jointly with the other effects after *exact algebraic* integration of the log-gamma random effect.

MOMENTS: this option may be used together with ESTIMATE to compute estimated mean, standard deviation and skewness of the marginal persterior value of the distributional parameter. The computation is based on an iterated Gauss-Hermite quadrature of the approximate marginal posterior densities. The number of points used for the integration is NPGAUSS, which is specified in the *parinclu* file and it is changable in the parameter file (see NPGAUSS statement).

When the amount of information available for the estimation is limited, unpredictable results (if any) may be obtained, although the mode of the distribution may have been successfully computed.

12.25.1 Model types It is particularly imporatant to note differences between animal and sire models in terms of modelling with the Survival Kit.

Sire model In case of the *sire model* the animals sire is included to the statistical model as a random variable. If pedigree information are available, these can be included via a pedigree file with PEDIGREE statement of Prepare. The user should use *one from the two available options here*. First is to include a relationship matrix based on sire's sires and sire's maternal-grandsires with specifying (for example):

RANDOM sire ESTIMATE MULTINORMAL MGS_RULES <*parameters here*>;

or to use a relationship matrix based on sire and dam of the sire specifying:

RANDOM sire ESTIMATE MULTINORMAL USUAL_RULES <*parameters here*>;

Animal model In case of an animal model the identification variable should be specifies in *both* IDREC and CLASS statements of Prepare. The users are advised not to use ID as a variable name for the animal identification number, because this is used by the Survival Kit elsewhere. If pedigree information are available, this can be included via a pedigree file with PEDIGREE statement of Prepare. Again the same *two options* are here as before, and the user should choose *one of them.*
If the relationship matrix based on animal's sire and animal's maternal-grandsire is available, estimares for the random effect of the animal could be found (for example):

RANDOM animal ESTIMATE MULTINORMAL MGS_RULES <*parameters here*>;

or the user can include the relationship matrix based on sire and dam of the animal:

RANDOM animal ESTIMATE MULTINORMAL USUAL_RULES <*parameters here*>;

12.26 INTEGRATE_OUT Statement

INTEGRATE_OUT < **JOINT_MODE** > < **BACK_SOLVE** > *variable* < **WITHIN** variable2 >;

This option is only available in the WEIBULL program (the COX program prints an error message and stops) and is *not* applicable to a grouped data model. When a random effect is assumed to follow a log-gamma distribution in a Weibull model, it is possible to algebraically integrate it out from the joint posterior density. This technique decreases the number of parameters to estimate (sometimes drastically). Algebraic integration is equivalent here to absorption of a group of equations in systems of linear equations. The consequences are similar: a smaller system (but usually much less sparse) and no direct availability of the estimates of the effects integrated out (other than the gamma parameter)

The INTEGRATE_OUT statement can be used alone, i.e., only followed by the name of the random variable to integrate out (must appear in the RANDOM statement too). Then, the gamma parameter of the log-gamma distribution is assumed to be known. For example,
MODEL *effect1 effect2 effect3*;
RANDOM LOGGAMMA *effect1* 10.0;
INTEGRATE_OUT *effect1*;
will lead to the integration of the log-gamma distributed random variable *effect1* and the estimation of effects *effect2* and *effect3* and of the Weibull parameters, assuming a value of 10.0 for the gamma parameter of *effect1.*

If the INTEGRATE_OUT statement is used with the JOINT_MODE option, (as INTEGRATE_OUT JOINT_MODE *effect1*; in the example above) then the gamma parameter is estimated jointly with the other effects as the mode of the marginal posterior distribution of the gamma parameter, *effect2, effect3* and the Weibull parameters. The value 10.0 specified in the RANDOM statement is just used as a starting value in the optimisation subroutine.
The advantage of this approach is that it performs *exactly* the marginalisation of the random effect, which otherwise is done only approximately via Laplacian integration (Ducrocq and Casella, 1996). Note, however, that the fixed and other random effects as well as the Weibull parameters still have to be integrated out if one wants the *full* marginal posterior density for the gamma parameter of the log-gamma distribution).
The combined use of RANDOM ESTIMATE ... and INTEGRATE_OUT JOINT_MODE ... offers an easy solution to the joint estimation of the distribution parameters of two random variables.

Note: In order to perform the algebraic integration, the recoded data file *file 1* **must** be sorted by increasing levels of the random effect to integrate out.

For very large datasets, this sorting according to the levels of the random effect to integrate out may be cumbersome, in particular because it is not compatible with the use of the BLOCKED_UNFORMATTED and COMPRESSED options. An alternative approach which avoids this sorting using the 'WITHIN' statement has been developed when the effect to integrate out has a hierachical structure. This will be better illustrated through the example of a random herd-year(-season) interaction effect.

The 'hard-work' approach is to have the *herd* and *year* effects included in the original data file. The data preparation data file (for PREPARE) includes the following statements:

INPUT ... *herd* I4 *year* I4 ...;
...
IDREC STORE_PREVIOUS_TIME;
CLASS ... *herd year* ...;
COMBINE ... *hy* = *herd* + *year* ...;
OUTPUT ... *hy* ...;
FORMOUT FREE_FORMAT;

Each possible herd-year combination is recoded separately and the recoded data file has to be sorted by increasing *hy* levels. The use of IDREC STORE_PREVIOUS_TIME is necessary because the effect *hy* to integrate out is a time-dependent covariate (see the description of the IDREC STORE_PREVIOUS_TIME statement and the sorting rules). The need to sort also forces the use of a free format for the output file. Then, in WEIBULL,

the integration of a random herd-year season effect implies the following statements (assuming a log-gamma distribution with initial parameter 10.0):

MODEL ... *hy* ...;
RANDOM *hy* LOGGAMMA 10.0 ...;
INTEGRATE_OUT JOINT_MODE *hy*;

The alternative (much easier) way simply requires that the initial data set be sorted by *herd* (and by *herd* only). There is no need to recode each herd-year interaction separately, no need to specify STORE_PREVIOUS_TIME, no need to restrict the output format to free format, no need to sort the recoded file. The data preparation parameter file looks like (for example):

INPUT ... *herd* I4 *year* I4 ...;
...
IDREC *animal_id*;
CLASS ... *herd year* ...;
OUTPUT ... *herd year* ...;
FORMOUT BLOCKED_UNFORMATTED;
...

and the parameter file "2" for WEIBULL:

MODEL ... *year* ...;
RANDOM *year* LOGGAMMA 10.0 ...;
INTEGRATE_OUT JOINT_MODE *year* WITHIN *herd*;

The output of these two approaches would be exactly the same.

12.27 ANIMAL_SOLUTION Statement

animal_solution *filename*;

The *animal_solution* keyword computes and writes out the results of an approximate animal model, as defined in (Ducrocq, 2001). In case of a random effect is integrated out in ithe model via *integrate_out* statement, the usage of *back_solve* statement is needed.

12.28 CORRELATION Statement

CORRELATION *variable variable* <**ESTIMATE** > *value*;

The *correlation* keyword is used to specify the variables for the correlated random effects and the value of the correlation coefficient. The variables specified here should be in the RANDOM statement. It is possible to include the relationship matrix for both correlated random effects, but in this case the CORRELATION keyword of PREPARE should be used as well (check the description there).

12.29 TEST Statement

TEST <*type*<*s*> *of test*>;

Hypothesis testing is generally performed via likelihood ratio tests. The following types of tests may be requested by the TEST statement:
(*GENERAL*: test of the full model *vs* the model with no covariate. This is the default option and will be used even without request.)
SEQUENTIAL : test of the effects included in the model in sequential order (i.e. depending on the sequence in the MODEL statement).
LAST: likelihood ratio test comparing the full model with models excluding one effect at a time. This is done for each effect separately.
EFFECT list of variables: the same type of test as with LAST is performed, but only for effects stated in the list.

SEQUENTIAL and *LAST* may be stated together. *EFFECT* may be stated either alone, with *SEQUENTIAL* or with *LAST*. When *EFFECT* is used with *LAST*, it is redundant. If it is used with *SEQUENTIAL*, the sequential inclusion of the effects in the model one at a time will start with the first effect appearing in the MODEL statement which is stated in the list following *EFFECT*. For example,
MODEL var1 var2 var3 var4 var5 var6;
TEST SEQUENTIAL EFFECT var4 var5 var6;
will lead to likelihood ratio tests corresponding to the sequential introduction of variables var4, var5, var6.

12.30 STD_ERROR Statement

Asymptotic standard errors of estimates may be requested by the following statement:

STD_ERROR;

Be cautious with the use of the STD_ERROR statement with large models as the Hessian matrix has to be calculated and stored to calculate standard deviations of estimates. In the case of the COX program or the WEIBULL program with the particular DENSE_HESSIAN statement, the full (square) Hessian matrix is stored. Its dimension is

set by default to a value equal to NDIMAX which is defined in the *parinclu* file or with *NDIMAX* statement in *weibull.txt* or *cox.txt*. This may become limiting with respect to storage capacity. If the Hessian matrix is not needed, it might be skipped by using the NO_HESSIAN statement.

In the case of the WEIBULL program, the Hessian matrix is stored in sparse form (unless the DENSE_HESSIAN statement is specified) and the vector space required in order to do so is specified by the parameter NZE_HESS in the *parinclu* file or with the NZE_HESS statement.

12.31 DENSE_HESSIAN and FORCE_POSITIVE Statements

DENSE_HESSIAN;

The full storage of the square Hessian matrix is the only option when the COX program is used (the DENSE_HESSIAN statement is not required). The default option for the WEIBULL program is the sparse storage of the Hessian, unless the DENSE_HESSIAN statement is used. In cases where the memory is not limiting or when a log-gamma random effect is algebraically integrated out (leading to smaller but denser Hessian), the DENSE_HESSIAN option may save substantial computing time.

FORCE_POSITIVE;

In case the DENSE_HESSIAN statement is used, the "matrix is not positive definite" message may appear for certain databases. To bypass this problem, the FORCE_POSITIVE statement should be used.

12.32 BASELINE Statement

BASELINE;

The statement specifies that the baseline hazard, the baseline cumulative hazard and the baseline survivor functions should be computed and printed, in the COX program. These values are computed at each distinct value of the TIME variable. In stratified analyses (use of the STRATA statement), separate baseline functions are calculated for strata. With the WEIBULL program, the computation of the baseline Weibull parameters is always implicit: the BASELINE statement is ignored.

12.33 KAPLAN Statement

KAPLAN;

The statement specifies that the product-limit (=Kaplan-Meier) estimate of the survivor function should be computed and printed. The Kaplan-Meier estimator is a non-parametric population estimator of the survivor function that does not take into account any of the effects stated in the MODEL statement. The KAPLAN statement is permitted only with the COX program (it is ignored otherwise).

12.34 SURVIVAL Statement

SURVIVAL *file7 file8 option<s>*;

The SURVIVAL statement may be used to get information about the survivor function of individuals with specific covariate values. It is very useful to relate the estimated regression coefficients to a more conventional scale like median survival time or probability of survival to a certain age.

file7 is the name of the recoded file produced by program PREPARE on request in the FUTURE statement of PREPARE (called *file6* there).

file8 holds the results invoked by the use of the SURVIVAL statement.

The following *options* may be used:

QUANTILES value<s>
With this option, times at which specific values (quantiles) of the survivor curve are reached are calculated and printed for each individual specified in *file7*. The *values* must lie between 0 and 100 and will be divided by 100 to produce the value of the survivor function (e.g., with a value of 25, S(t)= 0.25). The median survival time can therefore be requested by the option *QUANTILE 50*.

ALL_TIMES
The estimate of the survivor curve will be computed at all times (= failure time or change in time-dependent covariates) indicated for each individual in *file7*.

EQUAL_SPACE interval limit
The estimate of the survivor curve will be computed at equally spaced times for each individual. two figures are needed: time interval and time limit. For example, SURVIVAL EQUAL_SPACE 5 100 will compute the value of the survivor curve at time 5, 10, 15, ... 95, 100.

SPECIFIC time<s>
The estimate of the survivor curve will be computed at the specified times indicated for

each individual.

Note: The statements SURVIVAL and RESIDUALS are mutually exclusive and should not appear in the same parameter file.

12.35 RESIDUALS Statement

RESIDUALS *file7 file8* ;

The use of the RESIDUALS statement is (so far) limited to the COX program. It specifies that the generalized residuals (Cox and Snell, 1966) of the records in *file 7* should be computed and printed in the output file *file 8*. If all generalized residuals are requested (general case), *file 7* is the **unsorted** recoded data file (direct output of PREPARE). If the option STORAGE ON_DISK is used, the output will also include the martingale residuals and the deviance residuals (Klein and Moeschberger, 1997).

Note: The statements RESIDUALS and SURVIVAL are mutually exclusive and should not appear in the same parameter file.

12.36 CONSTRAINTS Statement

CONSTRAINTS *option*;

For models not of full rank (all models with fixed discrete class covariates), constraints can be imposed upon the effects to be estimated to get a set of meaningful, easy to interpret estimable effects. The program offers different *options* for setting those constraints:

LARGEST: the program will set to zero the level of each discrete covariate with the largest number of uncensored failures. This is the default procedure of handling constraints.

FIND: the constraints are found by the program. This will guarantee the correct number of constraints to produce a full rank Hessian matrix, so that the tests for the full model as well as for individual effects are based on the correct degrees of freedom (however, the constraints found do not guarantee an easy interpretation of the results...) Linear dependencies in the Hessian matrix are found by performing a Cholesky decomposition. For very large problems, the storage of this matrix may be a limiting factor (as for the computation of the standard error; see the STD_ERROR statement).

NONE: no constraints specified (but implicit ones will appear if standard errors are to be computed, i.e. a generalized inverse of the Hessian matrix will be used if needed).

IMPOSE <CHECK> variable rec_level variable rec_level ...: the constraints are supplied by the user. *variable* defines the effect for which a constraint is to be set (must be a CLASS variable that has been stated in the MODEL statement). *rec_level* is the **recoded** value of the effect that should be constraint.

Note: You have to look into the file that holds original and recoded levels of the CLASS effects (*file3* defined in the PREPARE parameter file) to look up what is the recoded value of the level you actually want to put the constraint on. To do so, look for the name of the effect of interest. Columns 4 (when there is no interaction), 4 and 5 (when two effects were combined into an interaction) or 4, 5, and 6 (when three effects were combined) display the original code. The last column defines the new code. By use of the option *check* the program will check whether the constraints specified by the user make sense.

12.37 CONVERGENCE_CRITERION Statement

CONVERGENCE_CRITERION real_number;

The *real_number* in the CONVERGENCE_CRITERION statement provides the termination point for the optimisation routine used in the likelihood maximisation. Its default value is parameter EPS_BFDEF set in *parinclu* (usually, 1.D-8). Alternative values may be invoked either by using this statement or by changing the value of EPS_BFDEF in *parinclu*.
Warning: an unnecessary strict convergence criterion set by the user will lead to a warning message by the optimisation routine telling (amongst others) "LINE SEARCH FAILED", although the results are printed and usually valid.

12.38 STORAGE Statement

STORAGE option;

The data file *file3* may either be read in and held in core (option *IN_CORE* or while being read for the first time, be written out on a temporary file in binary mode on disk (option *ON_DISK*) for faster access in subsequent readings of the file in cases where the core memory available is not big enough to hold the whole data file. The largest number of elementary records possibly stored in core at the same time is specified by the parameter NRECMAX changable in the parameter text file (see NRECMAX statement).

12.39 STORE_SOLUTIONS and READ_OLD_SOLUTIONS Statement

STORE_SOLUTIONS;
READ_OLD_SOLUTIONS;

These two statements are only useful for very large applications (applications running for hours and days). If the STORE_SOLUTIONS statement has been used to write the

final solutions to *file6* of the FILES statement, the READ_OLD_SOLUTIONS statement may be used in the next run to retrieve those solutions from *file5* of the FILES statement. The model does **not** need to be the same in both runs. This is mainly useful when one is fitting a complex model from a simpler one or in connection with the ESTIMATE option of the RANDOM statement. When the results of the estimation indicate that the true value of the distributional parameter for the log-gamma or Normal distributions lies outside the prespecified interval, the values of the interval may be changed and using READ_OLD_SOLUTIONS the optimisation may proceed from the point reached already. For storage of solutions not only at the end of the optimisation procedure but after each iteration, an alternative approach has to be taken. In the *parinclu* file, a character parameter BACKUP is defined. This parameter may be used to define the name of a file where solutions will be stored each iteration. If the solutions are needed (e.g. in a restart after a system shut-down), you only have to give the BACKUP filename as *file5* (they have exactly the same structure) in the FILES statement and include the READ_OLD_SOLUTIONS statement in your parameter file.

Chapter 3

A small example

The following (completely artificial) example will be used to illustrate how to analyze survival data using the Survival Kit.

1. Inital data file *small.dat*:
   Consider the data file:

```
 1   5 1 1 0 1 5 4 10
 2 10 1 1 0 1 5 6 10
 3 11 1 1 0 1 5 6 10
 4 11 0 1 0 0
 5 15 1 2 0 0
 6 15 1 2 0 1 5 6 10
 7 17 1 1 0 0
 8 19 1 1 0 0
 9 21 1 2 0 0
10 21 0 2 0 1 5 15 10
11 28 1 2 0 1 5 15 10
```

   As specified in the INPUT line of the data preparation parameter file (*prepare.txt*, see below), the first variable for each line corresponds to the animal identification number (not necessarily from 1 to n as it is the case here). The second represents failure or censoring time. The third is the censoring code (here, 0=censored). Column 4 refers to the level of sex effect and column 5 to the level of treatment effect. It is assumed that treatment does not start at time 0 but later on. Hence, the treatment variable is considered as a time-dependent covariate. This explains why the fifth column is always 0 here: the animal is not treated at t=0. The next column indicated the number of changes in time-dependent covariates. Here, there is only 1 change per animal, at most. When there is a change (column 6 = 1), a triplet follows indicating the column number of the relevant covariate (column 5 = 'treatment'), the time of change (e.g., t=4 for record 1) and the new value of the covariate (10, here): at time 4, the treatment level for animal 1 changes from the value 0 to the value 10.

2. Data preparation parameter file *prepare.txt*
   This is the file that will be used as input for the PREPARE program.

```
   /* FILE NAMES */
FILES  small.dat small2.dat codelist.txt  varlist.txt ;

   /* VARIABLE NAMES */
INPUT  id I4 longev I4 code  I4 sex I4 treat I4;

   /* TIME VARIABLE */
TIME longev ;

   /* CENSORING CODE */
CENSCODE code 0;

   /* TIME DEPENDENT COVARIATES   */
TIMEDEP treat I4;

   /* IDENTIFICATION VARIABLE FOR EACH RECORD */
IDREC id;

   /* DISCRETE VARIABLES */
CLASS  treat sex;

   /* DEFINITION OF AN INTERACTION */
COMBINE s_by_t= sex + treat;

   /* LIST OF RECODED VARIABLES */
OUTPUT longev id code treat sex s_by_t;

   /* FORMAT OF THE OUTPUT FILE */
FORMOUT FREE_FORMAT;
```

   The treatment variable is explicitly defined as a time-dependent covariate, with changes indicated as integers (I4), not dates. The COMBINE statement defines a new variable s_by_t, representing the interaction of sex by treatment. The *id* does not appear in the CLASS statement and therefore will not be recoded.

3. Log file from the PREPARE program

   The PREPARE program will search for file name *repare.txt* as the name of the input file. The log file indicates that 17 elementary (recoded) records were created (> 11, because treatment is time-dependent. The number of classes is also indicated for the discrete (class) variables.

```
 Records read from input data file      11
 Elementary records written              17
 Class variable treat    with      2 classes
 Class variable sex      with      2 classes
 Class variable s_by_t   with      4 classes

 REMINDER: File small2.dat      has to be sorted for the use of COX.
 Sorting order:
```

```
Strata variable, ascending (only if strata are used)
Time variable, descending
Censoring variable, ascending
```

4. Recoded data set *small2.dat*

```
4 -1 1 1 2 1 1 2
5 1 1 2 0 1 2 0
6 -1 2 1 2 1 1 2
10 1 2 2 0 1 2 0
6 -1 3 1 2 1 1 2
11 1 3 2 0 1 2 0
11 0 4 1 0 1 1 0
15 1 5 1 0 2 3 0
6 -1 6 1 2 2 3 4
15 1 6 2 0 2 4 0
17 1 7 1 0 1 1 0
19 1 8 1 0 1 1 0
21 1 9 1 0 2 3 0
15 -1 10 1 2 2 3 4
21 0 10 2 0 2 4 0
15 -1 11 1 2 2 3 4
28 1 11 2 0 2 4 0
```

The structure of this file is given in file *varlist.txt* which is an output file of the PREPARE program and an input files for the programs COX and WEIBULL (see below). The first column is the time of failure, of censoring, of change in at least one time-dependent covariate, or of truncature (for truncated records). These four categories are indicated by a code in column 2 with value 1, 0, -1 or -2 respectively. The third column indicates the id number. The remaining ones correspond to the covariates used, individually recoded from 1 to the total number of levels, for discrete covariates. When a covariate is time-dependent, two columns are used, one corresponding to the value of the covariate *before* the change and the next one for *after* the change. For example, for animal 1 (first elementary record), at time 4, the level of the treatment effect changes (code= -1) from recoded value 1 to recoded value 2 (columns 4 and 5) while the level for the s_by_t effect changes also from 1 to 2 (columns 7 and 8). At time 5 (second elementary record), animal 1 fails (code= 1).

5. parameter file "1" (for COX and WEIBULL) *varlist.txt*
   This file is created by the PREPARE program. It explains the structure of the *small2.dat* recoded file.

```
TIME    1;
CODE    2;
ID      3;
COVARIATE
/* class variables */
treat        2   4   5
sex          2   6   6
s_by_t       4   7   8
;
 FREE_FORMAT;
```

For example, there are 2 levels of treatment stored in columns 4 (before any change) and 5 (after the change). The sex effect is time-independent: only one column (6) is needed.

6. file *codelist.txt*

   This file is also an output file from the PREPARE program. It gives the correspondence between old and new codes.

```
treat       2 1          0          0          0       11       1
treat       2 1         10          0          0        6       2
sex         2 1          1          0          0        6       1
sex         2 1          2          0          0        5       2
s_by_t      4 2          1          0          0        6       1
s_by_t      4 2          1         10          0        3       2
s_by_t      4 2          2          0          0        5       3
s_by_t      4 2          2         10          0        3       4
```

   For each level of a discrete covariate, the name of the covariate (e.g., s_by_t for the last line) is followed by the total number of levels of this effect (4), the number of columns involved in its description (2, because it is a 2-term interaction) and the corresponding original codes (sex 2 x treatment 10). Up to 3 effects can be combined into an interaction. Column 7 indicates the number of elementary records found with this particular level of the covariate (3) and column 8 (last one) gives the new code (4).

7. recoded data set for COX *small2s.dat*

   This is the recoded data set *small2.dat* after sorting by decreasing time change (column 1) and increasing codes (column 2) (no stratification assumed).

```
28 1 11 2 0 2 4 0
21 0 10 2 0 2 4 0
21 1 9 1 0 2 3 0
19 1 8 1 0 1 1 0
17 1 7 1 0 1 1 0
15 -1 10 1 2 2 3 4
15 -1 11 1 2 2 3 4
15 1 5 1 0 2 3 0
15 1 6 2 0 2 4 0
11 0 4 1 0 1 1 0
11 1 3 2 0 1 2 0
10 1 2 2 0 1 2 0
6 -1 2 1 2 1 1 2
6 -1 3 1 2 1 1 2
6 -1 6 1 2 2 3 4
5 1 1 2 0 1 2 0
4 -1 1 1 2 1 1 2
```

8. parameter file "2" for COX - *cox.txt*

```
/* FILES */
FILES small2s.dat codelist.txt small.rco;

/* TITLE */
TITLE SMALL EXAMPLE - COX MODEL

/* MODEL */
MODEL sex treat ;

/* TEST */
TEST SEQUENTIAL LAST ;

/* STANDARD ERROR */
STD_ERROR;

/* OTHER COMPUTATIONS */
BASELINE;
RESIDUAL small2.dat small.res;
```

The input data file is the sorted, recoded data file *small2s.dat*. The model simply includes sex treatment effects (no interaction here). The results will be stored in *small.rco*. Likelihood ratio tests will be performed, comparing models with no covariates, with sex only and sex and treatment (TEST SEQUENTIAL) and comparing the full model with restricted model (excluding sex = with treatment only; and excluding treatment = with sex only; TEST LAST). After computing the regression coefficients using Cox's partial likelihood, the baseline will be computed (BASELINE) and generalized residuals will be calculated for all records in the **unsorted** file *small2.dat* and they will be stored in the *small.res* file.

9. log file from COX

   This file appears on the screen terminal if the answer 'NONE' or ' ' is given when a name for the detailed output file is requested.

   The file starts with the names of the 2 input parameter files and other file names, followed by a brief list of characteristics of the data set, of the model and some elementary statistics.

```
parameter file: varlist.txt
parameter file: cox.txt

      ****************************************

         SMALL EXAMPLE - COX MODEL

      ****************************************

 FILES USED :
    RECODED DATASET    :  small2s.dat
    NEW/OLD CODES      :  codelist.txt
    OUTPUT             :  small.rco
```

```
RESIDUALS :
   RECODED DATASET    :  small2.dat
   OUTPUT             :  small.res

     ****************************************

NO STRATIFICATION

FAILURE TIME READ IN COLUMN :                          1
CENSORING CODE READ IN COLUMN :                        2

IDENTIFICATION NUMBER READ IN COLUMN :                 3
TOTAL NUMBER OF COVARIATES =                           3
CONTINUOUS COVARIATES INCLUDED IN THE ANALYSIS =       0
DISCRETE COVARIATES INCLUDED IN THE ANALYSIS =         2
COVARIATES READ BUT NOT INCLUDED IN THE ANALYSIS =     1
POWERS OF CONTINUOUS COVARIATES                  =     0

     ************** COX MODEL ****************

COVARIATE            CHARACTERISTICS

1 sex     : DISCRETE TIME-INDEPENDENT READ IN COLUMN   6
2 treat   : DISCRETE TIME-DEPENDENT READ IN COLUMNS    4 (BEFORE T) AND   5 (AFTER T)

CONVERGENCE CRITERION USED =   .10000D-07 (= DEFAULT VALUE)

     ********** SIMPLE STATISTICS ************

TOTAL NUMBER OF STRATA =                    1
TOTAL NUMBER OF ELEMENTARY RECORDS =       17

RIGHT CENSORED RECORDS :         2 (  18.182%)
MINIMUM CENSORING TIME :        11
MAXIMUM CENSORING TIME :        21
AVERAGE CENSORING TIME :        16.000

UNCENSORED RECORDS :             9
MINIMUM FAILURE TIME :           5
MAXIMUM FAILURE TIME :          28
AVERAGE FAILURE TIME :          15.667

EFFECT : sex      MIN =      1       MAX =      2
EFFECT : treat    MIN =      1       MAX =      2
```

Then the first few recoded elementary records are printed, as well as the constraints used (here the COX program chose to set to zero level 1 of sex and level 10 of treatment, because they are the levels with the largest number of uncensored observations).

```
 DATA :
1  |  28 1 0  |  2 0
2  |  21 0 0  |  2 0
3  |  21 1 0  |  1 0
4  |  19 1 0  |  1 0
5  |  17 1 0  |  1 0
6  |  15 -1 0  |  1 2
7  |  15 -1 0  |  1 2
```

```
 8  |  15 1 0  |  1 0
 9  |  15 1 0  |  2 0

  NUMBER OF ELEMENTARY RECORDS KEPT =       17

       ******* CONSTRAINTS ******

  THE SOLUTION FOR THE FOLLOWING (RECODED) LEVELS IS SET TO 0:
  (= LEVEL WITH LARGEST NUMBER OF UNCENSORED FAILURES FOR EACH EFFECT)

 (WARNING : THE VALIDITY OF THESE CONSTRAINTS IS NOT CHECKED.
  IN CASE THEY ARE MORE DEPENDENCIES,
  THE DEGREES OF FREEDOM IN THE TEST(S) BELOW ARE INCORRECT)

   EFFECT sex      LEVEL        1
   EFFECT treat    LEVEL       10
```

Then limited storage quasi-Newton (Liu and Nocedal, 1989) minimization of FUNCT= minus the likelihood function starts for each (sub-)model. GNORM is the norm of vector of first derivative of FUNCT. CONV. CRITERION is equal to:

$$GNORM/max(1, FUNCT).$$

Let CC be the *requested* convergence criterion. The program stops the minimization process in the following situations:

- CONV. CRITERION is less than 0.01 * CC
- CONV. CRITERION is less than CC *and* the total number of FUNCT evaluations is at least equal to the number of corrections (15, below)
- After many attempts (usually 20), the program fails to find a smaller value of FUNCT. This is **very often** due to too strict a convergence criterion. A careful examination of the variation of the FUNCT value during the last iterations will help to reveal that. If FUNCT is still varying substantially, the program failed to find a proper minimum (often because the model is incorrect or inconsistent). If FUNCT is stable, one can safely assume that the results are correct and that a minimum has been found.

```
  DATA :
************************************************
  N=    4   NUMBER OF CORRECTIONS=15
       INITIAL VALUES
 F=  1.488D+01   GNORM=  2.289D+00
************************************************

   I   NFN      FUNCT          GNORM      STEPLENGTH CONV. CRITERION

   1    2    1.341197562D+01   6.570D-01   4.369D-01   4.898D-02
   2    3    1.327040139D+01   5.823D-02   1.000D+00   3.128D-03
   3    4    1.326918347D+01   4.121D-03   1.000D+00   2.154D-04
   4    5    1.326917728D+01   3.349D-05   1.000D+00   1.747D-06
   5    6    1.326917728D+01   1.974D-08   1.000D+00   1.030D-09
```

```
   6    7    1.326917728D+01   9.451D-14   1.000D+00   4.930D-15

 THE MINIMIZATION TERMINATED WITHOUT DETECTING ERRORS.

  CONVERGENCE AFTER     6 EVALUATIONS OF FUNCTION FCOX IN        .00 SECONDS
************************************************
  N=    4   NUMBER OF CORRECTIONS=15
       INITIAL VALUES
 F=  1.488D+01   GNORM=  1.534D+00
************************************************

   I   NFN      FUNCT          GNORM      STEPLENGTH CONV. CRITERION

   1    2    1.411382471D+01   3.266D-02   6.517D-01   2.314D-03
   2    3    1.411350211D+01   1.696D-03   1.000D+00   1.202D-04
   3    4    1.411350124D+01   2.899D-07   1.000D+00   2.054D-08
   4    5    1.411350124D+01   2.847D-12   1.000D+00   2.017D-13

 THE MINIMIZATION TERMINATED WITHOUT DETECTING ERRORS.

  CONVERGENCE AFTER     4 EVALUATIONS OF FUNCTION FCOX IN        .00 SECONDS
************************************************
  N=    4   NUMBER OF CORRECTIONS=15
       INITIAL VALUES
 F=  1.411D+01   GNORM=  2.542D+00
************************************************

   I   NFN      FUNCT          GNORM      STEPLENGTH CONV. CRITERION

   1    2    1.230257068D+01   1.122D+00   3.934D-01   6.511D-02
   2    3    1.172877696D+01   2.155D-01   1.000D+00   8.071D-03
   3    4    1.170091695D+01   6.500D-02   1.000D+00   2.212D-03
   4    5    1.169961281D+01   2.672D-02   1.000D+00   8.924D-04
   5    6    1.169943368D+01   3.665D-04   1.000D+00   1.226D-05
   6    7    1.169943364D+01   9.265D-06   1.000D+00   3.099D-07
   7    8    1.169943364D+01   1.296D-07   1.000D+00   4.334D-09
   8    9    1.169943364D+01   2.662D-10   1.000D+00   8.905D-12

 THE MINIMIZATION TERMINATED WITHOUT DETECTING ERRORS.

  CONVERGENCE AFTER     8 EVALUATIONS OF FUNCTION FCOX IN        .01 SECONDS
  VALUE OF F =   11.6994336385
  TIME FOR THE COMPUTATION OF THE HESSIAN =      .00 SECONDS
  TIME FOR NUMERICAL FACTORIZATION =      .00 SECONDS

  TOTAL TIME =        .05 SECONDS


       ******* BYE - BYE ********
```

10. Results from COX
    The beginning is exactly the same as the log file (files names, model characteristics, simple statistics, constraints).

```
       ****************************************
```

```
        SMALL EXAMPLE - COX MODEL

     ****************************************

 FILES USED :
    RECODED DATASET    :  small2s.dat
    NEW/OLD CODES      :  codelist.txt
    OUTPUT             :  small.rco

 RESIDUALS :
    RECODED DATASET    :  small2.dat
    OUTPUT             :  small.res

     ****************************************

 NO STRATIFICATION

 FAILURE TIME READ IN COLUMN :                          1
 CENSORING CODE READ IN COLUMN :                        2

 IDENTIFICATION NUMBER READ IN COLUMN :                 3
 TOTAL NUMBER OF COVARIATES =                           3
 CONTINUOUS COVARIATES INCLUDED IN THE ANALYSIS =       0
 DISCRETE COVARIATES INCLUDED IN THE ANALYSIS =         2
 COVARIATES READ BUT NOT INCLUDED IN THE ANALYSIS =     1
 POWERS OF CONTINUOUS COVARIATES                  =     0

     ************** COX MODEL ****************

 COVARIATE            CHARACTERISTICS

 1 sex     : DISCRETE TIME-INDEPENDENT READ IN COLUMN   6
 2 treat   : DISCRETE TIME-DEPENDENT READ IN COLUMNS    4 (BEFORE T) AND   5 (AFTER T)

     ********** SIMPLE STATISTICS ************

 TOTAL NUMBER OF STRATA =                    1
 TOTAL NUMBER OF ELEMENTARY RECORDS =       17

 RIGHT CENSORED RECORDS :        2 (  18.182%)
 MINIMUM CENSORING TIME :       11
 MAXIMUM CENSORING TIME :       21
 AVERAGE CENSORING TIME :       16.000

 UNCENSORED RECORDS :            9
 MINIMUM FAILURE TIME :          5
 MAXIMUM FAILURE TIME :         28
 AVERAGE FAILURE TIME :         15.667

 EFFECT : sex      MIN =      1       MAX =      2
 EFFECT : treat    MIN =      1       MAX =      2


 NUMBER OF ELEMENTARY RECORDS KEPT =       17

     ******* CONSTRAINTS ******

 THE SOLUTION FOR THE FOLLOWING (RECODED) LEVELS IS SET TO 0:
 (= LEVEL WITH LARGEST NUMBER OF UNCENSORED FAILURES FOR EACH EFFECT)

(WARNING : THE VALIDITY OF THESE CONSTRAINTS IS NOT CHECKED.
 IN CASE THEY ARE MORE DEPENDENCIES,
```

```
THE DEGREES OF FREEDOM IN THE TEST(S) BELOW ARE INCORRECT)

 EFFECT sex      LEVEL        1
 EFFECT treat    LEVEL       10

CONVERGENCE AFTER     6 EVALUATIONS OF FUNCTION FCOX IN        .00 SECONDS

CONVERGENCE AFTER     4 EVALUATIONS OF FUNCTION FCOX IN        .00 SECONDS

CONVERGENCE AFTER     8 EVALUATIONS OF FUNCTION FCOX IN        .01 SECONDS
```

The first part of the results includes all the likelihood ratio tests :

```
     *******  RESULTS  ********

-2 LOG LIKELIHOOD                            23.39886728
STANDARDIZED NORM OF GRAD(-2 LOG L) =          .00000

     *******  TESTS    ********

LIKELIHOOD RATIO TEST FOR H0: BETA = 0
MODEL CHI2 =    6.368872     WITH       2 DF   (PROB >CHI2 =     .0414)
R2 OF MADDALA (=MEASURE OF EXPLAINED VARIATION) =   .4395

LIKELIHOOD RATIO TESTS : SEQUENTIAL
VARIABLE    TOTAL        -2 LOG LIK     CHI2     DELTA    PROB   R2 OF
  Z          DF          INCLUDING Z               DF     >CHI2  MADDALA

 NO COVARIATE 0         29.76773961
 sex          1         26.53835455     3.2294       1   .0723   .2544
 treat        2         23.39886728     3.1395       1   .0764   .4395

LIKELIHOOD RATIO TESTS : LAST
VARIABLE    TOTAL        -2 LOG LIK     CHI2     DELTA    PROB   R2 OF
  Z         DF          EXCLUDING Z               DF     >CHI2  MADDALA

 sex          1         28.22700247     4.8281       1   .0280   .1307
 treat        1         26.53835455     3.1395       1   .0764   .2544
```

For example, when the treatment effect is added to a model with sex as only covariate, the full model includes *sex + treatment* while the restricted model includes *sex* only and the corresponding likelihood ratio statistic has a value of 3.1395. This statistic follows a chi-square distribution with 1 degree of freedom under H0 (treatment effects=0) with a corresponding p-value of 0.07. The R2 OF MADDALA is a measure of proportion of explained variation by the model (Maddala, 1983, quoted by Schemper, 1992). Its formal definition is:

$$R^2_M = 1 - \left(L_R/L_U\right)^{2/n}$$

where n denotes the total sample size and L_R and L_U represent the restricted and unrestricted (full model) maximum likelihoods, respectively. Under moderate censoring, R^2_M can be grossly interpreted as the usual R^2 for linear models.

Then the estimates of the regression coefficients are presented, followed by their asymptotic standard error, a chi-square statistic (=square of estimate/standard error) for a Wald test of each particular regression coefficient β_i (which tests $\beta_i = 0$), with its associated p-value. The risk ratio is the exponential of the estimate of the regression coefficient. The last column indicates the number of uncensored records for each level (it is related to the standard error of the estimate).

```
      ******* SOLUTIONS ********

COVARIATE :                ESTIMATE STANDARD    CHI2    PROB   RISK UNCENSORED
                                      ERROR            >CHI2   RATIO FAILURES

 1   sex      (DISCRETE)
      1                      .0000      *        *       *      1.000       5
      2                    -2.0251   1.0316     3.85   .0496     .132       4

 2   treat    (DISCRETE)
      0                    -1.5583    .9027     2.98   .0843     .210       4
     10                      .0000      *        *       *      1.000       5
```

For example, females (sex = 2) are at a 1./.132 = 7.577 lower risk of dying than males (sex = 1).

For ease of interpretation, it is a good idea to *impose* constraints to force the computation of specific contrasts. For example, one may be interested in evaluating the increase in risk of males compared to females and not treated compared to treated, by choosing non treated females as a reference. This is done by using the CONSTRAINTS IMPOSE ... statement in the parameter file *cox.txt*. To define the proper constraints, one has to look in the *codelist.txt* file: sex 2 (old code) is recoded as 2 (new code) and treatment 0 (old code) is recoded as 1 (new code). If we add:

CONSTRAINTS IMPOSE sex 2 treat 1;

to *cox.txt* and running COX, we now get:

```
      ******* SOLUTIONS ********

COVARIATE :                ESTIMATE STANDARD    CHI2    PROB   RISK UNCENSORED
                                      ERROR            >CHI2   RATIO FAILURES

 1   sex      (DISCRETE)
      1                     2.0251   1.0316     3.85   .0496    7.577       5
      2                      .0000      *        *       *      1.000       4

 2   treat    (DISCRETE)
      0                      .0000      *        *       *      1.000       4
     10                     1.5583    .9027     2.98   .0843    4.751       5
```

Now, it directly appears that males are 7.577 times more likely to fail than females and that treated animals are at a 4.751 times higher risk than non treated ones. The contrasts between males and females or between treated and non treated animals remain the same. The baseline however is changed. The one reported below corresponds to the situation without explicit constraints.

```
      ******* BASELINE  ********

   CUMULATIVE HAZARD FUNCTION AND SURVIVOR FUNCTION FOR STRATUM   1
      TIME     CUMULATIVE       SURVIVOR
                  HAZARD        FUNCTION

         5         .45633         .63361
        10         .80420         .44745
        11        1.33765         .26246
        15        4.34920         .01292
        17        5.75223         .00318
        19        7.74329         .00043
        21       11.17092         .00001
        28       18.74770         .00000

  TOTAL TIME =        .05 SECONDS
```

11. Residuals for the Cox model

    The generalized residuals of Cox and Snell (1966) are calculated for all records of the unsorted *small2.dat* file. If the model is correct, they are distributed according to a censored exponential distribution with parameter 1. This can be checked globally by plotting the cumulated hazard distribution (- log S(RES)) or the expected order statistics of the unit exponential distribution against the value of RESIDUALS. This should result in a straight line going through the origin and with slope 1. Note that expected order statistics are computed only when STORAGE IN_CORE is used.

```
ANIMALS  TOTAL   FAILURE RESIDUALS   S(RES)  -LOG S(RES)  EXPECTED
AT RISK  FAILED                                          ORDER STAT.
    11      1       1    .1208301   .9090909   .0953102   .0909091
     9      2       1    .3103519   .8080808   .2130932   .2020202
     8      3       1    .4439302   .7070707   .3466246   .3270202
     7      4       1    .4563280   .6060606   .5007753   .4698773
     6      5       1    .5264675   .5050505   .6830968   .6365440
     5      6       1    .9773750   .4040404   .9062404   .8365440
     3      7       1   1.2108388   .2693603  1.3117055  1.1698773
     2      8       1   1.6299547   .1346801  2.0048527  1.6698773
     1      9       1   2.0211752   .0000000        INF  2.6698773
```

It is also possible to get individual residuals, i.e., for each observation, including martingale and deviance residuals by simply specifying STORAGE ON_DISK in the *cox.txt* file.

TIME	ANIMAL	CENSORING CODE	STRATUM	GENERALIZED RESIDUAL	MARTINGALE RESIDUAL	DEVIANCE RESIDUAL
5	1	1	1	.4563280	.5436720	.6940770
10	2	1	1	.4439302	.5560698	.7155672
11	3	1	1	.9773750	.0226250	.0227979
11	4	0	1	.2815734	-.2815734	-.7504310
15	5	1	1	.1208301	.8791699	1.5711144
15	6	1	1	.5264675	.4735325	.5797123
17	7	1	1	1.2108388	-.2108388	-.1976130
19	8	1	1	1.6299547	-.6299547	-.5317941
21	9	1	1	.3103519	.6896481	.9802044
21	10	0	1	1.0211752	-1.0211752	-1.4291083
28	11	1	1	2.0211752	-1.0211752	-.7968640

12. parameter file "2" for Weibull

    The following parameter file *weibull.txt* now replaces *cox.txt*:

```
/* FILES */
FILES small2.dat codelist.txt small.rwe   ;

/* TITLE */
TITLE SMALL EXAMPLE - WEIBULL  MODEL ;

/* MODEL */
MODEL sex treat ;

STATISTICS;

/* TEST */
TEST SEQUENTIAL LAST ;

/* STANDARD ERROR */
STD_ERROR;
```

13. results from WEIBULL

    The beginning is similar to the Cox outputs, except that here, raw statistics by level of each discrete factor included in the model are given:

```
     ****************************************

        SMALL EXAMPLE - WEIBULL  MODEL

     ****************************************

FILES USED:
   RECODED DATASET    :  small2.dat
   NEW/OLD CODES      :  codelist.txt
   OUTPUT             :  small.rwe

     ****************************************

FAILURE TIME READ IN COLUMN :                        1
CENSORING CODE READ IN COLUMN :                      2
```

```
IDENTIFICATION NUMBER READ IN COLUMN :                3
TOTAL NUMBER OF COVARIATES =                          3
CONTINUOUS COVARIATES INCLUDED IN THE ANALYSIS =      0
DISCRETE COVARIATES INCLUDED IN THE ANALYSIS =        2
COVARIATES READ BUT NOT INCLUDED IN THE ANALYSIS =    1
POWERS OF CONTINUOUS COVARIATES                  =    0

     ********** MODEL ******************

COVARIATE            CHARACTERISTICS

1 sex     : DISCRETE TIME-INDEPENDENT READ IN COLUMN   6
2 treat   : DISCRETE TIME-DEPENDENT READ IN COLUMNS    4 AND   5

     ********** SIMPLE STATISTICS ************

TOTAL NUMBER OF WEIBULL PARAMETERS TO COMPUTE =    1
TOTAL NUMBER OF RECORDS =                         11
TOTAL NUMBER OF ELEMENTARY RECORDS =              17

RIGHT CENSORED RECORDS :        2 (  18.182%)
MINIMUM CENSORING TIME :       11
MAXIMUM CENSORING TIME :       21
AVERAGE CENSORING TIME :       16.000

UNCENSORED RECORDS :            9
MINIMUM FAILURE TIME :          5
MAXIMUM FAILURE TIME :         28
AVERAGE FAILURE TIME :         15.667

EFFECT : sex      MIN =     1       MAX =      2
EFFECT : treat    MIN =     1       MAX =      2


      ===========
 1 WEIBULL PARAMETER(S) RHO WILL BE ESTIMATED

CONVERGENCE CRITERION USED =   .10000D-07 (= DEFAULT VALUE)

NUMBER OF ELEMENTARY RECORDS KEPT =       17


     ********** STATISTICS /LEVEL ************

     EFFECT           NUMBER   %   OBSERVED  %     AGE AT     TOTAL      %   AVERAGE
                        OF         FAILURES       FAILURE      TIME            TIME
                  INDIVIDUALS

 1   sex
      1                   6  54.55     5  55.56    12.40         73.  42.20   12.17
      2                   5  45.45     4  44.44    19.75        100.  57.80   20.00

 2   treat
      0                  11 100.00     4  44.44    18.00        135.  78.03   12.27
     10                   6  54.55     5  55.56    13.80         38.  21.97    6.33
```

Note that these statistics should be interpreted with caution for time-dependent covariates: for example, here, 11 individuals had a record with level of treatment

effect = 0 (all animals at t=0) while 6 of them also had a record with level of treatment effect = 10...

```
      ******* CONSTRAINTS ******

 THE SOLUTION FOR THE FOLLOWING (RECODED) LEVELS IS SET TO 0:
 (= LEVEL WITH LARGEST NUMBER OF UNCENSORED FAILURES FOR EACH EFFECT)

(WARNING : THE VALIDITY OF THESE CONSTRAINTS IS NOT CHECKED.
 IF THEY ARE MORE DEPENDENCIES, DEGREES OF FREEDOM IN TEST(S) BELOW ARE INCORRECT)

 EFFECT sex      LEVEL        1
 EFFECT treat    LEVEL       10

 CONVERGENCE AFTER    19 EVALUATIONS OF FUNCTION FWEIB IN        .02 SECONDS

 CONVERGENCE AFTER    16 EVALUATIONS OF FUNCTION FWEIB IN        .03 SECONDS

 CONVERGENCE AFTER    17 EVALUATIONS OF FUNCTION FWEIB IN        .02 SECONDS

      *******  RESULTS  ********

 -2 LOG LIKELIHOOD                            55.40054940
 STANDARDIZED NORM OF GRAD(-2 LOG L) =          .00000

      *******  TESTS    ********

 LIKELIHOOD RATIO TEST FOR H0: BETA = 0
 MODEL CHI2 =    6.479457      WITH       4 DF   (PROB >CHI2 =     .1661)
 R2 OF MADDALA (=MEASURE OF EXPLAINED VARIATION) =   .4451

 LIKELIHOOD RATIO TESTS : SEQUENTIAL
 VARIABLE    TOTAL        -2 LOG LIK     CHI2     DELTA    PROB   R2 OF
   Z          DF          INCLUDING Z               DF     >CHI2  MADDALA

  NO COVARIATE 2         61.88000661
  sex          3         57.93613232     3.9439       1   .0470   .3013
  treat        4         55.40054940     2.5356       1   .1113   .4451

 LIKELIHOOD RATIO TESTS : LAST
 VARIABLE    TOTAL        -2 LOG LIK     CHI2     DELTA    PROB   R2 OF
   Z         DF          EXCLUDING Z               DF     >CHI2  MADDALA

  sex          3         61.28410415     5.8836       1   .0153   .0527
  treat        3         57.93613232     2.5356       1   .1113   .3013
```

The main difference in WEIBULL compared to COX is the inclusion of estimates of the Weibull parameter ρ and $\rho \log \lambda$ (called INTERCEPT here). These two parameters fully describe the baseline hazard function and the baseline survivor function $S_0(t) = exp\{-(\lambda t)^\rho\}$.

```
      ******* SOLUTIONS ********


      WEIBULL PARAMETER(S):
```

```
  FOR STRATUM   1 ==>  RHO =  3.38225            STE =     .97667

     INTERCEPT =   -8.07086    STE =    2.65373

COVARIATE :                ESTIMATE STANDARD    CHI2    PROB   RISK UNCENSORED
                                     ERROR            >CHI2   RATIO FAILURES

 1   sex      (DISCRETE)
      1                      .0000      *        *       *      1.000       5
      2                    -2.1895   1.0001     4.79   .0286     .112       4

 2   treat    (DISCRETE)
      0                    -1.2901    .8108     2.53   .1116     .275       4
     10                      .0000      *        *       *      1.000       5

TOTAL TIME =        .10 SECONDS
```

Chapter 4

Analysis of very large data sets

This section includes some advice to efficiently analyze very large data sets.

13 PREPARE program

Usually, computing time and memory requirements related to the PREPARE program are not limiting factors. The only potential problem that may occur is related to the size of the output (recoded) data file, because the use of time-dependent covariates may lead to a much larger number of *elementary records* (corresponding to each time interval for which all time-dependent covariates remain constant) than the actual number of records in the initial data set. In order to limit the size of the recoded data set:

1. in the OUPUT statement, specify only the covariates that you will need later;

2. because (at this stage) the PREPARE program requires all main effects to be included in the OUTPUT statement when an interaction is created with the COMBINE statement, it may be advantageous to recode the interactions beforehand, if the main effects are not to be used later. In contrast, if the interaction effect is going to be treated as a random, log-gamma effect that will be integrated out in a Weibull model, it is possible to avoid the recoding of the interaction in PREPARE if INTEGRATE_OUT ... WITHIN ... is used later on.

Also, unless when an individual random effect is to be fitted later on, the ID number should not be recoded, i.e., it should not appear in the CLASS statement (only in the IDREC and OUTPUT statements).

14 Before running the COX or WEIBULL programs

In order to limit memory requirements, it is essential to accurately specify some parameters in the *parinclu* file or using the appropriate keywords in the parameter files for COX or WEIBULL. In particular NRECMAX (maximum number of elementary records stored in core at one time). This number should be relatively large (in particular, it should be larger or equal to the exact number of elementary records if the statement STORAGE IN_CORE is used). However, for obvious reasons, this number should not exceed the memory capacity.

15 COX and WEIBULL programs

1. If the assumption of a Weibull baseline hazard function is reasonable, it is highly recommended to use a Weibull model instead of a Cox model, in particular when standard errors or distribution parameters of random effects are requested.

2. In order to decrease the size of the vectors of parameters:

   - Use stratification if possible (STRATA statement).
   - With WEIBULL, when a random effect is included in the model whose estimates are not really needed, a loggamma prior should be used if possible, and the effect should be integrated out (INTEGRATE_OUT statement): this is analogous to the absorption of the equations corresponding to the specified effect in a linear system of equations. Note that like for an "absorption", the resulting Hessian matrix is no longer very sparse. Hence, the DENSE_HESSIAN statement should also be used.

3. To limit the number of iterations or function evaluations required:

   - Store solutions at the end of each run (STORE_SOLUTIONS statement) or use backup files (BACKUP parameter in the *parinclu* file). Start the next run from the old solutions (READ_OLD_SOLUTIONS statement).
   - If only a few effects are added to a "well known" model, don't use the time consuming TEST SEQUENTIAL or TEST LAST statements which specify that all the effects in the model should be tested, starting from scratch . Use the TEST <SEQUENTIAL> EFFECTS ... statement instead.
   - Check that the convergence criterion is not too strict, by looking at the evolution of -(Log-likelihood), the function which is mimized to get the parameters estimates (CONVERGENCE statement). Be careful, however: an incorrect concergence criterion may lead to incorrect results.

- With WEIBULL, the use of the RHO_FIXED statement should lead to a faster convergence rate.

4. When estimating the hyperparameter(s) of the distribution(s) of random effects:

   - When a loggamma prior is used with the INTEGRATE_OUT statement, the JOINT_MODE option (which computes the gamma parameter jointly with the other effects) should be prefered (faster convergence).
   - Don't choose huge initial intervals or too strict final tolerances with the ESTIMATE statement.
   - Use the MOMENTS statement only if necessary or at least after a run to find the mode of the marginal posterior distribution of the hyperparameter.

Chapter 5

Database creation tutorial

The Survival Kit program package requires a special database structure in order to correctly read in and process data. This structure is even more complicated if time dependent data are present. These should be given in triplet form.

16 General example

Let's consider an example of one single line in the database, which uses *time dependent* and *time independent* input data. *No other characters than numbers are allowed.* The triplets should not be underlined. In the case below, this was done for demonstration purposes only.

12345 960 1 2007 3 2 4 390 2008 4 710 2009

The explanation for the following line is as follows:

- **12345** - The identification number of the animal. It should consist *only* of numbers, and its length should be in the range of an *integer*. In case of larger values of identification numbers, these should be recoded.
  In the Survival Kit it is stated with the IDREC statement in the parameter file of prepare.
- **960** - Length of productive life in days (time interval between first calving and culling or censoring)
  In the Survival Kit, it is given in the TIME statement in the parameter file of PREPARE.
- **1** - Censoring code, value which denotes censoring. It can be chosen freely by user.

Let's assume that this particular animal is not censored, i.e. it was culled by the breeder (code = 1).
In the Survival Kit the censoring code is given in the CENSCODE statement in the parameter file of PREPARE.

- **2007** - Year of calving, time dependent variable. Please notice its column number. In the Survival Kit time dependent variables are specified in the TIMEDEP statement in the parameter file of PREPARE.

- **3** - Age at first calving class, time independent variable (It does not change during the lifetime of the cow.) Classes can be created according to users consideration to cluster continuous variables into several distinct groups (as stated here for age at first calving) or to include "true" class variables (for example: gender).
Class variables (also time dependent ones!) should be stated in CLASS statement of the parameter file of PREPARE.

- **2** - Number of changes in all time dependent variables (i.e. total number of triplets for particular animal). Here the number denotes that there will be 2 changes in the time dependent variable, and the program expects 2 triplets to follow. This number is not stated anywhere in the parameter file, but should be always present in the database (with value 0 in case of absence or no changes in time dependent variables).

- Two triplets follows with identical structure:

  - **4** - *Column number:* This particular time dependent variable is in 4th column of the database
  - **390** - *Time of change:* Our cow calved for the second time on her 390th day of productive life
  - **2008** - *New value of the time dependent variable:* The value changes from 2007 to 2008

In fact every line of the database tells the "story" of the particular record. In this case cow number 12345 calved for the first time in 2007 at age which belongs to 3rd group for age at first calving (grouping created by the user). Then she calved at 390th day of her productive life in 2008, for the third time at 710th day in 2009 and finally on 960th day after her first calving was culled.

17 Database creation

Several methods can be used to create the input database for the Survival Kit. In case of large databases, it is unavoidable to use certain database handling program to create

the special structure of input data, especially when time dependent variables are present. This particular section is just to give some hints how to deal with the raw data using SAS on the database mentioned earlier in chapter *"A small example"*.

17.1 Input database

Let's consider an input database *small_example.dbf* with following structure:

ID	BEGINNING	FAILURE	CENSORED	SEX	INITIAL_TR	TR_DATE	TR_METHOD
1	1.1.2009	6.1.2009	1	1	0	5.1.2009	10
2	1.1.2009	11.1.2009	1	1	0	7.1.2009	10
3	1.1.2009	12.1.2009	1	1	0	7.1.2009	10
4	1.1.2009	12.1.2009	0	1	0	.	.
5	1.1.2009	16.1.2009	1	2	0	.	.
6	1.1.2009	16.1.2009	1	2	0	7.1.2009	10
7	1.1.2009	18.1.2009	1	1	0	.	.
8	1.1.2009	20.1.2009	1	1	0	.	.
9	1.1.2009	22.1.2009	1	2	0	.	.
10	1.1.2009	22.1.2009	0	2	0	16.1.2009	10
11	1.1.2009	29.1.2009	1	2	0	16.1.2009	10

Here BEGINNING denotes the start of the experiment, FAILURE the death of the particular test subject (those denoted 0 in the CENSORED column were right censored for some reason). If treatment was applied, the corresponding date and method are shown in columns TR_DATE and TR_METHOD.

17.2 The SAS code

In this section, one possible solution is indicated on how to create an input dataset for the Survival Kit from the (almost) raw database given above. ("Almost" raw, because id, censoring status, sex, initial treatment and treatment method were already recoded to the required form.) Please note, that variable names used *here* and in the parameter files *prepare.txt*, *cox.txt* or *weibull.txt* are completely **independent from each other**.

```
dm 'clear log';
dm 'clear output';

* creates SAS file from the input dbf;
```

```
proc import datafile="c:\small_example.dbf"
out=sasuser.small_example replace;
run;

data data_processing;
set sasuser.small_example;

* getting data to the required form;
life_length = failure - beginning;
change_time = tr_date - beginning;

* counting the number of triplets;
nr_triplet=0;
if change_time^=. then nr_triplet = nr_triplet +1;

* text file for the Survival Kit;
file 'c:\Survival_Kit_example\small.dat';
put id +1 life_length +1 censored +1 sex +1 initial_tr +1 nr_triplet;

* writing out triplets to the text file;
if change_time^=. then do;
    put '5' +1 change_time +1 tr_method;
end;

run;
```

This program creates a dataset *small.dat* with the following structure:

```
1  5  1  1  0  1
5  4  10
2  10 1  1  0  1
5  6  10
3  11 1  1  0  1
5  6  10
4  11 0  1  0  0
5  15 1  2  0  0
6  15 1  2  0  1
5  6  10
7  17 1  1  0  0
8  19 1  1  0  0
```

```
9  21 1  2  0  0
10 21 0  2  0  1
5  15 10
11 28 1  2  0  1
5  15 10
```

This is not completely corresponding to the structure of the input dataset given in chapter *"A small example"*, but the outcome is the same. The number of triplets (last number in the longer lines) denotes how many triplets will follow, but it doesn't matter if these are not in the same line.

18 Tips and Tricks

Some additional information that you might find useful.

- Copying and modifying existing parameter files is a good idea. It saves a lot of time compared to creating new parameter files for each run.
- Do not forget about possibility to comment out lines with /* and */ in the parameter files instead of deleting keywords. Again: It saves time in case they are needed later on.
- After *any* modification (especially if you have more versions in separate folders) check that the version you modified and the one you are running is the same. In case you keep getting the same results even after modifications, it is a reason to be suspicious.
- In case you are especially curious about results of a certain effect it is a good idea to put it as the first in the MODEL statement of COX or WEIBULL. In this case these will be displayed at the beginning of the results file and not only after many lines of other (less interesting) results. In other words: You don't have to scroll as much.

Bibliography

Cox, D. (1972). Regression models and life table. *J. Royal Stat. Soc. , Series B*, 34:187–220. with discussion.

Cox, D. R. and Oakes, D. (1984). *Analysis of survival data*. Chapman and Hall, London, UK.

Cox, D. and Snell, E. J. (1966). A general definition of residuals. *J. Royal Stat. Soc. , Series B*, 30:248–275. with discussion.

Ducrocq V. (2001). A two-step procedure to get animal model solutions in Weibull survival models used for genetic evaluations on length of productive life. *Interbull bulletin*, 27:147–152.

Ducrocq, V. and Casella G. (1996). A Bayesian nalysis of mixed survival models. *Genet. Sel. Evol.*, 28: 505-529.

Ducrocq, V. and Sölkner, J. (1994). "The Survival Kit", a FORTRAN package for the analysis of survival data. In: 5th World Cong. Genet. Appl. Livest. Prod., Volume 22, pages 51–52. Dep. Anim. Poultry Sci., Univ. of Guelph, Guelph, Ontario, Canada.

Ducrocq, V. and Sölkner, J. (1998). "The Survival Kit – V3.0", a package for large analyses of survival data. In: 6th World Cong. Genet. Appl. Livest. Prod., Volume 27, pages 447–448. Anim. Genetics and Breeding Unit, Univ. of New England, Armidale, Australia.

Ducrocq, V., Sölkner, J. and Mészáros G. (2010). Survival Kit v6 – a Software Package for Survival Analysis. In: 9th World Cong. Genet. Appl. Livest. Prod., Leipzig, Germany.

Kalbfleisch, J. D. and Prentice, R. L. (1980). *The statistical analysis of failure time data*. John Wiley and sons, New-York, USA.

Klein, J. P. and Moeschberger, M. (1997). *Survival analysis*. Springer-Verlag, New-York, USA.

Liu, D. C. and Nocedal, J. (1989). On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45:503–528.

Maddala, G. S. (1983). *Limited-dependent and quantitative variables in econometrics* Cambridge Univ. Press, UK.

Perez-Enciso, M., Mizstal, I., and Elzo, M. A. (1994). Fspak: an interface for public domain sparse matrix subroutine. In: 5th World Cong. Genet. Appl. Livest. Prod., Volume 22, pages 87–88. Dep. Anim. Poultry Sci., Univ. of Guelph, Guelph, Ontario, Canada.

Prentice, R. and Gloeckler, L. (1978). Regression analysis of grouped survival data with application to breast cancer data. *Biometrics*, 34:57–67.

Schemper M. (1992). Further results on the explained variation in proportional hazards regression. *Biometrika*, 79:202–204.

Index

**, 40
/*, */, 15, 31, 32
Makefile, 10
parinclu*, 9, 10*
parinclu *file, 47, 70*
prepinclu*, 9, 10*

algebraic integration, 43, 44
ALL_TIMES, 49
animal id, 20
animal model, 24, 42
ANIMAL_SOLUTION, 46

BACK_SOLVE, 44, 46
BACKUP, 52, 70
BASELINE, 48, 57, 64
baseline, 7–9
BLOCKED_UNFORMATTED, 45
Bug year 2000, 11, 18

case sensitive, 17, 38
CENSCODE, 19, 74
censored record, 19
CHECK, 50
CLASS, 16, 22, 34, 39, 40, 44, 54, 74
class variable, 8, 31, 40
CODE, 30, 55
COEFFICIENT, 41
COMBINE, 23, 40, 45, 54
comment, 15, 31, 32
compile, 10
COMPRESSED, 45
CONSTRAINT, 9, 36, 50, 58, 63
continuous variable, 8, 22, 31, 40
CONVDT, 21
convergence, 51, 59
convergence criterion, 70
CONVERGENCE_CRITERION, 51, 59
COVARIATE, 30
COX, 9, 15, 26, 29, 32, 36, 57, 70
Cox model, 7
cox.txt, 32, 47, 56, 75

data preparation, 54
date, 15, 18, 20, 21, 23
degrees of freedom, 50
DENSE_HESSIAN, 36, 47, 48, 70
deviance residual, 9, 50, 64
discrete scale, 8, 11, 19, 31, 32
discrete variable, 8, 22, 31, 39, 40
DISCRETE_SCALE, 11, 19, 31, 32

EFFECT, 47, 70
elementary record, 15, 19, 54, 69
elementary records, 31
EPS_BFDEF, 51
EQUAL_SPACE, 49
ESTIMATE, 42, 45, 71
explained variation, 62

FILES, 17, 37
FIND, 50
FIXED_FORMAT, 25, 32
FORCE_POSITIVE, 48
format, 25, 32
FORMOUT, 25
frailty, 7, 8, 42
FREE_FORMAT, 25, 32
fspak, 47
FUTURE, 24, 27, 49

Gauss-Hermite quadrature, 43
GENERAL, 47
generalized residual, 9, 29, 50, 64
genetic parameter estimation, 42
goodness-of-fit test, 64
group of unknown parents, 25
grouped data, 8, 11, 19, 26, 31, 32, 40, 41, 44

hazard function, 7
herd-year effect, 45

ID, 30
IDREC, 20, 24, 27, 30, 44, 45, 54, 69, 73
IMPOSE, 50, 63
IN_CORE, 51, 70
INPUT, 18, 53
INTEGRATE_OUT, 20, 27, 35, 43–46, 69–71
interaction, 23, 40, 45, 54, 69
intercept, 40
ITE_QUASI, 38

JOINCODE, 22, 24, 31
JOINT_MODE, 43, 45, 46, 71

KAPLAN, 40, 48
Kaplan-Meier estimate, 48

Laplacian integration, 42
large dataset, 7, 45, 69
LARGEST, 50
LAST, 47, 57
likelihood ratio test, 9, 47, 57, 62
LINE SEARCH FAILED, 51
log-gamma distribution, 8, 20, 27, 29, 42–44, 46, 48
LOGFILE, 37
LOGGAMMA, 27, 42

Maddala, 62
makefile, 10
martingale residual, 9, 50, 64
MAXFIG, 21
MGS_RULES, 42
minimization, 9, 59
MODEL, 40, 57
MOMENTS, 43, 71
MULTINORMAL, 42
multivariate normal distribution, 8, 42

NBUG_2000, 11, 18
NDIMAX, 17, 34, 47
NEFMAX, 16, 34
new code, 37, 56
NITER_GAUSS, 35
NLEVOUT, 35
NO_HESSIAN, 36, 47
NO_LOGARITHM, 36
NONE, 50
NORMAL, 42
normal distribution, 8, 42
NPGAUSS, 35, 43
NRECMAX, 16, 34, 51, 70
NSTIMAX, 35
NSTRAMAX, 35
NTIMMAX, 35
NVEC_BF, 36
NZE_HESS, 47

ON_DISK, 51
ONLY_STATISTICS, 41
ORIGIN, 39
OUTPUT, 23

partial likelihood, 7
PC, 25, 26
PEDIGREE, 24, 37, 43, 44
polynomial regression, 40
prediction, 24
Prentice and Gloeckler's model, 8, 11, 19, 26, 31, 32, 40, 41, 44
PREPARE, 9, 15, 37, 54, 69, 73, 74
Prepare, 43, 44
prepare.txt, 15, 75

PREVIOUS_TIME, 30
proportional hazards model, 7, 9, 39

QUANTILES, 49
Quasi-Newton, 9, 59

R2 of Maddala, 62
RANDOM, 35, 42, 45
random effect, 8, 20, 27, 42–44, 48
random effects, 7–9, 20, 27
rank, 50
READ_OLD_SOLUTIONS, 37, 43, 51, 70
relationship matrix, 8, 22, 24, 42
RESIDUALS, 50, 57, 64
results, 37
RHO_FIXED, 39, 71
risk ratio, 63

second() (timing subroutine), 10
SEQUENTIAL, 47, 57
sire model, 24, 42, 43
sire-dam model, 22, 24, 41, 42
sire-mgs model, 22, 24, 31, 41, 42
SIRE_DAM_MODEL, 11, 42
sort, 26, 39, 45, 56
sparse matrix, 9, 47, 59, 70
SPECIFIC, 49
standard error, 9, 47, 63
STATISTICS, 41, 65
STD, 47
STD_ERROR, 36, 47
STORAGE, 51, 70
STORE_PREVIOUS_TIME, 20, 27, 30, 45
STORE_SOLUTIONS, 38, 43, 51, 70
STRATA, 26, 35, 39, 48, 70
strata, 8
SURVIVAL, 35, 38, 49
syntax, 29

TEST, 47, 57, 70
TIME, 18, 30, 55, 73
time-dependent, 7, 8, 15, 20–22, 27, 31, 53, 67
time_unit, 11, 19
TIMECOV, 19, 21
TIMEDEP, 19, 22, 74
TITLE, 38
triplet, 22, 53, 74
TRUNCATE, 20
truncated record, 20, 55

UPPER_CASE, 17, 38
USUAL_RULES, 42

variable name, 18
variable type, 18
variance parameter, 42
varlist.txt, 17, 29, 55

Wald test, 63
WEIBULL, 9, 15, 26, 29, 32, 36, 65, 70
Weibull distribution, 7, 8, 20
weibull.txt, 32, 47, 65, 75
WITH_MGD, 37
WITHIN, 20, 44–46, 69