A User Manual for the Multivariate MLE Tool

Before running the main multivariate program saved in the SAS® file "Part2-Main.sas", the user must first compile the macros defined in the SAS® file "Part1-Macros.sas" using the first version of the %MULTICDF macro definition. If already compiled at a previous session, one can specify the directory to which the compiled macros are saved using the LIBNAME statement provided.

The data to be analyzed must be provided in an Excel workbook, where censored values are indicated by the corresponding limit of detection (LOD) values appended to the less than symbol and missing values are represented by a period. Using this form of the data, the LOD values are internally assigned to the censored observations; thus, it is not necessary to create a separate $p \times 1$ column vector of the LOD values for the p variables in the given dataset. This internal assignment enables the program to accommodate variables with multiple LOD values. Additionally, the program itself also creates separate censor indicator variables for each of the variables in the dataset, where a 0 indicates observed, 1 indicates censored, and 2 indicates missing.

In preparing the data to be read by the optimization program, the user must specify values for the global macro variables provided in the following list.

(1) *PATHNAME* must be a character string that specifies the exact location of the Excel workbook containing the data to be analyzed. It is not enclosed in quotation marks.

(2) *SHEETNAME* must be a character string that specifies the exact name of the sheet within the Excel workbook. It is not enclosed in quotation marks.

(3) *LIBNAME* must be a character string assigned to the name of the SAS® library where the SAS® dataset is to be stored.

(4) *DATASET* must be a character string assigned to the name of the SAS® dataset being created.

(5) *NSUBJ* must be a numeric scalar that represents the total number of subjects or observations in the dataset.

(6) *NUMVARS* must be a numeric scalar that represents the total number of variables in the given dataset.

(7) *VARLIST* must be a character string of the actual names of each of the variables to be analyzed. They should be separated by a single space, and each name should be enclosed in quotation marks. For example: ‘x1’ ‘x2’ ‘x3’.

To begin, the SAS® program imports the data from the indicated Excel workbook to the SAS® dataset *LIBNAME.DATASET* and prepares the data for the main procedure. In order to obtain good starting values of the means and variances for the Newton-Raphson optimization procedure, univariate analyses are performed on each of the *NUMVARS* variables in the dataset using SAS/STAT® PROC LIFEREG. The LIFEREG procedure can be used to fit parametric models to left-censored data, which we assume here have a normal distribution. Parameters are estimated by maximum likelihood using a Newton-Raphson algorithm. Using large sample normal approximations, standard errors of the parameter estimates are estimated from the inverse of the observed information matrix. For each univariate test performed, the parameter estimates are written to an output SAS® dataset to be used later on in the program as starting values.

In SAS/STAT® PROC LIFEREG, the log-likelihood function is computed using the log of the response as opposed to the raw data itself. This log-likelihood differs from the log-likelihood obtained using the response in its original form by an additive term of $\sum \log(x_i)$, where the sum is over the non-censored observations. Note, however, that this term is

independent of the unknown parameters and does not influence parameter or standard error estimates.

From the SAS OnlineDoc®, we know that the PROC LIFEREG statement invokes the procedure, and the required MODEL statement specifies the variables used in the regression part of the model and the distribution used for the error component of the model. The starting estimates are obtained by ordinary least squares. The MODEL statement is used to specify the response variable, any explanatory variables, the distribution, and the censored values. Syntactically, two values can be used to indicate the values of the endpoints of the censoring interval. If the two values are the same and are not missing, then it is assumed that there is no censoring and the actual response value is observed. If the lower value is missing and the upper value is not missing, then the upper value is used as a left-censored value. The documentation further specifies that convergence is declared when the maximum change in the parameter estimates between Newton-Raphson steps is less than the value 0.001. If no covariates are specified, then an intercept-only model is fit to the data. The DISTRIBUTION option available in the MODEL statement specifies the distribution type assumed for the response. As written in the SAS OnlineDoc®, by default the initial values for the parameters are computed using ordinary least squares while ignoring censoring, and the log-likelihood function is maximized via a ridge-stabilized Newton-Raphson algorithm. The maximized value of the log-likelihood can take positive or negative values, depending on the specified model and the values of the maximum likelihood estimates of the model parameters. The asymptotic covariance matrix is computed as the inverse of the observed information matrix. According to the SAS OnlineDoc®, the estimated covariance matrix of the parameter estimates is computed as the negative inverse of the information matrix of second derivatives of the log-likelihood function with respect to the

parameters evaluated at the final parameter estimates. If the information matrix is not positive definite, a positive definite submatrix of the information matrix is inverted, and the remaining rows and columns of the inverse are set to zero. The standard error estimates for the parameter estimates are taken as the square roots of the corresponding diagonal elements.

In order to obtain good starting values for the covariances for the Newton-Raphson optimization procedure, the Pearson correlations are computed for all pairs of variables using Base SAS® PROC CORR after imputing half of the LOD value for all of the censored values. The results are saved to another output dataset to be used later on in the program.

Within SAS/IML®, the data are first read into a (*NUMSUBJ* $\times$ *NUMVARS*) matrix **Y**, and the censor indicator variables are read into a (*NUMSUBJ* $\times$ *NUMVARS*) matrix **C**. Two numeric scalars *NR* and *NC* are assigned the number of rows and number of columns of **Y**, respectively, representing the number of subjects and the number of variables in the dataset, respectively.

The user-defined macro %BUBBLESORT is then called to:

(1) sort each row of **C** in the order observed (0) $\rightarrow$ censored (1) $\rightarrow$ missing (2),

(2) create an (*NR* $\times$ *NC*) matrix **VARS** that keeps track of the original variable order for each observation, and

(3) store the number of observed, censored, and missing values for each observation in (*NR* $\times 1$) column vectors **OBS**, **CEN**, and **MIS**, respectively.

Within the %BUBBLESORT macro, each row of **VARS** is initially assigned the values $\{1\,2\,\ldots\,NC\}$, which are used to identify the *NC* variables in the dataset. As **C** is sorted in the appropriate order, the corresponding elements of **VARS** are sorted to keep track of the original variable order. **VARS** is then used throughout the program to correctly reference elements of the

mean vector and (co)variance matrix based on whether the variable values are observed or censored.

For coding convenience, global macro variables are then created within a DATA step using Base SAS® CALL SYMPUT. Since SAS/IML® must be exited before entering the DATA step, **VARS**, **OBS**, and **CEN** need to be read into datasets so that their values are not lost. Once SAS/IML® is exited, a DATA step is used to create the *NR* global macro variables $m_1,\ldots,m_{NR}$ with Base SAS® CALL SYMPUT, which are assigned the number of censored values for each of the *NR* observations. Additionally, *NR* global macro variables $decm_1,\ldots,decm_{NR}$ are created with Base SAS® CALL SYMPUT, which are assigned one less than the number of censored values for each of the *NR* observations. These macro variables are used in another macro as termination values of DO loops.

After the global macro variables are assigned using the DATA step, SAS/IML® is entered once again, and the data are re-read into the matrix **Y**. The numbers of observed and censored values for each observation that were previously stored in the datasets created by the SAS/IML® code are read into the column vectors **NOBS** and **NCEN**. Additionally, the original orderings of the variables for each observation that were stored in a dataset are read into the matrix **VARS**.

The log-likelihood function needed by the Newton-Raphson optimization procedure is then defined within the user-defined SAS® function FULLLIKE. There are *NUMVARS* means, *NUMVARS* variances, and $\frac{NUMVARS(NUMVARS-1)}{2}$ covariances that need to be estimated, for a total of $NUMPARMS=\frac{NUMVARS(NUMVARS+3)}{2}$ parameters that need to be estimated.

The parameter of the FULLLIKE function is defined as **X**, which is a (1 × *NUMPARMS*) row vector of parameter starting values. The vector **X** is initially set to the parameter estimates obtained from SAS/STAT® PROC LIFEREG and Base SAS® PROC CORR as described above. It represents the initial, or starting, values for each of the *NUMPARMS* unique parameters included in the likelihood function. These starting values are to be used by the nonlinear optimization procedure, which employs the Newton-Raphson method. The parameters must be entered in the following order given by

$$\left\{\mu_1, \sigma_1^2, \ldots, \mu_p, \sigma_p^2, \rho_{12}, \rho_{13}, \ldots, \rho_{1p}, \rho_{23}, \ldots, \rho_{2p}, \ldots, \rho_{p-1,p}\right\},$$

where p = *NUMPARMS* . As the Newton-Raphson algorithm iterates, the elements of **X** are updated.

Within the FULLIKE function, the log-likelihood function *LIKE* is first initialized to 0. The counter variable, *COUNT*, used to reference the appropriate elements of **X** during the optimization procedure, is initialized to 1. The value of the global macro variable *NUMVARS*, which is the number of variables in the given dataset, is assigned to the variable *p*. The $p \times 1$ mean column vector **MU** is initialized so that each element has a value of 0, the $p \times p$ (co)variance matrix **SIG** is initialized so that each element has a value of 1, and the $p \times p$ correlation matrix **RHO** is initialized so that each element has a value of 1. The elements of **MU**, **SIG** and **RHO** are then updated in DO loops based on the order of the parameters in the row vector **X** of starting values.

The user-defined macro %MAINPROG is then called, which actually constructs the log-likelihood function based on the characteristics of the given data. Using a DO loop with index

variable *SUBJ*, the contribution of each observation to the log-likelihood function is formed. Through each iteration of the DO loop, the log-likelihood function is updated as follows.

First, the number of observed values and the number of censored values for the current observation are assigned to the numeric scalar variables *CURNOBS* and *CURNCEN*, respectively. The data for the current observation stored in the ($NR \times NC$) matrix **Y** are assigned to the ($NC \times 1$) column vector **CURY**, and the corresponding row of the **VARS** matrix for the current observation is transposed and then assigned to the ($NC \times 1$) column vector **CURVAR**.

Using only the data for the current observation, the user-defined macro %LIKELIHOOD is called, which determines the type of function that the current observation contributes to the overall log-likelihood function as follows.

(1) If all nonmissing values are observed, then the multivariate normal probability density function (PDF) evaluated at the observed values needs to be calculated using the user-defined macro %MULTIPDF. Based on the values of the elements of **CURVAR** for the current observation, the macro %MULTIPDF assigns the appropriate subset of the mean vector **MU** and of the data vector **CURY** to the temporary mean vectors **TEMPMU** and **TEMPY**, respectively. Similarly, the macro assigns the appropriate subset of the (co)variance matrix **SIG** to the temporary (co)variance matrix **TEMPSIG**. It then calculates the multivariate normal PDF using the formula

$$f = \frac{1}{\sqrt{(2\pi)^{CURNOBS}\left|\mathbf{TEMPSIG}\right|}}$$

$$\times \exp\left\{-\frac{1}{2}\left(\mathbf{TEMPY}-\mathbf{TEMPMU}\right)'\mathbf{TEMPSIG}^{-1}\left(\mathbf{TEMPY}-\mathbf{TEMPMU}\right)\right\}$$

and assigns the resulting value to the variable *PDFCONTRIBUTION*.

(2) On the other extreme, if all nonmissing values are censored, then the multivariate normal cumulative distribution function (CDF) evaluated at the LOD values of the respective variables needs to be calculated using the user-defined macro %MULTICDF. Before calculating the CDF, the user-defined macro %CENMUSIG must be called within the macro %LIKELIHOOD in order to construct the temporary mean vector **TEMPMU**, the temporary (co)variance matrix **TEMPSIG**, and the temporary LOD vector **TEMPLOD** for the current observation. These values are then used in calculating the CDF. If the current observation has only one censored value, then the normal CDF is calculated using the available Base SAS® function CDF, which uses the formula

$$F = \frac{1}{\sqrt{2\pi \times TEMPSIG}} \int_{-\infty}^{TEMPLOD} \exp\left[-\frac{(u - TEMPMU)^2}{2 \times TEMPSIG^2} \right] du ,$$

where *TEMPMU*, *TEMPLOD*, and *TEMPSIG* are all numeric scalars. If, on the other hand, the current observation has more than one censored value, then the multivariate normal CDF is estimated using the %MULTICDF macro, which is constructed based on an algorithm presented in Genz (1992), with the exception that the integrals are approximated using an 8-point Legendre-Gauss quadrature rule instead of Monte Carlo estimation. For coding purposes, the additional macros %GENDOSTMTS, %DEFINEE, %GENWGHTS, and %GENENDSTMTS are created, each of whose purpose is evident in examining the code. The resulting value is assigned to the variable *CDFCONTRIBUTION*.

(3) Finally, if some of the nonmissing values are observed while others are censored, then the multivariate normal PDF evaluated at the observed values needs to be calculated using the macro %MULTIPDF, and then the conditional multivariate normal CDF, conditioned

on the observed variables and evaluated at the LOD values of the respective censored variables, needs to be calculated with the user-defined macro %CONDCDF. For the observed values, the multivariate normal PDF is calculated as in (1) above, and the resulting value is assigned to the variable *PDFCONTRIBUTION* . Within the %CONDCDF macro, the mean vector **MU** and the (co)variance matrix **SIG** are then partitioned into submatrices of the censored variables and the observed variables by calling the user-defined macros %PARTMU and %PARTSIG, respectively. These submatrices **PARTMUO**, **PARTMUC**, **PARTSIGOO**, **PARTSIGOC**, **PARTSIGCO**, and **PARTSIGCC** are partitioned based on the values of the **VARS** vector of the current observation, where 'O' represents the observed portion and 'C' represents the censored portion. Then the temporary mean vector **TEMPMU** for the current observation is assigned using the formula:

$$\mathbf{TEMPMU} = \mathbf{PARTMUC} + \mathbf{PARTSIGCO}\left(\mathbf{PARTSIGOO}\right)^{-1}\left(\mathbf{PARTYO} - \mathbf{PARTMUO}\right),$$

where **PARTYO** is the observed partition of the data vector **CURY**. Similarly, the temporary (co)variance matrix **TEMPSIG** for the current observation is assigned using the formula

$$\mathbf{TEMPSIG} = \mathbf{PARTSIGCC} - \mathbf{PARTSIGCO}\left(\mathbf{PARTSIGOO}\right)^{-1}\mathbf{PARTSIGOC}.$$

The LOD values of the censored variables of the current observation are assigned to the column vector **TEMPLOD**. If the current observation has only one censored value, then the normal CDF is calculated using the available Base SAS® function CDF as described earlier.

If, on the other hand, the current observation has more than one censored value, then the multivariate normal CDF is estimated using the %MULTICDF macro. As stated in (2) above, the %MULTICDF macro uses an algorithm presented in Genz (1992), with the exception that the integrals are approximated using an 8-point Legendre-Gauss quadrature rule instead of Monte Carlo estimation. The resulting value is assigned to the variable *CONDCONTRIBUTION*. Thus, the contribution of the observation is the product of *PDFCONTRIBUTION* and *CONDCONTRIBUTION*.

After the contribution of the current observation is calculated using (1), (2) or (3) above, the resulting value *PDFCONTRIBUTION*, *CDFCONTRIBUTION* or *PDFCONTRIBUTION* $\times$ *CONDCONTRIBUTION* is finally assigned to the variable *f* within the %LIKELIHOOD macro. If the value of *f* is greater than 10^{-10}, then back in the macro %MAINPROG the natural logarithm of *f* is subtracted from the log-likelihood function *LIKE*, which is adjusted as each individual observation passes through the algorithm just described. After the log-likelihood function *LIKE* is adjusted for all *NSUBJ* observations, the FULLLIKE function is exited and the value of *LIKE* is returned.

The constraints on the parameters that must be passed to the optimization algorithm are assigned with the user-defined macro %CONSTRAINTS, which generates the list of upper and lower parameter constraints. Restrictions are defined so that the variances are all greater than or equal to 10^{-10} and correlations between unlike variables lie within the interval [-1, 1]. The constraints are printed assuming the following order of the parameters: $\left\{\mu_1, \sigma_1^2, \ldots, \mu_p, \sigma_p^2, \rho_{12}, \rho_{13}, \ldots, \rho_{1p}, \rho_{23}, \ldots, \rho_{2p}, \ldots, \rho_{p-1,p}\right\}$. Specifically, the macro prints the lower bounds, followed by a comma, followed by the upper bounds for these means, variances, and correlations.

Finally, the SAS/IML® subroutine NLPNRA is ready to be called, which performs the nonlinear optimization by the Newton-Raphson method. Described in detail in the SAS OnlineDoc®, the SAS/IML® subroutine NLPNRA is called using the phrase CALL NLPNRA(*RC*, **XRES**, "*FULLLIKE*", **X**, , **CON**), where the parameters are defined as follows.

(1) The *FULLIKE* module argument specifies the user-defined SAS/IML® module defining the objective function. It returns the value of the objective function $f = f(\mathbf{X})$, which is evaluated at the point **X**.

(2) The argument **X** specifies a row vector that defines the number of parameters, and it represents a starting point for the iterative optimization process.

(3) The **CON** argument specifies a constraint matrix that defines lower and upper bounds for the parameters.

(4) *RC* is the scalar return code that indicates the reason for the termination of the optimization process, where successful termination is signified with a return code greater than zero, whereas unsuccessful termination is denoted with a return code less than zero—meaning that the result **XRES** is not reliable.

(5) **XRES** is the row vector of the parameters that contains the optimal point, of course only when the return code is greater than zero.

As noted in the SAS OnlineDoc®, the SAS/IML® subroutine NLPNRA uses a pure Newton step at each iteration when both the Hessian is positive definite and the Newton step successfully reduces the value of the objective function. Otherwise, it performs a combination of ridging and line-search to compute successful steps. If the Hessian is not positive definite, a multiple of the identity matrix is added to the Hessian matrix to make it positive definite. The documentation also states that the subroutine requires continuous first- and second-order

derivatives of the objective function inside the feasible region. If second-order derivatives are computed efficiently and precisely, the SAS/IML® subroutine NLPNRA does not need many function, gradient, and Hessian calls. If only function calls are used to compute finite difference approximations for second-order derivatives, computational time can be extremely long and the results can carry significant rounding errors; however, the "*GRD*" input argument can be used to specify a module that computes first-order derivatives analytically, which can drastically reduce the computation time for numerical second-order derivatives. During each iteration, as explained in the SAS OnlineDoc®, a line search is done along the search direction to find an approximate optimum of the objective function. The default line-search method uses quadratic interpolation and cubic extrapolation.

Finally, the SAS/IML® subroutine NLPFDD is then called to calculate the Hessian matrix using the optimal solution provided by the SAS/IML® subroutine NLPNRA. The variances of the parameters are found by taking the inverse of the Hessian matrix, from which standard errors are obtained by taking the square roots of the individual elements.