# A review of mixed-effects models in S-plus (version 6.2)

Edmond S.W. Ng
Centre for Multilevel Modelling
Institute of Education

First version: Dec. 2004
Last update: May 2005

## 1. A brief introduction to S-plus

### *1.1 Background*

S-plus is a software for exploratory data modelling and statistical analysis, with advanced graphic facilities and a graphical user interface (GUI).

It is a commercial implementation of the object-oriented S[i] programming language which was originally developed by John Chambers[ii] and his colleagues at Bell Labs (of AT&T, now Lucent Technologies) in the late 70's. This is compared with R[iii] which is sometimes referred to as the "free clone" of the S language distributed freely by the Free Software Foundation[iv]. The S language was specifically designed for organising, visualizing and programming with data.

The S language gained popularity in post-graduate and higher academic environment in North America in the early 80's. The commercialization of the S language in the form of S-plus came about in 1987 through the initiatives of R. Douglas Martin who founded StatSci Inc. In 1993 StatSci obtained an exclusive license for selling products incorporating S. In the same year the company was sold to MathSoft Inc. and StatSci became the Data Analysis Products Division (DAPD) of MathSoft. MathSoft was later consolidated, had their headquarters migrated to Seattle and renamed the company to Insightful in 2001. S-plus has since been marketed and supported by Insightful Corporation[v]. At the time of writing the latest version of S-plus for Windows is 6.2 and is the version under review in this report. The incorporated S language is at version 4.

User-contributed add-on packages or libraries are held at an electronic archive, StatLib. It is independent of Insightful and the packages are downloadable from the StatLib website: http://lib.stat.cmu.edu/S. S-news is an email newsgroup for both S and S-plus users. See the following link for further information: http://www.biostat.wustl.edu/s-news/s-news-intro.html.

### *1.2 Operating systems*

S-plus 6 is available on Windows:

- recommended system requirements to run S-plus 6 for Windows are Pentium II/233 processor with 96 MB of RAM
- at least 275MB of free disk space for the typical installation (and, if not installing on drive **C:\**, an additional 75MB free disk space on drive **C:\** to unpack the distribution
- support platforms: Windows NT 4.0 (Service Pack 6), 2000, XP Professional and 2003 server

[i] http://cm.bell-labs.com/cm/ms/departments/sia/S/
[ii] http://cm.bell-labs.com/cm/ms/who/jmc/index.html
[iii] http://www.r-project.org/
[iv] http://www.gnu.org/
[v] http://www.insightful.com/

and UNIX platforms including Solaris and Linux:

- supported platforms: Sun Solaris 2.8 or 2.9 on SPARC 32-bit architecture, Sun Solaris 2.8 or 2.9 on SPARC 64-bit architecture, Red Hat Enterprise Linux 3.0, Red Hat Linux 8, 9, IBM AIX 5.1, HP Alpha running Tru64 UNIX 4.0F or 5.1A, and HP-UX 11.0 or later.

## *1.3 Data input/output functionality*

S-plus can read and write a wide variety of data formats and supports the Open DataBase Connectivity (ODBC) standard, allowing users to import and export between S-plus and any ODBC-compliant database. In addition, it can import data from financial databases such as Bloomberg, FAME, and MIM.

Data can be read in by using the Import From File dialog from the GUI or by commands in the Commands window. Data in ASCII (tab delimited, fixed-width or other spreadsheet-like) format can be read in by command-line functions like `scan` and `read.table`. A more versatile command to use is `importData` which lets users read in data from a wide variety of formats.

Data types that can be imported and exported are SAS, SPSS, Excel, Text (ASCII), Quattro Pro, Paradox, Lotus 1-2-3, dBase, Sigma Plot (import only), Systat, STATA, Gauss, Access, MATLAB, LIM, Bloomberg, FAME, Minitab, FoxPro, Epi Info, Oracle, Sybase, DB2, SQL Server, ODBC and HTML (export only).

Large data sets are supported in S-plus 6 through new techniques in handling larger objects in S version 4. It is, however, not clear from its current documentation of the size of the largest data object that S-plus 6.2 can handle.

Graphics can be exported in the following formats: Windows Bitmap (.BMP), Encapsulated PostScript (.EPS), CompuServe (.GIF), GEM Bitmap (.IMG), JPEG (.JPG), Adobe Photoshop (.PSD), Adobe PDF (.PDF), HP Printer Control Language (.PCL), PaintBrush (.PCX), Tagged Image Format (.TIF), True Vision Targa (.TGA), Windows Metafile (.WMF), and Portable Network Graphics (.PNG). Users can publish interactive graphic information on the Web using the new S-plus Graphlets (see reference on S-plus website: http://www.insightful.com/graphlets).

Connectivity with programming languages C++ (Windows) and JAVA (Unix) is made possible in S-plus 6 through its CONNENT/C++ Foundation Class Library. This allows C++/JAVA programmers to write their own program using data objects from the S engine, run S functions and manipulate the results or vice versa.

See S-plus User's Guide (Insightful (2001)) for more detailed description of the import and export facilities in S-plus 6 for Windows.

## *1.4 Interface features*

### 1.4.1 Launching S-plus

S-plus can be launched in three different modes (BATCH, CONSOLE and GUI) from the Windows Start menu.

When it is launched from the batch mode, a dialog called BATCH appears on the screen. There are fields to enter the Input File, Output File, Log File and Working Directory (the default directory is shown in the field). The input file containing S-plus commands can be prepared by any word-processor in plain-text (or ASCII) beforehand. Options are given to provide control over the output and to stop processing when the first error is encountered.

When one chooses to start S-plus from the CONSOLE mode, a DOS-type window appears showing copyrights information and the full path of the default directory. After clicking once on Enter, a rightpointing arrow prompt appears and it is ready for the use to enter commands interactively. For those who use R, this console is very similar to the R Console window in the R GUI.

### 1.4.2 Starting with GUI mode

When S-plus is launched from the GUI mode, the user is prompted to enter a folder in which a S-plus project is to be stored in the optional Open S-plus Project window. A S-plus project folder may include data sets, graph sheets, Excel worksheets, Report Files, Script Files, and project settings for the Object Explorer. A default folder with full path is automatically inserted for the user. Clicking on the OK button launches the S-plus GUI. This is how the reviewer uses S-plus in this review.

The user is then presented with the Object Explorer and Commands windows in the GUI. Data sets, graphs, functions, and other objects used during the current S-plus session are shown in the Object Explorer. The Commands window allows the user to enter command-line functions interactively. The function `source` allows users to run S-plus in batch mode within the GUI.

Users may also choose to open a script file (with file extension *.ssc) from the File menu to store and run S-plus commands from. The Script File screen is split into two halves. The top half shows the content of the script file while the bottom half shows the output from a run. Users may choose to run the whole script or a section of the script by highlighting a section and then issue a run by either clicking the play button in the menu bar or by pressing F10.

The shell (or GUI) of S-plus offers point-and-click facilities allowing users to access the built-in functions for data manipulations, graphing and data analysis though advanced users might find issuing commands in the Commands window allows more flexibilities. An extensive list of statistical techniques is available from the S-plus GUI (they can all be called using the equivalent commands). They include linear models, ANOVA, nonlinear regression (maximum/quasi- likelihood), nonparametric regression (including GAMs and Loess), linear and nonlinear mixed-effects models, tree models, smoothing, resampling, multivariate analysis, cluster analysis, calculation of power and sample size, survival analysis, time series analysis, and more. In this review all models are fitted using S-plus/S commands.

### 1.4.3 Getting help

Two types of on-line help are available from the GUI. S-plus Help from the menu bar provides help on using the GUI while Language Reference is for the S-plus/S language. On-line help is also available from the Commands windows using the function `help()`. Once a S-plus function library has been loaded its reference and documentations can be accessed under Help from the menu bar.

### 1.4.4 Menus

The software comes with a number of online manuals (in PDF format) accessible from Help. The general references are Getting Started Guide, User's Guide, Programmer's Guide and S-plus supplement. The advanced references include Guide to Statistics Volumes 1 and 2, LME for SAS PROC MIXED Users, Robust Library and Missing Data Analysis Library.

## 2. Standard modelling tools for multilevel analysis

### *2.1 Features and syntax structure*

The main S-plus functions for fitting multilevel models are `lme` and `nlme` (of the `nlme` library by Pinheiro and Bates (2000)) for normal responses and `glme (correlatedData` library (Chao (2003)) for normal and discrete outcomes. They share a useful function, `groupedData`, for specifying grouped data structure of data objects in S-plus which greatly facilitates the displaying of and model fitting on hierarchical data. The `groupedData` object has the following syntax structure:

```
groupedData(response~covariate(s)|group,data,options)       (1)
```

The first part of the expression specifies the response and the primary covariate(s) of interest. If none cannot be identified as the primary covariate, "1" (for constant) can be used. The multilevel grouping of data, e.g. a three-level structure of pupils in classes nested within schools, can be specified as *`g1/g2`* in the *group* part of the syntax where g1 is replaced with the school identifier and g2 by the class identifier. In S-plus pupil is not taken as a level of grouping. That is, level one in, say, MLwiN is not taken as a level in S-plus. Once such hierarchical structure has been specified functions are available for plotting (e.g. trellis plots) and summarizing the data over each level of grouping.

The function, `lme`, fits linear mixed-effects model with much flexibility given to the specification of the within-group variation structure. Within-group residuals are allowed to be correlated and/or have unequal variances. `lme` has the following syntax structure:

```
lme(response~covariate(s),data,random,correlation,weights,method,options) (2)
```

The argument `response` in the expression is the response of the data being modelled. The next section, after the ~ operator, is for specifying the fixed part of the model and is sometimes known as the formula part of the syntax. `data` can be a data frame or `groupedData` object. If it is the latter, `lme` will pick up the grouped data structure inherited in the object. `random` is optional and is for specifying the random part of the model. For example, using the school example again if one wants to include a constant term and a covariate of GCSE score in the random part of the model, the syntax is `random=~gcse`. The constant term ("1") is included implicitly while a "-1" in the syntax will remove it. The result of the syntax is to include both the constant term and the GCSE score in the random part in all the grouping levels (i.e. school and class levels) of the model.

If different covariates are to be included in different levels, the function `list` is used to bind the formulae for different levels as a single argument for `random`. For example, `random=list(School=~GCSE, Class=~1)` specifies a random intercept and random slope for GCSE at the school level and only a random intercept at the class level.

To update part(s) of and re-fit an existing fitted model object use the function `update()`.

`correlation` is used to specify the optional within-group correlation structure. A set of classes of correlation structure are available, for example, `corCompSymm` for compound symmetry, `corAR1` and `corARMA` for time series models, etc. See `corStruct` in S-plus documentation for details. `weights` is used for specifying the within-group heteroscedasticity structure (see `varFunc`). For example, `varIdent` can be used to allow different variances in different stratum, gender say, of a stratification variable. Maximum likelihood and (REstricted)ML estimates can be derived via `method`. The default is REML.

The function `glme` of the `correlatedData` library for normal and discrete outcomes uses similar syntax as `lme` except that it has the additional options for specifying `family` for different distributions (including Gaussian, binomial, poisson, Gamma, inverse.Gaussian and quasi) with different link functions and `dispersion` for modelling under- or over-dispersion. The estimation methods available in `glme` are (restricted) penalized quasi-likelihood, (restricted) marginal quasi-likelihood, adaptive Gaussian quadrature and Laplace approximation to the likelihood. At its current release the function `glme` is limited to fitting hierarchical models of two grouping levels (i.e. pupils in classes nested with schools where pupil is not considered as a level).

General estimating equations (GEE) or population average models can be fitted by the function `gee` of the same library. Since the focus of this review is on random effects models the function `gee` is not considered any further. Interested readers are referred to the documentation of the `correlatedData` library.

The function `glmmPQL` of the MASS library is also available for fitting mixed-effects models with discrete outcomes. Its syntax is similar to those of (2) above. Its estimation method is penalized quasi-likelihood.

The use of the S-plus functions mentioned in this section will be demonstrated in the models shown in section 3. A brief check of modelling capability of S-plus has been summarized in Table 1.

### *2.2 Tools for statistical inference and model diagnostics*

For models fitted with `lme`, supply as an argument a fitted object to `summary.lme(`*fitted obj*`)` to extract details of the model like AIC (Akaike Information Criterion), BIC (Bayesian Information Criterion) to assess the model fit, and standard errors, t-values, degrees of freedom and p-values of the individual fixed effect parameters. `plot.lme` provides a series of diagnostic plots including residuals plots versus fitted values, boxplots of residuals, normal plots and Cook's distance plots.

Nested models can be compared using the function `anova.lme` which performs likelihood ratio tests and display the AIC and BIC values for each model being compared, or by `simulate.lme` which performs simulation-based evaluation of the likelihood ratio statistics. Some warnings regarding the use of REML estimates in such evaluations are given in Pinheiro and Bates (2000) (in section 2.4).

Approximate confidence intervals for fixed as well as random parameters can be calculated by `intervals.lme()`. Random effects at any level can be extracted by `random.effects.lme()` and predictions by `predict.lme()`. Most of these functions can also be used with models fitted using `glme` of the `correlatedData` library by removing "`.lme`" from the end of the commands. Documentation on the use of these functions with `glme` is, however, limited.

The available tools for inference and diagnostics in S-plus have been summarized in Table 2.

## 3. Model specifications – Basic models

The following benchmarking is performed with S-plus for Windows 6.2 (Rev Date: 2 Dec. 2003 Build: 6713) on a Samsung laptop computer running on Windows XP Professional with Intel Pentium M processor at 1700Mhz and 512MB of RAM.

The standard notation as used in the software, MLwiN, is adopted here. Lowest level unit is denoted by the subscript *i* (e.g. pupil), nested within the next higher level *j* (e.g. class) and then *k* (e.g. school) and so on as the number of levels increases. Note that levels in S-plus are defined as the number of levels of nested random effects in a model. For example, a two-level model in MLwiN is essentially a "one group-level" model in S-plus. Clear distinction will be made between the two conventions whenever it is called for.

In the current version of S-plus linear and nonlinear mixed-effects models can be fitted by the functions `lme` and `nlme` from the user-contributed `nlme` package by Pinheiro and Bates (2000). This library is included in S-plus 6. Generalised linear mixed-effects models can be handled by the `glme` function of the `correlatedData` library (Chao (2003)) written by S-plus' in-house research library. It is downloadable from the Insightful research libraries website: http://www.insightful.com/downloads/libraries.

## *3.1 Two-level Normal models*

The dataset, EXAM, described in the MLwiN User's Manual is used here for illustration. The data came from a much larger set of examination results from six inner London Education Authorities. The primary aim of the original analysis was to examine whether schools varied in terms of their 'effectiveness' in promoting students' learning and development while taking into account the variations in characteristics of students when they started secondary school. There are 4,059 students from 65 schools in this dataset. The response variable consists of exam scores obtained by each student at age 16 normalised to have an approximate standard normal distribution (normexam). Covariates include:

- $x_{1ij}$ : school intake variable, standardised London reading test (`standlrt`);
- $x_{2ij}$ : gender of students with boys as the reference group (`gender` as factor);
- $x_{3j}$ : school gender for mixed school against girls' school (`schgend` as factor);
- $x_{4j}$ : school gender for boys' school against girls' school (`schgend` as factor).

Five models will be considered with varying fixed and random parts and the results are shown in Table 3.

The S-plus function `groupedData`, discussed in 2.1, facilitates the modelling of hierarchical data by offering a way to specify a data frame object together with its hierarchical structure. The hierarchical structure of the data in this example can be specified as follows.

```
m1.0 <- groupedData(normexam~1|School, data=EXAM, FUN=mean,
      labels=list(x="constant", y="Normalised exam score"), units=list(x="",
      y="score point"))
```

The above statement specifies that the response, `normexam`, is grouped by a group-level identifier School with a constant term as the only covariate in the structure. The "~" in the expression can be read as "is modelled as". `EXAM` is a data frame object. Mean is chosen as the optional summary function, `FUN`, of the response for ordering the data in plots. Finally optional `labels` and `units` are given for display purpose for the response and the covariates in the grouped data object. The entire grouped data structure is stored as object `m1.0`.

A basic variance component model with covariates `standlrt`, `gender` and `schgend` can be expressed as follows.

$$y_{ij} = \beta_{0j} + \beta_1 x_{1ij} + \beta_2 x_{2ij} + \beta_3 x_{3j} + \beta_4 x_{4j} + e_{0ij}$$

$$\beta_{0j} = \beta_0 + u_{0j} \ ,$$

where $e_{0ij} \sim N(0, \sigma_{e0}^2)$ and $u_{0j} \sim N(0, \sigma_{u0}^2)$. Student's gender and school gender are declared as factors. The "contr.treatment" contrast method is chosen for the factors in which each level of a factor variable is compared against its first level. The syntax for these specifications is as follows.

```
exam$schgend <- factor(exam$schgend)
exam$gender  <- factor(exam$gender)
options(contrasts=c(factor="contr.treatment",ordered="contr.poly")).
```

The model is then fitted by supplying the grouped data object, `m1.0`, to `lme` as an argument and then saved as a fitted object, `fm1.0`, with the following syntax:

```
fm1.0 <- lme(m1.0).
```

A more complicated model explored is the one with an interaction between `gender` and `standlrt`. The corresponding model is expressed as:

$$y_{ij} = \beta_{0j} + \beta_1 x_{1ij} + \beta_2 x_{2ij} + \beta_3 x_{3j} + \beta_4 x_{4j} + \beta_5 (x_{3j} \times x_{4j}) + e_{0ij}$$

$\beta_{0j} = \beta_0 + u_{0j}$,

where $e_{0ij}$ and $u_{0j}$ are as previously defined. This new model can be fitted by using the `update` function to update the fixed part of the model only, for example:

```
fm1.2 <- update(fm1.0, fixed=normexam~standlrt*gender+schgend).
```

Interaction and its main effects of `standlrt` and `gender` are included in the model by putting `standlrt*gender` in the covariate part of the syntax as shown above. The same model could have been specified by `standlrt+gender+standlrt:gender+schgend` where `standlrt:gender` is the interaction term.

A model with random slopes on `standlrt` can be fitted using the `update` function to update only the random part of the model. Since the only difference between this and the earlier fitted model, `fm1.2`, is in the random part, only the new specification for the random part needs to be included in the syntax. The constant term in included in the random part implicitly.

```
fm1.3 <- update(fm1.2, random=~standlrt|School)
```

Stratum-specific variances at the lowest level (level 0 in S-plus terms or level 1 for MLwiN) can be specified in the `weights` option of the modelling statement. In this case boys and girls are assumed to have different variances at the lowest level of the hierarchy. The model can be expressed as:

$y_{ij} = \beta_{0j} + \beta_{1j}x_{1ij} + \beta_2 x_{2ij} + \beta_3 x_{3j} + \beta_4 x_{4j} + \beta_5(x_{1ij} \times x_{2ij}) + e_{ij}$

$\beta_{0j} = \beta_0 + u_{0j}$

$\beta_{1j} = \beta_1 + u_{1j}$

, where $e_{ij} = \alpha_0 + \alpha_1 x_{2ij}$ and it can be fitted by updating the previous fitted object using the syntax:

```
fm1.4 <- update(fm1.3, weights=varIdent(form=~1|gender))
```

`varIdent` belongs to a set of classes of variance functions, `varFunc`, provided in the `nlme` library to model the variance structure of within-group errors using covariates. It can be used to represent a variance model with different variances for each level of a stratification variable `s`, where *s=1…S*, $Var(\varepsilon_{ij}) = \sigma_e^2 \delta^2_{S_{ij}}$. $\delta_1$ is set at 1 for identifiability purpose as the baseline. Variances at other levels of the stratification are represented in the output as proportions to this baseline.

Level one variance may also be dependent on a continuous covariate, e. g. `standlrt`. The model is:

$y_{ij} = \beta_{0j} + \beta_{1j}x_{1ij} + \beta_2 x_{2ij} + \beta_3 x_{3j} + \beta_4 x_{4j} + \beta_5(x_{1ij} \times x_{2ij}) + e_{ij}$

$\beta_{0j} = \beta_0 + u_{0j}$

$\beta_{1j} = \beta_1 + u_{1j}$,

where $e_{ij} = \alpha_0 + \alpha_1 x_{1ij}$ and $x_1$ is the continuous variable `standlrt`. Another built-in class of variance function to be considered here is `varConstPower`. This variance model is defined as $Var(e_{ij}) = \sigma_e^2 (\delta_1 + |\upsilon_{ij}|^{\delta_2})^2$. Under this specification the variance is modelled as a function of a constant, $\delta_1$, plus a power, $\delta_2$, of the absolute value of the variance covariate, $\upsilon_{ij}$, or `standlrt` in this case. The syntax for specifying this variance function via the optional `weights` while holding the constant, $\delta_1$, fixed at the value of 1 is:

```
weights= varConstPower(form=~standlrt,fixed=list(const=1)).
```

In general, the within-group variance function is defined as $Var(\varepsilon_{ij} \mid u_j) = \sigma_e^2 g^2(\mu_{ij}, \upsilon_{ij}, \delta)$, $i = 1 \ldots n_j, j = 1 \ldots M$ where $\mu_{ij} = E[y_{ij} \mid u_j]$, $\upsilon_{ij}$ is a vector covariates involved in the variance function and *g(.)* is the variance function. Other forms of the function *g(.)* are available in the `nlme` library. Interested readers are referred to section 5.2 of Pinheiro and Bates (2000).

Parameter estimates and S-plus syntaxes of the various models considered in this section are shown in Table 3.

## *3.2 Three-level Normal models*

CHEM97 is a large dataset and is useful to test the efficiency of the algorithm. It contains A/AS-level chemistry examination results of 1997 on 31,022 students from 2,280 schools in 131 Local Education Authorities (LEA) in England. The explanatory variable, $x_1$, is a school intake score, the average GCSE score of students. It has been centred around its mean before putting into the model.

Two models are considered. The first one is a three-level (or two group levels in S-plus terms) variance component model:
$y_{ijk} = \beta_0 + v_{0k} + u_{0jk} + e_{0ijk}$,
where $e_{0ijk} \sim N(0, \sigma_{e0}^2)$, $u_{0jk} \sim N(0, \sigma_{u0}^2)$ and $v_{0k} \sim N(0, \sigma_{v0}^2)$ and *i* indicates student, *j* for school and *k* for LEA. The other one is the same model but including the averaged GCSE (centred) score as the only covariate. The model is:
$y_{ijk} = \beta_0 + \beta_1 x_{1ijk} + v_{0k} + u_{0jk} + e_{0ijk}$,
where $x_1$ is the centred averaged GCSE intake score.

The `groupedData` function is used to declare the hierarchical structure of the data before fitting the models. The syntax for specifying a three-level structure with constant as its main covariate is as follows.

```
fm3.2.1 <- groupedData(points~1|Lea/School, data=CHEM97,
   FUN=max, labels= list(x="constant", y="A-lev chem point score"),
   units=list(x="", y="score point"))
```

School (group-level 2 in S-plus terms) is nested within LEA (group-level 1), that is, g1/g2, as shown in the *group* part of the expression above. The results of fitting these two models are summarized in Table 4.

## *3.3 Two-level models for binary data*

The BANG data come from the 1998 Bangladesh Fertility Survey. The file consists of a sub-sample of 1,934 women grouped in 60 districts. The binary response variable is the use of contraceptive at the time of the survey (1 for those who used contraception and 0 otherwise). The covariates of interest are:

| | |
|---|---|
| `urban`: | region of residence (1 for urban, 0 rural), |
| `cage`: | centred age about the mean, |
| `c2-c4`: | number of living children (no children set at 0 as baseline, c2=1 child, c3=2 children and c4=3 or more children) |

Two models are explored on the BANG data. A variance component model with all covariates included and the same model with random effect on `urban` are fitted using the function `glmmPQL` (using Penalized Quasi-Likelihood) from the `MASS` library. The syntax for `glmmPQL` is similar to those of `lme` except that users have to specify the family of the distribution of the response and the link function to use. Dispersion can be modelled or prespecified in `glmmPQL`. Results of the fitted models with logit and probit links are shown in Table 5.

## 3.4 *Two/three-level models for count data*

The MMMEC data come from a study in the European Community looking at the impact of UV radiation exposure on Malignant Melanoma Mortality. A three-level hierarchical structure with which county is nested within region and region nested with nation is present in the data.

Only one variable namely, `uvb`, is considered as the covariate in the models explored in this section. It is a measure of the ultraviolet light (UVB) dose reaching the earth's surface in each county and has been centred around it mean.

The function, `glme`, of the `correlatedData` library (Chao (2003)) fits generalised linear mixed-effects models in the formulation described in Breslow and Clayton (1993) and it will be used for modelling these data. Syntax for `glme` is again similar to those for `lme` except that family of the distribution of the response needs to be specified (with Gaussian as default) and dispersion can either be prespecified or left to be estimated. Optional starting values can be provide as initial values for estimating parameter values. Offsets are allowed in `glme` and it can be specified as `offset(lnexp)` for the log of expected deaths in the formula.

Two and three level (in MLwiN terms) models for the standardized mortality rates (SMR) each with `uvb` as the only covariate are considered. The models are:

$$\lambda_{ij} = \frac{y_{ij}}{E_{ij}} = \exp(\beta_0 + \beta_1 x_{1ij} + u_{0j})$$

$$\lambda_{ijk} = \frac{y_{ijk}}{E_{ijk}} = \exp(\beta_0 + \beta_1 x_{1ijk} + v_{0k} + u_{0jk}),$$

where $E_{ijk}$ is the expected number of death, $x_{1ijk}$ is the level of exposure to UVB, $u_{0jk} \sim N(0, \sigma^2_{u0})$ and $v_{0k} \sim N(0, \sigma^2_{v0})$. Results of the above models are shown in Table 6.

## 3.5 *Growth type models for repeated measures data*

The OXBOYS dataset consists of repeated measurements of 26 boys taken on 9 occasions between the age of 11 and 13 years. The measurements are approximately 0.25 years apart. Two models are explored on these data. The first polynomial growth curve is expressed as:

$$ht_{ij} = \beta_{0j} + \beta_{1j} t_{ij} + \beta_{2j} t^2_{ij} + \beta_3 t^3_{ij} + \beta_4 t^4_{ij} + e_{ij}$$

$$\beta_{hj} = \beta_h + u_{hj}, \ h=0,1,2$$

$$u_{hj} \sim MVN(0, \Omega_2), \ e_{ij} \sim N(0, \sigma^2_e),$$

and $t_{ij}$ is the centred age of a boy. The random effects for $\beta_0$, $\beta_1$ and $\beta_2$ are allowed into the model by specifying `random=~cage+cage2|id` in the syntax where `id` is boy-level identifier.

A more complicated model which adds sine and cosine functions into the fixed part of the model with the autocorrelation structure for level 1 (or within-group in S-plus terms) residuals is expressed as:

$$ht_{ij} = \beta_{0j} + \beta_{1j} t_{ij} + \beta_{2j} t^2_{ij} + \beta_3 t^3_{ij} + \beta_4 t^4_{ij} + \beta_5 \sin_{ij} + \beta_6 \cos_{ij} + e_{ij}$$

$$\beta_{hj} = \beta_h + u_{hj}, \ h=0,1,2$$

$$u_{hj} \sim MVN(0, \Omega_2), \ e_{ij} = \rho e_{(i-1)j} + \delta_{ij}$$

The sine and cosine terms in the model are the sine and cosine of $\pi$`*season/6`. `Season` is the season number in decimal year and the autoregressive structure incorporated into the model is that of order 1 for the level 1 residuals. The autoregressive structure is specified in the syntax via the optional `correlation` argument as `cor=corAR(1)`. A number of built-in correlation structures are available via the `corStruct` classes of the `glme` library. Interested readers are referred to section 5.3 of Pinheiro and Bates (2000). Results of the models explored in this section are shown in Table 7.

## 4. Model specifications – other random effects models

### *4.1 Multinomial models for ordered or unordered response*

Multinomial logistic models for unordered and ordered categorical models can be handled by the functions `multinom` (library `nnet`) and `polr` (`MASS`) respectively. Both of these libraries are included in the version of S-plus being reviewed. Neither of them is designed for data with hierarchical structure. Interested readers are referred to their respective online documentation.

### *4.2 Cross-classified models*

The data, 2LEV-XC (renamed as XC in the syntax), were collected from 3,435 children who attended 148 primary schools (`pid`) and 19 secondary schools (`sid`) in Scotland. One of the goals of the original analysis was to disentangle the effects of attending the 148 primary schools and 19 secondary schools on the attainment score (`attain`) at age 16. The six variables in the data set are:

| | |
|---|---|
| `vrq`: | a verbal reasoning score from tests pupils took when entered secondary school. |
| `attain`: | attainment score of pupils at age 16. |
| `pid`: | primary school identifying code. |
| `sex`: | pupil's gender (boy=0, girl=1). |
| `sc`: | pupil's social class scale (continuous score from low to high social class) |
| `sid`: | secondary school identifying code |

In these data children are cross-classified by the secondary and primary schools that they attended. The secondary and primary schools are indexed by *j* and *k* respectively. Two models are considered here. The first one is a cross-classified model without any covariate and the second includes `sex` as a covariate in its fixed part. The second model is:

$$y_{i(jk)} = \beta_0 + \beta_1 x_{1ij} + v_{0k} + u_{0j} + e_{i(jk)}$$

$$v_{0k} \sim N(0, \sigma^2_{v0}),\ u_{0j} \sim N(0, \sigma^2_{u0}),\ e_{i(jk)} \sim N(0, \sigma^2_e),$$

$$x_{1ij} = \begin{cases} 0 & boy \\ 1 & girl \end{cases}$$

A two-level (in MLwiN terms) model is fitted to these data with a block-diagonal variance-covariance matrix where the blocks corresponds to the primary and the secondary school effects in this case. The variance-covariance matrix of the random effects can be expressed as: $\begin{bmatrix} \sigma^2_{v0} I_{(148)} & 0 \\ 0 & \sigma^2_{u0} I_{(19)} \end{bmatrix}$ where $I_{(d)}$ is the identity matrix of dimension *d* (148 for primary and 19 for secondary).

The data are first organised into a `groupedData` object, `XcgroupedData`, as follows.

```
XCgroupedData <- groupedData(attain~sex|cons, data=XC)
```

The grouping factor, `cons`, is a column of ones of length 3,435 which specified that all observations come from the same group. `sex` is specified as the primary  covariate. The syntax for this model is:

```
lme(attain~sex,random=pdBlocked(list(pdIdent(~pid-1),pdIdent(~sid-1))),
  data=XCgroupedData)
```

The special variance-covariance structure entailed by this model is handled by the `pdMat` (positive-definite matrices) feature in the `nlme` library. A combination of two `pdMat` classes, `pdBlocked` and `pdIdent`, is required in the random part of the modelling syntax. The former specified that the variance-covariance matrix is block-diagonal and the latter specified its elements are multiples of the identity matrix. The "-1" in `pdIdent` is required to remove the constant term.

A number of other built-in `pdMat` classes are available in the `nlme` library and are discussed in detail in section 4.2.2 of Pinheiro and Bates (2000). The results of the two models are shown in Table 8.

## *4.3 Multivariate Normal response models*

The data, GCSEMV, contain GCSE examination scores on a science subject. Two components of the data are considered as the outcome variables: written paper and course work scores. The aim of the following illustrative analysis is to examine the effect of gender on the scores of the two components of the exam scores. There are 1,905 students from 73 schools in England in the data. There are missing values in the two components and they are coded as -1. The five fields in the dataset are:

| | |
|---|---|
| `school`: | school ID |
| `student`: | student ID |
| `girl`: | girl (0 = boy, 1 = girl) |
| `wrtnscr`: | total score of written papers |
| `crwkscr`: | total score of course work |

The multiple response model for the *j*-th student in the *k*-th school is:

$$y_{1jk} = \beta_0 + \beta_1 x_{jk} + v_{1k} + u_{1jk}$$

$$y_{2jk} = \alpha_0 + \alpha_1 x_{jk} + v_{2k} + u_{2jk}$$

$$\begin{pmatrix} v_{1k} \\ v_{2k} \end{pmatrix} \sim N(0,\Omega_v) : \ \Omega_v = \begin{pmatrix} \sigma^2_{v_1} & \\ \sigma_{v_{12}} & \sigma^2_{v_2} \end{pmatrix}$$

$$\begin{pmatrix} u_{1jk} \\ u_{2jk} \end{pmatrix} \sim N(0,\Omega_u) : \ \Omega_u = \begin{pmatrix} \sigma^2_{u_1} & \\ \sigma_{u_{12}} & \sigma^2_{u_2} \end{pmatrix}.$$

$y_1$ is the written test score and $y_2$ is the coursework score in the model.

There is not any standard function in S-plus for fitting multiple normal responses random effects models. However, `lme` can, with suitable manipulation of the data, be used for fitting such models as if to repeated measures while taking into proper account of the correlated structure between the two responses.

To analyse such data in `lme` the multiple responses must first be stacked into a single column while the covariates are duplicated within the grouping variable which is `student` in this case. For example, the data of two students in School 1 are:

```
school student girl wrtnscr crwkscr
 1         1     0      23    20.1
 1         2     1      65    71.2
```

This is often referred to as the wide format where there is a row of data per lowest lever unit (or student in this case). Stacking the multiple responses turns the data into a long format where there are multiple rows per student. For example, after stacking, the data of the two students with a few new variables are as follows.

```
school student girl  score    wtn   cwk  I(wtn*girl) I(cwk*girl)
 1        1     0      23     1     0       0          0
 1        1     0      20.1   0     1       0          0
 1        2     1      65     1     0       1          0
 1        2     1      71.2   0     1       0          1
```

Dummy version of the covariates have been created to match up with the right responses in the new response variable, `score`. The dummy variables for the constant terms and `girl` with exam component terms are `wtn`, `cwk`, `I(wtn*girl)` and `I(cwk*girl)` respectively. The long-formatted data are saved as `gcselong`.

The syntax for fitting this model is:

```
lme(score~-1+wtn+cwk+I(wtn*girl)+I(cwk*girl),
  random=~-1+wtn+cwk|school, weights=varIdent(form=~1|wtn),
  correlations=corCompSymm(form=~1|school/student), na.action=na.omit,
  data=gcselong)
```

A three-level model is used to fit these data while a new level one is created as the "pseudo" level one. The word pseudo is used because this level is not a genuine hierarchical level of the data but a "level" created only to make it possible to model the complex variation between the observations within a student. This is, however, regarded as a two-level model in S-plus terms as S-plus does not consider the multiple responses within a student as a grouping level. S-plus refers to it as level zero.

The complex level 1 (or within-group in S-plus terms) variation is handled by the optional `weights` which allows it to be modelled as a function of covariate(s). By specifying `weights= varIdent(form=~1|wtn)`, different within-group (or pseudo level 1 in MLwiN terms) variances at each level of the stratifying variable, `wtn`, are allowed. Because `wtn` and `cwk` are mutually exclusive terms, one could have used `cwk` in the syntax and got exactly the same results.

The compound symmetric (i.e. all off-diagonal elements being equal to the same value) structure of the level 1 correlation is specified by `correlations=corCompSymm(form=~1|school/student)`in the syntax. The `corStruct` class, `corCompSymm`, specifies the compound symmetric structure. The right hand side of the `|` operator specifies the grouping of the data in which students are nested within schools. Results of the two models explored in this section are shown in Table 9.

## *4.4 Models for Meta-analysis*

The data META are used to illustrate how to perform a meta-analysis in S-plus. Only aggregate level data are available and they consist of a collection of 19 published studies on the effect of teacher expectancy on pupil IQ (Bryk and Raudenbush (1992)). The four fields are in the dataset are:

(1) Effect size estimate
(2) Variance of effect size estimate
(3) Weeks of prior contact
(4) Study code

The type of model being considered can be written as:

$$y_{\bullet j} = (X\beta)_{\bullet j} + u_j + e_{\bullet j}$$

$$u_j \sim N(0, \sigma_u^2),\ e_{\bullet j} \sim N\left(0, {\sigma_{e_j}^2}\big/{n_j}\right),$$

where the $\bullet$ denotes the mean or some summary measure of the corresponding variable, e.g. mean of $y_{ij}$ over the $i$'s within study $j$. $\sigma_u^2$ is the between-study variance to be estimated and $\sigma_{e_j}^2$ is within-study between-subject variance for the $j$-th study and is assumed known from the study. Results of this model are shown in Table 10.

## *4.5 Other models*

Other models that can be handled by existing S-plus functions or functions in user-contributed libraries but have not been explored in this review are briefly described in this section.

### 4.5.1 Survival models

Extensive facilities written by Terry Therneau[i] (Mayo Foundation) implementing parametric and nonparametric techniques in survival analysis are available in S-plus for Windows version 6.0 and above. Nonparametric methods like Kaplan-Meier estimates of survival and Cox proportion hazards models are performed by the functions `survfit` and `coxph` respectively. Survival models are not explored in this review. Interested readers are referred to Chapter 13 of Venables and Ripley (2002) and Chapters 28 to 33 of the S-plus online manual Guide to Statistics, Vol. 2.

### 4.5.2 Structural equation models and factor analysis

There is currently no function in S-plus for fitting structural equation models. A SEM package by John Fox[ii] is nonetheless available in R (see Packages on the CRAN website: http://cran.r-project.org).

Factor analysis in S-plus is performed using the function `factanal`. Two main techniques for estimating the factors, namely the principle factor estimate and maximum likelihood estimate, along with a variety of choice of rotations, e.g. varimax, oblimin, etc., are available in `factanal`. Interested readers are referred to Chapter 21 of the S-plus online manual Guide to Statistics, Vol. 2 (Insightful (2001)).

### 4.5.3 Nonlinear models

Nonlinear models are handled by the command `nlme` of the similarly named library, `nlme3` (Pinheiro and Bates (2000)). The number three being the latest version of this library. This library is part of the current version of S-plus. The syntaxes used for this command are similar to those used for `lme`. Interested users are referred to Pinheiro and Bates (2000).

## 5. Conclusions

S-plus is a versatile piece of software for data manipulation, thanks to the object-orientated nature of the programming language S, graphing and statistical modelling. Multilevel or random-effects modelling in S-plus can be handled in S-plus by primarily the functions `lme` in the `nlme` library and `glme` in the `correlatedData` library for normal and discrete responses. `glme` is essentially an extension to the extensive work done by Pinheiro and Bates for the `nlme` library.

The function, `groupedData`, in the `nlme` library is a well thought-out function for specifying hierarchical data structure for use with graphing and modelling such data. The flexibility offered by this library through the facilities (or classes in S-plus terminology) `pdMat` (for specifying special forms for the random-effects variance-covariance matrix), `varFunc` (for modelling heteroscedasticity among groups, e.g. boys and girls) and `corStruct` (for modelling dependencies among within-group (or level one in MLwiN terms) errors) enables S-plus users to fit complex multilevel models without reductionistic compromise.

[i] http://www.mayo.edu/hsr/people/therneau.html
[ii] http://socserv.mcmaster.ca/jfox/

Although there is plenty of online documentation for the `nlme` library, users will find the book by Pinhero and Bates (2000) indispensable in fitting linear and nonlinear mixed models using S-plus. A book on the `correlatedData` library for mixed models with discrete outcomes with a comparable degree of depth and clarity is currently not available. However, the reviewer suggests that this might prove equally enlightening for multilevel modellers using S-plus.

In summary, S-plus is very versatile due to its core S programming language and is highly recommended by the reviewer for use in multilevel modelling.

## 6. Acknowledgement

The author would like to thank James Krieger for providing the syntax for constraining level 1 variance to produce the results shown in Table 10.

## 7. Reference

Breslow, N. E. and D. G. Clayton (1993). "Approximate Inference in Generalized Linear Mixed Models." Journal of the American Statistical Association **88**(421): 9-25.

Bryk, A. S. and S. W. Raudenbush (1992). Hierarchical Linear Models. London, Sage Publications.

Chao, E. C. (2003). S-plus Correlated Data Library. Seattle, Insightful Corporation.

Insightful (2001). S-PLUS 6 for Windows Guide to Statistics. Seattle, WA, Insightful Corporation.

Insightful (2001). S-PLUS 6 for Windows User’s Guide. Seattle, WA.

Pinheiro, J. C. and D. M. Bates (2000). Mixed-Effects Models in S and S-PLUS. New York, Springer-Verlag.

Venables, W. N. and B. D. Ripley (2002). Modern Applied Statistics with S - Fourth Edition. New York, Springer-Verlag.

## 8. Tables

**Table 1 Multilevel models that can or cannot be fitted in S-plus version 6.2**

| **Data / model type** | **S-plus (S) function (library)** | **Estimation procedures (see notes)[1]** | **Limitation on group levels in data[2]** | **Allowing covariates** | **Allowing random slopes** | **Weighting** | **Fitting variance function (any level)** | **Other comments/limitations** |
|---|---|---|---|---|---|---|---|---|
| Normal response<br><br>Repeated measures with Normal response<br><br>Cross-classified<br><br>Multivariate Normal<br><br>Time Series<br><br>Nonlinear | lme, nlme (NLME) or glme (correlatedData with family=gaussian) | ML/REML | No | Yes | Yes | Yes | Yes | `Use lmList` to fitted separate regression within each group and for other sub-group explorations. |
| Binary/Binomial<br><br>Poisson | glmmPQL (MASS) | PQL(1) followed by Laplace approximation | 1 | Yes | Yes | Yes | Yes | `glmList` is the equivalent function of `lmList` for discrete outcome models. Dispersion parameter can be pre-specified or modelled. |
| | glme (correlatedData) | (REstricted)PQL, (RE)MQL, AGQUAD and LAPLACE | 2 | Yes | Yes | Yes | Yes | |
| Negative Binomial | glm.nb (MASS) | ML | Arbitrary | Yes | No | ? | ? | |
| Nominal multinomial | multinom (nnet) | ML | *Can't do multilevel* | Yes | No | No | No | |
| Ordered multinomial | polr (MASS) | ML | *Can't do multilevel* | Yes | No | No | No | |
| Multiple membership<br><br>Multiple mixed responses | Not available | | | | | | | |
| Survival | coxph | ML | *Can't do multilevel* | Yes | No | No | No | |
| | survReg | Newton-Raphson | | | | | | |
| Structural Equation models | factanal (for factor analysis) | principle factor solution or maximum likelihood estimate | *Can't do multilevel* | No | No | No | No | |
| Generalized Estimating Equations models (GEE) | gee (correlatedData) | QL | Arbitrary | Yes | No | Yes | Yes | |

Notes:
1. (RE)ML: (restricted) maximum likelihood; (RE)P/MQL: (Restricted) Penalized/Marginal quasi-likelihood; AGQUAD: adaptive Guassian quadrature; LAPLACE: Laplace approximation to the likelihood

2. These are group-level in S-plus terms. That is, pupils in classes nested within schools implies a two group-level model in S-plus. Pupil is not considered as a grouping level.
?: not clear from documentation

**Table 2 S-plus tools for inference and diagnostics**

| Models that can be fitted | Tests for nested models[1] | Inference on fixed effects, (i.e. tests, CI) | Inference on random effects (i.e. tests, CI) | Diagnostics (give list) | Other specific features |
|---|---|---|---|---|---|
| Normal response fitted by `lme` or `nlme` | AIC, BIC, log-likehood | t-test, CI | LR test, CI | `plots.lme` for residual plots (verus fitted values), boxplots of residual at any higher levels, normal plots | Residuals at different levels can be extracted using `random.effects` on a fitted model object |
| Generalised Linear model fitted by `glme` | AIC, BIC, (approx.) log-likehood | t-test, CI | LR test, CI | most tools available to `lme` and `nlme` are also available to `glme` | Residuals at different levels can be extracted using `random.effects()` on a fitted model object |

Notes:
1. AIC (Akaike Information Criterion) = -2logL + 2$n_{param}$,
BIC (Bayesian Information Criterion) = -2logL + $n_{param}$ log(N)
where $n_{par}$ is the number of parameters in the model

**Table 3 Parameter estimates and S-plus syntaxes for two-level Normal models (using *lme*)**

| Model | Fixed parameter estimate | Random effect estimate | REML estimates | (95% C.I.) | Syntax specifications to set up models | -2*logL | Secs. to convergence |
|---|---|---|---|---|---|---|---|
| Variance components without covariates | constant ( $\beta_0$ ) | | -0.013 | (-0.119, 0.093) | `# create data structure`<br>`m1.0 <-`<br>`groupedData(normexam~1\|Schoo`<br>`l,`<br>`data=EXAM, FUN=mean,`<br>`labels= list(x="constant",`<br>`y="Normalised exam score"),`<br>`units=list(x="", y="score`<br>`point"))`<br>`# Fit model 1.0`<br>`fm1.0 <- lme(m1.0)` | 11014.65 | 0.5 |
| | | school ( $\sigma_u$ ) | 0.414 | (0.342, 0.502) | | | |
| | | residual ( $\sigma_e$ ) | 0.921 | (0.901, 0.941) | | | |
| Variance component with covariates 'standlrt', 'gender' and 'schgend' [i] | constant ( $\beta_0$ ) | | -0.009 | (-0.162, 0.143) | `# set first levels of factors`<br>`as baseline`<br>`options(contrasts=c(factor="con`<br>`tr.treatment",ordered="contr`<br>`.poly"))`<br>`fm1.1 <-`<br>`update(fm1.0 ,fixed =`<br>`normexam ~`<br>`standlrt+gender+schgend )` | 9347.67 | 0.5 |
| | standlrt ( $\beta_1$ ) | | 0.560 | (0.535, 0.584) | | | |
| | gender ( $\beta_2$ ) | | 0.167 | (0.101, 0.234) | | | |
| | schgend1 ( $\beta_3$ ) | | -0.159 | (-0.338, 0.020) | | | |
| | schgend2 ( $\beta_4$ ) | | 0.019 | (-0.233, 0.271) | | | |
| | | school ( $\sigma_u$ ) | 0.293 | (0.239, 0.359) | | | |
| | | residual ( $\sigma_e$ ) | 0.750 | (0.734, 0.767) | | | |
| Variance component with interaction, 'gender' * 'standlrt' | constant ( $\beta_0$ ) | | -0.009 | (-0.162, 0.143) | `fm1.2 <-`<br>`update(fm1.1,fixed =`<br>`normexam ~`<br>`standlrt*gender+schgend )` | 9353.20 | 0.6 |
| | standlrt ( $\beta_1$ ) | | 0.563 | (0.527, 0.599) | | | |
| | gender ( $\beta_2$ ) | | 0.167 | (0.100, 0.234) | | | |
| | schgend1 ( $\beta_3$ ) | | -0.159 | (-0.338, 0.020) | | | |
| | schgend2 ( $\beta_4$ ) | | 0.019 | (-0.233, 0.271) | | | |
| | standlrt*gender ( $\beta_5$ ) | | -0.005 | (-0.053, 0.043) | | | |
| | | school ( $\sigma_u$ ) | 0.293 | (0.239, 0.359) | | | |
| | | residual ( $\sigma_e$ ) | 0.750 | (0.734, 0.767) | | | |
| Random slopes on | constant ( $\beta_0$ ) | | -0.012 | (-0.158, 0.134) | `fm1.3 <-` | 9308.24 | 0.8 |
| | standlrt ( $\beta_1$ ) | | 0.550 | (0.500, 0.600) | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 'standlrt' | gender ( $\beta_2$ ) | | 0.169 | (0.102, 0.235) | `update(fm1.2, random=~standlrt\|School)` | | |
| | schgend1 ( $\beta_3$ ) | | -0.178 | (-0.342, -0.014) | | | |
| | schgend2 ( $\beta_4$ ) | | -0.0004 | (-0.233, 0.232) | | | |
| | standlrt*gender ( $\beta_5$ ) | | 0.007 | (-0.051, 0.065) | | | |
| | | school ( $\sigma_{u0}$ ) | 0.289 | (0.236, 0.355) | | | |
| | | standlrt ( $\sigma_{u1}$ ) | 0.123 | (0.091, 0.167) | | | |
| | | school.standlrt ( $\sigma_{u01}$ ) | 0.575 | (0.238, 0.788) | | | |
| | | residual ( $\sigma_e$ ) | 0.742 | (0.726, 0.758) | | | |
| Level 1 variances by 'gender' | constant ( $\beta_0$ ) | | -0.012 | (-0.157, 0.133) | `fm1.4 <- update(fm1.3, weights=varIdent(form=~1\|gender))` | 9302.21 | 2.9 |
| | standlrt ( $\beta_1$ ) | | 0.550 | (0.499, 0.602) | | | |
| | gender ( $\beta_2$ ) | | 0.169 | (0.102, 0.235) | | | |
| | schgend1 ( $\beta_3$ ) | | -0.178 | (-0.341, -0.015) | | | |
| | schgend2 ( $\beta_4$ ) | | -0.0005 | (-0.234, 0.233) | | | |
| | standlrt*gender ( $\beta_5$ ) | | 0.007 | (-0.051, 0.065) | | | |
| | Level 2 | school ( $\sigma_{u0}$ ) | 0.289 | (0.236, 0.355) | | | |
| | | standlrt ( $\sigma_{u1}$ ) | 0.125 | (0.092, 0.168) | | | |
| | | school.standlrt ( $\rho_{u01}$ ) | 0.573 | (0.237, 0.787) | | | |
| | Level 1 | Boy ( $\sigma_b$ ) | 0.767 | (0.741, 0.794) | | | |
| | | Girl ( $\sigma_g$ ) [ii] | 0.725 | (0.700, 0.751) | | | |
| Level 1 variance as function of 'standlrt' | constant ( $\beta_0$ ) | | -0.012 | (-0.157, 0.134) | `fm1.5 <- update(fm1.4, weights=varConstPower(form=~standlrt,fixed=list(const=1)))` | 9308.24 | 2.9 |
| | standlrt ( $\beta_1$ ) | | 0.550 | (0.500, 0.601) | | | |
| | gender ( $\beta_2$ ) | | 0.169 | (0.102, 0.235) | | | |
| | schgend1 ( $\beta_3$ ) | | -0.178 | (-0.342, -0.014) | | | |
| | schgend2 ( $\beta_4$ ) | | -0.0004 | (-0.233, 0.232) | | | |
| | standlrt*gender ( $\beta_5$ ) | | 0.007 | (-0.051, 0.065) | | | |
| | Level 2 | school ( $\sigma_{u0}$ ) | 0.289 | (0.236, 0.354) | | | |
| | | standlrt ( $\sigma_{u1}$ ) | 0.123 | (0.091, 0.167) | | | |
| | | school.standlrt ( $\rho_{u01}$ ) | 0.574 | (0.237, 0.787) | | | |
| | Level 1[iii] | $\sigma_e$ | 0.371 | (0.359, 0.384) | | | |
| | | $\delta_2$ | 0.0009 | (-0.078, 0.080) | | | |

Notes:
i. The variable ‘schgend’ is fitted in the models as two dummy variables. ‘schgend1’ is 1 for mixed schools and 0 otherwise, while ‘schgend2’ is 1 for boys’ school and 0 otherwise.
ii. Point estimate and its 95% C.I. are calculated by multiplying the estimates of the level 1 standard deviation for boys by a ratio provided under Variance function in the `lme` output. In the documentation of `lme`, the ratio is said to between two variances but in fact it is between two SDs. The `lme` output says `“Different standard deviations per stratum”` and it is the correct interpretation.
iii. The class of variance functions in the `nlme` library used here to specify the within-group variance is `varConstPower`. The level 1 variance is modelled as $Var(e_{ij}) = \sigma_e^2(\delta_1 + |\upsilon_{ij}|^{\delta_2})^2$ where ‘standlrt’ is the variance covariate. $\delta_1$ is restricted to be positive but is held fixed at 1 (by “`fixed=list(const=1)`”) in this example. $\delta_2$ is unrestricted and is estimated in this case. It can take any value in the real line allowing the variance to increase as `standlrt` moves away from zero.

**Table 4 Parameter estimates and S-plus syntaxes for three-level Normal models (using *lme*)**

| Model | Fixed parameter estimate | Random effect Ests | REML estimates | (95% C.I.) | Syntax specifications to set up models | -2*logL | Secs. to converg ence |
|---|---|---|---|---|---|---|---|
| Variance component without covariate | constant ( $\beta_0$ ) | | 5.319 | (5.205, 5.433) | `# create data structure`<br>`m3.2.1 <- groupedData(points~1\|Lea/School,`<br>`   data=CHEM97, FUN=max,`<br>`   labels=list(x="constant", y="A-lev chem`<br>`   point score"),`<br>`   units=list(x="", y="score point") )`<br><br>`# Model: var comp with no covariate`<br>`fm2.1 <- lme(m2.1)` | 157873.8 | 3.3 |
| | | lea ( $\sigma_{v0}$ ) | 0.392 | (0.278, 0.553) | | | |
| | | school ( $\sigma_{u0}$ ) | 1.658 | (1.589, 1.729) | | | |
| | | pupil ( $\sigma_e$ ) | 2.918 | (2.894, 2.942) | | | |
| Variance component with the only covariate average GCSE (centred) score | constant ( $\beta_0$ ) | | 5.635 | (5.574, 5.697) | `fm3.2.2 <- update(fm3.2.1, fixed= points~`<br>`   avegcse)` | 141697 | 3.5 |
| | GCSE centred ( $\beta_1$ ) | | 2.473 | (2.439, 2.506) | | | |
| | | lea ( $\sigma_{v0}$ ) | 0.123 | (0.049, 0.305) | | | |
| | | school ( $\sigma_{u0}$ ) | 1.080 | (1.029, 1.133) | | | |
| | | pupil ( $\sigma_e$ ) | 2.270 | (2.252, 2.289) | | | |

**Table 5 Parameter estimates and S-plus syntaxes for two-level logistic model (using *glmmPQL*)**

| Model | Fixed effect parameter | Random effect parameter | Logit link | | Probit link | | Syntax specifications to set up models |
|---|---|---|---|---|---|---|---|
| Variance component with all covariates | | | REML estimates | (95% C.I.) | REML estimates | (95% C.I.) | `# declare ‘c’ as a factor and specify what type of contrast to use`<br>`BANG$c <- factor(BANG$c)`<br>`options(contrasts=c(factor="contr.treatment",ordered="contr.poly"))`<br><br>`# for logit link`<br>`glmmPQL(use~urban+cage+c, random=~1\|district, family=binomial(link=logit), data=bang)`<br><br>* replace ‘logit’ with ‘probit’ for probit link |
| | constant ( $\beta_0$ ) | | -1.661 | (-1.941, -1.380) | -1.020 | (-1.187, -0.855) | |
| | urban ( $\beta_1$ ) | | 0.719 | (0.491, 0.948) | 0.446 | (0.306, 0.585) | |
| | centred age ( $\beta_2$ ) | | -0.026 | (-0.041, -0.011) | -0.016 | (-0.025, -0.007) | |
| | c2 ( $\beta_3$ ) | | 1.092 | (0.790, 1.394) | 0.665 | (0.483, 0.848) | |
| | c3 ( $\beta_4$ ) | | 1.355 | (1.021, 1.689) | 0.829 | (0.627, 1.030) | |
| | c4 ( $\beta_5$ ) | | 1.324 | (0.982, 1.667) | 0.809 | (0.603, 1.015) | |
| | | district ( $\sigma_{u0}$ ) | 0.457 | (0.331, 0.631) | 0.280 | (0.204, 0.386) | |
| | Dispersion ( $\varphi$ ) | | 0.984 | (0.953, 1.016) | 0.985 | (0.954, 1.017) | |
| | ***-2logL***[i] | | 8489.0 | | 6554.5 | | |
| | Secs to convergence | | 2.2 | | 2.4 | | |
| Random effect on Urban | | | | | | | `glmmPQL(use~urban+cage+c,random=~ urban\|district, family= binomial(link=logit), data=bang)`<br><br>* replace ‘logit’ with ‘probit’ for probit link |
| | constant ( $\beta_0$ ) | | -1.667 | (-1.968, -1.365) | -1.029 | (-1.209, -0.850) | |
| | urban ( $\beta_1$ ) | | 0.792 | (0.470, 1.114) | 0.494 | (0.296, 0.692) | |
| | centred age ( $\beta_2$ ) | | -0.026 | (-0.041, -0.011) | -0.016 | (-0.025, -0.007) | |
| | c2 ( $\beta_3$ ) | | 1.099 | (0.796, 1.401) | 0.674 | (0.491, 0.856) | |
| | c3 ( $\beta_4$ ) | | 1.334 | (1.000, 1.668) | 0.821 | (0.619, 1.023) | |
| | c4 ( $\beta_5$ ) | | 1.323 | (0.979, 1.666) | 0.815 | (0.609, 1.022) | |
| | | district ( $\sigma_{u0}$ ) | 0.609 | (0.447, 0.830) | 0.373 | (0.275, 0.507) | |
| | | urban ( $\sigma_{u1}$ ) | 0.799 | (0.509, 1.255) | 0.493 | (0.313, 0.774) | |
| | | district.urban ( $\rho_{u01}$ ) | -0.795 | (-0.933, -0.456) | -0.796 | (-0.933, -0.458) | |
| | Dispersion ( $\varphi$ ) | | 0.976 | (0.945, 1.008) | 0.977 | (0.946, 1.009) | |
| | ***-2logL*** | | 8519.5 | | 6576.1 | | |
| | Secs. to convergence | | 4.3 | | 4.3 | | |

Notes:

i. The –2logL value is an approximate as pseudo-quasi likelihood is used in the estimation.

**Table 6 Parameter estimates and S-plus syntaxes for multilevel Poisson models (using *glme*)**

| Model | Fixed parameter estimate | Random effect Ests | REML estimates | (95% C.I.) | Syntax specifications to set up models | -2*logL (approx.) | Secs. to convergence |
|---|---|---|---|---|---|---|---|
| Two-level variance component model with 'uvb' as the only covariate | intercept ( $\beta_0$ ) | | -0.131 | (-0.229, -0.033) | `glme(deaths~offset(lnexp)+uvb,`<br>`random=~1\|region,`<br>`family=poisson(link=log), data=mmmec,`<br>`dispersion=1)` | 341.726 | 0.8 |
| | uvb ( $\beta_1$ ) | | -0.034 | (-0.053, -0.015) | | | |
| | Level 2 | Intercept ( $\sigma_u$ ) | 0.416 | (0.347, 0.498) | | | |
| Two-level variance component model with 'uvb' as the only covariate (allowing dispersion) | intercept ( $\beta_0$ ) | | -0.129 | (-0.226, -0.315) | `glme(deaths~offset(lnexp)+uvb,`<br>`random=~1\|region,`<br>`family=poisson(link=log), data=mmmec)` | 329.576 | 0.7 |
| | uvb ( $\beta_1$ ) | | -0.038 | (-0.057, -0.018) | | | |
| | Level 2 | Intercept ( $\sigma_u$ ) | 0.408 | (0.339, 0.490) | | | |
| | Level 1 | Dispersion ( $\varphi$ ) | 1.135 | (1.044, 1.234) | | | |
| Three-level variance component model with 'uvb' as the only covariate | intercept ( $\beta_0$ ) | | -0.057 | (-0.335, 0.222) | `glme(deaths~offset(lnexp)+uvb,`<br>`random=~1\|nation/region,`<br>`family=poisson(link=log), data=mmmec,`<br>`dispersion=1)` | 283.537 | 1.4 |
| | uvb ( $\beta_1$ ) | | -0.028 | (-0.050, -0.006) | | | |
| | Level 3 | Intercept ( $\sigma_v$ ) | 0.395 | (0.230, 0.679) | | | |
| | Level 2 | Intercept ( $\sigma_u$ ) | 0.221 | (0.177, 0.276) | | | |
| Three-level variance component model with 'uvb' as the only covariate (allowing dispersion) | intercept ( $\beta_0$ ) | | -0.061 | (-0.338, 0.215) | `glme(deaths~offset(lnexp)+uvb,`<br>`random=~1\|nation/region,`<br>`family=poisson(link=log), data=mmmec)` | 272.404 | 1.4 |
| | uvb ( $\beta_1$ ) | | -0.031 | (-0.054, -0.008) | | | |
| | Level 3 | Intercept ( $\sigma_v$ ) | 0.391 | (0.227, 0.672) | | | |
| | Level 2 | Intercept ( $\sigma_u$ ) | 0.212 | (0.167, 0.268) | | | |
| | Level 1 | Dispersion ( $\varphi$ ) | 1.133 | (1.043, 1.231) | | | |

**Table 7 Parameter estimates and S-plus syntaxes for repeated measurement normal ‘growth’ model (using *lme*)**

| Model | Fixed parameter estimate | Random effect Ests | REML estimates | (95% C.I.) | Syntax specifications to set up models | -2*logL | Secs. to converg ence |
|---|---|---|---|---|---|---|---|
| Polynomi al growth model (upto quartic term) with random coeff. (upto quadratic) without time series | intercept ( $\beta_0$ ) | | 148.98 | (145.88, 152.07) | `m5.1 <- groupedData(height~cage\|id,data=oxboys ,  FUN= mean, labels=list(x="centred age", y="height"), units=list(x="",y="(cm)"))` `fm5.1 <- lme(m5.1,fixed=height~cage+cage2+cage3 +cage4,random=~cage+cage2\|id)` | 629.825 | 0.8 |
| | cage ( $\beta_1$ ) | | 6.166 | (5.461, 6.870) | | | |
| | cage2 ( $\beta_2$ ) | | 1.091 | (0.396, 1.786) | | | |
| | cage3 ( $\beta_3$ ) | | 0.468 | (0.145, 0.790) | | | |
| | cage4 ( $\beta_4$ ) | | -0.340 | (-0.936, 0.255) | | | |
| | Level 2 | intercept ( $\sigma_{u0}$ ) | 8.001 | (6.062, 10.559) | | | |
| | | cage ( $\sigma_{u1}$ ) | 1.696 | (1.278, 2.251) | | | |
| | | cage2 ( $\sigma_{u2}$ ) | 0.813 | (0.571, 1.157) | | | |
| | | int.cage ( $\rho_{01}$ ) | 0.613 | (0.305, 0.805) | | | |
| | | int.cage2 ( $\rho_{02}$ ) | 0.218 | (-0.221, 0.583) | | | |
| | | cage.cage2 ( $\rho_{12}$ ) | 0.660 | (0.289, 0.859) | | | |
| | Level 1 | residual ( $\sigma_e$ ) | 0.469 | (0.420, 0.525) | | | |
| Same model as above but with auto-correlatio n structure (AR1) | intercept ( $\beta_0$ ) | | 148.87 | (145.78, 151.97) | `# create the sine and cosine functions` `oxboys$sinseas=sin(pi*oxboys$season/6)` `oxboys$cosseas=cos(pi*oxboys$season/6)` `fm5.2 <- update(fm5.1, fixed=height~cage+cage2+cage3+cage4+ sinseas+ cosseas, cor=corAR1(),data=oxboys)` | 623.554 | 2.6 |
| | cage ( $\beta_1$ ) | | 6.174 | (5.461, 6.887) | | | |
| | cage2 ( $\beta_2$ ) | | 2.061 | (1.149, 2.973) | | | |
| | cage3 ( $\beta_3$ ) | | 0.406 | (0.051, 0.760) | | | |
| | cage4 ( $\beta_4$ ) | | -1.432 | (-2.310, -0.555) | | | |
| | sinseas () | | 0.015 | (-0.088, 0.118) | | | |
| | cosseas () | | -0.223 | (-0.356, -0.090) | | | |
| | Level 2 | intercept ( $\sigma_{u0}$ ) | 7.992 | (6.055, 10.550) | | | |
| | | cage ( $\sigma_{u1}$ ) | 1.665 | (1.250, 2.216) | | | |
| | | cage2 ( $\sigma_{u2}$ ) | 0.764 | (0.513, 1.140) | | | |
| | | int.cage ( $\rho_{01}$ ) | 0.618 | (0.308, 0.809) | | | |
| | | int.cage2 ( $\rho_{02}$ ) | 0.261 | (-0.206, 0.632) | | | |
| | | cage.cage2 ( $\rho_{12}$ ) | 0.702 | (0.285, 0.896) | | | |
| | Level 1 | residual ( $\sigma_e$ ) | 0.507 | (0.443, 0.580) | | | |
| | | AR(1) corr. structure ( $\phi$ ) | 0.239 | (0.115, 0.355) | | | |

**Table 8 Parameter estimates and S-plus syntaxes for cross-classification model (using *lme*)**

| Model | Fixed parameter estimate | Random effect Ests | REML estimates | (95% C.I.) | Syntax specifications to set up models | -2*logL | Secs. to convergence |
|---|---|---|---|---|---|---|---|
| Cross-classified model without covariate in the fixed part | intercept ( $\beta_0$ ) | | 5.502 | (5.151, 5.852) | `XCgroupedData <- groupedData(attain~sex|cons, data=XC)` <br> `lme(attain~1,random=pdBlocked(list(pdIdent(~pid-1),pdIdent(~sid-1))), data=XCgroupedData)` <br> (Note: it is essential to declare ‘pid’ and ‘sid’ as factors beforehand) | 17150.76 | 30.9 |
| | Level 2 | secondary ( $\sigma_{u_j}$ ) | 0.611 | (0.384, 0.973) | | | |
| | | primary ( $\sigma_{u_k}$ ) | 1.064 | (0.865, 1.307) | | | |
| | Level 1 | residual ( $\sigma_e$ ) | 2.848 | (2.780, 2.918) | | | |
| Same model with ‘sex’ as a covariate in the fixed part | intercept ( $\beta_0$ ) | | 5.255 | (4.894, 5.617) | `lme(attain~sex,random=pdBlocked(list(pdIdent(~pid-1),pdIdent(~sid-1))),data=XCgroupedData)` | 17127.91 | 31.4 |
| | sex ( $\beta_1$ ) | | 0.499 | (0.306, 0.691) | | | |
| | Level 2 | secondary ( $\sigma_{u_j}$ ) | 0.608 | (0.386, 0.958) | | | |
| | | primary ( $\sigma_{u_k}$ ) | 1.053 | (0.904, 1.228) | | | |
| | Level 1 | residual ( $\sigma_e$ ) | 2.838 | (2.770, 2.908) | | | |

**Table 9 Parameter estimates and S-plus syntaxes for multivariate Normal response model (using *lme*)**

| Model | Fixed parameter estimate | Random effect Ests | REML estimates | (95% C.I.) | Syntax specifications to set up models | -2*logL | Secs. to convergence |
|---|---|---|---|---|---|---|---|
| Bivariate normal response model with ‘girl’ as the only covariate | intercept ( $\beta_0$ ) | | 49.009 | (47.172, 50.847) | `lme(score~-1+wtn+cwk+I(wtn*girl)+I(cwk*girl), random=~-1+wtn+cwk\|school, weights=varIdent(form=~1\|wtn), corr=corCompSymm(form=~1\|school/student ), na.action=na.omit, data=gcselong)`[ii] | 26794.6 | 11.3 |
| | ( $\alpha_0$ ) | | 69.622 | (67.313, 71.931) | | | |
| | girl ( $\beta_1$ ) | | -2.492 | (-3.591, -1.392) | | | |
| | ( $\alpha_1$ ) | | 6.757 | (5.442, 8.073) | | | |
| | Level 2 (school) | wtn ( $\sigma_{v_1}$ ) | 6.877 | (5.644, 8.378) | | | |
| | | cwk ( $\sigma_{v_2}$ ) | 8.727 | (7.204, 10.572) | | | |
| | | wtn.cwk ( $\rho_{v_{12}}$ ) | 0.423 | (0.178, 0.618) | | | |
| | Level 1 (student) | wtn ( $\sigma_{v_1}$ ) | 11.160 | (10.794, 11.539) | | | |
| | | cwk ( $\sigma_{v_2}$ )[i] | 13.422 | (12.981, 13.878) | | | |
| | | wtn.cwk ( $\rho_{v_{12}}$ ) | 0.486 | (0.456, 0.514) | | | |

Notes:

i) Estimates of $\sigma_{v_2}$ are derived from multiplying those of $\sigma_{v_1}$ by the parameter of the variance function (see documentation on `varIdent`) of 1.203.

ii) The I( ) is the syntax is necessary for creating the indicator variables for the two responses. The first one takes the value 1 when the response is a written paper score and the student is a girl. The latter does so when the response is the course work score and the student is a girl.

**Table 10 Parameter estimates and S-plus syntaxes for meta-analysis (using *lme*)**

| Model | Fixed parameter estimate | Random effect Ests | REML estimates | (95% C.I.) | Syntax specifications to set up models | -2*logL | Secs. to convergence |
|---|---|---|---|---|---|---|---|
| Model with ‘weeks’ as the only covariate | $\beta_0$ | | 0.409 | (0.225, 0.592) | `lme(fixed=d~wks, data=meta, random=~1\|study, weights=varFixed(~Vofd), control=lmeControl(sigma=1))` | -2.679 | 1.0 |
| | $\beta_1$ | | -0.158 | (-0.234, -0.082) | | | |
| | Level 2 | $\sigma_u$ | 0.000096 | - | | | |
| | Level 1 | $\sigma_u$ | 1.0 | - | | | |

Note: Level 1 residual has been constrained to one for correct model specification. Standard error cannot be estimated for level 2 variance because of non-positive definite approximate variance-covariance.