XLISP-Stat tools for building Generalised Estimating Equation models

Thomas Lumley
Department of Biostatistics
University of Washington
Box 357232
Seattle WA 98195–7232

December 8, 1996

Abstract

This paper describes a set of Lisp-Stat tools for building Generalised Estimating Equation models to analyse longitudinal or clustered measurements. The user interface is based on the built-in regression and generalised linear model prototypes, with the addition of object-based error functions, correlation structures and model formula tools. Residual and deletion diagnostic plots are available on the cluster and observation level and use the dynamic graphics capabilities of Lisp-Stat.

1 Introduction

Generalised Estimating Equations models, proposed by Liang and Zeger in 1986, are probably the simplest method for analysing data collected in groups where observations within a group may be correlated but observations in separate groups are independent. A complete description of the method is given in their two 1986 papers. The basic principle of the method is a generalisation of the fact that weighted least squares analyses give unbiased parameter estimates no matter what weights are used. Generalised linear models, such as logistic regression, have similar robustness properties, giving asymptotically correct parameter estimates even when the data are correlated. This means that it is possible to estimate regression parameters using any convenient or plausible assumptions about the true correlation between observations and get the right answer even when the assumptions are not correct. It is only necessary to use a "model-robust" or "agnostic" estimate of the standard errors. It would be unreasonable to expect this freedom of choice to be without cost and it turns out that there is a moderate gain in efficiency resulting from choosing a working correlation structure close to the true one.

Useful references include the two original papers (Zeger & Liang 1986, Liang & Zeger 1986) and two recent books: Diggle, Liang & Zeger (1993) and Fahrmeir & Tutz (1995). As far as I know the most elementary treatment anywhere in the literature is still Zeger & Liang (1986).

Section 2 gives an overview of the theory and use of Generalised Estimating Equations. Section 3 describes how to use the Lisp-Stat code, including diagnostics. Finally there is a brief discussion of missing data handling and of other software for fitting GEE models. Appendix A describes some aspects of the implementation, including the global variables (Table 3) that control many program options.

2 An introduction to GEE models

2.1 GEE models in theory

Suppose observations and p covariates (Y_{gm}, X_{gm}) are available in $g = 1 \dots G$ groups with at most $m = 1 \dots M_g \leq M$ in each group, and that the mean and variance of each observation is thought to satisfy a generalised linear model:

$$\begin{aligned} E[Y_{gm}] &= \mu_{gm} = \beta' X_{gm} \\ \mathrm{var}[Y_{gm}] &= V(\mu_{gm}) \end{aligned}$$

but that the observations within a group are correlated. If the variables were uncorrelated the (quasi)score equations would be:

$$\begin{aligned} \sum_{g=1}^{G} \sum_{m=1}^{M_g} \frac{\partial \mu_{gm}}{\partial \beta} V^{-1}(\mu_{gm})[Y_{gm}] &= 0 \\ &\text{or} \\ \sum_{g=1}^{G} D_g' V_g^{-1} S_g &= 0 \end{aligned}$$

where D_g is $M_g \times p$ matrix of derivatives, V_g is a diagonal $M_g \times M_g$ matrix of variances and S_g is the vector of the M_g residuals $Y_{gi} - \mu_{gi}$.

The left-hand side of this equation has zero expectation at the true value of β even if there is correlation within groups, so the estimates of β will be consistent even for correlated data. In this equation the changing variance of $Y - \mu$ is accounted for in the weighting by V^{-1} but no account is made for the correlations. The consistency results still hold if V_g is replaced by a covariance matrix for group g, either prespecified or depending on a finite set of parameters. If this covariance matrix is close to the true covariance of S_g then β will be estimated more efficiently. The assumed form of V_g is called the "working covariance model".

In order for this to be useful it is necessary to have reliable standard error estimates for the estimates $\hat{\beta}$. An argument based on the delta method (or a first-order Taylor series expansion) shows that

$$\hat{\mathrm{cov}}(\hat\beta) = \left(\sum_{g=1}^{G} D_g' V_g^{-1} D_g\right) \left(\sum_{g=1}^{G} D_g' V_g^{-1} S_g S_g' V_g^{-1} D_g\right) \left(\sum_{g=1}^{G} D_g' V_g^{-1} D_g\right),$$

the so-called "sandwich estimator" gives asymptotically correct standard errors and various simulation studies have shown that its properties are good when M is small (5 or fewer). Less is known for large M.

2.2 GEE models in practice

The models are used in the same way as standard generalised linear models, and the coefficients have the same interpretation. They measure differences in the response for a unit change in the predictor, averaged over the whole sample. GEE models are thus particularly suitable when the correlation is of no substantive interest and is merely a nuisance parameter.

The data used for the example in section 3 illustrate a common, but very simple situation. The number of hospital visits for 72 children were recorded, together with the sex of the child and the age and smoking status of the mother. We are interested in whether maternal smoking is related to the number of hospital visits and the obvious analysis is a Poisson regression. The complication is that the number of hospital visits was recorded over four separate time periods. Using these as separate observations violates the independence assumption for Poisson regression; adding them together to get a single outcome is likely to result in overdispersion and loses any information about variation over time.

With a working covariance model of independence the parameter estimates will be the same as from a Poisson regression, but the standard errors will be valid. This is often sufficient, especially when the correlations are not too high and the number of observations on each individual is the same.

Another popular model is the exchangeable correlation model, in which all pairwise correlations between different times are the same. This is analogous to the popular but unreliable method of analysing repeated Normal measurements using split-plot ANOVA. An exchangeable correlation GEE fits the same working modelbut gets asymptotically valid standard error estimates even when the correlations are not truly exchangeable.

It is probably the case that observations close together in time are more similar than those far apart in time. A working covariance model that incorporated this structure might give better estimates. One possible correlation structure for these data would be stationary 3-dependence. This working model estimates three correlation parameters: the correlations at lag 1, lag 2 and lag 3. If there were sufficient data it would even be possible to estimate all 6 correlation parameters. This is known as the "saturated" working model. In addition to possibly greater efficiency these alternative working models provide estimates of the correlations between different time points. It should be noted, however, that these estimates have low efficiency and may be biased if the working model is not close to the truth.

Little is known about good ways of choosing correlation structures. The options provided in this implementation allow reasonable flexibility to approximate a wide range of structures. In the next section these data are analysed in Lisp-Stat using multiple correlation structures and diagnostic methods.

3 How to use the Lisp-Stat code

3.1 Fitting a model

There is a wrapper function `gee-model` which takes the following arguments

`:x`	sequence, list of sequences matrix, or model-formula	predictor variables
`:y`	sequence	response variable
`:g`	sequence	grouping variable
`:times`	sequence	for longitudinal data: observation number or time
`:link`	link object	link function
`:error`	error (variance) object	variance function
`:correlation`	correlation object	working correlation
`:offset`	sequence	offset (default 0)
`:prior-weights`	sequence	prior weights (default 1s)
`:intercept`	logical	include an intercept? (default `t`)
`:verbose`	logical	(default `*gee-verbose*`)
`:print`	logical	display the results? (default `t`)
`:allow-missing`	logical	allow missing observations (only affects some correlations) default `*gee-allow-missing*`
`:init-beta`	sequence	initial guess for β
`:count-limit`	integer	maximum number of iterations (default 30)
`:predictor-names`	list of strings	names

You must specify the first three arguments and at least one of `:link` and `:error`. If only one of `:link` and `:error` is present then the other is set to a default value given in Table 1. If `:correlation` is not specified then the value of the global variable `*gee-default-correlation*` (typically the independence working model) is used. If `:verbose` is true the package will print a message indicating that these default choices have been used.

Note that for binomial data $y = r/n$ it is necessary to use the observed proportion not the observed count as the response and then to specify the denominators using `:prior-weights`. Correlated binomial data (other than binary) are comparatively rare; the individual binary observations are usually available.

The independence and exchangeable correlation structures ignore the observation `:times` argument, for other structures it is always required (in contrast

Table 1: Default link and variance functions

Variance object	$V(\mu)$	Default link
`binomial-error`	$\mu(1-\mu)$	`logit-link`
`poisson-error`	μ	`log-link`
`normal-error`	1	`identity-link`
`gamma-error`	μ^2	`inverse-link`
`(power-error k)`	μ^k	none
Link object	$g(\mu)$	Default variance function
`logit-link`	$\log(\mu/(1-\mu))$	`binomial-error`
`cloglog-link`	$\log(-\log(1-\mu))$	`binomial-error`
`probit-link`	$\Phi^{-1}(\mu)$	`binomial-error`
`log-link`	$\log(\mu)$	`poisson-error`
`identity-link`	μ	`normal-error`
`inverse-link`	$1/\mu$	`gamma-error`
`(power-link k)`	μ^k	none

to other packages). This is a feature, not a bug. The GEE functions written for other general-purpose packages mostly assume the data are sorted so that each group is contiguous and in the right order. In my experience data do not naturally occur that way. If the times column is in the data set it is easy to use it, if it isn't it is hard to be confident that the ordering is correct (and virtually impossible with missing observations). The performance implications of this and some theoretical considerations with missing data are discussed later.

The default pairs of variance and link functions are given in Table 1.

These defaults are stored in the variables `default-link` and `default-error`. Most of these are well-known standards for generalised linear models. The use of power link and variance functions in modelling data with non-Normal, skew error distribution was discussed by Nelder (1994). The link functions are taken from the `glim-proto` system but have had extra methods `:valideta` and `:validmu` added to allow domain checking. Any link-variance combination is allowed and the program will usually find a solution even for badly-fitting combinations.

The available working correlation structures are described in Table 2. The parameter `:allow-missing` controls the assumptions made when the observation times are not the same for every group. Under the exchangeable and independence models it has no effect; under the other models unbalanced data are not allowed when `:allow-missing` is `nil`. When `:allow-missing` is `t` unbalanced data are allowed and the correlation parameters are estimated from the available data.

There is no need for the observation times to be consecutive integers. In fact they need not be integers at all, or even numbers. The requirements are that `equalp` is a valid test for equality and that `rank` will rank them. This means in practice that they must be all numbers or all strings and that if they are string the correct order is alphabetic. Group identifiers may also be either strings or numbers.

Table 2: Working correlation structures.

Object Name	Description
`independence-corr`	Independence
`exchangeable-corr`	All correlations equal
`saturated-corr`	Estimates complete correlation matrix
`saturated-ml-corr`	Estimates complete covariance matrix
`(m-dependence m)`	Non-stationary m-dependence
`(stat-m-dependence m)`	Stationary m-dependence
or `(m-dependence m :stationary t)`	
`AR-1-corr`	first-order autoregressive
`(fixed-corr R)`	known correlation matrix R

The following example uses the data set described in section 2 to fit the stationary 3-dependence model discussed there. All the example code can be found in the file `example.lsp`. The files `geetest.lsp` and `geetestout` give further examples to test the function and compare its results to the S-PLUS gee() function by Carey & Macdermott.

```
>(load "gee.1.0.lsp")
; loading "gee.1.0.lsp"
; loading "/usr/local/lib/xlispstat/Autoload/glim.fsl"
; loading "modelformula.lsp"
T
>(load "karim.lsp")
; loading "karim.lsp"
T
>(def example1 (gee-model :y num-visits :x (list sex smoke age time2
time3 time4) :g id :error poisson-error :correlation (stat-m-dependence 3)
 :times times :predictor-names (list "sex" "smoker" "mother's age"
 "2nd vs 1st" "3rd vs 1st" "4th vs 1st")))
No link specified - canonical link used
Iteration 1: quasideviance = 517.114
Iteration 2: quasideviance = 478.439
Iteration 3: quasideviance = 475.966
Iteration 4: quasideviance = 476.062
Iteration 5: quasideviance = 476.087
Iteration 6: quasideviance = 476.088
Iteration 7: quasideviance = 476.088

GEE Estimates:

                         Coefficient     Std Error           Naive Std Error
Constant                  0.240922      (0.364061)          (0.360940)
sex                      -0.205502      (0.203637)          (0.220660)
smoker                    0.176987      (0.236101)          (0.240601)
```

```
mother's age              3.157128E-3  (6.490476E-3)       (6.210925E-3)
2nd vs 1st               -0.435318     (0.195649)          (0.190492)
3rd vs 1st               -0.307485     (0.202848)          (0.185973)
4th vs 1st                -1.12847     (0.221787)          (0.227792)

Scale Estimate:             1.83565
Independence model deviance:   476.088
Number of cases:                292
Link:                    #<Glim Link Object: LOG-LINK>
Variance function:       Poisson error
Stationary  3 - dependence Working Model:
 correlation matrix =
#2a(
    (  1.0      0.21      0.19      0.34    )
    ( 0.21       1.0      0.21      0.19    )
    ( 0.19      0.21       1.0      0.21    )
    ( 0.34      0.19      0.21       1.0    )
   )
EXAMPLE1
```

Adding `:verbose nil` to the arguments will suppress everything before the line `GEE Estimates`. Adding `:print nil` will suppress everything from that point on.

The function returns a `gee-proto` object which responds to a lot of messages, including many of the messages implemented for generalised linear models. Some description of most of the methods is available using the `:help` method (eg `(send mygeeobject :help :scale)` describes the `:scale` method. To find all the methods which have help omit the topic specifier: `(send gee-proto :help)`.

There is also a method for producing confidence intervals for the coefficients, transformed to any appropriate scale. The message `:conf-interval` gives the estimates and a 95% confidence interval. There are three optional keyword arguments: `:coverage` specifies the desired coverage, `:names` is `nil` to prevent predictor names being returned and `:transform` specifies a function to apply to the confidence interval. As an example we produce produce 90% confidence intervals for the hazard (incidence density) ratio in the example model above

```
>(print-matrix (send example1 :conf-interval :transform #'exp :coverage 0.9))
#2a(
    (Intercept         1.27242       0.699139        2.31579    )
    (sex              0.814238       0.582483        1.13820    )
    (smoker            1.19362       0.809480        1.76004    )
    (mother's age      1.00316       0.992509        1.01393    )
    (2nd vs 1st       0.647059       0.469010       0.892700    )
    (3rd vs 1st       0.735294       0.526692        1.02652    )
```

```
    (4th vs 1st       0.323529       0.224636       0.465959    )
   )
NIL
```

3.2 Model formulae

It is possible to use a model formula object to include factor (class) and interaction terms more easily in a model. The support is less simple than either SAS or S-PLUS and less complete, allowing interactions but not nesting. The function `as-formula` creates formula objects which contain a design matrix `:design-matrix`, a list of column labels `:name-list`, a list of block (eg factor or interaction) names `:block-names` and a list of lists of column numbers specifying each block `:block-indices`. The form of a call to `as-formula` is

```
(as-formula '((factor a) (factor b) (term x) (interaction (list a b))
 (interaction (list a x) :is-factor (list t nil))))
```

where `A` and `B` are factor variables and `X` is a metric (continuous) variable. The keyword argument `:is-factor` to `interaction` indicates whether of the terms in the interaction should be treated as factors. It is optional; the default is to treat them all as factors. The reason for this default is partly that it is more common and partly that making a mistake is more likely to be obvious. Note that the argument to `(as-formula)` must be quoted, as in the example.

The `gee-model` command and `gee-proto` object have some built-in recognition of formula objects, such as the ability to use a formula object as the `:x` argument to `gee-model`. The file `modelformula.lsp` which defines the objects also contains methods for `regression-model-proto` and some more primitive functions which may be useful for creating and manipulating design matrices. In particular the functions `interaction`, `factor` and `term` can be used on their own to create contrast matrices and variable names. More documentation of this is in `modelformula.txt`. The test file contains some further examples.

The main difference caused by using a model formula argument to `gee-model` is that the default output changes to report the Wald chisquare and p-value for each block. This is produced by the message `:display-with-formula :block-only nil`. To get only the test statistics, use the message `:display-with-formula :block-only t`. To get the output as if a model formula had not been given use `:display`. The default value of the `block-only` keyword is controlled by a global variable `*gee-display-block-only*` which is initially `nil`.

The example given above can be replaced by the following code

```
> (def karim-model (as-formula '((term age) (factor gender)
 (factor smoker) (factor times))))
KARIM-MODEL
> (def example2 (gee-model :y num-visits :x karim-model :g id
:error poisson-error :correlation (stat-m-dependence 3) :times times))
 No link specified - canonical link used
Iteration 1: quasideviance = 517.114
```

```
Iteration 2: quasideviance = 478.439
Iteration 3: quasideviance = 475.966
Iteration 4: quasideviance = 476.062
Iteration 5: quasideviance = 476.087
Iteration 6: quasideviance = 476.088
Iteration 7: quasideviance = 476.088

GEE Estimates:

Block              Wald Chisq     p-value
Intercept            0.43793      0.5081
      Variable            Estimate       Std.Err.       p-value
     Intercept             0.24092      (0.364061)     0.5081
TIMES                 26.659      0.0000
      Variable            Estimate       Std.Err.       p-value
     (TIMES 4)             -1.1285      (0.221787)     0.0000
     (TIMES 3)            -0.30748      (0.202848)     0.1296
     (TIMES 2)            -0.43532      (0.195649)     0.0261
SMOKER               0.56194      0.4535
      Variable            Estimate       Std.Err.       p-value
     (SMOKER Y)            0.17699      (0.236101)     0.4535
GENDER                1.0184      0.3129
      Variable            Estimate       Std.Err.       p-value
     (GENDER M)           -0.20550      (0.203637)     0.3129
AGE                  0.23661      0.6267
      Variable            Estimate       Std.Err.       p-value
     AGE                     3.15713E-3  (6.490476E-3)  0.6267

Scale Estimate:            1.83565
Independence model deviance:   476.088
Number of cases:               292
Link:                    #<Glim Link Object: LOG-LINK>
Variance function:       Poisson error
Stationary  3 - dependence Working Model:
 correlation matrix =
#2a(
    (  1.0      0.21      0.19      0.34    )
    ( 0.21       1.0      0.21      0.19    )
    ( 0.19      0.21       1.0      0.21    )
    ( 0.34      0.19      0.21       1.0    )
   )
EXAMPLE2
```

where `GENDER` and `SMOKER` are character versions of the variables `SEX` and `SMOKE` used in the earlier model. The predictor names are automatically generated from the variable names and values. This is particularly useful when the variable names and values are easily interpretable as in this example. As can be seen here, the reference category for the indicator variables is the first one in alphabetic order (more generally in ASCII order).

3.3 Diagnostics

Deletion diagnostics are available as well as a variety of flavours of residual. Current graphical methods are

`:plot-raw-residuals`	$y - \hat{y}$
`:plot-pearson-residuals`	same as the independence case
`:plot-deviance-residuals`	same as the independence case
`:plot-standardised-residuals`	Pearson residuals standardised by
or `:plot-standardized-residuals`	working correlation
`:plot-leverage`	Leverage statistics
`:plot-dbeta`	$\Delta\beta$s
`:plot-cooks-distance`	Cook's distance (influence)

The standardised residuals are the Pearson residuals multiplied by the Cholesky square-root of the inverse of the correlation matrix. They are approximately independent with equal variance if the working model is roughly true.

The last three methods are generalisations of the leverage (diagonal of hat matrix), DBETA (change in coefficient when point is deleted) and Cook's distance (influence) statistics for linear and generalised linear models. The formulae are from Preisser & Qaqish (1996) and are natural generalisations of the independent data versions (though the derivations and calculations are rather more complicated). The methods take a keyword argument `:unit` of 0 for individual observation deletion and 1 for group deletion. Constants `+group+` and `+cluster+`, and `+not-group+` and `+observation+` have been defined to make these easier to use (eg `:unit +group+`). The default is diagnostics for individual observations.

The residual plots and `plot-dbeta` take a keyword argument `:variable`. The argument for `:variable` is a list of indices specifying which delta-betas to plot or which variables to plot the residuals against. For `:plot-dbetas` variable 0 is the intercept, for `:plot-...-residuals` it is the linear predictor. If `variable` is not specified the delta-betas plot uses all variables; the residual plots just use the linear predictor. In addition, `:plot-...-residuals` has a `:smooth` keyword which indicates whether a lowess smoother should be added to the plot. The default is `t` for both `show-labels` and `smooth`. Any other arguments given to these functions will be passed to a call to `plot-points`.

The plots produced by these methods are linked, so that selections in one plot show up in the others. The residual plots also have an additional mouse mode "group highlighting" where selecting a point also selects all the other points in the same group. The linking is different from the usual mechanism in that all group-level plots are linked and all observation-level plots are linked but plots at different levels are not linked to each other. The gee object contains lists of observation-level and group-level plots that are separate from the linking list maintained by XLISP-Stat. To link another plot (eg a scatterplot of covariates) use the `:bind-plot-to-gee` message

```
(send mygee :bind-plot-to-gee a-plot +group+)
```

links `a-plot` to the group-level plots and similarly

```
(send mygee :bind-plot-to-gee a-plot +observation+)
```

links it to the observation-level diagnostic plots.

The same method names without the `plot-` prefix return the values of the diagnostics. The individual observation diagnostics are returned in a list or matrix in the same order as the data set; the group diagnostics have the group identifier attached and are in no reliable order. The plotting methods call two plotting functions which may be useful for other purposes: `index-plot` and `scatter-smooth`. The former takes a single argument and plots it against the integers; the second behaves like `plot-points` but also adds a lowess smooth to the plot.

Applying these methods to the model above, we do the following

```
>(send example1 :plot-dbeta)
computing diagnostic information
(#<Object: 2799672, prototype = SCATTERPLOT-PROTO> #<Object: 2771752,
prototype = SCATTERPLOT-PROTO> #<Object: 2847416, prototype =
SCATTERPLOT-PROTO> #<Object: 1545912, prototype = SCATTERPLOT-PROTO>
 #<Object: 1665112, prototype = SCATTERPLOT-PROTO> #<Object: 2305384
, prototype = SCATTERPLOT-PROTO> #<Object: 2399720, prototype =
SCATTERPLOT-PROTO>)
```

This gives seven plots of delta-betas showing the approximate effect on the intercept and six regression coefficients of deleting each observation. The plot titles are taken from the `predictor-names` argument of the model. Looking at the plot for "smoker" (the main variable of interest) we see that there is a large positive delta-beta value. Clicking on this point labels it as observation 15, and also highlights the other three observations on the same individual, showing that one has the largest negative delta-beta and another has a substantial positive delta-beta. Looking at the highlighting in the other plots shows that observation 15 also has large influence on the coefficent of the last time variable.

As more than one of the observations on this individual is influential we examine the effect of deleting all four of her observations. In order to reduce the number of superfluous windows, only the smoking and the last time variable will be examined. These are variables 2 and 6, counting the intercept as 0.

```
>(send example1 :plot-dbeta :unit +group+ :variable (list 2 6))
(#<Object: 2694968, prototype = SCATTERPLOT-PROTO>
#<Object: 2724856, prototype = SCATTERPLOT-PROTO>)
```

These plots show individual 19 as having very strong influence on the estimated coefficient of the smoking variable. Looking at the data we find that individual

19 is the same one we previously identified. The plot shows that removing this individual from the data would increase the coefficient of the smoking variable by about .12, nearly doubling it. The effect on the time variable is also large, but proportionally much smaller and of less intrinsic interest.

Residual plots against the linear predictor are also illuminating. The following commands plot the Pearson residuals and the standardised (decorrelated) Pearson residuals against the linear predictor. Plotting against another variables is not very useful in these data as most of the variables are binary.

```
> (send example1 :plot-pearson-residuals :variable (list 0))
#<Object: 2688968, prototype = SCATTERPLOT-PROTO>
> (send example1 :plot-standardised-residuals :variable (list 0))
#<Object: 2731752, prototype = SCATTERPLOT-PROTO>
```

The largest positive Pearson residual belongs to observation 15. Its value appears very large even considering that these residuals have not been divided by the square-root of the estimated scale factor (1.35 in this case). This residual is substantially reduced when the residuals are standardised by the square-root of the working correlation matrix. The large reduction may indicate that this observation has a large influence on the correlation estimates as well as on the regression coefficients.

To examine this further we refit the model without individual 19. The Lisp-Stat commands that create the new data set are in the example file and will not be repeated here.

```
(def example3 (gee-model :y newy :x (list newsex newsmoke newage newtime2
 newtime3 newtime4) :g newid :error poisson-error :correlation
(stat-m-dependence 3) :times newtimes :predictor-names (list "sex"
 "smoker" "mother's age" "2nd vs 1st" "3rd vs 1st" "4th vs 1st")))
No link specified - canonical link used
Iteration 1: quasideviance = 497.561
Iteration 2: quasideviance = 448.323
Iteration 3: quasideviance = 444.118
Iteration 4: quasideviance = 444.210
Iteration 5: quasideviance = 444.265
Iteration 6: quasideviance = 444.268
Iteration 7: quasideviance = 444.268

GEE Estimates:

                        Coefficient     Std Error          Naive Std Error
Constant                 -8.675037E-2  (0.277216)          (0.365146)
sex                      -8.533069E-2  (0.188420)          (0.210624)
smoker                    0.343040     (0.216824)          (0.235934)
mother's age              6.217507E-3  (6.000891E-3)       (5.775124E-3)
2nd vs 1st              -0.462035      (0.200852)          (0.196008)
3rd vs 1st              -0.356675      (0.207296)          (0.189279)
```

```
4th vs 1st               -1.27297     (0.202666)          (0.233578)

Scale Estimate:            1.68717
Independence model deviance:   444.268
Number of cases:               288
Link:                    #<Glim Link Object: LOG-LINK>
Variance function:       Poisson error
Stationary  3 - dependence Working Model:
 correlation matrix =
#2a(
    (  1.0      0.12      0.13      0.35    )
    ( 0.12       1.0      0.12      0.13    )
    ( 0.13      0.12       1.0      0.12    )
    ( 0.35      0.13      0.12       1.0    )
   )
EXAMPLE3
```

As predicted, the coefficient of the smoking variable has increased markedly, though it still has a p-value greater than 0.1. The correlation estimates also change, with the correlation between the first and fourth times now being much larger than the other two. This sort of correlation pattern seems surprising. The model can be refitted with a saturated correlation structure to estimate all the correlations separately.

```
> (def example4 (gee-model :y newy :x (list newsex newsmoke newage newtime2
 newtime3 newtime4) :g newid :error poisson-error :correlation saturated-corr
 :times newtimes :predictor-names (list "sex" "smoker" "mother's age"
"2nd vs 1st" "3rd vs 1st" "4th vs 1st")))

No link specified - canonical link used
Iteration 1: quasideviance = 497.561
Iteration 2: quasideviance = 448.346
Iteration 3: quasideviance = 444.266
Iteration 4: quasideviance = 444.521
Iteration 5: quasideviance = 444.638
Iteration 6: quasideviance = 444.649
Iteration 7: quasideviance = 444.650
Iteration 8: quasideviance = 444.650

GEE Estimates:

                         Coefficient     Std Error          Naive Std Error
Constant                  -7.202194E-2  (0.274819)          (0.357783)
sex                      -0.112027      (0.188841)          (0.205367)
smoker                    0.328388      (0.213805)          (0.230251)
mother's age               6.666360E-3  (5.977990E-3)       (5.662043E-3)
2nd vs 1st               -0.462035      (0.200852)          (0.201192)
```

```
3rd vs 1st                -0.356675      (0.207296)          (0.195908)
4th vs 1st                 -1.27297      (0.202666)          (0.233904)

Scale Estimate:             1.69459
Independence model deviance:   444.650
Number of cases:               288
Link:                     #<Glim Link Object: LOG-LINK>
Variance function:        Poisson error
Saturated Working Model:
 correlation matrix =
#2a(
    (  1.0       7.88E-2   6.85E-2  0.36    )
    (  7.88E-2   1.0      0.16      0.19    )
    (  6.85E-2  0.16       1.0      0.14    )
    ( 0.36      0.19      0.14       1.0    )
   )
EXAMPLE4
```

This does not seem to improve matters. The correlation is still stronger between widely separated than adjacent observations, which seems unlikely. The conclusions we would draw are that there is no evidence that smoking increases the number of hospital visits, but that the models considered do not appear to fit the data well, and in particular that the estimated effect of smoking is very strongly influenced by the experience of just one of the participants.

4 Missing Data

As described earlier, if the `:allow-missing` flag is set to `t` the program will allow different observation times to be present in different groups. It is convenient to be able to analyse data which are unbalanced either by design or because of a small number of missing observations but when data are missing, especially in longitudinal studies, it is important to consider the missingness mechanism. GEEs, like other moment-based estimates, are valid with data missing completely at random (MCAR) but not necessarily with data missing at random (MAR). Likelihood ratios, on the other hand, are preserved when data are missing at random, implying that direct likelihood and Bayesian inference, and asymptotic frequentist inference based on maximum likelihood estimates are valid.

This disadvantage of moment-based methods is more theoretical than practical for several reasons. First, and most obvious, is that likelihood methods must be using the right likelihood to have these theoretical properties. Score equations based on an incorrectly or incompletely modelled likelihood have no necessary advantage over any other estimating functions in handling missing data. Secondly, correctly modelled likelihood estimation can handle any MAR data pattern at least asymptotically, but other estimation methods will typically give correct answers with some MAR data patterns. For example, Liang & Zeger

(1986) claim that the Normal-model GEE is consistent when the missingness depends on any number of previous observations and the binomial-model GEE when the missingness depends on any single previous observation. Finally, and most importantly, it is the exception rather than the rule that data are missing at random, especially if there is any measurement error in the response, as the MAR condition requires that the missingness depend only on the observed values of non-missing observations. If there is any substantial amount of missing data no analysis ignoring the missingness mechanism should be trusted.

5 Other available software

General purpose packages providing GEE software include SPIDA (Statistical Laboratory, Macquarie University, Australia) and Stata (StataCorp). I am also informed by a reviewer that SAS 6.12 (SAS Corporation) includes this facility in PROC GENMOD but have not seen this version of the package. The implementation in SPIDA is somewhat limited, providing only the canonical link for the Normal, binomial and Poisson variance functions. The ordering of observations within a group is assumed to be correct in the data, leading to possible misleading results. Stata allows more freedom in combining link and variance functions but still prohibits, for example, binomial data with identity link. The Stata implementation requires a time identifier for correlation structures other than exchangeable and independence, but does not correctly handle missing observations. When the observation times for different groups are different a warning is produced to indicate that the results may be incorrect but the calculations are carried out as if the non-missing observations were actually adjacent in time.

User-contributed code is available for SAS and S. The `GEE` macro for SAS by Karim and Zeger and the `cgee` library for S by Carey and Macdermott provide similar functionality to Stata but require that the data are presorted by group and then by time within group. The S library does not check this and will return incorrect answers with no warning if the data are not correctly sorted.

While these programs (with the possible exception of the new SAS PROC GENMOD which I have not examined) provide fewer features than this Lisp-Stat implementation and in particular do not provide diagnostics they all have the advantage of computing environments providing superior data management and editing facilities to Lisp-Stat.

6 Acknowledgements

This work was supported by a Howard Hughes Medical Institute Predoctoral Fellowship in the Biological Sciences. Thanks to Anthony Rossini, Charlie Hall and John Preisser for help with diagnostics and to three reviewers for advice on presentation.

7 References

Diggle P., Liang, K-Y., Zeger, S.L. (1994) *Analysis of Longitudinal Data* Oxford University Press.

Fahrmeir, Tutz, (1994) *Multivariate Statistical Modelling using Generalized Linear Models* Springer Statistics, New York.

Huber, P.J. "The behaviour of maximum likelihood estimators under non-standard conditions" in LeCam, L.M. and Neyman J. [eds] *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability.* University of California Press.

Liang, K-Y., Zeger, S.L., (1986) "Longitudinal analysis using generalized linear models." *Biometrika* 73: 13–22.

Nelder, J. (1994) "An alternative view of the splicing data." *Applied Statistics.* 43: 469–476.

Preisser, J.S., Qaqish, B.F., (1996) "Deletion diagnostics for generalized estimating equations" *Biometrika* 83: 551-562.

Tierney, L. (1989) *XLISP-Stat: A Statistical Environment based on the XLISP language.* Technical Report No. 528. University of Minnesota School of Statistics.

Zeger, S.L., Liang K-Y., (1986) "Longitudinal data analysis for discrete and continuous outcomes" *Biometrics* 42: 121–130.

A Theoretical and Computational notes

A.1 Algorithms

The main computations are carried out in the method `:compute-step-beta`. This does not use the iteratively reweighted least squares method of `glim-proto` which would require rewriting `regression-proto` to handle non-diagonal weight matrices (feasible) and would involve inverting $n \times n$ matrices (undesirable). As observations in different groups are independent the weight matrix is block diagonal and so is mostly easily dealt with in blocks correponding to the groups. This only requires inverting the correlation matrix for each group and the final quasi-information matrix, all of which are small. The algorithm is as described by Liang & Zeger (1986), and is similar to Fisher scoring, with iteration continued until either the parameter estimates or the "independence model deviance" changes by less than `*gee-tolerance*`. Variable names in `compute-step-beta` are generally based on the notation in that paper. Some of the correlation estimates are slightly modified from the original paper to agree with other software.

The information required for the deletion diagnostics is computed by the same function as updates the regression parameter estimates. At the moment this information is not computed in the usual fitting process and another iteration is required the first time any deletion diagnostics are calculated. It would be possible to calculate this information while the model is fitted. This would cause extra garbage collection but would save time in plotting diagnostics. To do this, set the global variable `*gee-diagnostics-while-fitting*` to `t`. Variable names in the deletion diagnostic methods are based on the notation of Preisser & Qaqish, and so differ from those in the rest of the code.

As mentioned above, the functions do not assume that the data are presorted by group and observation time and instead require that observation times are supplied for any correlation structure where it would make a difference. This is because missing data handling is practically impossible without having a record of observation times (a recent version of a SAS macro introduces observation times to handle missing cases). It is also safer: other packages will give you the wrong answer with no warning if the data are not sorted by time within group. The way this is now implemented is not very efficient: asymptotically the calculation time is dominated by the time taken to find each group. More efficient code using hash tables would be possible. In practical examples, though, this is not a real issue. For very large data sets it is clearly worth assuming the data are sorted but it is also worth writing special-purpose compiled code.

As the observations within a group may vary in number and be in random order the correlation structures use an association list to match up the `:times` variable with the appropriate indices for selecting a submatrix of the correlation matrix. This produces very simple code for the method that returns the inverse correlation matrix and also simplifies the code for estimating correlations.

A.2 Performance

Despite various inefficiencies described above the code is fairly fast. A test data set with 72 groups of 5 observations and 5 variables using the autoregressive correlation structure (one of the slowest) took about 5 seconds per iteration on a (shared) Sparcstation 10 or 6 seconds per iteration on a 75mHz Pentium-based PC running Linux. This data set had the same number of observations in each group; unbalanced data require an extra matrix inversion for each group that is smaller than the largest one and so would be slightly slower. The speed is only improved about 20% after compiling the code, presumably because it is mostly vectorised and so none of the inefficiency is easy to optimise away.

A.3 Numerical Stability

One feature of GEEs and the generalised linear models from which they arose is the ability to combine arbitrary link and variance functions. Some care is needed to ensure that the numerical estimation doesn't crash when strange combinations of link and variance are chosen. The `:compute-step-beta` method contains a check that the new values of the regression coefficient produce valid values for the linear predictor and the fitted mean. If this is not the case then the step size is halved and the check repeated until the values are valid. A warning message is also printed. If there is no solution to the estimating equations in the permissible parameter space (as might happen with binomial error and identity link, for example), the algorithm will converge to a point on the boundary of the parameter space which will be a local extremum of the estimating function.

The corresponds to finding the maximum of a likelihood at the boundary of the parameter space. There is one situation when the program may crash rather than producing a solution, when the data are inconsistent with the chosen error function. The only example likely to occur is trying to fit the Gamma error model to data with observed zeroes.

The choice of starting values for the regression parameters is also complicated when the link and variance can be chosen arbitrarily (this is a disadvantage of not using iterated reweighted least squares). The program deals with three cases. For all the commonest models $\beta = 0$ gives valid estimates. For models where this is not the case (eg gamma/reciprocal link, power variance/power link) the initial values depend on whether the model contains an intercept. If it does then all the other regression parameters are set to 0 and the intercept is chosen to give valid starting values. If there is no intercept then a valid set of starting values need not exist. The program tries $\beta_i = g(\bar{y})/p\bar{x}_i$ (where $g(\cdot)$ is the link function) and if this doesn't work it gives up and asks for starting values to be specified.

In data sets with only a small number of groups it may not be possible to estimate all the parameters in certain correlation structures and the code will crash. If the program will not run with a certain correlation structure then it is probably wise to choose a simpler structure. As mentioned in Section 2, the standard error estimates may be unreliable when the number of groups is not larger, and if the correlation matrix is not well-determined by the data there is no guarantee that the method is even consistent.

A.4 Extending the software

The object-oriented implementation of the link, variance and correlation options makes it straightforward to add extra choices. The code uses the built-in `link-proto` type and defines prototypes `gee-varfun-proto` and gee-corr-proto. Any object responding to the same messages as these prototypes can be supplied in place of those provided. The easiest way to create new options (for example, an inverse gaussian variance structure) is to modify one of the existing examples. The code documents the messages that each object needs to respond to.

A.5 Definition of saturated correlation structures

The saturated correlation model estimates all the pairwise correlation parameters separately. There is a unique sensible estimate of the covariance matrix (assuming a balanced design) but not of the correlation matrix. The problem is that the diagonal elements of the covariance matrix are assumed equal under the model but will not be estimated as equal. The method used in the S `gee()` library, the Stata `xtgee`command and Karim & Zeger's `GEE` macro for SAS is to estimate the dispersion ϕ by the variance of the Pearson residuals, to estimate the correlations ρ_{ij} by $\rho_{ij} = \mathrm{cov}(r_i, r_j)/\phi$ for $i \neq j$ where r_i are the set of Pearson residuals at time i and to set $\rho_{ii} = 1$. The *Biometrika* paper used a similar

Table 3: Configuring gee-model using global variables.

Name	Default value
gee-verbose	t
gee-count-limit	30
gee-display-block-only	nil
gee-allow-missing	nil
gee-default-correlation	independence-corr
gee-tolerance	(sqrt (sqrt machine-epsilon)))
gee-diagnostics-while-fitting	nil
gee-corr-digits	2

estimate but did not set the diagonals to 1. This gives the ML estimate of the covariance matrix if the dispersion is not assumed constant over time. A third possibility would be to use the Normal ML estimate of the correlation matrix. It is not clear which is best, but most of the programs have settled on the same choice to make software testing easier. There is one significant disadvantage of this choice: it is the only one where the "correlation matrix" need not be positive definite. This can happen when the sample size is not large relative to the number of correlations being estimated and especially if the true time-specific variances are not all the same. It can only happen for three or more time points as for two time points the arithmetic-geometric mean inequality guarantees that the estimated correlation will be less than the usual estimate. The usual choice (diagonals forced to 1) is correlation structure saturated-corr, the covariance matrix divided by the scale parameter (Zeger & Liang's estimate, and the one given in the book by Fahrmeir & Tutz) is saturated-ml-corr

A.6 Options controlled by global variables

Much of the default behaviour is under the control of global variables. These are defined and documented at the end of the source file and in Table 3. The variable names are all of the form *gee-···* They are defined at the end of the file because the default correlation cannot be selected until after the correlation objects have been defined.