SAS and R Functions to Compute Pseudo-values for Censored Data Regression

By

John P Klein[1], Mette Harhoff[2], Per Kragh Andersen[2], and Sergey Tarima[1]

1. Introduction

In many applications investigators are interested in regression modeling of covariates on a survival outcome. The outcome may be the time to some event or the time until a competing risk event has occurred. Most applications use a Cox regression [1] model for the data. This approach models the hazard rate of the time to an event or in the case of competing risk data the crude hazard rate of the event in the presence of all the other risks [2]. Statistical procedures for the Cox model are available in most

regression model for the survival (or failure) probability at a single point in time. When M>1 then inference is to an entire survival cu

The final set of functions deal with regression models for the cumulative incidence function [6,7,8], $C_k(t)$, k=1,2. For two competing

The corresponding R function is the object pseudosurv which has arguments

Time—event time variable
Cens---the event indicator (1 event, 0 censored)
Tmax---a vector with the time points at which the pseudo-values are to be computed.

The function returns a new object with the original time and censoring variables and new variables containing the pseudo values. Here for M time points in Tmax an additional M columns are appended to the time and censoring matrix. Since no sorting of the data occurs in the function this can be appended to the original data to obtain an augmented file with the pseudo-values. The function uses the package "survival" in R.

To find pseudo-values for the restricted mean we have the SAS macro and R function pseudomean. The arguments of the two functions are the same as above, with the exception that the data set datatau in the SAS macro and Tmax in the R function are replaced by the maximum cut-off point τ for the restricted mean. For both functions the value of τ needs to be an interior point of the data. The functions are again based on Proc Lifereg in SAS and the package "survival" in R.

To find pseudo-values for the cumulative incidence functions the SAS macro 'pseudoci' and the R function 'pseudoci' are available. The SAS macro 'pseudoci' makes use of a macro 'cuminc' to compute the cumulative incidence function. The arguments of the SAS macro cuminc are

datain---the input data set
x --- the event time variable
re --- the indicator of the first competing risk (1--- occurred, 0 --- otherwise)
de --- the indicator of the second competing risk (1--- occurred, 0 --- otherwise)
dataout --- the name of an output data set
cir, cid --- variable names for the cumulative incidence function of the first and second competing risks, respectively

The macro uses PROC PHREG to obtain the crude hazard rates. $h_k(t)$,by fitting two Cox models, one for each competing risk, with a single covariate defined to be zero for all

cases. The output statement yields the cumulative crude hazard rate which is converted to the hazard rate at the event times. These are combined in a data step to yield the cumulative incidence functions.

The cumulative incidence macro is called in the macro pseudoci (datain, x, re, de , howmany, datatau,dataout) which computes the pseudo-values at the time points in the data set datatau. An expanded data set, dataout, includes all the data in the dataset datain and for each tau in datatau an entry for each observation with the variables rpseudo, dpseudo, the pseudo values for risks one and two respectively and tpseudo, the time point at which each pseudo-value is computed.

The R function pseudoci has arguments time (the event time variable), status (1 if occurrence of risk 1, 2 if occurrence of risk 2 and 0 otherwise). The final argument is Tmax which is a list of time points at which the pseudo-values are to be computed. The function use routines in the "cmprsk" library. The routine produces an object containing the pseudo-values for both competing risks. The output object consists of columns for the time and status variable and the pseudo-values, alternating between the two competing risks.

All the SAS and R functions are available at our website at http://www.biostat.mcw.edu/software/SoftMenu.html

4. Example

To illustrate the macros and functions we use a data set on HLA matched sibling donor bone marrow transplants [12]. This data set, which consists of data on 137 transplant patients, can be found on our website at http://www.biostat.mcw.edu/homepgs/klein/bmt.html.

An abbreviated data set constructed from these data consists of the time to death, relapse or lost to follow-up (tdfs), the indicators of relapse and death (relapse, trm), the indicator of treatment failure (dfs=relapse+trm), an id number from 1-137 (zid) and three factors that may be related to outcome: disease (1-Acute Lymphocytic Leukemia (ALL), 2-Low risk Acute Myeloid Leukemia (AML) and 3-High risk AML), the French-American-British Disease grade for AML (fab=1 if AML and Grade 4 or 5, 0 otherwise), and recipient age at transplant (age).

We first will examine regression models for disease free survival based on the Kaplan-Meier estimator We will use the SAS macro ‘pseudosurv’ to compute the pseudo-values. In this example we compute pseudo values at 100, 200, 400 and 600 days. We assume that the macro is in a file ‘sasmac’ in the current directory. The SAS code to compute the pseudo-values and put them into a permanent SAS data set ‘pseudoval’ is as follows

```
data one;
input tdfs trm relapse dfs id disease fab age;
lines;
2081 0 0 0 1 1 0 26
1602 0 0 0 2 1 0 21
      . . .
113 0 1 1 136 3 0 31
363 1 0 1 137 3 0 52
;
libname out ‘‘;
%include ‘sasmac’;
data times;
input tau;
lines;
100
200
400
600;
run;
%pseudosurv(one,tdfs,dfs,137,times,in.pseudoval)
proc print;
```

The data set `in.pseudoval` contains the following.

```
Obs  tdfs  trm  rel  dfs   id  disease  fab  rage  pseudo  tpseudo
1      1    1    0    1    35     1      0    42     0       100
2      2    1    0    1   108     3      1    20     0       100
                              . . .
336  390    0    1    1   117     3      1    23    -0.01    400
337  414    1    0    1    68     2      1    21     1.00    400
                              . . .
547 2569    0    0    0    39     2      1    19     1       600
548 2640    0    0    0    93     3      0    18     1       600
```

To compute regression estimates we use proc GENMOD. The code to fit a model using the complementary log-log link is as follows:

```
proc genmod;
```

The model shows that patients with AML low risk have better disease free survival than ALL patients (Relative Risk, RR=exp(-.9736)=0.38) and that AML patients with grade 4 or 5 FAB have a lower disease free survival (RR=exp(0.5737)=1.77).

Without re-computing the pseudo values we could examine the effect of FAB over time. We need to create in the data set a FAB indicator at each of the time points and rerun PROC GENMOD. The code is

```
data timedep; set in.pseudoval;
if tpseudo=100 then fab100=fab;  else fab100=0;
if tpseudo=200 then fab200=fab;  else fab200=0;
if tpseudo=400 then fab400=fab;  else fab400=0;
if tpseudo=600 then fab600=fab;  else fab600=0;
proc genmod;
class zid disease (param=ref ref=first) tpseudo (param=ref
ref=first);
FWDLINK LINK=LOG(-log(1-_MEAN_));
INVLINK ILINK=1-EXP(-Exp(_XBETA_));
model pseudo= tpseudo disease fab100 fab200 fab400 fab600
repeated subject=zid/corr=ind ;
contrast 'fab'
fab100 1 fab200 0 fab400 0 fab600 0,
fab100 0 fab200 1 fab400 0 fab600 0,
fab100 0 fab200 0 fab400 1 fab600 0,
fab100 0 fab200 0 fab400 0 fab600 1/wald;
contrast 'fab by time'
fab100 1 fab200 -1 fab400 0 fab600 0,
fab100 0 fab200 1 fab400 -1 fab600 0,
fab100 0 fab200 0 fab400 1 fab600 -1/wald;
```

Here the two contrast statements test for an overall FAB effect and if the FAB effect changes with time, respectively. The relevant output is

```
              Analysis Of GEE Parameter Estimates
               Empirical Standard Error Estimates
                         Standard   95% Confidence
Parameter      Estimate  Error         Limits      Pr > |Z|
Intercept       0.7575   0.1950   0.3753   1.1397   0.0001
tpseudo   200  -0.7413   0.1617  -1.0582  -0.4244   <.0001
tpseudo   400  -1.1113   0.1873  -1.4784  -0.7443   <.0001
tpseudo   600  -1.3184   0.1980  -1.7065  -0.9302   <.0001
disease   2     0.8362   0.2841   0.2792   1.3931   0.0033
disease   3    -0.2002   0.3079  -0.8037   0.4034   0.5157
fab100         -0.6454   0.3207  -1.2741  -0.0168   0.0442
```

```
fab200          -0.3262   0.2753  -0.8658   0.2134   0.2361
fab400          -0.4075   0.2883  -0.9726   0.1575   0.1575
fab600          -0.5906   0.3180  -1.2139   0.0327   0.0633
                       Contrast Results for GEE Analysis
                                Chi-
    Contrast           DF     Square    Pr > ChiSq    Type
    fab                 4       6.19        0.1857    Wald
    fab by time         3       2.59        0.4593    Wald
```

This model shows that there is no difference in the FAB effect over time (p=0.4593).

Now we implement the same operations with R. First we download the data into an R object, define the required time points and generate pseudo-values. We assume that the data are in the file ‘data.txt’ in the current directory.

```
a<-read.table(file="data.txt", header=T)
cutoffs <- c(100,200,400,600)
pseudo <- pseudosurvival(a$tdfs,a$dfs,cutoffs)
```

The “pseudo” object is as follows.

```
> pseudo[order(pseudo$time),]

      time cens    tmax =100  tmax =200  tmax =400   tmax =600
35     1    1         0          0         0.0000      0.0000
108    2    1         0          0         0.0000      0.0000
                   .   .   .
117  390    1         1          1        -0.0094     -0.0080
68   414    1         1          1         1.0019     -0.0080
                   .   .   .
39  2569    0         1          1         1.0019      1.0032
93  2640    0         1          1         1.0019      1.0032
```

The second step requires some data manipulation to prepare for the GEE step.

```
b <- NULL
for(j in 3:ncol(pseudo)) b <- rbind(b,cbind(a,pseudo =
pseudo[,j],tpseudo = cutoffs[j-2]))
b <- b[order(b$id),]
library(geepack)
b$tpseudo <- as.factor(b$tpseudo)
b$disease <- as.factor(b$disease)
b$fab <- as.factor(b$fab)
b$id <- as.factor(b$id)
```

```
PVal=round(2-2*pnorm(abs(fit$beta/sqrt(diag(fit$vbeta.ajs)))),4))
```

```
                mean   SD     Z       PVal
 (Intercept)  1.1897 0.3670  3.2416 0.0012
 tpseudo200  -0.6181 0.1272 -4.8602 0.0000
 tpseudo400  -1.0398 0.1533 -6.7831 0.0000
 tpseudo600  -1.3025 0.1654 -7.8758 0.0000
 disease2     0.9735 0.3003  3.2422 0.0012
 disease3    -0.0490 0.3251 -0.1507 0.8803
 fab1        -0.5737 0.2676 -2.1437 0.0321
 rage        -0.0200 0.0125 -1.5962 0.1105
```

To examine the effect of FAB over time we create four new variables

```
b$fab100 <- 0; b$fab100[b$tpseudo==100] <- b$fab[b$tpseudo==100];
b$fab200 <- 0; b$fab200[b$tpseudo==200] <- b$fab[b$tpseudo==200];
b$fab400 <- 0; b$fab400[b$tpseudo==400] <- b$fab[b$tpseudo==400];
b$fab600 <- 0; b$fab600[b$tpseudo==600] <- b$fab[b$tpseudo==600];

b$fab100 <- as.factor(b$fab100)
b$fab200 <- as.factor(b$fab200)
b$fab400 <- as.factor(b$fab400)
b$fab600 <- as.factor(b$fab600)
```

and use them in the GEE regression model

```
fit <- geese(pseudo ~ tpseudo + disease + fab100 + fab200 + fab400
+   fab600,   data   =   b,   id=id,   jack=TRUE,   scale.fix=TRUE,
family=gaussian, mean.link = "cloglog", corstr="independence")
cbind(mean = round(fit$beta,4),
SD = round(sqrt(diag(fit$vbeta.ajs)),4),
Z = round(fit$beta/sqrt(diag(fit$vbeta.ajs)),4),
PVal = round(2-2*pnorm(abs(fit$beta/sqrt(diag(fit$vbeta.ajs)))),4))
```

The results are

```
                mean   SD       Z     PVal
 (Intercept)  0.7575 0.1917  3.9510 0.0001
 tpseudo200  -0.7413 0.1578 -4.6970 0.0000
 tpseudo400  -1.1113 0.1828 -6.0801 0.0000
 tpseudo600  -1.3183 0.1933 -6.8219 0.0000
 disease2     0.8362 0.2797  2.9900 0.0028
 disease3    -0.2002 0.3058 -0.6546 0.5127
 fab1001     -0.6454 0.3187 -2.0253 0.0428
 fab2001     -0.3262 0.2728 -1.1955 0.2319
 fab4001     -0.4075 0.2855 -1.4275 0.1534
 fab6001     -0.5906 0.3155 -1.8719 0.0612
```

To test the overall FAB effect we use the following R code.

```
C <- rbind( c(0,0,0,0,0,0,1,0,0,0), c(0,0,0,0,0,0,0,1,0,0),
c(0,0,0,0,0,0,0,0,1,0), c(0,0,0,0,0,0,0,0,0,1))
SSH0 <- t(C %*% fit$beta) %*% solve(C %*% fit$vbeta.ajs %*% t(C))
%*% (C %*% fit$beta)
1-pchisq(SSH0,nrow(C))
          [,1]
[1,] 0.1790
```

To test if the FAB effect differs with time we use the following R code.

```
C <- rbind(c(0,0,0,0,0,0,-1,1,0,0), c(0,0,0,0,0,0,0,1,-1,0),
c(0,0,0,0,0,0,0,0,1,-1))
SSH0 <- t(C %*% fit$beta) %*% solve(C %*% fit$vbeta.ajs %*% t(C))
%*% (C %*% fit$beta)
1-pchisq(SSH0,nrow(C))
          [,1]
[1,] 0.4443
```

For the restricted mean time to treatment failure we use the SAS macro or the R function “pseudomean”. To illustrate we look at a regression model for the mean time to treatment failure restricted to 2000 days. Here we use the identity link function. The SAS code, assuming the macro was in the file ‘pseudomu’ is

```
%include 'pseudomu';
%pseudomean(one, tdfs, dfs, 137,2000,outdata);
proc genmod;
class zid disease (param=ref ref=first);
model psumean= disease fab rage/dist=normal link=id
noscale;
repeated subject=id/corr=ind;
```

The relevant output is

Analysis Of GEE Parameter Estimates
Empirical Standard Error Estimates

Parameter		Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept		1154.997	219.2613	725.2530	1584.741	5.27	<.0001
disease	2	630.5407	185.4911	266.9848	994.0967	3.40	0.0007
disease	3	143.5041	216.8834	-281.580	568.5878	0.66	0.5082
fab		-518.600	169.5438	-850.900	-186.301	-3.06	0.0022
age		-11.5556	6.8876	-25.0551	1.9438	-1.68	0.0934

Here we see that AML low risk patients have the longest restricted mean life, namely 630.5 days longer than ALL patients and that AML patients with FAB class 4/5 have lifetimes 578.6 days shorter than the reference group.

The analogous R commands and output would be

```
a<-read.table(file="data.txt", header=T)
a <- cbind(a,pseudo = pseudomean(time=a$tdfs, dead=a$dfs,
tmax=2000)$psumean)
library(geepack)
a$disease <- as.factor(a$disease)
summary(fit <- geese(pseudo ~ rage + fab + disease, data = a, id=id,
jack = T, family=gaussian, corstr="independence", scale.fix=F))
cbind(mean = round(fit$beta,4),
SD = round(sqrt(diag(fit$vbeta.ajs)),4),
Z = round(fit$beta/sqrt(diag(fit$vbeta.ajs)),4),
PVal = round(2-2*pnorm(abs(fit$beta/sqrt(diag(fit$vbeta.ajs)))),4))
```

```
               mean     SD         Z     PVal
(Intercept) 1154.9972 223.1147  5.1767 0.0000
disease2     630.5407 187.2927  3.3666 0.0008
disease3     143.5041 220.7480  0.6501 0.5156
fab         -518.6004 172.8409 -3.0004 0.0027
rage         -11.5556   7.0672 -1.6351 0.1020
```

This R output shows elevated standard deviations resulting in higher P-values than in SAS output.

The restricted mean pseudo values with an identity link can also be used with the “gee” function from the “gee” package [14] as follows.

```
library(gee)
fit <- gee(pseudo ~ disease + fab + rage, data = a, id=id,
family=gaussian, corstr="independence", scale.fix=F)
cbind(mean = round(fit$coef,4),
SD=round(sqrt(diag(fit$robust.variance)),4),
Z=round(fit$coef/sqrt(diag(fit$robust.variance)),4),
PVal=round(2-
2*pnorm(abs(fit$coef/sqrt(diag(fit$robust.variance)))),4))
```

```
                mean     SD       Z     PVal
(Intercept) 1154.9972 219.2613  5.2677 0.0000
disease2     630.5407 185.4911  3.3993 0.0007
disease3     143.5041 216.8834  0.6617 0.5082
fab         -518.6004 169.5438 -3.0588 0.0022
age         -11.5556   6.8876 -1.6777 0.0934
```

The function “gee” which used the sandwich estimator (3) to estimate the variance shows results identical to SAS. However, “gee” requires the use of a default link function (identity for the normal) and does not allow the selection of the complementary log-log as needed with the pseudovalue approach for survival and cumulative hazard functions.

For the cumulative incidence function we use the SAS macro and the R function “pseudoci” to compute the pseudo-values. To illustrate the SAS code we fit the complementary log-log model to the relapse cumulative incidence evaluated at 100, 200, 400, 600 days. Assuming the macro is in the file ‘pseudoci.txt’ the SAS code is

```
%include 'pseudoci.txt';
 data times;
input tau ;
cards;
100
200
400
600
run;
%pseudoci(one,tdfs,rel,trm,137,times,in.dataoutcr);
data two;  set in.dataoutcr ;
 dis2=0; dis3=0; if disease=2 then dis2=1;
 if disease=3 then dis3=1;
proc print data=two round;
proc genmod;
class zid tpseudo ;
FWDLINK LINK=LOG(-log(1-_MEAN_));
INVLINK ILINK=1-EXP(-Exp(_XBETA_));
model rpseudo= tpseudo dis3 dis2 fab /dist=normal noscale  noint;
repeated subject=zid/corr=ind ;
```

A partial listing of the SAS output is as follows:

```
                     d              r  d   t
                     i              p  p   p
                     s              s  s   s
     t               e      r       e  e   e   d  d
O    d  t  r  d      a  f   a    t   u  u   u   i  i
b    f  r  e  f   i  s  a   g    a   d  d   d   s  s
s    s  m  l  s   d  e  b   e    u   o  o   o   2  3
1    1  1  0  1   35 1  0  42   17w  s   s       s    inalys.0Of GEE Parameter E
```

Empirical Standard Error Estimates

Parameter		Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
Intercept		0.0000	0.0000	0.0000	0.0000	.	.
tpseudo	100	2.5704	0.6280	1.3395	3.8012	4.09	<.0001
tpseudo	200	2.0100	0.6183	0.7982	3.2218	3.25	0.0012
tpseudo	400	1.5875	0.6004	0.4107	2.7644	2.64	0.0082
tpseudo	600	1.4160	0.5935	0.2527	2.5792	2.39	0.0170
dis3		0.3183	0.5775	-0.8136	1.4502	0.55	0.5815
dis2		1.7435	0.6561	0.4577	3.0294	2.66	0.0079
fab		-1.1645	0.5079	-2.1600	-0.1689	-2.29	0.0219
rage		-0.0146	0.0209	-0.0555	0.0263	-0.70	0.4847

Parameter		Estimate	Standard Error	95% Confidence Limits		Z	Pr > |Z|
tpseudo	100	-2.5704	0.6280	-3.8012	-1.3395	-4.09	<.0001
tpseudo	200	-2.0100	0.6183	-3.2218	-0.7982	-3.25	0.0012
tpseudo	400	-1.5875	0.6004	-2.7644	-0.4107	-0	0.629eudo 0.4107 <.0001 0 s0r

dij0 -1.12-m-.00107527 p1s7scs2.2 Z|

trm pseudo-value at 200 days, and so forth. In order to use the “geese” function we need only relapse pseudo-values arranged in one column and in another column we need the pseudo-value’s time points. The six lines of code in bold after the call to “pseudoci” merge the output of the function with the original data and prepare it for analysis using the function “geese.” The program and output are given below:

```
a<-read.table(file="data.txt", header=T)
cutoffs <- c(100,200,400,600)
a$icr <- a$rel + 2 * a$trm
#This code creates a competing risk indicator with value
# 1 if relapse, 2 if dead in remission, 0 if censored
pseudo <- pseudoci(a$tdfs,a$icr,cutoffs)

rel_mask <- c(100,-1,200,-1,400,-1,600,-1)
b <- NULL
for(j in 3:ncol(pseudo)) b <- rbind(b,cbind(a,pseudo =
pseudo[,j],tpseudo = rel_mask[j-2]))
b <- b[order(b$id),]
b <- b[b$tpseudo != -1,]

library(geepack)
b$tpseudo    <- as.factor(b$tpseudo)
b$disease <- as.factor(b$disease)
b$fab <- as.factor(b$fab)
fit <- geese(pseudo ~ tpseudo + disease + fab + rage - 1 , data =
b, id=id, jack = T,
scale.fix=TRUE, family=gaussian, mean.link = "cloglog",
corstr="independence")
cbind(mean = round(fit$beta,4),
SD = round(sqrt(diag(fit$vbeta.ajs)),4),
Z = round(fit$beta/sqrt(diag(fit$vbeta.ajs)),4),
PVal = round(2-2*pnorm(abs(fit$beta/sqrt(diag(fit$vbeta.ajs)))),4))
```

```
               mean     SD       Z   PVal
 tpseudo100 -2.5704 0.6495 -3.9577 0.0001
 tpseudo200 -2.0100 0.6404 -3.1387 0.0017
 tpseudo400 -1.5875 0.6237 -2.5455 0.0109
 tpseudo600 -1.4160 0.6170 -2.2948 0.0217
 disease2   -1.7435 0.6687 -2.6072 0.0091
 disease3   -0.3183 0.5910 -0.5386 0.5902
 fab1        1.1645 0.5235  2.2244 0.0261.1387 0.0017
```

We have presented SAS macros and R functions to find pseudo-values for the survival function, the restricted mean and the cumulative incidence function. The routines can be found on our website at

http://www.biostat.mcw.edu/software/SoftMenu.html

The regression models for the survival function and cumulative incidence functions can be based on the functions at a single point in time or they can be for several points of the curves. When a regression model for the entire curve is to be studied we recommend five to ten time points roughly evenly spaced on the event scale. In the examples we used an independent working covariance matrix for the GEE calculations. Another possibility is to use the empirical correlations between the pseudo-values. [6]

The “geese” function from the R package “geepack” was used for GEE fitting. The “gee” function did not allow us to change mean link function to complementary log for the Gaussian family. However, “gee” sandwich variance estimates are identical to those in SAS, which is not true for “geese”.

Acknowledgement

This research is supported by a grant R01-54706-12 from the National Cancer Institute. Comments on the R functions from Maja Pohar Perme are greatly appreciated.

References

1. Cox, D.R. (1972). Regression Models and Life-Tables (with discussion).

3. Klein, J.P. and Zhang, M.J. Survival Analysis, Software : *Encyclopaedia of Biostatistics 2nd Edition Volume 8* (Armitage and Colton, Editors) p 5377-5382, 2005
4. Andersen, P.K., Klein, J.P. and Rosthøj, S. (2003). Generalized Linear Models for Correlated Pseudo-Observations with Applications to Multi-State Models. *Biometrika* **90**, 15-27.
5. Andersen, P.K., Hansen, M.G., and Klein, J.P. (2004), Regression Analysis of Restricted Mean Survival Time Based on Pseudo-Observations, *Life Time Data Analysis* **10**, 335-350
6. Klein, J.P., and Andersen P.K. (2005), Regression Modeling of Competing Risks Data Based on Pseudo-Values of the Cumulative Incidence Function. *Biometrics,* **61**, 223-229
7. Klein, J.P. Modeling Competing Risks in Cancer Studies. (2006) *Statistics in Medicine* **25**, 1015-1034, 2006.
8. Andersen, P.K., Klein J.P.. (2007) Regression Analysis for Multistate Models Based on a Pseudo-value Approach, with Applications to Bone Marrow Transplantation Studies. *Scandinavian Journal of Statistics **34***, 3-16.
9. Klein, J.P., Andersen, P.K., Logan, B.L. and Harhoff, M. G.(2007) Analyzing survival curves at a fixed point in time. Statistics in Medicine (In Press).
10. Liang, K.-Y. and Zeger, S.L. (1986). Longitudinal Data Analysis Using Generalized Linear Models. *Biometrika* **78**, 13-22.
11. Kaplan, E. L. and Meier, P. (1958). Non-Parametric Estimation from Incomplete Observations. *Journal of the American Statistical Association,* **53**, 457-481.

12. Copelan, E. A., Biggs, J. C., Thompson, J. M., Crilley, P., Szer, J., Klein, J. P., Kapoor, N., Avalos, B. R., Cunningham, I., Atkinson, K., Downs, K., Harmon, G. S., Daly, M. B., Brodsky, I., Bulova, S. I., and Tutschka, P. J. (1991). Treatment for Acute Meyelocytic Leukemia with Allogeneic Bone Marrow Transplantation Following Preparation with Bu/Cy. *Blood,* **78**, 838-843.
13. http://cran.r-project.org/
14. Yan, J., and Fine J. (2004) Estimating Equations for Association Structures. *Statistics in Medicine*, 23, 859-874.