General Relative Rate Models for the Analysis of Studies Using Case-Cohort Designs

David B. Richardson, Bryan Langholz, and Kaitlin Kelly-Reif

Web Appendices 1–4

WEB APPENDIX 1
Creating a Risk-Set Data Structure From Case-Cohort Data

Suppose that the basic case-cohort data (case_cohort) include the following variables: study id (id), entry time (ageentry), exit time (ageexit), an exposure variable of interest (exposure), and a failure indicator (ccoh_ind) that takes values 0, 1, or 2 for subcohort non-failures, subcohort failures, and non-subcohort failures, respectively.   The risk-set data structure (riskset) can be generated from these data via a simple SAS program, as shown below.

```
*   Adjust entry time for non-cohort cases;
data case_cohort;
  set case_cohort;
  if ccoh_ind eq 2 then ccohentry=ageexit-.0001;
  else ccohentry=ageentry; run;

proc sort data=case_cohort; by ageexit; run;

*  Select out failures ;
data failures;
  set case_cohort;
  where ccoh_ind ne 0;
  caseid = id; setno = _n_; rstime = ageexit; rsentry=rstime-.0001;
  keep caseid setno rstime rsentry; run;

* Create case-cohort risk sets;
proc sql;
  create table ccoh_rs as
    select *
      from failures, case_cohort
      where  (case_cohort.ccohentry < failures.rstime <=
case_cohort.ageexit)  ;
quit;

*  Find the subject who is the failure (case) in this risk set;
data analytic;
  set ccoh_rs;
  cc = caseid eq id; drop caseid; run;

* Find the maximum risk set size;
proc freq data= analytic;
table setno / noprint out=_ncovals (rename=(count=_ncovals)
drop=percent);run;

proc sql noprint;
  select max(_ncovals)
    into :maxsetsize
    from _ncovals; quit;

%let maxsetsize= &maxsetsize;

proc sort data=analytic; by setno descending cc; run;
```

```
* Transform to one record per risk set;
data riskset (keep=setno y z1-z&maxsetsize ntot);
  set analytic;
  by setno;
  array z{&maxsetsize}; retain i z1-z&maxsetsize;
  if first.setno then do;
     do i=1 to &maxsetsize; z{i} =.;end; i = 0;end;
  i = i + 1;  z{i} = exposure;
  y=1; ntot=i; if last.setno then output; run;
```

WEB APPENDIX 2
Robust Variance

To compute dfbetas for general relative rate models using the risk-set data structure (riskset) created in Appendix 1, the predicted phi and its derivative is needed for each person (from which we calculate dfbetas). This is accommodated by using a separate predict statement for each person. A macro is used to generate all the predict statements. Here we illustrate code for the empirical example in this paper, in which a model is fitted to estimate the coefficient for a linear excess relative rate model. The code can be readily extended to regression models with multiple explanatory variables.

```
* To compute dfbetas the predicted phi and its derivative is needed
for each person;
%macro predictphi(size);
%do i = 1 %to &size;
predict phi&i out=_phi&i (keep=setno ntot pred der_beta ) der;
%end;
%mend;

options nomprint;ods output parameterestimates=ests CovMatParmEst=cov;
proc nlmixed data=riskset cov;
  parms beta=0  ;
  array z{1,&maxsetsize} z1-z&maxsetsize;
  array phit{&maxsetsize} phi1-phi&maxsetsize;
  sum=0; ntot=ntot;
  do i = ntot to 1 by -1;
    phit{i} = exp(beta*z{1,i}); sum=sum + phit{i};
  end;
  L = log(phit{1} /sum);
  model y~general(L);
  predict sum out=_sum (keep=setno ntot pred der_beta ) der;
  %predictphi(&maxsetsize);
  estimate 'beta' exp(beta); run;  ods output;

proc sql;
  create table delta1 as
    select _sum.setno,cc,id,pred,der_beta,dose
      from td_analytic,_sum
      where td_analytic.setno eq _sum.setno;
      quit;

%macro phis(size);
  %do i = 1 %to &size;  _phi&i %end;
%mend;

data phis (rename=(pred=i_pred der_beta=i_der_beta ));
```

```
 set %phis(&maxsetsize) ;
 by setno;
 retain count;
 if first.setno then count = 0;
 count = count + 1;
 if count le ntot then output; else delete;  run;

data dfbetas1;
  merge delta1 phis;
  retain pred der_beta  b_1 cov_11  ;
  if _n_ eq 1 then do;
   do i = 1 to 1;
    merge ests cov;
      by parameter;
    if parameter eq 'beta' then do; b_1 = estimate; cov_11=beta;  end;
    end;  end;
  r1 = (i_der_beta/i_pred - der_beta/pred)*(cc - i_pred/pred);
  dfb_1 = cov_11*r1  ;
 drop parameter dose i_pred i_der_beta pred der_beta  b_1  cov_11  r1
; run;

proc summary data=dfbetas1 sum nway;
   class id cc;
   var dfb_1 ; output out=dfbetas2 sum=dfb_1  ; run;

* include additional persons that dont contribute to the analysis;
proc sort data=case_cohort out=cc_ids (keep=id ccoh_ind) nodupkey;
by id ccoh_ind; run;

data dfbetas;
  merge cc_ids dfbetas2 (in=in_rs) ;
  by id;
  if not in_rs and ccoh_ind in(0,1) then do; cc=0; dfb_1 = 0;
_freq_=0; _type_=1; output; end;
  if not in_rs and ccoh_ind in(1,2) then do; cc=0; dfb_1 = 0;
_freq_=0; _type_=1; output; end;
  if in_rs then output;
  delete;  run;

title3 'Robust variance estimator';
proc summary data=dfbetas sum nway;
   class id;
   var dfb_1 ;   output out=summed sum=dfb_1  ; run;

ods output Sscp=sscp; proc corr data=summed sscp; var dfb_1 ; run;

data ci (keep=beta var se L95CI U95CI);
merge sscp (keep=dfb_1) ests (keep= estimate);
beta=estimate; var=dfb_1; se=sqrt(var);
L95CI= (beta-1.96*se); U95CI= (beta+1.96*se); run;

proc print data=ci;run;
```

WEB APPENDIX 3
SAS Code to Create Analytical Data File for Bootstrapping Confidence Intervals and Fit the Bootstrap Models

A risk-set data structure (riskset) is generated from the case-cohort data, similar to the approach in Appendix 1; however, *B* weights are appended to the case-cohort data. Once risk sets have been enumerated, *B* copies of the risk set data are created and indexed by a variable (boot). A weighted regression is fitted using PROC NLMIXED. The “parms” statement tells SAS that the parameter of interest (beta) is to be estimated and sets the initial value to zero. The “by boot” statement fits all the bootstrapped models in a single call of the procedure. The estimated parameter(s) of interest are saved after each model fitting in file named “out”. Bootstrap intervals are obtained after fitting weighted regression models over *B* iterations.

```
*   Append B random weights to each subject;
%let nboot=500;
data case_cohort;
  set case_cohort;
    array wt{&nboot} wt1-wt&nboot;
    do r = 1 to &nboot;   wt{r}= ranexp(123) ; end;  run;

*   Adjust entry time for non-cohort cases;
data case_cohort;
  set case_cohort;
  if ccoh_ind eq 2 then ccohentry=ageexit-.0001;
  else ccohentry=ageentry;run;

proc sort data=case_cohort; by ageexit; run;

* Select out failures ;
data failures;
  set case_cohort;
  where ccoh_ind ne 0;
  caseid = id; setno = _n_; rstime = ageexit; rsentry=rstime-.0001;
  keep caseid setno rstime rsentry; run;

* Create case-cohort risk sets;
proc sql;
  create table ccoh_rs as
    select *
      from failures, case_cohort
      where  (case_cohort.ccohentry < failures.rstime <=
case_cohort.ageexit)  ;
quit;

* Find the subject who is the failure (case) in this risk set;
data analytic;
  set ccoh_rs;
  cc = caseid eq id; drop caseid; run;

*Establish maximum risk set size;
```

```
proc freq data= analytic;
table setno / noprint out=_ncovals (rename=(count=_ncovals)
drop=percent);run;

proc sql noprint;
  select max(_ncovals)
    into :maxsetsize
    from _ncovals;
quit;

%let maxsetsize= &maxsetsize;


data long (keep=boot setno cc exposure weight);
  set analytic ;
  array wt{&nboot} wt1-wt&nboot;
  do r = 1 to &nboot;  boot=r; weight = wt{r}  ;  output; end; run;


proc sort data=long ; by boot setno descending cc; run;

*Transform to one record per risk set;
data riskset (keep=boot setno z1-z&maxsetsize w1-w&maxsetsize ntot);
  set long;
  by boot setno;
  array z{&maxsetsize};
  array w{&maxsetsize};
  retain i z1-z&maxsetsize w1-w&maxsetsize;
  if first.setno then do;
     do i=1 to &maxsetsize; z{i} =.;w{i} =.; end; i = 0;end;
  i = i + 1;  z{i} = Z;
  w{i} = weight; ntot=i; if last.setno then output; run;


ods select  parameterestimates; ods output parameterestimates=ests  ;
 proc nlmixed data= riskset;
  parms beta=0  ;
   array z{1,&maxsetsize} z1-z&maxsetsize;
  array w{&maxsetsize} w1-w&maxsetsize;
   cc =1;   sum=0;  ntot=ntot;
  do i = ntot to 1 by -1;
    Phi=1 + ( z{1,i}*beta   );
    sum = sum + ( Phi   * w{i}) ;
   caseprod =  Phi **   w{i}     ;
  end;
  L=caseprod / sum**   w{1} ;
  model cc ~ general(log(L));  by boot;  run;

data rout1;
  set ests;
  if Parameter='beta'; e_beta=  (Estimate); run;
```

```
title2 'Estimate and 95% limits by percentile';
     proc univariate data=rout1 noprint;
           var e_beta  ;
           output out=cis1_pct pctlpts=2.5 50 97.5 pctlpre=cis; run;
           data cis1_pct;
                set cis1_pct;
                l95 =  (cis2_5);
                est =  (cis50);
                u95 =  (cis97_5); run;
           proc print data=cis1_pct noobs; var l95 est u95; run;

title2 'Estimate and 95% limits by normality assumption';
     proc means data=rout1 mean std noprint;
           var Estimate;
           output out=means; run;
           proc sql;
                create table b as select Estimate as mean
                from means where _stat_="MEAN";
                create table c as select Estimate as stderr
                from means where _stat_="STD";
                select      (b.mean-1.96*c.stderr) as l95,
                       (b.mean) as est,
                       (b.mean+1.96*c.stderr) as u95
                from b,c; quit; title2; run;
```

WEB APPENDIX 4

Stratified Case-Cohort Data Analysis

To illustrate the proposed method as applied to a stratified case-cohort study, we used data for a cohort of 1742 women who were treated for tuberculosis by chest fluoroscopy at two sanatoria in Massachusetts between 1930 and 1956 (12). The cohort was followed through December 31, 1980 to ascertain information on vital status, date of death, and underlying cause of death. The outcome of interest here is death due to breast cancer. The exposure of interest was estimated radiation dose to breast for those women who received radiation exposure to the chest from the X-ray fluoroscopy lung examination and was computed based on the number of fluoroscopies, type of equipment used, and other exposure conditions. The remaining women received treatments that did not require fluoroscopic monitoring and were radiation unexposed. Seventy-five breast cancer cases were identified. The case-cohort sample we will use to illustrate the methods include a subcohort of 100 sampled subjects and all breast cancer cases who were not sampled. In this stratified case-cohort sample, the 100 subjects were sampled from age-at-first-exposure strata (< 15, 15–19, 20–29, 30+) in numbers proportional to the number of breast cancer cases in the strata. Three subcohort members with missing radiation dose information were excluded from the analysis.

We fitted a regression model with indicator variables for radiation dose 1-249 rad and greater than 250 rad compared to 0 rad. We fitted a log-linear regression model with two binary indicator terms for radiation dose. We compared estimates obtained using the method in the current paper with bootstrap-based confidence intervals to the estimate and robust confidence intervals reported by Langholz and Jiao (2007) for age at first treatment stratified case-cohort analyses of these same data.

For the contrast of <250 to 0 mGy, the estimated hazard ratios was 1.8 and the bootstrap confidence intervals based on either a normal approximation or based on percentiles of the bootstrap were similar (0.9, 3.7) to each other, and to the robust 95% confidence interval (0.9, 3.7). For the contrast >250 mGy to 0 mgy, the estimated hazard ratio was 2.5 and the bootstrapped bounds based on a normal assumption (1.0, 26.4) and based on percentiles (1.0, 24.5) were again similar to each other and to the robust 95% confidence interval (0.9, 23.9).

Below we provide SAS code to create analytic data files for bootstrapping confidence intervals and fit the bootstrap models for a stratified case-cohort study. The approach is nearly identical to the approach described above for a case-cohort study based on a simple random sample of cohort. In this example, the case-cohort data are stratified on a baseline variable (agefirstgr). *B* weights are appended to the case-cohort data. A risk-set data structure (riskset) is generated from the case-cohort data; risk sets are stratified on the potential confounder, agefirstgr. Once risk sets have been enumerated, *B* copies of the risk set data are created and indexed by a variable (boot). A weighted regression is fitted using PROC NLMIXED. In this example, there are two estimated parameter of interest (beta1 and beta2) associated with two explanatory variables, dcat1 and dcat2. Bootstrap intervals are obtained after fitting weighted regression models over *B* iterations.

```
*   Append B random weights to each subject;
%let nboot=500;
data case_cohort;
  set st_case_cohort;
    array wt{&nboot} wt1-wt&nboot;
    do r = 1 to &nboot;   wt{r}= ranexp(123) ; end;  run;

*   Adjust entry time for non-cohort cases;
data case_cohort;
  set case_cohort;
  if ccoh_ind eq 2 then ccohentry=ageexit-.0001;
  else ccohentry=ageentry;run;

proc sort data=case_cohort; by ageexit; run;


* Select out failures ;
data failures;
  set case_cohort;
  where ccoh_ind ne 0;
  caseid = id; setno = _n_; rstime = ageexit; rsentry=rstime-.0001;
strata=agefirstgr;
  keep caseid setno rstime rsentry strata; run;

* Create case-cohort risk sets;
proc sql;
  create table ccoh_rs as
    select *
      from failures, case_cohort
      where  (case_cohort.ccohentry < failures.rstime <=
case_cohort.ageexit) and (failures.strata = case_cohort.agefirstgr);
quit;

* Find the subject who is the failure (case) in this risk set;
data analytic;
  set ccoh_rs;
  cc = caseid eq id; drop caseid; run;


*Establish maximum risk set size;
proc freq data= analytic;
table setno / noprint out=_ncovals (rename=(count=_ncovals)
drop=percent);run;

proc sql noprint;
  select max(_ncovals)
    into :maxsetsize
    from _ncovals;
quit;

%let maxsetsize= &maxsetsize;
```

```
data long (keep=boot setno cc exposure weight);
  set analytic ;
  array wt{&nboot} wt1-wt&nboot;
  do r = 1 to &nboot;  boot=r; weight = wt{r}  ;  output; end; run;


proc sort data=long ; by boot setno descending cc; run;
```

```
*Calculate length of array for 2 explanatory variables, dcat1 and dcat2;
 %let arraysize=%sysevalf(&maxsetsize*2);

*Transform to one record per risk set;
data riskset (keep=boot setno z1-z&arraysize w1-w&maxsetsize  ntot);
  set analyzeLONG;
  by boot setno;
  array z{2,&maxsetsize};
  array w{&maxsetsize};
  retain i z1-z&arraysize w1-w&maxsetsize;
  if first.setno then do;
     do i=1 to &maxsetsize; z{1,i} =.;z{2,i} =.;w{i} =.; end; i = 0;end;
  i = i + 1;  z{1,i} = dcat1; z{2,i} = dcat2; * enter name for explanatory
variable;
  w{i} = weight; ntot=i; if last.setno then output; run;


ods select  parameterestimates; ods output parameterestimates=ests  ;
 proc nlmixed data= riskset;
  parms beta1=0, beta2=0 ;
   array z{2,&maxsetsize} z1-z&arraysize;
  array w{&maxsetsize} w1-w&maxsetsize;
   cc =1;   sum=0;  ntot=ntot;
  do i = ntot to 1 by -1;
    Phi= exp( z{1,i}*beta1 +z{2,i}*beta2   );
    sum = sum + ( Phi   * w{i}) ;
   caseprod =  Phi **   w{i}     ;
  end;
  L=caseprod / sum**   w{1} ;
  model cc ~ general(log(L));  by boot;  run;

data rout1;  set ests; if Parameter='beta1'; e_beta=  exp(Estimate);    run;

      title2 'Estimate and 95% limits by percentile';
      proc univariate data=rout1 noprint;
            var e_beta  ;
            output out=cis1_pct pctlpts=2.5 50 97.5 pctlpre=cis; run;
            data cis1_pct;
                  set cis1_pct;
                  l95 =  (cis2_5);
                  est =  (cis50);
                  u95 =  (cis97_5); run;
            proc print data=cis1_pct noobs; var l95 est u95; run;

title2 'Estimate and 95% limits by normality assumption';
      proc means data=rout1 mean std noprint;
```

```
var Estimate;
output out=means; run;
proc sql;
      create table b as select Estimate as mean
      from means where _stat_="MEAN";
      create table c as select Estimate as stderr
      from means where _stat_="STD";
      select      exp(b.mean-1.96*c.stderr) as l95,
                exp(b.mean) as est,
                exp(b.mean+1.96*c.stderr) as u95
      from b,c; quit; title2;
```