Estimated Fish Consumption Rates for the U.S. Population and Selected Subpopulations (NHANES 2003-2010)

Appendix D

EPA Methodd SAS Code

The SAS code for the EPA method was implemented with two SAS macros, similar to subroutines, named "GetMaxCorrLambda" "EPAFCRMethodKernel". The EPAFCRMethodKernel macro also calls the "GetUniqueNames" macro. All three macros are provided below. The GetUniqueNames macro extract unique variable names from the input strings and creates a space delimited list of those names in a macro variable.

The GetMaxCorrLambda macro finds the value of lambda (λ^*) used for the transformation before fitting the models. The macro tries various values of λ^* and selects the one with the highest correlation with the corresponding quantiles from a normal distribution. The range of lambda values considered was adjusted to locate the value with the maximum correlation (values from -0.2 to 0.3 were used). The macro can create plots of correlation versus lambda and a QQPlot for the transformed data using the selected lambda. The input parameters are:

- InFile = the file name with the reported fish consumption derived from the NHANES data. The input dataset must include the full sample weight variable from the NHANES files.
- Wgt = the name of the full sample weight variable to use.
- MinLambda = the minimum value of lambda to consider.
- MaxLambda = the minimum value of lambda to consider.
- LambdaStep = the difference between successive lambda values (always .01)
- ShowPlots: if ShowPlots = Yes, the macro created plots to helps assess the results.

The EPAFCRMethodKernel macro calculates the parameter estimates from the EPA method and simulates the usual fish consumption. The input parameters are:

- InFile = the file name with the reported fish consumption derived from the NHANES data. The input dataset must include the variance strata and PSU variables from the NHANES files (sdmvstra and sdmvpsu) and the sampling weights.
- Y = the dependent variable
- lambda = the values of λ^* to use for the Box-Cox transformation (determined in the GetMaxCorrLambda macro)
- Wgt = the variable name of the full sample or replicate weight to use

- OtherProbDepVars = the variable names of independent predictors for the probability model, not including Weekend and Seq2 (an indicator of the second recall) which are in all models. The list of variables can include class variables and interactions (using "*").

- OtherAmtDepVars = the variable names of independent predictors for the amount model, not including Weekend and Seq2 (an indicator of the second recall) which are in all models. The list of variables can include class variables and interactions (using "*").

- ClassVars = the names of any class variables (categorical variables) among the predictors defined by OtherProbDepVars and OtherAmtDepVars.

- NSim = the number of simulated values of usual fish consumption for each unique respondent (100 was used for the final runs).

The GetMaxCorrLambda macro was run using the full sample weight to get the transformation to use. The EPAFCRMethodKernel macro is run using the full sample weight to get the parameter estimates. The EPAFCRMethodKernel macro is then run using each replicate weight to get the confidence intervals. The macro output from the runs is subsequently processed by other SAS code and macros to get the percentiles by demographic characteristics and the confidence intervals.

```
Title1 'GetUniqueNames Macro';
%macro GetUniqueNames(Str1 = , Str2 = , Str3 = , OutStrName = UniqueNames);
%global &OutStrName;
%let &OutStrName = ;
Data _null_;
      length Str $10000 UniqueVars $2000 u n $200;
      Str = Catx(' ',&Str1", " &Str2", " &Str3");
      NU = 0;
      do i = 1 to 100 until (N = ' ');
            N = scan(Str,i);
            Old = 0;
            do j = 1 to NU;
                  U = scan(UniqueVars,j);
                  if upcase(N) = upcase(U) then Old = 1;
            end;
            if Old = 0 then do; UniqueVars = catx(' ',UniqueVars, N); NU = NU
+ 1; end;
      end;
      Call Symput("&OutStrName",strip(UniqueVars));
run;
%mend;


Title1 'GetMaxCorrLambda Macro';
%macro GetMaxCorrLambda(Infile = , Y = , Wgt = 1, MinLambda = .01, MaxLambda
= .30, LambdaStep = .01, ShowPlots = No);
%global MaxCorrLambda;
%let MaxCorrLambda = .;
Data _Lambda(keep = &Y _Wgt_);
      retain _SumWgt_ 0 _NGt0_ 0 _GM_ 0;
      set &InFile end=eof;
      if &Y>0 then do;
            _Wgt_ = &Wgt;
            _SumWgt_ = _SumWgt_ + _Wgt_;
            _NGt0_ = _NGt0_ + 1;
            _GM_ = _GM_ + _Wgt_*log(&Y);
            output;
            end;
      if eof then do;
            _GM_ = exp(_GM_/_SumWgt_) ;
            call symput('_SumWgt_',strip(put(_SumWgt_,best.)));
            call symput('_NGt0_',strip(put(_NGt0_,best.)));
            call symput('_GM_',strip(put(_GM_,best.)));
            end;
run;
%put GetMaxCorrLambda: _NGt0_ = &_NGt0_  _SumWgt_ = &_SumWgt_  _GM_ = &_GM_;
proc sort data = _Lambda; by &Y; run;
proc summary data = _Lambda;
      var _Wgt_;
      by &Y;
      output out = _Lambda(drop = _freq_ _type_) sum = ;
run;
Data _Lambda;
      retain _Sum_ 0;
      set _Lambda;
      _Sum_ = _Sum_ + _Wgt_;
      _P_ = (_Sum_ - _Wgt_/2)/&_SumWgt_;
```

```
      _Z_ = quantile('Normal',_P_,0,1);
run;
Data _Lambda1;
      set _Lambda end=eof;
      if &Y>0 then do;
            do _Lambda_ = &MinLambda to &MaxLambda by &LambdaStep;
                  if round(_Lambda_,0.001) = 0 then _T_ = log(&Y)*&_GM_;
                  else _T_ = (&Y**_Lambda_ -1)/(_Lambda_*&_GM_**(_Lambda_ -
1));
                  output;
            end;
            end;
run;
Proc sort data = _Lambda1; by _Lambda_ _T_; run;

proc corr data = _Lambda1 outp = _CorrOutP noprint ;
      var _Z_ _T_;
      weight _Wgt_;
      by _Lambda_;
run;
Data _CorrOutP(keep = _Lambda_ _T_);
      set  _CorrOutP;
      if _TYPE_ = 'CORR' and _NAME_ = '_Z_';
run;
Proc sort data = _CorrOutP out = trash; by descending _T_; run;
Data _null_;
      set trash;
      if _N_ = 1 then do;
            if abs(_Lambda_) < &LambdaStep/2 then _Lambda_ = &LambdaStep/2;
            call symput('MaxCorrLambda',strip(put(_Lambda_,best.)));
            end;
run;
%put GetMaxCorrLambda: Global MaxCorrLambda = &MaxCorrLambda;

%if &ShowPlots = Yes %then %do;
Title1 "Corr(_Z_,Transformed(&Y)) versus _Lambda_ (MaxCorrLambda =
&MaxCorrLambda)";
ods graphics / imagename = "CorrByLambda&Y" reset = index;
proc sgplot data =  _CorrOutP;
      series X = _Lambda_ Y = _T_;
      xaxis grid;
run;
ods graphics / imagename = "trash" reset = index;
Title1 "MaxCorrLambda QQPlot for Lambda = &MaxCorrLambda";
ods graphics / imagename = "QQPlotAtMaxCorr&Y" reset = index;
proc sgplot data =  _Lambda1;
      series X = _Z_ Y = _T_;
      xaxis grid;
      where Abs(_Lambda_ - &MaxCorrLambda)<0.0001;
run;
ods graphics / imagename = "trash" reset = index;
%end;
Title1;
%mend;


Title1 'EPAFCRMethodKernel ';
```

```
%macro EPAFCRMethodKernel(InFile = Dat4Fit, Y = FWEst24, lambda = .15, Wgt =
wtdrd1,
      OtherProbDepVars = ProbFreq ,
      OtherAmtDepVars = AmtFreq ,
      ClassVars = ,
      NSim = 2);
Data _Time;
      EPAStartTime = datetime();
run;
Data EPAByVars; run;
Data EPA_Parms; run;

%GetUniqueNames(Str1 = &OtherProbDepVars, Str2 = &OtherAmtDepVars, OutStrName
= UniqueNames);
%put UniqueNames = &UniqueNames;

Title1 'Set up data, transform amuont';
proc sort data = &InFile out = _analysis; by seqn seq2 ; run;
Data _analysis(keep = &Wgt &Y &UniqueNames &ClassVars Fish30 sdmvstra sdmvpsu
seqn
            tfish tfishGT0 Consumer ProbFreq AmtFreq weekend seq2)
      _Stats(keep = MinAmt NSubjects NTotalRecalls NSubjWith2Recalls
NSubjWith2FishRecalls);
      retain MinAmt . NSubjects 0 NTotalRecalls 0 NSubjWith2Recalls 0
NSubjWith2FishRecalls 0 NRecalls 0 NFishRecalls 0;
      set _analysis end=eof;
      by seqn;
      if first.seqn then do; NRecalls = 0; NFishRecalls = 0; end;
      if      &Y < 0 then do; Consumer = . ;tfish = .; tfishGT0 = .; end;
      else if &Y = 0 then do; Consumer = 0 ;tfish = .; tfishGT0 = .;
NTotalRecalls + 1; NRecalls + 1; end;
      else if &Y>0 then do;
            NTotalRecalls + 1; NRecalls + 1; NFishRecalls + 1;
            Consumer = 1;
            tfish = (&Y**&lambda -1)/&lambda;
            tfishGT0 = tfish;
            if MinAmt = . then MinAmt = &Y; else MinAmt = Min(MinAmt, &Y);
            end;
      if last.seqn then do;
            if NRecalls > 0 then NSubjects + 1;
            if NRecalls = 2 then NSubjWith2Recalls + 1;
            if NFishRecalls = 2 then NSubjWith2FishRecalls + 1;
            end;
      ProbFreq = log(Fish30+ .1);
      AmtFreq = Fish30**.45;
      output _analysis;
      if eof then do;
            call symput('MinAmt',strip(put(MinAmt,best.)));
            call symput('NSubjects',strip(put(NSubjects,best.)));
            call symput('NTotalRecalls',strip(put(NTotalRecalls,best.)));
            call
symput('NSubjWith2Recalls',strip(put(NSubjWith2Recalls,best.)));
            call
symput('NSubjWith2FishRecalls',strip(put(NSubjWith2FishRecalls,best.)));
            output _Stats;
            end;
run;
```

```
%put EPAFCRMethod:  MinAmt = &MinAmt;

Title1 'Survey logistic for probability model';
Data SLout SL_PE; run;
Proc surveylogistic data = _analysis;
      Class &ClassVars;
      model Consumer(event='1') = &OtherProbDepVars weekend seq2;
      weight &Wgt;
      strata sdmvstra;
      cluster sdmvpsu;
      output out = SLout p = predP xbeta = xbeta;
      ods output ParameterEstimates = SL_PE;
run;

Title1 'NLMixed for probability model variance component';
Data NL_ProbPE; length Parameter $32; run;
Data NL_ProbPE1; length Label $32; run;
proc nlmixed data = SLout;
      **parms XBFac=1 VAR_U1=1;
      parms XBFac=1 LOGSDU1 = 0;
      bounds LOGSDU1 > -25;
      VAR_U1 = exp(LOGSDU1*2);
      eta = XBFac*xbeta+PBet;
      p = 1/(1+exp(-eta));
      model Consumer ~ binomial(1,p)  ;
      random PBet ~ normal(0,VAR_U1) subject = seqn;*** out = PBetEst;
      replicate &Wgt ;
      **estimate "LOGSDU1" Log(VAR_U1)/2 ;
      estimate "VAR_U1" exp(LOGSDU1*2) ;
      ods output ParameterEstimates = NL_ProbPE;
      ods output AdditionalEstimates  = NL_ProbPE1;
run;
Data NL_ProbPE2;
      length Parameter $32;
      set NL_ProbPE NL_ProbPE1(rename=(Label=Parameter));
RUN;
Data _null_; set NL_ProbPE; if Parameter = 'XBFac' then call
symput('P_XBFac',strip(put(Estimate,best.))); run;
%put EPAFCRMethod:  P_XBFac = &P_XBFac;

Title1 'Survey Reg for amount model';
Data SRout NR_PE; run;
Proc surveyreg data = SLout;
      Class &ClassVars;
      model tfishGT0 = &OtherAmtDepVars Weekend Seq2 / Solution;
      weight &Wgt;
      strata sdmvstra;
      cluster sdmvpsu;
      output out = SRout p = predA ;
      ods output ParameterEstimates = NR_PE;
run;

Title1 'NLMixed for amount model variance components';
Data NL_AmtPE; length Parameter $32; run;
Data NL_AmtPE1; length Label $32; run;
proc nlmixed data = SRout;
      **parms VAR_U2=.25 VAR_E=.25;
```

```
      parms LOGSDE = -.5 LOGSDU2 = -.5;
      bounds LOGSDE LOGSDU2 > -25;
      VAR_E = exp(LOGSDE*2);
      VAR_U2 = exp(LOGSDU2*2);
      eta = predA+ABet;
      model TFishGT0 ~ normal(eta,VAR_E)  ;
      random ABet ~ normal(0,VAR_U2) subject = seqn;
      replicate &Wgt ;
      **estimate "LOGSDE" Log(VAR_E)/2 ;
      **estimate "LOGSDU2" Log(VAR_U2)/2 ;
      estimate "VAR_E" exp(LOGSDE*2) ;
      estimate "VAR_U2" exp(LOGSDU2*2) ;
      ods output ParameterEstimates = NL_AmtPE;
      ods output AdditionalEstimates  = NL_AmtPE1;
run;
Data NL_AmtPE2;
      length Parameter $32;
      set NL_AmtPE NL_AmtPE1(rename=(Label=Parameter));
RUN;

Title1 'Collect parameter estimates into one file';
Data _Parms;
      Length ClassVal0 ClassVal1 $8 ;
      Array levels ClassVal0 ClassVal1;
      length Parameter $32;
      set SL_PE(in=a) NL_ProbPE2(in=b) NR_PE(in=c) NL_AmtPE2(in=d);
      if a then do; Parameter = "P_"||Variable ; Parameter = catx('
',Parameter,ClassVal0,ClassVal1); PValue = ProbChiSq;
                        EPAEstimate = Estimate*&P_XBFac; EPAStdErr =
StdErr*&P_XBFac; end;
      if b then do; Parameter = "P_"||Parameter; PValue = .; StdErr =
StandardError;
                        EPAEstimate = Estimate; EPAStdErr = .; end;
      if C then do; Parameter = "A_"||Parameter;  PValue = Probt;
                        EPAEstimate = Estimate; EPAStdErr = StdErr; end;
      if D then do; Parameter = "A_"||Parameter;  PValue = .; StdErr =
StandardError;
                        EPAEstimate = Estimate; EPAStdErr = .; end;
      Parameter = Upcase(Parameter);
      format df;
run;

Title1 'Simulate fish consumption';
%let CountMin = .;
Title1 'Simulate usual intake';
Data simtfish(keep = seqn sim usualintake Weight WgtName probConsumer
SimUFish);
      retain CountMin 0 PWeekend 0 AWeekend 0 XBFac 0 SigPBet 0 SigABet 0
SigAWithin 0;
      length WgtName $32;
      set _Parms(in=a) SRout(in=b drop = tfish) end = eof;
      if a then do;
            Parameter = upcase(parameter);
            if Parameter = 'P_WEEKEND' then PWeekend = Estimate;
            if Parameter = 'A_WEEKEND' then AWeekend = Estimate;
            if Parameter = 'P_XBFAC  ' then XBFac = Estimate;
            if Parameter = 'P_VAR_U1' then SigPBet = sqrt(Estimate);
```

```
            if Parameter = 'A_VAR_U2' then SigABet = sqrt(Estimate);
            if Parameter = 'A_VAR_E' then SigAWithin = sqrt(Estimate);
            end;
      if eof then call symput("CountMin",strip(put(CountMin,best.)));
      if b and Seq2 = 0;
      WgtName = "&Wgt";
      Weight = &Wgt;
      do sim = 1 to &nsim;
            if SigPBet = 0 then rep = 0;
            else rep = rand('normal',0,SigPBet);
            if SigABet = 0 then rea = 0;
            else rea = rand('normal',0,SigABet);
            Xbeta4Pred = XBFac*(xbeta - Pweekend*weekend + PWeekend*3/7) +
rep;
            probConsumer = logistic(Xbeta4Pred);
            SimUTFish = predA -AWeekend*Weekend + AWeekend*3/7 + rea ;
            if SimUTFish*&lambda+1 > 0 then SimUFish =
(SimUTFish*&lambda+1)**(1/&lambda) +0.5*SigAWithin**2*(1-
&lambda)*(SimUTFish*&lambda+1)**(1/&lambda-2);
            else do; SimUFish = &MinAmt/2; CountMin + 1; end;
            usualintake = probConsumer*SimUFish;
            output;
      end;
run;

Title1 'Get final parameter values';
Data _Time;
      set _Time;
      EPAEndTime = datetime();
      EPASeconds = EPAEndTime - EPAStartTime;
run;
Data EPA_Parms(keep = Parameter Estimate StdErr EPAEstimate EPAStdErr
PValue);
      set _Parms(in=a) _Time(in=b) end=eof;
      if a then output;
      if b then do; Parameter = 'EPASeconds'; EPAEstimate = EPASeconds;
output; end;
      if eof then do;
            Estimate = .; StdErr = .; EPAStdErr = .;
            Parameter = 'MIN_AMT'; EPAEstimate = &MinAmt; output;
            Parameter = 'CountMinAmt'; EPAEstimate = &CountMin; output;
            Parameter = 'NSubjects'; EPAEstimate = &NSubjects; output;
            Parameter = 'NTotalRecalls'; EPAEstimate = &NTotalRecalls;
output;
            Parameter = 'NSubjWith2Recalls'; EPAEstimate =
&NSubjWith2Recalls; output;
            Parameter = 'NSubjWith2FishRecalls'; EPAEstimate =
&NSubjWith2FishRecalls; output;
            Parameter = 'A_LAMBDA'; EPAEstimate = &Lambda; output;
            end;
run;
Proc print data = EPA_Parms;
      var Parameter Estimate EPAEstimate EPAStdErr PValue;
run;

%mend;
```