SUPPLEMENTARY MATERIAL

A generalized Bayesian nonlinear mixed effects regression model for zero inflated longitudinal count data in tuberculosis trials

Divan Aristo Burger, Robert Schall, Rianne Jacobs, Ding-Geng Chen

1 ZERO INFLATED GENERALIZED POISSON REGRESSION MODEL

The likelihood function of the zero inflated generalized Poisson regression model is given by:

$$\begin{aligned} & L\left(\phi_{ij}, \theta_j, \tau_{jk}, \pi_{jk} \middle| y_{ijk}\right) \\ =& \pi_{jk} I\left(y_{ijk}=0\right) + \left(1-\pi_{jk}\right)\left(1-\omega_{jk}\right)\lambda_{ijk}\frac{\left[\left(1-\omega_{jk}\right)\lambda_{ijk}+\omega_{jk}y_{ijk}\right]^{y_{ijk}-1}}{y_{ijk}!}e^{-\left[\left(1-\omega_{jk}\right)\lambda_{ijk}+\omega_{jk}y_{ijk}\right]} \end{aligned}$$

where $E\left(y_{ijk}\right)=\left(1-\pi_{jk}\right)\lambda_{ijk}$ & $\mathrm{Var}\left(y_{ijk}\right)=\left(1-\pi_{jk}\right)\left[\lambda_{ijk}\left(1-\omega_{jk}\right)^{-2}+\pi_{jk}\left[\left(1-\pi_{jk}\right)\lambda_{ijk}\right]^2\right]$. The remainder model components (y_{ijk}, λ_{ijk} and π_{jk}) are analogous to those specified in the main paper. The dispersion parameters ω_{jk} are assumed to follow vague beta prior distributions, namely $\omega_{jk} \sim \mathrm{Beta}(0.1, 0.1)$.

2 SAS® CODE

```
option missing = " " msglevel = "I" nofmterr noquotelenmax;

%let ROOT = C:\Users\2004014359\Documents\PhD Thesis\Programs\Formal\SSCC;

%let BUGS = C:\Users\2004014359\Documents\PhD Thesis\Programs\BUGS;

filename MACROS "&BUGS."; /*DOWNLOAD FROM HTTP://WWW.PSYCHSTAT.ORG/US/ARTICLE.PHP/84.HTM*/
%include MACROS ("*.sas");

option notes;

%include "&ROOT.\Data\NC002\CFU.sas"; /*INCLUDE DATASET*/

/**************CFU DATASET*************/
/*-------------------------------------*/
/*VARIABLE   LABEL                     */
/*-------------------------------------*/
/*AVISIT     VISIT                     */
/*AVISITN    VISIT (NUMERIC)           */
/*CENSOR     CENSORING FLAG            */
/*CFU        CFU COUNT (CONVERTED)     */
/*DILFACT    DILUTION FACTOR           */
/*FACTOR     COMPENSATION FACTOR       */
/*KVTRTN     TREATMENT                 */
/*LOGCFU     LOG(CFU) COUNT            */
/*NSUBJID    SUBJECT NUMBER            */
/*NUMPLAT    NUMBER OF PLATE COUNTS    */
/*SUBJID     SUBJECT IDENTIFIER        */
/*SUMCFU     SUM OF PLATE COUNTS       */
/*TIME       COLLECTION TIME (DAYS)    */
/*TP         TIMEPOINT                 */
/**************************************/

%let OPENBUGS = C:/Program Files (x86)/OpenBUGS/OpenBUGS323;
%let OPENSHRT = C:/Progra~2/OpenBUGS/OpenBUGS323;
%let RSHORT = C:/Progra~1/R/R-3.1.0/bin;

libname INTERIM "&ROOT.\Interim Datasets";

%let MODELVAR = SUMCFU;

%let DATATYPE = TWO_CFU_O_ZINB;
%let DATATSHR = TCOZNB;

%let WDF = 3; /*WISHART DISTRIBUTION: DEGREES OF FREEDOM*/

%let INTERVAL = 7; /*TIME EXPRESSED IN WEEKS*/

data _NULL_; /*BOUNDS OF KAPPA*/
    call symput("LKAPPA", strip(put(3/&INTERVAL., BEST.)));
    call symput("UKAPPA", strip(put(11/&INTERVAL., BEST.)));
run;

proc format; /*FORMATS*/
    value KVTRTN
        1 = "M-PA100-Z"
        2 = "M-PA200-Z"
        3 = "Rifafour";
```

```
    value CENSOR
        0 = "Uncensored"
        1 = "Left-Censored"
        2 = "Right-Censored"
        3 = "Excluded";
run;

data DATA00; /*PROTOCOL TIMEPOINT (POOLED)*/
    do TIME = 0 to 56;
        if . < TIME <= 0 then PROTWIND = 1; else
        if 0 < TIME <= 3 then PROTWIND = 1; else
        if 3 < TIME <= 7 then PROTWIND = 1; else
        if 7 < TIME <= 14 then PROTWIND = 2; else
        if 14 < TIME <= 21 then PROTWIND = 2; else
        if 21 < TIME <= 28 then PROTWIND = 3; else
        if 28 < TIME <= 35 then PROTWIND = 3; else
        if 35 < TIME <= 42 then PROTWIND = 4; else
        if 42 < TIME <= 49 then PROTWIND = 4; else
        if 49 < TIME <= 56 then PROTWIND = 4;
        output;
    end;
run;

%_LEXPORT (DATA = DATA00, FILE = "&OPENBUGS./&DATATYPE._PROTWIND.txt", VAR = PROTWIND);

data DATA01;
    set CFU;
    by KVTRTN NSUBJID;
    PROTTIMN = input(compress(substr(AVISIT, index(AVISIT, "(") + 1), "DAY)"), BEST.) - 1;
    if . < PROTTIMN < 0 then PROTTIMN = 0;
    if PROTTIMN = 0 then PROTTIMC = 1; else
    if PROTTIMN = 3 then PROTTIMC = 1; else
    if PROTTIMN = 7 then PROTTIMC = 1; else
    if PROTTIMN = 14 then PROTTIMC = 2; else
    if PROTTIMN = 21 then PROTTIMC = 2; else
    if PROTTIMN = 28 then PROTTIMC = 3; else
    if PROTTIMN = 35 then PROTTIMC = 3; else
    if PROTTIMN = 42 then PROTTIMC = 4; else
    if PROTTIMN = 49 then PROTTIMC = 4; else
    if PROTTIMN = 56 then PROTTIMC = 4;
    CENSORF = CENSOR;
    if CENSORF = 2 then CENSORF = 0;
    format KVTRTN KVTRTN. CENSOR CENSORF CENSOR.;
    if CENSOR in (1 2) then CENSOR = 0;
    KAPPA = (&LKAPPA. + &UKAPPA.)/2;
    GAMMA = (2 + 0.05)/2;
    TIMEH1 = 1;
    TIMEH2 = -TIME/&INTERVAL.;
    TIMEH3 = -GAMMA*log((exp((TIME/&INTERVAL. - KAPPA)/GAMMA) + exp(-(TIME/&INTERVAL. - KAPPA)/GAMMA))/(exp((KAPPA)/GAMMA) + exp(-(KAPPA)/GAMMA)));
    call symput("NTRT", strip(put(KVTRTN, BEST.)));
    call symput("NSUBJ", strip(put(NSUBJID, BEST.)));
    OFFSET = log(NUMPLAT) - log(FACTOR) - log(10**DILFACT); /*OFFSET CONSTANT*/
    DUMMY = 1;
run;

proc glimmix data = DATA01 maxopt = 10000 inititer = 50 method = LAPLACE; /*FIT LINEAR MIXED MODEL TO OBTAIN HYPERPARAMETERS OF WISHART DISTRIBUTION*/
    class NSUBJID;
    model &MODELVAR. = TIMEH1 TIMEH2 TIMEH3 / noint link = LOG solution dist = POISSON offset = OFFSET cl;
    random TIMEH1 TIMEH2 TIMEH3 / subject = NSUBJID type = UN solution;
    nloptions technique = NEWRAP maxiter = 1000 gconv = 1E-8;
    ods output COVPARMS = DATA02;
run;
```

```
proc sort data = DATA01 (keep = KVTRTN PROTTIMC) out = PROT01 nodupkey;
    by KVTRTN PROTTIMC;
run;

%macro DOGLIMNEG; /*OBTAIN STARTING VALUES FOR RANDOM INTERCEPTS AND SLOPES*/

    %do i = 3 %to &NSUBJ.;

        proc glimmix data = DATA01 (where = (NSUBJID = &i.));
            by KVTRTN NSUBJID;
            model &MODELVAR. = TIMEH1 TIMEH2 TIMEH3 / noint link = LOG solution dist = NEGBIN offset = OFFSET;
            ods output PARAMETERESTIMATES = DATA04_1_&i.;
        run;
        quit;

    %end;

%mend;

%DOGLIMNEG;

data DATA04_1;
    set DATA04_1_:;
run;

proc datasets;
    delete DATA04_1_:;
run;

proc glimmix data = DATA01; /*STARTING VALUES*/
    by KVTRTN NSUBJID;
    model &MODELVAR. = TIMEH1 TIMEH2 TIMEH3 / noint link = LOG solution dist = POISSON offset = OFFSET;
    ods output PARAMETERESTIMATES = DATA04_2;
run;

proc sort data = DATA04_1;
    by KVTRTN NSUBJID EFFECT;
run;

proc sort data = DATA04_2;
    by KVTRTN NSUBJID EFFECT;
run;

data DATA04; /*SET MISSING INITIAL VALUES*/
    merge DATA04_1 (keep = KVTRTN NSUBJID EFFECT ESTIMATE rename = (ESTIMATE = EST1) where = (EFFECT ne "Scale"))
          DATA04_2 (keep = KVTRTN NSUBJID EFFECT ESTIMATE rename = (ESTIMATE = EST2));
    by KVTRTN NSUBJID EFFECT;
    if EST1 ne . then ESTIMATE = EST1; else ESTIMATE = EST2;
    if EFFECT = "TIMEH1" then do;
        if ESTIMATE > 20 then ESTIMATE = 20;
    end;
    else do;
        if ESTIMATE < -10 then ESTIMATE = -10;
        if ESTIMATE > 10 then ESTIMATE = 10;
    end;
run;

%put &NTRT.;
%put &NSUBJ.;

proc transpose data = DATA04 out = DATA05 (drop = _NAME_ rename = (TIMEH1 = SALPHA TIMEH2 = SBETA1 TIMEH3 = SBETA2));
    by NSUBJID;
    var ESTIMATE;
```

```
    id EFFECT;
run;

data DATA06 (drop = i); /*HYPERPARAMETERS OF WISHART DISTRIBUTION*/
    set DATA02;
    ESTC = strip(put(3*ESTIMATE, BEST.));
    do i = 1 to &NTRT.;
        if COVPARM = "UN(1,1)" then call symput("WAT" || strip(put(i, BEST.)), "5");
        if COVPARM = "UN(2,2)" then call symput("WB1T" || strip(put(i, BEST.)), "5");
        if COVPARM = "UN(3,3)" then call symput("WB2T" || strip(put(i, BEST.)), "5");
        if COVPARM = "UN(2,1)" then call symput("WAB1T" || strip(put(i, BEST.)), "0");
        if COVPARM = "UN(3,1)" then call symput("WAB2T" || strip(put(i, BEST.)), "0");
        if COVPARM = "UN(3,2)" then call symput("WB1B2T" || strip(put(i, BEST.)), "0");
    end;
run;

%put &WAT1.;
%put &WB1T1.;
%put &WB2T1.;
%put &WAB1T1.;
%put &WAB2T1.;
%put &WB1B2T1.;

%macro INITIAL (DISPER, NO); /*EXPORT INITIAL VALUES FOR MCMC CHAINS (OPENBUGS)*/

    data DATA07;
        merge DATA01 DATA05;
        by NSUBJID;
        if first.NSUBJID then do;
            if "&DISPER." = "Y" then do;
                SMU1 = SALPHA + 0;
                SMU2 = SBETA1;
                SMU3 = SBETA2;
                SKAPPA = 1;
                SGAMMA = 1;
            end; else
            if "&DISPER." = "N" then do;
                SMU1 = SALPHA;
                SMU2 = SBETA1;
                SMU3 = SBETA2;
                SKAPPA = 1;
                SGAMMA = 1;
            end;
        end;
    run;

    proc sort data = DATA07;
        by KVTRTN;
    run;

    proc means data = DATA07 noprint;
        by KVTRTN;
        var SMU1 - SMU3 SKAPPA SGAMMA;
        output out = DATA08 (drop = _TYPE_ _FREQ_) mean = MMU1 - MMU3 MKAPPA MGAMMA;
    run;

    data _NULL_;
        if "&DISPER." = "Y" then call symput("OMG", strip(put(1, BEST.))); else
        if "&DISPER." = "N" then call symput("OMG", strip(put(1, BEST.)));
    run;

    data _NULL_;
        length VAR $999;
```

```
    VAR = "";
    do i = 1 to &NTRT.;
        VAR = strip(VAR) || " &OMG., 0, 0, 0, &OMG., 0, 0, 0, &OMG.,";
    end;
    VAR = compbl(VAR);
    call symput("MOMGINV", substr(VAR, 1, length(VAR) - 1));
run;

%put &MOMGINV.;

data _NULL_;
    file "&OPENBUGS./&DATATYPE._Biphasic_Initial_MOMGINV_&NO..txt" linesize = 5000;
    put #01 "list(MOMGINV = structure(.Data = c(&MOMGINV.), .Dim = c(&NTRT., 3, 3)))";
run;

proc sort data = DATA07 (where = (SMU1 ne .));
    by NSUBJID;
run;

%_SEXPORT (DATA = DATA07, VAR = SMU1-SMU3, FILE = "&OPENBUGS./&DATATYPE._Biphasic_Initial_SMU_&NO..txt");

%_SEXPORT (DATA = DATA08, VAR = MMU1-MMU3, FILE = "&OPENBUGS./&DATATYPE._Biphasic_Initial_MMU_&NO..txt");
%_LEXPORT (DATA = DATA08, VAR = MKAPPA, FILE = "&OPENBUGS./&DATATYPE._Biphasic_Initial_MKAPPA_&NO..txt");
%_LEXPORT (DATA = DATA08, VAR = MGAMMA, FILE = "&OPENBUGS./&DATATYPE._Biphasic_Initial_MGAMMA_&NO..txt");

%macro INIT (VAR, VAL);

    data _NULL_;
        length VAR $999;
        VAR = "";
        do i = 1 to &NTRT.;
            VAR = strip(VAR) || " &VAL.,";
        end;
        VAR = compbl(VAR);
        call symput("V", substr(VAR, 1, length(VAR) - 1));
    run;

    data _NULL_;
        file "&OPENBUGS./&DATATYPE._Biphasic_Initial_&VAR._&NO..txt" linesize = 5000;
        put #01 "list(&VAR. = c(&V.))";
    run;

%mend;

data _NULL_;
    if "&DISPER." = "Y" then do;
        call symput("RH", strip(put(4, BEST.)));
        call symput("PZ", strip(put(0.5, BEST.)));
    end; else
    if "&DISPER." = "N" then do;
        call symput("RH", strip(put(4, BEST.)));
        call symput("PZ", strip(put(0.5, BEST.)));
    end;
run;

%macro INITMAT (VAR, VAL);

    %global MAXPROT;

    data PROT02;
        set PROT01;
        by KVTRTN PROTTIMC;
        if last.PROTTIMC then call symput("MAXPROT", strip(put(PROTTIMC, BEST.)));
```

```
            VAL = &VAL.;
        run;

        proc transpose data = PROT02 out = PROT03 (drop = _NAME_) prefix = &VAR.;
            by KVTRTN;
            var VAL;
            id PROTTIMC;
        run;

        %_SEXPORT (DATA = PROT03, VAR = &VAR.1-&VAR.&MAXPROT., FILE = "&OPENBUGS./&DATATYPE._Biphasic_Initial_&VAR._&NO..txt");

    %mend;

    %INITMAT (VAR = RHO, VAL = &RH.);
    %INITMAT (VAR = PZERO, VAL = &PZ.);

%mend;

%INITIAL (DISPER = Y, NO = 1);
%INITIAL (DISPER = N, NO = 2);

proc sort data = DATA01 out = DATA09;
    by CENSOR NSUBJID TIME;
run;

%let CS1 = NA;
%let CE1 = NA;

data DATA10; /*DETERMINE INDICES FOR CENSORED DATA (LOGNORMAL MODEL ONLY)*/
    set DATA09;
    by CENSOR NSUBJID;
    if CENSOR = 1 then do;
        C = &MODELVAR.; &MODELVAR. = .;
    end;
    if first.CENSOR then call symput("CS" || strip(put(CENSOR, BEST8.)), strip(put(_N_, BEST.)));
    if last.CENSOR then call symput("CE" || strip(put(CENSOR, BEST8.)), strip(put(_N_, BEST.)));
    DATA = &MODELVAR.;
    U = 0;
run;

%put &CS0. &CE0. &CS1. &CE1.;

%_LEXPORT (DATA = DATA10, FILE = "&OPENBUGS./&DATATYPE._DATA.txt", VAR = NSUBJID KVTRTN TIME PROTTIMC &MODELVAR. OFFSET C DATA); /*EXPORT DATASET FOR OPENBUGS*/

%_LEXPORT (DATA = DATA10, FILE = "&OPENBUGS./&DATATYPE._Biphasic_Initial_U.txt", VAR = U);

proc sort data = DATA10 (keep = NSUBJID KVTRTN SUBJID) out = DATA11 nodupkey;
    by NSUBJID KVTRTN;
run;

proc transpose data = DATA11 out = DATA12 (drop = _NAME_) prefix = PAT;
    var KVTRTN;
    id NSUBJID;
    format KVTRTN;
run;

data _NULL_; /*TREATMENT GROUP ASSIGNED TO PATIENTS*/
    set DATA12;
    length VAR $999;
    array PAT PAT:;
    do i = 1 to dim(PAT);
        VAR = strip(VAR) || ", " || strip(put(PAT(i), BEST.));
    end;
```

```
    VAR = compbl(VAR);
    call symput("PATKVT", strip(substr(VAR, 3, length(VAR) - 1)));
run;

%put &PATKVT.;

data _NULL_;
    file "&OPENBUGS./&DATATYPE._PATKVT.txt" linesize = 5000;
    put #01 "list(PATKVT = c(&PATKVT.))";
run;

%macro OPENBUGS; /*ZINB REGRESSION MODEL (OPENBUGS)*/

    data _NULL_;
        file "&OPENBUGS./&DATATYPE._Biphasic_Model.txt";
        put "model {";
        put "    for (i in &CS0.:&CE0.) {";
        put "        &MODELVAR.[i] ~ dnegbin(P[i], RHO[KVTRTN[i], PROTTIMC[i]])"; /*SPECIFICATION OF ZINB DISTRIBUTION*/
        put "        P[i] <- RHO[KVTRTN[i], PROTTIMC[i]]/(RHO[KVTRTN[i], PROTTIMC[i]] + (1 - U[i])*MEAN[i])";
        put "        U[i] ~ dbern(PZERO[KVTRTN[i], PROTTIMC[i]])";
        put "        LINK[i] <- SMU[NSUBJID[i], 1] - SMU[NSUBJID[i], 2]*TIME[i]/&INTERVAL. - SMU[NSUBJID[i], 3]*MGAMMA[KVTRTN[i]]*";
        put "                   log((exp((TIME[i]/&INTERVAL. - MKAPPA[KVTRTN[i]])/MGAMMA[KVTRTN[i]]) + exp(-(TIME[i]/&INTERVAL. - MKAPPA[KVTRTN[i]])/";
        put "                   MGAMMA[KVTRTN[i]])))/(exp((MKAPPA[KVTRTN[i]])/MGAMMA[KVTRTN[i]]) + exp(-(MKAPPA[KVTRTN[i]])/MGAMMA[KVTRTN[i]])))";
        put "        log(MEAN[i]) <- LINK[i] + OFFSET[i]"; /*NONLINEAR FUNCTION OVER TIME*/
        put "        NEGMAR[i] ~ dnegbin(PMAR[i], RHO[KVTRTN[i], PROTTIMC[i]])";
        put "        PMAR[i] <- RHO[KVTRTN[i], PROTTIMC[i]]/(RHO[KVTRTN[i], PROTTIMC[i]] + min(MEAN[i], 10000))"; /*CAP MEAN TO AVOID NUMERICAL OVERFLOW*/
        put "        PPO[i] <- PZERO[KVTRTN[i], PROTTIMC[i]]*equals(DATA[i], 0) + (1 - PZERO[KVTRTN[i], PROTTIMC[i]])*density(NEGMAR[i], DATA[i])"; /*USED TO CALCULATE DIC
      STATISTIC OUTSIDE OPENBUGS*/
        put "        LOGL[i] <- log(PPO[i])";
        put "        EXPY[i] <- (1 - PZERO[KVTRTN[i], PROTTIMC[i]])*MEAN[i]"; /*QUANTITIES FOR PEARSON GOODNESS OF FIT MEASURE*/
        put "        VARY[i] <- (1 - PZERO[KVTRTN[i], PROTTIMC[i]])*(MEAN[i]*(MEAN[i] + RHO[KVTRTN[i], PROTTIMC[i]])/RHO[KVTRTN[i], PROTTIMC[i]] + PZERO[KVTRTN[i], PROTTIMC[
      i]]*MEAN[i]*MEAN[i])";
        put "        PRES[i] <- (&MODELVAR.[i] - EXPY[i])/sqrt(VARY[i])";
        put "        PNEW[i] <- RHO[KVTRTN[i], PROTTIMC[i]]/(RHO[KVTRTN[i], PROTTIMC[i]] + (1 - U[i])*min(MEAN[i], 10000))";
        put "        YNEW[i] ~ dnegbin(PNEW[i], RHO[KVTRTN[i], PROTTIMC[i]])";
        put "        PRESNEW[i] <- (YNEW[i] - EXPY[i])/sqrt(VARY[i])";
        put "        SQRES[i] <- PRES[i]*PRES[i]";
        put "        SQRESNEW[i] <- PRESNEW[i]*PRESNEW[i]";
        put "    }";
        put "    for (i in 1:&NSUBJ.) {"; /*RANDOM EFFECTS*/
        put "        SALPHA[i] <- SMU[i, 1]";
        put "        SBETA1[i] <- SMU[i, 2]";
        put "        SBETA2[i] <- SMU[i, 3]";
        put "        SMU[i, 1:3] ~ dmnorm(MMU[PATKVT[i], 1:3], MOMGINV[PATKVT[i], 1:3, 1:3])";
        put "        for (j in 1:57) {";
        put "            SLOGMEAN[i, j] <- SMU[i, 1] - SMU[i, 2]*(j - 1)/&INTERVAL. - SMU[i, 3]*MGAMMA[PATKVT[i]]*log((exp(((j - 1)/&INTERVAL. - MKAPPA[PATKVT[i]])/MGAMMA[
      PATKVT[i]]) +";
        put "                              exp(-((j - 1)/&INTERVAL. - MKAPPA[PATKVT[i]])/MGAMMA[PATKVT[i]]))/(exp((MKAPPA[PATKVT[i]])/MGAMMA[PATKVT[i]]) + exp(-(MKAPPA[
      PATKVT[i]])/MGAMMA[PATKVT[i]])))";
        put "            SLAMBDA[i, j] <- exp(SLOGMEAN[i, j])";
        put "            SEXPECT[i, j] <- (1 - PZERO[PATKVT[i], PROTWIND[j]])*SLAMBDA[i, j]";
        put "            LOGCFU[i, j] <- log(SEXPECT[i, j])/log(10)";
        put "        }";
        put "        SEBA0056[i] <- -(log(SEXPECT[i, 57]) - log(SEXPECT[i, 1]))/(56*log(10))"; /*EBA PER PATIENT*/
        put "    }";
        put "    for (i in 1:&NTRT.) {"; /*PRIOR DISTRIBUTIONS*/
        put "        for (j in 1:&MAXPROT.) {";
        put "            RHO[i, j] ~ dgamma(0.1, 0.1)";
        put "            PZERO[i, j] ~ dbeta(0.1, 0.1)";
        put "        }";
        put "        MALPHA[i] <- MMU[i, 1]";
        put "        MBETA1[i] <- MMU[i, 2]";
        put "        MBETA2[i] <- MMU[i, 3]";
```

```
        put "        MOMGINV[i, 1:3, 1:3] ~ dwish(R[i, 1:3, 1:3], 3)";
        put "        MOMEGA[i, 1:3, 1:3] <- inverse(MOMGINV[i, 1:3, 1:3])";
        put "        ALPSIGSQ[i] <- MOMEGA[i, 1, 1]";
        put "        BT1SIGSQ[i] <- MOMEGA[i, 2, 2]";
        put "        BT2SIGSQ[i] <- MOMEGA[i, 3, 3]";
        put "        ALPBT1SIGSQ[i] <- MOMEGA[i, 1, 2]";
        put "        ALPBT2SIGSQ[i] <- MOMEGA[i, 1, 3]";
        put "        BT1BT2SIGSQ[i] <- MOMEGA[i, 2, 3]";
        put "        MMU[i, 1:3] ~ dmnorm(ZERO[1:3], I[1:3, 1:3])";
        put "        MKAPPA[i] ~ dunif(&LKAPPA., &UKAPPA.)";
        put "        MGAMMA[i] ~ dunif(0.05, 2)";
        put "        for (j in 1:57) {"; /*MEAN BACTERICIDAL ACTIVITY*/
        put "            LOGMEAN[i, j] <- MALPHA[i] - MBETA1[i]*(j - 1)/&INTERVAL. - MBETA2[i]*MGAMMA[i]*log((exp(((j - 1)/&INTERVAL. - MKAPPA[i])/MGAMMA[i]) +";
        put "                             exp(-((j - 1)/&INTERVAL. - MKAPPA[i])/MGAMMA[i])))/(exp((MKAPPA[i])/MGAMMA[i]) + exp(-(MKAPPA[i])/MGAMMA[i])))";
        put "            LAMBDA[i, j] <- exp(LOGMEAN[i, j])";
        put "            EXPLAM[i, j] <- (1 - PZERO[i, PROTWIND[j]])*LAMBDA[i, j]";
        put "        }";
        put "        MEBA0056[i] <- -(log(EXPLAM[i, 57]) - log(EXPLAM[i, 1]))/(56*log(10))";
        put "    }";
        put "    ZERO[1] <- 0"; /*DIAGONAL MATRIX*/
        put "    ZERO[2] <- 0";
        put "    ZERO[3] <- 0";
        put "    I[1, 1] <- 0.0001";
        put "    I[1, 2] <- 0";
        put "    I[1, 3] <- 0";
        put "    I[2, 1] <- I[1, 2]";
        put "    I[2, 2] <- 0.0001";
        put "    I[2, 3] <- 0";
        put "    I[3, 1] <- I[1, 3]";
        put "    I[3, 2] <- I[2, 3]";
        put "    I[3, 3] <- 0.0001";
        %do i = 1 %to &NTRT.;
            put "    for (i in 1:&NTRT.) {";
            put "        MEBADTRT&i.0056[i] <- MEBA0056[i] -  MEBA0056[&i.]";
            put "    }";
        %end;
        %do i = 1 %to &NTRT.; /*R MATRIX OF WISHART PRIOR*/
            put     "    R[&i., 1, 1] <- &&WAT&i..";
            put     "    R[&i., 1, 2] <- &&WAB1T&i..";
            put     "    R[&i., 1, 3] <- &&WAB2T&i..";
            put     "    R[&i., 2, 1] <- R[&i., 1, 2]";
            put     "    R[&i., 2, 2] <- &&WB1T&i..";
            put     "    R[&i., 2, 3] <- &&WB1B2T&i..";
            put     "    R[&i., 3, 1] <- R[&i., 1, 3]";
            put     "    R[&i., 3, 2] <- R[&i., 2, 3]";
            put     "    R[&i., 3, 3] <- &&WB2T&i..";
        %end;
        put "    FIT <- sum(SQRES[1:&CE0.])";
        put "    FITNEW <- sum(SQRESNEW[1:&CE0.])";
        put "    FITDIF <- FITNEW - FIT";
        put "    FITIND <- step(FITDIF)"; /*PEARSON GOODNESS OF FIT MEASURE*/
        put "    DEVIANCE <- -2*sum(LOGL[1:&CE0.])";
        put "}";
    run;

%mend;

%OPENBUGS;

%macro SCRIPT (); /*SCRIPT TO CALL OPENBUGS REMOTELY FROM SAS*/

    data _NULL_;
        file "&OPENBUGS./&DATATSHR._RUN.txt";
```

```
put "modelDisplay('log')";
put "modelCheck('&OPENBUGS./&DATATYPE._Biphasic_Model.txt')";
put "modelData('&OPENBUGS./&DATATYPE._DATA.txt')";
put "modelData('&OPENBUGS./&DATATYPE._PATKVT.txt')";
put "modelData('&OPENBUGS./&DATATYPE._PROTWIND.txt')";
put "modelCompile(2)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_MOMGINV_1.txt', 1)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_MOMGINV_2.txt', 2)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_SMU_1.txt', 1)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_SMU_2.txt', 2)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_MMU_1.txt', 1)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_MMU_2.txt', 2)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_MKAPPA_1.txt', 1)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_MKAPPA_2.txt', 2)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_MGAMMA_1.txt', 1)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_MGAMMA_2.txt', 2)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_RHO_1.txt', 1)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_RHO_2.txt', 2)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_PZERO_1.txt', 1)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_PZERO_2.txt', 2)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_U.txt', 1)";
put "modelInits('&OPENBUGS./&DATATYPE._Biphasic_Initial_U.txt', 2)";
put "modelGenInits()";
put "modelUpdate(30000)";
put "samplesSet('MEBA0056')";
%do i = 1 %to &NTRT.;
    put "samplesSet('MEBADTRT&i.0056')";
%end;
put "samplesSet('PPO')";
put "samplesSet('RHO')";
put "samplesSet('PZERO')";
put "samplesSet('SALPHA')";
put "samplesSet('SBETA1')";
put "samplesSet('SBETA2')";
put "samplesSet('MALPHA')";
put "samplesSet('MBETA1')";
put "samplesSet('MBETA2')";
put "samplesSet('MKAPPA')";
put "samplesSet('MGAMMA')";
put "samplesSet('ALPSIGSQ')";
put "samplesSet('BT1SIGSQ')";
put "samplesSet('BT2SIGSQ')";
put "samplesSet('ALPBT1SIGSQ')";
put "samplesSet('ALPBT2SIGSQ')";
put "samplesSet('BT1BT2SIGSQ')";
put "summarySet('LOGCFU')";
put "summarySet('SEBA0056')";
put "summarySet('RHO')";
put "summarySet('PZERO')";
put "summarySet('MALPHA')";
put "summarySet('MBETA1')";
put "summarySet('MBETA2')";
put "summarySet('MKAPPA')";
put "summarySet('MGAMMA')";
put "summarySet('MEBA0056')";
%do i = 1 %to &NTRT.;
    put "summarySet('MEBADTRT&i.0056')";
%end;
put "summarySet('DEVIANCE')";
put "summarySet('FITIND')"
put "modelUpdate(1000, 50)";
put "modelSaveState('&OPENBUGS./&DATATYPE._SaveState')";
put "samplesCoda('*','&ROOT./&DATATYPE._Biphasic_␣')"; /*MCMC SAMPLES FOR POSTPROCESSING*/
```

```
put "samplesHistory('RHO')"; /*TRACEPLOTS*/
put "samplesHistory('PZERO')";
put "samplesHistory('SALPHA')";
put "samplesHistory('SBETA1')";
put "samplesHistory('SBETA2')";
put "samplesHistory('MALPHA')";
put "samplesHistory('MBETA1')";
put "samplesHistory('MBETA2')";
put "samplesHistory('MKAPPA')";
put "samplesHistory('MGAMMA')";
put "samplesHistory('ALPSIGSQ')";
put "samplesHistory('BT1SIGSQ')";
put "samplesHistory('BT2SIGSQ')";
put "samplesHistory('ALPBT1SIGSQ')";
put "samplesHistory('ALPBT2SIGSQ')";
put "samplesHistory('BT1BT2SIGSQ')";
put "samplesAutoC('RHO')"; /*AUTOCORRELATION PLOTS*/
put "samplesAutoC('PZERO')";
put "samplesAutoC('SALPHA')";
put "samplesAutoC('SBETA1')";
put "samplesAutoC('SBETA2')";
put "samplesAutoC('MALPHA')";
put "samplesAutoC('MBETA1')";
put "samplesAutoC('MBETA2')";
put "samplesAutoC('MKAPPA')";
put "samplesAutoC('MGAMMA')";
put "samplesAutoC('ALPSIGSQ')";
put "samplesAutoC('BT1SIGSQ')";
put "samplesAutoC('BT2SIGSQ')";
put "samplesAutoC('ALPBT1SIGSQ')";
put "samplesAutoC('ALPBT2SIGSQ')";
put "samplesAutoC('BT1BT2SIGSQ')";
put "samplesBgr('RHO')"; /*BGR PLOTS*/
put "samplesBgr('PZERO')";
put "samplesBgr('SALPHA')";
put "samplesBgr('SBETA1')";
put "samplesBgr('SBETA2')";
put "samplesBgr('MALPHA')";
put "samplesBgr('MBETA1')";
put "samplesBgr('MBETA2')";
put "samplesBgr('MKAPPA')";
put "samplesBgr('MGAMMA')";
put "samplesBgr('ALPSIGSQ')";
put "samplesBgr('BT1SIGSQ')";
put "samplesBgr('BT2SIGSQ')";
put "samplesBgr('ALPBT1SIGSQ')";
put "samplesBgr('ALPBT2SIGSQ')";
put "samplesBgr('BT1BT2SIGSQ')";
put "summaryStats('LOGCFU')";
put "summaryStats('SEBA0056')";
put "summaryStats('RHO')";
put "summaryStats('PZERO')";
put "summaryStats('MALPHA')";
put "summaryStats('MBETA1')";
put "summaryStats('MBETA2')";
put "summaryStats('MKAPPA')";
put "summaryStats('MGAMMA')";
put "summaryStats('MEBA0056')";
%do i = 1 %to &NTRT.;
    put "summaryStats('MEBADTRT&i.0056')";
%end;
put "summaryStats('DEVIANCE')";
put "summaryStats('FITIND')"
```

```
        put "modelSaveLog('&ROOT./&DATATYPE._Biphasic_LOG.odc')";
        put "modelSaveLog('&ROOT./&DATATYPE._Biphasic_LOG.txt')";
        put "modelQuit(yes)";
    run;

%mend;

%SCRIPT ();

data _NULL_;
    file "&OPENBUGS./&DATATSHR._RUN.bat";
    put "CD &OPENBUGS./";
    put "&OPENSHRT./OpenBUGS.exe /PAR &OPENSHRT./&DATATSHR._RUN.txt";
    put 'EXIT';
run;

data _NULL_; /*BAT FILE TO CALL OPENBUGS*/
    X "&OPENSHRT./&DATATSHR._RUN.bat";
run; quit;

data DATA13 (drop = LOG i); /*DESCRIPTIVE STATISTICS FROM OPENBUGS*/
    infile "&ROOT.\&DATATYPE._Biphasic_LOG.txt" truncover;
    input LOG $80.;
    LOG = strip(translate(LOG, " ", "09"x));
    if prxmatch("m/LOGCFU|SEBA0056|RHO|PZERO|MEBA0056|FITIND/o", LOG) > 0;
    array SUMMARYC $60. PARAMC MEANC SDC Q025C MEDC Q975C SMPC;
    array SUMMARYN PARAM MEAN SD Q025 MED Q975 SMP;
    do i = 1 to dim(SUMMARYC);
        SUMMARYC(i) = substr(LOG, 1, index(LOG, " "));
        if i ne 1 then SUMMARYN(i) = input(SUMMARYC(i), BEST.);
        LOG = substr(LOG, index(LOG, " ") + 1);
    end;
    length VARIABLE TMP1 $25.;
    if index(PARAMC, "[") then VARIABLE = substr(PARAMC, 1, index(PARAMC, "[") - 1); else VARIABLE = PARAMC;
    TMP1 = strip(tranwrd(compress(PARAMC, "[]"), strip(VARIABLE), ""));
    if not index(PARAMC, ",") then INDEX = input(strip(tranwrd(compress(PARAMC, "ABCDEFGHIJKLMNOPQRSTUVWXYZ[]"), "0056", "")), BEST.);
    else do;
        INDEX1 = input(substr(TMP1, 1, index(TMP1, ",") - 1), BEST.);
        INDEX2 = input(substr(TMP1, index(TMP1, ",") + 1), BEST.);
    end;
    if VARIABLE in ("RHO" "PZERO") then do;
        KVTRTN_ = INDEX1;
        PROTTIMC_ = INDEX2;
    end; else
    if VARIABLE in ("LOGCFU") then do;
        NSUBJID = INDEX1;
        TIME = INDEX2 - 1;
    end; else
    if VARIABLE in ("SEBA0056") then do;
        NSUBJID = INDEX;
    end; else KVTRTN_ = INDEX;
proc sort;
    by NSUBJID TIME;
run;

proc sort data = DATA10;
    by NSUBJID TIME;
run;

data DATA14;
    merge DATA13 (in = x) DATA10;
    by NSUBJID TIME;
    if x;
```

```
run;

data DATA15;
    merge DATA14 (drop = KVTRTN SUBJID) DATA11;
    by NSUBJID;
    if KVTRTN_ ne . then KVTRTN = KVTRTN_;
    if PROTTIMC_ ne . then PROTTIMC = PROTTIMC_;
    if CENSORF = . then CENSORF = 0;
    label MEAN = " ";
run;

data INTERIM.&DATATYPE._DESCR;
    set DATA15;
    KVTRTC = put(KVTRTN, KVTRTN.);
    MODEL = "&DATATYPE.";
run;

%macro PANELPLT; /*PRELIMINARY CHECKS: CFU COUNT ON LOG-SCALE*/

    ods graphics / reset noborder width = 6in height = 3.6in imagename = "&DATATYPE._PROFILE";

    ods rtf file = "&ROOT.\&DATATYPE._LOGCFU.rtf" dpi = 300;

    %do i = 1 %to &NTRT.;

        proc template;
            define STATGRAPH PANELGEN;
                begingraph;
                    layout DATAPANEL CLASSVARS = (NSUBJID) /
                           HEADERLABELDISPLAY = VALUE
                           ROWS = 3 COLUMNS = 5
                           CELLHEIGHTMIN = 50 CELLWIDTHMIN = 50
                           SKIPEMPTYCELLS = TRUE
                           COLUMNAXISOPTS = (LABEL = "Day" LINEAROPTS = (VIEWMIN = 0 VIEWMAX = 56
                                             TICKVALUESEQUENCE = (START = 0 INCREMENT = 1 END = 56)))
                           ROWAXISOPTS = (LABEL = "log(CFU) Count" LINEAROPTS = (VIEWMIN = 1 VIEWMAX = 8
                                          TICKVALUESEQUENCE = (START = 1 INCREMENT = 1 END = 8)));
                        layout PROTOTYPE;
                            scatterplot x = TIME y = LOGCFU / PRIMARY = TRUE GROUP = CENSORF NAME = "OBS";
                            seriesplot x = TIME y = MEAN / NAME = "FITOPN" LINEATTRS = (COLOR = RED);
                        endlayout;
                        sidebar / ALIGN = BOTTOM;
                            discretelegend "OBS" / BORDER = FALSE;
                        endsidebar;
                        sidebar / ALIGN = BOTTOM;
                            discretelegend "FITOPN" / BORDER = FALSE;
                        endsidebar;
                    endlayout;
                endgraph;
            end;
        run;

        proc sgrender data = DATA15 (where = (VARIABLE = "LOGCFU" and KVTRTN = &i.)) template = PANELGEN;
        run;

    %end;

    ods rtf close;

%mend;

%PANELPLT;
```

```
%let CODAIND = &DATATYPE._Biphasic CODAINDEX.txt;

data SMPL01;
    length VARNAME $32.;
    infile "&ROOT./&CODAIND." expandtabs;
    input VARNAME $ FIRST LAST;
    if VARNAME ^= '' then do;
        VARNAME = compress(upcase(translate(VARNAME, '___␣', '.[,]')));
    end;
    VARNUM = _N_;
run;

%macro CODA (CHAIN); /*EXPORT MCMC SAMPLES FOR POSTPROCESSING*/

    %let CODAOUT = &DATATYPE._Biphasic CODACHAIN&CHAIN..txt;

    data SMPL02C&CHAIN.;
        infile "&ROOT./&CODAOUT." expandtabs;
        input SAMPNUM VALUE;
    run;

    %global NITER;

    data SMPL03C&CHAIN.;
        set SMPL01;
        call symput("NITER", strip(put(2*(LAST - FIRST + 1), BEST.)));
        do i = FIRST to LAST;
            output;
        end;
        drop i;
    run;

    %put &NITER.;

    data SMPL04C&CHAIN.;
        set SMPL02C&CHAIN.;
        ROW = _N_;
    run;

    data SMPL05C&CHAIN.;
        set SMPL03C&CHAIN.;
        ROW = _N_;
    run;

    data SMPL06C&CHAIN.;
        merge SMPL04C&CHAIN. SMPL05C&CHAIN.;
        by ROW;
    run;

%mend;

%CODA (CHAIN = 1);
%CODA (CHAIN = 2);

data SMPL07;
    set SMPL06C1 (in = x) SMPL06C2 (in = y);
    if x then CHAIN = 1; else
    if y then CHAIN = 2;
    DELIM = "_";
    NUMVAR = countw(VARNAME, DELIM);
    array SPLIT $50. VAR1 - VAR10;
    do i = 1 to NUMVAR;
        SPLIT(i) = scan(VARNAME, i, DELIM);
```

```
    end;
    IVAR = strip(VAR1);
    INDEX1 = input(VAR2, BEST.);
    INDEX2 = input(VAR3, BEST.);
    if substr(VARNAME, 1, 1) = "S" then NSUBJID = INDEX1; else
    if IVAR ne "PPO" then KVTRTN = INDEX1;
    if IVAR in ("RHO" "PZERO") then PROTTIMC = INDEX2;
    if index(VARNAME, "SIGSQ") then COVPARM = 1;
    if IVAR = "PPO" then do;
        LOGLIKE = -2*log(VALUE);
        if 0 <= VALUE <= input("10E-50", BEST8.) then CPO = input("10E-50", BEST8.); else CPO = 1/VALUE;
    end;
proc sort;
    by VARNUM VARNAME IVAR NSUBJID KVTRTN PROTTIMC COVPARM;
run;

proc univariate noprint data = SMPL07 (where = (IVAR ne "PPO")); /*DESCRIPTIVE STATISTICS OF POSTERIOR DISTRIBUTION*/
    by VARNUM VARNAME IVAR NSUBJID KVTRTN PROTTIMC COVPARM;
    var VALUE;
    output out = SMPL08 n = N mean = MEAN var = VAR std = STD min = MIN max = MAX range = RANGE pctlpts = 2.5
    median = MEDIAN pctlpts = 97.5 kurtosis = KURTOSIS skewness = SKEWNESS pctlpre = P;
run;

data INTERIM.&DATATYPE._PARM;
    set SMPL08;
    MODEL = "&DATATYPE.";
run;

proc sort data = SMPL07;
    by CHAIN SAMPNUM;
run;

proc transpose data = SMPL07 (where = (IVAR ne "PPO")) out = SMPL09;
    by CHAIN SAMPNUM;
    var VALUE;
    id VARNAME;
run;

proc sort data = SMPL08 out = MSEL01;
    by KVTRTN PROTTIMC;
run;

proc transpose data = MSEL01 (where = (COVPARM = 1)) out = MSEL02 (drop = _NAME_ _LABEL_);
    by KVTRTN;
    var MEAN;
    id IVAR;
run;

data MSEL03;
    length MOMEGA $32767.;
    set MSEL02;
    if KVTRTN ne . then MOMEGA = strip(put(ALPSIGSQ, BEST.)) || ", " || strip(put(ALPBT1SIGSQ, BEST.)) || ", " || strip(put(ALPBT2SIGSQ, BEST.)) || ", " ||
                                 strip(put(ALPBT1SIGSQ, BEST.)) || ", " || strip(put(BT1SIGSQ, BEST.)) || ", " || strip(put(BT1BT2SIGSQ, BEST.)) || ", " ||
                                 strip(put(ALPBT2SIGSQ, BEST.)) || ", " || strip(put(BT1BT2SIGSQ, BEST.)) || ", " || strip(put(BT2SIGSQ, BEST.));
run;

/********************************/
/**********DIC MEASURE**********/
/********************************/

%macro EXPORTMEAN (SORT, VAR); /*EXPORT MEAN OF POSTERIOR DISTRIBUTION*/

    data &VAR._MEAN01 (rename = (MEAN = D&VAR.));
```

```
        set SMPL08 (where = (IVAR = "&VAR."));
    proc sort;
        by &SORT.;
    run;

    %_LEXPORT (DATA = &VAR._MEAN01, FILE = "&OPENBUGS./&DATATYPE._D&VAR..txt", VAR = D&VAR.);

%mend;

%EXPORTMEAN (SORT = NSUBJID, VAR = SALPHA);
%EXPORTMEAN (SORT = NSUBJID, VAR = SBETA1);
%EXPORTMEAN (SORT = NSUBJID, VAR = SBETA2);
%EXPORTMEAN (SORT = KVTRTN, VAR = MKAPPA);
%EXPORTMEAN (SORT = KVTRTN, VAR = MGAMMA);

proc means data = MSEL01 (where = (PROTTIMC ne .)) noprint;
    var PROTTIMC;
    output out = MAXPROT max = MAX;
run;

data _NULL_;
    set MAXPROT;
    call symput("MAXPROT", strip(put(MAX, BEST.)));
run;

%put &MAXPROT.;

proc datasets;
    delete MAXPROT;
run;

%macro EXPORTMMAT (SORT, VAR);

    proc sort data = SMPL08 (where = (IVAR = "&VAR." and PROTTIMC ne .)) out = &VAR._MEAN01;
        by &SORT.;
    run;

    proc transpose data = &VAR._MEAN01 out = &VAR._MEAN02 (drop = _NAME_ _LABEL_) prefix = D&VAR.;
        by KVTRTN;
        var MEAN;
        id PROTTIMC;
    run;

    %_SEXPORT (DATA = &VAR._MEAN02, VAR = D&VAR.1-D&VAR.&&MAXPROT., FILE = "&OPENBUGS./&DATATYPE._D&VAR..txt");

%mend;

%EXPORTMMAT (SORT = KVTRTN, VAR = RHO);
%EXPORTMMAT (SORT = KVTRTN, VAR = PZERO);

%macro DHAT (FILE); /*EXPORT PARAMETER ESTIMATES VALUES BACK TO OPENBUGS*/

    data _NULL_;
        file "&FILE." linesize = 5000;
        put "model {";
        put "    for (i in &CS0.:&CE0.) {";
        put "        DLINK[i] <- DSALPHA[NSUBJID[i]] - DSBETA1[NSUBJID[i]]*TIME[i]/&INTERVAL. - DSBETA2[NSUBJID[i]]*DMGAMMA[KVTRTN[i]]*";
        put "                    log((exp((TIME[i]/&INTERVAL. - DMKAPPA[KVTRTN[i]])/DMGAMMA[KVTRTN[i]]) + exp(-(TIME[i]/&INTERVAL. - DMKAPPA[KVTRTN[i]])/";
        put "                    DMGAMMA[KVTRTN[i]]))/(exp((DMKAPPA[KVTRTN[i]])/DMGAMMA[KVTRTN[i]]) + exp(-(DMKAPPA[KVTRTN[i]])/DMGAMMA[KVTRTN[i]])))";
        put "        DP[i] <- DRHO[KVTRTN[i], PROTTIMC[i]]/(DRHO[KVTRTN[i], PROTTIMC[i]] + exp(DLINK[i] + OFFSET[i]))";
        put "        NEGBDENS[i] <- max(DUMMY[i], 1)*(loggam(DATA[i] + DRHO[KVTRTN[i], PROTTIMC[i]]) - loggam(DRHO[KVTRTN[i], PROTTIMC[i]]) - loggam(DATA[i] + 1) + DRHO[
      KVTRTN[i], PROTTIMC[i]]*log(DP[i]) + DATA[i]*log(1 - DP[i]))";
        put "        P&MODELVAR.[i] <- max(DUMMY[i], 1)*(DPZERO[KVTRTN[i], PROTTIMC[i]]*equals(DATA[i], 0) + (1 - DPZERO[KVTRTN[i], PROTTIMC[i]])*exp(NEGBDENS[i]))";
```

```
        put "        DUMMY[i] ~ dunif(0.1, 0.2)";
        put "    }";
        put "}";
    run;

%mend;

%DHAT (FILE = &OPENBUGS./&DATATYPE._DHAT.txt);

data _NULL_; /*OPENBUGS SCRIPT*/
    file "&OPENBUGS./&DATATSHR._DR.txt";
    put "modelDisplay('log')";
    put "modelCheck('&OPENBUGS./&DATATYPE._DHAT.txt')";
    put "modelData('&OPENBUGS./&DATATYPE._DATA.txt')";
    put "modelData('&OPENBUGS./&DATATYPE._DSALPHA.txt')";
    put "modelData('&OPENBUGS./&DATATYPE._DSBETA1.txt')";
    put "modelData('&OPENBUGS./&DATATYPE._DSBETA2.txt')";
    put "modelData('&OPENBUGS./&DATATYPE._DMKAPPA.txt')";
    put "modelData('&OPENBUGS./&DATATYPE._DMGAMMA.txt')";
    put "modelData('&OPENBUGS./&DATATYPE._DRHO.txt')";
    put "modelData('&OPENBUGS./&DATATYPE._DPZERO.txt')";
    put "modelCompile(1)";
    put "modelGenInits()";
    put "summarySet('P&MODELVAR.')";
    put "modelUpdate(1)";
    put "samplesCoda('*','&ROOT./&DATATYPE._DHAT_')";
    put "summaryStats('P&MODELVAR.')";
    put "modelSaveLog('&ROOT./&DATATYPE._DHAT_LOG.txt')";
    put "modelQuit(yes)";
run;

data _NULL_;
    file "&OPENSHRT./&DATATSHR._DR.bat";
    put "CD &OPENSHRT.";
    put "&OPENSHRT./OpenBUGS.exe /PAR &DATATSHR._DR.txt";
    put 'EXIT';
run;

data _NULL_;
    X "&OPENSHRT./&DATATSHR._DR.bat";
run; quit;

proc means data = SMPL07 (where = (LOGLIKE ne .)) noprint;
    var LOGLIKE;
    output out = DICB01
    sum = LOGLIKE;
run;

data DICB02;
    set DICB01;
    LOGLIKE = LOGLIKE/&NITER.;
run;

data DICB03 (drop = LOG i MEANC SDC Q025C MEDC Q975C SMPC SD Q025 MED Q975 SMP rename = (MEAN = PROB));
    infile "&ROOT./&DATATYPE._DHAT_LOG.txt" truncover;
    input LOG $80.;
    LOG = strip(translate(LOG, " ", "09"x));
    if substr(LOG, 1, 1) = "P";
    array SUMMARYC $60. PARAMC MEANC SDC Q025C MEDC Q975C SMPC;
    array SUMMARYN PARAM MEAN SD Q025 MED Q975 SMP;
    do i = 1 to dim(SUMMARYC);
        SUMMARYC(i) = substr(LOG, 1, index(LOG, " "));
        if i ne 1 then SUMMARYN(i) = input(SUMMARYC(i), BEST.);
```

```
        LOG = substr(LOG, index(LOG, " ") + 1);
    end;
run;

data DICB04;
    set DICB03;
    INDEX = input(compress(PARAMC, "ABCDEFGHIJKLMNOPQRSTUVWXYZ[]"), BEST.);
    LOGLIKE = -2*log(PROB);
run;

proc means data = DICB04 noprint;
    var LOGLIKE;
    output out = DICB05
    sum = LOGLIKE;
run;

data DICB06; /*DEVIANCE INFORMATION CRITERION STATISTIC*/
    merge DICB02 (drop = _FREQ_ _TYPE_ rename = (LOGLIKE = DBAR)) DICB05 (drop = _FREQ_ _TYPE_ rename = (LOGLIKE = DHAT));
    DIC = 2*DBAR - DHAT;
    PD = DBAR - DHAT;
run;

data INTERIM.&DATATYPE._DIC;
    set DICB06;
    MODEL = "&DATATYPE.";
run;

%include "C:\Users\2004014359\Documents\PhD Thesis\Programs\Formal\Macros\DWISHART & MTCNTOBS.sas";

proc transpose data = MSEL01 (where = (KVTRTN ne .)) out = BFAC01 (drop = _NAME_ _LABEL_);
    by KVTRTN PROTTIMC;
    id IVAR;
    var MEAN;
run;

proc sort data = DATA10;
    by KVTRTN PROTTIMC;
run;

/*********************************************************************/
/**********COMPOUND LAPLACE METROPOLIS MARGINAL LIKELIHOOD**********/
/*********************************************************************/

data BFAC02 (drop = RC RHO PZERO);
    merge DATA10 BFAC01 (rename = (PROTTIMC = RC) where = (RC = .));
    by KVTRTN;
run;

data BFAC03;
    merge BFAC02 BFAC01 (keep = KVTRTN PROTTIMC RHO PZERO where = (PROTTIMC ne .));
    by KVTRTN PROTTIMC;
run;

data BFAC04;
    merge BFAC03 MSEL03 (keep = KVTRTN MOMEGA);
    by KVTRTN;
proc sort;
    by NSUBJID;
run;

data BFAC05; /*LIKELIHOOD FUNCTION PER PATIENT*/
    length LIKER TOTALLIKER $9999. LINK P $200.;
    set BFAC04;
```

```
    by NSUBJID;
    retain TOTALLIKER;
    LINK = "x[1] - x[2]*" || strip(put(TIME/&INTERVAL., BEST.)) || " - x[3]*" || strip(put(MGAMMA*log((exp((TIME/&INTERVAL. - MKAPPA)/MGAMMA) +
           exp(-(TIME/&INTERVAL. - MKAPPA)/MGAMMA))/(exp((MKAPPA)/MGAMMA) + exp(-(MKAPPA)/MGAMMA))), BEST.));
    P = strip(put(RHO, BEST.)) || "/(" || strip(put(RHO, BEST.)) || " + exp(" || strip(LINK) || " + (" || strip(put(OFFSET, BEST.)) || ")))";
    LIKER = "(" || strip(put(PZERO, BEST.)) || "*(" || strip(put(&MODELVAR., BEST.)) || " == 0) + (1 - " || strip(put(PZERO, BEST.)) ||
            ")*dnbinom(x = " || strip(put(&MODELVAR., BEST.)) || ", prob = " || strip(P) || ", size = " || strip(put(RHO, BEST.)) || "))";
    if first.NSUBJID then TOTALLIKER = strip(LIKER); else TOTALLIKER = strip(TOTALLIKER) || "*" || strip(LIKER);
    if last.NSUBJID;
    TOTALLIKER = strip(TOTALLIKER) || "*dmnorm(cbind(x[1], x[2], x[3]), c(" || strip(put(MALPHA, BEST.)) || ", "  || strip(put(MBETA1, BEST.)) ||
                 ", " || strip(put(MBETA2, BEST.)) || "), SIGMA <- matrix(c(" || strip(MOMEGA) || "), 3, 3))";
    call symput("TOTALLIKER" || strip(put(NSUBJID, BEST.)), strip(TOTALLIKER));
run;

%put &TOTALLIKER1.;

data _NULL_;
    call symput("SETWD", "setwd('"␣||␣"&OPENBUGS."␣||␣"')");
run;

%put &SETWD.;

proc sort data = SMPL07 (where = (CPO ne .));
    by INDEX1;
run;

proc univariate noprint data = SMPL07; /*CONDITIONAL POSTERIOR ORDINATES*/
    by INDEX1;
    var CPO;
    output out = CPO01 mean = CPO;
run;

data CPO02;
    set CPO01;
    CPO = 1/CPO;
    ICPO = 1/CPO;
    if ICPO < 40 then ICPO40 = 1; else ICPO40 = 0;
run;

proc univariate noprint data = CPO02;
    var ICPO40;
    output out = CPO03 mean = ICPO40;
run;

data INTERIM.&DATATYPE._ICPO;
    set CPO03;
    MODEL = "&DATATYPE.";
run;

%macro DOINTLIKE; /*R CODE TO INTEGRATE OUT RANDOM EFFECTS OF LIKELIHOOD FUNCTION PER PATIENT*/

    %do i = 1 %to &NSUBJ.;

        data _NULL_;
            set SMPL08 (where = (NSUBJID = &i.)); /*INTEGRATION BOUNDS*/
            if IVAR = "SALPHA" then do;
                call symput("ALPLOW", strip(put(MIN - 1.5*RANGE, BEST.)));
                call symput("ALPHIH", strip(put(MAX + 1.5*RANGE, BEST.)));
            end;
            if IVAR = "SBETA1" then do;
                call symput("BT1LOW", strip(put(MIN - 1.5*RANGE, BEST.)));
                call symput("BT1HIH", strip(put(MAX + 1.5*RANGE, BEST.)));
            end;
```

```
        if IVAR = "SBETA2" then do;
            call symput("BT2LOW", strip(put(MIN - 1.5*RANGE, BEST.)));
            call symput("BT2HIH", strip(put(MAX + 1.5*RANGE, BEST.)));
        end;
    run;

    %put &ALPLOW. &ALPHIH.;
    %put &BT1LOW. &BT1HIH.;
    %put &BT2LOW. &BT2HIH.;

    data _NULL_; /*R CODE TO CALCULATE INTEGRALS*/
        file "&OPENBUGS./&DATATSHR._RP&i..r" linesize = 32767;
        put "&SETWD.";
        put 'library("grDevices")';
        put 'library("R2Cuba")'; /*MULTIDIMENSIONAL INTEGRATION PACKAGE*/
        put 'library("mnormt")';
        put 'library("stats4")';
        put 'library("numDeriv")';
        put 'library("sn")';
        put 'library("truncnorm")';
        put 'integrand␣<-␣function(x)␣{';
        put "    &&TOTALLIKER&i..";
        put '}';
        put "cuhre(ndim = 3, ncomp = 1, integrand, lower = c(&ALPLOW., &BT1LOW., &BT2LOW.), upper = c(&ALPHIH., &BT1HIH., &BT2HIH.), rel.tol = 1e-3, abs.tol = 5e-6,
  flags = list(verbose = 1, final = 1, pseudo.random = 0, mersenne.seed = NULL), min.eval = 1e5, max.eval = 2e5)";
        put "q()";
    run;

    data _NULL_;
        file "&OPENSHRT./&DATATSHR._RU&i..bat";
           put "CD &OPENSHRT.";
           put "&RSHORT./R.exe CMD BATCH --vanilla --quiet &OPENSHRT./&DATATSHR._RP&i..r &OPENSHRT./&DATATSHR._RP&i..txt";
        put "EXIT";
    run;

    data _NULL_;
        X "&OPENSHRT./&DATATSHR._RU&i..bat";
    run; quit;

    data RLOG&i.;
         infile "&OPENBUGS./&DATATSHR._RP&i..txt" length = LEN;
         input VAR1 $VARYING8192. LEN;
    run;

    data RLOG&i. (where = (LOGLIKL ne .)); /*CALL "PER-PATIENT" INTEGRALS*/
        length LOGLIKE $200.;
        set RLOG&i.;
        where index(VAR1, "integral:");
        LOGLIKE = strip(tranwrd(VAR1, "integral: ", ""));
        LOGLIKE = substr(LOGLIKE, 1, index(LOGLIKE, " "));
        LOGLIKL = input(LOGLIKE, BEST32.);
    run;

  %end;

%mend;

%DOINTLIKE;

data BFAC06 (drop = VAR1 LOGLIKE);
    set RLOG1 - RLOG&NSUBJ.;
run;
```

```
%macro DOMATRIX (); /*PRIOR DISTRIBUTIONS*/

    data BFAC07 (drop = PROTTIMC);
        merge BFAC01 (where = (PROTTIMC = .) drop = RHO PZERO) MSEL03 (keep = KVTRTN MOMEGA) RHO_MEAN02 PZERO_MEAN02;
        by KVTRTN;
        %do i = 1 %to &NTRT.; /*R MATRIX (WISHART DISTRIBUTION)*/
            if KVTRTN = &i. then do;
                WA = &&WAT&i..;
                WAB1 = &&WAB1T&i..;
                WAB2 = &&WAB2T&i..;
                WB1 = &&WB1T&i..;
                WB1B2 = &&WB1B2T&i..;
                WB2 = &&WB2T&i..;
            end;
        %end;
        length PMMU PMKAPPA PMGAMMA PMOMGINV $9999.;
        PMMU = "dmnorm(cbind(" || strip(put(MALPHA, BEST.)) || ", " || strip(put(MBETA1, BEST.)) || ", " ||
               strip(put(MBETA2, BEST.)) || "), c(0, 0, 0), SIGMA <- 10^4*matrix(c(1, 0, 0, 0, 1, 0, 0, 0, 1), 3, 3))";
        PMKAPPA = "dunif(" || strip(put(MKAPPA, BEST.)) || ", min = &LKAPPA., max = &UKAPPA.)";
        PMGAMMA = "dunif(" || strip(put(MGAMMA, BEST.)) || ", min = 0.05, max = 2)";
        PMOMGINV = "dWishart(solve(matrix(c(" || strip(MOMEGA) || "), 3, 3)), &WDF., S <- solve(matrix(c(" || strip(put(WA, BEST.)) ||
                   "," || strip(put(WAB1, BEST.)) || ", " || strip(put(WAB2, BEST.)) || ", " || strip(put(WAB1, BEST.)) || ", " ||
                   strip(put(WB1, BEST.)) || ", " || strip(put(WB1B2, BEST.)) || "," || strip(put(WAB2, BEST.)) || ", " ||
                   strip(put(WB1B2, BEST.)) || ", " || strip(put(WB2, BEST.)) || "), 3, 3)))";
        %do i = 1 %to &MAXPROT.;
            length PRHO&i. PPZERO&i. $9999.;
            PRHO&i. = "dgamma(" || trim(left(put(DRHO&i., BEST.))) || ", 0.1, 0.1)";
            PPZERO&i. = "dbeta(" || strip(put(DPZERO&i., BEST.)) || ", 0.1, 0.1)";
        %end;
    run;

%mend;

%DOMATRIX;

%macro PRIORLIKE (PARAMETER);

    %do i = 1 %to &NTRT.;

        data _NULL_;
            set BFAC07;
            if KVTRTN = &i.;
            call symput("&PARAMETER." || strip(put(KVTRTN, BEST.)), strip(&PARAMETER.));
        run;

        %put &&&PARAMETER&i..;

        data _NULL_;
            file "&OPENBUGS./&DATATSHR._PT&i..r" linesize = 32767;
            put "&SETWD.";
            put 'library("grDevices")';
            put 'library("mnormt")';
            %if "&PARAMETER." = "PMOMGINV" %then %do;
                %DWISHART;
            %end;
            put "&&&PARAMETER&i..";
            put "q()";
        run;

        data _NULL_;
            file "&OPENSHRT./&DATATSHR._RU&i..bat";
            put "CD &OPENSHRT.";
            put "&RSHORT./R.exe CMD BATCH --vanilla --quiet &OPENSHRT./&DATATSHR._PT&i..r &OPENSHRT./&DATATSHR._PT&i..txt";
```

```
            put "EXIT";
        run;

        data _NULL_;
            X "&OPENSHRT./&DATATSHR._RU&i..bat";
        run; quit;

        data &PARAMETER.&i.;
             infile "&OPENBUGS./&DATATSHR._PT&i..txt" length = LEN;
             input VAR1 $VARYING8192. LEN;
        run;

        data &PARAMETER.&i. (drop = VAR1);
            set &PARAMETER.&i.;
            KVTRTN = &i.;
            if index(VAR1, "[1]");
            &PARAMETER. = input(substr(VAR1, 4), BEST32.);
        run;

    %end;

    data &PARAMETER.;
        set &PARAMETER.:;
    run;

%mend;

%PRIORLIKE (PARAMETER = PMMU);
%PRIORLIKE (PARAMETER = PMKAPPA);
%PRIORLIKE (PARAMETER = PMGAMMA);
%PRIORLIKE (PARAMETER = PMOMGINV);

%macro LOOPPROT;

    %do j = 1 %to &MAXPROT.;

        %PRIORLIKE (PARAMETER = PRHO&j.);
        %PRIORLIKE (PARAMETER = PPZERO&j.);

    %end;

%mend;

%LOOPPROT;

%macro NUMPOST; /*NUMBER OF PARAMETERS*/

    %global BFPRMSN;

    data _NULL_;
        PMMU = 3;
        PMKAPPA = 1;
        PMGAMMA = 1;
        PMOMGINV = 6;
        %do i = 1 %to &MAXPROT.;
            PRHO&i. = 1;
            PPZERO&i. = 1;
        %end;
        call symput("BFPRMSN", strip(put(&NTRT.*sum(of _ALL_), BEST.)));
    run;

%mend;
```

```
%NUMPOST;

%put &BFPRMSN.;

%macro SETPOSDTR;

    data BFAC08;
        merge PMMU PMKAPPA PMGAMMA PMOMGINV %do i = 1 %to &MAXPROT.; PRHO&i. PPZERO&i. %end;;
        by KVTRTN;
    run;

%mend;

%SETPOSDTR;

proc transpose data = BFAC08 out = BFAC09 (where = (_NAME_ ne 'KVTRTN'));
    by KVTRTN;
    var _ALL_;
run;

data BFAC10;
    set BFAC06 (in = x) BFAC09 (in = y rename = (COL1 = LOGLIKL));
    if x then PTYPE = "DATA"; else
    if y then PTYPE = "PARM";
    LPROB = log(LOGLIKL);
run;

proc means data = BFAC10 noprint;
    var LPROB;
    output out = BFAC11
    sum = LSPRB;
run;

proc corr data = SMPL09 (keep = MALPHA: MBETA1: MBETA2: MKAPPA: MGAMMA: RHO: PZERO:) out = BFAC12 (rename = (_NAME_ = VARNAME)); /*CORRELATION MATRIX OF MCMCM SAMPLES*/
    var _ALL_;
proc sort;
    by _TYPE_;
run;

data BFAC13;
    set BFAC12 (where = (_TYPE_ = "CORR"));
run;

data BFAC14;
    set BFAC13 (drop = _TYPE_ VARNAME);
    array MTRX _ALL_;
    call symput("NPRMSFIX", strip(put(dim(MTRX), BEST.)));
    length DMATRIX $32767.;
    do i = 1 to dim(MTRX);
        if i = 1 then DMATRIX = strip(put(MTRX(i), BEST.)); else DMATRIX = strip(DMATRIX) || " " || strip(put(MTRX(i), BEST.));
    end;
    call symput("MTRX" || strip(put(_N_, BEST.)), strip(DMATRIX));
run;

%put &MTRX1.;
%put &NPRMSFIX.;

ods listing;

%macro DETR ();

    proc iml; /*CALCULATE DETERMINANT OF CORRELATION MATRIX*/
        M = {%do i = 1 %to %eval(&NPRMSFIX. - 1); &&MTRX&i.. , %end; &&MTRX&NPRMSFIX.};
```

```
        DETM = det(M);
        print DETM;
        create BFAC15 var {DETM};
        append;
        close BFAC15;
    quit;

%mend;

%DETR ();

data BFAC16;
    set BFAC12 (where = (_TYPE_ = "STD"));
run;

data BFAC17;
    set BFAC16 (drop = _TYPE_ VARNAME);
    array LSTD _ALL_;
    do i = 1 to dim(LSTD);
        LSTD(i) = log(LSTD(i));
    end;
    LSTDP = sum(of _ALL_);
run;

data BFAC18;
    merge BFAC11 (drop = _FREQ_ _TYPE_) BFAC15 BFAC17 (keep = LSTDP);
    MARGINAL = 0.5*&BFPRMSN.*log(2*3.141592654) + 0.5*log(DETM) + LSTDP + LSPRB; /*CLMML*/
run;

data INTERIM.&DATATYPE._BF;
    set BFAC18;
    MODEL = "&DATATYPE.";
run;
```