This Appendix provides the programming code used to estimate the distribution of salpingitis and number of subsequent PID episodes in the population. The appendix is set out in three sections. (A) provides the WBDev functions written in Component Pascal which are called by the WinBUGS code shown in sections B and C. (B) is the WinBUGS code that calculates the numbers of PIDs, CT-related and non-CT-related. (C) is the WinBUGS code that develops the comparisons with the Lund data.

(A). WBDEV code

Note that in appendix A the, philap (the proportion of PID cases that would be diagnosed as salpingitis on laparoscopy) parameter can be removed to generate estimates of the distribution of clinical PID and diagnosed clinical PID. To generate estimates for the distribution of salpingitis it can be included as a multiplier in the exponents for all of the transition probabilities.

```
MODULE WBDevCumulativePIDlap;

 IMPORT
   WBDevVector,
   Math;

  TYPE
   Function = POINTER TO RECORD (WBDevVector.Node) END;
   Factory = POINTER TO RECORD (WBDevVector.Factory) END;

  VAR
   fact-: WBDevVector.Factory;

  PROCEDURE (func: Function) DeclareArgTypes (OUT args: ARRAY OF CHAR);
  BEGIN
   args := "vv";
  END DeclareArgTypes;


 PROCEDURE readindata1 (func: Function; OUT lambda_PID: ARRAY OF REAL);
  VAR
   index,a:                          INTEGER;

  BEGIN
   index := 0;
   FOR a := 1 TO 4 DO
    lambda_PID[a] := func.arguments[0][index].Value();
    INC(index);
   END;
  END readindata1;


PROCEDURE readindata2 (func: Function; OUT psi: REAL);
```

```
 BEGIN
  psi := func.arguments[0][4].Value();
 END readindata2;


PROCEDURE readindata3 (func: Function; OUT eta1: REAL);
 BEGIN
  eta1 := func.arguments[0][5].Value();
 END readindata3;


PROCEDURE readindata4 (func: Function; OUT EF: ARRAY OF REAL);
 VAR
  index,a:                          INTEGER;

 BEGIN
  index := 6;
  FOR a := 1 TO 4 DO
   EF[a] := func.arguments[0][index].Value();
   INC(index);
  END;
 END readindata4;

PROCEDURE readindata5 (func: Function; OUT philap: REAL);
 BEGIN
  philap := func.arguments[0][10].Value();
 END readindata5;

PROCEDURE readindata6 (func: Function; OUT N: ARRAY OF REAL);
 VAR
  index,a:                          INTEGER;

 BEGIN
  index := 0;
  FOR a := 16 TO 44 DO
   N[a] := func.arguments[1][index].Value();
   INC(index);
  END;
 END readindata6;


PROCEDURE tranratesall (lambda_PID: ARRAY OF REAL; eta1: REAL;
                        OUT lambda_PID2: ARRAY OF ARRAY OF REAL);
 VAR
  a       :                         INTEGER;

 BEGIN
  FOR a := 16 TO 19 DO
   lambda_PID2[1,a] := lambda_PID[1] * 0.85;
   lambda_PID2[2,a] := lambda_PID2[1,a] * eta1;
  END;

  FOR a := 20 TO 24 DO
   lambda_PID2[1,a] := lambda_PID[2] * 0.85;
   lambda_PID2[2,a] := lambda_PID2[1,a] * eta1;
  END;

  FOR a := 25 TO 34 DO
   lambda_PID2[1,a] := lambda_PID[3] * 0.85;
   lambda_PID2[2,a] := lambda_PID2[1,a] * eta1;
```

```
 END;

 FOR a := 35 TO 44 DO
  lambda_PID2[1,a] := lambda_PID[4] * 0.85;
  lambda_PID2[2,a] := lambda_PID2[1,a] * eta1;
 END;
END tranratesall;

PROCEDURE tranratesnct (lambda_PID: ARRAY OF REAL; eta1: REAL; EF: ARRAY OF
REAL;
                        OUT lambda_PID2nct: ARRAY OF ARRAY OF REAL);
 VAR
 a       :                          INTEGER;

 BEGIN
  FOR a := 16 TO 19 DO
   lambda_PID2nct[1,a] := (lambda_PID[1]  - lambda_PID[1] * EF[1]) * 0.85;
   lambda_PID2nct[2,a] := lambda_PID2nct[1,a] * eta1;
  END;

  FOR a := 20 TO 24 DO
   lambda_PID2nct[1,a] := (lambda_PID[2]  - lambda_PID[2] * EF[2]) * 0.85;
   lambda_PID2nct[2,a] := lambda_PID2nct[1,a] * eta1;
  END;

  FOR a := 25 TO 34 DO
   lambda_PID2nct[1,a] := (lambda_PID[3]  - lambda_PID[3] * EF[3]) * 0.85;
   lambda_PID2nct[2,a] := lambda_PID2nct[1,a] * eta1;
  END;

  FOR a := 35 TO 44 DO
   lambda_PID2nct[1,a] := (lambda_PID[4]  - lambda_PID[4] * EF[4]) * 0.85;
   lambda_PID2nct[2,a] := lambda_PID2nct[1,a] * eta1;
  END;
 END tranratesnct;


 PROCEDURE tranproball (lambda_PID2: ARRAY OF ARRAY OF REAL; philap: REAL;
                        OUT p: ARRAY OF ARRAY OF ARRAY OF REAL);
 VAR
 a:                          INTEGER;

 BEGIN
  FOR a := 14 TO 15 DO
   p[1,2,a] := 0;
   p[1,1,a] := 0;

   p[2,2,a] := 0;
   p[2,5,a] := 0;
   p[2,3,a] := 0;

   p[3,3,a] := 0;
   p[3,5,a] := 0;
   p[3,4,a] := 0;

   p[4,5,a] := 0;
   p[4,4,a] := 0;

   p[5,5,a] := 0;
   p[5,8,a] := 0;
```

```
  p[5,6,a] := 0;

  p[6,6,a] := 0;
  p[6,8,a] := 0;
  p[6,7,a] := 0;

  p[7,8,a] := 0;
  p[7,7,a] := 0;

  p[8,8,a] := 0;
 END;

 FOR a := 16 TO 44 DO
  p[1,2,a] := 1 - Math.Exp(-lambda_PID2[1,a] * philap);
  p[1,1,a] := 1 - p[1,2,a];

  p[2,2,a] := 0;
  p[2,5,a] := 1 - Math.Exp(-lambda_PID2[2,a]);
  p[2,3,a] := 1 - p[2,5,a];

  p[3,3,a] := 0;
  p[3,5,a] := 1 - Math.Exp(-lambda_PID2[2,a]);
  p[3,4,a] := 1 - p[3,5,a];

  p[4,5,a] := 1 - Math.Exp(-lambda_PID2[1,a]);
  p[4,4,a] := 1 - p[4,5,a];

  p[5,5,a] := 0;
  p[5,8,a] := 1 - Math.Exp(-lambda_PID2[2,a]);
  p[5,6,a] := 1 - p[5,8,a];

  p[6,6,a] := 0;
  p[6,8,a] := 1 - Math.Exp(-lambda_PID2[2,a]);
  p[6,7,a] := 1 - p[6,8,a];

  p[7,8,a] := 1 - Math.Exp(-lambda_PID2[1,a]);
  p[7,7,a] := 1 - p[7,8,a];

  p[8,8,a] := 1;
 END;
END tranproball;


PROCEDURE tranprobnct (lambda_PID2nct: ARRAY OF ARRAY OF REAL; philap: REAL;
                       OUT pnct: ARRAY OF ARRAY OF ARRAY OF REAL);
VAR
a       :                       INTEGER;

BEGIN
 FOR a := 16 TO 44 DO
  pnct[1,2,a] := 1 - Math.Exp(-lambda_PID2nct[1,a] * philap);
  pnct[1,1,a] := 1 - pnct[1,2,a];

  pnct[2,2,a] := 0;
  pnct[2,5,a] :=  1 - Math.Exp(-lambda_PID2nct[2,a]);
  pnct[2,3,a] :=  1 - pnct[2,5,a];

  pnct[3,3,a] := 0;
  pnct[3,5,a] := 1 - Math.Exp(-lambda_PID2nct[2,a]);
  pnct[3,4,a] := 1 - pnct[3,5,a];
```

```
  pnct[4,5,a] := 1 - Math.Exp(-lambda_PID2nct[1,a]);
  pnct[4,4,a] := 1 - pnct[4,5,a];

  pnct[5,5,a] := 0;
  pnct[5,8,a] := 1 - Math.Exp(-lambda_PID2nct[2,a]);
  pnct[5,6,a] := 1 - pnct[5,8,a];

  pnct[6,6,a] := 0;
  pnct[6,8,a] := 1 - Math.Exp(-lambda_PID2nct[2,a]);
  pnct[6,7,a] := 1 - pnct[6,8,a];

  pnct[7,8,a] := 1 - Math.Exp(-lambda_PID2nct[1,a]);
  pnct[7,7,a] := 1 - pnct[7,8,a];

  pnct[8,8,a] :=1;
 END;
END tranprobnct;


PROCEDURE stateoccpropall (p: ARRAY OF ARRAY OF ARRAY OF REAL;
                           OUT pi: ARRAY OF ARRAY OF REAL);
VAR
a,s:                    INTEGER;

BEGIN
 FOR a := 13 TO 15 DO
  pi[1,a] := 1;
  FOR s := 2 TO 8 DO
   pi[s,a] := 0;
  END;
 END;

 FOR a := 16 TO 44 DO
  pi[1,a] := pi[1,a-1] * p[1,1,a];
  pi[2,a] := pi[1,a-1] * p[1,2,a];
  pi[3,a] := pi[2,a-1] * p[2,3,a];
  pi[4,a] := pi[3,a-1] * p[3,4,a] + pi[4,a-1] * p[4,4,a];
  pi[5,a] := pi[2,a-1] * p[2,5,a] + pi[3,a-1] * p[3,5,a] + pi[4,a-1] * p[4,5,a] + pi[5,a-1] * p[5,5,a];
  pi[6,a] := pi[5,a-1] * p[5,6,a];
  pi[7,a] := pi[6,a-1] * p[6,7,a] + pi[7,a-1] * p[7,7,a];
  pi[8,a] := pi[5,a-1] * p[5,8,a] + pi[6,a-1] * p[6,8,a] + pi[7,a-1] * p[7,8,a] + pi[8,a-1];
 END;

END stateoccpropall;

PROCEDURE stateoccpropnct (pnct: ARRAY OF ARRAY OF ARRAY OF REAL;
                           OUT pinct: ARRAY OF ARRAY OF REAL);
VAR
a,s:                    INTEGER;

BEGIN
 FOR a := 13 TO 15 DO
  pinct[1,a] := 1;
  FOR s := 2 TO 8 DO
   pinct[s,a] := 0;
  END;
 END;

 FOR a := 16 TO 44 DO
```

```
   pinct[1,a] := pinct[1,a-1] * pnct[1,1,a];
   pinct[2,a] := pinct[1,a-1] * pnct[1,2,a];
   pinct[3,a] := pinct[2,a-1] * pnct[2,3,a];
   pinct[4,a] := pinct[3,a-1] * pnct[3,4,a] + pinct[4,a-1] * pnct[4,4,a];
   pinct[5,a] := pinct[2,a-1] * pnct[2,5,a] + pinct[3,a-1] * pnct[3,5,a] + pinct[4,a-1] * pnct[4,5,a] +
                 pinct[5,a-1] * pnct[5,5,a];
   pinct[6,a] := pinct[5,a-1] * pnct[5,6,a];
   pinct[7,a] := pinct[6,a-1] * pnct[6,7,a] + pinct[7,a-1] * pnct[7,7,a];
   pinct[8,a] := pinct[5,a-1] * pnct[5,8,a] + pinct[6,a-1] * pnct[6,8,a] + pinct[7,a-1] * pnct[7,8,a] +
                 pinct[8,a-1];
  END;
 END stateoccpropnct;


 PROCEDURE numberPIDsageall (pi: ARRAY OF ARRAY OF REAL; N: ARRAY OF REAL;
                             OUT PIDnum: ARRAY OF ARRAY OF REAL);
 VAR
 a:              INTEGER;

 BEGIN
  FOR a := 16 TO 44 DO
   PIDnum[1,a] := (pi[2,a] + pi[3,a] + pi[4,a]);
   PIDnum[2,a] := (pi[5,a] + pi[6,a] + pi[7,a]);
   PIDnum[3,a] := pi[8,a];
  END;
 END numberPIDsageall;


PROCEDURE numberPIDsagegpall (PIDnum: ARRAY OF ARRAY OF REAL; N: ARRAY OF
REAL;
                              OUT numPIDs: ARRAY OF ARRAY OF REAL);
 VAR
 n:              INTEGER;

 BEGIN
  FOR n := 1 TO 3 DO
   numPIDs[n,1] := (PIDnum[n,16] * N[16] + PIDnum[n,17] * N[17] + PIDnum[n,18] * N[18] +
                    PIDnum[n,19] * N[19]) /
                    (N[16] + N[17] + N[18] + N[19]);
   numPIDs[n,2] := (PIDnum[n,20] * N[20] + PIDnum[n,21] * N[21]  + PIDnum[n,22] * N[22]  +
                    PIDnum[n,23] * N[23] + PIDnum[n,24] * N[24]) /
                    (N[20] + N[21] + N[22] + N[23] + N[24]);
   numPIDs[n,3] := (PIDnum[n,25] * N[25] + PIDnum[n,26] * N[26] + PIDnum[n,27] * N[27]  +
                    PIDnum[n,28] * N[28] + PIDnum[n,29] * N[29] + PIDnum[n,30] * N[30] +
                    PIDnum[n,31] * N[31] + PIDnum[n,32] * N[32] + PIDnum[n,33] * N[33]+
                    PIDnum[n,34] * N[34]) /
                  (N[25] + N[26] + N[27] + N[28] + N[29] + N[30] + N[31] + N[32] + N[33] +
N[34]);
numPIDs[n,4] := (PIDnum[n,35] * N[35] + PIDnum[n,36] * N[36] + PIDnum[n,37] * N[37] +
                 PIDnum[n,38] * N[38] + PIDnum[n,39] * N[39] + PIDnum[n,40] * N[40] +
                 PIDnum[n,41] * N[41] + PIDnum[n,42] * N[42] + PIDnum[n,43] * N[43] +
                 PIDnum[n,44] * N[44]) /
                (N[35] + N[36] + N[37] + N[38] + N[39] + N[40] + N[41] + N[42] + N[43] +
N[44]);
  END;
 END numberPIDsagegpall;


 PROCEDURE numberPIDsobsageall (PIDnum: ARRAY OF ARRAY OF REAL; psi: REAL;
                                OUT PIDnumobs: ARRAY OF ARRAY OF REAL);
```

```
 VAR
 a:                INTEGER;

 BEGIN
  FOR a := 16 TO 44 DO
   PIDnumobs[1,a] :=  PIDnum[1,a] * psi + 2 * PIDnum[2,a] * psi * (1 - psi) +
                      3 * PIDnum[3,a] * psi * (1 - psi) * (1 - psi);
   PIDnumobs[2,a] :=  PIDnum[2,a] * psi * psi + 3 * PIDnum[3,a] * psi * psi * (1 - psi);
   PIDnumobs[3,a] :=  PIDnum[3,a] * psi * psi * psi
  END;
 END numberPIDsobsageall;


PROCEDURE numberPIDsagenct (pinct: ARRAY OF ARRAY OF REAL;  N: ARRAY OF
REAL;
                            OUT PIDnumnct: ARRAY OF ARRAY OF REAL);
 VAR
 a:                INTEGER;

 BEGIN
  FOR a := 16 TO 44 DO
   PIDnumnct[1,a] := (pinct[2,a] + pinct[3,a] + pinct[4,a]);
   PIDnumnct[2,a] := (pinct[5,a] + pinct[6,a] + pinct[7,a]);
   PIDnumnct[3,a] := pinct[8,a];
  END;
 END numberPIDsagenct;


  PROCEDURE numberPIDsagegpnct (PIDnumnct: ARRAY OF ARRAY OF REAL; N:
ARRAY OF
                                REAL;
                              OUT numPIDsnct: ARRAY OF ARRAY OF REAL);

 VAR
 n:                INTEGER;

 BEGIN
  FOR n := 1 TO 3 DO
   numPIDsnct[n,1] := (PIDnumnct[n,16] * N[16] + PIDnumnct[n,17] * N[17] +
                       PIDnumnct[n,18] * N[18] + PIDnumnct[n,19] * N[19]) /
                      (N[16] + N[17] + N[18] + N[19]);
   numPIDsnct[n,2] := (PIDnumnct[n,20] * N[20] + PIDnumnct[n,21] * N[21] +
                       PIDnumnct[n,22] * N[22] + PIDnumnct[n,23] * N[23] +
                       PIDnumnct[n,24] * N[24]) /
                      (N[20] + N[21] + N[22] + N[23] + N[24]);
   numPIDsnct[n,3] := (PIDnumnct[n,25] * N[25] + PIDnumnct[n,26] * N[26] +
                       PIDnumnct[n,27] * N[27] + PIDnumnct[n,28] * N[28] +
                       PIDnumnct[n,29] * N[29] + PIDnumnct[n,30] * N[30] +
                       PIDnumnct[n,31] * N[31] + PIDnumnct[n,32] * N[32] +
                       PIDnumnct[n,33] * N[33] + PIDnumnct[n,34] * N[34]) /
                     (N[25] + N[26] + N[27] + N[28] + N[29] + N[30] + N[31] + N[32] + N[33] +
N[34]);
   numPIDsnct[n,4] := (PIDnumnct[n,35] * N[35] + PIDnumnct[n,36] * N[36] +
                       PIDnumnct[n,37] * N[37] + PIDnumnct[n,38] * N[38] +
                       PIDnumnct[n,39] * N[39] + PIDnumnct[n,40] * N[40] +
                       PIDnumnct[n,41] * N[41] + PIDnumnct[n,42] * N[32] +
                       PIDnumnct[n,43] * N[43] + PIDnumnct[n,44] * N[44]) /
                     (N[35] + N[36] + N[37] + N[38] + N[39] + N[40] + N[41] + N[42] + N[43] +
N[44]);
  END;
```

```
 END numberPIDsagegpnct;


 PROCEDURE numberPIDsobsageallnct (PIDnumnct: ARRAY OF ARRAY OF REAL; psi:
REAL;
                                   OUT PIDnumobsnct: ARRAY OF ARRAY OF REAL);
 VAR
 a:              INTEGER;

 BEGIN
  FOR a := 16 TO 44 DO
   PIDnumobsnct[1,a] :=  PIDnumnct[1,a] * psi + 2 * PIDnumnct[2,a] * psi * (1 - psi) +
                         3 * PIDnumnct[3,a] * psi * (1 - psi) * (1 - psi);
   PIDnumobsnct[2,a] :=  PIDnumnct[2,a] * psi * psi + 3 * PIDnumnct[3,a] * psi * psi * (1 - psi);
   PIDnumobsnct[3,a] :=  PIDnumnct[3,a] * psi * psi * psi
  END;
 END numberPIDsobsageallnct;


PROCEDURE eventsin8 (pi: ARRAY OF ARRAY OF REAL; p: ARRAY OF ARRAY OF
ARRAY OF
                        REAL; lambda_PID2: ARRAY OF ARRAY OF REAL;  N: ARRAY
OF REAL;
                   OUT PIDsin8: ARRAY OF REAL);
 VAR
 a:              INTEGER;
 last2yrs:       ARRAY 45 OF REAL;

 BEGIN
  PIDsin8[14] := 0;
  PIDsin8[15] := 0;
  FOR a := 16 TO 44 DO
   last2yrs[a] := pi[5,a-3] * p[5,8,a-2] + pi[5,a-2] * p[5,8,a-1] +
                  pi[6,a-3] * p[6,8,a-2] + pi[6,a-2] * p[6,8,a-1] +
                  pi[7,a-3] * p[7,8,a-2] + pi[7,a-2] * p[7,8,a-1];

   PIDsin8[a] := last2yrs[a] * ( 1 - Math.Exp(-lambda_PID2[2,a])) +
                 (pi[8,a] - last2yrs[a]) * (1 - Math.Exp(-lambda_PID2[1,a]))
  END;
 END eventsin8;

PROCEDURE eventsin8nct (pinct: ARRAY OF ARRAY OF REAL; pnct: ARRAY OF ARRAY
OF
                           ARRAY OF REAL; lambda_PID2nct: ARRAY OF ARRAY OF
REAL;  N:
                           ARRAY OF REAL;
                      OUT PIDsin8nct: ARRAY OF REAL);
 VAR
 a:              INTEGER;
 last2yrs:       ARRAY 45 OF REAL;

 BEGIN
  PIDsin8nct[14] := 0;
  PIDsin8nct[15] := 0;
  FOR a := 16 TO 44 DO
   last2yrs[a] := pinct[5,a-3] * pnct[5,8,a-2] + pinct[5,a-2] * pnct[5,8,a-1] +
                  pinct[6,a-3] * pnct[6,8,a-2] + pinct[6,a-2] * pnct[6,8,a-1] +
                  pinct[7,a-3] * pnct[7,8,a-2] + pinct[7,a-2] * pnct[7,8,a-1];

   PIDsin8nct[a] := last2yrs[a] * ( 1 - Math.Exp(-lambda_PID2nct[2,a])) +
```

```
                    (pinct[8,a] - last2yrs[a]) * (1 - Math.Exp(-lambda_PID2nct[1,a]));
  END;
 END eventsin8nct;


PROCEDURE predincforpop (pi: ARRAY OF ARRAY OF REAL; p: ARRAY OF ARRAY OF
ARRAY
                                OF REAL; PIDsin8: ARRAY OF REAL;
                            OUT PIDinc: ARRAY OF ARRAY OF REAL);
 VAR
  a:              INTEGER;

 BEGIN
  FOR a := 16 TO 44 DO
   PIDinc[1,a] := pi[1,a-1] * p[1,2,a];
   PIDinc[2,a] := pi[2,a-1] * p[2,5,a] + pi[3,a-1] * p[3,5,a] + pi[4,a-1] * p[4,5,a];
   PIDinc[3,a] := pi[5,a-1] * p[5,8,a] + pi[6,a-1] * p[6,8,a] + pi[7,a-1] * p[7,8,a];
   PIDinc[4,a] := PIDsin8[a];
  END;
 END predincforpop;

PROCEDURE predincforpopnct (pinct: ARRAY OF ARRAY OF REAL; pnct: ARRAY OF
ARRAY OF
                                    ARRAY OF REAL; PIDsin8nct: ARRAY OF REAL;
                                OUT PIDincnct: ARRAY OF ARRAY OF REAL);
 VAR
 a:              INTEGER;

 BEGIN
  FOR a := 16 TO 44 DO
   PIDincnct[1,a] := pinct[1,a-1] * pnct[1,2,a];
   PIDincnct[2,a] := pinct[2,a-1] * pnct[2,5,a] + pinct[3,a-1] * pnct[3,5,a] +
                     pinct[4,a-1] * pnct[4,5,a];
   PIDincnct[3,a] := pinct[5,a-1] * pnct[5,8,a] + pinct[6,a-1] * pnct[6,8,a] +
                     pinct[7,a-1] * pnct[7,8,a];
   PIDincnct[4,a] := PIDsin8nct[a];
   END;
 END predincforpopnct;


PROCEDURE predincgpforpop (PIDinc: ARRAY OF ARRAY OF REAL; N: ARRAY OF
REAL;
                                PIDsin8: ARRAY OF REAL;
                            OUT pred_prop_ag: ARRAY OF REAL);
 VAR
 a:              INTEGER;
 pred_pop:       ARRAY 45 OF REAL;
 sumN:           REAL;

BEGIN
  FOR a := 16 TO 44 DO
   pred_pop[a] := (PIDinc[1,a] + PIDinc[2,a] + PIDinc[3,a] + PIDinc[4,a]) *  N[a];
  END;
  pred_prop_ag[1] := (pred_pop[16] + pred_pop[17] + pred_pop[18] + pred_pop[19]) /
                     (N[16] + N[17] + N[18] + N[19]);
  pred_prop_ag[2] := (pred_pop[20] + pred_pop[21] + pred_pop[22] + pred_pop[23] +
pred_pop[24]) /
                     (N[20] + N[21] + N[22] + N[23] + N[24]);
  pred_prop_ag[3] := (pred_pop[25] + pred_pop[26] + pred_pop[27] + pred_pop[28] +
pred_pop[29] +
```

```
                        pred_pop[30] + pred_pop[31] + pred_pop[32] + pred_pop[33] +
pred_pop[34]) /
                        (N[25] + N[26] + N[27] + N[28] + N[29] + N[30] + N[31] + N[32] + N[33]
+ N[34]);
  pred_prop_ag[4] := (pred_pop[35] + pred_pop[36] + pred_pop[37] + pred_pop[38] +
pred_pop[39] +
                        pred_pop[40] + pred_pop[41] + pred_pop[42] + pred_pop[43] +
pred_pop[44]) /
                       (N[35] + N[36] + N[37] + N[38] + N[39] + N[40] + N[41] + N[42] + N[43] +
N[44]);

  pred_prop_ag[0] := 0;
  sumN := 0;
  FOR a := 16 TO 44 DO
   pred_prop_ag[0] := pred_prop_ag[0] + pred_pop[a];
   sumN := sumN + N[a];
  END;
  pred_prop_ag[0] := pred_prop_ag[0] / sumN;
 END predincgpforpop;

PROCEDURE predincgpforpopnct (PIDincnct: ARRAY OF ARRAY OF REAL; N: ARRAY OF
REAL;
                                    PIDsin8nct: ARRAY OF REAL;
                                OUT pred_prop_agnct: ARRAY OF REAL);
 VAR
 a:              INTEGER;
 pred_pop:       ARRAY 45 OF REAL;
 sumN:           REAL;

 BEGIN

  FOR a := 16 TO 44 DO
   pred_pop[a] := (PIDincnct[1,a] + PIDincnct[2,a] + PIDincnct[3,a] + PIDincnct[4,a]) *  N[a];
  END;
  pred_prop_agnct[1] := (pred_pop[16] + pred_pop[17] + pred_pop[18] + pred_pop[19]) /
                        (N[16] + N[17] + N[18] + N[19]);
  pred_prop_agnct[2] := (pred_pop[20] + pred_pop[21] + pred_pop[22] + pred_pop[23] +
                        pred_pop[24])  / (N[20] + N[21] + N[22] + N[23] + N[24]);
  pred_prop_agnct[3] := (pred_pop[25] + pred_pop[26] + pred_pop[27] + pred_pop[28] +
                        pred_pop[29] + pred_pop[30] + pred_pop[31] + pred_pop[32] +
                        pred_pop[33] + pred_pop[34]) /
                       (N[25] + N[26] + N[27] + N[28] + N[29] + N[30] + N[31] + N[32] + N[33]
+ N[34]);
  pred_prop_agnct[4] := (pred_pop[35] + pred_pop[36] + pred_pop[37] + pred_pop[38] +
                        pred_pop[39] + pred_pop[40] + pred_pop[41] + pred_pop[42] +
                        pred_pop[43] + pred_pop[44])   /
                       (N[35] + N[36] + N[37] + N[38] + N[39] + N[40] + N[41] + N[42] + N[43]
+ N[44]);
  pred_prop_agnct[0] := 0;
  sumN := 0;
  FOR a := 16 TO 44 DO
   pred_prop_agnct[0] := pred_prop_agnct[0] + pred_pop[a];
   sumN := sumN + N[a];
  END;
  pred_prop_agnct[0] := pred_prop_agnct[0] / sumN;
 END predincgpforpopnct;


 PROCEDURE (func: Function) Evaluate (OUT values: ARRAY OF REAL);
 VAR
```

```
 a, n, i:                                                    INTEGER;
 eta1, psi, philap:                                          REAL;
 lambda_PID, EF, pred_prop_ag, pred_prop_agnct:              ARRAY 5 OF REAL;
 PIDsin8,PIDsin8nct,N:                                       ARRAY 45 OF REAL;
 lambda_PID2, lambda_PID2nct:                                ARRAY 3,45 OF REAL;
 p, pnct:                                                    ARRAY 9,9,45 OF REAL;
 pi, pinct:                                                  ARRAY 9,45 OF REAL;
 numPIDs, numPIDsnct:                                        ARRAY 4,5 OF REAL;
 PIDinc, PIDincnct:                                          ARRAY 5,45 OF REAL;
 PIDnum, PIDnumnct, PIDnumobs,PIDnumobsnct:                  ARRAY 4,45 OF REAL;

BEGIN
 readindata1(func, lambda_PID);
 readindata2(func, psi);
 readindata3(func, eta1);
 readindata4(func, EF);
 readindata5(func, philap);
 readindata6(func, N);

 tranratesall(lambda_PID, eta1,   lambda_PID2);
 tranratesnct(lambda_PID, eta1, EF,   lambda_PID2nct);

 tranproball(lambda_PID2, philap,   p);
 tranprobnct(lambda_PID2nct, philap,   pnct);

 stateoccpropall(p,  pi );
 stateoccpropnct(pnct,  pinct );

 numberPIDsageall(pi, N,  PIDnum);
 numberPIDsagenct(pinct, N,   PIDnumnct);

 numberPIDsagegpall(PIDnum, N,  numPIDs);
 numberPIDsagegpnct(PIDnumnct, N,   numPIDsnct);

 numberPIDsobsageall(PIDnum, psi,  PIDnumobs);
 numberPIDsobsageallnct(PIDnumnct, psi,  PIDnumobsnct);

 eventsin8(pi, p, lambda_PID2, N,   PIDsin8);
 predincforpop(pi, p, PIDsin8,  PIDinc);
 predincgpforpop(PIDinc, N, PIDsin8,     pred_prop_ag);

 eventsin8nct(pinct, pnct, lambda_PID2nct, N,   PIDsin8nct);
 predincforpopnct(pinct, pnct, PIDsin8nct,     PIDincnct);
 predincgpforpopnct(PIDincnct, N, PIDsin8nct,       pred_prop_agnct);

 i := 0;
 FOR n := 1TO 3 DO
  FOR a := 1 TO 4 DO
   values[i] := numPIDs[n,a];
   values[i+12] := numPIDsnct[n,a];
   INC(i);
  END;
 END;
 FOR i := 24 TO 28 DO
  values[i] := pred_prop_ag[i-24];
 END;

 i := 29;
 FOR n := 1 TO 3 DO
  FOR a := 16 TO 44 DO
```

```
    values[i] := PIDnum[n,a];
    values[i+87] := PIDnumnct[n,a];
    INC(i);
   END;
  END;
  i := 203;
  FOR n := 1 TO 4 DO
   FOR a := 16 TO 44 DO
    values[i] := PIDinc[n,a];
    values[i+116] := PIDincnct[n,a];
    INC(i);
   END;
  END;
  i := 435;
  FOR n := 1 TO 3 DO
   FOR a := 16 TO 44 DO
    values[i] := PIDnumobs[n,a];
    values[i+87] := PIDnumobsnct[n,a];
    INC(i);
   END;
  END;


 END Evaluate;

 PROCEDURE (f: Factory) New (option: INTEGER): Function;
 VAR
  func: Function;
 BEGIN
  NEW(func); func.Initialize; RETURN func;
 END New;

 PROCEDURE Install*;
 BEGIN
  WBDevVector.Install(fact);
 END Install;

 PROCEDURE Init;
 VAR
  f: Factory;
 BEGIN
  NEW(f); fact := f;
 END Init;

BEGIN
 Init;
END WBDevCumulativePIDlap.
```

(B). WinBUGS code to calculate numbers of PIDs and non-CT related PIDs

```
model {
# PID incidence and ct GP re-infection rate ratio informative priors
Y[1:10] ~ dmnorm(mu[], Omega[ , ])
for (ag in 1:4) {
 log(lambda.PID[ag]) <- Y[ag]
 EF[ag]  <- Y[ag+5] * Y[ag+5]
 }
logit(psi) <- Y[5]
```

```
log(eta1) <- Y[10]

# WBDEV call
for (ag in 1:4) {
 input1[ag] <- lambda.PID[ag]
 input1[ag+6] <- EF[ag]
 }
input1[5] <- psi
input1[6] <- eta1

for (i in 1:29) {
 input2[i] <- N[i+15]
 }

philap ~ dbeta(12,16)
input1[11] <- philap

solution[1:609] <- cumulativePIDlap(input1[1:11],input2[1:29])

for (ag in 1:4) {
 Expect.prop[ag] <- 1 - exp(-lambda.PID[ag])
 }
total.expect.prop <- (Expect.prop[1] * sum(N[16:19]) +
                      Expect.prop[2] * sum(N[20:24]) +
                      Expect.prop[3] * sum(N[25:34]) +
                      Expect.prop[4] * sum(N[35:44])) /
                      sum(N[16:44])
}


# Data
list(
# PID incidence, r-infection raqte, Etological fractions and re-
# infection rate
mu = c(-3.865, -3.595, -3.964, -4.402, -0.5856, 0.7104, 0.4815, 0.3117, 0.3277,
1.919),
Omega = structure(.Data =c(
573.855, -49.264, -10.817, -3.992, 323.371, 97.687, -112.351, -30.198, -5.401, -2.713,
-49.264, 301.419, -14.194, -3.299, 147.591, -78.069, 176.480, -72.284, -11.551, 2.012,
-10.817, -14.194, 105.908, -20.063, 37.684, -13.275, -46.636, 236.305, -118.245,
0.515,
-3.992, -3.299, -20.063, 57.687, 19.172, -2.664, -8.418, -125.677, 138.728, 0.088,
323.371, 147.591, 37.684, 19.172, 351.620, 1.363, -1.443, -1.164, 1.537, 0.030,
97.687, -78.069, -13.275, -2.664, 1.363, 285.706, -324.389, -85.846, -16.864, -7.957,
-112.351, 176.480, -46.636, -8.418, -1.443, -324.389, 734.182, -295.254, -50.046,
8.090,
-30.198, -72.284, 236.305, -125.677, -1.164, -85.846, -295.254, 1512.713, -758.618,
3.088,
-5.401, -11.551, -118.245, 138.728, 1.537, -16.864, -50.046, -758.618, 839.634, 0.705,
-2.713, 2.012, 0.515, 0.088, 0.030, -7.957, 8.090, 3.088, 0.705, 13.711
),
.Dim = c(10,10)),

# Population sizes from census, age =1...44 - 2002
N=c(NA,NA,NA,NA,NA, NA,NA,NA,NA,NA, NA,NA,NA,NA,NA,
305500,306300,296400,291400,294800,
310100,313900,305600,294700,295000,
304100,317000,329600,349600,370300,
380900,376900,387800,390900,399400,
401200,402600,398700,391900,381900, 370900,356200,349000,343800)
)

# Initial values 1
list(
```

```
Y = c(-5,-5,-5,-5,-1,-5,-5,-5,-5,-1)
)

# Initial values 2
list(
Y = c(-1,-1,-1,-1,3,-1,-1,-1,-1,3)
)
```

(C). WinBUGS code comparisons to Lund data

Westrom proportions

```
model {
r1[1:3] ~ dmulti(p1[1:3],N1)
r2[1:3] ~ dmulti(p2[1:3],N2)

p1[1:3] ~ ddirch(a1[1:3])
p2[1:3] ~ ddirch(a2[1:3])
}

# Data
list(r1 = c(771,158,61),r2 = c(220,27,4), N1 = 990, N2 = 251,
     a1 =c(1,1,1), a2 = c(1,1,1))

# Initial Values
list(p1 = c(0.7,0.2,0.1),p2 = c(0.7,0.2,0.1))

# twelve - comparison of cumulative PID to Westrom
model {
 Y[1:10] ~ dmnorm(mu[], Omega[ , ])
 for (ag in 1:4) {
  log(lambda.PID[ag]) <- Y[ag]
  }
 logit(psi) <- Y[5]
 log(eta1) <- Y[10]


# transition rates
 for (a in 16:19) {
  lambda.PID2[1,a] <- lambda.PID[1] * 0.85
  lambda.PID2[2,a] <- lambda.PID2[1,a] * eta1
  }
 for (a in 20:24) {
  lambda.PID2[1,a] <- lambda.PID[2] * 0.85
  lambda.PID2[2,a] <- lambda.PID2[1,a] * eta1
  }
 for (a in 25:34) {
  lambda.PID2[1,a] <- lambda.PID[3] * 0.85
  lambda.PID2[2,a] <- lambda.PID2[1,a] * eta1
 }
 for (a in 35:44) {
  lambda.PID2[1,a] <- lambda.PID[4] * 0.85
  lambda.PID2[2,a] <- lambda.PID2[1,a] * eta1
  }

 # transition probabilities
 # state 1 = 1 PID <1 year
 # state 2 = 1 PID 1-2 years
 # state 3 = 1 PID 2+ years
 # state 4 = 2 PIDs <1 year
```

```
# state 5 = 2 PIDs 1-2 years
# state 6 = 2 PIDs 2+ years
# state 7 = 3 PIDs < 1 year
# state 8 = 3 PIDs 1-2 years
# state 9 = 3 PIDs 2+ years
# state 10 = 4 PIDs < 1 year
# state 11 = 4 PIDs < 1-2 years
# state 12 = 4 PIDs < 2+ years
# state 13 = 5 PIDs assume no more than 5 PIDs

for (a in 16:44) {
 p[1,1,a] <- 0
 p[1,2,a] <- 1 - p[1,4,a]
 p[1,4,a] <- 1 - exp(-lambda.PID2[2,a])

 p[2,2,a] <- 0
 p[2,3,a] <- 1 - p[2,4,a]
 p[2,4,a] <- 1 - exp(-lambda.PID2[2,a])

 p[3,3,a] <- 1 - p[3,4,a]
 p[3,4,a] <- 1 - exp(-lambda.PID2[1,a])

 p[4,4,a] <- 0
 p[4,5,a] <- 1 - p[4,7,a]
 p[4,7,a] <- 1 - exp(-lambda.PID2[2,a])

 p[5,5,a] <- 0
 p[5,6,a] <- 1 - p[5,7,a]
 p[5,7,a] <- 1 - exp(-lambda.PID2[2,a])

 p[6,6,a] <- 1 - p[6,7,a]
 p[6,7,a] <- 1 - exp(-lambda.PID2[1,a])

 p[7,7,a] <- 0
 p[7,8,a] <- 1 - p[7,10,a]
 p[7,10,a] <- 1 - exp(-lambda.PID2[2,a])

 p[8,8,a] <- 0
 p[8,9,a] <- 1 - p[8,10,a]
 p[8,10,a] <- 1 - exp(-lambda.PID2[2,a])

 p[9,9,a] <- 1 - p[9,10,a]
 p[9,10,a] <- 1 - exp(-lambda.PID2[1,a])

 p[10,10,a] <- 0
 p[10,11,a] <- 1 - p[10,13,a]
 p[10,13,a] <- 1 - exp(-lambda.PID2[2,a])

 p[11,11,a] <- 0
 p[11,12,a] <- 1 - p[11,13,a]
 p[11,13,a] <- 1 - exp(-lambda.PID2[2,a])

 p[12,12,a] <- 1 - p[9,10,a]
 p[12,13,a] <- 1 - exp(-lambda.PID2[1,a])

 p[13,13,a] <- 1
}

# State occupancy proportions
for (i in 16:37) {
 pi[i,1,i-1] <- 1
```

```
  pi[i,2,i-1] <- 0
  pi[i,3,i-1] <- 0
  pi[i,4,i-1] <- 0
  pi[i,5,i-1] <- 0
  pi[i,6,i-1] <- 0
  pi[i,7,i-1] <- 0
  pi[i,8,i-1] <- 0
  pi[i,9,i-1] <- 0
  pi[i,10,i-1] <- 0
  pi[i,11,i-1] <- 0
  pi[i,12,i-1] <- 0
  pi[i,13,i-1] <- 0
 }

 for (i in 16:37) {
  for (a in i:i+7) {
   pi[i,1,a] <- pi[i,1,a-1] * p[1,1,a]
   pi[i,2,a] <- pi[i,1,a-1] * p[1,2,a] + pi[i,2,a-1] * p[2,2,a]
   pi[i,3,a] <- pi[i,2,a-1] * p[2,3,a] + pi[i,3,a-1] * p[3,3,a]
   pi[i,4,a] <- pi[i,1,a-1] * p[1,4,a] + pi[i,2,a-1] * p[2,4,a] +
                pi[i,3,a-1] * p[3,4,a] + pi[i,4,a-1] * p[4,4,a]
   pi[i,5,a] <- pi[i,4,a-1] * p[4,5,a] + pi[i,5,a-1] * p[5,5,a]
   pi[i,6,a] <- pi[i,5,a-1] * p[5,6,a] + pi[i,6,a-1] * p[6,6,a]
   pi[i,7,a] <- pi[i,4,a-1] * p[4,7,a] + pi[i,5,a-1] * p[5,7,a] +
                pi[i,6,a-1] * p[6,7,a] + pi[i,7,a-1] * p[7,7,a]

   pi[i,8,a] <- pi[i,7,a-1] * p[7,8,a] + pi[i,8,a-1] * p[8,8,a]
   pi[i,9,a] <- pi[i,8,a-1] * p[8,9,a] + pi[i,9,a-1] * p[9,9,a]
   pi[i,10,a] <- pi[i,7,a-1] * p[7,10,a] + pi[i,8,a-1] * p[8,10,a] +
                 pi[i,9,a-1] * p[9,10,a] + pi[i,10,a-1] * p[10,10,a]
   pi[i,11,a] <- pi[i,10,a-1] * p[10,11,a] + pi[i,11,a-1] *
p[11,11,a]
   pi[i,12,a] <- pi[i,11,a-1] * p[11,12,a] + pi[i,12,a-1] *
p[12,12,a]
   pi[i,13,a] <- pi[i,10,a-1] * p[10,13,a] + pi[i,11,a-1] *
p[11,13,a] +
                 pi[i,12,a-1] * p[12,13,a] + pi[i,13,a-1] *
p[13,13,a]
   }
  }

 # Proportion in each state by age
 for (i in 16:37) {
  prop[i,1] <- sum(pi[i,1:3,i+7])
  prop[i,2] <- sum(pi[i,4:6,i+7])
  prop[i,3] <- sum(pi[i,7:9,i+7])
  prop[i,4] <- sum(pi[i,10:12,i+7])
  prop[i,5] <- pi[i,13,i+7]
  propcomp[i,1] <- prop[i,1]
  propcomp[i,2] <- prop[i,2]
  propcomp[i,3] <- sum(prop[i,3:5])
      }
 propunder25[1] <- sum(propcomp[16:24,1]) / 9
 propunder25[2] <- sum(propcomp[16:24,2]) / 9
 propunder25[3] <- sum(propcomp[16:24,3]) / 9
 propover25[1] <- sum(propcomp[25:37,1]) / 13
 propover25[2] <- sum(propcomp[25:37,2]) / 13
 propover25[3] <- sum(propcomp[25:37,3]) / 13

 # Proportion expected to be observed in state by age
 for (i in 16:37) {
```

```
  prop.obs[i,1] <- prop[i,1] +
                   prop[i,2] * (1 - psi) +
                   prop[i,3] * (1 - psi) * (1 - psi) +
                   prop[i,4] * (1 - psi) * (1 - psi) * (1 - psi) +
                  prop[i,5] * (1 - psi) * (1 - psi) * (1 - psi) * (1
- psi)

  prop.obs[i,2] <- prop[i,2] * psi +
                   2 * prop[i,3] * psi * (1 - psi) +
                   3 * prop[i,4] * psi * (1 - psi) * (1 - psi) +
                   4 * prop[i,5] * psi * (1 - psi) * (1 - psi) * (1 -
psi)

  prop.obs[i,3] <- prop[i,3] * psi * psi +
                   3 * prop[i,4] * psi * psi * (1 - psi) +
                   6 * prop[i,5] * psi * psi * (1 - psi) * (1 - psi)

  prop.obs[i,4] <- prop[i,4] * psi * psi * psi  +
                   4 * prop[i,5] * psi * psi * psi * (1 - psi)
  prop.obs[i,5] <- prop[i,5] * psi * psi * psi * psi

  prop.obscomp[i,1] <- prop.obs[i,1]
  prop.obscomp[i,2] <- prop.obs[i,2]
  prop.obscomp[i,3] <- sum(prop.obs[i,3:5])
  }
 prop.obsunder25[1] <- sum(prop.obscomp[16:24,1]) / 9
 prop.obsunder25[2] <- sum(prop.obscomp[16:24,2]) / 9
 prop.obsunder25[3] <- sum(prop.obscomp[16:24,3]) / 9
 prop.obsover25[1] <- sum(prop.obscomp[25:37,1]) / 13
 prop.obsover25[2] <- sum(prop.obscomp[25:37,2]) / 13
 prop.obsover25[3] <- sum(prop.obscomp[25:37,3]) / 13
}


# Data
list(
# PID incidence, r-infection raqte, Etological fractions and re-
# infection rate
mu = c(-3.865, -3.595, -3.964, -4.402, -0.5856, 0.7104, 0.4815, 0.3117, 0.3277,
1.919),
Omega = structure(.Data =c(
573.855, -49.264, -10.817, -3.992, 323.371, 97.687, -112.351, -30.198, -5.401, -2.713,
-49.264, 301.419, -14.194, -3.299, 147.591, -78.069, 176.480, -72.284, -11.551, 2.012,
-10.817, -14.194, 105.908, -20.063, 37.684, -13.275, -46.636, 236.305, -118.245,
0.515,
-3.992, -3.299, -20.063, 57.687, 19.172, -2.664, -8.418, -125.677, 138.728, 0.088,
323.371, 147.591, 37.684, 19.172, 351.620, 1.363, -1.443, -1.164, 1.537, 0.030,
97.687, -78.069, -13.275, -2.664, 1.363, 285.706, -324.389, -85.846, -16.864, -7.957,
-112.351, 176.480, -46.636, -8.418, -1.443, -324.389, 734.182, -295.254, -50.046,
8.090,
-30.198, -72.284, 236.305, -125.677, -1.164, -85.846, -295.254, 1512.713, -758.618,
3.088,
-5.401, -11.551, -118.245, 138.728, 1.537, -16.864, -50.046, -758.618, 839.634, 0.705,
-2.713, 2.012, 0.515, 0.088, 0.030, -7.957, 8.090, 3.088, 0.705, 13.711
),
.Dim = c(10,10))
)


# Initial values 1
list(
Y = c(-5,-5,-5,-5,-1,-5,-5,-5,-5,-1)
)
```

```
# Initial values 2
list(
Y = c(-1,-1,-1,-1,3,-1,-1,-1,-1,3)
)
```