REFERENCES

Aaltola, J. (2002) Simultaneous synthesis of flexible heat exchanger network. Applied Thermal Engineering 22(8), 907-918.

Biegler, L.T., Grossmann, I.E., and Westerberg, A.W. (1997) Systematic Methods of Chemical Process Design. Upper Saddle River (US): Pearson Education.

Björk, K.M. and Westerlund, T. (2002) Global optimization of heat exchanger network synthesis problems with and without the isothermal mixing assumption. Computers & Chemical Engineering 26(11), 1581-1593.

Briones, V. and Kokossis, A.C. (1999) Hypertargets: a Conceptual Programming approach for the optimisation of industrial heat exchanger networks — II. Retrofit design. Chemical Engineering Science 54(4), 541-561.

Chen, C.L. and Hung, P.S. (2004) Simultaneous synthesis of flexible heat-exchange networks with uncertain source-stream temperatures and flow rates. Industrial & Engineering Chemistry Research 43(18), 5916-5928.

Chen, J.J.J. (1987) Comments on improvements on a replacement for the logarithmic mean. Chemical Engineering Science 42(10), 2488-2489.

Ciric, A.R. and Floudas, C.A. (1991) Heat exchanger network synthesis without decomposition. Computers & Chemical Engineering 15(6), 385-396.

Floudas, C.A. (1995) Nonlinear and Mixed-Integer Optimization. New York: Oxford University Press.

Floudas, C.A. and Ciric, A.R. (1989) Strategies for overcoming uncertainties in heat exchanger network synthesis. Computers & Chemical Engineering 13(10), 1133-1152.

Floudas, C.A., Ciric, A.R., and Grossmann, I.E. (1986) Automatic synthesis of optimum heat exchanger network configurations. AIChE Journal 32(2), 276-290.

Floudas, C.A. and Grossmann, I.E. (1986) Synthesis of flexible heat exchanger networks for multiperiod operation. Computers & Chemical Engineering 10(2), 153-168.

Floudas, C.A. and Grossmann, I.E. (1987) Automatic generation of multiperiod heat exchanger network configurations. Computers & Chemical Engineering 11(2), 123-142.

Hohmann, E.C. (1971) Optimum networks for heat exchange. Ph.D. Dissertation. University of Southern California, Los Angeles, California, USA.

Iyer, R. and Grossmann, I. (1996) Global Optimization of Heat Exchanger Networks with Fixed Configuration for Multiperiod Design. In I.E. Grossmann, Global Optimization in Engineering Design (pp. 289-308). Boston: Kluwer Academic.

Linnhoff, B. and Flower, J.R. (1978) Synthesis of heat exchanger networks: I. Systematic generation of energy optimal networks. AIChE Journal 24(4), 633-642.

Linnhoff, B. and Flower, J.R. (1978) Synthesis of heat exchanger networks: II. Evolutionary generation of networks with various criteria of optimality. AIChE Journal 24(4), 642-654.

Ma, X., Yao, P., Luo, X., and Roetzel, W. (2008) Synthesis of multi-stream heat exchanger network for multi-period operation with genetic/simulated annealing algorithms. Applied Thermal Engineering 28(8–9), 809-823.

Papalexandri, K.P. and Pistikopoulos, E.N. (1994) Synthesis and retrofit design of operable heat exchanger networks. 1. Flexibility and structural controllability aspects. Industrial & Engineering Chemistry Research 33(7), 1718-1737.

Papalexandri, K.P. and Pistikopoulos, E.N. (1994) Synthesis and retrofit design of operable heat exchanger networks. 2. Dynamics and control structure considerations. Industrial & Engineering Chemistry Research 33(7), 1738-1755.

Papoulias, S.A. and Grossmann, I.E. (1983) A structural optimization approach in process synthesis—II: Heat recovery networks. Computers & Chemical Engineering 7(6), 707-721.

Pejpichestakul, W. (2013) Retrofit of distillation column in petrochemical processes. M.S. Thesis. The Petroleum and Petrochemical College, Chulalongkorn University, Bangkok, Thailand.

Paterson, W.R. (1984) A replacement for the logarithmic mean. Chemical Engineering Science 39(11), 1635-1636.

Dawe, R.A. (2000) Modern Petroleum Technology. New York: Wiley.

Quesada, I. and Grossmann, I.E. (1993) Global optimization algorithm for heat exchanger networks. Industrial & Engineering Chemistry Research 32(3), 487-499.

Saboo, A.K., Morari, M., and Woodcock, D.C. (1985) Design of resilient processing plants—VIII. A resilience index for heat exchanger networks. Chemical Engineering Science 40(8), 1553-1565.

Shenoy, U.V. (1995) Heat Exchanger Network Synthesis: Optimization Process by Energy and Resource Analysis. Texas: Gulf Publishing Company.

Siemanond, K. and Kosol, S. (2012) Heat exchanger network retrofit by pinch design method using stage-model mathematical programming. Chemical Engineering Transaction 29, 367-372.

Sieniutycz, S. and Jezowski, J. (2009) Energy Optimization in Process Systems: Elsevier Science.

Smith, R. (2005). Chemical Process: Design and Integration. West Sussex: Wiley.

Swaney, R.E. and Grossmann, I.E. (1985) An index for operational flexibility in chemical process design. Part I: Formulation and theory. AIChE Journal 31(4), 621-630.

Tantimuratha, L., Asteris, G., Antonopoulos, D.K., and Kokossis, A.C. (2001) A conceptual programming approach for the design of flexible HENs. Computers & Chemical Engineering 25(4–6), 887-892.

Verheyen, W. and Zhang, N. (2006) Design of flexible heat exchanger network for multi-period operation. Chemical Engineering Science 61(23), 7730-7753.

Yee, T.F. and Grossmann, I.E. (1990) Simultaneous optimization models for heat integration—II. Heat exchanger network synthesis. Computers & Chemical Engineering 14(10), 1165-1184.

Yee, T.F., Grossmann, I.E., and Kravanja, Z. (1990) Simultaneous optimization models for heat integration—I. Area and energy targeting and modeling of multi-stream exchangers. Computers & Chemical Engineering 14(10), 1151-1164.

Yee, T.F., Grossmann, I.E., and Kravanja, Z. (1990) Simultaneous optimization models for heat integration—III. Process and heat exchanger network optimization. Computers & Chemical Engineering 14(11), 1185-1200.

Zhu, X.X. (1995) Automated synthesis of HENs using block decomposition and heuristic rules. Computers & Chemical Engineering 19, Supplement 1(0), 155-160.

Zhu, X.X. (1997) Automated design method for heat exchanger network using block decomposition and heuristic rules. Computers & Chemical Engineering 21(10), 1095-1104.

Zhu, X.X., O'Neill, B.K., Roach, J.R., and Wood, R.M. (1995) Area-targeting methods for the direct synthesis of heat exchanger networks with unequal film coefficients. Computers & Chemical Engineering 19(2), 223-239.

Zhu, X.X., O'Neill, B.K., Roach, J.R., and Wood, R.M. (1995) A new method for heat exchanger network synthesis using area targeting procedures. Computers & Chemical Engineering 19(2), 197-222.

APPENDICES

Appendix A Source Code of MINLP Single Period Model

The following text is source code which is entered in GAMS for synthesizing HEN for single period.

```
SETS
       I  hot streams                    /I1*I3/
       J  cold streams                   /J1*J4/
       K  index of stage or location  /K1*K4,KL/    //last number is no. of stages

//dynamic sets//
       stage(k)    all stages
       ;
       stage(k) = yes;     stage('KL') = no;

SCALARS

//Cost//
       CA     per unit cost of heat exchanger area($ per unit)     /641.7/
       B      exponent for area cost(dimensionless)                 /1/
       CF     fixed cost of heat exchanger($ per unit)              /8333.3/

       CHU    per unit cost of hot utility($ per kW)                /115.2/
       CCU    fixed cost of heat exchanger($ per kW)                /1.3/

       AF     annualisation factor                                  /0.2/

//other//
       EMAT   exchanger minimum approach temperature(C)         /5/

       HHU    heat transfer coefficient of hot utility(kW|(m2.C))   /2/
       HCU    heat transfer coefficient of cold utility(kW|(m2.C))  /1/
       ;

PARAMETERS
//Hot Streams//
       THIN(i)  inlet temperature of hot streams(C)
       /      I1    406
              I2    160
              I3    362
       /
       THOUT(i)  outlet temperature of hot streams(C)
       /      I1    60
```

```
            I2    40
            I3    60
      /
//Cold Streams//
      TCIN(j)   inlet temperature of cold streams(C)
      /     J1    72
            J2    62
            J3    220
            J4    250
      /
      TCOUT(j)  outlet temperature of cold streams(C)
      /     J1    365
            J2    210
            J3    370
            J4    290
      /
//Heat Capacity Flowrates//
      FCPH(i)   heat capaity flowrates fo hot streams(C)
      /     I1    205.0
            I2    198.8
            I3    136.4
      /
      FCPC(j)   heat capaity flowrates fo cold streams(C)
      /     J1    210.3
            J2    141.0
            J3    175.4
            J4    318.7
      /
//Utility//
      THUIN(j)  inlet temperature of hot utility(C)
      /     J1    400
            J2    400
            J3    400
            J4    400
      /
      THUOUT(j) outlet temperature of hot utility(C)
      /     J1    399
            J2    399
            J3    399
            J4    399
      /
      TCUIN(i)  inlet temperature of cold utility(C)
      /     I1    15
            I2    15
            I3    15
      /
      TCUOUT(i) outlet temperature of cold utility(C)
```

```
/       I1   20
        I2   20
        I3   20
/
//Heat transfer coefficient//
    HH(i)  heat capaity flowrates of hot streams(kW|(m2.C))
/       I1   2
        I2   2
        I3   2
/
    HC(j)  heat capaity flowrates of cold streams(kW|(m2.C))
/       J1   1.5
        J2   1.5
        J3   2
        J4   2
/
//overall heat transfer coefficient//
    U(i,j)          Overall heat transfer coefficient for match(i.j)
    UHU(j)          Overall heat transfer coefficient for cold stream(j) and hot utility
    UCU(i)          Overall heat transfer coefficient for hot stream(i) and cold utility

//upper bound for logical constraints//
    GAMMA_HX(i,j,k)   upper bound heat load for match (i.j)
    GAMMA_HU(j)       upper bound heat load for hot utility
    GAMMA_CU(i)       upper bound heat load for cold utility

    TAU_HX(i,j,k)     upper bound of temperature difference for heat exchanger
    TAU_HUL(j)        upper bound of temperature difference for left side of hot
                      utility exchanger
    TAU_HUR(j)        upper bound of temperature difference for right side of hot
                      utility exchanger
    TAU_CUL(i)        upper bound of temperature difference for left side pf cold
                      utility exchanger
    TAU_CUR(i)        upper bound of temperature difference for right side of cold
                      utility exchanger

//parameter for answer verification//
    z1(i,j,k)       correct existence of match(i.j) at stage k
    zhu1(j)         correct existence of hot utility for cold stream j
    zcu1(i)         correct existence of cold utility for hot stream i

    dt1(i,j,k)      correct temperature difference(C) for match(i.j) at location k
    dthul1(j)       correct temperature difference(C) for left side of hot utility ex
                    changer
    dthur1(j)       correct temperature difference(C) for right side of hot utility ex
                    changer
```

dtcul1(i) correct temperature difference(C) for left side of cold utility exchanger
dtcur1(i) correct temperature difference(C) for right side of cold utility exchanger

LMTD1(i,j,k) correct log mean temperature difference of match(i.j) at stage k
LMTDHU1(j) correct log mean temperature difference of hot utility exchanger for cold stream j
LMTDCU1(i) correct log mean temperature difference of cold utility exchanger for cold stream i

AREAHX1(i,j,k) correct area of heat exchanger of match(i.j) at stage k
AREAHU1(j) correct area of hot utility exchanger of cold stream j
AREACU1(i) correct area of cold utility exchanger of hot stream i

Cost_fix1 correct fixed cost
Cost_area1 correct area cost
Cost_utility1 correct utility cost

TAC1 correct total annualized cost

TOT_AREA total area of heat exchangers including utility exchangers
TOT_HU total hot utility
TOT_CU total cold utility
;

//Overall heat transfer coefficient calculations//
U(i,j) = 1/(1/HH(i)+1/HC(j));
UHU(j) = 1/(1/HC(j)+1/HHU);
UCU(i) = 1/(1/HH(i)+1/HCU);

VARIABLES
th(i,k) temperature(C) of hot streams i at location k
tc(j,k) temperature(C) of cold streams j at location k

q(i,j,k) heat load(kW) of match(i.j) at stage k
qhu(j) heat load(kW) of hot utility for cold stream j
qcu(i) heat load(kW) of cold utility for hot stream i

z(i,j,k) existence of match(i.j) at stage k
zhu(j) existence of hot utility for cold stream j
zcu(i) existence of cold utility for hot stream i

dt(i,j,k) temperature difference(C) for match(i.j) at location k
dthul(j) temperature difference(C) for left side of hot utility exchanger
dthur(j) temperature difference(C) for right side of hot utility exchanger
dtcul(i) temperature difference(C) for left side of cold utility exchanger

```
        dtcur(i)        temperature difference(C) for right side of cold utility exchanger

        LMTD(i,j,k)      log mean temperature difference of match(i.j) at stage k
        LMTDHU(j)       log mean temperature difference of hot utility exchanger for
                        cold stream j
        LMTDCU(i)       log mean temperature difference of cold utility exchanger for
                        cold stream i

        AREA(i,j,k)      area of heat exchanger of match(i.j) at stage k
        AREAHU(j)        area of hot utility exchanger of cold stream j
        AREACU(i)        area of cold utility exchanger of hot stream i

        Cost_fix         fixed cost
        Cost_area        area cost
        Cost_utility     utility cost

        TAC              total annualized cost
        ;

POSITIVE VARIABLES   q(i,j,k), qhu(j), qcu(i),
               dt(i,j,k), dthul(j), dthur(j), dtcul(i), dtcur(i),
               LMTD(i,j,k),LMTDHU(j),LMTDCU(i),
               AREA(i,j,k),AREAHU(j),AREACU(i)
               ;

BINARY VARIABLES    z(i.j.k), zhu(j), zcu(i)   ;

//variable bounding//

        th.up(i,k) = THIN(i);
        th.lo(i,k) = THOUT(i);
        tc.up(j,k) = TCOUT(j);
        tc.lo(j,k) = TCIN(j);

        q.up(i,j,k)$stage(k) = max(min(  FCPH(i)*(th.up(i,k)-
max(th.lo(i,k+1),tc.lo(j,k+1)+EMAT))  ,   FCPC(j)*(min(th.up(i,k)-
EMAT,tc.up(j,k))-tc.lo(j,k+1))  )  ,  0)$( (th.up(i,k)-tc.lo(j,k) >= EMAT) and
(th.up(i.k+1)-tc.lo(j.k+1) >= EMAT))        +0$( (th.up(i,k)-tc.lo(j,k) < EMAT) or
(th.up(i,k+1)-tc.lo(j,k+1) < EMAT)) ;

        qhu.up(j)          = max(  FCPC(j)*(min(THUIN(j)-EMAT,TCOUT(j))-
tc.lo(j,'K1'))  ,  0)$((THUIN(j)-TCOUT(j) >= EMAT) and (THUOUT(j)-
tc.lo(j,'K1') >= EMAT)) +0$( (THUIN(j)-TCOUT(j) < EMAT) or (THUOUT(j)-
tc.lo(j,'K1') < EMAT)) ;
```

```
        qcu.up(i)          = max(  FCPH(i)*(th.up(i,'KL')-
max(THOUT(i),TCUIN(i)+EMAT))  ,  0)$( (th.up(i,'KL')-TCUOUT(i) >= EMAT)
and (THOUT(i)-TCUIN(i) >= EMAT)) +0$( (th.up(i,'KL')-TCUOUT(i) < EMAT) or
(THOUT(i)-TCUIN(i) < EMAT)) ;

        z.up(i,j,k)$stage(k) = 0$(q.up(i,j,k)=0)    +  1$(q.up(i,j,k) ne 0);
        zhu.up(j)          = 0$(qhu.up(j)  =0)    +  1$(qhu.up(j)   ne 0);
        zcu.up(i)          = 0$(qcu.up(i)  =0)    +  1$(qcu.up(i)   ne 0);

        dt.up(i,j,k) = max(EMAT,(th.up(i,k)-tc.lo(j,k)));
        dt.lo(i,j,k) = EMAT;
        dthul.up(j)  = max(EMAT,THUIN(j)-TCOUT(j));
        dthul.lo(j)  = EMAT;
        dthur.up(j)  = max(EMAT,THUOUT(j)-tc.lo(j,'K1'));
        dthur.lo(j)  = EMAT;
        dtcul.up(i)  = max(EMAT,th.up(i,'KL')-TCUOUT(i));
        dtcul.lo(i)  = EMAT;
        dtcur.up(i)  = max(EMAT,THOUT(i)-TCUIN(i));
        dtcur.lo(i)  = EMAT;

        LMTD.up(i,j,k)$stage(k)   =
(dt.up(i,j,k)*dt.up(i,j,k+1)*(dt.up(i,j,k)+dt.up(i,j,k+1))/2)**(1/3);
        LMTD.lo(i,j,k)$stage(k)   =
(dt.lo(i,j,k)*dt.lo(i,j,k+1)*(dt.lo(i,j,k)+dt.lo(i,j,k+1))/2)**(1/3);
        LMTDHU.up(j)              =
(dthur.up(j)*dthul.up(j)*(dthur.up(j)+dthul.up(j))/2)**(1/3);
        LMTDHU.lo(j)              =
(dthur.lo(j)*dthul.lo(j)*(dthur.lo(j)+dthul.lo(j))/2)**(1/3);
        LMTDCU.up(i)              =
(dtcul.up(i)*dtcur.up(i)*(dtcul.up(i)+dtcur.up(i))/2)**(1/3);
        LMTDCU.lo(i)              =
(dtcul.lo(i)*dtcur.lo(i)*(dtcul.lo(i)+dtcur.lo(i))/2)**(1/3);

        AREA.up(i,j,k)$stage(k)   = q.up(i,j,k)/LMTD.lo(i,j,k)/U(i,j);
        AREAHU.up(j)              = qhu.up(j)/LMTDHU.lo(j)/UHU(j);
        AREACU.up(i)              = qcu.up(i)/LMTDCU.lo(i)/UCU(i);

//Assign bounding to pamaters in logical constraint//
        GAMMA_HX(i,j,k) = q.up(i,j,k);
        GAMMA_HU(j)     = qhu.up(j);
        GAMMA_CU(i)     = qcu.up(i);

        TAU_HX(i,j,k)   =  -th.lo(i,k)+tc.up(j,k)+dt.lo(i,j,k);
        TAU_HUL(j)      =  -THUIN(j)+TCOUT(j)+dthul.lo(j);
        TAU_HUR(j)      =  -THUOUT(j)+tc.up(j,'K1')+dthur.lo(j);
        TAU_CUL(i)      =  -th.lo(i,'KL')+TCUOUT(i)+dtcul.lo(i);
        TAU_CUR(i)      =  -THOUT(i)+TCUIN(i)+dtcur.lo(i);
```

```
//Last stage forcing bound//
        q.fx(i,j,'KL')        =0;
        z.fx(i,j,'KL')        =0;
        LMTD.fx(i,j,'KL')     =0;
        AREA.fx(i,j,'KL')  =0;


EQUATIONS
        OVERALL_H(i)        overall energy balance of hot stream i
        OVERALL_C(j)        overall energy balance of cold stream j

        EBAL_H(i,k)         energy balance of hot stream i in stage k
        EBAL_C(j,k)         energy balance of cold stream j in stage k

        TIN_H(i)            assignment inlet temperature of hot stream i
        TIN_C(j)            assignment inlet temperature of cold stream j

        MONOT_H(i,k)        monotonic in temperature of hot stream i at location k
        MONOT_C(j,k)        monotonic in temperature of cold stream j at location k
        MONOT_HU(j)         monotonic in outlet temperature of cold stream j
        MONOT_CU(i)         monotonic in outlet temperature of hot stream i

        EBAL_HU(j)          energy balance of hot utility of cold stream j
        EBAL_CU(i)          energy balance of cold utility of hot stream i

        LOGIC_HX(i,j,k)     logical constraint to define z of match(i.j) at stage k
        LOGIC_HU(j)         logical constraint to define zhu of cold stream j at stage k
        LOGIC_CU(i)         logical constraint to define zcu of hot stream i at stage k

        DTFEAS_HXL(i,j,k)   temperature feasibility of heat exchanger of match(i.j)
                            at location k
        DTFEAS_HXR(i,j,k)   temperature feasibility of heat exchanger of match(i.j)
                            at location k+1
        DTFEAS_HUL(j)       temperature feasibility of hot utility of outlet cold
                            stream j
        DTFEAS_HUR(j)       temperature feasibility of hot utility of inlet cold stream j
        DTFEAS_CUL(i)       temperature feasibility of cold utility of inlet hot stream i
        DTFEAS_CUR(i)       temperature feasibility of cold utility of outlet hot stream i

        LOGMTD(i,j,k)       log mean temperature of match(i.j) at stage k
        LOGMTDHU(j)         log mean temperature of hot utility exchanger
        LOGMTDCU(i)         log mean temperature of cold utility exchanger

        A_HX(i,j,k)         area of heat exchanger of match(i.j) at stage k
        A_HU(j)             area of hot utility exchanger of cold stream j
        A_CU(i)             area of cold utility exchanger of hot stream i
```

```
     FIXCOST             fixed cost
     AREACOST              area cost
     UTILITYCOST            utility cost

     OBJ              objective function (total annualized cost)
     ;

OVERALL_H(i)              ..  (THIN(i) -THOUT(i))*FCPH(i)  =e=
                              sum((j,k),q(i,j,k))+qcu(i);
OVERALL_C(j)              ..  (TCOUT(j)-TCIN(j)) *FCPC(j)  =e=
                              sum((i,k),q(i,j,k))+qhu(j);

EBAL_H(i,k)$stage(k)      ..  (th(i,k)-th(i,k+1))*FCPH(i) =e=  sum(j,q(i,j,k));
EBAL_C(j,k)$stage(k)      ..  (tc(j,k)-tc(j,k+1))*FCPC(j) =e=  sum(i.q(i,j,k));

TIN_H(i)               ..  THIN(i)  =e=  th(i,'K1');
TIN_C(j)               ..  TCIN(j)  =e=  tc(j,'KL');

MONOT_H(i,k)$stage(k)     ..  th(i,k)  =g=  th(i,k+1);
MONOT_C(j,k)$stage(k)     ..  tc(j,k)  =g=  tc(j,k+1);
MONOT_HU(j)               ..  TCOUT(j) =g=  tc(j,'K1');
MONOT_CU(i)               ..  THOUT(i) =l=  th(i.'KL');

EBAL_HU(j)                ..  (TCOUT(j)   - tc(j,'K1'))*FCPC(j)  =e=  qhu(j);
EBAL_CU(i)                ..  (th(i,'KL')  -  THOUT(i))*FCPH(i)   =e=  qcu(i);

LOGIC_HX(i,j,k)$stage(k)  ..  q(i,j,k)  -  GAMMA_HX(i,j,k)*z(i,j,k)  =l=  0;
LOGIC_HU(j)               ..  qhu(j)   -  GAMMA_HU(j)   *zhu(j)   =l=  0;
LOGIC_CU(i)               ..  qcu(i)   -  GAMMA_CU(i)   *zcu(i)    =l=  0;

DTFEAS_HXL(i,j,k)$stage(k) ..  dt(i,j,k)  =l=  th(i,k)   -tc(j,k)   +TAU_HX(i,j,k)
                               *(1-z(i,j,k));
DTFEAS_HXR(i,j,k)$stage(k) ..  dt(i,j,k+1)  =l=  th(i,k+1) -tc(j.k+1)
                               +TAU_HX(i,j,k+1)*(1-z(i,j,k));
DTFEAS_HUL(j)              ..  dthul(j)    =l=  THUIN(j)  -TCOUT(j)  +TAU_HUL(j)
                               *(1-zhu(j));
DTFEAS_HUR(j)              ..  dthur(j)    =l=  THUOUT(j) -tc(j,'K1')+TAU_HUR(j)
                               *(1-zhu(j));
DTFEAS_CUL(i)              ..  dtcul(i)    =l=  th(i,'KL')-TCUOUT(i) +TAU_CUL(i)
                               *(1-zcu(i));
DTFEAS_CUR(i)              ..  dtcur(i)    =l=  THOUT(i)  -TCUIN(i)  +TAU_CUR(i)
                               *(1-zcu(i));


LOGMTD(i,j,k)$stage(k)     ..  LMTD(i,j,k)  =e=
                               (dt(i,j,k)*dt(i,j,k+1)*(dt(i,j,k)+dt(i,j,k+1))/2)**(1/3);
```

```
LOGMTDHU(j)            .. LMTDHU(j)   =e= (dthur(j) *dthul(j)  *(dthur(j)
                          +dthul(j))  /2)**(1/3);
LOGMTDCU(i)            .. LMTDCU(i)   =e= (dtcul(i) *dtcur(i)  *(dtcul(i)
                          +dtcur(i))  /2)**(1/3);

A_HX(i,j,k)$stage(k)  .. AREA(i,j,k)  *LMTD(i,j,k)*U(i,j)  =e= q(i,j,k);
A_HU(j)               .. AREAHU(j)   *LMTDHU(j) *UHU(j)  =e= qhu(j);
A_CU(i)               .. AREACU(i)   *LMTDCU(i) *UCU(i)  =e= qcu(i);

FIXCOST               .. cost_fix    =e= AF*(sum((i,j,k)$stage(k),CF*z(i,j,k)) +
                         sum(j,CF*zhu(j)) + sum(i,CF*zcu(i)));
AREACOST              .. cost_area   =e=
                         AF*(sum((i,j,k)$stage(k),CA*AREA(i,j,k)))
                         + AF*(sum(j,CA*AREAHU(j)))
                         +  AF*(sum(i,CA*AREACU(i)));
UTILITYCOST           .. cost_utility =e= sum(j,CHU*qhu(j)) +
                         sum(i,CCU*qcu(i));

OBJ                 .. TAC  =e=  cost_fix + cost_area + cost_utility;

model STAGEMODEL_SINGLEPERIOD /all/;

option iterlim=1e9;
option domlim=0;
option reslim=1e8;
STAGEMODEL_SINGLEPERIOD.optfile=0;

solve STAGEMODEL_SINGLEPERIOD using MINLP minimizing TAC;

display th.l,tc.l,
     z.l,zhu.l,zcu.l,
     q.l,qhu.l,qcu.l,
     AREA.l,AREAHU.l,AREACU.l,
     dt.l,dthur.l,dthul.l,dtcul.l,dtcur.l,
     LMTD.l,
     TAC.l,
     TAU_HX,
     q.up,q.lo,qhu.up,qhu.lo,qcu.up,qcu.lo,
     dt.up,dt.lo,dthul.lo,dthul.up, dthur.lo,dthur.up, dtcul.lo,dtcul.up,
     dtcur.lo,dtcur.up,
     cost_fix.l, cost_area.l, cost_utility.l;
```

Appendix B Source Code of Models for HEN Adaptation

The following text is source code which is entered in GAMS for each model used in HEN adaptation step. Please note that this is just an example for only one case.

B1 Model A

```
SETS
      I  hot streams                    /I1*I3/
      J  cold streams                   /J1*J4/
      K  index of stage or location  /K1*K4,KL/    //last number is no. of stages

      stage(k)            all stages
      HEN(i,j,k)          define where HX exists
      HEN_HOT(i,k)        define where HX exists 2 dimensions for hot streams
      HEN_COLD(j,k)       define where HX exists 2 dimensions for cold streams
      HU(j)               define where HU exists
      HUNEW_HU(j)         define where HU could have in cold stream
      HUNEW_CU(j)         define where CU could have in cold stream
      CU(i)               define where CU exists
      ;

//dynamic set for stage pointer
      stage(k) = yes;
      stage('KL') = no;

//dynamic set for existing HEN//    <---- To fix topology

      HEN(i,j,k)  =  no;

      TABLE Z_OLD(i,j,k) Existing HX from original HEN of each period
                    K1        K2        K3        K4

      I1.J1      1.000      1.000      1.000
      I1.J2                      1.000      1.000
      I1.J3                1.000
      I3.J1                1.000      1.000
      I3.J4      1.000
      ;

      loop((i,j,k)$(Z_OLD(i,j,k)=1), HEN(i,j,k)  = yes);


//dynamic set for existing Utility exchanger//      <---- To fix topology
   //hot utility
```

```
      HU(j)   = no;
      PARAMETER ZHU_OLD(j) Existing HU from original HEN of each period
      /J3 1.000/;
      loop(j$(ZHU_OLD(j)=1), HU(j)  = yes);
      HUNEW_HU('J1') = yes;
      HUNEW_HU('J2') = yes;
      HUNEW_CU('J4') = yes;
   //cold utility
      CU(i)   = no;
      PARAMETER ZCU_OLD(i) Existing CU from original HEN of each period
      /I1 1.000,   I2 1.000,   I3 1.000/;
      loop(i$(ZCU_OLD(i)=1), CU(i)  = yes);


//define dynamic set for existing HEN but only 2 dimensions (for hot and cold)//
      HEN_HOT(i,k)   =  no;
      HEN_COLD(j,k)  =  no;
      loop((i,j,k)$(HEN(i,j,k)), HEN_HOT(i,k)  = yes);
      loop((i,j,k)$(HEN(i,j,k)), HEN_COLD(j,k)  = yes);


SCALARS

//Cost//
      CA        per unit cost of heat exchanger area($ per unit)      /641.7/
      B         exponent for area cost(dimensionless)                 /1/
      CF        fixed cost of heat exchanger($ per unit)              /8333.3/

      CHU       per unit cost of hot utility($ per kW)                /115.2/
      CCU       fixed cost of heat exchanger($ per kW)                /1.3/

      AF        annualisation factor                                  /0.2/

//other//
      EMAT        exchanger minimum approach temperature(C)                /5/

      HHU         heat transfer coefficient of hot utility(kW|(m2.C))      /2/
      HCU         heat transfer coefficient of cold utility(kW|(m2.C))     /1/

      TAU         tau /500/
      GAMMA       gamma /1e6/
      ;

PARAMETERS
//Hot Streams//
      THIN(i)  inlet temperature of hot streams(C)
      /       I1    393
```

```
        I2    160
        I3    354
    /
    THOUT(i)  outlet temperature of hot streams(C)
    /   I1    60
        I2    40
        I3    60
    /
//Cold Streams//
    TCIN(j)   inlet temperature of cold streams(C)
    /   J1    72
        J2    62
        J3    220
        J4    253
    /
    TCOUT(j)  outlet temperature of cold streams(C)
    /   J1    356
        J2    210
        J3    370
        J4    284
    /
//Heat Capacity Flowrates//
    FCPH(i)   heat capaity flowrates fo hot streams(C)
    /   I1    201.6
        I2    185.1
        I3    137.4
    /
    FCPC(j)   heat capaity flowrates fo cold streams(C)
    /   J1    209.4
        J2    141.6
        J3    176.4
        J4    294.4
    /
//Utility//
    THUIN(j)  inlet temperature of hot utility(C)
    /   J1    400
        J2    400
        J3    400
        J4    15
    /
    THUOUT(j) outlet temperature of hot utility(C)
    /   J1    399
        J2    399
        J3    399
        J4    20
    /
    TCUIN(i)  inlet temperature of cold utility(C)
```

```
/       I1   15
        I2   15
        I3   15
/
TCUOUT(i) outlet temperature of cold utility(C)
/       I1   20
        I2   20
        I3   20
/
//Heat transfer coefficient//
   HH(i)  heat capaity flowrates of hot streams(kW|(m2.C))
   /       I1   2
           I2   2
           I3   2
   /
   HC(j)  heat capaity flowrates of cold streams(kW|(m2.C))
   /       J1   1.5
           J2   1.5
           J3   2
           J4   2
   /
//overall heat transfer coefficient//
   U(i,j)          Overall heat transfer coefficient for match(i.j)
   UHU(j)          Overall heat transfer coefficient for cold stream(j) and hot utility
   UCU(i)          Overall heat transfer coefficient for hot stream(i) and cold utility

//upper bound for logical constraints//
   GAMMA_HX(i,j,k)   upper bound heat load for match (i.j)
   GAMMA_HU(j)       upper bound heat load for hot utility
   GAMMA_CU(i)       upper bound heat load for cold utility

   TAU_HX(i,j,k)     upper bound of temperature difference for heat exchanger
   TAU_HUL(j)        upper bound of temperature difference for left side of hot
                     utility exchanger
   TAU_HUR(j)        upper bound of temperature difference for right side of hot
                     utility exchanger
   TAU_CUL(i)        upper bound of temperature difference for left side pf cold
                     utility exchanger
   TAU_CUR(i)        upper bound of temperature difference for right side of cold
                     utility exchanger
   ;

//Old value of variables from original HEN//
```

```
//Area//
   TABLE AREA_OLD(i,j,k) old area from original HEN of each period
              K1         K2         K3         K4

   I1.J1    680.055     166.126    751.970
   I1.J2                     1290.040     132.794
   I1.J3             597.190
   I3.J1             356.123    1470.846
   I3.J4    324.859
   ;

   PARAMETERS
   AREAHU_OLD(j) old area of HU from original HEN
   /J3 256.025/
   AREACU_OLD(i) old area of CU from original HEN
   /I1 117.036,    I2 541.396,    I3  97.762/
   ;

//Temperature//
   TABLE TH_OLD(i,k) old hot temperature from original HEN
         K1          K2          K3          K4          KL

   I1    406.000     293.498     231.178     88.555      79.812
   I2    160.000     160.000     160.000     160.000     160.000
   I3    362.000     268.540     234.038     86.252      86.252
   ;

   TABLE TC_OLD(j,k) old cold temperature from original HEN
         K1          K2          K3          K4          KL

   J1    365.000     255.334     216.175     72.000      72.000
   J2    210.000     210.000     210.000     74.710      62.000
   J3    272.717     272.717     220.000     220.000     220.000
   J4    290.000     250.000     250.000     250.000     250.000
   ;

//Heat load//
   TABLE Q_OLD(i,j,k) old heat load from original HEN
              K1         K2         K3         K4

   I1.J1   23062.852    3528.959   10162.028
   I1.J2                     19075.844    1792.156
   I1.J3             9246.623
   I3.J1             4706.050   20158.011
   I3.J4   12748.000
   ;
```

```
PARAMETERS
QHU_OLD(j) old heat load of HU from original HEN
/J3 17063.377/
QCU_OLD(i) old heat load of CU from original HEN
/I1  4061.538,   I2 23856.000,   I3  3580.739/
;

//Temperature Difference//
  TABLE DT_OLD(i,j,k) old temperature differene from original HEN
             K1        K2        K3        K4        KL

  I1.J1    41.000    38.165    15.003    16.555
  I1.J2                  21.178    13.844    17.812
  I1.J3            20.781    11.178
  I3.J1            13.206    17.863    14.252
  I3.J4    72.000    18.540
  ;

  PARAMETERS
  DTHUL_OLD(j) old left-side temperature differene of HU from original HEN
  /J3 30.000/
  DTHUR_OLD(j) old right-side temperature differene of HU from original
  HEN
  /J3 126.283/
  DTCUL_OLD(i) old left-side temperature differene of CU from original HEN
  /I1  59.812,   I2 140.000,   I3  66.252/
  DTCUR_OLD(i) old right-side temperature differene of CU from original HEN
  /I1 45.000,   I2 25.000,   I3 45.000/
  ;

//Log Mean Temperature Difference//
  TABLE LMTD_OLD(i,j,k) old log mean temperature from original HEN
             K1        K2        K3        K4

  I1.J1    39.565    24.783    15.766
  I1.J2                  17.252    15.745
  I1.J3            15.484
  I3.J1            15.417    15.989
  I3.J4    39.242
  ;

  PARAMETERS
  LMTDHU_OLD(j) old log mean temperature of HU from original HEN
  /J3 66.647/
  LMTDCU_OLD(i) old log mean temperature of CU from original HEN
  /I1 52.055,   I2 66.096,   I3 54.941/
  ;
```

```
//Overall heat transfer coefficient calculations//
      U(i,j) = 1/(1/HH(i)+1/HC(j));
      UHU(j)$(HU(j) or HUNEW_HU(j)) = 1/(1/HC(j)+1/HHU);
      UHU(j)$HUNEW_CU(j) = 1/(1/HC(j)+1/HCU);     //suppose it is cold utility
      UCU(i) = 1/(1/HH(i)+1/HCU);

VARIABLES
      th(i,k)         temperature(C) of hot streams i at location k
      tc(j,k)         temperature(C) of cold streams j at location k

      q(i,j,k)        heat load(kW) of match(i.j) at stage k
      qhu(j)          heat load(kW) of hot utility for cold stream j
      qcu(i)          heat load(kW) of cold utility for hot stream i

      zhu(j)          addition of hot utility
      zcu(i)          addition of cold utility

      dt(i,j,k)       temperature difference(C) for match(i.j) at location k
      dthul(j)        temperature difference(C) for left side of hot utility exchanger
      dthur(j)        temperature difference(C) for right side of hot utility exchanger
      dtcul(i)        temperature difference(C) for left side of cold utility exchanger
      dtcur(i)        temperature difference(C) for right side of cold utility exchanger

      lmtd(i,j,k)     log mean temperature difference of match(i.j) at stage k
      lmtdhu(j)       log mean temperature difference of hot utility exchanger for cold
                      stream j
      lmtdcu(i)       log mean temperature difference of cold utility exchanger for cold
                      stream i

      area(i,j,k)     area of heat exchanger of match(i.j) at stage k
      areahu(j)       area of hot utility exchanger of cold stream j
      areacu(i)       area of cold utility exchanger of hot stream i

      areahu_add(j)   additional area of hot utility exchanger of cold stream j
      areacu_add(i)   additional area of cold utility exchanger of hot stream i

      Cost_fix_add    additional fixed cost from new utility
      Cost_area_add   area cost
      Cost_utility    utility cost

      tot_area_add    total additional cost
      ;
POSITIVE VARIABLES  q(i,j,k), qcu(i),      // qhu(j)
            dt(i,j,k), dthul(j), dthur(j), dtcul(i), dtcur(i),
            lmtd(i,j,k).lmtdhu(j),lmtdcu(i),
            areahu(j),areacu(i),
            areahu_add(j), areacu_add(i);
```

```
BINARY VARIABLES     zhu(j), zcu(i);

//variable bounding//

        th.up(i,k) = THIN(i);
        th.lo(i,k) = THOUT(i);
        tc.up(j,k)$(HU(j) or HUNEW_HU(j)) = TCOUT(j);       //bound only exist HU
        tc.lo(j,k) = TCIN(j);

        q.up(i,j,k)$stage(k) = max(min(  FCPH(i)*(th.up(i,k)-
max(th.lo(i,k+1),tc.lo(j,k+1)+EMAT))  ,  FCPC(j)*(min(th.up(i,k)-
EMAT,tc.up(j,k))-tc.lo(j,k+1))  )  ,  0)$( (th.up(i,k)-tc.lo(j,k) >= EMAT) and
(th.up(i,k+1)-tc.lo(j,k+1) >= EMAT))        +0$( (th.up(i,k)-tc.lo(j,k) < EMAT) or
(th.up(i,k+1)-tc.lo(j,k+1) < EMAT)) ;

    //only for HU(j) that exist hot utility
        qhu.up(j)$(HU(j) or HUNEW_HU(j))            =
max(  FCPC(j)*(min(THUIN(j)-EMAT,TCOUT(j))-tc.lo(j,'K1'))  ,  0)$((THUIN(j)-
TCOUT(j) >= EMAT) and (THUOUT(j)-tc.lo(j,'K1') >= EMAT)) +0$( (THUIN(j)-
TCOUT(j) < EMAT) or (THUOUT(j)-tc.lo(j,'K1') < EMAT)) ;

        qhu.lo(j)$(HU(j) or HUNEW_HU(j)) = 0;
        qcu.up(i)           = max(  FCPH(i)*(th.up(i,'KL')-
max(THOUT(i),TCUIN(i)+EMAT))  ,  0)$( (th.up(i,'KL')-TCUOUT(i) >= EMAT)
and (THOUT(i)-TCUIN(i) >= EMAT)) +0$( (th.up(i,'KL')-TCUOUT(i) < EMAT) or
(THOUT(i)-TCUIN(i) < EMAT)) ;

        qcu.lo(i) = 0;

        dt.up(i,j,k) = max(EMAT,(th.up(i,k)-tc.lo(j,k)));
        dt.lo(i,j,k) = EMAT;
        dthul.up(j)$(HU(j) or HUNEW_HU(j))  = max(EMAT,THUIN(j)-TCOUT(j));
//bound only exist HU
        dthul.lo(j)  = EMAT;
        dthur.up(j)$(HU(j) or HUNEW_HU(j))  = max(EMAT,THUOUT(j)-
tc.lo(j,'K1'));     //bound only exist HU
        dthur.lo(j)  = EMAT;
        dtcul.up(i)  = max(EMAT,th.up(i,'KL')-TCUOUT(i));
        dtcul.lo(i)  = EMAT;
        dtcur.up(i)  = max(EMAT,THOUT(i)-TCUIN(i));
        dtcur.lo(i)  = EMAT;

        lmtd.up(i,j,k)$stage(k)  =
                    (dt.up(i,j,k)*dt.up(i,j,k+1)*(dt.up(i,j,k)+dt.up(i,j,k+1))/2)**(1/3);
        lmtd.lo(i,j,k)$stage(k)  =
                      (dt.lo(i,j,k)*dt.lo(i,j,k+1)*(dt.lo(i,j,k)+dt.lo(i,j,k+1))/2)**(1/3);
```

```
    lmtdhu.up(j)$(HU(j) or HUNEW_HU(j))         =
(dthur.up(j)*dthul.up(j)*(dthur.up(j)+dthul.up(j))/2)**(1/3);    //bound only exist HU
    lmtdhu.lo(j)      = (dthur.lo(j)*dthul.lo(j)*(dthur.lo(j)+dthul.lo(j))/2)**(1/3);
    lmtdcu.up(i)      = (dtcul.up(i)*dtcur.up(i)*(dtcul.up(i)+dtcur.up(i))/2)**(1/3);
    lmtdcu.lo(i)      = (dtcul.lo(i)*dtcur.lo(i)*(dtcul.lo(i)+dtcur.lo(i))/2)**(1/3);

    area.up(i,j,k)$stage(k)  = q.up(i,j,k)/lmtd.lo(i,j,k)/U(i,j);
    areahu.up(j)$(HU(j) or HUNEW_HU(j))   = qhu.up(j)/lmtdhu.lo(j)/UHU(j);
                                                    //bound only exist HU
    areacu.up(i)            = qcu.up(i)/lmtdcu.lo(i)/UCU(i);

//Assign bounding to pamaters in logical constraint//
    GAMMA_HX(i,j,k) = q.up(i,j,k);
    GAMMA_HU(j)$(HU(j) or HUNEW_HU(j))    = qhu.up(j);
    GAMMA_HU(j)$HUNEW_CU(j) = GAMMA;
    GAMMA_CU(i)    = qcu.up(i);

    TAU_HX(i,j,k)  =  -th.lo(i,k)+tc.up(j,k)+dt.lo(i,j,k);
    TAU_HUL(j)$(HU(j) or HUNEW_HU(j))  =  -THUIN(j)+TCOUT(j)
                                         +dthul.lo(j); //bound only exist HU
    TAU_HUL(j)$HUNEW_CU(j)     =  TAU;          //Arbitrary create a value
    TAU_HUR(j)$(HU(j) or HUNEW_HU(j))     =  -THUOUT(j)+tc.up(j,'K1')
                                         +dthur.lo(j); //bound only exist HU
    TAU_HUR(j)$HUNEW_CU(j)     =  TAU;          //Arbitrary create a value
    TAU_CUL(i)     =  -th.lo(i,'KL')+TCUOUT(i)+dtcul.lo(i);
    TAU_CUR(i)     =  -THOUT(i)+TCUIN(i)+dtcur.lo(i);

//Last stage forcing bound//
    q.fx(i,j,'KL')          = 0;
    lmtd.fx(i,j,'KL')       = 0;
    area.fx(i,j,'KL')       = 0;

//To force no q, area change where there is no existing HX//
    q.fx(i,j,k)$(not HEN(i,j,k))         =  0;
    area.fx(i,j,k)$(not HEN(i,j,k))      =  0;

//Fix area in the process-process HX//
    area.fx(i,j,k)$(HEN(i,j,k)) = AREA_OLD(i,j,k);     //and(not HEN('I3','J3','K1'))

//initial value for variables//
    areahu.l(j)       = AREAHU_OLD(j);
    areacu.l(i)       = AREACU_OLD(i);

    areahu_add.l(j)     =  0;
    areacu_add.l(i)     =  0;
```

```
th.l(i,k)        = TH_OLD(i,k);
tc.l(j,k)        = TC_OLD(j,k);

q.l(i,j,k)       = Q_OLD(i,j,k);
qhu.l(j)         = QHU_OLD(j);
qcu.l(i)         = QCU_OLD(i);

dt.l(i,j,k)      = DT_OLD(i,j,k);
dthul.l(j)       = DTHUL_OLD(j);
dthur.l(j)       = DTHUR_OLD(j);
dtcul.l(i)       = DTCUL_OLD(i);
dtcur.l(i)       = DTCUR_OLD(i);

lmtd.l(i,j,k)    = LMTD_OLD(i,j,k);
lmtdhu.l(j)      = LMTDHU_OLD(j);
lmtdcu.l(i)      = LMTDCU_OLD(i);

zhu.l(j)$(not HU(j)) = 0;
zcu.l(i)$(not CU(i)) = 0;

EQUATIONS
     OVERALL_H(i)       overall energy balance of hot stream i
     OVERALL_C(j)       overall energy balance of cold stream j

     EBAL_H(i,k)        energy balance of hot stream i in stage k
     EBAL_C(j,k)        energy balance of cold stream j in stage k

     TIN_H(i)           assignment inlet temperature of hot stream i
     TIN_C(j)           assignment inlet temperature of cold stream j

     MONOT_H(i,k)       monotonic in temperature of hot stream i at location k
     MONOT_C(j,k)       monotonic in temperature of cold stream j at location k
     MONOT_HNOT(i,k)    monotonic in temperature of hot stream i at location k
                        if no HX
     MONOT_CNOT(j,k)    monotonic in temperature of cold stream j at
                        location k if no HX

     MONOT_HU(j)        monotonic in outlet temperature of cold stream j
     MONOT_CU(i)        monotonic in outlet temperature of hot stream i

     EBAL_HU(j)         energy balance of hot utility of cold stream j
     EBAL_CU(i)         energy balance of cold utility of hot stream i
     EBAL_HUNEW_CU(j)   energy balance of hot utility of cold stream j
     EBAL_CUNOT(i)      energy balance of cold utility of hot stream i

     LOGIC_HUNEW_HU(j)  logical constraint to define zhu of cold stream j at
                        stage k (hot utility)
```

LOGIC_HUNEW_CU(j) logical constraint to define zhu of cold stream j at stage k (cold utility)
LOGIC_CU(i) logical constraint to define zcu of hot stream i at stage k

DTFEAS_HXL(i,j,k) temperature feasibility of heat exchanger of match(i.j) at location k
DTFEAS_HXR(i,j,k) temperature feasibility of heat exchanger of match(i.j) at location k+1
DTFEAS_HUL(j) temperature feasibility of hot utility of outlet cold stream j
DTFEAS_HUR(j) temperature feasibility of hot utility of inlet cold stream j
DTFEAS_CUL(i) temperature feasibility of cold utility of inlet hot stream i
DTFEAS_CUR(i) temperature feasibility of cold utility of outlet hot stream i

DTFEAS_HUL_HUNEW_HU(j) temperature feasibility of new hot utility of outlet cold stream j
DTFEAS_HUR_HUNEW_HU(j) temperature feasibility of new hot utility of inlet cold stream j
DTFEAS_HUL_HUNEW_CU(j) temperature feasibility of new cold utility of outlet cold stream j
DTFEAS_HUR_HUNEW_CU(j) temperature feasibility of new cold utility of inlet cold stream j

DTFEAS_CULNOT(i) temperature feasibility of cold utility of inlet hot stream i
DTFEAS_CURNOT(i) temperature feasibility of cold utility of outlet hot stream i

LOGMTD(i,j,k) log mean temperature of match(i.j) at stage k
LOGMTDHU(j) log mean temperature of hot utility exchanger
LOGMTDCU(i) log mean temperature of cold utility exchanger

ADDIAREA_HU1(j) additional area needed for hot utility
ADDIAREA_HU2(j) force ADDAREA_HU > 0
ADDIAREA_CU1(i) additional area needed for cold utility
ADDIAREA_CU2(i) force ADDAREA_CU > 0

A_HX(i,j,k) area of heat exchanger of match(i.j) at stage k
A_HU(j) area of hot utility exchanger of cold stream j
A_HUNOT(j) area for cold utility for J1 J2
A_CU(i) area of cold utility exchanger of hot stream i

OBJ objective function (total additional area)
;

OVERALL_H(i) .. (THIN(i) -THOUT(i))*FCPH(i) =e= sum((j,k),q(i,j,k))+qcu(i);

```
OVERALL_C(j)                .. (TCOUT(j)-TCIN(j)) *FCPC(j) =e=
                               sum((i,k),q(i,j,k))+qhu(j);

EBAL_H(i,k)$HEN_HOT(i,k)       .. (th(i,k)-th(i,k+1))*FCPH(i) =e=
                                  sum(j,q(i,j,k));    //do only exist HEN
EBAL_C(j,k)$HEN_COLD(j,k)      .. (tc(j,k)-tc(j,k+1))*FCPC(j) =e=
                                  um(i,q(i,j,k));

TIN_H(i)                 .. THIN(i) =e= th(i,'K1');
TIN_C(j)                 .. TCIN(j) =e= tc(j,'KL');

MONOT_H(i,k)$HEN_HOT(i,k)                   .. th(i,k) =g= th(i,k+1);   //if HX
                                                                 --> monotonic
MONOT_C(j,k)$HEN_COLD(j,k)                  .. tc(j,k) =g= tc(j,k+1);
MONOT_HNOT(i,k)$((not HEN_HOT(i,k)) and stage(k)) .. th(i,k) =e= th(i,k+1);
                                                             //if no HX --> equal
MONOT_CNOT(j,k)$((not HEN_COLD(j,k)) and stage(k)) .. tc(j,k) =e= tc(j,k+1);

MONOT_HU(j)$(HU(j) or HUNEW_HU(j)) .. TCOUT(j) =g= tc(j,'K1');
                                                         //if HX --> monotonic
MONOT_CU(i)$CU(i)              .. th(i,'KL') =g= THOUT(i);

EBAL_HU(j)$(HU(j) or HUNEW_HU(j))    .. (TCOUT(j)  - tc(j,'K1'))*FCPC(j)
                                        =e= qhu(j);  //do only exist HEN
EBAL_CU(i)$CU(i)              .. (th(i,'KL') - THOUT(i)) *FCPH(i) =e= qcu(i);
EBAL_HUNEW_CU(j)$HUNEW_CU(j)       .. (TCOUT(j)   - tc(j,'K1'))*FCPC(j)
                                        =e= qhu(j);
EBAL_CUNOT(i)$(not CU(i))  .. (th(i,'KL') - THOUT(i)) *FCPH(i) =e= qcu(i);

LOGIC_HUNEW_HU(j)$HUNEW_HU(j)       .. qhu(j)  - GAMMA_HU(j)
                                       *zhu(j)  =l= 0;
LOGIC_HUNEW_CU(j)$HUNEW_CU(j)       .. qhu(j)  + GAMMA_HU(j)
                          *zhu(j)  =g= 0;         //change - to + because qhu = -
LOGIC_CU(i)$(not CU(i))        .. qcu(i)  - GAMMA_CU(i)  *zcu(i)  =l= 0;

DTFEAS_HXL(i,j,k)$HEN(i,j,k)   .. dt(i,j,k)   =e= th(i,k)  -tc(j,k) ;
                                                      //if exist HEN  --> equal
DTFEAS_HXR(i,j,k)$HEN(i,j,k)   .. dt(i,j,k+1) =e= th(i,k+1) -tc(j,k+1);
DTFEAS_HUL(j)$HU(j)          .. dthul(j)   =e= THUIN(j) -TCOUT(j) ;
DTFEAS_HUR(j)$HU(j)          .. dthur(j)   =e= THUOUT(j) -tc(j,'K1');
                                                     //if exist utility  --> equal
DTFEAS_CUL(i)$CU(i)          .. dtcul(i)   =e= th(i,'KL')-TCUOUT(i) ;
DTFEAS_CUR(i)$CU(i)          .. dtcur(i)   =e= THOUT(i)  -TCUIN(i) ;
DTFEAS_HUL_HUNEW_HU(j)$HUNEW_HU(j)   .. dthul(j)   =l= THUIN(j) -
  TCOUT(j) +TAU_HUL(j)   *(1-zhu(j));      //if no utility --> use TAU and binary
DTFEAS_HUR_HUNEW_HU(j)$HUNEW_HU(j)   .. dthur(j)   =l= THUOUT(j)
                                     -tc(j,'K1')+TAU_HUR(j)   *(1-zhu(j));
```

```
DTFEAS_HUL_HUNEW_CU(j)$HUNEW_CU(j)    ..  dthul(j)    =l=  TCOUT(j) -
THUIN(j)  +TAU_HUL(j)    *(1-zhu(j));      //if no utility  --> use TAU and binary
DTFEAS_HUR_HUNEW_CU(j)$HUNEW_CU(j)    ..  dthur(j)    =l=  tc(j,'K1')
                                     -THUOUT(j) +TAU_HUR(j)    *(1-zhu(j));
DTFEAS_CULNOT(i)$(not CU(i))    ..  dtcul(i)    =l=  th(i,'KL')-TCUOUT(i)
                                     +TAU_CUL(i)    *(1-zcu(i));
DTFEAS_CURNOT(i)$(not CU(i))    ..  dtcur(i)    =l=  THOUT(i)  -TCUIN(i)
                                     +TAU_CUR(i)    *(1-zcu(i));

LOGMTD(i,j,k)$HEN(i,j,k)        ..  lmtd(i,j,k)  =e=
          (dt(i,j,k)*dt(i,j,k+1)*(dt(i,j,k)+dt(i,j,k+1))/2)**(1/3);     //only  HEN
LOGMTDHU(j)                   ..  lmtdhu(j)    =e=  (dthur(j) *dthul(j)   *(dthur(j)
                                  +dthul(j))  /2)**(1/3);
LOGMTDCU(i)                   ..  lmtdcu(i)    =e=  (dtcul(i) *dtcur(i)   *(dtcul(i)
                                  +dtcur(i))  /2)**(1/3);

A_HX(i,j,k)$HEN(i,j,k)  ..  area(i,j,k)  *lmtd(i,j,k)*U(i,j)   =e=  q(i,j,k); //only HEN
A_HU(j)$(HU(j) or HUNEW_HU(j)) .. areahu(j)    *lmtdhu(j)  *UHU(j)    =e= qhu(j);
A_HUNOT(j)$HUNEW_CU(j) ..  areahu(j)    *lmtdhu(j)  *UHU(j)    =e=  -qhu(j);
                                                                      //qhu = -
A_CU(i)                   ..  areacu(i)    *lmtdcu(i)  *UCU(i)    =e=  qcu(i);

ADDIAREA_HU1(j)          ..  areahu_add(j)    =g= areahu(j) - AREAHU_OLD(j);
ADDIAREA_HU2(j)          ..  areahu_add(j)    =g= 0;
ADDIAREA_CU1(i)          ..  areacu_add(i)    =g= areacu(i)  - AREACU_OLD(i);
ADDIAREA_CU2(i)          ..  areacu_add(i)    =g= 0;

//Only area penalty
OBJ                      ..  TOT_AREA_ADD =e= sum(j,areahu_add(j))  +
                             sum(i,areacu_add(i));

model STAGEMODEL_SINGLEPERIOD /all/;

option iterlim=1e9;
option domlim=0;
option reslim=1e8;
STAGEMODEL_SINGLEPERIOD.optfile=0;

solve STAGEMODEL_SINGLEPERIOD using MINLP minimizing
TOT_AREA_ADD;
```

```
display th.l,tc.l,
      q.l,qhu.l,qcu.l,
      zhu.l, zcu.l,
      area.l,areahu.l,areacu.l,
      dt.l,dthul.l,dthur.l,dtcul.l,dtcur.l,
      lmtd.l,lmtdhu.l,lmtdcu.l,
      tot_area_add.l,
      area.lo,area.up,areahu.lo,areahu.up,areacu.lo,areacu.up,
      q.up,q.lo,qhu.up,qhu.lo,qcu.up,qcu.lo
      dt.up,dt.lo,dthul.lo,dthul.up, dthur.lo,dthur.up, dtcul.lo,dtcul.up,
      dtcur.lo,dtcur.up
      ;
```

B2 Model B

```
SETS
       I  hot streams                    /I1*I3/
       J  cold streams                   /J1*J4/
       K  index of stage or location  /K1*K4,KL/     //last number is no. of stages

       stage(k)                      all stages
       HEN(i,j,k)                    define where HX exists
       HEN_HOT(i,k)                  define where HX exists 2 dimensions for hot streams
       HEN_COLD(j,k) d               efine where HX exists 2 dimensions for cold streams
       HU(j)                         define where HU exists
       CU(i)                         define where CU exists
       ;

//dynamic set for stage pointer
       stage(k) = yes;
       stage('KL') = no;

//dynamic set for existing HEN//     <---- To fix topology

       HEN(i,j,k)  =  no;

       TABLE Z_OLD(i,j,k) Existing HX from original HEN of each period
                  K1        K2        K3        K4

       I1.J1      1.000     1.000     1.000
       I1.J2                          1.000     1.000
       I1.J3                1.000
       I3.J1                1.000     1.000
       I3.J4      1.000
       ;

       loop((i,j,k)$(Z_OLD(i,j,k)=1), HEN(i,j,k)   = yes);


//dynamic set for existing Utility exchanger//        <---- To fix topology
    //hot utility
       HU(j)    = no;
       PARAMETER ZHU_OLD(j) Existing HU from original HEN of each period
       /J3 1.000/;
       loop(j$(ZHU_OLD(j)=1), HU(j)   = yes);

    //cold utility
       CU(i)    = no;
```

```
      PARAMETER ZCU_OLD(i) Existing CU from original HEN of each period
      /I1 1.000,   I2 1.000,   I3 1.000/;
      loop(i$(ZCU_OLD(i)=1), CU(i)  = yes);


//define dynamic set for existing HEN but only 2 dimensions (for hot and cold)//
      HEN_HOT(i,k)   = no;
      HEN_COLD(j,k)  = no;
      loop((i,j,k)$(HEN(i,j,k)). HEN_HOT(i,k)  = yes);
      loop((i,j,k)$(HEN(i,j,k)), HEN_COLD(j,k) = yes);


SCALARS

//Cost//
      CA        per unit cost of heat exchanger area($ per unit)     /641.7/
      B         exponent for area cost(dimensionless)                /1/
      CF        fixed cost of heat exchanger($ per unit)             /8333.3/

      CHU       per unit cost of hot utility($ per kW)               /115.2/
      CCU       per unit cost of cold utility($ per kW)              /1.3/

      AF        annualisation factor                                 /0.2/

//other//
      EMAT      exchanger minimum approach temperature(C)            /5/

      HHU       heat transfer coefficient of hot utility(kW|(m2.C))  /2/
      HCU       heat transfer coefficient of cold utility(kW|(m2.C)) /1/

      ;


PARAMETERS
//Hot Streams//
      THIN(i)  inlet temperature of hot streams(C)
      /     I1   393
            I2   160
            I3   354
      /
      THOUT(i)  outlet temperature of hot streams(C)
      /     I1   60
            I2   40
            I3   60
      /
//Cold Streams//
      TCIN(j)  inlet temperature of cold streams(C)
      /     J1   72
```

```
          J2    62
          J3    220
          J4    253
      /
      TCOUT(j)  outlet temperature of cold streams(C)
      /   J1    356
          J2    210
          J3    370
          J4    284
      /
//Heat Capacity Flowrates//
      FCPH(i)   heat capaity flowrates fo hot streams(C)
      /   I1    201.6
          I2    185.1
          I3    137.4
      /
      FCPC(j)   heat capaity flowrates fo cold streams(C)
      /   J1    209.4
          J2    141.6
          J3    176.4
          J4    294.4
      /
//Utility//
      THUIN(j)  inlet temperature of hot utility(C)
      /   J1    400
          J2    400
          J3    400
          J4    400
      /
      THUOUT(j) outlet temperature of hot utility(C)
      /   J1    399
          J2    399
          J3    399
          J4    399
      /
      TCUIN(i)  inlet temperature of cold utility(C)
      /   I1    15
          I2    15
          I3    15
      /
      TCUOUT(i) outlet temperature of cold utility(C)
      /   I1    20
          I2    20
          I3    20
      /
//Heat transfer coefficient//
      HH(i)   heat capaity flowrates of hot streams(kW|(m2.C))
```

```
       /      I1    2
              I2    2
              I3    2
       /
       HC(j)  heat capaity flowrates of cold streams(kW|(m2.C))
       /      J1    1.5
              J2    1.5
              J3    2
              J4    2
       /
//overall heat transfer coefficient//
       U(i,j)         Overall heat transfer coefficient for match(i.j)
       UHU(j)         Overall heat transfer coefficient for cold stream(j) and hot utility
       UCU(i)         Overall heat transfer coefficient for hot stream(i) and cold utility

//upper bound for logical constraints//
       GAMMA_HX(i,j,k)   upper bound heat load for match (i.j)
       GAMMA_HU(j)       upper bound heat load for hot utility
       GAMMA_CU(i)       upper bound heat load for cold utility

       TAU_HX(i,j,k)     upper bound of temperature difference for heat exchanger
       TAU_HUL(j)        upper bound of temperature difference for left side of hot
                         utility exchanger
       TAU_HUR(j)        upper bound of temperature difference for right side of hot
                         utility exchanger
       TAU_CUL(i)        upper bound of temperature difference for left side pf cold
                         utility exchanger
       TAU_CUR(i)        upper bound of temperature difference for right side of cold
                         utility exchanger
       ;

//Old value of variables from original HEN//

    //Area//
       TABLE AREA_OLD(i,j,k) old area from original HEN of each period
                  K1         K2         K3         K4

       I1.J1    680.055    166.126    751.970
       I1.J2                       1290.040    132.794
       I1.J3             597.190
       I3.J1             356.123   1470.846
       I3.J4    324.859
       ;

       PARAMETERS
       AREAHU_OLD(j) old area of HU from original HEN
       /J3 256.025/
```

```
   AREACU_OLD(i) old area of CU from original HEN
   /I1 117.036,   I2 541.396,   I3  97.762/
   ;

//Temperature//
   TABLE TH_OLD(i,k) old hot temperature from original HEN
          K1         K2         K3         K4         KL

   I1    406.000    293.498    231.178    88.555     79.812
   I2    160.000    160.000    160.000    160.000    160.000
   I3    362.000    268.540    234.038    86.252     86.252
   ;

   TABLE TC_OLD(j,k) old cold temperature from original HEN
          K1         K2         K3         K4         KL

   J1    365.000    255.334    216.175    72.000     72.000
   J2    210.000    210.000    210.000    74.710     62.000
   J3    272.717    272.717    220.000    220.000    220.000
   J4    290.000    250.000    250.000    250.000    250.000
   ;

//Heat load//
   TABLE Q_OLD(i,j,k) old heat load from original HEN
              K1         K2         K3         K4

   I1.J1   23062.852   3528.959   10162.028
   I1.J2                          19075.844   1792.156
   I1.J3               9246.623
   I3.J1               4706.050   20158.011
   I3.J4   12748.000
   ;
   PARAMETERS
   QHU_OLD(j) old heat load of HU from original HEN
   /J3 17063.377/
   QCU_OLD(i) old heat load of CU from original HEN
   /I1  4061.538,   I2 23856.000,   I3  3580.739/
   ;

//Temperature Difference//
   TABLE DT_OLD(i,j,k) old temperature differene from original HEN
              K1         K2         K3         K4         KL

   I1.J1     41.000     38.165     15.003     16.555
   I1.J2                           21.178     13.844     17.812
   I1.J3                20.781     11.178
   I3.J1                13.206     17.863     14.252
```

```
I3.J4     72.000     18.540
;

PARAMETERS
DTHUL_OLD(j) old left-side temperature differene of HU from original HEN
/J3 30.000/
DTHUR_OLD(j) old right-side temperature differene of HU from original HEN
/J3 126.283/
DTCUL_OLD(i) old left-side temperature differene of CU from original HEN
/I1  59.812,   I2 140.000,   I3  66.252/
DTCUR_OLD(i) old right-side temperature differene of CU from original HEN
/I1 45.000,   I2 25.000,   I3 45.000/
;

//Log Mean Temperature Difference//
TABLE LMTD_OLD(i,j,k) old log mean temperature from original HEN
            K1        K2        K3        K4

I1.J1     39.565    24.783    15.766
I1.J2                         17.252    15.745
I1.J3               15.484
I3.J1               15.417    15.989
I3.J4     39.242
;

PARAMETERS
LMTDHU_OLD(j) old log mean temperature of HU from original HEN
/J3 66.647/
LMTDCU_OLD(i) old log mean temperature of CU from original HEN
/I1 52.055,   I2 66.096,   I3 54.941/
;
//Overall heat transfer coefficient calculations//
U(i,j)  =  1/(1/HH(i)+1/HC(j));
UHU(j)  =  1/(1/HC(j)+1/HHU);
UCU(i)  =  1/(1/HH(i)+1/HCU);


VARIABLES
th(i,k)        temperature(C) of hot streams i at location k
tc(j,k)        temperature(C) of cold streams j at location k

q(i,j,k)       heat load(kW) of match(i.j) at stage k
qhu(j)         heat load(kW) of hot utility for cold stream j
qcu(i)         heat load(kW) of cold utility for hot stream i

dt(i,j,k)      temperature difference(C) for match(i.j) at location k
```

```
     dthul(j)       temperature difference(C) for left side of hot utility exchanger
     dthur(j)       temperature difference(C) for right side of hot utility exchanger
     dtcul(i)       temperature difference(C) for left side of cold utility exchanger
     dtcur(i)       temperature difference(C) for right side of cold utility exchanger

     lmtd(i,j,k)     log mean temperature difference of match(i.j) at stage k
     lmtdhu(j)       log mean temperature difference of hot utility exchanger for cold
                     stream j
     lmtdcu(i)       log mean temperature difference of cold utility exchanger for cold
                     stream i

     area(i,j,k)     area of heat exchanger of match(i.j) at stage k
     areahu(j)       area of hot utility exchanger of cold stream j
     areacu(i)       area of cold utility exchanger of hot stream i

     tot_lsqr_error   total least square error
     ;

POSITIVE VARIABLES   q(i,j,k), qhu(j), qcu(i),
             dt(i,j,k), dthul(j), dthur(j), dtcul(i), dtcur(i),
             lmtd(i,j,k),lmtdhu(j),lmtdcu(i),
             area(i,j,k),areahu(j),areacu(i)
             area_add(i,j,k), areahu_add(j), areacu_add(i);

//variable bounding//

     th.up(i,k) = THIN(i);
     th.lo(i,k) = THOUT(i);
     tc.up(j,k) = TCOUT(j);
     tc.lo(j,k) = TCIN(j);

     q.up(i,j,k)$stage(k) = max(min(  FCPH(i)*(th.up(i,k)-
max(th.lo(i,k+1),tc.lo(j,k+1)+EMAT))  ,  FCPC(j)*(min(th.up(i,k)-
EMAT,tc.up(j,k))-tc.lo(j,k+1))  )  ,  0)$((th.up(i,k)-tc.lo(j,k) >= EMAT) and
(th.up(i,k+1)-tc.lo(j,k+1) >= EMAT))        +0$( (th.up(i,k)-tc.lo(j,k) < EMAT) or
(th.up(i,k+1)-tc.lo(j,k+1) < EMAT)) ;

     qhu.up(j)          = max(  FCPC(j)*(min(THUIN(j)-EMAT,TCOUT(j))-
tc.lo(j,'K1'))  ,  0)$((THUIN(j)-TCOUT(j) >= EMAT) and (THUOUT(j)-
tc.lo(j,'K1') >= EMAT)) +0$( (THUIN(j)-TCOUT(j) < EMAT) or (THUOUT(j)-
tc.lo(j,'K1') < EMAT)) ;

     qcu.up(i)          = max(  FCPH(i)*(th.up(i,'KL')-
max(THOUT(i),TCUIN(i)+EMAT))  ,  0)$( (th.up(i,'KL')-TCUOUT(i) >= EMAT)
and (THOUT(i)-TCUIN(i) >= EMAT)) +0$( (th.up(i,'KL')-TCUOUT(i) < EMAT) or
(THOUT(i)-TCUIN(i) < EMAT)) ;
```

```
dt.up(i,j,k)  = max(EMAT,(th.up(i,k)-tc.lo(j,k)));
dt.lo(i,j,k)  = EMAT;
dthul.up(j)  = max(EMAT,THUIN(j)-TCOUT(j));
dthul.lo(j)  = EMAT;
dthur.up(j)  = max(EMAT,THUOUT(j)-tc.lo(j,'K1'));
dthur.lo(j)  = EMAT;
dtcul.up(i)  = max(EMAT,th.up(i,'KL')-TCUOUT(i));
dtcul.lo(i)  = EMAT;
dtcur.up(i)  = max(EMAT,THOUT(i)-TCUIN(i));
dtcur.lo(i)  = EMAT;

lmtd.up(i,j,k)$stage(k)  =
        (dt.up(i,j,k)*dt.up(i,j,k+1)*(dt.up(i,j,k)+dt.up(i,j,k+1))/2)**(1/3);
lmtd.lo(i,j,k)$stage(k)  =
        (dt.lo(i,j,k)*dt.lo(i,j,k+1)*(dt.lo(i,j,k)+dt.lo(i,j,k+1))/2)**(1/3);
lmtdhu.up(j)   = (dthur.up(j)*dthul.up(j)*(dthur.up(j)+dthul.up(j))/2)**(1/3);
lmtdhu.lo(j)   = (dthur.lo(j)*dthul.lo(j)*(dthur.lo(j)+dthul.lo(j))/2)**(1/3);
lmtdcu.up(i)   = (dtcul.up(i)*dtcur.up(i)*(dtcul.up(i)+dtcur.up(i))/2)**(1/3);
lmtdcu.lo(i)   = (dtcul.lo(i)*dtcur.lo(i)*(dtcul.lo(i)+dtcur.lo(i))/2)**(1/3);

area.up(i,j,k)$stage(k) = q.up(i,j,k)/lmtd.lo(i,j,k)/U(i,j);
areahu.up(j)            = qhu.up(j)/lmtdhu.lo(j)/UHU(j);
areacu.up(i)            = qcu.up(i)/lmtdcu.lo(i)/UCU(i);

//Assign bounding to pamaters in logical constraint//
GAMMA_HX(i,j,k) = q.up(i,j,k);
GAMMA_HU(j)     = qhu.up(j);
GAMMA_CU(i)     = qcu.up(i);

TAU_HX(i,j,k)  =  -th.lo(i,k)+tc.up(j,k)+dt.lo(i,j,k);
TAU_HUL(j)     =  -THUIN(j)+TCOUT(j)+dthul.lo(j);
TAU_HUR(j)     =  -THUOUT(j)+tc.up(j,'K1')+dthur.lo(j);
TAU_CUL(i)     =  -th.lo(i,'KL')+TCUOUT(i)+dtcul.lo(i);
TAU_CUR(i)     =  -THOUT(i)+TCUIN(i)+dtcur.lo(i);

//Last stage forcing bound//
q.fx(i,j,'KL')           =0;
lmtd.fx(i,j,'KL')        =0;
area.fx(i,j,'KL')        =0;
area_add.fx(i,j,'KL')    =0;

//To force no q, area change where there is no existing HX//
q.fx(i,j,k)$(not HEN(i,j,k))         =  0;
area_add.fx(i,j,k)$(not HEN(i,j,k))  =  0;
```

```
//initial value for variables//
        area.l(i,j,k)       = AREA_OLD(i,j,k);
        areahu.l(j)         = AREAHU_OLD(j);
        areacu.l(i)         = AREACU_OLD(i);

        th.l(i,k)           = TH_OLD(i,k);
        tc.l(j,k)           = TC_OLD(j,k);

        q.l(i,j,k)          = Q_OLD(i,j,k);
        qhu.l(j)            = QHU_OLD(j);
        qcu.l(i)            = QCU_OLD(i);

        dt.l(i,j,k)         = DT_OLD(i,j,k);
        dthul.l(j)          = DTHUL_OLD(j);
        dthur.l(j)          = DTHUR_OLD(j);
        dtcul.l(i)          = DTCUL_OLD(i);
        dtcur.l(i)          = DTCUR_OLD(i);

        lmtd.l(i,j,k)       = LMTD_OLD(i,j,k);
        lmtdhu.l(j)         = LMTDHU_OLD(j);
        lmtdcu.l(i)         = LMTDCU_OLD(i);


EQUATIONS
        OVERALL_H(i)        overall energy balance of hot stream i
        OVERALL_C(j)        overall energy balance of cold stream j

        EBAL_H(i,k)         energy balance of hot stream i in stage k
        EBAL_C(j,k)         energy balance of cold stream j in stage k

        TIN_H(i)            assignment inlet temperature of hot stream i
        TIN_C(j)            assignment inlet temperature of cold stream j

        MONOT_H(i,k)        monotonic in temperature of hot stream i at location k
        MONOT_C(j,k)        monotonic in temperature of cold stream j at location k
        MONOT_HNOT(i,k)     monotonic in temperature of hot stream i at location k
                            if no HX
        MONOT_CNOT(j,k)     monotonic in temperature of cold stream j at location
                            k if no HX

      MONOT_HU(j)         monotonic in outlet temperature of cold stream j
      MONOT_CU(i)         monotonic in outlet temperature of hot stream i
      MONOT_HUNOT(j)  monotonic in outlet temperature of cold stream j if no HX
      MONOT_CUNOT(i)  monotonic in outlet temperature of hot stream i if no HX

        EBAL_HU(j)          energy balance of hot utility of cold stream j
        EBAL_CU(i)          energy balance of cold utility of hot stream i
```

```
DTFEAS_HXL(i,j,k)   temperature feasibility of heat exchanger of match(i.j)
                    at location k
DTFEAS_HXR(i,j,k)   temperature feasibility of heat exchanger of match(i.j)
                    at location k+1
DTFEAS_HUL(j)       temperature feasibility of hot utility of outlet cold
                    stream j
DTFEAS_HUR(j)       temperature feasibility of hot utility of inlet cold stream j
DTFEAS_CUL(i)       temperature feasibility of cold utility of inlet hot stream i
DTFEAS_CUR(i)       temperature feasibility of cold utility of outlet hot stream i

LOGMTD(i,j,k)       log mean temperature of match(i.j) at stage k
LOGMTDHU(j)         log mean temperature of hot utility exchanger
LOGMTDCU(i)         log mean temperature of cold utility exchanger

A_HX(i,j,k)         area of heat exchanger of match(i.j) at stage k
A_HU(j)             area of hot utility exchanger of cold stream j
A_CU(i)             area of cold utility exchanger of hot stream i

OBJ                 objective function (total additional area)
;

OVERALL_H(i)        .. (THIN(i) -THOUT(i))*FCPH(i) =e=
                       sum((j,k),q(i,j,k))+qcu(i);
OVERALL_C(j)        .. (TCOUT(j)-TCIN(j)) *FCPC(j) =e=
                       sum((i,k),q(i,j,k))+qhu(j);

EBAL_H(i,k)$HEN_HOT(i,k)     .. (th(i,k)-th(i,k+1))*FCPH(i) =e=
                                sum(j,q(i,j,k));        //do only exist HEN
EBAL_C(j,k)$HEN_COLD(j,k)    .. (tc(j,k)-tc(j,k+1))*FCPC(j) =e=
                                sum(i,q(i,j,k));

TIN_H(i)            .. THIN(i) =e= th(i,'K1');
TIN_C(j)            .. TCIN(j) =e= tc(j,'KL');

MONOT_H(i,k)$HEN_HOT(i,k)               .. th(i,k) =g= th(i,k+1);
                                                        //if HX --> monotonic
MONOT_C(j,k)$HEN_COLD(j,k)              .. tc(j,k) =g= tc(j,k+1);
MONOT_HNOT(i,k)$((not HEN_HOT(i,k)) and stage(k)) .. th(i,k) =e= th(i,k+1);
                                                        //if no HX --> equal
MONOT_CNOT(j,k)$((not HEN_COLD(j,k)) and stage(k)) .. tc(j,k) =e= tc(j,k+1);

MONOT_HU(j)$HU(j)       .. TCOUT(j) =g= tc(j,'K1');    //if HX --> monotonic
MONOT_CU(i)$CU(i)       .. th(i,'KL') =g= THOUT(i);
MONOT_HUNOT(j)$(not HU(j)) .. TCOUT(j) =e= tc(j,'K1'); //if no HX --> equal
MONOT_CUNOT(i)$(not CU(i)) .. th(i,'KL') =e= THOUT(i);
```

```
EBAL_HU(j)$HU(j)              .. (TCOUT(j)  - tc(j,'K1'))*FCPC(j) =e= qhu(j);
                                                          //do only exist HEN
EBAL_CU(i)$CU(i)              .. (th(i,'KL') - THOUT(i)) *FCPH(i) =e= qcu(i);

DTFEAS_HXL(i,j,k)$HEN(i,j,k)   .. dt(i,j,k)   =e= th(i,k)  -tc(j,k) ;
DTFEAS_HXR(i,j,k)$HEN(i,j,k)   .. dt(i,j,k+1) =e= th(i,k+1) -tc(j,k+1);
DTFEAS_HUL(j)$HU(j)            .. dthul(j)    =e= THUIN(j) -TCOUT(j) ;
DTFEAS_HUR(j)$HU(j)            .. dthur(j)    =e= THUOUT(j) -tc(j,'K1');
                                                    //if exist HEN and no TAU
DTFEAS_CUL(i)$CU(i)            .. dtcul(i)    =e= th(i,'KL')-TCUOUT(i) ;
DTFEAS_CUR(i)$CU(i)            .. dtcur(i)    =e= THOUT(i) -TCUIN(i) ;

LOGMTD(i,j,k)$HEN(i,j,k)       .. lmtd(i,j,k) =e=
(dt(i,j,k)*dt(i,j,k+1)*(dt(i,j,k)+dt(i,j,k+1))/2)**(1/3);     //only HEN
LOGMTDHU(j)$HU(j)   .. lmtdhu(j)   =e= (dthur(j) *dthul(j)  *(dthur(j)
                        +dthul(j))  /2)**(1/3);
LOGMTDCU(i)$CU(i)          .. lmtdcu(i)   =e= (dtcul(i) *dtcur(i)
                              *(dtcul(i) +dtcur(i))  /2)**(1/3);

A_HX(i,j,k)$HEN(i,j,k)         .. area(i,j,k) *lmtd(i,j.k)*U(i,j)  =e= q(i,j,k);
                                                               //only HEN
A_HU(j)$HU(j)              .. areahu(j)  *lmtdhu(j) *UHU(j)  =e= qhu(j);
A_CU(i)$CU(i)              .. areacu(i)  *lmtdcu(i) *UCU(i)  =e= qcu(i);

//least square error
OBJ                    .. TOT_LSQR_ERROR =e=
                       sum((i,j,k)$HEN(i,j,k),power(area(i,j,k)-AREA_OLD(i,j,k),2))
                       + sum(j$HU(j),power(areahu(j)-AREAHU_OLD(j),2))
                       + sum(i$CU(i),power(areacu(i)-AREACU_OLD(i),2));


model STAGEMODEL_SINGLEPERIOD /all/;

option iterlim=1e9;
option domlim=0;
option reslim=1e8;
STAGEMODEL_SINGLEPERIOD.optfile=0;

solve STAGEMODEL_SINGLEPERIOD using NLP minimizing
TOT_LSQR_ERROR;
```

```
display th.l,tc.l,
        q.l,qhu.l,qcu.l,
        area.l,areahu.l,areacu.l,
        dt.l,dthul.l,dthur.l,dtcul.l,dtcur.l,
        lmtd.l,lmtdhu.l,lmtdcu.l,
        tot_lsqr_error.l
        ;
```

B3 Model C

```
SETS
      I  hot streams                    /I1*I3/
      J  cold streams                   /J1*J4/
      K  index of stage or location     /K1*K4,KL/    //last number is no. of stages

      stage(k)                  all stages
      HEN(i,j,k)                define where HX exists
      HEN_HOT(i,k)              define where HX exists 2 dimensions for hot streams
      HEN_COLD(j,k)             define where HX exists 2 dimensions for cold streams
      HU(j)                     define where HU exists
      CU(i)                     define where CU exists
      ;

//dynamic set for stage pointer
      stage(k) = yes;
      stage('KL') = no;

//dynamic set for existing HEN//     <---- To fix topology

      HEN(i,j,k)  =  no;

      TABLE Z_OLD(i,j,k) Existing HX from original HEN of each period
                K1        K2        K3        K4

      I1.J1     1.000      1.000      1.000
      I1.J2                      1.000      1.000
      I1.J3           1.000
      I3.J1           1.000      1.000
      I3.J4     1.000
      ;

      loop((i,j,k)$(Z_OLD(i,j,k)=1), HEN(i,j,k)  = yes);

//dynamic set for existing Utility exchanger//        <---- To fix topology
   //hot utility
      HU(j)    = no;
      PARAMETER ZHU_OLD(j) Existing HU from original HEN of each period
      /J3 1.000/;
      loop(j$(ZHU_OLD(j)=1), HU(j)   = yes);

   //cold utility
      CU(i)    = no;
      PARAMETER ZCU_OLD(i) Existing CU from original HEN of each period
      /I1 1.000,   I2 1.000,   I3 1.000/;
      loop(i$(ZCU_OLD(i)=1), CU(i)   = yes);
```

```
//define dynamic set for existing HEN but only 2 dimensions (for hot and cold)//
        HEN_HOT(i,k)   = no;
        HEN_COLD(j,k)  = no;
        loop((i,j,k)$(HEN(i,j,k)), HEN_HOT(i,k)  = yes);
        loop((i,j,k)$(HEN(i,j,k)), HEN_COLD(j,k)  = yes);

SCALARS

//Cost//
        CA        per unit cost of heat exchanger area($ per unit)      /641.7/
        B         exponent for area cost(dimensionless)                 /1/
        CF        fixed cost of heat exchanger($ per unit)              /8333.3/

        CHU       per unit cost of hot utility($ per kW)                /115.2/
        CCU       fixed cost of heat exchanger($ per kW)                /1.3/

        AF        annualisation factor                                  /0.2/

//other//
        EMAT      exchanger minimum approach temperature(C)             /5/

        HHU       heat transfer coefficient of hot utility(kW|(m2.C))   /2/
        HCU       heat transfer coefficient of cold utility(kW|(m2.C))  /1/

        ;

PARAMETERS
//Hot Streams//
        THIN(i)   inlet temperature of hot streams(C)
        /     I1   393
              I2   160
              I3   354
        /
        THOUT(i)  outlet temperature of hot streams(C)
        /     I1   60
              I2   40
              I3   60
        /
//Cold Streams//
        TCIN(j)   inlet temperature of cold streams(C)
        /     J1   72
              J2   62
              J3   220
              J4   253
        /
        TCOUT(j)  outlet temperature of cold streams(C)
```

```
/       J1    356
        J2    210
        J3    370
        J4    284
/
//Heat Capacity Flowrates//
    FCPH(i)  heat capaity flowrates fo hot streams(C)
/       I1    201.6
        I2    185.1
        I3    137.4
/
    FCPC(j)  heat capaity flowrates fo cold streams(C)
/       J1    209.4
        J2    141.6
        J3    176.4
        J4    294.4
/
//Utility//
    THUIN(j)  inlet temperature of hot utility(C)
/       J1    400
        J2    400
        J3    400
        J4    400
/
    THUOUT(j) outlet temperature of hot utility(C)
/       J1    399
        J2    399
        J3    399
        J4    399
/
    TCUIN(i)  inlet temperature of cold utility(C)
/       I1    15
        I2    15
        I3    15
/
    TCUOUT(i) outlet temperature of cold utility(C)
/       I1    20
        I2    20
        I3    20
/
//Heat transfer coefficient//
    HH(i)   heat capaity flowrates of hot streams(kW|(m2.C))
/       I1    2
        I2    2
        I3    2
/
    HC(j)   heat capaity flowrates of cold streams(kW|(m2.C))
```

```
/       J1    1.5
        J2    1.5
        J3    2
        J4    2
/
//overall heat transfer coefficient//
      U(i,j)          Overall heat transfer coefficient for match(i.j)
      UHU(j)          Overall heat transfer coefficient for cold stream(j) and hot utility
      UCU(i)          Overall heat transfer coefficient for hot stream(i) and cold utility

//upper bound for logical constraints//
      GAMMA_HX(i,j,k)    upper bound heat load for match (i.j)
      GAMMA_HU(j)        upper bound heat load for hot utility
      GAMMA_CU(i)        upper bound heat load for cold utility

      TAU_HX(i,j,k)      upper bound of temperature difference for heat exchanger
      TAU_HUL(j)         upper bound of temperature difference for left side of hot
                         utility exchanger
      TAU_HUR(j)         upper bound of temperature difference for right side of hot
                         utility exchanger
      TAU_CUL(i)         upper bound of temperature difference for left side pf cold
                         utility exchanger
      TAU_CUR(i)         upper bound of temperature difference for right side of cold
                         utility exchanger
      ;

//Old value of variables from original HEN//

   //Area//
      TABLE AREA_OLD(i,j,k) old area from original HEN of each period
                 K1          K2          K3          K4

      I1.J1    680.055     166.126     751.970
      I1.J2                            1290.040    132.794
      I1.J3                597.190
      I3.J1                356.123     1470.846
      I3.J4    324.859
      ;

      PARAMETERS
      AREAHU_OLD(j) old area of HU from original HEN
      /J3 256.025/
      AREACU_OLD(i) old area of CU from original HEN
      /I1 117.036,   I2 541.396,   I3  97.762/
      ;
```

```
//Temperature//
  TABLE TH_OLD(i,k) old hot temperature from original HEN
          K1         K2         K3         K4         KL

  I1    406.000    293.498    231.178    88.555     79.812
  I2    160.000    160.000    160.000    160.000    160.000
  I3    362.000    268.540    234.038    86.252     86.252
  ;

  TABLE TC_OLD(j,k) old cold temperature from original HEN
          K1         K2         K3         K4         KL

  J1    365.000    255.334    216.175    72.000     72.000
  J2    210.000    210.000    210.000    74.710     62.000
  J3    272.717    272.717    220.000    220.000    220.000
  J4    290.000    250.000    250.000    250.000    250.000
  ;

//Heat load//
  TABLE Q_OLD(i,j,k) old heat load from original HEN
              K1         K2         K3         K4

  I1.J1   23062.852   3528.959   10162.028
  I1.J2                   19075.844   1792.156
  I1.J3            9246.623
  I3.J1            4706.050   20158.011
  I3.J4   12748.000
  ;
  PARAMETERS
  QHU_OLD(j) old heat load of HU from original HEN
  /J3 17063.377/
  QCU_OLD(i) old heat load of CU from original HEN
  /I1  4061.538,   I2 23856.000,   I3  3580.739/
  ;

//Temperature Difference//
  TABLE DT_OLD(i,j,k) old temperature differene from original HEN
              K1         K2         K3         K4         KL

  I1.J1     41.000     38.165     15.003     16.555
  I1.J2                       21.178     13.844     17.812
  I1.J3            20.781     11.178
  I3.J1            13.206     17.863     14.252
  I3.J4     72.000     18.540
  ;
```

```
PARAMETERS
DTHUL_OLD(j) old left-side temperature differene of HU from original HEN
/J3 30.000/
DTHUR_OLD(j) old right-side temperature differene of HU from original HEN
/J3 126.283/
DTCUL_OLD(i) old left-side temperature differene of CU from original HEN
/I1  59.812,   I2 140.000,   I3  66.252/
DTCUR_OLD(i) old right-side temperature differene of CU from original HEN
/I1 45.000,   I2 25.000,   I3 45.000/
;

//Log Mean Temperature Difference//
TABLE LMTD_OLD(i,j,k) old log mean temperature from original HEN
            K1        K2        K3        K4

I1.J1     39.565    24.783    15.766
I1.J2                    17.252    15.745
I1.J3               15.484
I3.J1               15.417    15.989
I3.J4     39.242
;

PARAMETERS
LMTDHU_OLD(j) old log mean temperature of HU from original HEN
/J3 66.647/
LMTDCU_OLD(i) old log mean temperature of CU from original HEN
/I1 52.055,   I2 66.096,   I3 54.941/
;
//Overall heat transfer coefficient calculations//
U(i,j)  =  1/(1/HH(i)+1/HC(j));
UHU(j)  =  1/(1/HC(j)+1/HHU);
UCU(i)  =  1/(1/HH(i)+1/HCU);


VARIABLES
th(i,k)        temperature(C) of hot streams i at location k
tc(j,k)        temperature(C) of cold streams j at location k

q(i,j,k)       heat load(kW) of match(i.j) at stage k
qhu(j)         heat load(kW) of hot utility for cold stream j
qcu(i)         heat load(kW) of cold utility for hot stream i

dt(i,j,k)      temperature difference(C) for match(i.j) at location k
dthul(j)       temperature difference(C) for left side of hot utility exchanger
dthur(j)       temperature difference(C) for right side of hot utility exchanger
dtcul(i)       temperature difference(C) for left side of cold utility exchanger
```

```
dtcur(i)      temperature difference(C) for right side of cold utility exchanger

lmtd(i,j,k)    log mean temperature difference of match(i.j) at stage k
lmtdhu(j)      log mean temperature difference of hot utility exchanger for cold
               stream j
lmtdcu(i)      log mean temperature difference of cold utility exchanger for cold
               stream i

area(i,j,k)    area of heat exchanger of match(i.j) at stage k
areahu(j)      area of hot utility exchanger of cold stream j
areacu(i)      area of cold utility exchanger of hot stream i

area_add(i,j,k)  additional area of heat exchanger of match(i.j) at stage k
areahu_add(j)    additional area of hot utility exchanger of cold stream j
areacu_add(i)    additional area of cold utility exchanger of hot stream i

tot_area_add     total additional area needed
;

POSITIVE VARIABLES   q(i,j,k), qhu(j), qcu(i),
              dt(i,j,k), dthul(j), dthur(j), dtcul(i), dtcur(i),
              lmtd(i,j,k),lmtdhu(j),lmtdcu(i),
              area(i,j,k),areahu(j),areacu(i)
              area_add(i,j,k), areahu_add(j), areacu_add(i);

//variable bounding//

      th.up(i,k) = THIN(i);
      th.lo(i,k) = THOUT(i);
      tc.up(j,k) = TCOUT(j);
      tc.lo(j,k) = TCIN(j);

      q.up(i,j,k)$stage(k) = max(min(  FCPH(i)*(th.up(i,k)-
max(th.lo(i,k+1),tc.lo(j,k+1)+EMAT))  ,  FCPC(j)*(min(th.up(i,k)-
EMAT,tc.up(j,k))-tc.lo(j,k+1))  )  ,  0)$( (th.up(i,k)-tc.lo(j,k) >= EMAT) and
(th.up(i,k+1)-tc.lo(j,k+1) >= EMAT))        +0$( (th.up(i,k)-tc.lo(j,k) < EMAT) or
(th.up(i,k+1)-tc.lo(j,k+1) < EMAT)) ;

      qhu.up(j)          = max(  FCPC(j)*(min(THUIN(j)-EMAT,TCOUT(j))-
tc.lo(j,'K1'))  ,  0)$((THUIN(j)-TCOUT(j) >= EMAT) and (THUOUT(j)-
tc.lo(j,'K1') >= EMAT)) +0$( (THUIN(j)-TCOUT(j) < EMAT) or (THUOUT(j)-
tc.lo(j,'K1') < EMAT)) ;

      qcu.up(i)          = max(  FCPH(i)*(th.up(i,'KL')-
max(THOUT(i),TCUIN(i)+EMAT))  ,  0)$( (th.up(i,'KL')-TCUOUT(i) >= EMAT)
and (THOUT(i)-TCUIN(i) >= EMAT)) +0$( (th.up(i,'KL')-TCUOUT(i) < EMAT) or
(THOUT(i)-TCUIN(i) < EMAT)) ;
```

```
       dt.up(i,j,k) = max(EMAT,(th.up(i,k)-tc.lo(j,k)));
       dt.lo(i,j,k) = EMAT;
       dthul.up(j)  = max(EMAT,THUIN(j)-TCOUT(j));
       dthul.lo(j)  = EMAT;
       dthur.up(j)  = max(EMAT,THUOUT(j)-tc.lo(j,'K1'));
       dthur.lo(j)  = EMAT;
       dtcul.up(i)  = max(EMAT,th.up(i,'KL')-TCUOUT(i));
       dtcul.lo(i)  = EMAT;
       dtcur.up(i)  = max(EMAT,THOUT(i)-TCUIN(i));
       dtcur.lo(i)  = EMAT;

       lmtd.up(i,j,k)$stage(k)   =
(dt.up(i,j,k)*dt.up(i,j,k+1)*(dt.up(i,j,k)+dt.up(i,j,k+1))/2)**(1/3);
       lmtd.lo(i,j,k)$stage(k)   =
(dt.lo(i,j,k)*dt.lo(i,j,k+1)*(dt.lo(i,j,k)+dt.lo(i,j,k+1))/2)**(1/3);
       lmtdhu.up(j)              =
(dthur.up(j)*dthul.up(j)*(dthur.up(j)+dthul.up(j))/2)**(1/3);
       lmtdhu.lo(j)              = (dthur.lo(j)*dthul.lo(j)*(dthur.lo(j)+dthul.lo(j))/2)**(1/3);
       lmtdcu.up(i)              =
(dtcul.up(i)*dtcur.up(i)*(dtcul.up(i)+dtcur.up(i))/2)**(1/3);
       lmtdcu.lo(i)              = (dtcul.lo(i)*dtcur.lo(i)*(dtcul.lo(i)+dtcur.lo(i))/2)**(1/3);

       area.up(i,j,k)$stage(k) = q.up(i,j,k)/lmtd.lo(i,j,k)/U(i,j);
       areahu.up(j)            = qhu.up(j)/lmtdhu.lo(j)/UHU(j);
       areacu.up(i)            = qcu.up(i)/lmtdcu.lo(i)/UCU(i);

//Assign bounding to pamaters in logical constraint//
       GAMMA_HX(i,j,k) = q.up(i,j,k);
       GAMMA_HU(j)     = qhu.up(j);
       GAMMA_CU(i)     = qcu.up(i);

       TAU_HX(i,j,k)  =  -th.lo(i,k)+tc.up(j,k)+dt.lo(i,j,k);
       TAU_HUL(j)     =  -THUIN(j)+TCOUT(j)+dthul.lo(j);
       TAU_HUR(j)     =  -THUOUT(j)+tc.up(j,'K1')+dthur.lo(j);
       TAU_CUL(i)     =  -th.lo(i,'KL')+TCUOUT(i)+dtcul.lo(i);
       TAU_CUR(i)     =  -THOUT(i)+TCUIN(i)+dtcur.lo(i);

//Last stage forcing bound//
       q.fx(i,j,'KL')        =0;
       lmtd.fx(i,j,'KL')     =0;
       area.fx(i,j,'KL')     =0;
       area_add.fx(i,j,'KL') =0;

//To force no q, area change where there is no existing HX//
       q.fx(i,j,k)$(not HEN(i,j,k))         = 0;
       area_add.fx(i,j,k)$(not HEN(i,j,k))  = 0;
```

```
//initial value for variables//
      area.l(i,j,k)      = AREA_OLD(i,j,k);
      areahu.l(j)        = AREAHU_OLD(j);
      areacu.l(i)        = AREACU_OLD(i);

      area_add.l(i,j,k)   =  0;
      areahu_add.l(j)     =  0;
      areacu_add.l(i)     =  0;

      th.l(i,k)         = TH_OLD(i,k);
      tc.l(j,k)         = TC_OLD(j,k);

      q.l(i,j,k)        = Q_OLD(i,j,k);
      qhu.l(j)          = QHU_OLD(j);
      qcu.l(i)          = QCU_OLD(i);

      dt.l(i,j,k)       = DT_OLD(i,j,k);
      dthul.l(j)        = DTHUL_OLD(j);
      dthur.l(j)        = DTHUR_OLD(j);
      dtcul.l(i)        = DTCUL_OLD(i);
      dtcur.l(i)        = DTCUR_OLD(i);

      lmtd.l(i,j,k)      = LMTD_OLD(i,j,k);
      lmtdhu.l(j)        = LMTDHU_OLD(j);
      lmtdcu.l(i)        = LMTDCU_OLD(i);


EQUATIONS
      OVERALL_H(i)        overall energy balance of hot stream i
      OVERALL_C(j)        overall energy balance of cold stream j

      EBAL_H(i,k)        energy balance of hot stream i in stage k
      EBAL_C(j.k)        energy balance of cold stream j in stage k

      TIN_H(i)          assignment inlet temperature of hot stream i
      TIN_C(j)          assignment inlet temperature of cold stream j

      MONOT_H(i,k)        monotonic in temperature of hot stream i at location k
      MONOT_C(j.k)        monotonic in temperature of cold stream j at location k
      MONOT_HNOT(i,k)     monotonic in temperature of hot stream i at location k
                          if no HX
      MONOT_CNOT(j,k)     monotonic in temperature of cold stream j at location
                          k if no HX

      MONOT_HU(j)         monotonic in outlet temperature of cold stream j
     MONOT_CU(i)         monotonic in outlet temperature of hot stream i
     MONOT_HUNOT(j)  monotonic in outlet temperature of cold stream j if no HX
```

```
MONOT_CUNOT(i)   monotonic in outlet temperature of hot stream i if no HX

 EBAL_HU(j)        energy balance of hot utility of cold stream j
 EBAL_CU(i)        energy balance of cold utility of hot stream i

 DTFEAS_HXL(i,j,k)    temperature feasibility of heat exchanger of match(i.j)
                      at location k
 DTFEAS_HXR(i,j,k)    temperature feasibility of heat exchanger of match(i.j)
                      at location k+1
 DTFEAS_HUL(j)        temperature feasibility of hot utility of outlet cold
                      stream j
 DTFEAS_HUR(j)      temperature feasibility of hot utility of inlet cold stream j
 DTFEAS_CUL(i)      temperature feasibility of cold utility of inlet hot stream i
 DTFEAS_CUR(i)        temperature feasibility of cold utility of outlet hot
                      stream i

 LOGMTD(i,j,k)         log mean temperature of match(i.j) at stage k
 LOGMTDHU(j)           log mean temperature of hot utility exchanger
 LOGMTDCU(i)           log mean temperature of cold utility exchanger

 ADDIAREA_HX1(i,j,k)  additional area needed for heat exchanger
 ADDIAREA_HX2(i,j,k)  force ADDAREA_HX > 0
 ADDIAREA_HU1(j)      additional area needed for hot utility
 ADDIAREA_HU2(j)      force ADDAREA_HU > 0
 ADDIAREA_CU1(i)      additional area needed for cold utility
 ADDIAREA_CU2(i)      force ADDAREA_CU > 0

 A_HX(i,j,k)        area of heat exchanger of match(i.j) at stage k
 A_HU(j)            area of hot utility exchanger of cold stream j
 A_CU(i)            area of cold utility exchanger of hot stream i

 OBJ              objective function (total additional area)
 ;

OVERALL_H(i)               ..  (THIN(i) -THOUT(i))*FCPH(i)  =e=
                               sum((j,k),q(i,j,k))+qcu(i);
OVERALL_C(j)               ..  (TCOUT(j)-TCIN(j)) *FCPC(j)  =e=
                               sum((i,k),q(i,j,k))+qhu(j);

EBAL_H(i,k)$HEN_HOT(i,k)   ..  (th(i,k)-th(i,k+1))*FCPH(i)  =e=  sum(j,q(i,j,k));
                                                                 //do only exist HEN
EBAL_C(j,k)$HEN_COLD(j,k)  ..  (tc(j,k)-tc(j,k+1))*FCPC(j)  =e=  sum(i,q(i,j,k));

TIN_H(i)                ..  THIN(i)  =e=  th(i,'K1');
TIN_C(j)                ..  TCIN(j)  =e=  tc(j,'KL');
```

```
MONOT_H(i,k)$HEN_HOT(i,k)                    ..  th(i,k)  =g=  th(i,k+1);
                                                              //if HX --> monotonic
MONOT_C(j,k)$HEN_COLD(j,k)                    ..  tc(j,k)  =g=  tc(j,k+1);
MONOT_HNOT(i,k)$((not HEN_HOT(i,k)) and stage(k)) ..  th(i,k)  =e=  th(i,k+1);
                                                              //if no HX --> equal
MONOT_CNOT(j,k)$((not HEN_COLD(j,k)) and stage(k)) ..  tc(j,k)  =e=  tc(j,k+1);

MONOT_HU(j)$HU(j)      ..  TCOUT(j)  =g=  tc(j,'K1');       //if HX --> monotonic
MONOT_CU(i)$CU(i)      ..  th(i,'KL') =g=  THOUT(i);
MONOT_HUNOT(j)$(not HU(j)) ..  OUT(j)   =e=  tc(j,'K1'); _  //if no HX --> equal
MONOT_CUNOT(i)$(not CU(i))      ..  th(i,'KL') =e=  THOUT(i);

EBAL_HU(j)$HU(j)              ..  (TCOUT(j)   -  tc(j,'K1'))*FCPC(j)  =e=  qhu(j);
                                                              //do only exist HEN
EBAL_CU(i)$CU(i)              ..  (th(i,'KL')  -  THOUT(i))  *FCPH(i)  =e=  qcu(i);

DTFEAS_HXL(i,j,k)$HEN(i,j,k)    ..  dt(i,j,k)    =e=  th(i,k)   -tc(j,k)  ;
DTFEAS_HXR(i,j,k)$HEN(i,j,k)    ..  dt(i,j,k+1)  =e=  th(i,k+1) -tc(j,k+1);
DTFEAS_HUL(j)$HU(j)             ..  dthul(j)     =e=  THUIN(j)  -TCOUT(j)  ;
DTFEAS_HUR(j)$HU(j)             ..  dthur(j)     =e=  THUOUT(j) -tc(j,'K1');
                                                     //if exist HEN  and no TAU
DTFEAS_CUL(i)$CU(i)             ..  dtcul(i)     =e=  th(i,'KL')-TCUOUT(i) ;
DTFEAS_CUR(i)$CU(i)             ..  dtcur(i)     =e=  THOUT(i)  -TCUIN(i)  ;

LOGMTD(i,j,k)$HEN(i,j,k)        ..  lmtd(i,j,k)  =e=
                      (dt(i,j,k)*dt(i,j,k+1)*(dt(i,j,k)+dt(i,j,k+1))/2)**(1/3);
                                                              //only HEN
LOGMTDHU(j)$HU(j)               ..  lmtdhu(j)    =e=  (dthur(j) *dthul(j)   *(dthur(j)
                                    +dthul(j))  /2)**(1/3);
LOGMTDCU(i)$CU(i)               ..  lmtdcu(i)    =e=  (dtcul(i) *dtcur(i)   *(dtcul(i)
                                    +dtcur(i))  /2)**(1/3);

A_HX(i,j,k)$HEN(i,j,k) ..  area(i,j,k)  *lmtd(i,j,k)*U(i,j)   =e=  q(i,j,k); //only  HEN
A_HU(j)$HU(j)          ..  areahu(j)    *lmtdhu(j)  *UHU(j)   =e=  qhu(j);
A_CU(i)$CU(i)          ..  areacu(i)    *lmtdcu(i)  *UCU(i)   =e=  qcu(i);

ADDIAREA_HX1(i,j,k)$HEN(i,j,k)  ..  area_add(i,j,k)  =g= area(i,j,k)  -
                                    AREA_OLD(i,j,k);
ADDIAREA_HX2(i,j,k)$HEN(i,j,k)  ..  area_add(i,j,k)  =g= 0;
ADDIAREA_HU1(j)$HU(j) ..  areahu_add(j)   =g=   areahu(j) - AREAHU_OLD(j);
ADDIAREA_HU2(j)$HU(j) ..  areahu_add(j)   =g= 0;
ADDIAREA_CU1(i)$CU(i) ..  areacu_add(i)   =g= areacu(i) - AREACU_OLD(i);
ADDIAREA_CU2(i)$CU(i) ..  areacu_add(i)   =g= 0;
```

```
//least square
//OBJ                    ..  TAC  =e=
               sum((i,j,k)$(HEN(i,j,k)) , power(AREAHX(i,j,k)-AREAOLD(i,j,k),2) )
               +  sum(j$(HU(j)) , power(AREAHU(j)-AREAHUOLD(j),2) )
               +   sum(i$(CU(i)) ,  power(AREACU(i)-AREACUOLD(i),2) );

//penalty only increased area
OBJ                       ..  TOT_AREA_ADD  =e=
                              sum((i,j,k)$HEN(i,j,k),area_add(i,j,k))
                              + sum(j$HU(j),areahu_add(j))
                              + sum(i$CU(i),areacu_add(i));


model STAGEMODEL_SINGLEPERIOD /all/;

option iterlim=1e9;
option domlim=0;
option reslim=1e8;
STAGEMODEL_SINGLEPERIOD.optfile=0;

solve STAGEMODEL_SINGLEPERIOD using NLP minimizing
TOT_AREA_ADD;

display th.l,tc.l.
     q.l,qhu.l,qcu.l,
     area.l,areahu.l.areacu.l,
     dt.l,dthul.l,dthur.l,dtcul.l,dtcur.l,
     lmtd.l,lmtdhu.l,lmtdcu.l,
     tot_area_add.l,
     ;
```

Appendix C Source Code of Simultaneous MINLP Multiperiod Model

The following text is source code which is entered in GAMS for synthesizing multi-period HEN by simultaneous approach.

```
SETS
      I  hot streams                  /I1*I3/
      J  cold streams                 /J1*J4/
      K  index of stage or location   /K1*K4,KL/    //last number is no. of stages
      P  period of operaion           /P1*P3/

//dynamic sets//
      stage(k)    all stages
      ;
      stage(k) = yes;    stage('KL') = no;

SCALARS
//Cost//
      CA    per unit cost of heat exchanger area($ per unit)    /641.7/
      CF    fixed cost of heat exchanger($ per unit)            /8333.3/

      CHU   per unit cost of hot utility($ per kW)              /115.2/
      CCU   fixed cost of heat exchanger($ per kW)              /1.3/

      AF    annualisation factor                                /0.2/

//other//
      EMAT  exchanger minimum approach temperature(C)               /5/

      HHU   heat transfer coefficient of hot utility(kW|(m2.C))     /2/
      HCU   heat transfer coefficient of cold utility(kW|(m2.C))    /1/
      ;

//Hot Streams//
      TABLE   THIN(i,p)  inlet temperatures of hot stream i in period p(C)
                 P1        P2        P3
         I1      393       406       420
         I2      160       160       160
         I3      354       362       360
         ;
      TABLE   THOUT(i,p) outlet temperatures of hot stream i in period p(C)
                 P1        P2        P3
         I1      60        60        60
         I2      40        40        40
         I3      60        60        60
```

;

//Cold Streams//

TABLE TCIN(j,p) inlet temperatures of cold stream j in period p(C)

	P1	P2	P3
J1	72	72	72
J2	62	62	62
J3	220	220	220
J4	253	250	249

;

TABLE TCOUT(j,p) outlet temperatures of cold stream j in period p(kW|K)

	P1	P2	P3
J1	356	365	373
J2	210	210	210
J3	370	370	370
J4	284	290	286

;

//Heat Capacity Flowrates//

TABLE FCPH(i,p) heat capaity flowrates of hot stream i in period p(kW|K)

	P1	P2	P3
I1	201.6	205.0	208.5
I2	185.1	198.8	175.2
I3	137.4	136.4	134.1

;

TABLE FCPC(j,p) heat capaity flowrates of cold stream j in period p(kW|K)

	P1	P2	P3
J1	209.4	210.3	211.1
J2	141.6	141.0	140.5
J3	176.4	175.4	174.5
J4	294.4	318.7	271.2

;

//Utility//

TABLE THUIN(j,p) inlet temperature of hot utility for cold stream j in period p(C)

	P1	P2	P3
J1	400	400	400
J2	400	400	400
J3	400	400	400
J4	400	400	400

;

TABLE THUOUT(j,p) outlet temperature of hot utility for cold stream j in period p(C)

	P1	P2	P3
J1	399	399	399
J2	399	399	399
J3	399	399	399

```
        J4    399         399          399
        ;
   TABLE   TCUIN(i,p)  inlet temperature of cold utility for hot stream i in
                             period p(C)
              P1          P2           P3
        I1     15          15           15
        I2     15          15           15
        I3     15          15           15
        ;
   TABLE   TCUOUT(i,p) outlet temperature of cold utility for hot stream i in
                             period p(C)
              P1          P2           P3
        I1     20          20           20
        I2     20          20           20
        I3     20          20           20
        ;

PARAMETERS
//Heat transfer coefficient//
   HH(i)   heat capaity flowrates of hot streams(kW|(m2.C))
   /       I1   2
           I2   2
           I3   2
   /
   HC(j)   heat capaity flowrates of cold streams(kW|(m2.C))
   /       J1   1.5
           J2   1.5
           J3   2
           J4   2
   /

//Duration of each period//
   DOP(p) duration of each period
   /       P1    4
           P2    4
           P3    4
   /

//Total duration//
   TOP    total duration of all period

//overall heat transfer coefficient//
   U(i.j)         Overall heat transfer coefficient for match(i.j)
   UHU(j)         Overall heat transfer coefficient for cold stream(j) and hot utility
   UCU(i)         Overall heat transfer coefficient for hot stream(i) and cold utility

//upper bound for logical constraints//
```

```
GAMMA_HX(i,j,k,p)  upper bound heat load for match (i.j) for period p
GAMMA_HU(j,p)      upper bound heat load for hot utility for period p
GAMMA_CU(i,p)      upper bound heat load for cold utility for period p

TAU_HX(i,j,k,p)    upper bound of temperature difference for heat exchanger
                   for period p
TAU_HUL(j,p)       upper bound of temperature difference for left side of hot
                   utility exchanger for period p
TAU_HUR(j,p)       upper bound of temperature difference for right side of hot
                   utility exchanger for period p
TAU_CUL(i,p)       upper bound of temperature difference for left side pf cold
                   utility exchanger for period p
TAU_CUR(i,p)       upper bound of temperature difference for right side of cold
                   utility exchanger for period p
;

//Overall heat transfer coefficient calculations//
    U(i,j) = 1/(1/HH(i)+1/HC(j));
    UHU(j) = 1/(1/HC(j)+1/HHU);
    UCU(i) = 1/(1/HH(i)+1/HCU);

//TOP calculation//
    TOP   = sum(p,DOP(p));

VARIABLES
    th(i,k,p)      temperature(C) of hot streams i at location k for period p
    tc(j,k,p)      temperature(C) of cold streams j at location k for period p

    q(i,j,k,p)     heat load(kW) of match(i.j) at stage k for period p
    qhu(j,p)       heat load(kW) of hot utility for cold stream j for period p
    qcu(i,p)       heat load(kW) of cold utility for hot stream i for period p

    z(i,j,k)       existence of match(i.j) at stage k
    zhu(j)         existence of hot utility for cold stream j
    zcu(i)         existence of cold utility for hot stream i

    dt(i,j,k,p)    temperature difference(C) for match(i.j) at location k for period p
    dthul(j,p)     temperature difference(C) for left side of hot utility exchanger
                   for period p
    dthur(j,p)     temperature difference(C) for right side of hot utility exchanger
                   for period p
    dtcul(i,p)     temperature difference(C) for left side of cold utility exchanger
                   for period p
    dtcur(i,p)     temperature difference(C) for right side of cold utility exchanger
                   for period p
```

```
lmtd(i,j,k,p)      log mean temperature difference of match(i.j) at stage k for
                   period p
lmtdhu(j,p)        log mean temperature difference of hot utility exchanger for
                   cold stream j for period p
lmtdcu(i,p)        log mean temperature difference of cold utility exchanger for
                   cold stream i for period p

area(i,j,k,p)      area of heat exchanger of match(i.j) at stage k for period p
areahu(j,p)        area of hot utility exchanger of cold stream j for period p
areacu(i,p)        area of cold utility exchanger of hot stream i for period p

area_max(i,j,k)    maximum area of heat exchanger of match(i.j) at stage k for
                   all period
areahu_max(j)      maximum area of hot utility exchanger of cold stream j for all
                   period
areacu_max(i)      maximum area of cold utility exchanger of hot stream i for all
                   period

cost_fix           fixed cost
cost_area          area cost
cost_utility       utility cost

TAC                total annualized cost
;

POSITIVE VARIABLES   q(i,j,k,p), qhu(j,p), qcu(i,p),
          dt(i,j,k,p), dthul(j,p), dthur(j,p), dtcul(i,p), dtcur(i,p),
          LMTD(i,j,k,p), LMTDHU(j,p), LMTDCU(i,p),
          AREAHX(i,j,k,p), AREAHU(j,p), AREACU(i,p)
          AREAHX_max(i,j,k), AREAHU_max(j), AREACU_max(i);

BINARY VARIABLES    z(i,j,k), zhu(j), zcu(i)   ;

//variable bounding//

    th.up(i,k,p) = THIN(i,p);
    th.lo(i,k,p) = THOUT(i,p);
    tc.up(j,k,p) = TCOUT(j,p);
    tc.lo(j,k,p) = TCIN(j,p);

    q.up(i,j,k,p)$stage(k) = max(min(  FCPH(i,p)*(th.up(i,k,p)-
max(th.lo(i,k+1,p),tc.lo(j,k+1,p)+EMAT))  ,   FCPC(j,p)*(min(th.up(i,k,p)-
EMAT,tc.up(j,k,p))-tc.lo(j,k+1,p))  )  ,  0)$( (th.up(i,k,p)-tc.lo(j,k,p) >= EMAT) and
(th.up(i,k+1,p)-tc.lo(j,k+1,p) >= EMAT))        +0$( (th.up(i,k,p)-tc.lo(j,k,p) < EMAT)
or (th.up(i,k+1,p)-tc.lo(j,k+1,p) < EMAT)) ;
```

```
qhu.up(j,p)          = max( FCPC(j,p)*(min(THUIN(j,p)-EMAT,TCOUT(j,p))-
tc.lo(j,'K1',p)) , 0)$((THUIN(j,p)-TCOUT(j,p) >= EMAT) and (THUOUT(j,p)-
tc.lo(j,'K1',p) >= EMAT)) +0$( (THUIN(j,p)-TCOUT(j,p) < EMAT) or
(THUOUT(j,p)-tc.lo(j,'K1',p) < EMAT)) ;

qcu.up(i,p)          = max( FCPH(i,p)*(th.up(i,'KL',p)-
max(THOUT(i,p),TCUIN(i,p)+EMAT)) , 0)$( (th.up(i,'KL',p)-TCUOUT(i,p) >=
EMAT) and (THOUT(i,p)-TCUIN(i,p) >= EMAT)) +0$( (th.up(i,'KL',p)-
TCUOUT(i,p) < EMAT) or (THOUT(i,p)-TCUIN(i,p) < EMAT)) ;

z.up(i,j,k)$stage(k) = 0$( sum(p, q.up(i,j,k,p)) =0)  +  1$( sum(p, q.up(i,j,k,p))  ne 0);
    zhu.up(j)    = 0$( sum(p, qhu.up(j,p)  ) =0)   +  1$( sum(p, qhu.up(j,p)  )  ne 0);
    zcu.up(i)    = 0$( sum(p, qcu.up(i,p)  ) =0)   +  1$( sum(p, qcu.up(i,p)  )  ne 0);

    dt.up(i,j,k,p)  =  max(EMAT,(th.up(i,k,p)-tc.lo(j,k,p)));
    dt.lo(i,j,k,p)  =  EMAT;
    dthul.up(j,p)   =  max(EMAT,THUIN(j,p)-TCOUT(j,p));
    dthul.lo(j,p)   =  EMAT;
    dthur.up(j,p)   =  max(EMAT,THUOUT(j,p)-tc.lo(j,'K1',p));
    dthur.lo(j,p)   =  EMAT;
    dtcul.up(i,p)   =  max(EMAT,th.up(i,'KL',p)-TCUOUT(i,p));
    dtcul.lo(i,p)   =  EMAT;
    dtcur.up(i,p)   =  max(EMAT,THOUT(i,p)-TCUIN(i,p));
    dtcur.lo(i,p)   =  EMAT;

    lmtd.up(i,j,k,p)$stage(k)  =
          (dt.up(i,j,k,p)*dt.up(i,j,k+1,p)*(dt.up(i,j,k,p)+dt.up(i,j,k-1,p))/2)**(1/3);
    lmtd.lo(i,j,k,p)$stage(k)  =
            (dt.lo(i,j,k,p)*dt.lo(i,j,k+1,p)*(dt.lo(i,j,k,p)+dt.lo(i,j,k-1,p))/2)**(1/3);
    lmtdhu.up(j,p)             =
              (dthur.up(j,p)*dthul.up(j,p)*(dthur.up(j,p)+dthul.up(j,p))/2)**(1/3);
    lmtdhu.lo(j,p)             =
                (dthur.lo(j,p)*dthul.lo(j,p)*(dthur.lo(j,p)+dthul.lo(j,p))/2)**(1/3);
    lmtdcu.up(i,p)             =
              (dtcul.up(i,p)*dtcur.up(i,p)*(dtcul.up(i,p)+dtcur.up(i,p))/2)**(1/3);
    lmtdcu.lo(i,p)             =
                (dtcul.lo(i,p)*dtcur.lo(i,p)*(dtcul.lo(i,p)+dtcur.lo(i,p))/2)**(1/3);

    area.up(i,j,k,p)$stage(k)= q.up(i,j,k,p)/LMTD.lo(i,j,k,p)/U(i,j);
    areahu.up(j,p)            = qhu.up(j,p)/LMTDHU.lo(j,p)/UHU(j);
    areacu.up(i,p)            = qcu.up(i,p)/LMTDCU.lo(i,p)/UCU(i);

//Assign bounding to pamaters in logical constraint//
    GAMMA_HX(i,j,k,p)   = q.up(i,j,k,p);
    GAMMA_HU(j,p)       = qhu.up(j,p);
    GAMMA_CU(i,p)       = qcu.up(i,p);
```

```
TAU_HX(i,j,k,p)  = -th.lo(i,k,p)+tc.up(j,k,p)+dt.lo(i,j,k,p);
TAU_HUL(j,p)     = -THUIN(j,p)+TCOUT(j,p)+dthul.lo(j,p);
TAU_HUR(j,p)     = -THUOUT(j,p)+tc.up(j,'K1',p)+dthur.lo(j,p);
TAU_CUL(i,p)     = -th.lo(i,'KL',p)+TCUOUT(i,p)+dtcul.lo(i,p);
TAU_CUR(i,p)     = -THOUT(i,p)+TCUIN(i,p)+dtcur.lo(i,p);

//Last stage forcing bound//
q.fx(i,j,'KL',p)      =0;
z.fx(i,j,'KL')        =0;
lmtd.fx(i,j,'KL',p)   =0;
area.fx(i,j,'KL',p)   =0;


EQUATIONS
OVERALL_H(i,p)       overall energy balance of hot stream i for period p
OVERALL_C(j,p)       overall energy balance of cold stream j for period p

EBAL_H(i,k,p)        energy balance of hot stream i in stage k for period p
EBAL_C(j,k,p)        energy balance of cold stream j in stage k for period p

TIN_H(i,p)           assignment inlet temperature of hot stream i for period p
TIN_C(j,p)           assignment inlet temperature of cold stream j for period p

MONOT_H(i,k,p)       monotonic in temperature of hot stream i at location k
                     for period p
MONOT_C(j,k,p)       monotonic in temperature of cold stream j at location k
                     for period p
MONOT_HU(j,p)        monotonic in outlet temperature of cold stream j for
                     period p
MONOT_CU(i,p)        monotonic in outlet temperature of hot stream i for
                     period p

EBAL_HU(j,p)         energy balance of hot utility of cold stream j for period p
EBAL_CU(i,p)         energy balance of cold utility of hot stream i for period p

LOGIC_HX(i,j,k,p)    logical constraint to define z of match(i.j) at stage k for
                     period p
LOGIC_HU(j,p)        logical constraint to define zhu of cold stream j at stage
                     k for period p
LOGIC_CU(i,p)        logical constraint to define zcu of hot stream i at stage k
                     for period p

DTFEAS_HXL(i,j,k,p)  temperature feasibility of heat exchanger of match(i.j)
                     at location k for period p
DTFEAS_HXR(i,j,k,p)  temperature feasibility of heat exchanger of match(i.j)
                     at location k+1 for period p
```

```
DTFEAS_HUL(j,p)      temperature feasibility of hot utility of outlet cold
                     stream j for period p
DTFEAS_HUR(j,p)      temperature feasibility of hot utility of inlet cold
                     stream j for period p
DTFEAS_CUL(i,p)      temperature feasibility of cold utility of inlet hot
                     stream i for period p
DTFEAS_CUR(i,p)      temperature feasibility of cold utility of outlet hot
                     stream i for period p

LOGMTD(i,j,k,p)   log mean temperature of match(i.j) at stage k for period p
LOGMTDHU(j,p)   log mean temperature of hot utility exchanger for period p
LOGMTDCU(i,p)   log mean temperature of cold utility exchanger for period p

A_HX(i,j,k,p)      area of heat exchanger of match(i.j) at stage k for period p
A_HU(j,p)          area of hot utility exchanger of cold stream j for period p
A_CU(i,p)          area of cold utility exchanger of hot stream i for period p

AMAX_HX(i,j,k,p)     maximum area of heat exchanger of match(i.j) at
                     stage k for all period
AMAX_HU(j,p)         maximum area of hot utility exchanger of cold stream j
                     for all period
AMAX_CU(i,p)         maximum area of cold utility exchanger of hot stream i
                     for all period

FIXCOST              fixed cost
AREACOST             area cost
UTILITYCOST          utility cost

OBJ             objective function (total annualized cost)
;


OVERALL_H(i,p)          .. (THIN(i,p) -THOUT(i,p))*FCPH(i,p) =e=
                           sum((j,k),q(i,j,k,p))+qcu(i,p);
OVERALL_C(j,p)          .. (TCOUT(j,p)-TCIN(j,p)) *FCPC(j,p) =e=
                           sum((i,k),q(i,j,k,p))+qhu(j,p);

EBAL_H(i,k,p)$stage(k)    .. (th(i,k,p)-th(i,k+1,p))*FCPH(i,p) =e=
                             sum(j,q(i,j,k,p));
EBAL_C(j,k,p)$stage(k)    .. (tc(j,k,p)-tc(j,k+1,p))*FCPC(j,p) =e=
                             sum(i,q(i,j,k,p));

TIN_H(i,p)            .. THIN(i,p) =e= th(i,'K1',p);
TIN_C(j,p)            .. TCIN(j,p) =e= tc(j,'KL',p);

MONOT_H(i,k,p)$stage(k)    .. th(i,k,p) =g= th(i,k+1,p);
```

```
MONOT_C(j,k,p)$stage(k)     .. tc(j,k,p) =g= tc(j,k+1,p);
MONOT_HU(j,p)               .. TCOUT(j,p) =g= tc(j,'K1',p);
MONOT_CU(i,p)               .. THOUT(i,p) =l= th(i,'KL',p);

EBAL_HU(j,p)        .. (TCOUT(j,p)  - tc(j,'K1',p))*FCPC(j,p) =e= qhu(j,p);
EBAL_CU(i,p)        .. (th(i,'KL',p) - THOUT(i,p))*FCPH(i,p)  =e= qcu(i,p);

LOGIC_HX(i,j,k,p)$stage(k)  .. q(i,j,k,p) - GAMMA_HX(i,j,k,p)*z(i,j,k) =l= 0;
LOGIC_HU(j,p)               .. qhu(j,p)  - GAMMA_HU(j,p)  *zhu(j)   =l= 0;
LOGIC_CU(i,p)               .. qcu(i,p)  - GAMMA_CU(i,p)  *zcu(i)   =l= 0;

DTFEAS_HXL(i,j,k,p)$stage(k) .. dt(i,j,k,p)   =l= th(i,k,p)  -tc(j,k,p)
                                 +TAU_HX(i,j,k,p)  *(1-z(i,j,k));
DTFEAS_HXR(i,j,k,p)$stage(k) .. dt(i,j,k+1,p) =l= th(i,k+1,p) -tc(j,k+1,p)
                                 +TAU_HX(i,j,k+1,p)*(1-z(i,j,k));
DTFEAS_HUL(j,p)              .. dthul(j,p)   =l= THUIN(j,p) -TCOUT(j,p)
                                +TAU_HUL(j,p)    *(1-zhu(j));
DTFEAS_HUR(j,p)              .. dthur(j,p)   =l= THUOUT(j,p)
                                - tc(j,'K1',p)+TAU_HUR(j,p)   *(1-zhu(j));
DTFEAS_CUL(i,p)              .. dtcul(i,p)   =l= th(i,'KL',p)-TCUOUT(i,p)
                                +TAU_CUL(i,p)    *(1-zcu(i));
DTFEAS_CUR(i,p)              .. dtcur(i,p)   =l= THOUT(i,p) -TCUIN(i,p)
                                +TAU_CUR(i,p)    *(1-zcu(i));


LOGMTD(i,j,k,p)$stage(k)     .. lmtd(i,j,k,p) =e=
                        (dt(i,j,k,p)*dt(i,j,k+1,p)*(dt(i,j,k,p)+dt(i,j,k+1,p))/2)**(1/3);
LOGMTDHU(j,p)                .. lmtdhu(j,p)   =e= (dthur(j,p) *dthul(j,p)  *(dthur(j,p)
                                +dthul(j,p))  /2)**(1/3);
LOGMTDCU(i,p)                .. lmtdcu(i,p)   =e= (dtcul(i,p) *dtcur(i,p)  *(dtcul(i,p)
                                +dtcur(i,p))  /2)**(1/3);

A_HX(i,j,k,p)$stage(k)    .. area(i,j,k,p) *lmtd(i,j,k,p)*U(i,j)  =e= q(i,j,k,p);
A_HU(j,p)                 .. areahu(j,p)   *lmtdhu(j,p) *UHU(j)  =e= qhu(j,p);
A_CU(i,p)                 .. areacu(i,p)   *lmtdcu(i,p) *UCU(i)  =e= qcu(i,p);

AMAX_HX(i,j,k,p)          .. area_max(i,j,k)   =g= area(i,j,k,p);
AMAX_HU(j,p)              .. areahu_max(j)     =g= areahu(j,p);
AMAX_CU(i,p)              .. areacu_max(i)     =g= areacu(i,p);

FIXCOST                 .. cost_fix     =e= AF*(sum((i,j,k)$stage(k),CF*z(i,j,k))
                          + sum(j,CF*zhu(j)) + sum(i,CF*zcu(i)));
AREACOST                  .. cost_area    =e=
                            AF*(sum((i,j,k)$stage(k),CA*area_max(i,j,k)))
                            + AF*(sum(j,CA*areahu_max(j)))
                            +  AF*(sum(i,CA*areacu_max(i)));
```

```
UTILITYCOST              ..  cost_utility  =e=
                             sum(p,DOP(p)/TOP*(sum(j,CHU*qhu(j,p))))
                             +  sum(p,DOP(p)/TOP*(sum(i,CCU*qcu(i,p))));

OBJ                  ..  TAC  =e=  cost_fix + cost_area + cost_utility;


model STAGEMODEL_SINGLEPERIOD /all/;

option iterlim=1e9;
option domlim=0;
option reslim=1e8;
STAGEMODEL_SINGLEPERIOD.optfile=0;

solve STAGEMODEL_SINGLEPERIOD using MINLP minimizing TAC;

display th.l,tc.l,
      z.l,zhu.l,zcu.l,
      q.l,qhu.l,qcu.l,
      area.l,areahu.l,areacu.l,
      area_max.l,areahu_max.l,areacu_max.l,
      dt.l,dthur.l,dthul.l,dtcul.l,dtcur.l,
      lmtd.l, lmtdhu.l, lmtdcu.l
      cost_fix.l, cost_area.l, cost_utility.l, TAC.l
      ;
```

CURRICULUM VITAE

Name: Ms. Parawinee Tangnanthanakan

Date of Birth: June 23, 1988

Nationality: Thai

University Education:

2007–2011 Bachelor Degree of Chemical Engineering, Faculty of Engineering, Mahidol University, Nakhon Pathom, Thailand

Proceedings:

1. Tangnanthanakan, P.; Siemanond, K. (2014, April 22) Sequential and simultaneous approaches for multiperiod heat exchanger network synthesis. Proceedings of The 5th Research Symposium on Petrochemical, and Materials Technology and The 20th PPC Symposium on Petroleum, Petrochemicals, and Polymers, Bangkok, Thailand.
2. Tangnanthanakan, P.; Siemanond, K. (2014, August 17-21) Comparison of sequential and simultaneous approaches for multiperiod heat exchanger network synthesis and application for crude preheat train. Proceedings of the 17th Conference on Process Integration, Modelling and Optimisation for Energy Saving and Pollution Reduction, Prague, Czech Republic.