REFERENCES

B. Carnahan, H.A Luther, and J.O. Wilkes, Applied Numerical Methods. John Wiley and Sons, New York, 1969.

Stephen M. Pizer, and Victor L. Wallace, To Compute Numerically, Concepts and strategies. Little, Brown & Company (Canada) Limited, Boston 1983.

Sanjay Kumar, Gas Production Engineering (Volume 4). Gulf Publishing Company, Houston, Texas, 1987.

James o. Wilkes, Macroscopic Fluid Mechanics for Mechanical Engineers. Department of Chemical Engineering, University of Michigan, Ann Arbor, Michigan, 1995.

Robert W. Fox, and Alan T. Mcdonald, Introduction to Fluid Mechanics (Fourth Edition, SI Version). Purdue University, John Wiley & Son, Inc. New York, 1994.

Richard S. H. Mah, Chemical Process Structures and Information Flows. Northwestern University, Butterworth-Heinemann, Stoneham, 1990.

Stoner, M. A., Steady State Analysis of Gas Production, Transmission and Distribution System. paper SPE 2554 presented at the SPE 44th Ann. Fall Meet., Denver, Colorado, 1969.

Stoner, M. A. Sensitivity Analysis Applied to a Steady State Model of Natural Gas Transportation Systems. Soc. Pet. Eng. J., 115-125, 1972.

R. E. Carlile, and B. E. Gillett, FORTRAN and Computer Mathematics for the Engineer and Scientist. The Petroleum Publishing Company, Tulsa, Oklahoma, 1973.

Microsoft Corporation, Microsoft FORTRAN Power Station. Professional Development System Version 1.0 for MS-Dos and Windows Operating Systems, Document No. DB38033-0293, 1993.

Chi U. Ikoku, Natural Gas Production Engineering. The Pennsylvania State University, John Wiley & Sons, New York, 1984.

B. Coulbeck & E. Evans, Pipeline Systems. Fluid Mechanics Mechanics and Its Applications, Kluwer Academic Publishers, London, 1992.

Gordon J. Van Wylen & Richard E. Sonntag, Fundamentals of Classical Thermodynamics. SI Version, John Wiley & Sons, New York, 1976.

APPENDIX A

FANNING FRICTION FACTOR

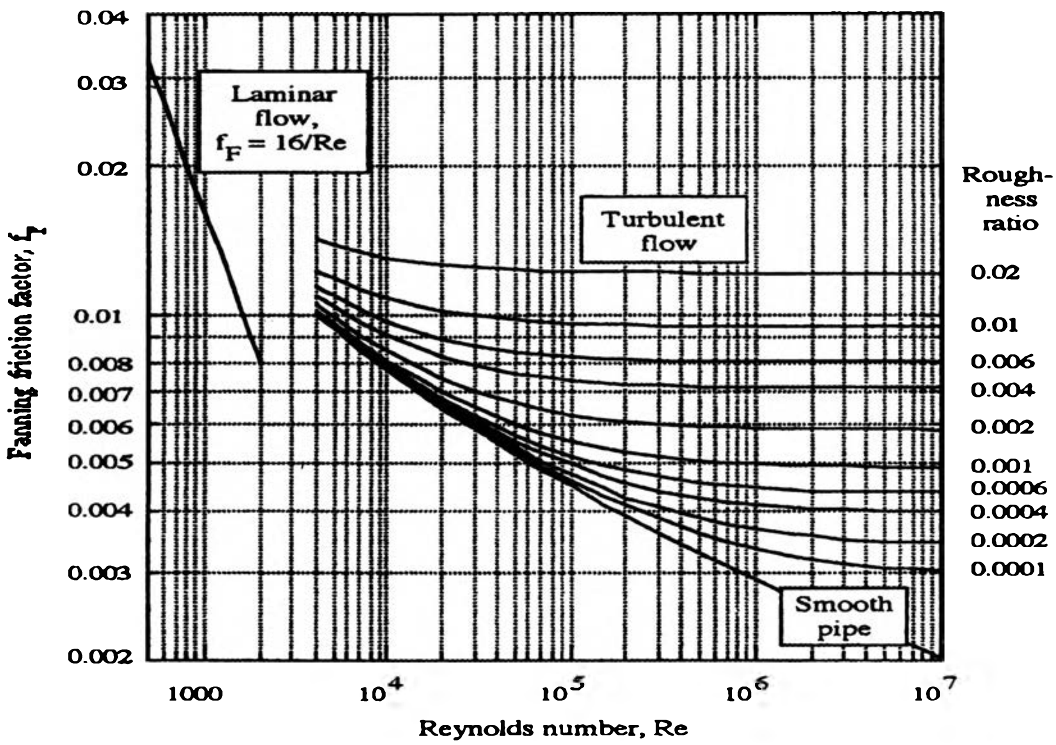

Fig. A.1 Fanning friction factor for flow in pipelines.

APPENDIX B
BANDWIDTH SETTING

The computation of the half bandwidth of the coefficient matrix (the maximum difference between adjacent nodal numbers) from the nodal connections is specified to identify the non-zero coefficients in the banded matrix in order to accelerate solution of the simultaneous linear equations generated at each new iteration of the Newton-Raphson method. Mark off the lowest and highest column subscripts within the banded matrix as follows:

*	*	*	*	*	*	0	0	0	0	0
*	*	*	*	*	*	*	0	0	0	0
*	*	*	*	*	*	*	*	0	0	0
*	*	*	*	*	*	*	*	*	0	0
*	*	*	*	*	*	*	*	*	*	0
*	*	*	*	*	*	*	*	*	*	*
0	*	*	*	*	*	*	*	*	*	*
0	0	*	*	*	*	*	*	*	*	*
0	0	0	*	*	*	*	*	*	*	*
0	0	0	0	*	*	*	*	*	*	*
0	0	0	0	0	*	*	*	*	*	*

|←——— Bandwidth ———→|

Note:

There may be zeros within the band, but outside the band all the elements are zero.

1. Consider node i (row subscript) and node j (column subscript) which are joined by a pipeline for which $i < j$.

2. Evaluate the absolute value of (i - j) respectively for each non-zero element C_{ij}, and compare with each other to find out the maximum difference between adjacent nodes that are connected by a pipeline segment or item of equipment. The result is the half bandwidth of the coefficient matrix.

3. To illustrate how to determine the bandwidth as implemented above in items 1 and 2 respectively, a simple hypothetical network is given in Fig. B.1

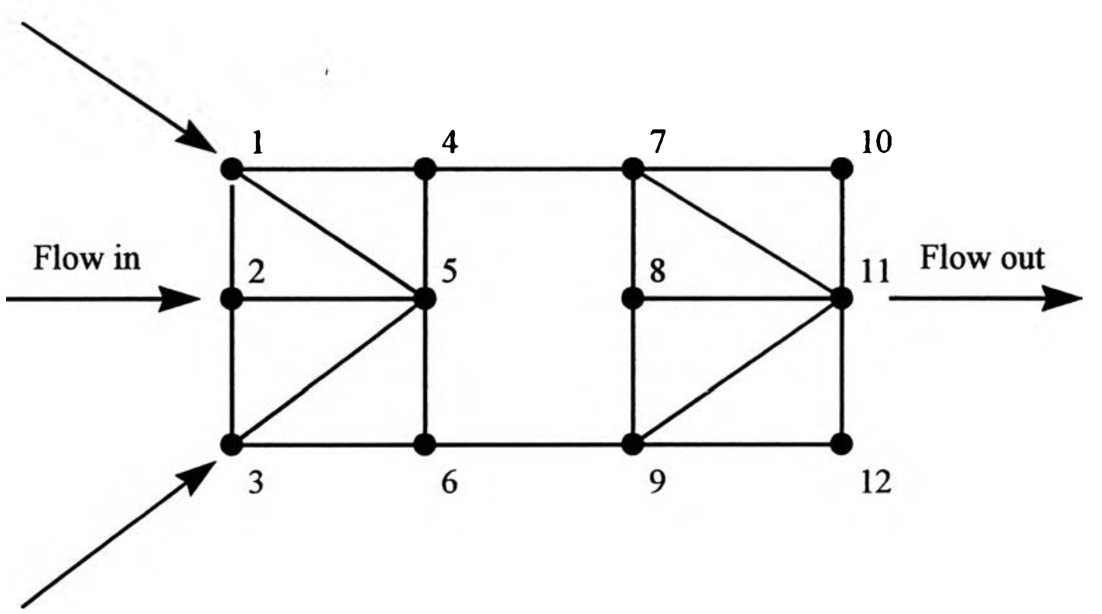

Fig. B.1 Topological descriptions of a network whose bandwidth is to be computed.

The network, shown in Fig. B.1, consists of 12 nodes, and 20 nodal connections.

The topological representation as shown in Fig. B.1 is used to determine the bandwidth as shown in Table B.1

Table B.1 Determination of bandwidth

I	J	ABS(I - J)
1	2	1
1	4	3
1	5	4*
2	3	1
2	5	3
3	5	2
3	6	3
4	5	1
4	7	3
5	6	1
6	9	3
7	8	1
7	10	3
7	11	4*
8	9	1
8	11	3
9	11	2
9	12	3
10	11	1
11	12	2

* The maximum difference in adjacent nodal numbers is 4, which is therefore the half bandwidth for the network shown in Fig. B.1.

APPENDIX C
GAS CODE

```
C     A GENERAL FLUID NETWORK SIMULATOR
C     PART I FOR A SINGLE GAS PHASE AT STEADY-STATE
C     BY PATIKOM SAELEE - GP961016
C     PETROCHEMICAL TECHNOLOGY PROGRAM
C     PETROLEUM AND PETROCHEMICAL COLLEGE
C     CHULALONGKORN UNIVERSITY
C     This program analyses n-nodes networks of single gas at steady state
C     where nodes may be connected by pipeline segments or compressors.
C     The simultaneous nonlinear equations generated from nodal material
C     balance at every node i as functions of the unknown nodal pressures in
C     the whole network are solved by Newton-Raphson iterative technique.
C     If C(I , J) is 1, the node i and j are connected by a pipeline segment of
C     diameter, D(I , J) and length, L(I , J). If T(I) is 1, the pressure at node i,
C     P(I) is fixed. Otherwise, P(I) assumes successive estimates of the
C     pressure at node i. DT is the density of gas assumed for all pipeline
C     segments in gas law equation and represented as below:
C                   DT   =    P(I)*MW
C                            -------------------
C                            ZAVG*RG*TG
C     Here:
C           P(I) = Pressure at node i, (absolute pressure)
C           MW = Molecular weight
C           ZAVG = Average gas compressibility factor
C           RG = Universal gas constant
```

```
C         TG = gas temperature assumed as isothermal flow
C         (If not, the average temperature is generally satisfactory)
C     The associated coefficients (*) from the Newton-Raphson method will
C     form a banded matrix, F(I , J) as below:
C
C       | *  *  *  *  *  0  0  0  0 |   | ? |     | ! |
C       | *  *  *  *  *  *  0  0  0 |   | ? |     | ! |
C       | *  *  *  *  *  *  *  0  0 |   | ? |     | ! |
C       | *  *  *  *  *  *  *  *  0 |   | ? |     | ! |
C       | *  *  *  *  *  *  *  *  * |   | ? |  =  | ! |
C       | 0  *  *  *  *  *  *  *  * |   | ? |     | ! |
C       | 0  0  *  *  *  *  *  *  * |   | ? |     | ! |
C       | 0  0  0  *  *  *  *  *  * |   | ? |     | ! |
C       | 0  0  0  0  *  *  *  *  * |   | ? |     | ! |
C
C     A special Gaussian elimination method for banded systems is
C     implemented by the normalization and reduction scheme with partial
C     pivot strategy to solve the simultaneous linear equations on the left
C     hand side of coefficient matrix, F(I , J). ITER is the iteration counter.
C     Iteration from Newton-Raphson method is stopped when ITER exceeds
C     ITMAX or all nodal pressures changes are lower than some criterion
C     value. QSC(I , J) is the flow rate at standard conditions between node i
C     and node j. QSC(I , J) is a positive value for fluid flow from node i to
C     node j. Otherwise, reverse direction. Any others  are described in the
C     program as C (comment).
C     The output in the program consists of nodal pressures at each node i in
C     the whole network and internodal flow rates in all pipeline segments.
```

```
C    Nomenclature                                                    Units
C    SF     Stability Factor                                         none
C     C     Nodal connection matrix                                  none
C    C(I , J) : 0 = No nodal connection between nodes i and j
C    C(I , J) : 1 = Pipeline connection between nodes i and j
C    C(I , J) : 3 = Compressor compress from node i to node j
C    C(J , I) : 3 = Compressor that compresses from node j to node i
C    CP     Average specific heat capacity at constant pressure     (BTU)
C           (1 BTU = 778.2 ft*lbf)                                -----------------
C                                                                 (lbm*Rankine)
C    CV     Average specific heat capacity at constant volume       (BTU)
C           (1 BTU = 778.2 ft*lbf)                                -------------------
C                                                                 (lbm*Rankine)
C    D      Pipeline diameters matrix                                inch
C    DP     The current vector of pressure changes                   psia
C    DPP    The previous vector of pressure changes                  psia
C    E      Pipeline roughness matrix                                ft
C    FF     Fanning friction factor matrix                           none
C    GC     Gravitational acceleration                               ft
C           (1 lbf = 32.2 lbm*ft/(sec)**2)                           --------
C                                                                    (sec)**2
C    ITMAX        Maximum number of iterations                       none
C    L      Pipeline lengths matrix                                  ft
C    MW     Molecular weight (single component)                      lbm
C    N      Number of nodes                                          none
C    NBAND        Half bandwidth of coefficient matrix               none
C    NPC    Number of compressors needed                             none
C    NC     Number of nodal connections                              none
```

```
C     NDL   Number of pipeline connections                          none
C     NT    Number of node-types                                    none
C     NV    Number of nodes with specified injection                none
C           or withdrawal rates
C     P     Nodal pressures [absolute pressure]                     psia
C     PC    Compression power matrix                                hp
C     QSC   Flow rates matrix (at standard conditions)              MMscfd
C           (unit : million standard cubic ft per day)
C     RG    Universal gas constant                                  (ft*lbf)
C           [(1545.3 (ft*lbf)/(lbm*Rankine)]                    ----------------
C                                                               (lbm*Rankine)
C     T     Type of node                                            none
C           0 : Pressure not specified
C           1 : Pressure specified
C           2 : Injection specified
C           3 : Terminal node with specified injection  or withdrawal rate
C     TG    Gas temperature (isothermal flow)                       Fahrenheit
C     V     Node at which there is a specified injection or withdrawal rate
C           V(I) : positive value = injection  rate
C           V(I) : negative value = withdrawal rate
C     VT    Average gas viscosity                                   Centipoise
C     Z     Nodal elevations                                        ft
C     ZAVG        Average gas compressibility factor                none
C     Type declaration variables
      REAL*8 ALPHA(35,35), AREA(35,35), D(35,35), E(35,35),
   +  EPS(35,35), F(36,36), FF(35,35), L(35,35), LAMDA(35,35),
   +  PC(35,35), PHI(35,35), QSC(35,35),
   +  SF(35), DP(35), DPP(35), FRIJ(1225), P(35), QIJ(1225),
```

```
   +  TINLET(35), V(35), Z(35),
   +  SIGMA(35),
   +  AL, CALPHA, CAREA, CEPS, CLAMDA, CP, CPHI, CTG, CTSC,
   +  CV, DT, DTSC, DTSCZE, DTSCZN, EP, EPI, EPJ, IGEN, K, KK, LA,
   +  MW, PSC, QADD, RG, RIJ, RIJK, RK1, TG, TP, TSC, W, ZAVG,
   +  ZSC, ZZ
      INTEGER*4 C(35,35), II(35), JJ(35),  JLOW(35),  JHIGH(35), T(35),
   +  COUNT, I, ITER, ITMAX, J,  M, TM
C     Identify input file
      OPEN (5, FILE='GAS.DAT')
C     Identify output file
      OPEN (6, FILE='GAS.OUT')
C     Read input data of network and gas property
      READ (5,100)  N, MW, RG, TG, ZAVG, VT, CP, CV, PSC, TSC,
   +  ZSC, NPC, NC, NDL,  NT, NV, ITMAX
C     Print input data of network and gas property
      WRITE (6,300) N, MW, RG, TG, ZAVG, VT, CP, CV, PSC, TSC,
   +  ZSC
      WRITE (6,310) NPC, NC, NDL,  NT, NV, ITMAX
      WRITE (6,500)
C     All parameters are initial as zero
      DO  4  I = 1, 35
            P(I) = 0.
            T(I) = 0
            V(I) = 0.
            Z(I) = 0.
            DP(I) = 0.
            DPP(I) = 0.
```

```
            TINLET(I) = 0.
      DO 5  J = 1, 35
            C(I , J) = 0
            D(I , J) = 0.
            E(I , J) = 0.
            L(I , J) = 0.
            PC(I , J) = 0.
            FF(I , J) = 0.
            AREA(I , J) = 0.
            ALPHA(I , J) = 0.
            EPS(I , J) = 0.
            LAMDA(I , J) = 0.
            PHI(I , J) = 0.
  5   CONTINUE
  4   CONTINUE
C     Read inlet temperature and compression power of compressor (if any)
      READ (5,200)
      DO 6  COUNT = 1, NPC
      READ (5,*)  I,  J,  TINLET(I),  PC(I , J)
  6   CONTINUE
C     Read nonzero nodal connection matrix
      READ (5,200)
      DO 7  COUNT = 1, NC
      READ (5,*)  I,  J,  C(I , J)
  7   CONTINUE
C     Read pipeline diameters, lengths, initial Fanning friction factors
C     and pipeline roughnesses matrix respectively joining node i and node j
      READ (5,200)
```

```
      DO 8 COUNT = 1, NDL
      READ (5,*) I, J, D(I , J), L(I , J), FF(I , J), E(I , J)
    8 CONTINUE
C     Read nodal estimated and specified pressures
      READ (5,200)
      READ(5,*) (P(I), I = 1, N)
C     Read type of node
      READ (5,200)
      DO 9 COUNT = 1, NT
      READ (5,*) I, T(I)
    9 CONTINUE
C     Read nodal injection or withdrawal rates
      READ (5,200)
      DO 10 COUNT = 1, NV
      READ (5,*) I, V(I)
   10 CONTINUE
C     Read nodal elevations
      READ (5,200)
      READ (5,*) (Z(I), I = 1, N)
C     Compute constant conversion unit from the flow rate equations:
C     The flow rate of nodal connection in all pipeline segments:
C     Inclined flow:
C     QSC(J , I) = +/- LAMDA(J , I) * SQRT             +/-W
C                      --------------------     ---------------------------------
C                               DTSC             ALPHA(J , I)*(PHI(J , I)-1)
C     Here:
C     W = P(J)**2-PHI(J , I)*P(I)**2
```

```
C      DTSC =      PSC*MW                                  (given as DTSCZN)
C                 ------------------
C                 ZSC**RG*TSC
C      LAMDA(J , I) =    (MW) * (PI*D(J , I)**2)           (given as CLAMDA)
C                       ------------------------------
C                         (ZAVG*RG*TG)* 4
C      ALPHA(J , I) =      [2*FF(J , I)*L(J , I)]          (given as CALPHA)
C                        ----------------------------
C                        [GC*D(J , I)*(Z(I)-Z(J))]
C      PHI(J , I) = EXP[2*MW*GC*(Z(I)-Z(J))]               (given as CPHI)
C                  -----------------------------------
C                        [ZAVG*RG*TG]
C      Horizontal flow:
C       QSC(J , I) =  +/-AREA(J , I) * SQRT  +/- (EPS(J , I)*W)
C                       -----------------
C                         DTSC
C      Here:
C      W = (P(J))**2-(P(I))**2
C      DTSC =      PSC*MW                                  (given as DTSCZE)
C                  ------------------
C                  ZSC**RG*TSC
C      AREA = PI*(D(J , I))**2                             (given as CAREA)
C                  ------------------
C                          4
C      EPS(J , I) =            (MW) * D(J , I)             (given as CEPS)
C                  -------------------------------------------
C                  (ZAVG*RG*TG) 4*FF(J , I)*L(J , I)
```

```
C      The flow rate across a compressor for flow from node j to node i:
C      QSC(J , I)  =                 (1/DTSC) * PC(J , I)
C                     ------------------------------------------------------
C                     [1/KK]*IGEN*TINLET(J)*{[(RIJ)**(KK)-1]+(WIJ)}
C      The flow rate across a compressor For flow from node i to node j:
C      QSC(J , I)  =                 (1/DTSC)*-PC(I , J)
C                     ------------------------------------------------------
C                     [1/KK]*IGEN*TINLET(I)*{[(RJI)**(KK)-1]+(WJI)}
C      Here:
C      WIJ = Z(I)-Z(J)            WJI = Z(J)-Z(I)
C      DTSC =      PSC*MW                          (given as DTSC)
C                 ------------------
C                 ZSC**RG*TSC
C      IGEN = ZAVG*RG/MW                           (given as IGEN)
C      K = CP/CV                  KK = (K-1)/K
C      RIJ = P(I)/P(J)            RJI = P(J)/P(I)
       PI  =3.1415
       CTG   =TG+459.67
       CTSC =TSC+459.67
       CLAMDA=24.*60.*60.*PI/(4.*144.*ZAVG*CTG*10.**6)
       CALPHA=2./(32.2*12.)**3
       CPHI=2.*MW/(1545.3*ZAVG*CTG)
       DTSCZN=32.2*12.**2*PSC/(ZSC*CTSC)
       DTSCZE=12.**2*PSC*MW/(1545.3*ZSC*CTSC)
       CAREA=24.*60.*60.*PI/(4.*144.*10.**6)
       CEPS=32.2*(12.)**3*MW/(4.*1545.3*ZAVG*CTG)
       DTSC=144.*10.**6*PSC*MW/(1545.3*24.*3600.*550.*ZSC*CTSC)
       IGEN=1545.3*ZAVG/MW
```

```
      K=CP/CV
C     Compute NBAND:
C     NBAND is the maximum difference between adjacent nodal numbers.
C     It is used for limiting upper and lower parts of the associated coefficient
C     matrix, F(I , J) computed from the Newton-Raphson method,
C     at the J_th column during generate I _th row in banded matrix,
C     where; 1 =< I , J =< n
C           |<-  NBAND  ->|
C
C         *  *  *  *  *  0  0  0  0
C         *  *  *  *  *  *  0  0  0
C         *  *  *  *  *  *  *  0  0
C         *  *  *  *  *  *  *  *  0
C         *  *  *  *  *  *  *  *  *     -
C         0  *  *  *  *  *  *  *  *     |
C         0  0  *  *  *  *  *  *  *     |  NBAND
C         0  0  0  *  *  *  *  *  *     |
C         0  0  0  0  *  *  *  *  *     -
      NBAND = 0
      DO  11  I = 1,  N
        DO  12  J = 1,  N
          IF (C(I , J) .NE. 0)  THEN
             IF (ABS(I-J) .GT. NBAND)  THEN
                    ABSIJ = ABS(I-J)
                    WIDTH = ABSIJ
             ENDIF
          ENDIF
 12   CONTINUE
```

```
      NBAND = WIDTH
   11 CONTINUE
      WRITE (6,320)  NBAND
      WRITE (6,500)
      WRITE (6,330)
      CALL OUT (C, D, E, L, N, NBAND, NC, P, PC, T, TINLET, V, Z)
C     Set lower and upper limit of the JLOWK_th and JHIGH_th column
C     respectively at the I_th row in order to save time consumed to compute
C     an associated coefficient banded matrix, F(I , J).
C     Set symmetrical metrics of any nodal connection, C(I , J) in pipeline
C     diameter, D(J , I) and length, L(I , J) including Fanning friction factor,
C     FF(I , J) through pipeline roughness, E(I , J).
C     Compute preliminary values at any nodal connection for ALPHA(I , J),
C     LAMDA(I , J), PHI(I , J), AREA(I , J), and EPS(I , J) in all flow rate
C     equations within pipeline segments.
      DO  13  I = 1,  N
            JLOW(I) = MAX0 (1,  I-NBAND)
            JHIGH(I) = MIN0 (N,  I+NBAND)
            JLOWK = JLOW(I)
            JHIGHK = JHIGH(I)
      DO  14  J = JLOWK,  JHIGHK
        IF (J .NE. I)  THEN
            D(J , I) = D(I , J)
            E(J , I) = E(I , J)
            L(J , I) =L(I , J)
            FF(J , I) = FF(I , J)
         IF (C(I , J) .EQ. 1)  THEN
            C(J , I) = C(I , J)
```

```
      IF (Z(J) .NE. Z(I))  THEN
        ALPHA(J , I) = CALPHA*FF(J , I)*L(J , I)/(D(J , I)*(Z(I)-Z(J)))
        LAMDA(J , I) = CLAMDA*(D(J , I))**2
        PHI(J , I) = DEXP(CPHI*(Z(I)-Z(J)))
      ELSE
        AREA(J , I) = CAREA*(D(J , I))**2
        EPS(J , I) = CEPS*D(J , I)/(FF(J , I)*L(J , I))
      ENDIF
      ENDIF
    ENDIF
 14 CONTINUE
 13 CONTINUE
C   Using Newton-RAPHSON method to find out the element values of
C   associated coefficient on the left hand side in banded matrix as follows:
C
C     *  *  *  *  *  0  0  0  0        ?         !
C     *  *  *  *  *  *  0  0  0        ?         !
C     *  *  *  *  *  *  *  0  0        ?         !
C     *  *  *  *  *  *  *  *  0        ?         !
C     *  *  *  *  *  *  *  *  *        ?    =    !
C     0  *  *  *  *  *  *  *  *        ?         !
C     0  0  *  *  *  *  *  *  *        ?         !
C     0  0  0  *  *  *  *  *  *        ?         !
C     0  0  0  0  *  *  *  *  *        ?         !
C
C   * Represent one of following integral values:  1 =< I , J =< N
C   F(I , I) = Partial derivative of the function, F(P) with respect P(I)
C   F(I , J) = Partial derivative of the function, F(P) with respect P(J)
```

```
C     ? = The correction of pressure change, DP(I)
C     ! = The function F(P) represent -F(I , NP1). [NP1=N+1] in N*1 matrix
C     Note: F(P) is the simultaneous nonlinear nodal material balance
C     equations at every node i in the whole network based on the function of
C     the unknown nodal pressures.
C     Nodal material balance equations at every node i:
C     F(P) = V(I), [injection or withdrawal rate]
C     +summation of QSC(I , J) at C(J , I) = 1,
C     [pipeline flow from node j to node i]
C     +summation of QSC(I , J) at C(J , I) = 3,
C     [compressor flow from node j to node i]
C     +summation of QSC(I , J) at C(I , J) = 3,
C     [compressor flow from node i to node j]
C     Beginning iteration counter to solve the elements of coefficient in
C     banded matrix until DP(I) lower than some criterion value or exceeds
C     ITMAX. (maximum number iteration).
   15 DO  60  ITER = 1,  ITMAX
C     Initialized all elements in the banded matrix to zero.
C     Give all F(I , J) = 0.0
      NP1 = N+1
      DO  20  I = 1,  N
        DO  20  J = 1,  NP1
             F(I , J) = 0.
   20 CONTINUE
C     Set lower and upper limit at each I_th row on the left hand side of
C     coefficient banded matrix to save time consumed in the number
C     of columns computed.
      DO  41  I = 1,  N
```

```
            JLOW(I) = MAX0 (1,  I-NBAND)
            JHIGH(I) = MIN0 (N,  I+NBAND)
            JL = JLOW(I)
            JH = JHIGH(I)
C     Checking type of node, T(I) whether it is unknown nodal pressures.
C     T(I) = 1 Nodal pressures specified
C     T(I) = 3 Terminal node with specified injection or withdrawal rate
      IF  (T(I) .NE. 1)  THEN
C     Initial node i to include a possibly specified injection or withdrawal rate
C     given as V(I) (for nodal material balance)
      F(I , NP1) = -V(I)
        DO  40  J = JL,  JH
          IF  (J .NE. I)  THEN
C     Compute in case of pipeline flow between nodes j and i
         IF  (C(J , I) .EQ. 1)  THEN
C     For inclined flow: [Z(J) not equal to Z(I)]
          IF (Z(J) .NE. Z(I))  THEN
             W = (P(J))**2-PHI(J , I)*(P(I))**2
             AL = ALPHA(J , I)*(PHI(J , I)-1)
             WA = W/AL
             LA = LAMDA(J , I)/AL
             DT = LA/DTSCZN
             LC = LAMDA(J , I)/DTSCZN
C     Special case at which i is terminal node with specified injection
C     or withdrawal rate in case of inclined flow.
           IF  (T(I) .EQ. 3)  THEN
             QADD = DTSCZN**2*V(I)*ABS(V(I))
             F(I , J) = -2*(LAMDA(J , I))**2*P(J)/AL
```

```
          F(I , I) = 2*(LAMDA(J , I))**2*P(I)*PHI(J , I)*P(I)/AL
          F(I , NP1) = QADD+LAMDA(J , I)**2*WA
        ELSE
C     Checking whether the W value for inclined flow from node j to node i
C     equals zero. If so, the next iteration of do-loop 40 is performed again.
C     [F(I , J), F(I , I), and F(I , NP1) given as zero]
          IF  (W .NE. 0.)  THEN
C     For inclined flow from node i to node j
           IF  (W .LT. 0.)  THEN
             F(I , J) = DT*P(J)/SQRT(-WA)
             F(I , I) = F(I , I)-DT*PHI(J , I)*P(I)/SQRT(-WA)
             F(I , NP1) = F(I , NP1)+LC*SQRT(-WA)
C     For inclined flow from node j to node i
           ELSE
             F(I , J) = DT*P(J)/SQRT(WA)
             F(I , I)=F(I , I)-DT*PHI(J , I)*P(I)/SQRT(WA)
             F(i , NP1)=F(I , NP1)-LC*SQRT(WA)
           ENDIF
          ENDIF
        ENDIF
C     For horizontal flow: [Z(J) equal to Z(I)]
       ELSE
          W = (P(J))**2-(P(I))**2
          EP = EPS(J , I)*W
          AC = AREA(J , I)/DTSCZE
          EPJ = EPS(J , I)*P(J)
          EPI = EPS(J , I)*P(I)
C     Special case at which i is terminal node with specified injection
```

```
C     or withdrawal rate in case of horizontal flow.
            IF  (T(I) .EQ. 3)  THEN
              QADD = DTSCZE**2*V(I)*ABS(V(I))
              F(I , J) = -2*P(J)*EPS(J , I)*(AREA(J , I))**2
              F(I , I) = 2*P(I)*EPS(J , I)*(AREA(J , I))**2
              F(I , NP1) = QADD+EP*(AREA(J , I))**2
            ELSE
C     Checking whether the W value for horizontal flow from node j to node i
C     equals zero. If so, the next iteration of do-loop 40 is performed again.
C     [F(I,J), F(I,I), and F(I,NP1) given as zero]
             IF  (W .NE. 0.)  THEN
C     For horizontal flow from node i to node j
              IF  (W .LT. 0.)  THEN
                F(I , J) = AC*EPJ/SQRT(-EP)
                F(I , I) = F(I , I)-AC*EPI/SQRT(-EP)
                F(I , NP1)=F(I , NP1)+AC*SQRT(-EP)
C     For horizontal flow from node j to node i
              ELSE
                F(I , J) = AC*EPJ/SQRT(EP)
                F(I , I) = F(I , I)-AC*EPI/SQRT(EP)
                F(I , NP1) = F(I , NP1)-AC*SQRT(EP)
              ENDIF
             ENDIF
            ENDIF
           ENDIF
C     Compute in case of a compressor compresses from node j to node i
          ELSEIF  (C(J , I) .EQ. 3)  THEN
              KK = (K-1)/K
```

```
            RIJ = P(I)/P(J)
            RIJK = RIJ**KK
            RK1 = RIJ**KK-1
            ZZ = Z(I)-Z(J)
            TP = IGEN*TINLET(J)*PC(J , I)*RIJK
            W = (1/KK)*IGEN*TINLET(J)*RK1+ZZ
C     Checking whether the W value for a compressor compressing from
C     node j to node i equals zero. If so, the next iteration of do-loop 40 is
C     performed again. [F(I , J), F(I , I), and F(I , NP1) given as zero]
            IF ((P(J) .NE. P(I)) .OR. (Z(J) .NE. Z(I))) THEN
               F(I , J) = TP/(P(J)*DTSC*W**2)
               F(I , I) = F(I , I)-TP/(P(I)*DTSC*W**2)
               F(I , NP1) = F(I , NP1)-PC(J , I)/(DTSC*W)
            ENDIF
C     Compute in case of a compressor compressing from node i to node j
         ELSEIF  (C(I , J) .EQ. 3)  THEN
            KK=(K-1)/K
            RIJ=P(J)/P(I)
            RIJK=RIJ**KK
            RK1=RIJ**KK-1
            ZZ=Z(J)-Z(I)
            TP=IGEN*TINLET(I)*PC(I , J)*RIJK
            W=(1/KK)*IGEN*TINLET(I)*RK1+ZZ
C     Checking whether the W value for a compressor compressing from
C     node i to node j equals zero. If so, the next iteration of do-loop 40 is
C     performed again. [F(I , J), F(I , I), and F(I , NP1) given as zero]
            IF ((P(J) .NE. P(I)) .OR. (Z(J) .NE. Z(I))) THEN
               F(I , J) = TP/(P(J)*DTSC*W**2)
```

```
                F(I , I) = F(I , I)-TP/(P(I)*DTSC*W**2)
                F(I , NP1)=F(I , NP1)+PC(I , J)/(DTSC*W)
             ENDIF
          ENDIF
        ENDIF
  40    CONTINUE
C      If type of node, T(I) = 1, pressure is fixed [specified], the pressure
C      change, DP(I) is always zero. Therefore, it can be achieved by in
C      the banded matrix as follows:
C                       F(I , J) = 0
C                       F(I , I) =1.
C                       F(I , NP1) = 0
C      In the simultaneous linear equations, the elements in banded matrix can
C      be shown as follows:
C
C                  *  *  *  *  *  0  0  0  0     ?       !
C                  *  *  *  *  *  *  0  0  0     ?       !
C                  *  *  *  *  *  *  *  0  0     ?       !
C      F(I,  I) = 1 *  *  *  *  *  *  *  *  0     ?       !
C      ------->    0  0  0  0  1  0  0  0  0     ?   =   0
C                  0  *  *  *  *  *  *  *  *     ?       !
C                  0  0  *  *  *  *  *  *  *     ?       !
C                  0  0  0  *  *  *  *  *  *     ?       !
C                  0  0  0  0  *  *  *  *  *     ?       !
C
       ELSE
         F(I , I) = 1.
       ENDIF
```

```
 41   CONTINUE
C     SGEM is Gaussian elimination method implemented by the
C     normalization and reduction scheme with column pivoting strategy.
C     Call on subroutine SGEM to solve the elements of coefficient in banded
C     matrix which becomes a diagonal matrix. There results the set solution
C     of DP(I) equal the elements of coefficient of N*1 matrix on the right
C     hand side. The matrix can be shown as follows:
C
C             | 1 0 0 0 0 0 0 0 0 |  | ? |     | 0 |
C             | 0 1 0 0 0 0 0 0 0 |  | ? |     | 0 |
C             | 0 0 1 0 0 0 0 0 0 |  | ? |     | 0 |
C             | 0 0 0 1 0 0 0 0 0 |  | ? |     | 0 |
C             | 0 0 0 0 1 0 0 0 0 |  | ? |  =  | 0 |
C             | 0 0 0 0 0 1 0 0 0 |  | ? |     | 0 |
C             | 0 0 0 0 0 0 1 0 0 |  | ? |     | 0 |
C             | 0 0 0 0 0 0 0 1 0 |  | ? |     | 0 |
C             | 0 0 0 0 0 0 0 0 1 |  | ? |     | 0 |
      CALL  SGEM  (F,  N,  NBAND,  ERR)
C     If ERR = 1 it means some pivot element in the K_th
C     column at any elimination step is equal to zero.
C     The users are requested to check whether they input wrong parameters
C     or setting new initial guesses for pressures and testing program again.
      IF  (ERR .NE. 0)  THEN
      WRITE(*,*)  'AFTER ', ITER,'  ITERATION,  FOUND SOME PIVOT
   +  = 0',
   +  PLEASE CHECKING INPUT PARAMETERS OR SETTING NEW
   +  INITIAL GUESS PRESSURES AGAIN'
      GO TO 95
```

```
      ENDIF
C     Improve the set solution of pressure change, DP(I) by stability factor,
C     SF(I) as follows:
C                   P(I) = P(I)+DP(I)*SF(I)
C     A mathematical technique, it is recommended to use SF(I) = 0.5 for the
C     first iteration to ensure convergence. In subsequent iterations, propose
C     the following scheme for obtaining SF(I) determined by SIGMA(I)
C     which depends on the ratio of DP(I) for the current and previous
C     iteration in every other iteration as below:
C     For SIGMA(I) lower or equal -1    , SF(I) = C1*ABS(SIGMA(I))
C     For SIGMA(I) between -1 and 0     , SF(I) = C2-C3*ABS(SIGMA(I))
c     For SIGMA(I) between 0 and 1      , SF(I) = C4+C5*ABS(SIGMA(I))
C     For SIGMA(I) greater or equal 1   , SF(I) = C6
C     Where:   SIGMA(I) = DP(I) at (K+1)_th iteration
C                         --------------------------------
C                           DP(I) at K_th iteration
C     Note the users must obtain these specifications for SF(I) to improve the
C     stability by giving the coefficients C1, C2, C3, C4, C5 and C6
C     respectively. "The users have to do some experimentation to obtain
C     these coefficients to prove better schemes for the stability, SF(I)
C     applicable according to their own system."
C     (usually 0.0 =< C1, C2, C3, C4, C5, C6 =< 1.0)
C     Setting all initial stability factor, SF(I) equal to 0.5
            DO 43 I = 1, N
                  AF(I) = 0.5
              DP(I) = F(I, N+1)
   43 CONTINUE
            DO 44 I = 1, N
```

```
C       If node i is fixed pressure, DP(I) equals default value as zero.
                IF  (T(I) .NE. 1)  THEN
C       Beginning the second iteration, the stability factor, SF(I) will be
C       improved to avoid instability.
                    IF  (ITER .NE. 1)  THEN
                    SIGMA(I) = DP(I)/DPP(I)
            IF  (SIGMA(I) .LE. -1.)  THEN
                 SF(I) = 0.5*ABS(SIGMA(I))
            ELSEIF ((SIGMA(I) .LT. 0.) .AND. (SIGMA(I) .GT. -1.))  THEN
                 SF(I) = 0.4-0.15*ABS(SIGMA(I))
            ELSEIF ((SIGMA(I) .LT. 1.) .AND. (SIGMA(I) .GT. 0.))  THEN
                 SF(I) = 0.4+0.15*ABS(SIGMA(I))
            ELSEIF (SIGMA(I) .GE. 1.)  THEN
                 SF(I) = 0.5
            ENDIF
                DP(I) = DP(I)*SF(I)
                P(I) = P(I)+DP(I)
                DPP(I) = DP(I)
              ELSE
                   DP(I) = DP(I)*SF(I)
                   P(I) = P(I)+DP(I)
                   DPP(I) = DP(I)
                ENDIF
              ENDIF
  44    CONTINUE
C       Find maximum value of DP(I), given as CONVERG, in order to
C       check convergence whether it less than some criterion value.
        CONVERG = 0.
```

```
      DO 50 I = 1, N
        FIJNP1 = DP(I)
           IF (ABS(FIJNP1) .GT. CONVERG) THEN
              CONVERG = ABS(FIJNP1)
           ENDIF
   50 CONTINUE
C     Checking the convergence for all DP(I) given to compare whether it is
C     less than 0.01.
      IF (CONVERG .LT. 0.01) THEN
C     If the convergence represented as CONVERGE is lower than 0.01.
C     Print messages for CONVERGENCE and then getting the result of new
C     pressures at every node i, at ITER_th iteration.
      WRITE (6,350) ITER
      WRITE (6,360) (I, P(I), I = 1, N)
      WRITE (6,500)
      GO TO 65
C     If the convergence represented as CONVERGE is more than or equal to
C     0.01. Print messages for NO CONVERGENCE and then getting the
C     result of current pressures at every node i, at ITER_th iteration.
      ELSE
      WRITE (6,370) ITER
      WRITE (6,380) (I, P(I), I = 1, N)
      WRITE (6,500)
      ENDIF
C     Call on subroutine UP to generate the new flow rates in all pipeline
C     segments from the new pressures and the old Fanning friction factor
C     after it gave no convergence. and then...
C     compute the Reynolds number that will be used to compute the next
```

```
C       Fanning friction factor for the next iteration.
C       (ITER = 1, ... ITMAX)
        CALL UP (ALPHA, AREA, C, D, DTSCZE, DTSCZN, E, EPS,  FF,
     +  LAMDA, MW, N, NBAND, P, PHI, TG, VT, Z, ZAVG)
   60   CONTINUE
C       If all DP(I) represented as CONVERGE achieve convergence to lower
C       than 0.01 as mentioned above.
C       Compute flow rates between nodes i and j in all nodal connections.
C       QSC(J , I) is positive value for flow from node j to node i.
C       QSC(J , I) is negative value for flow from node i to node j
   65   CONTINUE
        DO  70 I = 1,  N
        DO  70 J = 1,  N
        QSC(I , J) = 0.
   70   CONTINUE
C       Set upper and lower limit of element in banded matrix at J_th column
C       in order to save time consumed for computing.
        DO  80  I = 1,  N
                JLOW(I) = MAX0 (1,  I-NBAND)
                JHIGH(I) = MIN0 (N,  I+NBAND)
                JL = JLOW(I)
                JH = JHIGH(I)
        DO  75  J = JL,  JH
                IF (J .NE. I)  THEN
C       Compute flow rate in case of pipeline flow between nodes j and i.
        IF (C(J , I) .EQ. 1)  THEN
C       In case of inclined flow: [Z(J) .NE. Z(I)]
            IF (Z(J) .NE. Z(I))  THEN
```

```
          W = (P(J))**2-PHI(J , I)*(P(I))**2
          AL = ALPHA(J , I)*(PHI(J , I)-1)
          WA = W/AL
          DT = LAMDA(J , I)/DTSCZN
C     Checking whether the W value of inclined flow from node j to node i
C     equals zero. If so, Q(J , I) = 0.0
         IF (W .NE. 0.)  THEN
C     The flow rate for inclined pipeline flow from node j to node i
          IF (W .GT. 0.)  THEN
             QSC(J , I) = DT*SQRT(WA)
C     The flow rate for inclined pipeline flow from node i to node j
          ELSE
             QSC(J , I) = -DT*SQRT(-WA)
          ENDIF
         ENDIF
C     In case of horizontal flow: [Z(J) .EQ. Z(I)]
        ELSE
          W = (P(J))**2-(P(I))**2
          EP = EPS(J , I)*W
          AC = AREA(J , I)/DTSCZE
C     Checking whether the W value of horizontal flow from node j to node i
C     equals zero. If so, Q(J , I) = 0.0
         IF  (W .NE. 0)  THEN
C     The flow rate for horizontal pipeline flow from node j to node i
          IF (W .GT. 0.)  THEN
             QSC(J , I) = AC*SQRT(EP)
C     The flow rate for horizontal pipeline flow from node i to node j
          ELSE
```

```
             QSC(J , I) = -AC*SQRT(-EP)
           ENDIF
         ENDIF
       ENDIF
C      Compute flow rate in case of a compressor flow from node j to node i
        ELSEIF  (C(J , I) .EQ. 3)  THEN
            KK = (K-1)/K
            RIJ = P(I)/P(J)
            RIJK = RIJ**KK
            RK1 = RIJ**KK-1
            ZZ = Z(I)-Z(J)
            TP = IGEN*TINLET(J)*PC(J , I)*RIJK
            W = (1/KK)*IGEN*TINLET(J)*RK1+ZZ
C      Checking whether both nodal pressures and elevations change for flow
C      from node j to node i. [P(J) .EQ. P(I)...AND...Z(J) .EQ. Z(I)]
C      If not, QSC(J , I) = 0.0
            IF ((P(J) .NE. P(I)) .OR. (Z(J) .NE. Z(I)))  THEN
               QSC(J , I) = PC(J , I)/(DTSC*W)
            ENDIF
C      Compute flow rate in case of a compressor flow from node i to node j
         ELSEIF  (C(I , J) .EQ. 3)  THEN
            KK = (K-1)/K
            RIJ = P(J)/P(I)
            RIJK = RIJ**KK
            RK1 = RIJ**KK-1
            ZZ = Z(J)-Z(I)
            TP = IGEN*TINLET(I)*PC(I , J)*RIJK
            W = (1/KK)*IGEN*TINLET(I)*RK1+ZZ
```

```
C     Checking whether both nodal pressures and elevations change for flow
C     from node i to node j. [P(J) .EQ. P(I)...AND...Z(J) .EQ. Z(I)]
C     If not, QSC(J , I) = 0.0
             IF ((P(J) .NE. P(I)) .OR. (Z(J) .NE. Z(I))) THEN
                 QSC(J , I) = -PC(I , J)/(DTSC*W)
             ENDIF
        ENDIF
      ENDIF
   75 CONTINUE
   80 CONTINUE
C     Arrange non-zero flow rate from node i to node j, where i < j.
C     If the QSC(I , J) < 0.0, change subscript from node j to node i according
C     to the positive direction of flow rate and print it out as positive value.
      TM = 0
      DO  90  I = 1,  N
      DO  85  J = I,  N
      IF (ABS(QSC(I , J)) .NE. 0.)  THEN
         TM = TM+1
         FRIJ(TM) = FF(I , J)
       IF (QSC(I , J) .GT. 0.)  THEN
         II(TM) = I
         JJ(TM) = J
         QIJ(TM) = QSC(I , J)
       ELSE
         II(TM) = J
         JJ(TM) = I
         QIJ(TM) = QSC(J , I)
       ENDIF
```

```
      ENDIF
   85 CONTINUE
   90 CONTINUE
      WRITE (6,390)
      WRITE (6,400) (II(M), JJ(M), QIJ(M), M = 1, TM)
      WRITE (6,500)
      WRITE (6,410)
      WRITE (6,420) (II(M), JJ(M), FRIJ(M), M = 1, TM)
      WRITE (6,500)
   95 STOP
C     Format output statements and parameters for the main program
C     Solution for gas network:
C     N          = ?
C     MW         = ?
C     RG         = ?
C     TG         = ?
C     ZAVG       = ?
C     VT         = ?
C     CP         = ?
C     CV         = ?
C     PSC        = ?
C     TSC        = ?
C     ZSC        = ?
C     NPC        = ?
C     NC         = ?
C     NDL        = ?
C     NT         = ?
C     NV         = ?
```

```
C     ITMAX       = ?
C     The bandwidth of associated coefficient matrix is ?
C     After   ?  iterations for the Newton-Raphson method:
C     It gave no convergence, the current pressures (psia.) are:
C     I    P(I)    I    P(I)    I    P(I)    I    P(I)    I    P(I)
C     ?     ?      ?     ?      ?     ?      ?     ?      ?     ?
C     ?     ?      ?     ?      ?     ?      ?     ?      ?     ?
C     ?     ?      ?     ?      ?     ?      ?     ?      ?     ?
C     After   ?  iterations for the Newton-Raphson method:
C     It gave convergence, the new pressures (psia.) are:
C     I    P(I)    I    P(I)    I    P(I)    I    P(I)    I    P(I)
C     ?     ?      ?     ?      ?     ?      ?     ?      ?     ?
C     ?     ?      ?     ?      ?     ?      ?     ?      ?     ?
C     ?     ?      ?     ?      ?     ?      ?     ?      ?     ?
C     The flow rates, QSC (MMSCFD.) from node i to node j are:
C     I - J   QSC(I , J)   I - J   QSC(I , J)   I - J   QSC(I , J)
C     ? - ?       ?        ? - ?       ?        ? - ?       ?
C     ? - ?       ?        ? - ?       ?        ? - ?       ?
C     ? - ?       ?        ? - ?       ?        ? - ?       ?
C     The Fanning friction factors connecting nodes i and j are:
C     I - J    F(I , J)    I - J    F(I , J)    I - J    F(I , J)
C     ? - ?       ?        ? - ?       ?        ? - ?       ?
C     ? - ?       ?        ? - ?       ?        ? - ?       ?
C     ? - ?       ?        ? - ?       ?        ? - ?       ?
 100  FORMAT (79X/10X,I6/10X,F12.6/10X,F12.6/10X,F12.6/10X,
    + F12.6/10X,F12.6/10X,F12.6/10X,F12.6/10X,F12.6/10X,F12.6/10X,
    + F12.6/(10X, I6))
 200  FORMAT (10X)
```

```
300   FORMAT (/5X'Solution for gas network :'//
  +          5X'N          = ' I4/
  +          5X'MW         = ' F8.3/
  +          5X'RG         = ' F7.2/
  +          5X'TG         = ' F7.2/
  +          5X'ZAVG       = ' F7.2/
  +          5X'VT         = ' F8.3/
  +          5X'CP         = ' F9.4/
  +          5X'CV         = ' F9.4/
  +          5X'PSC        = ' F7.2/
  +          5X'TSC        = ' F7.2/
  +          5X'ZSC        = ' F7.2)
310   FORMAT (5X'NPC = ' I4/
  +          5X'NC         = ' I4/
  +          5X'NDL        = ' I4/
  +          5X'NT         = ' I4/
  +          5X'NV         = ' I4/
  +          5X'ITMAX      = ' I4)
320   FORMAT (5X'The bandwidth of associated coefficient matrix is'  I2)
330   FORMAT (5X,'I  -  J',5X,'C(I , J)',6X,'D(I , J)',9X,'L(I , J)',7X,'E(I , J
  +   )',8X,'PC(I , J)')
350   FORMAT (5X,'After'2X,I3,3X'iterations for the Newton-Raphson
  +   method :'//
  +   5X,'It gave convergence, the new pressures (psia) are:'//
  +   5X,'I'6X,'P(I)'5X,'I'6X,'P(I)'5X,'I'6X,'P(I)'5X,'I'6X,'P(I)',5X,'I
  +   '6X,'P(I)')
360   FORMAT (5(3X,(I3,F10.3)))
370   FORMAT (5X,'After'2X,I3,3X'iterations for the Newton-Raphson
```

```
     +  method:'//
     +  5X,'It gave no convergence, the current (psia) pressures are:'//
     +  5X,'I'6X,'P(I)'5X,'I'6X,'P(I)'5X,'I'6X,'P(I)'5X,'I'6X,'P(I)',5X,'I
     +  '6X,'P(I)')
 380    FORMAT (5(3X,(I3,F10.3)))
 390    FORMAT (4x'The flow rates, QSC (MMSCFD) at standard conditions
     +  from node i to node j are ;'//
     +  5X,'I  -  J',6X,'QSC(I , J)'5X,'I  -  J'6X,'QSC(I , J)'5X,'I  -  J'
     +  6X,'QSC(I , J)')
 400    FORMAT (3(4X,(I2,'  -',I4,1X,F13.3)))
 410    FORMAT (4x'The Fanning friction factor connecting node i and j are:'//
     +  5X,'I  -  J',5X,'F(I , J)'5X,'I  -  J'5X,'F(I , J)'5X,'I  -  J'5X,'F(I , J)')
 420    FORMAT (3(4X,(I2,'  -',I4,1X,F10.6)))
 500    FORMAT (10X/4X'------------------------------------------------
     +  ---------------------------'/)
        END
        SUBROUTINE OUT (C, D, E, L, N, NBAND, NC, P, PC, T, TINLET,
     +  V, Z)
C        Print input parameters for gas network:
C       1. node i to node j
C       2. nodal connection, C
C       3. inlet temperature, TINLET and compression power, PC
C       4. pipeline diameters, D and lengths, L joining node i and j
C       5. starting guesses and specified pressure, P
C       6. node-type vector, T:
C        T(I) : 1 = pressure specified
C        T(I) : 2 = injection rate specified
C       T(I) : 3 = terminal node with specified injection or withdrawal rate
```

```
C      7. nodal injection or withdrawal rates, V:
C      V(I) : positive value = injection  rates
C      V(I) : negative value = withdrawal rates
C      8. nodal elevations, Z
C      Type declaration variables
       REAL*8 D(35,35), E(35,35), L(35,35), PC(35,35), P(35), Z(35),
    +  TINLET(35), V(35), VD(1225), VE(1225), VL(1225), VPC(1225)
       INTEGER*4 C(35,35), IRI(35), JCJ(35), JHIGH(35), JLOW(35),
    +  T(35), VC(1225),
    +  IR, JC, JH, JL, N, NBAND, NC, TM
C      Set upper and lower limit of a banded matrix in order to save time
C      consumed for computing
       TM = 0
       DO  50  IR = 1 ,  N
             JLOW(I) =MAX0 (1 ,  IR-NBAND)
             JHIGH(I) = MIN0 (N ,  IR+NBAND)
             JL = JLOW(I)
             JH=JHIGH(I)
       DO  45  JC = JL ,  JH
          IF  (IR .NE. JC)  THEN
C      Checking all parameters used in the network whether both node i, (IR)
C      and node j, (JC) are connected. [C(I , J) not equal to zero] If so, it is
C      stored in one dimensional array of variables before printing it out later.
       IF (C(IR , JC) .EQ. 1  .OR.  C(IR , JC) .EQ. 3)  THEN
          TM = TM+1
          IRI(TM) = IR
          JCJ(TM) = JC
          VC(TM) = C(IR , JC)
```

```
         VD(TM) = D(IR , JC)
         VL(TM) = L(IR , JC)
         VE(TM) = E(IR , JC)
         VPC(TM) = PC(IR , JC)
       ENDIF
       ENDIF
 45    CONTINUE
 50    CONTINUE
       WRITE (6,209) (IRI(TM), JCJ(TM), VC(TM),  VD(TM), VL(TM),
  +    VE(TM),VPC(TM), TM = 1,  NC)
       WRITE (6,109)
       WRITE (6,309)
       WRITE (6,409) (I, T(I), P(I), Z(I), V(I), TINLET(I), I = 1,  N)
       WRITE (6,109)
C      Format input parameters for the SUBROUTINE OUT:
C      I  -  J  :  C(I , J)  :  D(I , J)  :  L(I , J)  :  E(I , J)  :  PC(I , J)
C      ?  -  ?  :     ?      :     ?      :     ?      :     ?      :     ?
C      ?  -  ?  :     ?      :     ?      :     ?      :     ?      :     ?
C        I      :   T(I)     :    P(I)    :    Z(I)    :    V(I)    :  TINLET(I)
C        ?      :     ?      :     ?      :     ?      :     ?      :     ?
C        ?      :     ?      :     ?      :     ?      :     ?      :     ?
109    FORMAT (10X/4X'-----------------------------------------------------
  +    ----------------------------'/)
209    FORMAT (4X,I2,'  -',I3,3X,':',3X,I2,4X,':'1X,F8.3,'   :',3X,F8.2,
  +    13X,':',2X,F7.5,3X,':',4X,F6.2)
309    FORMAT (8X,'I',9X,'T(I)',9X,'P(I)',11X,'Z(I)',9X,'V(I)',6X,'TINLE
  +    T(I)')
409    FORMAT (6X,I3,6X,':',3X,I2,4X,':',1X,F8.3,3X,':',3X,F8.3,3X,':',
```

```
     +  1X,F8.3,3X,':',4X,F6.2,)
        RETURN
        END
C       Solution for special Gaussian elimination method
C       Subroutine SGEM
C       Purpose: to solve a system of simultaneous linear equations with
C       elements on the left hand side in a banded matrix.
C
C             * * * * * 0 0 0 0     ?         !
C             * * * * * * 0 0 0     ?         !
C             * * * * * * * 0 0     ?         !
C             * * * * * * * * 0     ?         !
C             * * * * * * * * *     ?    =    !
C             0 * * * * * * * *     ?         !
C             0 0 * * * * * * *     ?         !
C             0 0 0 * * * * * *     ?         !
C             0 0 0 0 * * * * *     ?         !
C
C       Usage: Call SGEM (F, N, NBAND, ERR)
C       Description of parameters:
C       N - Number of columns in square matrix.
C       NBAND - Number of upper or lower codiagonals in square matrix.
C       ERR - Error parameter coded as below:
C       ERR = 0 : No error
C       ERR = 1 : Found some pivot element at any elimination step equal to
C       zero (initial guess value for nodal pressures should be changed)
C       F - Element of associated coefficient in the simultaneous linear
C       equations represent as two separated cases in matrix as follows:
```

```
C
C        * * * * * 0 0 0 0
C        * * * * * * 0 0 0
C        * * * * * * * 0 0
C        * * * * * * * * 0     * Represent F(I , J):
C        * * * * * * * * *       1 =< I , J =< N
C        0 * * * * * * * *
C        0 0 * * * * * * *
C        0 0 0 * * * * * *
C        0 0 0 0 * * * * *
C
C        !
C        !
C        !     ! Represent as F(I , N+1):
C        !       1 =< I =< N
C        !
C
C     Note: Return of F(I,N+1) contains the solution given as DP(I).
C     Method: to get set of solution, DP(I) by Gaussian elimination method
C     with column pivoting only, implemented by normalization and
C     reduction scheme until banded matrix becomes diagonal matrix.
      SUBROUTINE SGEM (F, N, NBAND, ERR)
C     Type declaration variables
      REAL*8 F(36,36), AIJCK, TB, TM
      INTEGER*4 IROW(35), I, ID, II, ILR, IROWK, J, JJ, KST, N
C     Start L-U decomposition loop at K = 1, 2, 3, ... , N
      ERR = 0
      KST = 1
```

```
      DO 38  K = 1,  N
            ILR = K+NBAND
            IF (ILR .GT. N)  THEN
                  ILR = N
            ENDIF
C     Search pivot in KST_th column for row indexes from I = K up to
C     I  = ILR.·The element in the K_th column with the greatest absolute
C     value is the pivot element.
C                         KST_th column = 1, 2, ...
C                         ------->
C                       ┌─                       ─┐
C        I = K   ------>│ *  *  *  *  *  0  0  0  0  !
C                       │ *  *  *  *  *  *  0  0  0  !
C                       │ *  *  *  *  *  *  *  0  0  !
C                       │ *  *  *  *  *  *  *  *  0  !  Represent as F(I , J):
C        I = ILR ------>│ *  *  *  *  *  *  *  *  *  !  1  =<  I , J  =<  N+1
C                       │ 0  *  *  *  *  *  *  *  *  !
C                       │ 0  0  *  *  *  *  *  *  *  !
C                       │ 0  0  0  *  *  *  *  *  *  !
C                       │ 0  0  0  0  *  *  *  *  *  !
C                       └─                       ─┘
      PIVOT = 0.
      J = KST
      DO 22  I = K,  ILR
            IF  (ABS(F(I,J)) .GT. ABS(PIVOT))  THEN
                  PIVOT = F(I , J)
                  IROW(K) = I
            ENDIF
 22   CONTINUE
C     Checking whether the pivot becomes zero. If not,
```

```
C       the banded matrix, F(I , J) can be implemented by normalization and
C       reduction further. If so, the subroutine SGEM will return and give
C       warning error messages.
        IF  (PIVOT .EQ. 0.)  THEN
        ERR = 1
        GO TO 50
        ENDIF  .
C       Normalize pivot row elements:
C       At the row of the KST_th column given as pivot element will be
C       normalized by dividing with pivot from KST_th column = 1 to N+1.
C                           KST_th column = 1, 2, ...
C                          ------->
C                        |                             |
C                        |  *  *  *  *  *  0  0  0  0  !  |
C       F(I , J) = F(I , J)| #  #  #  #  #  #  0  0  0  #  |  <-- Normalized Row
C                --------|  *  *  *  *  *  *  *  0  0  !  |
C                   PIVOT|  *  *  *  *  *  *  *  *  0  !  | Represent as F(I , J):
C                        |  *  *  *  *  *  *  *  *  *  !  | 1 =< I , J =< N+1
C                        |  0  *  *  *  *  *  *  *  *  !  |
C                        |  0  0  *  *  *  *  *  *  *  !  |
C                        |  0  0  0  *  *  *  *  *  *  !  |
C                        |  0  0  0  0  *  *  *  *  *  !  |
        NP1 = N+1
        IROWK = IROW(K)
        DO  14  J = K,  NP1
        F(IROWK , J) = F(IROWK , J)/PIVOT
  14    CONTINUE
C       Interchange pivot row elements:
C       At the row of the KST_th column given as pivot element after
```

```
C      normalized already, it is checked to see whether the row (IROWK) of
C      pivot element equal to the top of row indexes from I = K up to I = ILR.
C      If not, it will be interchanged between the row of pivot element,
C      (IROWK) from J_th column = K to N+1 and the row indexes, (K) from
C      J_th column = K to N+1 each other. (in order to get main diagonal
C      becomes 1)
C                    KST_th column = 1, 2, ...
C
C                       * * * * * 0 0 0 0 !    Index Row
C           Pivot Row   # # # # # # 0 0 0 #
C           1             * * * * * * * 0 0 !
C                       * * * * * * * * 0 !    Represent as F(I , J):
C                       * * * * * * * * * !    1 =< I , J =< N+1
C                       0 * * * * * * * * !
C                       0 0 * * * * * * * !
C                       0 0 0 * * * * * * !
C                       0 0 0 0 * * * * * !
C
       IF  (K .NE. IROWK)  THEN
       DO  30  J = K,  NP1
             TM = F(K , J)
             F(K , J) = F(IROWK , J)
             F(IROWK , J) = TM
 30    CONTINUE
       ENDIF
C      Carry out pivot row reduction:
C      Compute to reduce the elements below main diagonal becomes zero.
C      (to set as lower triangular matrix)
```

```
C                    KST_th column = 1, 2, ...
C                    ------>
C      I = K    ---> 1 # # # # # # # # #
C                    0 1 # # # # # # # #
C                    0 0 1 # # # # # # #
C                    0 0 0 1 # # # # # #   Represent as F(I ,J):
C      I = ILR  ---> 0 0 0 0 1 # # # # #   1 =< I , J =< N+1
C                    0 0 0 0 0 1 # # # #
C                    0 0 0 0 0 0 1 # # #
C                    0 0 0 0 0 0 0 1 # #
C                    0 0 0 0 0 0 0 0 1 #
      II = K+1
      JJ = KST
      DO 18 I = II, ILR
      AIJCK = -F(I , JJ)
      DO 17 J = JJ, NP1
      F(I , J) = F(I , J)+AIJCK*F(JJ , J)
 17   CONTINUE
 18   CONTINUE
C     Iterative element forward reduction from K = 1 until K = N until
C     elements of lower codiagonal become zero.
C     KST is the value for generating the next column and pivot element
C     and reducing all elements of pivot column below main diagonal to zero.
      KST = KST+1
 38   CONTINUE
C     End of L-U decomposition loop
C     Back substitution:
C     Compute the correction, DP(I) represent as F(I , NP1), (!) [NP1=N+1]
```

```
C       from the coefficient on the end of right hand side in matrix by
C       successive iterative substitution in all the elements of main diagonal
C       until the upper triangular matrix becomes zero.
C
C               1  0  0  0  0  0  0  0  0  #
C               0  1  0  0  0  0  0  0  0  #
C               0  0  1  0  0  0  0  0  0  #
C               0  0  0  1  0  0  0  0  0  #
C               0  0  0  0  1  0  0  0  0  #
C               0  0  0  0  0  1  0  0  0  #
C               0  0  0  0  0  0  1  0  0  #
C               0  0  0  0  0  0  0  1  0  #
C               0  0  0  0  0  0  0  0  1  #
C
        II = N
        DO 45 I = 2, N
            II = II-1
            ID = II
            TB = F(II , NP1)
            ID = ID+1
        DO 43 JJ = ID, N
 43         TB = TB-F(II , JJ)*F(JJ , NP1)
        F(II , NP1) = TB
 45     CONTINUE
 50     RETURN
        END
C       Solution for the next approximation of Fanning friction factor
C       Subroutine UP
```

```
C     Purpose: to compute the new approximation for the Fanning friction
C     factor after the set solution of DP(I) gave no convergence.
C     They are used for generating to find out the new nodal pressures at the
C     next iteration of the Newton-Raphson method.
C     Usage: Call UP (ALPHA, AREA, C, D, DTSCZE, DTSCZN, E, EPS,
C     FF, LAMDA, MW, N, NBAND, P, PHI, TG, VT, Z, ZAVG)
C     Description of parameters:
C     C - Nodal connection matrix
C     D - Pipeline diameter matrix
C     E - Pipeline roughness matrix
C     FF - Fanning friction matrix
C     MW - Molecular weight
C     N - Number of column in square matrix.
C     NBAND - Number of upper or lower codiagonals in square matrix.
C     P - Nodal pressures
C     TG - Gas temperature
C     VT - Average gas viscosity
C     Z - Nodal elevations
C     ZAVG - Average gas compressibility factor
C     Method: to get the new approximation of Fanning friction factor by
C     computing the Reynolds number given from the flow rate equations
C     then compare pattern of flow region to compute the value of FF(I , J)
      SUBROUTINE UP (ALPHA, AREA, C, D, DTSCZE, DTSCZN, E,
    + EPS, FF, LAMDA, MW, N, NBAND, P, PHI, TG, VT, Z, ZAVG)
C     Type declaration variables
      REAL*8 ALPHA(35,35), AREA(35,35), D(35,35), E(35,35),
    + EPS(35,35), FF(35,35), FIJ(35,35), LAMDA(35,35), QSC(35,35),
    + PAVG(35,35),PHI(35,35),RE(35,35), P(35), Z(35),
```

```
     +  AC, AL, DT, DTAVG, DTSCZE, DTSCZN, ED, EP, MW, RIJ, TG,
     +  W, WA, ZAVG
        INTEGER*4 C(35,35), JHIGH(35), JLOW(35), JH, JL, N
C       Compute to update flow rates, QSC(I , J) for all pipeline segments
C       connecting nodes i and j  C(I , J) = 1 from the new pressure, P(I) and
C       the old Fanning friction factor, FF(I , J) in the equation as follows:
C       Inclined flow from node j to node i:
C       QSC(J , I) = +/-(LAMDA(J , I)) * SQRT        +/-(W)
                       -------------------        --------------------------------
                             DTSCZN               ALPHA(J , I)*(PHI(J , I)-1)
C       Here:
C       W = (P(J))**2-PHI(J , I)*(P(I))**2
C       Horizontal flow flow from node j to node i:
C       QSC(J , I) =  +/-(AREA(J , I)) * SQRT        +/-(W)
C                      -------------------        --------------------------------
C                            DTSCZE               ALPHA(J , I)*(PHI(J , I)-1)
C       Here:
C       W = (P(J))**2-(P(I))**2
C       The flow rates are generated to compute the Reynolds number, RE(I , J)
C       by considering two regions to get the new Fanning friction factor as
C       follows:
C       RE(J , I)  =   4*12*10**5*144*10**6*QSC(J , I)*PAVG(J , I)*MW
C                    --------------------------------------------------------------------
C                    32.2*2.089*1545.3*24*3600*PI*VT*D(J , I)*ZAVG*TG
C       Here:
C       PAVG(J , I) = 2.0*[P(J)**3-P(I)**3]
C                     -------------------------
C                     3.0*[P(J)**2-P(I)**2]
```

```
C     For RE(J , I) > 4000
C     FF(J , I) = {-1.737*DLOG[ED-RIJ*DLOG(ED+(14.5/RE(J , I)))]}**-2
C     Here:
C     ED = 0.269*E(J , I)/D(J , I)        RIJ = 2.185/RE(J , I)
C     For RE(J , I)  =<  2000
C     FF(J , I)  =      16
C                    ---------
C                    RE(J , I)
      PI = 3.14159
      DO  10  I = 1,  N
      DO  10  J = 1,  N
            QSC(I , J) = 0.
            RE(I , J) = 0.
            FIJ(I , J) = 0.
            PAVG(J , I) = 0.
 10   CONTINUE
C     Set upper and lower limit of element in banded matrix at J_TH column
C     in order to save time consumed for computing.
      DO  30  I = 1,  N
            JLOW(I) = MAX0 (1,  I-NBAND)
            JHIGH(I) = MIN0 (N,  I+NBAND)
            JL = JLOW(I)
            JH = JHIGH(I)
      DO  20  J = JL,  JH
            IF  (J .NE. I)  THEN
C     Checking whether node j and node i is nodal connection.
       IF  (C(J , I) .EQ. 1)  THEN
C     In case of inclined flow: [Z(J) .NE. Z(I)]
```

```
      IF  (Z(J) .NE. Z(I))  THEN
        W = (P(J))**2-PHI(J , I)*(P(I))**2
        AL = ALPHA(J , I)*(PHI(J , I)-1)
        WA = W/AL
        DT = LAMDA(J , I)/DTSCZN
       IF  (W .NE. 0.)  THEN
        IF  (W .GT. 0.)  THEN
          QSC(J , I) = DT*SQRT(WA)
        ELSE
          QSC(J , I) = -DT*SQRT(-WA)
        ENDIF
        P3 = (P(J))**3-(P(I))**3
        P2 = (P(J))**2-(P(I))**2
        PAVG(J , I) = (2./3.)*(P3/P2)
        DTAVG = PAVG(J , I)*MW/(ZAVG*TG)
        RE(J , I)=76963.156*ABS(QSC(J , I)*DTAVG)/(PI*VT*D(J , I))
C     For  RE(J , I)  >  4000:
C     The Fanning friction factor is computed as follows:
        IF  (RE(J , I) .GT. 4000.)  THEN
            ED = 0.269*12.*E(J , I)/D(J , I)
            RIJ = 2.185/RE(J , I)
            A = DLOG(ED+(14.5/RE(J , I)))
            B = DLOG(ED-(RIJ*A))
            FIJ(J , I) = (-1.737*B)**(-2)
C     For  R(J , I)  =<  2000:
C     The Fanning friction factor is computed as follows:
        ELSEIF  (RE(J , I) .LE. 2000.)  THEN
             FIJ(J , I) = 16/RE(J , I)
```

```
C     If the Reynolds number given in the transition region.
C     [2000 < RE =<4000]
C     It is assumed as the default value of the old Fanning friction factor.
             ELSE
               FIJ(J , I) = FF(J , I)
             ENDIF
C     If the flow rate at standard conditions is computed as zero.
C     It is assumed as the default value of the old Fanning friction factor.
           ELSE
             FIJ(J , I) = FF(J , I)
           ENDIF
C     In case of horizontal flow: [Z(J) .EQ. Z(I)]
         ELSE
           W = (P(J))**2-(P(I))**2
           EP = EPS(J , I)*W
           AC = AREA(J , I)/DTSCZE
           IF (W .NE. 0) THEN
           IF (W .GT. 0.) THEN
              QSC(J , I) = AC*SQRT(EP)
           ELSE
              QSC(J , I) = -AC*SQRT(-EP)
           ENDIF
           P3 = (P(J))**3-(P(I))**3
           P2 = (P(J))**2-(P(I))**2
           PAVG(J , I) = (2./3.)*(P3/P2)
           DTAVG = PAVG(J , I)*MW/(ZAVG*TG)
           RE(J , I) = 76963.156*ABS(QSC(J , I)*DTAVG)/(PI*VT*D(J , I))
C     For RE(J , I) > 4000:
```

```
C      The Fanning friction factor is computed as follows:
            IF  (RE(J , I) .GT. 4000.)  THEN
                ED = 0.269*12.*E(J , I)/D(J , I)
                RIJ = 2.185/RE(J , I)
                A = DLOG(ED+(14.5/RE(J , I)))
                B = DLOG(ED-(RIJ*A))
                FIJ(J , I) = (-1.737*B)**(-2)
C      For  R(J , I)  =<  2000 :
C      The Fanning friction factor is computed as follows:
            ELSEIF  (RE(J , I) .LE. 2000.)  THEN
                FIJ(J , I) = 16/RE(J , I)
C      If the Reynolds number is given in the transition region.
C      [2000  <  RE  =<  4000]
C      It is assumed as the default value of the old Fanning friction factor.
            ELSE
              FIJ(J , I) = FF(J , I)
            ENDIF
C      If the flow rate at standard conditions is computed as zero.
C      It is assumed as the default value of the old Fanning friction factor.
          ELSE
            FIJ(J , I) = FF(J , I)
          ENDIF
        ENDIF
       ENDIF
      ENDIF
 20   CONTINUE
 30   CONTINUE
C      Return all values of FIJ(J , I) into FF(J , I) before leaving subroutine UP
```

```
      DO  50  I =1,  N
            JL = JLOW(I)
            JH = JHIGH(I)
      DO  40  J = JL,  JH
            IF  (J .NE. I)  THEN
                  IF  (C(J , I) .EQ. 1)  THEN
                        FF(J , I) = FIJ(J , I)
                  ENDIF
            ENDIF
40    CONTINUE
50    CONTINUE
      RETURN
      END
```

CURRICULUM VITAE

Name	Mr. Patikom Saelee
Birth Date	October 13, 1965
Nationality	Thai
University Education	
1985-1989	Bachelor's Degree of Engineering Mechanical Engineering Prince of Songkla University
Working Experience	
1989-1994	Maintenance Engineer Gas Separation Plant Petroleum Authority of Thailand
1994-Present	Researcher Research and Technology Institute Petroleum Authority of Thailand