## 1. Listings of sample program valuing put options

```
C*********************************************************************
C
C   PROGRAM:  SAMPLE 1: STRIPPED DOWN PROGRAM USING CRANK-NICHOLSON
C             SUBROUTINES CNSET & CNSTEP() TO VALUE PUT OPTIONS ON
C             A NON-DIVIDEND PAYING SECURITY.
C
C*********************************************************************

      IMPLICIT  DOUBLE PRECISION (A-H,K-L,O-Z)
      DIMENSION U(41), ARR(41,4), PARM(15)

C**** SET MODEL PARAMETERS ********************************************

C     Crank-Nicholson algorithm parameters:

      DATA N, IMIN, IMAX, SMIN, SMAX, K / 41, 0, 0, 0.0, 100.0, .025 /

C     Financial model parameters:

      DATA TMAT, R, SIGMA, XPRICE, IFUT / .25, .10, .20, 50.0, 0 /
      PARM(1) = SIGMA
      PARM(2) = R

C**** SETUP TO SOLVE PDE **********************************************

      CALL CNSET (N-1,SMIN,SMAX,K,IFN,IFUT,IMIN,IMAX,PARM,ARR)

      DO 100 I = 1, N
         S    = SMIN + ( SMAX - SMIN ) * DBLE(I-1) / DBLE(N-1)
100      U(I) = MAX ( 0.0D0 , XPRICE - S )

C**** SOLUTION LOOP TO SOLVE PDE **************************************

      DO 200 T = 0.0D0, TMAT, K
200      CALL CNSTEP ( T, U, ARR )

C**** PRINT RESULTS TO SCREEN *****************************************

      DO 300 I = 1, 41, 2
         S    = SMIN + ( SMAX - SMIN ) * DBLE(I-1) / DBLE(N-1)
300      PRINT 350, S, U(I)

350   FORMAT (F8.2, 10X, F10.4)

      STOP
      END
```

```
C**** FUNCTION DEFINITIONS REQUIRED ************************************
C
C     The coefficients for the Black-Scholes option pricing model, with
C     S being the stock price, and R and D the assumed constant interest
C     rate and proportional dividend rate respectively, are
C     FNA()  = SIGS * SIGS * S * S / 2.0
C     FNB()  = (R - D) * S
C     FNC()  = -R
C
C***********************************************************************

      DOUBLE PRECISION FUNCTION COEFF()

      IMPLICIT  DOUBLE PRECISION (A-H,K-L,O-Z)
      DIMENSION PARM(15)

      ENTRY FNA(S,IFN,PARM)
         SIGMA  = PARM(1)
         FNA    = (SIGMA * S) ** 2 / 2.0
      RETURN

      ENTRY FNB(S,IFN,PARM)
         R      = PARM(2)
         FNB    = R * S
      RETURN

      ENTRY FNC(S,IFN,PARM)
         FNC    = -R
      RETURN

      ENTRY FMIN(T,IFN,PARM)
         FMIN    = 0.0
      RETURN

      ENTRY FMAX(T,IFN,PARM)
         FMAX    = 0.0
      RETURN
      END
```

# Solving One Factor Models

## 2. Listings of CNSET and CNSTEP

```
      SUBROUTINE CNSET (IN,SMIN,SMAX,K,IFN,IFUT,ISMN,ISMX,PARM,ARR)
```

```
C*********************************************************************
C
C  SUBROUTINE CNSET (...)
C
C  Subroutine sets up coefficient array needed for Crank-Nicholson
C  algorithm to solve 1 state variable plus time partial differential
C  equations.  Use in conjunction with routine CNSTEP.  PDE has form
C
C               FNA * Uss + FNB * Us + FNC * U - Ut = 0
C
C  Arguments: IN      number of grid intervals in state space S
C             SMIN    minimum value of state variable S
C             SMAX    maximum value of state variable S
C             K       step size in time direction
C             IFN     flag available for passing to coeff. fcns.
C             IFUT    flag setting FNC = 0 for futures contract pricing
C             ISMN    flag for SMIN boundary (0 for quadratic extrapol.)
C             ISMX    flag for SMAX boundary (1 for given values      )
C             PARM    vector of model parameters for coeff. fcns.
C             ARR     output array of coefficients for CNSTEP
C                     dimension as (IN+1,4) in calling program
C
C  Other routines called:  functions FNA, FNB, FNC(S,IFN,PARM) must be
C                     externally defined and available to subroutine.
C
C  Author:     R. A. Jones     15 December 1988
C
C*********************************************************************

      IMPLICIT  DOUBLE PRECISION ( A-H, K-L, O-Z )
      COMMON   /CNCOM/ N,ISMIN,ISMAX
      DIMENSION PARM( 15 ), ARR( 0:IN, 4 )

      N      = IN
      ISMIN  = ISMN
      ISMAX  = ISMX
      H      = ( SMAX - SMIN ) / DBLE( N )
      FUTURE = 1.0
      IF ( IFUT .EQ. 1)    FUTURE = 0.0

C**** FIRST DO 'INTERIOR' COEFFICIENTS ********************************

      DO 100  I = 1, N-1

         S        =  SMIN +  DBLE(I) * H
         AX       =  FNA(S,IFN,PARM) * 2.0 * K
         BX       =  FNB(S,IFN,PARM) * H * K
         CX       =  FNC(S,IFN,PARM) * FUTURE * 2.0 * H * H * K
         DENOM    =  CX - 2.0 * AX - 4.0 * H * H
```

```
          ARR(I,1) =  ( AX - BX ) / DENOM
          ARR(I,2) =  1.0
          ARR(I,3) =  ( AX + BX ) / DENOM
          ARR(I,4) =  1.0 + 8.0 * H * H / DENOM

 100  CONTINUE

C**** THEN HANDLE BOUNDARIES ACCORDING TO FLAGS ***********************

      IF ( ISMIN .EQ. 1 )  THEN

C         CASE OF KNOWN VALUE AT SMIN: ISMIN = 1
          ARR(0,1) =  0.0
          ARR(0,2) =  1.0
          ARR(0,3) =  0.0

      ELSE

C         CASE OF QUADRATIC EXTRAPOLATION AT SMIN: ISMIN = 0
          G        =  ARR(1,3) / ( ARR(2,2) + 3.0 * ARR(2,3) )
          ARR(0,1) =  0.0
          ARR(0,2) =  G * ARR(2,3) - ARR(1,1)
          ARR(0,3) =  G * ( ARR(2,1) - 3.0 * ARR(2,3) ) - ARR(1,2)
          ARR(0,4) =  G

      ENDIF

      IF ( ISMAX .EQ. 1 )  THEN

C         CASE OF KNOWN VALUE AT SMAX: ISMAX = 1
          ARR(N,1) =  0.0
          ARR(N,2) =  1.0
          ARR(N,3) =  0.0

      ELSE

C         CASE OF QUADRATIC EXTRAPOLATION AT SMAX: ISMAX = 0
          G        =  ARR(N-1,1) / ( ARR(N-2,2) + 3.0 * ARR(N-2,1) )
          ARR(N,1) =  G * ( ARR(N-2,3) - 3.0 * ARR(N-2,1) ) - ARR(N-1,2)
          ARR(N,2) =  G * ARR(N-2,1) - ARR(N-1,3)
          ARR(N,3) =  0.0
          ARR(N,4) =  G

      ENDIF

      RETURN
      END
```

```
      SUBROUTINE CNSTEP ( T, U, ARR )

C***********************************************************************
C
C  SUBROUTINE CNSTEP (...)
C
C  Subroutine takes 1 step in time direction in solving 1 state variable
C  PDE using Crank-Nicholson algorithm.  T is current time used only for
C  passing to boundary value functions FMIN(T) and FMAX(T) if ISMIN or
C  ISMAX are set to 1.  U(0:N) is N+1 dimensional vector of solution so
C  far.  ARR() is coefficient array set up by CNSET().
C
C***********************************************************************

      IMPLICIT   DOUBLE PRECISION ( A-H, K-L, O-Z )
      COMMON    /CNCOM/ N, ISMIN, ISMAX
      DIMENSION  ARR( 0:N, 4 ), U( 0:N )

C     NOTE: PARAMETER NMAX MUST BE .GE. N FOR TRIDAG ALGORITHM
      PARAMETER ( NMAX = 200 )
      COMMON    /TRICOM/ D( 0:NMAX ), GAM( 0:NMAX )

C     SET UP RIGHT HAND SIDE OF SYSTEM TRIDIAGONAL SYSTEM (ABC)U = D

      DO 100  I = 1, N-1
         D(I) = - ARR(I,1)*U(I-1) - ARR(I,4)*U(I) - ARR(I,3)*U(I+1)
 100  CONTINUE

      IF ( ISMIN .EQ. 1 )  THEN
C        GET SOLUTION VALUE AT RMIN
         D(0) = FMIN(T)
      ELSE
         D(0) = D(2) * ARR(0,4) - D(1)
      ENDIF

      IF ( ISMAX .EQ. 1 )  THEN
C        GET SOLUTION VALUE AT RMAX
         D(N) = FMAX(T)
      ELSE
         D(N) = D(N-2) * ARR(N,4) - D(N-1)
      ENDIF

      CALL TRIDAG ( ARR(0,1), ARR(0,2), ARR(0,3), D, GAM, U, N )

      RETURN
      END
```

```
C**** TRIDIAGONAL SOLN. ALGORITHM FROM "NUMERICAL RECIPES", P. 40 ******

C     SOLVES: (ABC)X = D FOR X.  N=DIMENSION.  A,B,C,D, NOT ALTERED
C     NOTE: SUBSCRIPTS RUN FROM 0 AND SCRATCH VECTOR GAM VARIABLE DIMEN.

      SUBROUTINE TRIDAG ( A, B, C, D, GAM, X, N )

      IMPLICIT  DOUBLE PRECISION ( A-H, K-L, O-Z )
      DIMENSION A(0:*), B(0:*), C(0:*), D(0:*), GAM(0:*), X(0:*)

      BET  = B(0)
      X(0) = D(0) / BET
      DO 10 J = 1, N
         GAM(J) = C(J-1) / BET
         BET    =  B(J) - A(J) * GAM(J)
         X(J)   = (D(J) - A(J) * X(J-1)) / BET
   10 CONTINUE

      DO 20 J = N-1, 0, -1
   20    X(J)   =  X(J) - GAM(J+1) * X(J+1)
      RETURN
      END
```

## 3. Useful subroutines

```
      SUBROUTINE  CALEN ( DATE, FACTOR, DAY )

C     Subroutine taking DATE in form ddmmyy and returning
C     FACTOR, which is number of days from some reference
C     date, and DAY, which is day of week (0-6 for Sat.-Fri.)
C     Note that below presumes year is 19yy.  You may change
C     way that date in input, but routine is only good for dates
C     later than the year 1582.

      IMPLICIT INTEGER (A-Z)

C     Disect DATE:
         ID = DATE
         DD = DATE / 10000
         ID = ID - 10000 * DD
         MM = ID / 100
         ID = ID - 100 * MM
         YY = 1900 + ID

C     Calculate FACTOR:
         FACTOR = 365 * YY + DD + 31 * ( MM - 1 )
         IF (MM .GT. 2) GO TO 20
            FACTOR = FACTOR + (YY-1)/4 - ( 3* ((YY-1)/100 + 1) ) / 4
```

```
           GO TO 30
 20  FACTOR = FACTOR - INT( 0.4 * SNGL(MM) + 2.3 ) + ( YY/4 )
     &           - ( 3 * (YY/100 + 1) ) / 4

C     Calculate DAY of week:
 30      DAY = FACTOR + 7 * (-FACTOR / 7)

      RETURN
      END


      SUBROUTINE NDTR(X,P,D)

C   Calculates cumulative normal distribution and density functions.
C   X = value for which CDF is calculated
C   P = return value of normal CDF
C   D = return value of normal density function
C   Error is less than 1.E-7  Cf. Abramowitz & Stegun p.932 Eq. 26.2.17

      IMPLICIT REAL*8 (A-H,O-Z)

      AX = DMIN1(1.0D20, DABS(X))
      T = 1.0D0/(1.0D0 + .2316419D0 * AX)
      D = 0.3989423D0 * DEXP(-AX*AX/2.0D0)
      P = 1.0D0 - D*T*((((1.330274D0 * T - 1.821256D0) * T
     &    + 1.781478D0) * T - 0.3565638D0) * T + 0.3193815D0)
      IF (X) 1,2,2
  1   P = 1.0D0 - P
  2   RETURN
      END


C**********************************************************************
C     SUBROUTINE INTRP1() PERFORMING CUBIC INTERPOLATION ON 1
C     DIMENSIONAL EQUALLY SPACED GRID.  RETURNS INTERPOLATED VALUE
C     AND FIRST AND SECOND PARTIAL DERIVATIVES IN Y, YX, YXX.
C     INTERPOLATION INSURES CONTINUOUS Y AND YX ACROSS GRIDPOINTS.
C     USES CUBIC EXTRAPOLATION IF BEYOND GRID RANGE (DANGEROUS!).
C
C     AUTHOR: R. JONES  3 FEB 1989
C**********************************************************************

      SUBROUTINE INTRP1(X,XMIN,XMAX,N,GRID,Y,YX,YXX)

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION GRID(0:N)

      H  = ( XMAX - XMIN ) / N
      XX = ( X    - XMIN ) / H
```

```
      NN = INT ( XX )
C     Next line insures all elements of GRID used are in range 0-N
      NN = MAX ( 1 , MIN ( N - 2, NN ) )
      ZZ = XX - NN

      D = GRID(NN)
      C = 0.5 * ( GRID(NN+1) - GRID(NN-1) )
      A = C + 2.0 * D - 2.0 * GRID(NN+1) + 0.5 * (GRID(NN+2) - GRID(NN))
      B = GRID(NN+1) - A - C - D

      Y   = (( A * ZZ + B ) * ZZ + C ) * ZZ + D
      YX  = (( 3.0 * A * ZZ + 2.0 * B ) * ZZ + C ) / H
      YXX = ( 6.0 * A * ZZ + 2.0 * B ) / ( H * H )

      RETURN
      END
```

## 4. Listing of program CIR calling CNSET to value bonds

```
C***********************************************************************
C
C   PROGRAM:  CALLS ONE FACTOR DIFFERENTIAL EQUATION SOLVING
C             SUBROUTINE CNSET & CNSTEP() FOR TESTING.  AS BELOW IT
C             VALUES INTEREST RATE CONTINGENT CLAIMS FOR ONE-FACTOR
C             COX-INGERSOLL-ROSS INTEREST RATE MODEL.
C
C   AUTHOR:   R. JONES       5 JANUARY 1989
C
C***********************************************************************

      IMPLICIT  REAL*8 (A-H,K-L,O-Z)
      DIMENSION UMTX(101), UEXMTX(101), ARR(101,4), PARM(15)

C**** SET MODEL PARAMETERS *********************************************

C     THE UNIT OF TIME IS 1 YEAR FOR THESE PARAMETER VALUES

C     PARAMETERS OF THE INTEREST RATE MODEL
      KAPR  =  1.76D0
      LAMR  =  -.32D0
      RBAR  =   .10D0
      SIGR  =   .21D0
      ARG1  =  0.50D0

      PARM(1) = SIGR
      PARM(2) = KAPR
```

```
      PARM(3) = LAMR
      PARM(4) = RBAR
      PARM(5) = ARG1

C     PARAMETERS OF THE PDE SOLVING ALORITHM
      N     =  21
      IMIN  =   0
      IMAX  =   0
      RMIN  =  0.00D0
      RMAX  =  0.40D0
      K     =  0.02
      EPS   =  1.0D-8

C     PARAMETERS IDENTIFYING THE TYPE OF SECURITY BEING VALUED

 100  PRINT*, ' ENTER MATURITY IN YEARS OR NEGATIVE NUMBER TO QUIT: '
      READ *,   TMAX
      IF  ( TMAX .LE. 0.0D0 )  GO TO 200
      ICOUP  = 0
      IAMER  = 0
      TCOUP  = 0.0D0
      DTCOUP = 0.0D0
      PRINT*, ' ENTER 0 FOR NEW BOND, 1 FOR FUTURES, 2 FOR CALL OPT.'
      READ *,   IFUTUR
         IF ( IFUTUR .EQ. 0 ) THEN
            PRINT*, ' ENTER COUPON PAYMENT INTERVAL IN YEARS'
            READ *,   DTCOUP
            PRINT*, ' ENTER SIZE OF COUPON PAYMENT'
            READ *,   COUP
            IF ( COUP .NE. 0.0D0 ) ICOUP = 1
C           SET MATURITY VALUE OF BOND TO 100
            CALL COPYC(100.0D0,UMTX,N,1)
            GO TO 150
         ELSE IF (IFUTUR .EQ. 1) THEN
            GO TO 150
         ELSE IF (IFUTUR .EQ. 2) THEN
            PRINT*, ' ENTER 0 FOR EUROPEAN OPTION, 1 FOR AMERICAN'
            READ *,   IAMER
            PRINT*, ' ENTER EXERCISE PRICE'
            READ *,   XPRICE
            DO 120 I = 1, N
               UMTX(I)   = MAX ( 0.0D0, UMTX(I) - XPRICE )
 120           UEXMTX(I) = UMTX(I)
            GO TO 150
         ELSE
            GO TO 100
         ENDIF

C**** SETUP TO SOLVE PDE ************************************************
```

```
C     NEXT LINE SYNCHRONIZES PDE TIME STEP WITH COUPON PAYMENT DATES
 150  IF ( ICOUP .EQ. 1 .AND. DTCOUP .GT. 0.0 )
     &      K = DTCOUP / DNINT ( DTCOUP/K )

      CALL CNSET (N-1,RMIN,RMAX,K,IFN,IFUTUR,IMIN,IMAX,PARM,ARR)

C     INITIALIZE TIME LEFT TO MATURITY AND GO SEE IF COUPON AT MATURITY
      T  =  0.0D0
      GO TO 850

C**** SOLUTION LOOP TO SOLVE PDE **************************************
 500  CONTINUE

C     CHECK IF FINISHED, OTHERWISE INCREMENT TIME TO MATURITY
      IF ( T .GE. TMAX - EPS )  GO TO 999
         T  =  T + K

         CALL CNSTEP ( T, UMTX, ARR )

C******* CHECK FOR EXERCISE OF AMERICAN OPTION *************************

         IF ( IAMER .EQ. 1 )  THEN
            DO 800  I = 1, N
 800        UMTX(I) = MAX ( UMTX(I), UEXMTX(I) )
         ENDIF

C******* CHECK FOR COUPON PAYMENT **************************************

 850     IF (ICOUP .EQ. 0 .OR. DTCOUP .EQ. 0.0 )   GO TO 930
            IF ( T + EPS .GE. TMAX ) GO TO 930
            IF ( MOD ( T - TCOUP + EPS, DTCOUP ) .LT. K )  THEN
               DO 900  I = 1, N
 900           UMTX(I) = UMTX(I) + COUP
            ENDIF
 930     CONTINUE

      GO TO 500
C**** END OF LOOP FOR ONE TIME STEP ***********************************

 999  PRINT*
      PRINT*, ' LEVEL OF R       SECURITY PRICE'
      DO 950 I = 1, 20
        R = RMIN + (RMAX - RMIN) * (DBLE(I-1) / DBLE(N-1))
        R = 100 * R
        WRITE  (*, 9000)  R, UMTX(I)
 950  CONTINUE
      PRINT*
9000  FORMAT (F8.2, 10X, F10.4)
```

```
      GO TO 100

 200  STOP
      END

C*********************************************************************
C**** UTILITY SUBROUTINES CALLED **************************************

      SUBROUTINE COPYC( X, U, IM, IN )
      IMPLICIT DOUBLE PRECISION (A-H, K-L, O-Z)
      DIMENSION U(IM,*)
      DO 100 I = 1, IM
         DO 100 J = 1, IN
 100     U(I,J) = X
      RETURN
      END

C**** FUNCTIONS CALLED BY CNSET AND CNSTEP ****************************
C
C     FMIN(T) and FMAX(T) give boundary values at rmin and rmax if known
C     They are called only if flags ISMIN, ISMAX are set to 1. Otherwise
C     quadratic extrapolation at these boundaries is used (i.e., Usss=0)
C
C     Coefficients set here for one factor interest rate models.  When
C     parameter ARG1 = 0.5 it is the CIR one factor model.
C
C     The coefficients for a one factor version of the JJ log model are
C     FNA(R)  = SIGR * SIGR * R * R / 2.0
C     FNB(R)  = KAPR * R * ( LOG( RBAR ) - LOG( MAX( 1.D-20, R ) ) )
C     FNC(R)  = -R
C
C     The coefficients for the Black-Scholes option pricing model, with
C     S being the stock price, and R and D the assumed constant interest
C     rate and proportional dividend rate respectively, are
C     FNA(S)  = SIGS * SIGS * S * S / 2.0
C     FNB(S)  = ( R - D ) * S
C     FNC(S)  = -R
C
C*********************************************************************

      DOUBLE PRECISION FUNCTION COEFF()

      IMPLICIT  DOUBLE PRECISION (A-H,K-L,O-Z)
      DIMENSION PARM(15)

      ENTRY FNA(R,IFN,PARM)
         SIGR   = PARM(1)
         ARG1   = PARM(5)
```

```
         FNA    = SIGR * SIGR * R ** ( ARG1 * 2.0 ) / 2.0
      RETURN

      ENTRY FNB(R,IFN,PARM)
         KAPR   = PARM(2)
         LAMR   = PARM(3)
         RBAR   = PARM(4)
         ARG1   = PARM(5)
         FNB    = KAPR * ( RBAR - R ) - LAMR * R ** ( 0.5 + ARG1 )
      RETURN

      ENTRY FNC(R,IFN,PARM)
         FNC    = -R
      RETURN

      ENTRY FMIN(T)
         FMIN    = 1.0
      RETURN

      ENTRY FMAX(T)
         FMAX    = 0.0
      RETURN
      END
```

## Listing of program applying two factor pde routine ADISET

```
C************************************************************************
C       Sample program to generate equilibrium bond yields using
C       calls to ADISET and ADSTEP.  Two factor interest rate model
C       is Jacobs/Jones log model.
C************************************************************************

      IMPLICIT DOUBLE PRECISION (A-H, K-L, O-Z)
      PARAMETER ( IM =   20, IN =   20 )
      PARAMETER ( IM1= IM+1, IN1= IN+1 )
      DIMENSION   PARM(15), INFO(8)
      DIMENSION   U(0:IM, 0:IN), V(0:IM, 0:IN), ARR(8, 0:IM, 0:IN),
     &            C(0:IM, 0:IN)

C  Algorithm parmeters:
      DATA INFO / 0, 0, 0, 0, 0, 0, 0, 0 /
      DATA LMN, LMX, RMN, RMX / .00, .30, .00, .30 /
      DATA IFUT, KK  / 0, .10 /

C  Model params:  SIGR  SIGL  RHO  KAPR  KAPL  LAMR  LAML  LBAR
      DATA PARM / .72,  .19, -.28, 2.60, .064, -.23,-.008, .085, 7*0. /
```

```
      HL = ( LMX - LMN ) / IM
      HR = ( RMX - RMN ) / IN


C  Enter desired time to maturity from console:

 100  PRINT*, ' Enter maturity and coupon interval in years:'
      READ *, TMAX, DTCOUP
      IF ( TMAX .LE. 0.0 ) STOP

      CALL COPYC ( 1.0D0, U, IM1, IN1 )
      CALL COPYC ( 0.0D0, C, IM1, IN1 )

C     Adjust KK to be integer fraction of DTCOUP
      KK = DTCOUP / MAX ( 1.0D0, DNINT ( DTCOUP/KK ) )

      CALL ADISET(IM,IN,IMU,LMN,LMX,RMN,RMX,KK,IFN,IFUT,INFO,PARM,ARR)

      DO 200 T = 0.0D0, TMAX-KK/2, KK
C        Pay coupon if time has come
         IF ( MOD ( T+KK/2, DTCOUP ) .LT. KK ) THEN
            CALL ADDC( 1.0D0, C, IM1, IN1 )
         ENDIF
C        Step back KK
         CALL ADSTEP ( U, V, ARR )
         CALL ADSTEP ( C, V, ARR )
 200  CONTINUE

C     Calculate equilibrium coupon rates and store in U()
      DO 300 I = 0, IM
         DO 300 J = 0, IN
         COUPON = (1.0 - U(I,J)) / ( C(I,J) * DTCOUP )
         U(I,J) = COUPON * 100
 300  CONTINUE

C     Print out table of results (10 by 10)
      PRINT*
      PRINT 9010, TMAX
9010  FORMAT(' Equilbrium coupon rates on bonds of maturity ',F6.3)

      WRITE(*,9000) ( (RMN + I*HR), I = 0, IN, IN/10 )
9000  FORMAT('   R:  ', 12(1X,F5.3) )
      PRINT*,' L:'

      DO 400 I = 0, IM, IM/10
 400     WRITE(*,9020) (LMN + I*HL), (U(I,J), J=0, IM, IM/10)
9020  FORMAT(1X,F5.3,2X,12(1X,F5.2) )
```

```
      GO TO 100
      END

C**********************************************************************
C**** UTILITY SUBROUTINES CALLED ***************************************

      SUBROUTINE COPYC( X, U, IM, IN )
      IMPLICIT DOUBLE PRECISION (A-H, K-L, O-Z)
      DIMENSION U(IM,*)
      DO 100 I = 1, IM
         DO 100 J = 1, IN
 100     U(I,J) = X
      RETURN
      END

      SUBROUTINE ADDC( X, U, IM, IN )
      IMPLICIT DOUBLE PRECISION (A-H, K-L, O-Z)
      DIMENSION U(IM,*)
      DO 100 I = 1, IM
         DO 100 J = 1, IN
 100     U(I,J) = U(I,J) + X
      RETURN
      END

C *********************************************************************
C     DEFINE FUNCTIONS WHICH WILL BE USED TO GENERATE THE
C     COEFFICIENTS IN PDE .
C
C     A*ULL + B*URR + C*ULR + D*UL + E*UR + F*U + COUP(T) + UT = 0
C *********************************************************************

      DOUBLE PRECISION FUNCTION COEFFX()
      IMPLICIT DOUBLE PRECISION (A-H,J-Z)
      IMPLICIT INTEGER(I)
      DIMENSION PARM(*)

      ENTRY FNAKA(IFN,L,R,PARM)
          SIGL = PARM(2)
          FNAKA= SIGL*SIGL*L*L/2.0D0
          RETURN

      ENTRY FNAKB(IFN,L,R,PARM)
          SIGR = PARM(1)
          FNAKB= SIGR*SIGR*R*R/2.0D0
          RETURN

      ENTRY FNAKC(IFN,L,R,PARM)
          SIGR = PARM(1)
          SIGL = PARM(2)
```

```
          RHO  = PARM(3)
          FNAKC= RHO*SIGR*SIGL*R*L
          RETURN

      ENTRY FNAKD(IFN,L,R,PARM)
          KAPL = PARM(5)
          LAML = PARM(7)
          LBAR = PARM(8)
          IF ( L .GT. 0.0 ) THEN
             FNAKD = -KAPL * L * LOG(L/LBAR) - LAML * SQRT(R) * L
          ELSE
             FNAKD = 0.0
          ENDIF
          RETURN

      ENTRY FNAKE(IFN,L,R,PARM)
          SIGR = PARM(1)
          KAPR = PARM(4)
          KAPL = PARM(5)
          LAMR = PARM(6)
          LBAR = PARM(8)
          IF ( R .LE. 0.0 ) THEN
             FNAKE = 0.0
             RETURN
          ENDIF
          IF ( L .GT. 0.0 ) THEN
             G = LOG(L)
             ELSE
             G = -1.0D60
          ENDIF
          FNAKE= KAPR * R * (G - DLOG(R)) - LAMR * SQRT(R) * R
          RETURN

      ENTRY FNAKF(IFN,L,R,PARM)
          FNAKF= -R
          RETURN
      END

C**** FUNCTIONS CALLED BY ADSTEP GIVING BOUNDARY VALUES IF KNOWN *******

      DOUBLE PRECISION FUNCTION CXXXF()
      IMPLICIT DOUBLE PRECISION (A-H,K-L,O-Z)
      ENTRY FNULMN(R)
         FNULMN = 0.0
      RETURN
      ENTRY FNULMX(R)
         FNULMX = 0.0
      RETURN
      ENTRY FNURMN(L)
```

```
         FNURMN = 0.0
      RETURN
      ENTRY FNURMX(L)
         FNURMX = 0.0
      RETURN
      ENTRY FNLNRN
         FNLNRN = 0.0
      RETURN
      ENTRY FNLXRN
         FNLXRN = 0.0
      RETURN
      ENTRY FNLNRX
         FNLNRX = 0.0
      RETURN
      ENTRY FNLXRX
         FNLXRX = 0.0
      RETURN
      END
```

## Listing of MONTE CARLO program valuing put options

```
C***********************************************************************
C                                                                      *
C   PROGRAM:  SAMPLE 6: STRIPPED DOWN PROGRAM USING MONTE CARLO METHOD *
C             TO VALUE PUT OPTIONS ON A NON-DIVIDEND PAYING SECURITY.  *
C                                                                      *
C***********************************************************************

      IMPLICIT  DOUBLE PRECISION (A-H,K-L,O-Z)
      INTEGER   SEED, MULT, MODLUS
      PARAMETER ( N = 51 )
      DIMENSION U(N), DIS(N), PARM(15), INO(N), IUP(N)

C**** SET MODEL PARAMETERS *********************************************

C     Constants needed for Random Number Generator:

      SEED   = 1973272912
      MULT   = 65539
      MODLUS = 2147483647
      RNMAX  = DBLE(2 ** 31 - 1)

C     Grid parameters:

      DATA SMIN, SMAX, K / 8.0, 108.0, .025 /
      H = ( SMAX - SMIN ) / ( N - 1 )
```

```
C     Financial model parameters:

      DATA R, SIGMA, XPRICE / .10, .20, 50.0 /
      PARM(1) = SIGMA
      PARM(2) = R

C**** Setup transition probability arrays ******************************

 50   DO 100 I = 1, N
         S    = SMIN + ( SMAX - SMIN ) * DBLE(I-1) / DBLE(N-1)
         A    = FNA(S,IFN,PARM)
         B    = FNB(S,IFN,PARM)
         C    = FNC(S,IFN,PARM)

C        Probabilities of moves UP, DOWN, or NO change.
         PRUP = K * (2 * A + H * B) / ( 2 * H**2 * (1 + K * C ))
         PRDN = K * (2 * A - H * B) / ( 2 * H**2 * (1 + K * C ))

C           If any negative probabilities send message
            IF ( PRUP .LT. 0.0 .OR. PRDN .LT. 0.0 ) THEN
               PRINT*, ' *** Need finer state grid ***'
               STOP
            ENDIF
            IF ( 1.0 - PRUP - PRDN .LT. 0.0 ) THEN
               K = K / 2
               PRINT*, ' Changing time step to ', K
               GO TO 50
            ENDIF

C           Adjust probabilities at boundaries to stay within them
            IF ( I .EQ. 1 ) PRDN = 0.0
            IF ( I .EQ. N ) PRUP = 0.0

         PRNO = 1.0 - PRUP - PRDN

C        Let's store the critical random numbers as integers for speed
         INO(I) = INT ( PRNO * RNMAX )
         IUP(I) = INT ( (PRNO + PRUP) * RNMAX )

C        This line stores discount factor for state I in DIS(I)
         DIS(I) = 1 + K * C

C        While here, let us also store terminal option payoff in U()
         U(I) = MAX ( 0.0D0 , XPRICE - S )
100   CONTINUE

C**** Generate random walks and accumulate outcomes *******************

150   PRINT*, ' Enter maturity (yrs), sample size, current stock price:'
```

```
      READ *, TMAT, NWALKS, S0
      IF ( TMAT .LE. 0.0 ) STOP

      I0    = 1 + NINT ( (S0 - SMIN) / H )
      S0    = SMIN + H * ( I0 - 1 )
      TOTAL = 0.0
      TOT2  = 0.0

      DO 250 J = 1, NWALKS
         I      = I0
         DISFAC = 1.0

C        Take one random walk out to TMAT
         DO 200 T = 0.0D0, TMAT - K/2, K

C           Accumulate discount factor
            DISFAC = DISFAC * DIS(I)

C           Generate next random integer from the last
            SEED = SEED * MULT
            IF ( SEED .LT. 0 ) SEED = SEED + MODLUS + 1

C           See which way the state changed
            IF ( SEED .GT. INO(I) ) THEN
               IF ( SEED .LE. IUP(I) ) THEN
                  I = I + 1
                  ELSE
                  I = I - 1
               ENDIF
            ENDIF
200      CONTINUE

C        Determine payoff for this walk and accumulate result
         PAY   = DISFAC * U(I)
         TOTAL = TOTAL + PAY
         TOT2  = TOT2  + PAY * PAY
250   CONTINUE

C     Calculate average payoff and standard deviation
      VALUE = TOTAL / NWALKS
      STDEV = SQRT ( TOT2 - NWALKS * VALUE ** 2 ) / NWALKS

C**** Print results to screen ******************************************

      PRINT*
      PRINT '('' Initial stock P: '',F8.3)', S0
      PRINT '('' Option maturity: '',F8.3)', TMAT
      PRINT '('' Estimated value: '',F8.3)', VALUE
      PRINT '('' Standard deviat: '',F8.3)', STDEV
```

```
      PRINT*
      GO TO 150

      END

C**** FUNCTION DEFINITIONS REQUIRED ***********************************
C     The coefficients for the Black-Scholes option pricing model, with
C     S being the stock price, and R and D the assumed constant interest
C     rate and proportional dividend rate respectively, are            *
C     FNA()  = SIGS * SIGS * S * S / 2.0                               *
C     FNB()  = (R - D) * S                                             *
C     FNC()  = -R                                                      *
C**********************************************************************

      DOUBLE PRECISION FUNCTION COEFF()

      IMPLICIT  DOUBLE PRECISION (A-H,K-L,O-Z)
      DIMENSION PARM(15)

      ENTRY FNA(S,IFN,PARM)
         SIGMA  = PARM(1)
         FNA    = (SIGMA * S) ** 2 / 2.0
      RETURN

      ENTRY FNB(S,IFN,PARM)
         R      = PARM(2)
         FNB    = R * S
      RETURN

      ENTRY FNC(S,IFN,PARM)
         R      = PARM(2)
         FNC    = -R
      RETURN
      END
```

## 5. Listing of MINE program valuing operating options

```
C**********************************************************************
C
C   PROGRAM:  Calls one factor differential equation solving
C             subrouinte CNSET & CNSTEP().  As below it
C             determines optimal operating policy and value of a
C             mine with finite life and operating rate but unlimited
C             reserves. Based on Palm, Pearson and Read, "OPTION PRICING:
C             A NEW APPROACH TO MINE VALUATION," CIM Bulletin (May '86).
C             Shut-down, re-open, abandon options as in Brennan &
C             Schwartz are implemented.  Convenience yield on commodity
```

```
C            inventories implicit in futures market can be input.
C
C   AUTHOR:   R. A. Jones      10 January 1989
C
C*********************************************************************

      IMPLICIT  DOUBLE PRECISION (A-H,K-L,O-Z)
      PARAMETER ( EPS = 1.0D-8, N = 101 )
      DIMENSION UOPEN(N), UCLOS(N), ARR(N,4), PARM(15), P(N)

C**** SET MODEL PARAMETERS *********************************************

C     THE UNIT OF TIME IS 1 YEAR FOR THESE PARAMETER VALUES

C     PARAMETERS OF THE PDE SOLVING ALORITHM
      DATA IMIN, IMAX, IFUT, PMIN, PMAX, K / 0, 0, 0, 0., 2000., 0.10

      PSTEP = ( PMAX - PMIN ) / DBLE( N - 1 )
      DO 2 I = 1, N
   2    P(I) = PMIN + PSTEP * DBLE( I - 1 )

C     PARAMETERS DESCRIBING OUTPUT PRICE MODEL AND MINE CHARACTERISTICS
C            OUTPUT PRICE MODEL: dP = mu(.) P dt + SIG P dz

 100  PRINT*, ' Enter life of mine in years or zero to quit: '
      READ *,   TMAX
      IF  ( TMAX .LE. 0.0D0 )  GO TO 999
      PRINT*, ' Enter sigma, interest rate, convenience yld/year:'
      READ *,   SIG, R, CYLD
C     Next 3 lines to prevent 0-divide error if 0 values entered
         R    =  R    + EPS
         SIG  =  SIG  + EPS
         CYLD =  CYLD + EPS
      PRINT*, ' Enter units/year output, marg. cost/unit,',
     &        ' fixed costs/year:'
      READ *,   Q, C, F
      PRINT*, ' Enter shutdown cost, reopen cost, scrap value:'
      READ *,   K1, K2, S

C*********************************************************************
C**** SETUP TO SOLVE COUPLED PDE'S AND SET TERMINAL VALUES *************

      PARM(1) = SIG
      PARM(2) = CYLD
      PARM(3) = R
      K       =  TMAX / MAX ( 1.0D0, DNINT( TMAX / K ) )
      CALL CNSET (N-1,PMIN,PMAX,K,IFN,IFUT,IMIN,IMAX,PARM,ARR)
      CALL COPYC ( S-K1, UOPEN, N, 1 )
      CALL COPYC (    S, UCLOS, N, 1 )
```

```
      T       = 0.0

C**** SOLUTION LOOP TO SOLVE PDE ***************************************

 200  CONTINUE

C******* PAY OUT OPERATING INCOME FOR LATEST PERIOD WITH INTEREST ******

         DO 300  I = 1, N
            UOPEN(I) = UOPEN(I) + K * ( (P(I)-C) * Q - F )
            UCLOS(I) = UCLOS(I) + K * ( - F )
 300     CONTINUE

C******* STEP BACK ONE STEP ********************************************

         CALL CNSTEP ( T, UOPEN, ARR )
         CALL CNSTEP ( T, UCLOS, ARR )

         T  =  T + K

C******* CHECK IF OPTIMAL TO CHANGE MINE OPERATING STATUS **************

         IF ( T + EPS .GE. TMAX ) THEN
C           DETERMINE CRITICAL SWITCHING PRICES IF AT TMAX
C           THIS MUST BE DONE BEFORE DOING MAXIMIZATION OF LOOP 400
            I = 1
            CSCRAP = 0.0
            OCLOS  = 0.0
            COPEN  = 0.0
  10        IF ( I .GT. N ) GO TO 15
            IF ( S .GT. UCLOS(I) ) THEN
               I = I + 1
               GO TO 10
               ELSE
               IF ( I .EQ. 1 ) GO TO 15
               X = ( S - UCLOS(I-1) ) / ( UCLOS(I) - UCLOS(I-1) )
               CSCRAP = P(I-1) + X * PSTEP
            ENDIF
  15        I = 1
  20        IF ( I .GT. N ) GO TO 35
            IF ( UCLOS(I) - K1 .GT. UOPEN(I) ) THEN
               I = I + 1
               GO TO 20
               ELSE
               IF ( I .EQ. 1 ) GO TO 30
               X = ( UCLOS(I-1) - K1 - UOPEN(I-1) ) /
     &             ( UCLOS(I-1) - UOPEN(I-1) - UCLOS(I) + UOPEN(I) )
               OCLOS = P(I-1) + X * PSTEP
            ENDIF
```

```
  30       IF ( I .GT. N ) GO TO 35
           IF ( UCLOS(I) .GT. UOPEN(I) - K2 ) THEN
              I = I + 1
              GO TO 30
              ELSE
              IF ( I .EQ. 1 ) GO TO 35
              X = ( UCLOS(I-1) - UOPEN(I-1) + K2 ) /
     &            ( UCLOS(I-1) - UOPEN(I-1) - UCLOS(I) + UOPEN(I) )
              COPEN  = P(I-1) + X * PSTEP
           ENDIF
  35    ENDIF

        DO 400 I = 1, N
           UCLOS(I) = MAX ( S      , UCLOS(I)     , UOPEN(I) - K2 )
           UOPEN(I) = MAX ( S - K1 , UCLOS(I) - K1, UOPEN(I)      )
 400    CONTINUE

C******* CHECK IF FINISHED -- LOOP BACK IF NOT *************************

      IF ( T + EPS .LT. TMAX )  GO TO 200

C**** END OF LOOP *****************************************************
C**********************************************************************

C     PRINT OUT RESULTS:
      WRITE (*,9020) SIG,R,CYLD,C,Q,F,S,K1,K2
      WRITE (*,9010) CSCRAP, OCLOS, COPEN
      PRINT*
      PRINT*, ' P LEVEL   MINE VAL OPEN   VAL CLOSED   ',
     &         'OPTION VAL    HEDGE RATIO'
      DO 900 I = 6, 86, 5
C       CALCULATE VALUE OF MINE FIXED OPEN FROM KNOWN PDE SOLUTION.
C       NOTE DISCOUNTING OF FLOW COSTS BY EXTRA K/2 TO MATCH TIMING ABOVE

        VAL = - ( Q * C + F ) * (1.0 - EXP(-R*T )) * EXP(-R*K/2)  / R
     &        + Q * P(I) * (1.0 - EXP(-CYLD*T )) * EXP(-CYLD*K/2) / CYLD
     &        + EXP(-R*T ) * ( S - K1 )
        VAL = UOPEN(I) - VAL
        HEDGE = ( UOPEN(I+1) - UOPEN(I-1) ) / ( 2 * PSTEP )
        WRITE  (*, 9000)  P(I), UOPEN(I), UCLOS(I), VAL, HEDGE
 900  CONTINUE
      PRINT*

9000  FORMAT (F8.2, 5X, F10.2, 3X, F10.2, 3X, F10.2, 3X, F9.2)
9010  FORMAT(' CRITICAL PRICES:  SCRAP:',F8.2,'  CLOSE:',F8.2,
     &       '  OPEN:',F8.2)
9020  FORMAT(' PARAMETERS: SIG:',
     &      F5.2,' R:',F5.2,'  CYLD:',F5.2,'     C:',F8.2,
     &      /' Q:',F8.2,' F:',F8.2,' S:',F8.2,' K1:',F8.2,' K2:',F8.2)
```

```
      GO TO 100

 999  STOP
      END

C*********************************************************************
C**** UTILITY SUBROUTINES CALLED **************************************

      SUBROUTINE COPYC( X, U, IM, IN )
      IMPLICIT DOUBLE PRECISION (A-H, K-L, O-Z)
      DIMENSION U(IM,*)
      DO 100 I = 1, IM
         DO 100 J = 1, IN
 100     U(I,J) = X
      RETURN
      END

C**** COEFFICIENT FUNCTIONS FOR VALUATION PDE *************************

      DOUBLE PRECISION FUNCTION COEFF()

      IMPLICIT  DOUBLE PRECISION (A-H,K-L,O-Z)
      DIMENSION PARM(15)

      ENTRY FNA(P,IFN,PARM)
         SIG    = PARM(1)
         FNA    = SIG * SIG * P * P / 2.0
      RETURN

      ENTRY FNB(P,IFN,PARM)
         CYLD   = PARM(2)
         R      = PARM(3)
         FNB    = ( R - CYLD ) * P
      RETURN

      ENTRY FNC(P,IFN,PARM)
         R      = PARM(3)
         FNC    = -R
      RETURN

      ENTRY FMIN(T,IFN,PARM)
         FMIN   = 0.0
      RETURN

      ENTRY FMAX(T,IFN,PARM)
         FMAX   = 0.0
      RETURN
      END
```