APÉNDICE A

```
C
      PROGRAM xamoeba
C
C     DRIVER FOR ROUTINE AMOEBA
C
C  ESTE ES UN PROGRAMA PRINCIPAL QUE SIRVE COMO EJEMPLO
C  PARA EL USO DE LOS PROGRAMAS DE
C  OPTIMIZACIÓN UTILIZANDO EL ALGORITMO DE OPTIMIZACIÓN
C  NELDER MEAD QUE TIENE
C  EL LIBRO: NUMERICAL RECIPES IN C: THE ART OF SCIENTIFIC
C  COMPUTING.(1992).PRESS,W., TEUKOLSKY, S., VETTERLING, W. Y
C  FLANNERY, B. CAMBRIDGE UNIVERSITY PRESS. NEW YORK, U.S.A.
C  (QA297/N8.4/1994).
C
C  LA FUNCIÓN A OPTIMIZAR DEBE METERSE EN LA FUNCIÓN famoeb,
C  LA CUAL ES DECLARADA COMO REAL. VER UNAS LINEAS ABAJO LA
C  FUNCIÓN BAJO ESE NOMBRE.
C
C  TAMBIÉN EN LAS NOTAS TENGO UNAS COPIAS DEL ALGORITMO
C  CORRESPONDIENTE. LAS PÁGINAS QUE CORRESPONDEN AL LIBRO
C  MENCIONADO SON LAS 408 A 412.
C
C
C
C
C
C  ESTAS SON LAS DEFINICIONES DE LAS VARIABLES PROPIAS
C  DEL PROBLEMA DE LA HOCKEY STICK.
C
      REAL rdatos(1500)
      INTEGER ndatos
      REAL rlambda,ta,rm,c
      COMMON rdatos,ndatos
      CHARACTER*12 ENTRADA
      REAL rsss
C
C
C PARAMETROS
C
C LA MATRIZ P TIENE EN LAS FILAS LOS PARAMETROS
C Y EN LAS COLUMNAS EL SIMPLEX. P ES LA MATRIZ DE
C SALIDA.
C NP=NUMERO DE PARAMETROS.
C MP=NP+1
C FTOL=TOLERANCIA
C
      INTEGER np,mp
      REAL ftol
      PARAMETER(np=3,mp=4,ftol=1.0e-6)
      INTEGER i,iter,j,ndim
      REAL famoeb,p(mp,np),x(np),y(mp)
      EXTERNAL famoeb
      DATA p/100.,105.,100.,100.,
     1       .01,.01,.012,.01,
     1       2.0,2.0,2.0,2.5/
C
C  SI DATA p/0,1,0,0,0,0,1,0,0,0,0,1/
```

```
C
C  ENTONCES LA MATRIZ p TIENE LA FORMA:  0 0 0
C                                       1 0 0
C                                       0 1 0
C                                       0 0 1
C  SI DATA p/0,1,0,0/   p=0 0
C                         1 0
C NUMERO DE DIMENSIONES:
C
      ndim=np
C
C
C  ARCHIVOS DE ENTRADA Y SALIDA
C
C
C
C  EN ESTA SECCION SE SOLICITAN LOS NOMBRES DE LOS ARCHIVOS
C  DE ENTRADA Y SALIDA. TAMBIEN SE PONE UN MENSAJE INICIAL
C  EN EL ARCHIVO DE SALIDA.
C
      print*,' '
      print*,'Introduce el nombre del archivo de entrada (Input)'
      read (*,1) entrada
    1 format (a12)
      open (3,file=entrada,status='old')
C
C SE LEEN EL NUMERO DE OBSERVACIONES Y LAS OBSERVACIONES SE METEN
C EN EL VECTOR rdatos.
C
      PRINT*,' '
      PRINT*,' INTRODUCE EL NUMERO DE DATOS'
      READ*,ndatos
      PRINT*,' '
      PRINT*,' INTRODUCE LA SEMILLA'
      READ*,rsss
      DO 344 I=1,ndatos
      READ(3,*)rdatos(I)
  344 CONTINUE

C
C SE EVALUA LA FUNCION EN LOS PUNTOS DADOS EN LA MATRIZ
C p, DONDE SE ALMACENARON LOS VERTICES. ESTAS EVALUACIONES
C SE GUARDAN EN EL VECTOR Y.
C
      DO 12 I=1,mp
        DO 11 J=1,np
          x(j)=p(i,j)
11      CONTINUE
        y(i)=famoeb(x)
12    CONTINUE
C
C
C AQUI SE HACE LA LLAMADA A LA SUBRUTINA amoeba.
C
      CALL amoeba(p,y,mp,np,ndim,ftol,famoeb,iter)
C
C
```

```
C UNA VEZ OPTIMIZADA LA FUNCIÓN SE IMPRIMEN LOS RESULTADOS.
C
      Ta=p(1,1)
      RLAMBDA=p(1,2)
      RM=p(1,3)
      C=(RM*TA**(RM-1.)/RLAMBDA)**(1./RM)
      FVALUE=-Y(1)
      PRINT*,' '
      PRINT*,' '


      PRINT*,'      TA=',TA
      PRINT*,'     RM=',RM
      PRINT*,' RLAMBDA=',RLAMBDA
      PRINT*,'      C=',C
      PRINT*,' FUNCION=',FVALUE
C
C
C  SE DECLARAN LOS PUNTOS DE LA NUEVA MATRIZ
C
       P(1,1)=TA
       P(2,1)=TA*1.001
       P(3,1)=TA
       P(4,1)=TA
       P(1,2)=RLAMBDA
       P(2,2)=RLAMBDA
       P(3,2)=RLAMBDA*1.001
       P(4,2)=RLAMBDA
       P(1,3)=RM
       P(2,3)=RM
       P(3,3)=RM
       P(4,3)=RM*1.001

      SUMTAS=0
      SUM2TAS=0
      SUMRLS=0
      SUM2RLS=0
      SUMRMS=0
      SUM2RMS=0
      SUMCS=0
      SUM2CS=0
C
C CALCULO DE LA FUNCION DE DISTRIBUCION ACUMULATIVA
C EN EL PUNTO DE CORTE TA.(FTAS)
C
      FTAS=1-EXP(-RLAMBDA*TA)
C
C
      DO 388 IXXXXX=1,1000
C
C  SE GENERA LA MUESTRA BOOTSTRAP PARAMETRICA DE
C  f(X/TETA TILDE)
C
      DO 389 I=1,ndatos
  222 CONTINUE
         CALL random(rsss)
         IF(rsss.le.0.01.or.rsss.ge.0.99)go to 222
```

```
        yu=rsss
        IF(YU.LE.FTAS)rdatos(I)=-LOG(1.-YU)/RLAMBDA
        IF(YU.GT.FTAS)rdatos(I)=C*(-LOG(1.-YU)-
     1                    RLAMBDA*TA*(RM-1.)/RM)**(1./RM)
 389  CONTINUE
C
C LA FUNCION EN LOS VERTICES
C
      DO 386 I=1,MP
      DO 385 J=1, NP
        X(J)=P(I,J)

 385  CONTINUE

        Y(I)=FAMOEB(X)

 386  CONTINUE
C
C   SE LLMA LA FUNCION amoeba
C
      CALL amoeba(p,y,mp,np,ndim,ftol,famoeb,iter)
C
C YA OPTIMIZADA LA FUNCION
C
      TAS=P(1,1)
      RLAMBDAS=P(1,2)
      RMS=P(1,3)
      CS=(RMS*TAS**(RMS-1.)/RLAMBDAS)**(1./RMS)

C
C  ACUMULANDO VALORES PARA EL CALCULO DE LA VARIANZA
C
      SUMTAS=SUMTAS+TAS
      SUM2TAS=SUM2TAS+TAS*TAS
      SUMRMS=SUMRMS+RMS
      SUM2RMS=SUM2RMS+RMS*RMS
      SUMRLS=SUMRLS+RLAMBDAS
      SUM2RLS=SUM2RLS+RLAMBDAS*RLAMBDAS
      SUMCS=SUMCS+CS
      SUM2CS=SUM2CS+CS*CS
 388  CONTINUE
C
C   SE CALCULAN LAS VARIANZAS
C
      VARTA=(SUM2TAS-SUMTAS*SUMTAS/1000.)/999.
      IF(VARTA.LE.0.0)VARTA=0.0
      VARRL=(SUM2RLS-SUMRLS*SUMRLS/1000.)/999.
      IF(VARRL.LE.0.0)VARRL=0.0
      VARRM=(SUM2RMS-SUMRMS*SUMRMS/1000.)/999.
      IF(VARRM.LE.0.0)VARRM=0.0
      VARC=(SUM2CS-SUMCS*SUMCS/1000.)/999.
      IF(VARC.LE.0.0)VARC=0.0
      PRINT*,VARTA,VARRL,VARRM,VARC
C
C  CALCULO DE INTERVALOS DE CONFIANZA CON UN 95%
C
      CITA=TA-1.96*SQRT(VARTA)
```

```
      CSTA=TA+1.96*SQRT(VARTA)
      CIRL=RLAMBDA-1.96*SQRT(VARRL)
      CSRL=RLAMBDA+1.96*SQRT(VARRL)
      CIRM=RM-1.96*SQRT(VARRM)
      CSRM=RM+1.96*SQRT(VARRM)
      CICS=C-1.96*SQRT(VARC)
      CSCS=C+1.96*SQRT(VARC)
C
C IMPRESION DE LOS RESULTADOS
C
      PRINT*,'CITA',CITA
      PRINT*,'CSTA',CSTA
      PRINT*,'CIRL',CIRL
      PRINT*,'CSRL',CSRL
      PRINT*,'CIRM',CIRM
      PRINT*,'CSRM',CSRM
      PRINT*,'CIC',CICS
      PRINT*,'CSC',CSCS


C
C SE TERMINA LA EJECUCION DEL PROGRAMA
C
      STOP
      END
C
C

C
C
C AQUI VA LA SUBRUTINA PARA LA OPTIMIZACION.
C JUNTO CON LA FUNCION PARA OPTIMIZAR
C
C

C
C
      REAL FUNCTION famoeb(x)
      REAL rdatos(1500),x(3)
      INTEGER ndatos
      COMMON rdatos,ndatos
C
C
C     VECTOR QUE CONTIENE EL VALOR DE LOS PARAMETROS
C
      TA=X(1)
      RLAMBDA=X(2)
      RM=X(3)
      C=(RM*TA**(RM-1.)/RLAMBDA)**(1./RM)
C
C AQUI SE METEN LAS RESTRICCIONES QUE DEBEN HABER SOBRE LOS
C PARAMETROS.
C
      IF(TA.LE.0..OR.
     1   C.LE.0..OR.
     3  RM.LE.1..OR.
     6   RLAMBDA.LE.0.)GO TO 444
```

```
C
C     CALCULANDO EL VALOR DE LA FUNCIÓN DE MÁXIMA VEROSIMILITUD
C

      value=0.

      DO 55 i=1,ndatos
      DRD=RDATOS(I)
C
C
C     FUNCION DE MAXIMA VEROSIMILITUD PARA FALLAS ALEATORIAS
C

      IF(DRD.le.ta)term=log(rlambda)-rlambda*DRD
C
C
C     FUNCION DE MAXIMA VEROSIMILITUD PARA FALLAS TARDIAS
C

      IF(DRD.gt.tA)term=log(rm)+(rm-1)*LOG(DRD)-
     1rm*log(c)-rlambda*ta*(RM-1.)/RM-(DRD/c)**rm

      value=value+term

   55 continue
      VALUE=-VALUE
      GO TO 400
  444 VALUE=9D+10
C
C
C SE REGRESA AL PROGRAMA PRINCIPAL.
C
  400 CONTINUE
      FAMOEB=VALUE
C      PRINT*,'VALUE',VALUE
      RETURN
      END
C
C
C
C
C
C
C
C
      SUBROUTINE amoeba(p,y,mp,np,ndim,ftol,funk,iter)
      INTEGER iter,mp,ndim,np,NMAX,ITMAX
      REAL ftol,p(mp,np),y(mp),funk,TINY
      PARAMETER (NMAX=20,ITMAX=5000,TINY=1.e-10)
      EXTERNAL funk
C
C    USES amotry,funk
C
      INTEGER i,ihi,ilo,inhi,j,m,n
      REAL rtol,sum,swap,ysave,ytry,psum(NMAX),amotry
      iter=0
1     DO 12 n=1,ndim
```

```
        sum=0.
        DO 11 m=1,ndim+1
          sum=sum+p(m,n)
11      CONTINUE
        psum(n)=sum
12    CONTINUE
2     ilo=1
      IF (y(1).gt.y(2)) THEN
        ihi=1
        inhi=2
      ELSE
        ihi=2
        inhi=1
      ENDIF
      DO 13 i=1,ndim+1
        IF(y(i).le.y(ilo)) ilo=i
        IF(y(i).gt.y(ihi)) then
          inhi=ihi
          ihi=i
        ELSE IF(y(i).gt.y(inhi)) THEN
          if(i.ne.ihi) inhi=i
        ENDIF
13    CONTINUE
      rtol=2.*abs(y(ihi)-y(ilo))/(abs(y(ihi))+abs(y(ilo))+TINY)
      IF (rtol.lt.ftol) THEN
        swap=y(1)
        y(1)=y(ilo)
        y(ilo)=swap
        DO 14 n=1,ndim
          swap=p(1,n)
          p(1,n)=p(ilo,n)
          p(ilo,n)=swap
14      CONTINUE
        RETURN
      ENDIF
      IF (iter.ge.ITMAX) PAUSE 'ITMAX exceeded in amoeba'
      iter=iter+2
      ytry=amotry(p,y,psum,mp,np,ndim,funk,ihi,-1.0)
      IF (ytry.le.y(ilo)) THEN
        ytry=amotry(p,y,psum,mp,np,ndim,funk,ihi,2.0)
      ELSE IF (ytry.ge.y(inhi)) THEN
        ysave=y(ihi)
        ytry=amotry(p,y,psum,mp,np,ndim,funk,ihi,0.5)
        IF (ytry.ge.ysave) THEN
          DO 16 i=1,ndim+1
            IF(i.ne.ilo)THEN
              DO 15 j=1,ndim
                psum(j)=0.5*(p(i,j)+p(ilo,j))
                p(i,j)=psum(j)
15            CONTINUE
              y(i)=funk(psum)
            ENDIF
16        CONTINUE
          iter=iter+ndim
          GOTO 1
        ENDIF
      ELSE
```

```
        iter=iter-1
      ENDIF
      GOTO 2
      END


      FUNCTION amotry(p,y,psum,mp,np,ndim,funk,ihi,fac)
      INTEGER ihi,mp,ndim,np,NMAX
      REAL amotry,fac,p(mp,np),psum(np),y(mp),funk
      PARAMETER (NMAX=20)
      EXTERNAL funk
CU    USES funk
      INTEGER j
      REAL fac1,fac2,ytry,ptry(NMAX)
      fac1=(1.-fac)/ndim
      fac2=fac1-fac
      do 11 j=1,ndim
        ptry(j)=psum(j)*fac1-p(ihi,j)*fac2
11    CONTINUE
      ytry=funk(ptry)
      IF (ytry.lt.y(ihi)) THEN
        y(ihi)=ytry
        DO 12 j=1,ndim
          psum(j)=psum(j)-p(ihi,j)+ptry(j)
          p(ihi,j)=ptry(j)
12      CONTINUE
      ENDIF
      amotry=ytry
      RETURN
      END
C
C
```