# Apéndice A

```
      Program xamoeba
c
c     Driver for routine amoeba
c
c  Este es un programa principal que sirve como ejemplo para el uso de los programas de
c  optimización utilizando el algoritmo de optimización nelder mead que tiene el libro:
c   Numerical Recipes in C: The art of Scientific Computing.(1992).  Press,W, Teukolsky, s.,
c  Vetterling, W. y 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, la cual es declarada como
c  real. ver unas lineas abajo la función bajo ese nombre.
c


c
c  Estas son las definiciones de las variables propias del problema de la tina de baño.
c
      real rdatos(1200)
      integer ndatos
      real rma,ta,ca,rmb,tb,cb
      real rlambda
      common rdatos,ndatos
      character*12 entrada,salida
c
c
c  Parámetros.
c
      integer np,mp
      real ftol
      parameter(np=5,mp=6,ftol=1.0e-6)
      integer i,iter,j,ndim
      real famoeb,p(mp,np),x(np),y(mp)
      external famoeb
      data p/.4,.6,.4,.4,.4,
     1     11.,11.,13.,11.,11.,11.,
     1     10.,10.,10.,11.,10.,10.,
     1     2.5,2.5,2.5,2.5,3.,2.5,
     1     43.,43.,43.,43.,43.,46./
```

```
c
c  Si data p/0,1,0,0,0,0,1,0,0,0,0,1/
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 Número de dimensiones:
c
      ndim=np
c
c
c  Archivos de entrada y salida
c
c
c   En esta sección se solicitan los nombres de los archivos de entrada y salida. también se
c  pone un mensaje inicial en el archivo de salida.

      print*,' '
      print*,'introduce el nombre del archivo de entrada (input)'
      read (*,1) entrada
    1 format (a12)
      open (3,file=entrada,status='old')
      print *,' '
      print *,'introduce el nombre del archivo de salida (output)'
      read (*,1)salida
      open(5,file=salida,status='new')
      write(5,179)
  179 format(//10x,'los valores estimados de los parametros son:')

c
c Se leen el número de observaciones y las observaciones se meten en el vector rdatos.
c
      print*,' '
      print*,' introduce el numero de datos'
      read*,ndatos
      do 344 i=1,ndatos
      read(3,*)rdatos(i)
  344 continue
```

```
c
c
c Se evalúa la función en los puntos dados en la matriz p, donde se almacenaron los
c vértices. estas evaluaciones se guardan en el vector y.

      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 Aquí 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
      rma=p(1,1)
      ca=p(1,2)
      ta=p(1,3)
      rmb=p(1,4)
      tb=p(1,5)
      rlambda=rma/ta*(ta/ca)**rma
      cb=((rmb*tb**(rmb-1.))/rlambda)**(1./rmb)
      fvalue=-y(1)
      print*,' '
      print*,' '
      print*,'     rma=',rma
      print*,'      ca=',ca
      print*,'      ta=',ta
      print*,'     rmb=',rmb
      print*,'      tb=',tb
      print*,' rlambda=',rlambda
      print*,'      cb=',cb
      print*,' funcion=',fvalue
c
c
c Se termina la ejecución del programa.
c
      stop
      end
c
c
```

```
      real function famoeb(x)
      real rdatos(1200),x(5)
      integer ndatos
      common rdatos,ndatos
c
c
c     Vector que contiene el valor de los parámetros
c
      double precision rlambda
      double precision rma,ca,ta,rmb,tb,cb

      rma=x(1)
      ca=x(2)
      ta=x(3)
      rmb=x(4)
      tb=x(5)
c
c Aqui se meten las restricciones que deben haber sobre los parámetros.
c
      if(ta.le.0..or.
     1   ca.le.0..or.
     2  rma.le.0..or.
     3  rma.ge.1..or.
     4   tb.le.0..or.
     6   ta.gt.tb.or.
     7   rmb.le.1.)go to 444
      rlambda=rma/ta*(ta/ca)**rma
      cb=((rmb*tb**(rmb-1.))/rlambda)**(1/rmb)

c Calculando el valor de la función de máxima verosimilitud

      value=0.

      do 55 i=1,ndatos
c
c
c  Función de máxima verosimilitud para fallas tempranas
c
      if(rdatos(i).le.ta)term=log(rma)+(rma-1.)*log(rdatos(i))
     1-rma*log(ca)-(rdatos(i)/ca)**rma
c
c
c   Función de máxima verosimilitud para fallas aleatorias
c
      if(rdatos(i).gt.ta.and.rdatos(i).le.tb)term=log(rlambda)-
     1rlambda*(ta*((1.-rma)/rma)+rdatos(i))
```

```
c
c
c   Función de máxima verosimilitud para fallas tardías
c
      if(rdatos(i).gt.tb)term=log(rmb)+(rmb-1)*log(rdatos(i))-
     1rmb*log(cb)-(rlambda*(ta*(1.-rma)/rma+tb*(rmb-1)/rmb)+
     2(rdatos(i)/cb)**rmb)
c
c
c
c
      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
      return
      end
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   Usa 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
```