2

ELEMENTS OF SCIENTIFIC GRAPHICS

$$f \ : \ [x_{\min}, x_{\max}] \rightarrow \ [y_{\min}, y_{\max}],$$
$$f(x_i) = y_i, \quad i = 1, 2, ..., n.$$

$$\begin{aligned} \bar{x} &= A_x x + B_x \\ \bar{y} &= A_y y + B_y. \end{aligned}$$

$$\begin{aligned} \bar{x}_{\min} &= A_x x_{\min} + B_x \\ \bar{x}_{\max} &= A_x x_{\max} + B_x, \end{aligned}$$

$$\begin{aligned} A_x &= (\bar{x}_{\max} - \bar{x}_{\min})/(x_{\max} - x_{\min}) \\ B_x &= \bar{x}_{\min} - A_x x_{\max}. \end{aligned} \tag{2.1}$$

$$\begin{aligned} A_y &= (\bar{y}_{\max} - \bar{y}_{\min})/(y_{\max} - y_{\min}) \\ B_y &= \bar{y}_{\min} - A_y y_{\min}. \end{aligned} \tag{2.2}$$

$$\begin{aligned} \bar{x}_i &= A_x x_i + B_x, \quad i = 1, 2, ..., n \\ \bar{y}_i &= A_y y_i + B_y. \end{aligned}$$

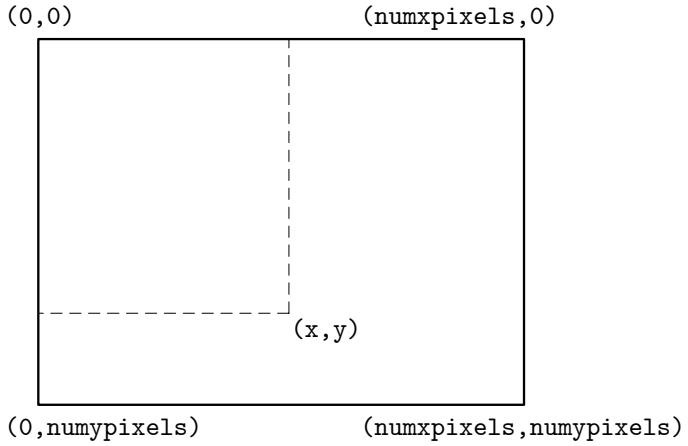

FIGURA 2.1. Physical coordinate system for graphics applications.

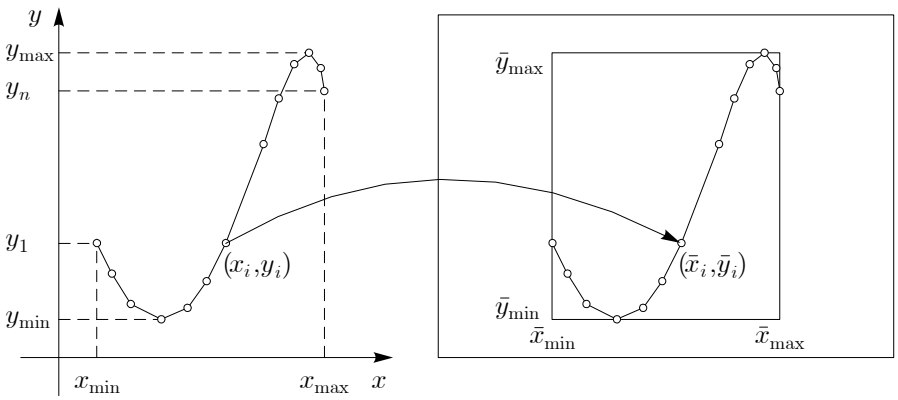

FIGURA 2.2. The correspondence between user coordinates (x_i, y_i) and physical coordinates $(\bar{x}_i, \bar{y}_i)$ is linear.

```
!------------------------------- graphlib.f90 -----------------------------
!=============================================================================
subroutine InitGraph()
!-----------------------------------------------------------------------------
   use MSFLIB
   implicit real(8) (a-h,o-z)
   logical statusmode
   type (windowconfig) screen
   common /graph/ screen

   screen.numxpixels  = -1
   screen.numypixels  = -1
   screen.numtextcols = -1
   screen.numtextrows = -1
   screen.numcolors   = -1
   screen.fontsize    = -1
   screen.title       = " "C
   statusmode = SETWINDOWCONFIG(screen)
   if (.not.statusmode) statusmode = SETWINDOWCONFIG(screen)

   statusmode = GETWINDOWCONFIG(screen)
   numfonts = INITIALIZEFONTS()

   dummy = SETBKCOLORRGB(0)
   dummy = SETCOLORRGB(#FFFFFF)
   dummy = SETTEXTCOLORRGB(#FFFFFF)
end

!=============================================================================
subroutine Plot0(x, y, n, fxmin, fxmax, fymin, fymax)
!-----------------------------------------------------------------------------
!  Plots a tabulated function of one variable and frames it in the window
!  (fxmin,fxmax) x (fymin,fymax), specified by fractional screen coordinates.
!  The axes are scaled automatically.
!
!  x(n)        - abscissas of the representation points
!  y(n)        - ordinates of the representation points
!  n           - no. of points
!  fxmin,fxmax - fractional horizontal extension of the plot on the screen
!                 (0 < fxmin < fxmax < 1)
!  fymin,fymax - fractional vertical extension of the plot on the screen
!                 (0 < fymin < fymax < 1)
!-----------------------------------------------------------------------------
   use MSFLIB
   implicit real(8) (a-h,o-z)

   real(8) x(n), y(n)
   type (windowconfig) screen
   type (xycoord) ixyold
   common /graph/ screen

   nxpix = screen.numxpixels                    ! absolute window coordinates
   nypix = screen.numypixels

   ixmin = nint(fxmin*nxpix); iymin = nint((1.0-fymin)*nypix)
   ixmax = nint(fxmax*nxpix); iymax = nint((1.0-fymax)*nypix)
   dummy = RECTANGLE($GBORDER,ixmin,iymax,ixmax,iymin)              ! border
```

```
   xmin = x(1); xmax = x(n)                                          ! X-AXIS
   ax = (ixmax-ixmin)/(xmax-xmin)                          ! scale coefficients
   bx = ixmin - ax*xmin

   ymin = y(1); ymax = y(1)                                          ! Y-AXIS
   do i = 1,n
      ymin = min(ymin,y(i))
      ymax = max(ymax,y(i))
   end do
   if (ymin == 0.d0 .and. ymax == 0.d0) then                ! horizontal plot
      ymin = -1.0d0
      ymax =  1.0d0
   end if
   if (ymin == ymax) then
      ymin = 0.9 * ymin
      ymax = 1.1 * ymax
   end if
   ay = (iymax-iymin)/(ymax-ymin)                          ! scale coefficients
   by = iymin - ay*ymin

   if (xmin*xmax < 0) then                                          ! draw axes
      call MOVETO(nint(bx),iymin,ixyold); dummy = LINETO(nint(bx),iymax)
   end if
   if (ymin*ymax < 0) then
      call MOVETO(ixmin,nint(by),ixyold); dummy = LINETO(ixmax,nint(by))
   end if
                                          ! draw lines between plot points
   ix = nint(ax*x(1)+bx); iy = nint(ay*y(1)+by)
   call MOVETO(ix,iy,ixyold)                                     ! first point

   do i = 2,n                                          ! loop over plot points
      ix = nint(ax*x(i)+bx); iy = nint(ay*y(i)+by)
      dummy = LINETO(ix,iy)
   end do
end
```

```
!-------------------------------------------------------------------------------
!-------------------------------------------------------------------------------
!-------------------------------------------------------------------------------
program Plot_0
!-------------------------------------------------------------------------------
!-------------------------------------------------------------------------------
!-------------------------------------------------------------------------------
   use MSFLIB
   implicit real(8) (a-h,o-z)

   real(8), allocatable :: x(:), y(:)

   call InitGraph()

   print '(" xmin = "\)'; read(*,*) xmin
   print '(" xmax = "\)'; read(*,*) xmax
   print '(" n    = "\)'; read(*,*) n

   allocate(x(n),y(n))

   h = (xmax-xmin)/(n-1)
   do i = 1,n
      x(i) = xmin + (i-1)*h
      y(i) = func(x(i))
   end do

   call CLEARSCREEN($GCLEARSCREEN)

   call Plot0(x,y,n,0.1d0,0.5d0,0.5d0,0.9d0)
end

!===============================================================================
function func(x)
!-------------------------------------------------------------------------------
   implicit real(8) (a-h,o-z)

   func = x**3 * exp(-x)
end
```

```
!------------------------------ graphlib.f90 -----------------------------
!=======================================================================
subroutine InitGraph()
!-----------------------------------------------------------------------
   use MSFLIB
   implicit real(8) (a-h,o-z)
   logical statusmode
   type (windowconfig) screen
   common /graph/ screen

   screen.numxpixels  = -1
   screen.numypixels  = -1
   screen.numtextcols = -1
   screen.numtextrows = -1
   screen.numcolors   = -1
   screen.fontsize    = -1
   screen.title       = " "C
   statusmode = SETWINDOWCONFIG(screen)
   if (.not.statusmode) statusmode = SETWINDOWCONFIG(screen)

   statusmode = GETWINDOWCONFIG(screen)
   numfonts = INITIALIZEFONTS()

   dummy = SETBKCOLORRGB(0)
   dummy = SETCOLORRGB(#FFFFFF)
   dummy = SETTEXTCOLORRGB(#FFFFFF)
end

!=======================================================================
subroutine Line(ix1,iy1,ix2,iy2)
!-----------------------------------------------------------------------
   use MSFLIB
   type (xycoord) ixyold

   call MOVETO(ix1,iy1,ixyold); dummy = LINETO(ix2,iy2)
end

!=======================================================================
subroutine OutTextXY(text,ix,iy)
!-----------------------------------------------------------------------
   use MSFLIB
   type (xycoord) ixyold
   character*(*) text

   call MOVETO(ix,iy,ixyold); call OUTGTEXT(text)
end

!=======================================================================
function FMagn(x)
!-----------------------------------------------------------------------
! Returneaza ordinul de marime al numarului float x sub forma 10^n
!-----------------------------------------------------------------------
   implicit real(8) (a-h,o-z)

   FMagn = 0.d0
   if (x /= 0d0) then
      if (abs(x) >= 1.d0) then
         FMagn = 10.d0**int(log10(abs(x)))
```

```
      else
         FMagn = 0.1d0**(int(abs(log10(abs(x))))+1)
      end if
   end if
end

!=======================================================================
subroutine Limits(xmin,xmax,scale,nsigd,nintv)
!-----------------------------------------------------------------------
!  Extends a real interval [xmin,xmax] to the least larger interval divisible
!  in at most 10 subintervals of length expressible as d * 10^p, where d is
!  1, 2 or 5, and p is an integer.
!  Other returned values:
!
!  scale   - scale factor (10^p)
!  nsigd   - no. of significant digits
!  nintv   - no. of subintervals
!-----------------------------------------------------------------------
   implicit real(8) (a-h,o-z)
   parameter (eps = 1d-6)                      ! relative precision of limits
   real(8) xfact(3)                            ! successive reduction factors
   data xfact /0.5d0,0.5d0,0.4d0/                        ! 0.5*0.5*0.4 = 0.1

   if (abs(xmax - xmin).lt.10*eps*abs(xmax)) then
      corrmin = (1.d0 - 10*sign(eps,xmin))
      corrmax = (1.d0 + 10*sign(eps,xmin))
      xmin = corrmin * xmin
      xmax = corrmax * xmax
   end if

   if ((xmin*xmax) == 0.0) then                      ! initial scale factor
      factor = 1.d0/(eps * max(FMagn(xmin),FMagn(xmax)))
   else
      factor = 1.d0/(eps * min(FMagn(xmin),FMagn(xmax)))
   end if

   corrmin = 1.d0 + sign(eps,xmin)                            ! corrections
   corrmax = 1.d0 - sign(eps,xmax)
   do i = 1,100
      xmins = floor  (xmin * factor * corrmin)   ! multiply factor cyclically
      xmaxs = ceiling(xmax * factor * corrmax)   ! with xfact() until the
      xnint = abs(xmaxs - xmins)                 ! number of subintervals
      if (xnint < 10) exit                       ! xnint < 10
      modi = mod(i,3)
      factor = factor * xfact(modi+1)
   end do
   factor = (1.d0 + eps)/factor                            ! correct factor
   xmin = xmins * factor                                    ! xmin and xmax
   xmax = xmaxs * factor
   scale  = max(FMagn(xmin),FMagn(xmax))                     ! scale factor
   factor = max(abs(xmins),abs(xmaxs));
   do i = 1,modi
      factor = factor / xfact(i+1)
   end do
   nsigd = log10(factor) + 1                     ! no. of significant digits
   nintv = nint(xnint)                                ! no. of subintervals
end
```

```
!=========================================================================
subroutine LabelFormat(x,scale,nsigd,mant,expn)
!-------------------------------------------------------------------------
!  Formats number x (with scale factor scale) to nsigd significant digits
!  returning the mantissa in mant and the exponent of 10 in expn
!-------------------------------------------------------------------------
   implicit real(8) (a-h,o-z)
   parameter (ndigmax = 5)                          ! maximum no. of digits
   character*(*) mant, expn

   n = nint(log10(scale))                                ! exponent of 10
   if ((n < 0).or.(n > 3)) then
      write(expn,'(i3)') n
      x = x / scale                                  ! divide x to scale
      n = 0
   end if

   n = n + 1                         ! no. of digits before the decimal point
   ndig = min(ndigmax,max(nsigd,n))                    ! total no. of digits
   ndec = ndig - n                                  ! no. of decimal places
   if (ndec .ne. 0) ndig = ndig + 1
   if (x < 0.d0) ndig = ndig + 1
   if (ndec == 0) then                                           ! mantissa
      write(mant,'(i'//char(ndig+48)//')') int(x)
   else
      write(mant,'(f'//char(ndig+48)//'.'// char(ndec+48)//')') x
   end if
end

!=========================================================================
subroutine Plot(x, y, nmax, n, icolor, style, nplot, &
                xming, xmaxg, yming, ymaxg, iopt, &
                fxmin, fxmax, fymin, fymax, xtext, ytext, ttext, ipos)
!-------------------------------------------------------------------------
!  Plots nplot tabulated functions of one variable and frames them in the
!  window (fxmin,fxmax) x (fymin,fymax), specified by fractional screen
!  coordinates. The axes are scaled and labeled automatically.
!
!  x(nmax)       - abscissas of the representation points
!  y(nmax,nplot) - ordinates of the representation points
!  nmax          - actual column dimension of x and y in the calling routine
!  n             - no. of points
!  icolor        - RGB color of the plots
!  style         - plot style: -1 - drop line
!                               0 - circles
!                               1 - continuous line
!                               2 - long dotted line
!                               3 - dotted line
!  nplot         - no. of plots
!  iopt          -  =0 - scale the plot automatically
!                   !=0 - frame the plot within the specified user coordinates
!                         (xming,xmaxg) x (yming,ymaxg)
!  fxmin,fxmax   - fractional horizontal extension of the plot on the screen
!                   (0 < fxmin < fxmax < 1)
!  fymin,fymax   - fractional vertical extension of the plot on the screen
!                   (0 < fymin < fymax < 1)
!  xtext         - x axis label
!  ytext         - y axis label
```

```
!  ttext         - plot title
!  ipos          - plot title positioning: 0 - center-up
!                                          2 - left-up  ; 1 - right-up
!                                          3 - left-down; 4 - right-down
!-------------------------------------------------------------------------------
   use MSFLIB
   implicit real(8) (a-h,o-z)

   real(8) x(nmax), y(nmax,nplot)
   integer icolor(nplot), style(nplot)
   character*(*) xtext, ytext, ttext
   character mant*10, expn*5, label*50
   integer tic
   type (fontinfo) info
   type (windowconfig) screen
   common /graph/ screen

   nxpix = screen.numxpixels                    ! absolute window coordinates
   nypix = screen.numypixels

   ixmin = nint(fxmin*nxpix); iymin = nint((1.0-fymin)*nypix)
   ixmax = nint(fxmax*nxpix); iymax = nint((1.0-fymax)*nypix)
   dummy = RECTANGLE($GBORDER,ixmin,iymax,ixmax,iymin)              ! border

   select case (ipos)                                     ! select title font
      case default
         nfont = SETFONT('t''Arial''h16')
         dummy = GETFONTINFO(info)
         ixtext = (ixmin + ixmax - GETGTEXTEXTENT(trim(ttext)))/2
         iytext = iymax - 2*info.pixheight
      case (1)
         nfont = SETFONT('t''Arial''h14')
         dummy = GETFONTINFO(info)
         ixtext = ixmax - GETGTEXTEXTENT(trim(ttext))
         iytext = iymax + info.pixheight/4
      case (2)
         nfont = SETFONT('t''Arial''h14')
         dummy = GETFONTINFO(info)
         ixtext = ixmin + info.pixheight/2
         iytext = iymax + info.pixheight/4
      case (3)
         nfont = SETFONT('t''Arial''h14')
         dummy = GETFONTINFO(info)
         ixtext = ixmin + info.pixheight/2
         iytext = iymin - 5*info.pixheight/4
      case (4)
         nfont = SETFONT('t''Arial''h14')
         dummy = GETFONTINFO(info)
         ixtext = ixmax - GETGTEXTEXTENT(trim(ttext))
         iytext = iymin - 5*info.pixheight/4
   end select

   call SETGTEXTROTATION(0)
   call OutTextXY(ttext,ixtext,iytext)                               ! title

   nfont = SETFONT('t''Arial''h14')
   dummy = GETFONTINFO(info)
```

```
if (iopt == 0) then                                          ! X-AXIS
   xmin = x(1); xmax = x(n)
else
   xmin = xming; xmax = xmaxg
end if
call Limits(xmin,xmax,scale,nsigd,nintv)                  ! extended limits
ax = (ixmax-ixmin)/(xmax-xmin)                          ! scale coefficients
bx = ixmin - ax*xmin
hh = (xmax-xmin)/nintv                                  ! labeling step size
tic = (ixmax-ixmin)/100                                        ! tic length
iytext = iymin + info.pixheight/2
do i = 1,nintv+1                                               ! label axis
   xi = xmin+(i-1)*hh
   ix = nint(ax*xi + bx)
   call Line(ix,iymin,ix,iymin-tic)                             ! draw tics
   call Line(ix,iymax,ix,iymax+tic)
   if (xtext /= "") then
      call LabelFormat(xi,scale,nsigd,mant,expn)
      ixtext = ix - GETGTEXTEXTENT(trim(mant))/2
      call OutTextXY(mant,ixtext,iytext)                        ! labels
   end if
end do
if (xtext /= "") then
   label = xtext
   if ((scale < 1.0).or.(scale > 1000.0)) &
      label = trim(label) // '   x 1e' // expn
   ixtext = (ixmin + ixmax - GETGTEXTEXTENT(trim(label)))/2
   iytext = iytext + 3*info.pixheight/2
   call OutTextXY(label,ixtext,iytext)                       ! axis label
end if

if (iopt == 0) then                                          ! Y-AXIS
   ymin = y(1,1); ymax = y(1,1)                            ! ymin and ymax
   do iplot = 1,nplot
      do i = 1,n
         ymin = min(ymin,y(i,iplot))
         ymax = max(ymax,y(i,iplot))
      end do
   end do
else
   ymin = yming; ymax = ymaxg
end if
if (ymin == 0.d0 .and. ymax == 0.d0) then                ! horizontal plot
   ymin = -1.0d0
   ymax =  1.0d0
end if
if (ymin == ymax) then
   ymin = 0.9 * ymin
   ymax = 1.1 * ymax
end if

call Limits(ymin,ymax,scale,nsigd,nintv)                  ! extended limits
ay = (iymax-iymin)/(ymax-ymin)                          ! scale coefficients
by = iymin - ay*ymin
hh = (ymax-ymin)/nintv                                  ! labeling step size
ixtextMin = ixmin
do i = 1,nintv+1
   yi = ymin+(i-1)*hh                                         ! label axis
```

```
   iy = nint(ay*yi + by)
   call Line(ixmin,iy,ixmin+tic,iy)                              ! draw tics
   call Line(ixmax,iy,ixmax-tic,iy)
   call LabelFormat(yi,scale,nsigd,mant,expn)
   ixtext = ixmin - info.pixheight/2 - GETGTEXTEXTENT(trim(mant))
   ixtextMin = min(ixtextMin,ixtext)
   iytext = iy - info.pixheight/2
   call OutTextXY(mant,ixtext,iytext)                               ! labels
end do
label = ytext
if ((scale < 1.0).or.(scale > 1000.0)) &
   label = trim(label) // '   x 1e' // expn
ixtext = ixtextMin - 3*info.pixheight/2
iytext = (iymin + iymax + GETGTEXTEXTENT(trim(label)))/2
nfont = SETFONT('t''Arial''h12')
call SETGTEXTROTATION(900)
call OutTextXY(label,ixtext,iytext)                              ! axis label

if (xmin*xmax < 0) then                                           ! draw axes
   call Line(nint(bx),iymin,nint(bx),iymax)
end if
if (ymin*ymax < 0) then
   call Line(ixmin,nint(by),ixmax,nint(by))
end if

call SETCLIPRGN(ixmin,iymax,ixmax,iymin)                   ! clip plot area

icolor0 = GETCOLORRGB()

do iplot = 1,nplot                                          ! loop over plots
   ix = nint(ax*x(1)+bx); iy = nint(ay*y(1,iplot)+by)
   dummy = SETCOLORRGB(icolor(iplot))                      ! choose plot syle
   select case (style(iplot))
      case (-1)
         call Line(ix,iy,ix,nint(ay*ymin+by))
      case (0)
         dummy = ELLIPSE($GBORDER,ix-tic,iy-tic,ix+tic,iy+tic)
      case (1)
         call SETLINESTYLE(#FFFF)
      case (2)
         call SETLINESTYLE(#FF00)
      case (3)
         call SETLINESTYLE(#F0F0)
   end select

   ix0 = ix; iy0 = iy
   do i = 2,n                                          ! loop over plot points
      ix = nint(ax*x(i)+bx); iy = nint(ay*y(i,iplot)+by)
      select case (style(iplot))                          ! choose plot style
         case (-1)
            call Line(ix,iy,ix,nint(ay*ymin+by))
         case (0)
            dummy = ELLIPSE($GBORDER,ix-tic,iy-tic,ix+tic,iy+tic)
         case (1,2,3)
            call Line(ix0,iy0,ix,iy)
            ix0 = ix; iy0 = iy
      end select
   end do
```

```
   end do

   dummy = SETCOLORRGB(icolor0)                              ! restore defaults
   call SETGTEXTROTATION(0)

   call SETLINESTYLE(#FFFF)
   call SETCLIPRGN(0,nypix,nxpix,0)
end
```

```
!------------------------------------------------------------------------------
!------------------------------------------------------------------------------
!------------------------------------------------------------------------------
program Plot_1
!------------------------------------------------------------------------------
!------------------------------------------------------------------------------
!------------------------------------------------------------------------------
   use MSFLIB
   implicit real(8) (a-h,o-z)

   real(8), allocatable :: x(:), y(:)
   integer icolor(1), style(1)
   data icolor/#FF0000/
   data style /1/

   call InitGraph()

   print '(" xmin = "\)'; read(*,*) xmin
   print '(" xmax = "\)'; read(*,*) xmax
   print '(" n    = "\)'; read(*,*) n

   allocate(x(n),y(n))

   h = (xmax-xmin)/(n-1)
   do i = 1,n
      x(i) = xmin + (i-1)*h
      y(i) = func(x(i))
   end do

   call CLEARSCREEN($GCLEARSCREEN)

   call Plot(x,y,n,n,icolor,style,1, &
             0.d0,0.d0,0.d0,0.d0,0, &
             0.1d0,0.5d0,0.5d0,0.9d0,"x","x**3 * exp(-x)","Plot 2D",0)
end

!==============================================================================
function func(x)
!------------------------------------------------------------------------------
   implicit real(8) (a-h,o-z)

   func = x**3 * exp(-x)
end
```

Bibliografie

[1] T.A. Beu, *Calcul numeric în C* (Microinformatica, Cluj-Napoca, 1992).

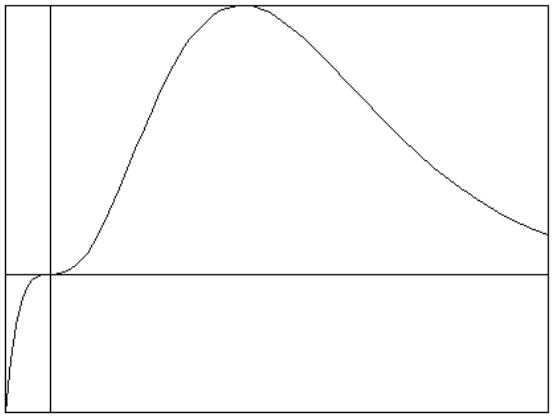

FIGURA 2.3. Plot of function x^3e^{-x} with routine `Plot0`.

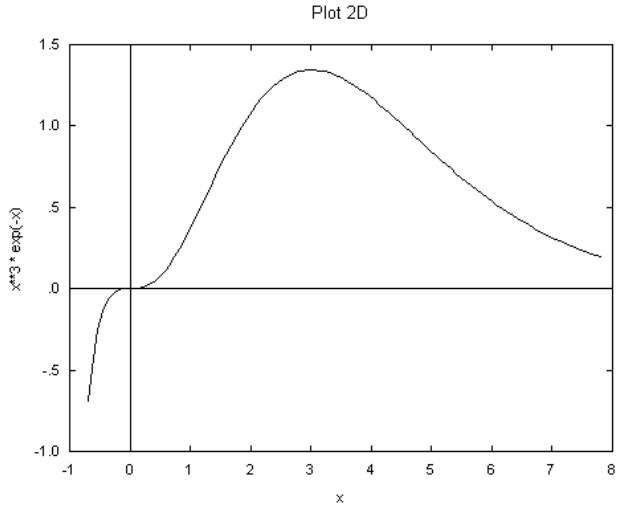

FIGURA 2.4. Plot of function x^3e^{-x} with routine `Plot`.