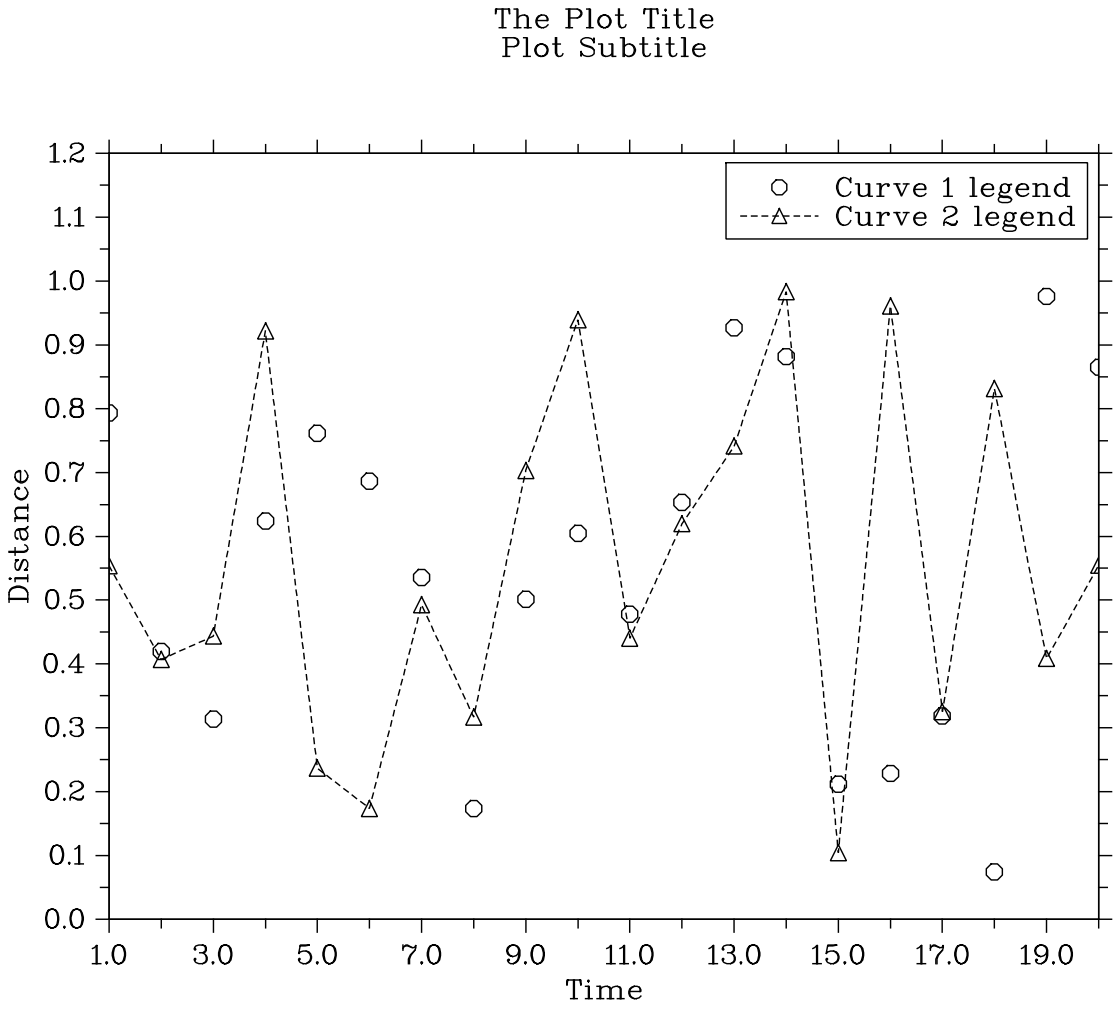
2-D Data Plotting with DISLIN

1. DISLIN web site is at `www.dislin.de`. The web site has an online manual and example plots (along with sample Fortran 90 code) that are very useful.
2. We are currently using DISLIN with the gfortran compiler. To compile, link, and run a program using the DISLIN graphics library, use the command

   `gf95link -a -r8` *source-file-name <other-compile-and-link-flags>*

   where *source-file-name* is given without the `.f90` ending.
3. You must `USE` the dislin Fortran module `DISLIN` found in the file `dislin.f90`. This file can be copied into your working directory from `/usr/local/dislin/gf/real64`. The `use dislin` statement should be placed in any program component (main program or subprogram) that calls a DISLIN routine. The module must be compiled (with the command `gfortran -c dislin.f90`) before using it in any program.
4. All floating point arguments to DISLIN subroutines **must** be type `real(wp)`, where parameter `wp` is defined as `selected_real_kind(15)`.
5. All character strings passed to DISLIN as control parameters can be either upper or lower case.
6. The simple 2-D plot shown on the next page was generated with the following program:

```
program distest
  use dislin
  implicit none
  integer, parameter::wp=selected_real_kind(14)
  integer::i,n
  real(wp)::xa,xe,xor,xstep,ya,ye,yor,ystep
  real(wp), dimension(:), allocatable::x,y,y2
  character (len=200)::legendstring
  ! Sample program for 2-d data plot using dislin
  call random_seed
  write(*,"(a)",advance="no")"Number of data point to generate? "
  read(*,*)n
  allocate(x(n),y(n),y2(n))
  do i=1,n
    x(i)=real(i,wp)
    call random_number(y(i))  !just some random points for the demo
    call random_number(y2(i))
  end do
  xa=1.0_wp      ! xa is the lower limit of the x-axis.
  xe=real(n,wp)  ! xe is the upper limit of the x-axis.
  xor=1.0_wp     ! xor is the first x-axis label.
  xstep=2.0_wp   ! xstep is the step between x-axis labels.
  ya=0.0_wp      ! ya is the lower limit of the y-axis.
  ye=1.2_wp      ! ye is the upper limit of the y-axis.
  yor=0.0_wp     ! yor is the first y-axis label.
  ystep=0.1_wp   ! ystep is the step between y-axis labels.
  !Plot data using DISLIN
  call metafl("XWIN") ! or "PS", "EPS", "PDF", "WMF" "BMP"
  call setpag("USAL")  !"USAL" is US size A landscape, "USAP" is portrait
  call scrmod("REVERS") !sets black on white background
  call disini() !Initialize dislin
  call complx ! Sets the font (call simplx is an alternative)
  call name("Time","X") ! Set label for x-axis
  call name("Distance","Y") ! Set label for y-axis
  call titlin("The Plot Title",1) ! Set 1st line of plot title
  call titlin("Plot Subtitle",2)  ! Set 2nd line of plot title (if any)
  call graf (xa, xe, xor, xstep, ya, ye, yor, ystep) ! sets up axis
  call title ! Actually draw the title in over the axis
  call legini(legendstring,2,20) ! Store 2 lines of legend text, max 20 characters/line
  call legtit("") ! set legend  title (default="legende")
  call leglin(legendstring,"Curve 1 legend",1) ! Specify the legend text for curve 1
  call leglin(legendstring,"Curve 2 legend",2) ! Specify the legend text for curve 2
  call incmrk(-1) !0=no symbols at points, 1=curve and symbol, -1=just symbols
  call marker(1) !symbol code from 0 to 21
  call hsymbl(30) !symbol size (default=35)
  call curve(x,y,n) ! draw the x-y curve
  call lintyp(2) ! Change the line style (values are from 1 to 7)
  call incmrk(1)
  call curve(x,y2,n) ! draw the x-y curve
  !draw legend in location 1-8.  1-4=page corner, 5-8=axis corner,1 and 5=lowerleft
  call legend(legendstring,7) ! draw legend in 7 (upper right inside axis)
  call disfin ! finish off the plot
end program distest
```


The Plot Title
Plot Subtitle
Curve 1 legend
Curve 2 legend
Distance
Time
1.2
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
1.0
3.0
5.0
7.0
9.0
11.0
13.0
15.0
17.0
19.0

7. As an alternative for a single curve, you can use the `quickplot` subroutine. With `quickplot`, you do not need to `USE` the DISLIN module; just call the `quickplot` routine. Documentation listing the arguments for quickplot is available in the file `/faculty-hsu/baf1/dislin/quickplot-doc.txt` and is listed below.

```
The subroutine quickplot is a very simple way to use the Dislin plotting package.
Quickplot is only suitable for plots with a single curve.
See http://dislin.de for documentation on dislin.

An interface for the subroutine quickplot is given below

interface
  subroutine quickplot(x,y,n,xlabel,ylabel,plottitle,device,drawline,linetype, &
                       drawsymbol,symboltype)
    integer, parameter::wp=selected_real_kind(15)
    integer, intent(in)::linetype,n,symboltype
    real(wp), dimension(:),intent(in)::x,y
    character (len=*), intent(in)::device,plottitle,xlabel,ylabel
    logical, intent(in)::drawline,drawsymbol
  end subroutine quickplot
end interface

Input arguments
  x - x axis values to plot (1-d real array)
  y - y axis values to plot (1-d real array)
  n - number of data pairs to plot (integer scaler)
  xlabel - x axis label (character string)
  ylabel - y axis label (character string)
  plottitle - plot title (character string)
  device - output device name (character string)
           choices are  "xwin"     console
                        "ps"       postscript file (dislin_?.ps)
                        "eps"      encapsulated postscript file (dislin_?.eps)
                        "pdf"      pdf file (dislin_?.pdf)
                        "gif"      gif file (dislin_?.gif)
                        "bmp"      windows Bitmap file (dislin_?.bmp)
            where ? is a dislin supplied integer value
  drawline - indicates whether a line drawn between points (logical scaler)
  linetype - code for type of line to draw (if any) (integer with value 0 to 7)
  drawsymbol - indicates whether a symbol drawn at points (logical scaler)
  linetype - code for type of symbol to draw (if any) (integer with value 0 to 21)
```

To utilize quickplot, copy the object code file `/faculty-hsu/baf1/dislin/quickplot.o` to your working directory and link it to your Fortran program with the command

`gf95link -a -r8` *source-file-name* `quickplot.o`

The following program demonstates the use of the quickplot subroutine:

```
program quicktest
  implicit none
  integer, parameter::wp=selected_real_kind(15)
  integer::i,n
  real(wp)::z
  real(wp), dimension(:),allocatable::x,y
  interface
    subroutine quickplot(x,y,n,xlabel,ylabel,plottitle,device,drawline, &
                         linetype,drawsymbol,symboltype)
      integer, parameter::wp=selected_real_kind(15)
      integer, intent(in)::linetype,n,symboltype
      real(wp), dimension(:),intent(in)::x,y
      character (len=*), intent(in)::device,plottitle,xlabel,ylabel
      logical::drawline,drawsymbol
    end subroutine quickplot
  end interface

  do
    write(*,"(a)",advance="no")"Number of data points (negative to stop)? "
    read(*,*)n
    if(n<0)exit
    allocate(x(n),y(n))
```

```
    call random_seed
    do i=1,n
      call random_number(z)
      x(i)=dble(i)+z
      y(i)=dble(i)+z*10
      write(*,*)x(i),y(i)
    end do
    call quickplot(x,y,n,"Time (sec)","Concentration (mg/l)","Sample Plot", &
                   "xwin",.true.,0,.true.,1)
    deallocate(x,y)
  end do
  stop
end program quicktest
```