Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

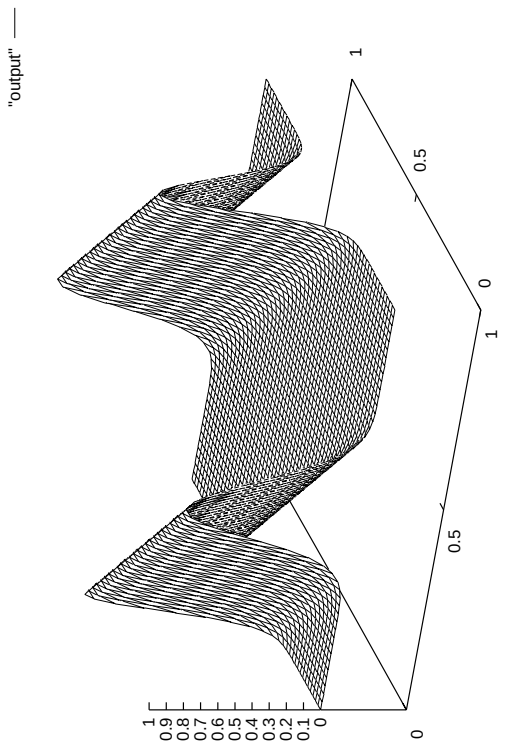
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

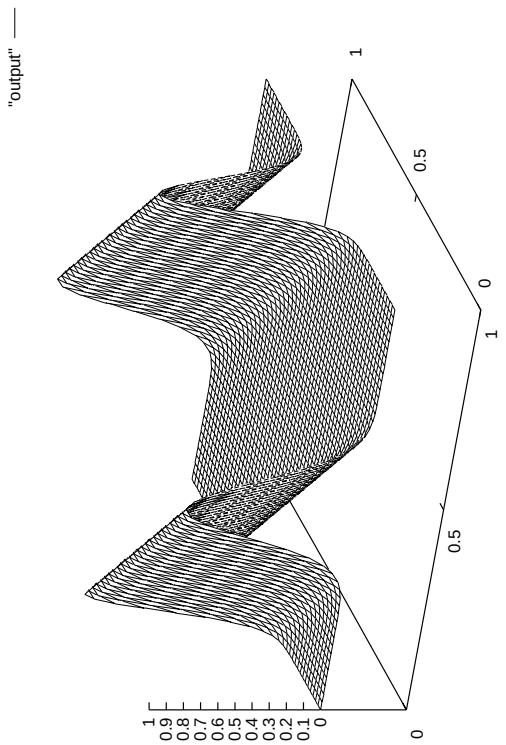
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

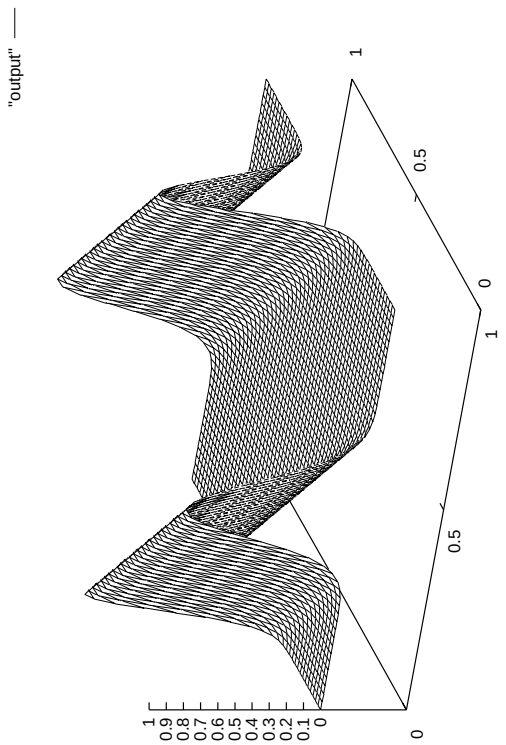
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

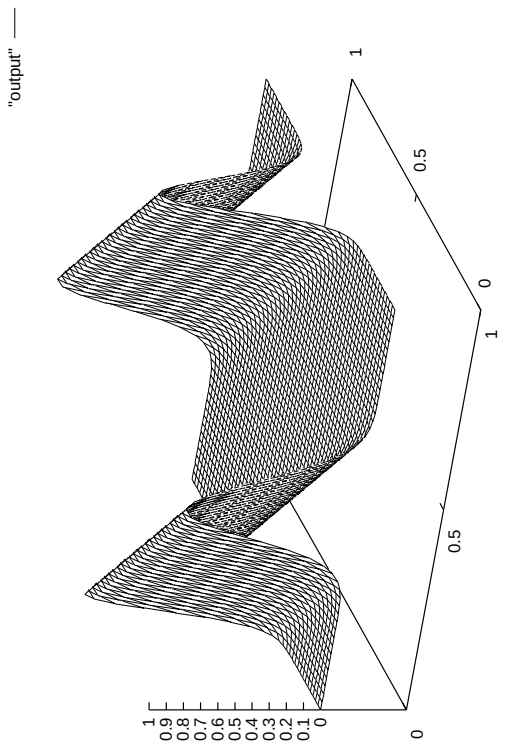
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

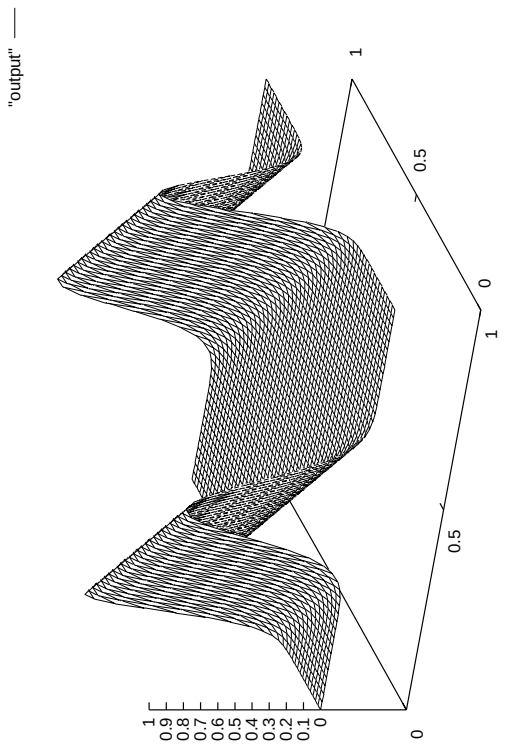
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

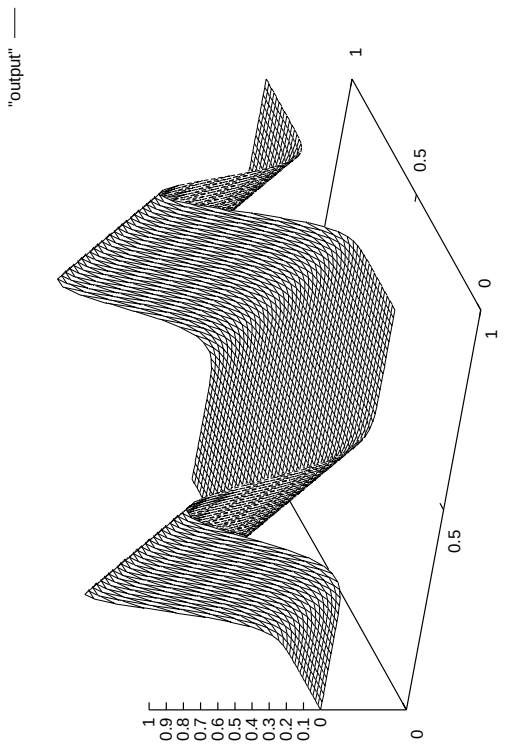
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

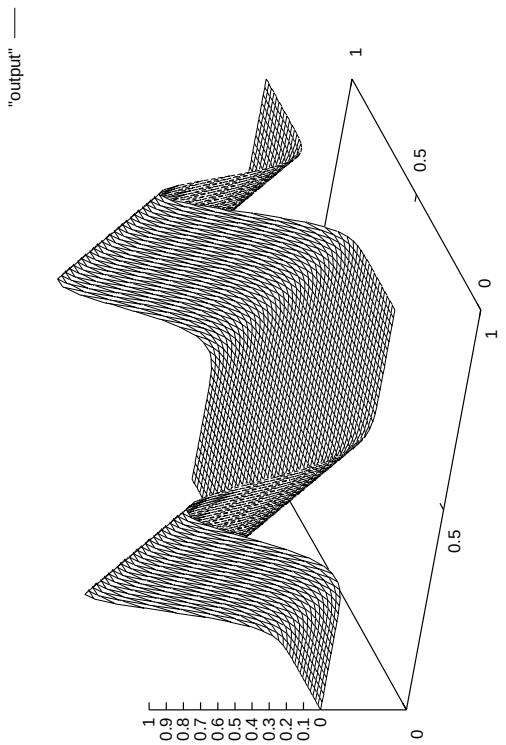
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

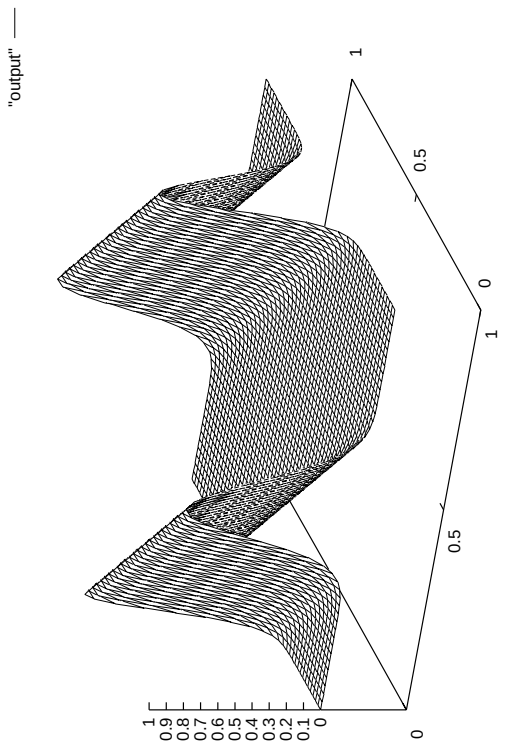
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

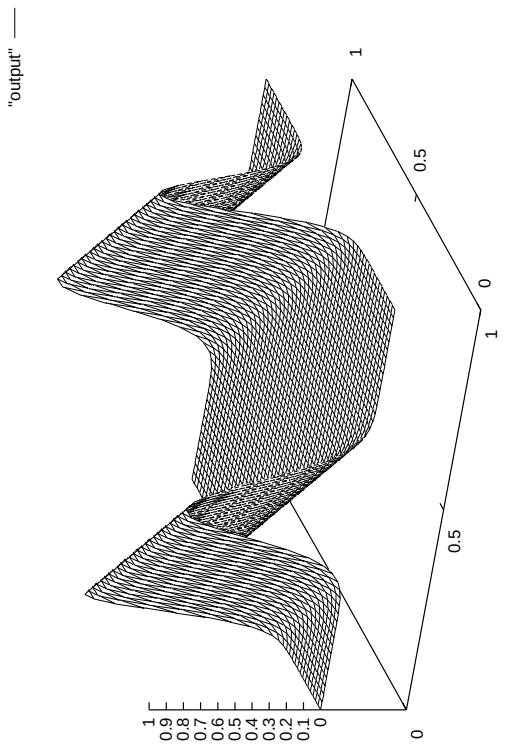
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

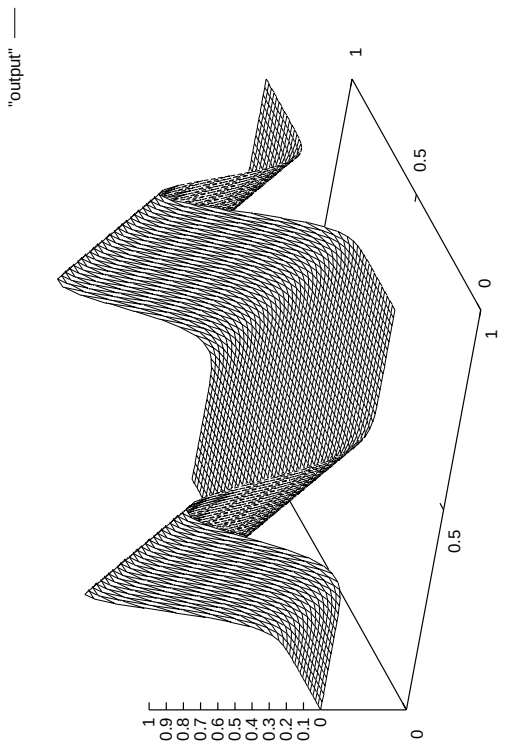
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

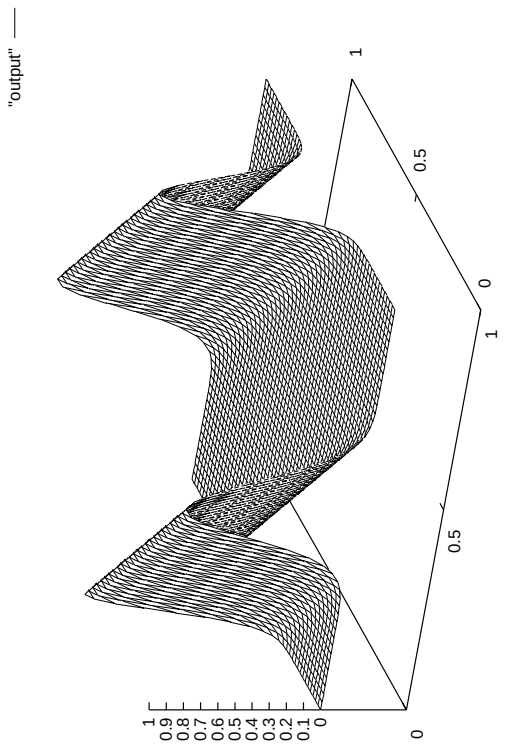
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

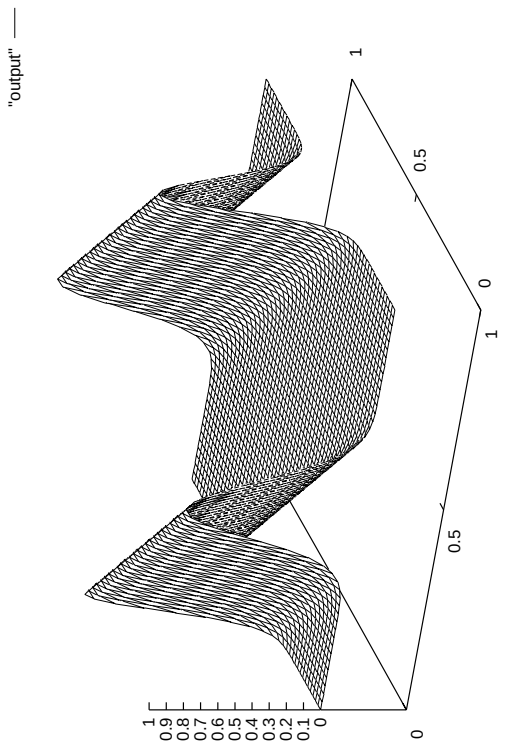
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

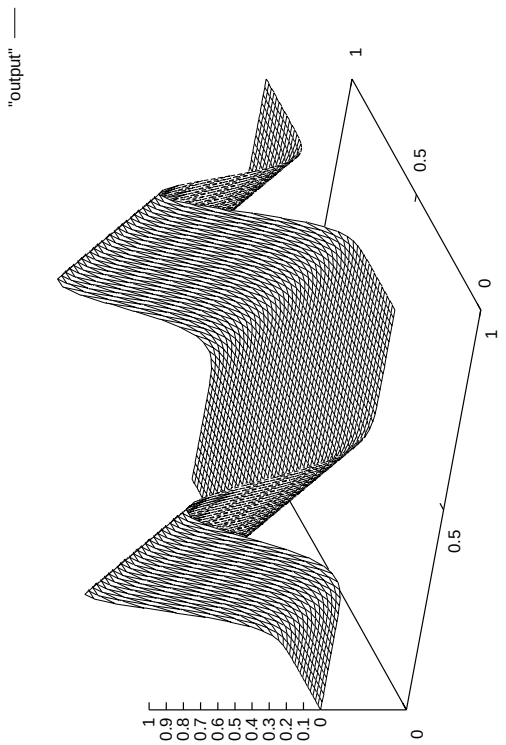
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

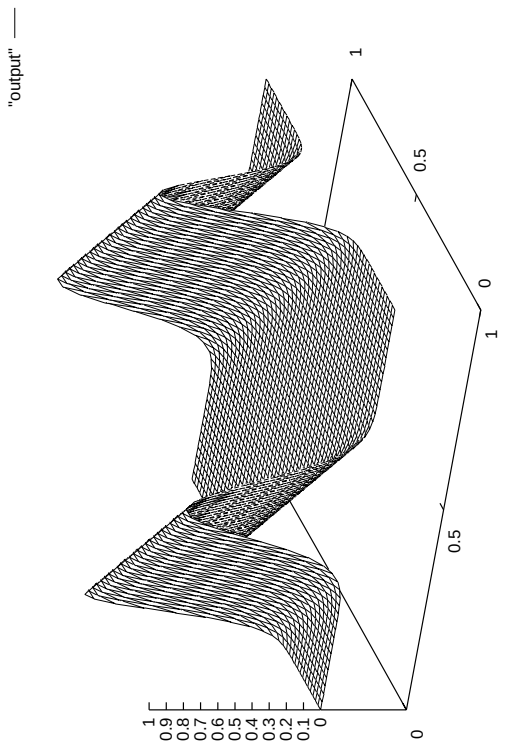
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

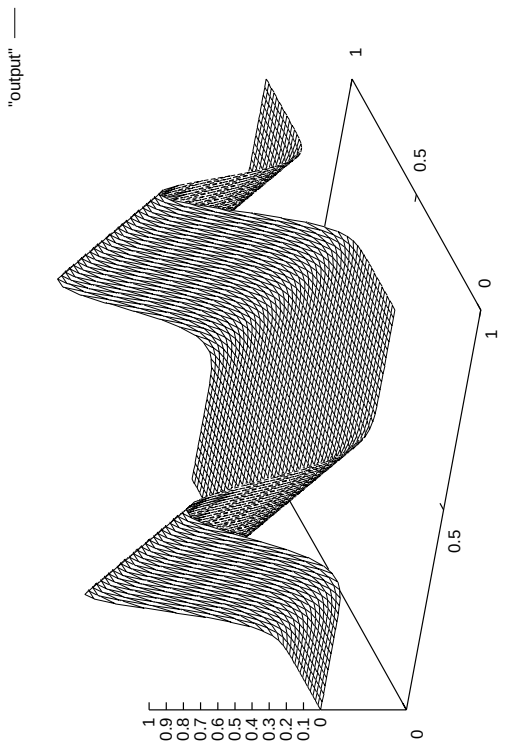
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

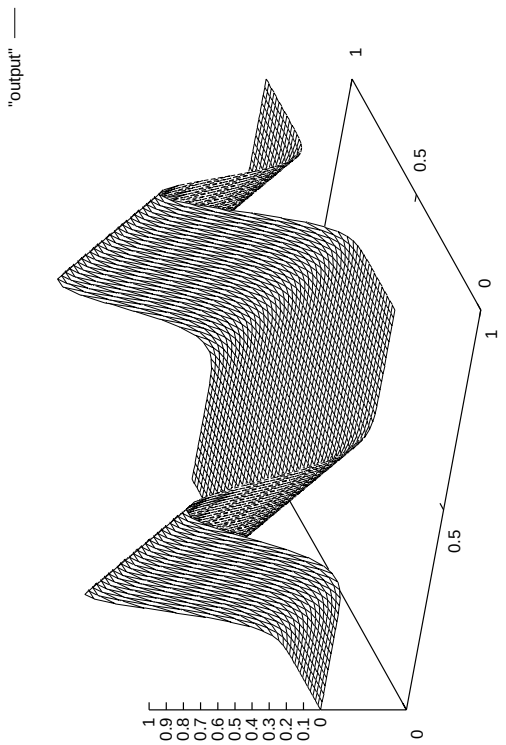
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

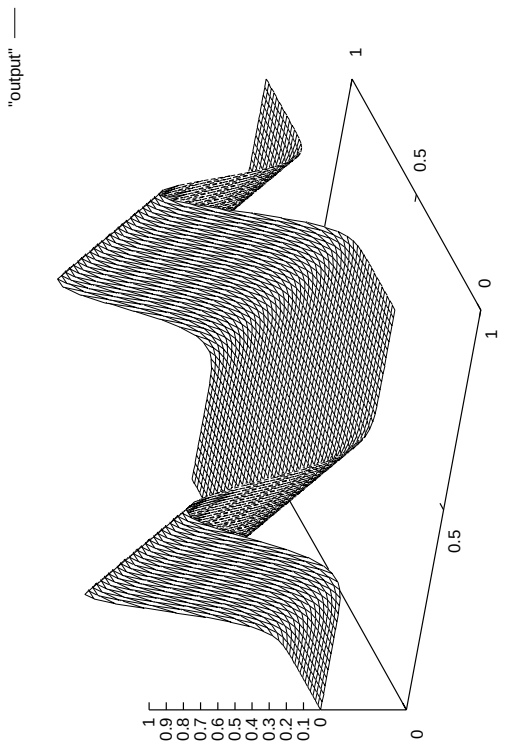
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

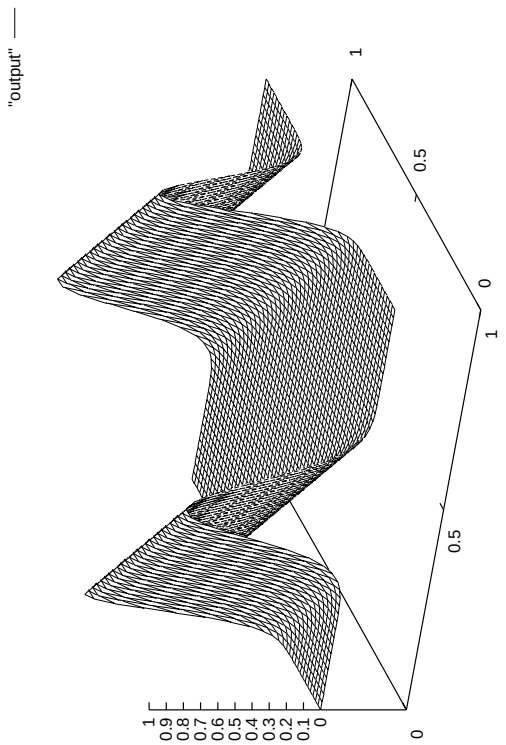
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

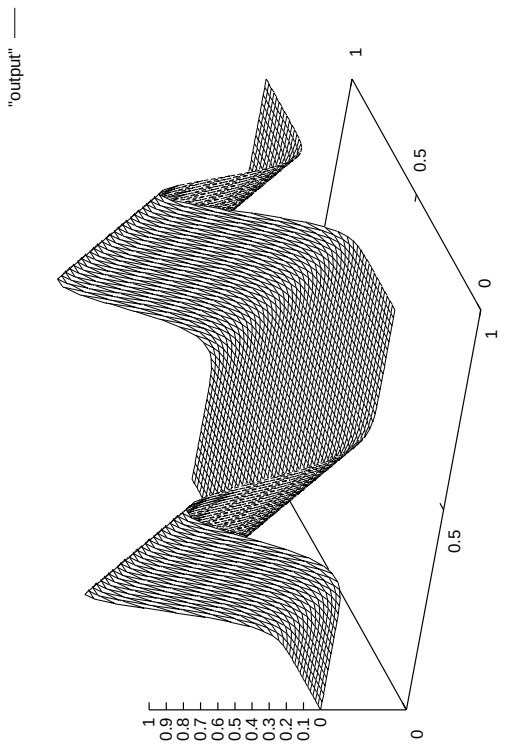
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

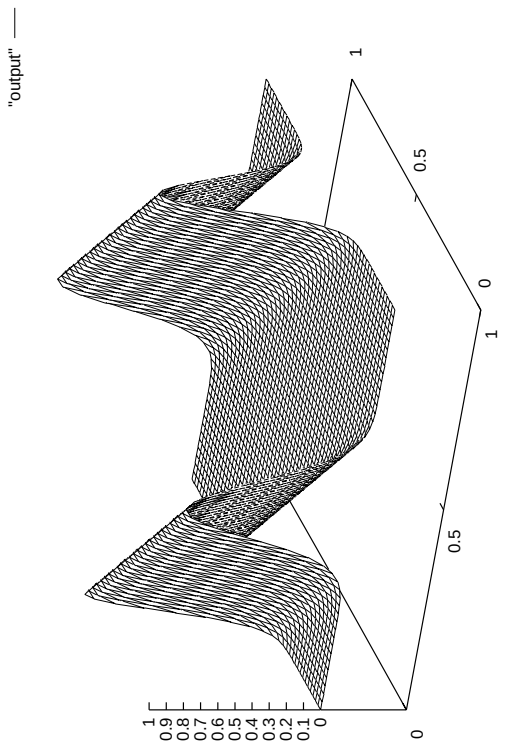
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

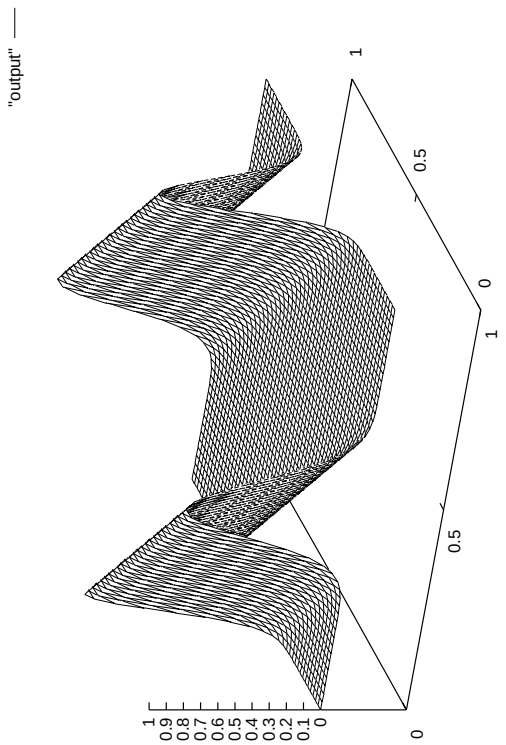
Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

Figure file: output.ps



Source file: `gpwave.f`

```
c============================================================
c
c     gpwave: Generates time-series of profiles of
c     left-moving "wave" (f(t+x) = constant) and outputs to
c     stdio in form suitable for susequent plotting with
c     'gnuplot'.
c
c     For parametric surface plots 'gnuplot', expects three
c     numbers per line:
c
c         x(i), y(j), f(i,j)
c
c     with all data points with the same x(i) on contiguous
c     lines (a group) and with empty lines separating
c     groups.  A quick glance at some sample output from this
c     program should make the arrangement clear.
c
c============================================================
      program     gpwave

      implicit    none

      integer     i4arg

      integer     maxn
      parameter ( maxn = 100 )

      real*8      f
      real*8      x(maxn)
      integer     i,           j,          n,         nx,
     &            nt
      real*8      h,           t,          dt

      n = i4arg(1,-1)
      if( n .lt. 1  .or.  n .gt. maxn ) goto 900
      nx = n
      nt = n

      h = 1.0d0 / (nx - 1)
      x(1) = 0.0d0
      do j = 1 , nx - 1
         x(j+1) = x(j) + h
      end do

      t = 0.0d0
      dt = 1.0d0 / (nt - 1)
      do i = 1 , nt
         do j = 1 , nx
c-------------------------------------------------------------
c           Output the cordinates and function value, three
c           per line, first coordinate (time) constant.
c-------------------------------------------------------------
            write(*,*) t, x(j), f(mod((x(j) + t),1.0d0))
         end do
c-------------------------------------------------------------
c           Empty line separates groups with distinct
c           first coordinate.
c-------------------------------------------------------------
         write(*,*)
         t = t + dt
      end do

      stop

900   continue
         write(0,*) 'usage: gpwave <n>'
      stop

      end

c-------------------------------------------------------------
c     Gaussian function.
c-------------------------------------------------------------
      double precision function f(x)
         implicit      none
         real*8        x
         f = exp(-((x-0.5d0)/0.1d0)**2)
         return
      end
```

Source file: `sgi_output`

```
############################################################
# Building and running 'gpwave' on SGIs
############################################################
einstein% pwd
/usr2/people/phy329/fd/wave

einstein% ls
Makefile   gpin       gpwave.f   vswave.f

############################################################
# Three executables are generated by default (including
# 'gpwave').  'vswave' and 'fvswave' use a different
# graphical interface which we *may* discuss later in the
# course.
############################################################
einstein% make
        f77 -g -n32 -c gpwave.f
        f77 -g -n32 -L/usr/localn32/lib gpwave.o \
            -lp329f -o gpwave
        f77 -g -n32 -c vswave.f
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lvs -o vswave
        f77 -g -n32 -L/usr/localn32/lib vswave.o \
            -lp329f -lfvs -lutilio -o fvswave

############################################################
# 'gpwave' expects a single argument, 'n'.  It then
# generates data which can be plotted as a two-dimensional
# surface (z(x,y)) using 'gnuplot'.
############################################################
einstein% gpwave
 usage: gpwave <n>

############################################################
# Generate data on a 51 x 51 mesh and save to file 'output'.
############################################################
einstein% gpwave 51 > output

einstein% more gpin
############################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
############################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit


############################################################
# Make the plot.
############################################################
einstein% gnuplot < gpin

einstein% ls
Makefile    gpin        gpwave.f    output      vswave*     vswave.o
fvswave*    gpwave*     gpwave.o    output.ps   vswave.f
```

Source file: Makefile

```
#################################################################
#  Note that this 'Makefile' assumes that the following
#  environment variables are set:
#
#     F77
#     F77FLAGS
#     F77CFLAGS
#     F77LFLAGS
#     LIBBLAS
#
#  Put the appropriate 'setenv' commands in your '~/.cshrc'.
#  See 'phy329@einstein:~/.cshrc' for an example.
#################################################################
.IGNORE:

F77_COMPILE  = $(F77) $(F77FLAGS) $(F77CFLAGS)
F77_LOAD     = $(F77) $(F77FLAGS) $(F77LFLAGS)

.f.o:
$(F77_COMPILE) $*.f

EXECUTABLES = gpwave vswave fvswave

all: $(EXECUTABLES)

gpwave: gpwave.o
$(F77_LOAD) gpwave.o -lp329f -o gpwave

vswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lvs -o vswave

fvswave: vswave.o
$(F77_LOAD) vswave.o -lp329f -lfvs -lutilio -o fvswave

clean:
rm *.o
rm $(EXECUTABLES)
```

Source file: gpin

```
#################################################################
# Sample gnuplot commands to read data in file 'output'
# and plot as 'parametric' surface plot with hidden lines
# removed.
#################################################################
set terminal postscript landscape
set output "output.ps"
set parametric
set hidden
splot "output" with lines
quit
```

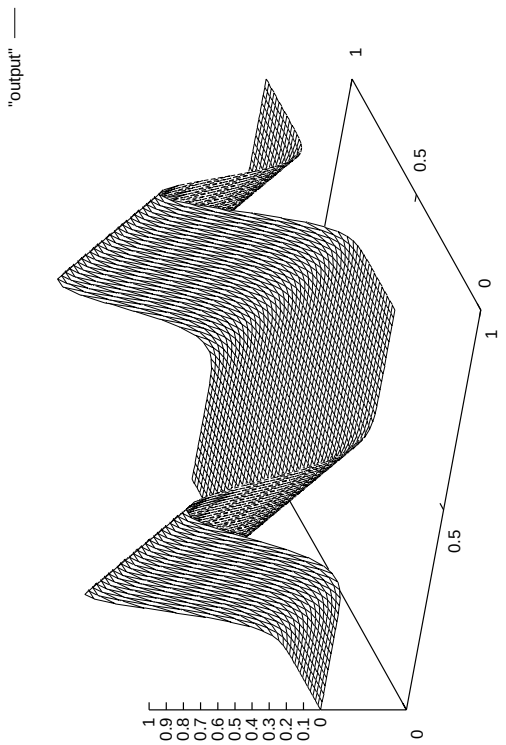



Figure file: output.ps