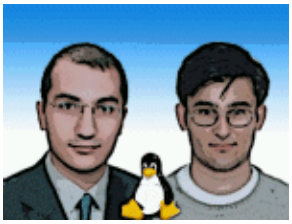
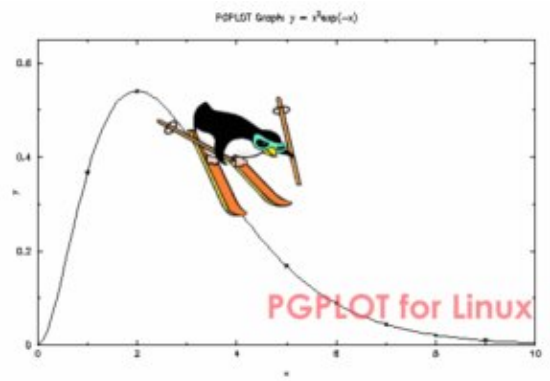
Benutzung von PGPLOT für interaktive Grafiken unter Linux



by Baybora Baran and Seckin Gokaltun
<baybora(at)be.itu.edu.tr
gokaltun(at)itu.edu.tr>

About the author:

Wir sind zwei Forschungsassistenten am Institut für Informatik der ITU. Wir arbeiten mit Ingenieur−Anwendungen an Computern und benutzen dafür Linux ... Seckins Webseite findet sich auf www.be.itu.edu.tr/~seckin



Abstract:

In diesem Artikel lernen Sie, wie man in Fortran geschriebene interaktive Grafikprogramme erstellt. PGPlot ist die Sammlung von Subroutinen, die wir in unserem Fortran−Code verwenden müssen. Wir werden die Installation und einige Anwendungen der PGPLOT−Subroutinen beschreiben. Wir werden zwei Beispiele (einschließlich Fortran−Code) zeigen, die Ihnen Ideen zu möglichen Anwendungen geben werden, die Sie mit PGPlot erstellen können.

Einführung

Was ist PGPLOT?

Die PGPLOT−Bibliothek ist ein aus Fortran oder C aufrufbares, geräteunabhängiges Grafikpaket zum Erstellen einfacher wissenschaftlicher Graphen. Es ist gedacht zum Erzeugen grafischer Bilder in Publikationsqualität mit minimalem Aufwand. Für die meisten Anwendungen kann das Programm geräteunabhängig sein und die Ausgabe wird erst zur Laufzeit auf das passende Gerät umgeleitet.

Die PGPLOT−Bibliothek besteht aus zwei Hauptteilen: einem geräteunabhänigen Teil und einer Menge an geräteabhängigen``Gerätetreiber"−Subroutinen für die Ausgabe auf verschiedene Terminals, Bildanzeigen, Dotmatrix−Drucker, Laserdrucker und Stiftplotter. Die Unterstützung für allgemeine Dateiformate umfasst PostScript und GIF. PGPLOT selbst ist überwiegend in Standard−Fortran−77 geschrieben.

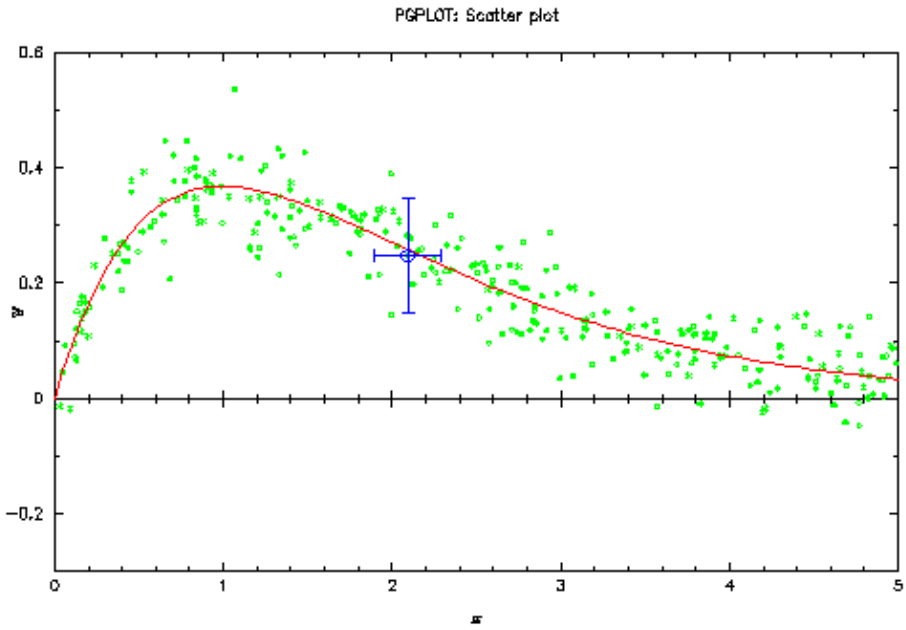
PGPLOT−Subroutinen können direkt aus einem Fortran−77− oder Fortran−90−Programm aufgerufen werden. Eine Bibliothek mit C−Bindungen (cpgplot) und eine Header−Datei (cpgplot.h) werden bereitgestellt, damit PGPLOT direkt aus einem C− oder C++−Programm aufgerufen werden kann; die Bibliothek übernimmt die Konvertierung zwischen C und Fortran. PGPLOT wurde mit UNIX (fast alle Varianten, einschließlich Linux, SunOS, Solaris, HPUX, AIX und Irix) und OpenVMS−Betriebssystemen getestet.

PGPLOT ist keine Public−Domain−Software. Für nichtkommerzielle Nutzung ist es jedoch frei verfügbar. Der Quellcode und die Dokumentation unterliegen dem Copyright des California Institute of Technology mit einigen wenigen nicht−Standard−, systemabhängigen Subroutinen. Zum Abruf der Installationsdatei und Instruktionen klicken Sie einfach hier.

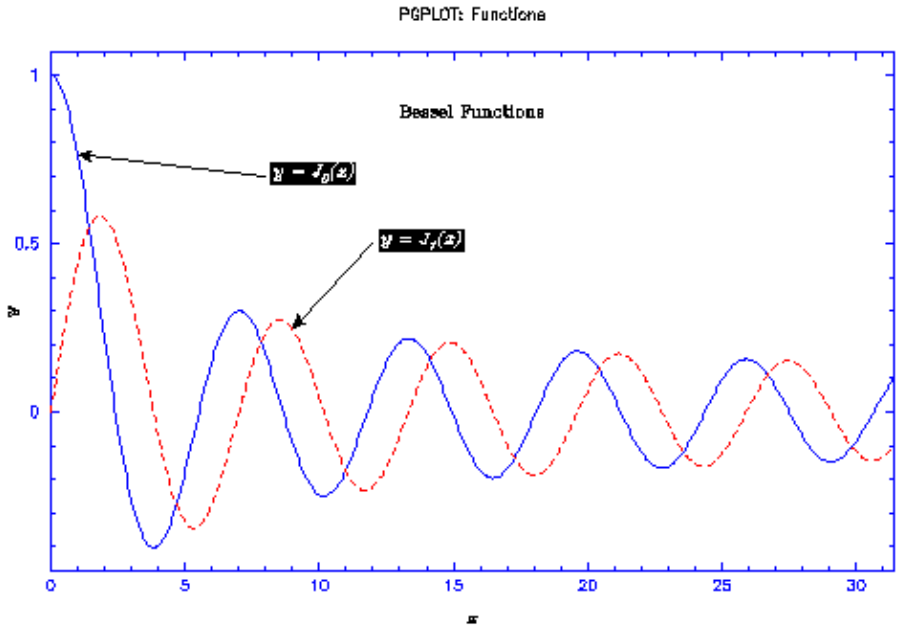
Einige Beispiele

Nachstehend präsentieren wir einige einfache Beispiele für PGPLOT−Anwendungen, um die Fähigkeiten von PGPLOT zu demonstrieren.

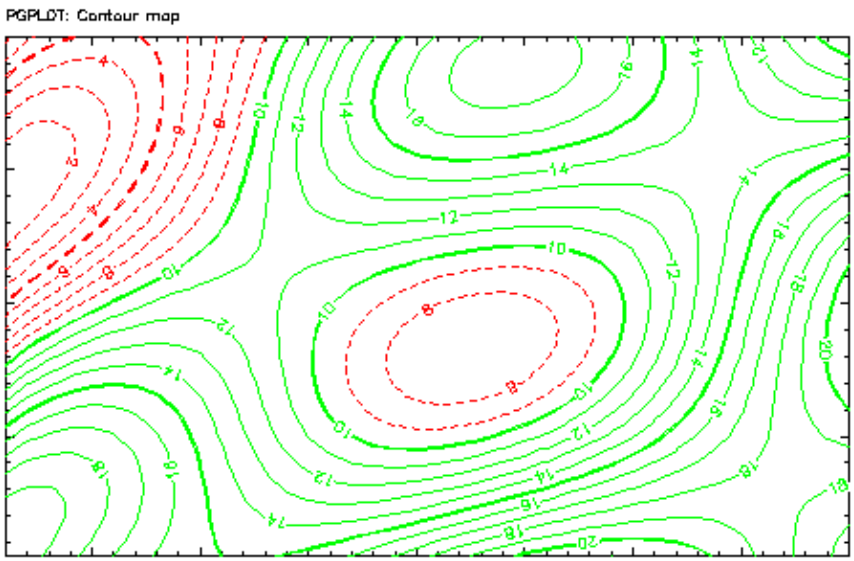
Beispiel 1) Scatter Plots:



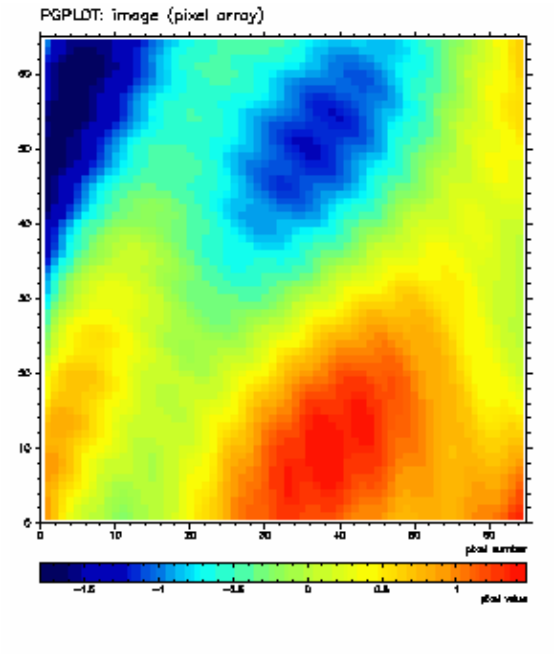
Beispiel 2) Function Plots:



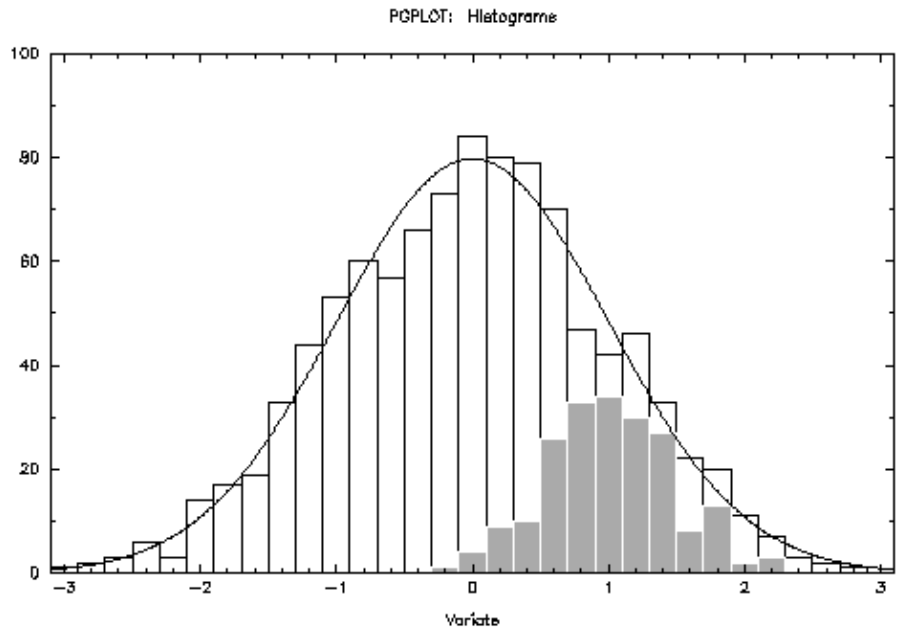
Beispiel 3) Contour Maps:



Beispiel 4) Images:



Beispiel 5) Histograms:



Installation für Unix−Systeme

Hinweis: Die folgenden Instruktionen beziehen sich auf zwei Verzeichnisse, das *Distributions−(Quell−)Verzeichnis*, welches den PGPLOT−Quellcode−Baum enthalten wird, und das *Zielverzeichnis*, in dem die maschinenspezifischen Bibliotheken, Datendateien und Beispielprogramme erstellt werden. Wir empfehlen, dass Sie neue, *leere* Verzeichnisse dafür anlegen. Es sollten nicht die gleichen Verzeichnisse wie in den folgenden Beispielen sein, diese heissen

`/usr/local/src/pgplot` (Distributions−Verzeichnis)
`/usr/local/pgplot` (Ziel−Verzeichnis)

aber Sie können jeden Namen benutzen. Ungewöhnliche (root−)Privilegien sind zur PGPLOT−Installation nicht erforderlich, vorausgesetzt, Sie haben Schreibrechte in den Verzeichnissen. Ein einzelnes Distributions−Verzeichnis kann benutzt werden, um Versionen von PGPLOT für verschiedene Architekturen in verschiedenen Zielverzeichnissen zu installieren.

Die Installation der PGPLOT−Subroutinen unter Linux wird hier kurz erläutert, bei weiteren Fragen können Sie tjp(At)astro.caltech.edu anschreiben.

Kopieren der Distributions−Datei

Kopieren Sie die Distributions−Datei mittels anonymous−ftp von Caltech. Benutzen Sie anonymous ftp (Benutzer: anonymous, Passwort: Ihre@Email.Adresse) zu ftp.astro.caltech.edu. Die Distributionsdatei ist eine UNIX−Tar−Datei, die mit gzip komprimiert wurde. Benutzen Sie die folgenden ftp−Befehle, um die Datei abzurufen:

```
cd pub/pgplot
binary
hash
get pgplot5.2.tar.gz
```

Die Textdateien in diesem Verzeichnis sind auch in der `tar`−Datei enthalten.

Die Distributions−Datei kann auch von der folgenden URL abgerufen werden:
`ftp://ftp.astro.caltech.edu/pub/pgplot/pgplot5.2.tar.gz`.

Dekomprimieren der Datei

Benutzen Sie `gunzip` und `tar` zum Entkomprimieren und Extrahieren des Archives. Damit wird das Verzeichnis pgplot (einschließlich Unterverzeichnissen) im aktuellen Verzeichnis erstellt. Stellen Sie sicher, das Ihr aktuelles Verzeichnis dasjenige ist, in dem Sie den ``PGPLOT−Distributions''−Verzeichnisbaum erstellen wollen.

```
cd /usr/local/src
gunzip −c pgplot5.2.tar.gz | tar xvof −
```

Dieses Beispiel erstellt `/usr/local/src/pgplot` und die Unterverzeichnisse.

Anlegen des Zielverzeichnisses

Erstellen Sie ein beschreibbares Verzeichnis, in dem die PGPLOT−Bibliothek und die dazugehörigen Dateien erstellt werden. Je ein solches Verzeichnis ist erforderlich für jede verschiedene Kombination von Betriebssystem und Compiler (``Zielsystem'), die Sie unterstützen wollen.

```
mkdir /usr/local/pgplot
```

Erstellen Sie die PGPLOT−Bibliothek nicht im Distributionsverzeichnis.

Auswahl der Gerätetreiber

Konfigurieren Sie PGPLOT durch die Auswahl von Gerätetreibern aus der verfügbaren Liste. Kopieren Sie zuerst die Datei `drivers.list` aus dem Distributionsverzeichnis in das Zielverzeichnis und benutzen Sie dann einen Editor, um die Gerätetreiber auszuwählen. Diese Datei enthält eine Zeile für jeden verfügbaren Gerätetreiber: löschen Sie das Ausrufezeichen (!) am Anfang der Zeile, um den Treiber zu selektieren oder stellen Sie sicher, dass ein Ausrufezeichen vorhanden ist, wenn Sie den Treiber nicht benutzen wollen. Viele der Treiber können nur mit bestimmten Betriebssystemen benutzt werden (siehe die Hinweise in `drivers.list`), daher sollten Sie nur **die Treiber selektieren, die Sie auch benutzen wollen**. PGPLOT kann später neu konfiguriert werden, indem die Installation an dieser Stelle neu aufgenommen wird. Die meisten Installationen sollten enthalten: das Null−Gerät (/NULL), PostScript−Drucker (/PS, /VPS, /CPS, und /VCPS), Tektronix−Terminal (/TEK, /XTERM, und evtl. andere Varianten), und, falls das X Window−System auf dem Zielrechner verfügbar ist, auch die X Window−Treiber (/XWINDOW, /XSERVE). Vielleicht möchten Sie auch Treiber für GIF−Dateien (/GIF, /VGIF) oder einige der anderen Drucker aktivieren.

```
cd /usr/local/pgplot
cp /usr/local/src/pgplot/drivers.list .
vi drivers.list    (oder benutzen Sie Ihren Lieblingseditor)
```

makefile−Erstellung

Die PGPLOT−Installationsprozedur für UNIX benutzt ein Skript namens `makemake`, um ein Standard−UNIX−makefile für Ihr Betriebssystem, Ihren Compiler und die ausgewählten PGPLOT−Gerätetreiber zu erstellen. Betriebssstem− und Compilerinformation werden aus einer *Konfigurationsdatei* genommen. Konfigurationsdateien sind für die folgenden Systeme vorhanden. Wenn Ihre Konfiguration keine der aufgeführten ist oder Sie Schwierigkeiten mit der erstellten makefile−Datei haben, finden Sie unten Informationen über die Erstellung Ihrer eigenen Konfigurationsdatei.

Beachten Sie, dass die Konfigurationsdateien für spezielle Compiler auf bestimmten Betriebssystemen gedacht sind. Wenn Sie z. B. Ihr System so eingerichtet haben, dass der Befehl `f77` den GNU g77−Compiler aufruft, können Sie keine Konfigurationsdatei benutzen, die z. B. für einen SPARC f77−Compiler erstellt wurde. In der folgenden Tabelle ist Arg#2 ein Code für das Betriebssystem, und Arg#3 ist ein Code für die Fortran− und C−Compiler. Für weitere Informationen zu den unterstützten Systemen siehe `pgplot/sys_*/aaaread.me`, wobei * für eine der Optionen für Arg#2 steht.

```
Arg#2   Arg#3
------  ------
aix     xlf_cc
alliant fortran_cc
bsd     g77_gcc
convex  fc_cc
cray    cf77_cc
epix2   f77_cc          (Control Data EP/IX 2.x)
freebsd f77_cc
fujitsu uxpm_frt_cc
fujitsu uxpv_frt_cc
hp      fort77_c89
hp      fort77_gcc
irix    f77_cc
linux   absoft_gcc
linux   f77_gcc
linux   g77_elf
linux   g77_gcc
```

```
next    af77_cc
next    f2c_cc
next    g77_cc
next    gf77_cc
osf1    f77_cc
osf1    f77_cc_shared
sol2    f77_cc          (Solaris 2.x, SunOs 5.x)
sol2    f77_gcc
sol2    f90_cc
sol2    g77_gcc
sun4    f77_acc         (SunOS 4.x)
sun4    f77_cc
sun4    f77_gcc
ultrix  f77_cc
```

Wenn Ihr System eines der aufgeführten ist, gehen Sie wie folgt vor: Machen Sie das Zielverzeichnis zu Ihrem aktuellen Verzeichnis, z.B.

```
cd /usr/local/pgplot
```

Führen Sie das Skript `makemake` aus dem Distributionsverzeichnis aus, z. B.

```
/usr/local/src/pgplot/makemake  /usr/local/src/pgplot  linux
```

Das erste an `makemake` übergebene Argument ist der Name des Distributionsverzeichnisses. Beachten Sie, dass, wenn Sie `makemake` starten, Ihr akuelles Verzeichnis das Zielverzeichnis sein sollte, d. h. das Verzeichnis, in dem Sie die kompilierte Bibliothek ablegen wollen

Das zweite Argument ist der Name des Betriebssystems (Arg#2 aus der obigen Tabelle); wenn Sie es weglassen oder einen falschen angeben, zeigt makemake die erlaubten Werte an. Bei einigen Systemen, wo mehr als ein Fortran– oder C–Compiler verfügbar ist, ist ein drittes Argument erforderlich (Arg#3 aus der obigen Tabelle); dieses setzt sich normalerweise aus den Namen der beiden Compiler, durch einen Unterstrich getrennt, zusammen. Wenn Sie es weglassen, zeigt makemake die erlaubten Werte an.

Sobald Sie die gültigen Werte angegeben haben, könnte makemake sich beschweren, dass es die Datei `drivers.list` nicht finden kann. Zurück zu Schritt 4!

Beispiel

baybora@bilgi>../pgplot/makemake ../pgplot linux g77_gcc
For additional information, read file ../pgplot/sys_linux/aaaread.me
Reading configuration file: ../pgplot/sys_linux/g77_gcc.conf
Selecting uncommented drivers from ./drivers.list
Found drivers NUDRIV PSDRIV XWDRIV
Creating make file: makefile
Determining object file dependencies.

Das Skript `makemake` generiert die Datei `makefile` zur weiteren Benutzung, eine Fortran–Datei `grexec.f` , die alle ausgewählten Gerätetreiber enthält und eine Textdatei `rgb.txt`, die Farbdefinitionen zur Benutzung durch die Routine PGSCRN enthält. (Wenn Sie bereits eine Datei `rgb.txt`haben, evtl. schon mit Ihren eigenen angepaßten Definitionen, wird `makemake` diese nicht verändern.) Es kopiert außerdem zwei Fortran–include–Dateien, die während der Kompilierung benötigt werden. Zu diesem Zeitpunkt haben Sie also mindestens die folgenden Dateien im Verzeichnis:

drivers.list
grexec.f
grpckg1.inc
makefile
pgplot.inc
rgb.txt

Sie sollten überprüfen, dass diese Dateien erstellt wurden und Sie sollten auch prüfen, ob die Liste der Treiber, die von `makemake` gefunden wurden, mit der übereinstimmt, die Sie in der Datei `drivers.list` ausgewählt haben. Wenn Ihr UNIX−System keines der oben aufgeführten unterstützten Systeme ist, erstellen Sie Ihre eigene Konfigurationsdatei mit dem Namen `local.conf` im Zielverzeichnis. Am besten kopieren Sie eine der in `pgplot/sys_*/*.conf` bereitgestellten Konfigurationsdateien und editieren diese entsprechend der Kommentare in der Datei. Die `makemake`−Prozedur wird die Datei `local.conf` benutzen, wenn sie im aktuellen Verzeichnis existiert und Sie Arg#3 nicht angeben. Beachten Sie, dass Sie trotzdem noch Arg#2 (Betriebssystem) angeben müssen.

Benutzung von `make' zur Kompilierung des Codes

Nun benutzen Sie den UNIX−`make`−Befehl, um die PGPLOT−Bibliothek entsprechend der Anweisungen in der Datei `makefile` zu kompilieren:

make

Standardmäßig erstellt `make`: eine Objektmodul−Bibliothek, `libpgplot.a`; eine Laufzeit−Bibliothek (falls auf dem gewählten Betriebssystem möglich), die binäre PGPLOT−Schriftendatei `grfont.dat`, die Beispielprogramme `pgdemo*`, und eine Dokumentationsdatei `pgplot.doc`. Zusätzlich, falls in Schritt 4 die Treiber /XWINDOW und/oder /XSERVE ausgewählt wurden, generiert es ein Programm `pgxwin_server`, und wenn der /XDISP−Treiber selektiert wurde, auch ein Programm `pgdisp`. Wenn dieser Schritt erfolgreich verläuft, könnten Sie nun eintippen:

make clean

um nicht benötigte Zwischendateien zu entfernen. Danach haben Sie die folgenden Dateien im aktuellen Verzeichnis:

drivers.list
grexec.f
grfont.dat (binäre Schriftendatei)*
libpgplot.a (PGPLOT−Bibliothek)*
libpgplot.so (Laufzeit−Bibliothek, optional)*
makefile
pgdemo1 ... pgdemo16 (Demonstrations−Programme)
pgdisp (erforderlich für den /XDISP−Treiber)*
pgplot.doc (ASCII−Dokumentations−Datei)
pgxwin_server (erforderlich für den /XWINDOW−Treiber)*
rgb.txt (Datenbank für Farbnamen)*

Wenn Sie XMDRIV oder TKDRIV selektiert haben, werden Sie auch einige der folgenden Dateien vorfinden:

pgmdemo (ausführbares Demo−Programm)
libXmPgplot.a (Objekt−Bibliothek, erforderlich für PGPLOT/Motif−Anwendungen)*
XmPgplot.h (header−Datei, erforderlich für PGPLOT/Motif−Anwendungen)*

libtkpgplot.a (Objekt−Bibliothek, erforderlich für PGPLOT/Tk−Anwendungen)*
pgtkdemo (ausführbares Demo−Programm)
pgtkdemo.tcl (vom Demo−Programm benutztes Skript)
tkpgplot.h (header−Datei, erforderlich für PGPLOT/Tk−Anwendungen)*

Wenn Sie die kompilierte Version von PGPLOT in ein anderes Verzeichnis kopieren möchten, müssen Sie mindestens die mit einem Stern (*) markierten Dateien kopieren. Die Dokumentationsdatei enthält die Beschreibung der PGPLOT−Subroutinen, die auch im Handbuch verfügbar sind.

Starten Sie die Beispielprogramme

Starten Sie die Beispielprogramme auf den von Ihnen gewählten Geräten und stellen Sie sicher, dass sie zufriedenstellend laufen. Bevor Sie irgendein PGPLOT−Programm starten, müssen Sie sicherstellen, dass die Umgebungsvariable `PGPLOT_DIR` korrekt definiert ist. Dies ist der Name des Verzeichnisses, in dem PGPLOT nach den Dateien `grfont.dat` und `rgb.txt` sucht (soweit nicht die Umgebungsvariablen `PGPLOT_FONT` und `PGPLOT_RGB` definiert sind, die dieses Standardverhalten abändern), und, falls erforderlich, auch nach dem X−window Server−Programm `pgxwin_server`:

UNIX csh or tcsh: setenv PGPLOT_DIR /usr/local/pgplot/
UNIX sh or bash: PGPLOT_DIR="/usr/local/pgplot/"; export PGPLOT_DIR

Es ist auch angenehm, aber nicht unbedingt erforderlich, ein Default−PGPLOT−Gerät mit der Umgebungsvariable `PGPLOT_DEV` zu setzen, z. B.

UNIX csh or tcsh: setenv PGPLOT_DEV /xwindow

Wenn Sie eine UNIX−Laufzeit−Bibliothek (z. B. unter Solaris 2.x) benutzen, könnte es erforderlich sein, das PGPLOT−Verzeichnis in den Suchpfad des Laders aufzunehmen, der in der Umgebungsvariablen `LD_LIBRARY_PATH` gesetzt wird. Um ein Programm zu starten, geben Sie dessen Namen ein (mit dem Verzeichnispfad, falls das aktuelle Verzeichnis nicht in Ihrem Pfad enthalten ist):

./pgdemo1

Alle Beispielprogramme fragen nach einem Gerätenamen und −Typ. Geben Sie ein Fragezeichen `?` ein, um eine Liste der verfügbaren Gerätetypen zu sehen und zu verifizieren, dass PGPLOT richtig konfiguriert wurde. Was Sie prüfen sollten: dass das PGPLOT−Programm korrekt die Schriftendatei liest und Hoch−/Tiefstellung und spezielle Zeichen korrekt anzeigt (pgdemo2); ob das PGPLOT−Programm die Farbdatenbank korrekt lesen kann (pgdemo10); bei interaktiven Geräten, ob der Cursor korrekt arbeitet (pgdemo5, pgdemo6).

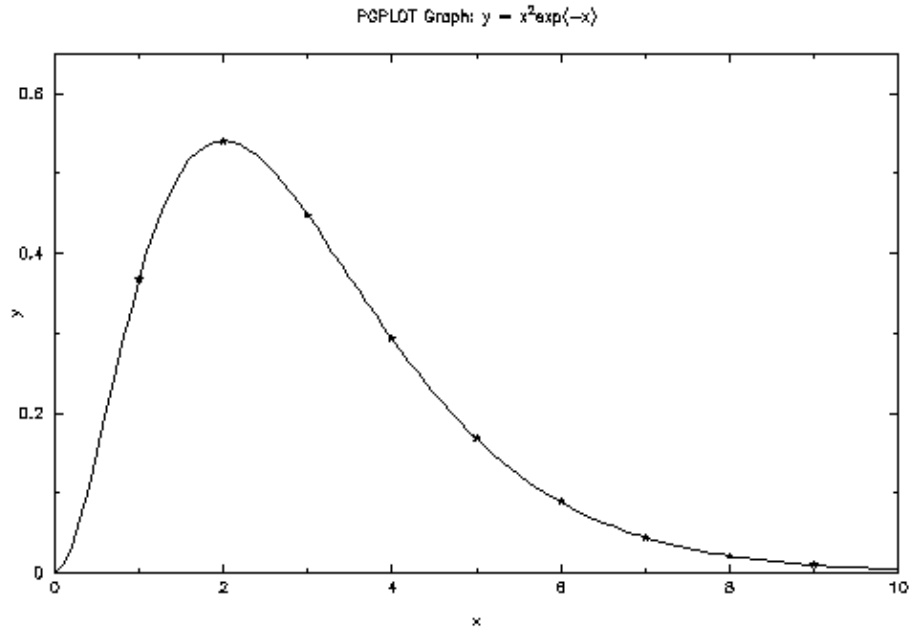
Wie kompilieren Sie Ihren Code?

Nachdem Sie die PGPLOT−Subroutinen−Bibliothek in Ihrem System installiert haben, können Sie die PGPLOT−Subroutinen in Ihrem Fortran−Code wie im folgenden Beispiel benutzen:

```
      PROGRAM EX1
      INTEGER PGOPEN, I
      REAL XS(9), YS(9), XR(101), YR(101)

C Compute numbers to be plotted.

      DO 10 I=1,101
          XR(I) = 0.1*(I-1)
```

```
          YR(I) = XR(I)**2*EXP(-XR(I))
 10   CONTINUE
      DO 20 I=1,9
          XS(I) = I
          YS(I) = XS(I)**2*EXP(-XS(I))
 20   CONTINUE

C Open graphics device.

      IF (PGOPEN('?') .LT. 1) STOP

C Define coordinate range of graph (0 < x < 10, 0 < y < 0.65),
C and draw axes.

      CALL PGENV(0., 10., 0., 0.65,  0,  0)

C Label the axes (note use of \u and \d for raising exponent).

      CALL PGLAB('x', 'y', 'PGPLOT Graph: y = x\u2\dexp(-x)')

C Plot the line graph.

      CALL PGLINE(101, XR, YR)

C Plot symbols at selected points.

      CALL PGPT(9, XS, YS, 18)

C Close the graphics device.

      CALL PGCLOS

      END
```

Dieser Code zeichnet den folgenden Graphen:



Um den obigen Code erfolgreich auszuführen, müssen Sie die PGPLOT−Bibliothek und die X11−Bibliotheken mit Ihrem Code linken. Das folgende Skript erreicht das:

g77 your_code_name.f −L/X11directory/ −lX11 −L/PGPLOTdirectory/ −lpgplot

Sie müssen die folgenden Dateien in das Verzeichnis einschließen, wenn Sie Ihren Code ausführen:

```
grfont.dat
rgb.txt
pgwin server
```

Kopieren Sie einfach diese Dateien aus dem pgplot−Verzeichnis in das Verzeichnis, aus dem Sie Ihren Code starten.

Anwendung 1: Durch 3 Punkte laufender Kreis

In dieser Anwendung war es unser Ziel, durch Klicken mit der Maus drei Punkte auf dem Bildschirm zu markieren und dann Fortran−Code den Kreis zeichnen zu lassen, der durch diese drei Punkte markiert ist. Dieses Problem war eine Aufgabe in unserer Klasse über "Computational Geometry", die von Dr.Serdar Celebi (mscelebi(at)itu.edu.tr) geleitet wurde.

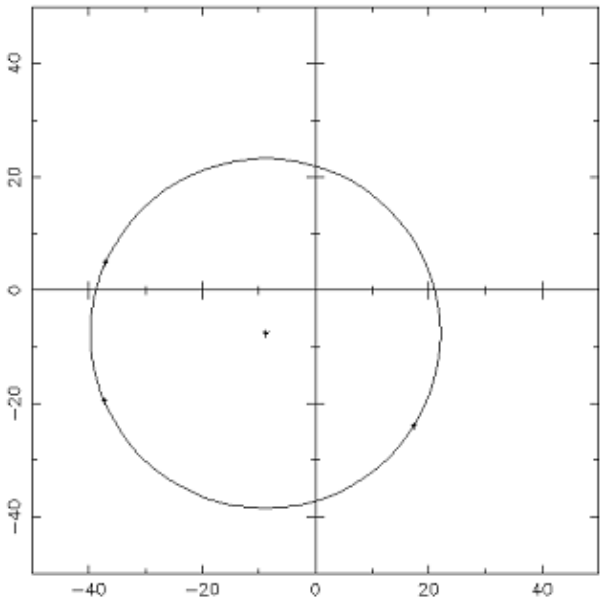
Die folgenden Routinen definieren den Hintergrund und die Einstellung für den Bereich, in dem der Graph gezeichnet wird. Die Benutzung dieser Subroutinen ist ausführlich im PGPLOT−Handbuch erklärt (s. Link zur PGPLOT−Webseite am Ende).

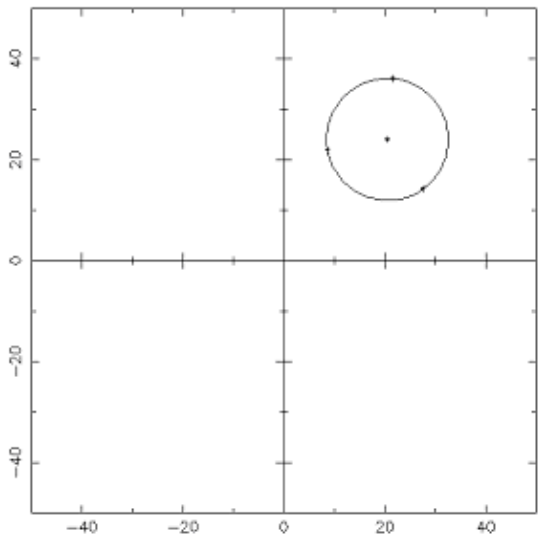
```
..
...
call PGSCR(0,1.0,1.0,1.0) !set color representation
call PGENV(−50.0,50.0,−50.0,50.0,1,1) !set window and viewport and draw labeled frame
call PGSCI(1) !set color index
call PGSFS(2) !set fill−area style
...
..
```

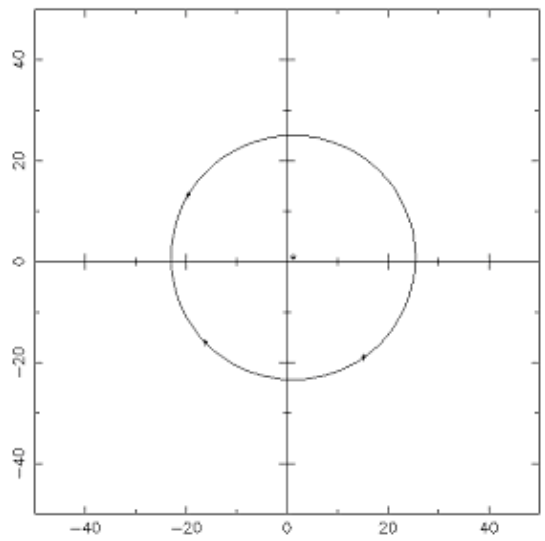
Wir benutzen die folgende Routine, PGPT1, um einen Zeiger an dem Punkt zu zeichnen, den wir mit der Maus markieren wollen.

```
..
...
WRITE (*,*) 'Cursor mode:', MODE
             GOTO 10
          END IF
            CALL PGPT1(X, Y, 3)  !draw one graph marker
        ic=ic+1
        xp(i)=x
        yp(i)=y
```

Nachdem wir 3 verschiedene Punkte auf dem Bildschirm markiert haben, berechnet der Fortran−Code den Mittelpunkt und zeichnet dann den Kreis, der durch diese 3 Punkte verläuft.








```
..
...
c-----find the radius-----------------
```

```
 r=(xcenter-xp(1))**2+(ycenter-yp(1))**2
 r=r**0.5
c-----draw the circle-------------------
 call PGCIRC(xcenter,ycenter,r) !draws a circle
 goto 1
...
..
```

Wir zeichnen den Kreis mit der oben gezeigten Routine "PGCIRC". Sie können den vorgezeichneten Kreis löschen und die Punkte erneut zuweisen, um einen anderen Kreis im gleichen Zeichenbereich zu zeichnen. Der vollständige Code ist in den Referenzen enthalten (siehe Ref. circle.f).

Anwendung 2: Zeichnung von "porcupine"–Linien auf einer Näherungskurve

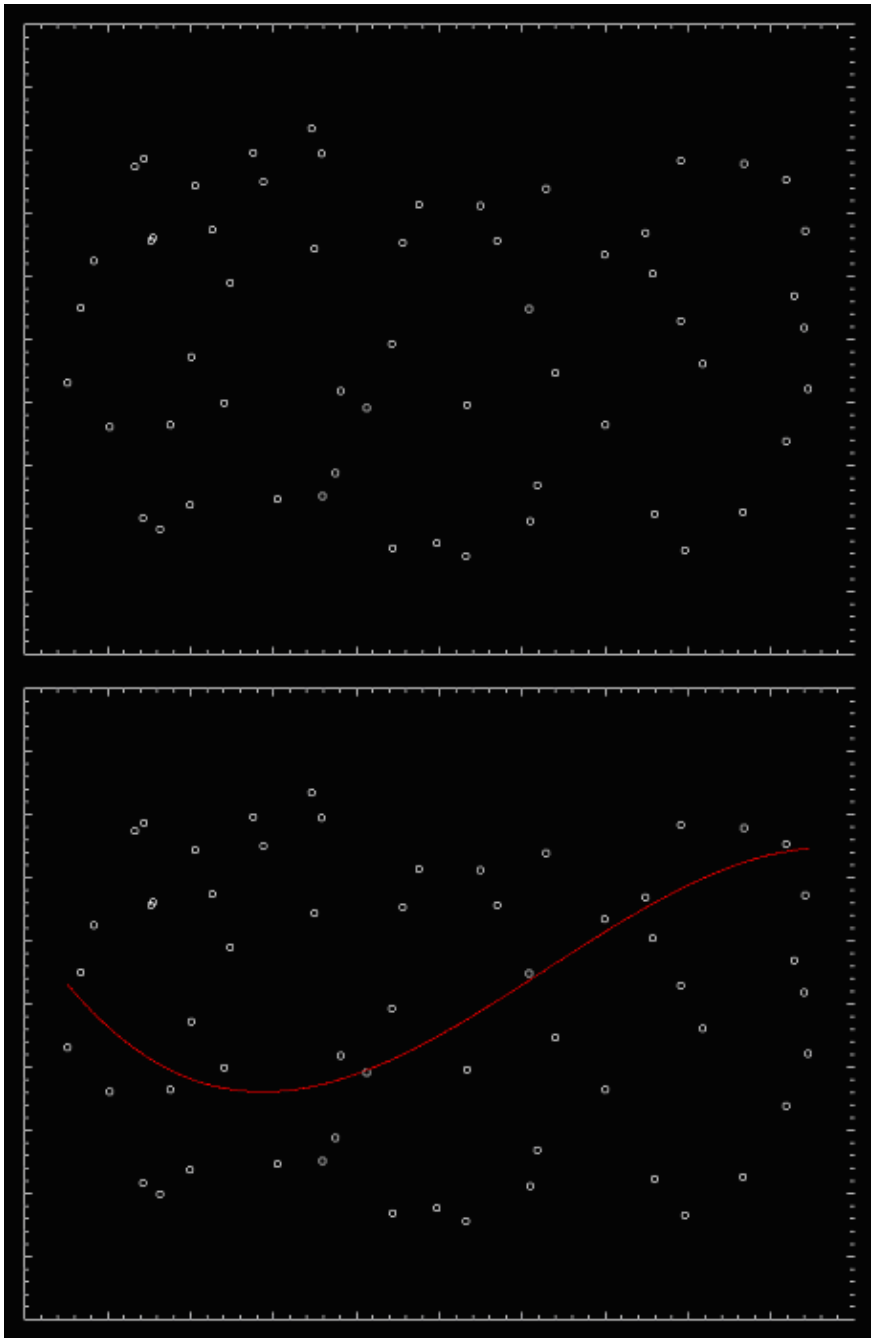
In dieser Anwendung war es unser Ziel, eine Näherungskurve durch die verschiedenen diskreten Punkte zu zeichnen, die mit der Maus auf dem Bildschirm markiert wurden, und dann die "porcupine"–Linien auf dieser Kurve zu zeichnen.

Porcupines sind die Linien, die benutzt werden, um unerwünschte Wendepunkte, flache Teile einer Kurve und Diskontinuitäten in Krümmungen zu entdecken. Deshalb ist dies ein wichtiges Konzept in der "Computational Geometry".

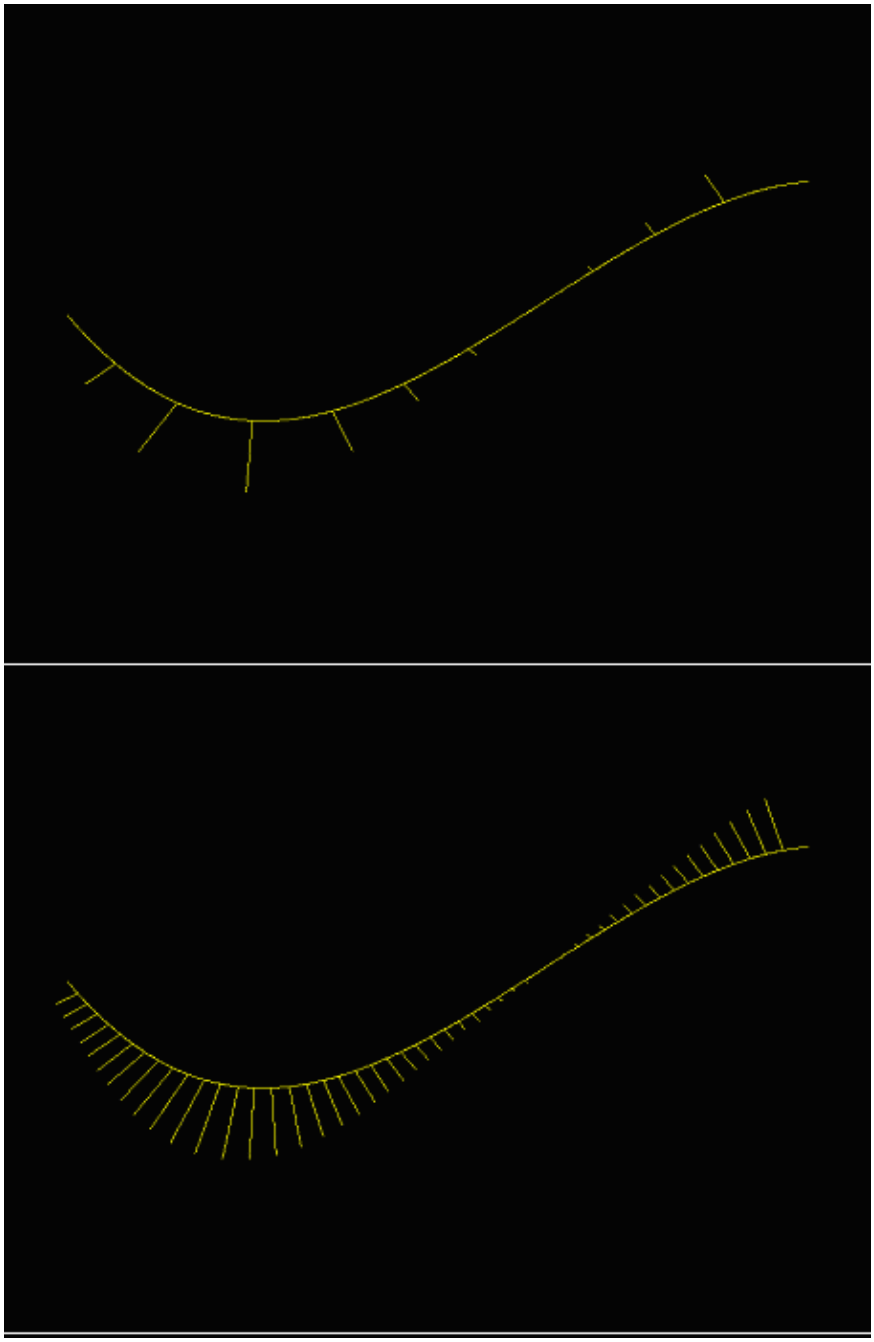
Wir entwickelten Code, der die kleinste quadratische Approximation benutzt, um eine Kurve der gewünschten Ordnung (1–4) durch eine Reihe von Punkten zu zeichnen, die vom Benutzer zufällig über eine grafische Schnittstelle mittels der Maus generiert werden können. Er hat die Fähigkeit, "porcupine"–Linien (deren Frequenz durch den Benutzer angepasst werden kann) entlang der Kurve zu zeichnen; dies ist ein Indikator für die Krümmung der Kurve.

Wir überspringen die Details des von uns geschriebenen Fortran–Codes und präsentieren hier die grafischen Ergebnisse (Sie finden den Code am Ende des Artikels).

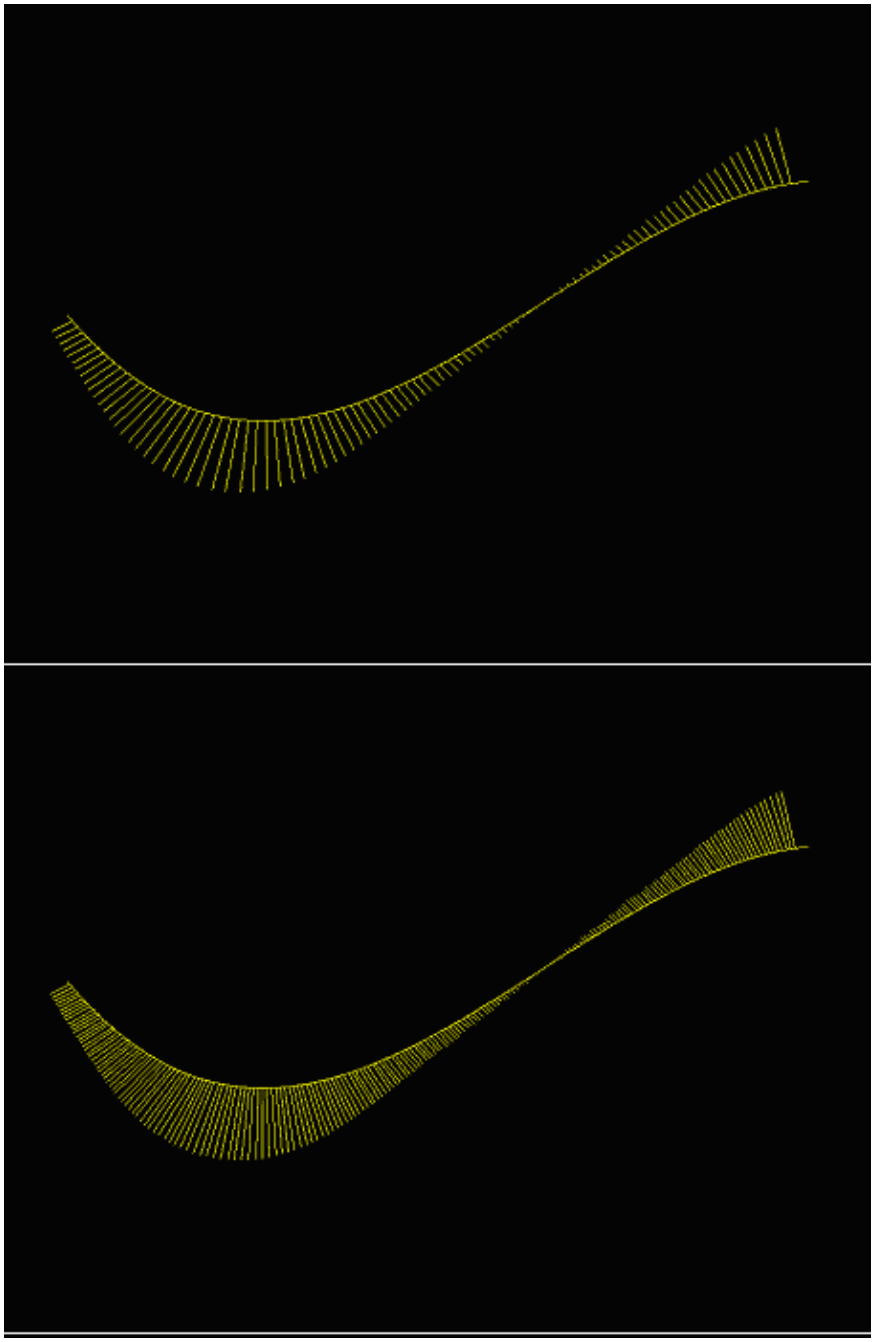
Wir geben zuerst die diskreten Punkte auf dem Bildschirm mit der Maus ein und der Fortran–Code nimmt die Koordinaten dieser Punkte und speichert sie in einem Feld. Dann approximiert er die durch diese Punkte verlaufende Kurve in dem vom Benutzer angezeigten Bereich.(1–4)



Dann werden die "Porcupine"−Linien entlang der Kurve gezeichnet. Die Anzahl der "Porcupine"−Linien kann vom Benutzer verändert werden, und ohne den Hintergrund oder die Punkte zu verändern, kann die neue Menge an "Porcupine"−Linien entlang der gleichen Kurve gezeichnet werden.



Die Häufigkeit der "Porcupine"−Linien kann, wie unten zu sehen ist, ebenfalls erhöht werden. Außerdem kann das Programm eine neue Kurve mit der gleichen Datenmenge zeichnen.



Zusammenfassung

In diesem Text wollten wir zeigen, wie ein interaktives Zeichenprogramm, PGPLOT, unter Linux installiert werden kann und wie weit der Anwendungsbereich dieses Programmes sein kann. Mit dieser Liste von Subroutinen sind Sie völlig unabhängig und es spart wirklich Zeit, wenn Sie bei jedem Lauf Ihres Programms häufig die grafische Ausgabe sehen müssen. Durch die Einbettung dieser Routinen in Ihren Code wird die Nachbearbeitung sehr schnell und robust.

Referenzen

- Die offizielle PGPLOT−Webseite: PGPLOT Webseite
- Der Quellcode für Anwendung 1):circle.f
- Der Quellcode für Anwendung 2):porcup.f
- Ein technischer Bericht über Anwendung 2):601_2.pdf
- Webseite der Klasse "Computational Geometry" Dr.Serdar CELEBI

Translation information:
en −−> −− : Baybora Baran and Seckin Gokaltun
<baybora(at)be.itu.edu.tr gokaltun(at)itu.edu.tr>
en −−> de: Hermann−Josef Beckers
<beckerst/at/lst−oneline.de>

2005−01−11, generated by lfparser_pdf version 2.51